Burroughs

ROBERT L. OLSEN
TERRITORY MANAGER
7555 BEACH BLVD., SUITE 116
JACKSONVILLE, FLORIDA 32216
721-1660

B 90 / B 900/CP 9500
B 1800/B 1900
Computer Management Systems (CMS)
System Software

OPERATION GUIDE

PRICED ITEM

Printed in U.S. America November 1980 2015228

Burroughs

ROBERT L. OLSEN
TERRITORY MANAGER
7555 BEACH BLVD., SUITE 116
JACKSONVILLE, FLORIDA 32216
721-1660

B 90/B 900/CP 9500 B1800/B1900 Computer Management Systems (CMS) System Software

OPERATION GUIDE

PRICED ITEM

F.E. Dist. Code E
Printed in U.S. America

Cover Revised
October 1981

For Library Binder 702
Form 2015228-003

This edition includes the information released under the following:

PCN 2015228-001 (November 30, 1980)
PCN 2015228-002 (January 1981)
PCN 2015228-003 (October 1981)

Correspondence regarding this publication should be forwarded using the Remarks form at the back of the manual, or may be addressed directly to TIO Europe Documentation, Burroughs Machines Limited, Cumbernauld, Glasgow, Scotland G68 OBN

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1
INTRODUCTION

THE CMS CONCEPT

CMS (Computer Management System) software is a powerful set of software items designed to operate on a number of different hardware products.

To the user of an individual hardware product running CMS software, there is a well-defined operator interface and set of programming languages. The importance of CMS is that the same user may use a different hardware product running CMS software, and with the same languages. This portability eliminates major operator retraining between different CMS products. It also allows freedom of interchange of programs between hardware products, limited only by availability of hardware features. For example, a program may be developed and compiled on one system, and run on another. Also, because the compilers are also programs, there is portability of compilers between hardware systems as well. Data files are similarly transferable from one system to another. This portability is achieved by building on the "soft machine" concept. Refer to figure 1-1.

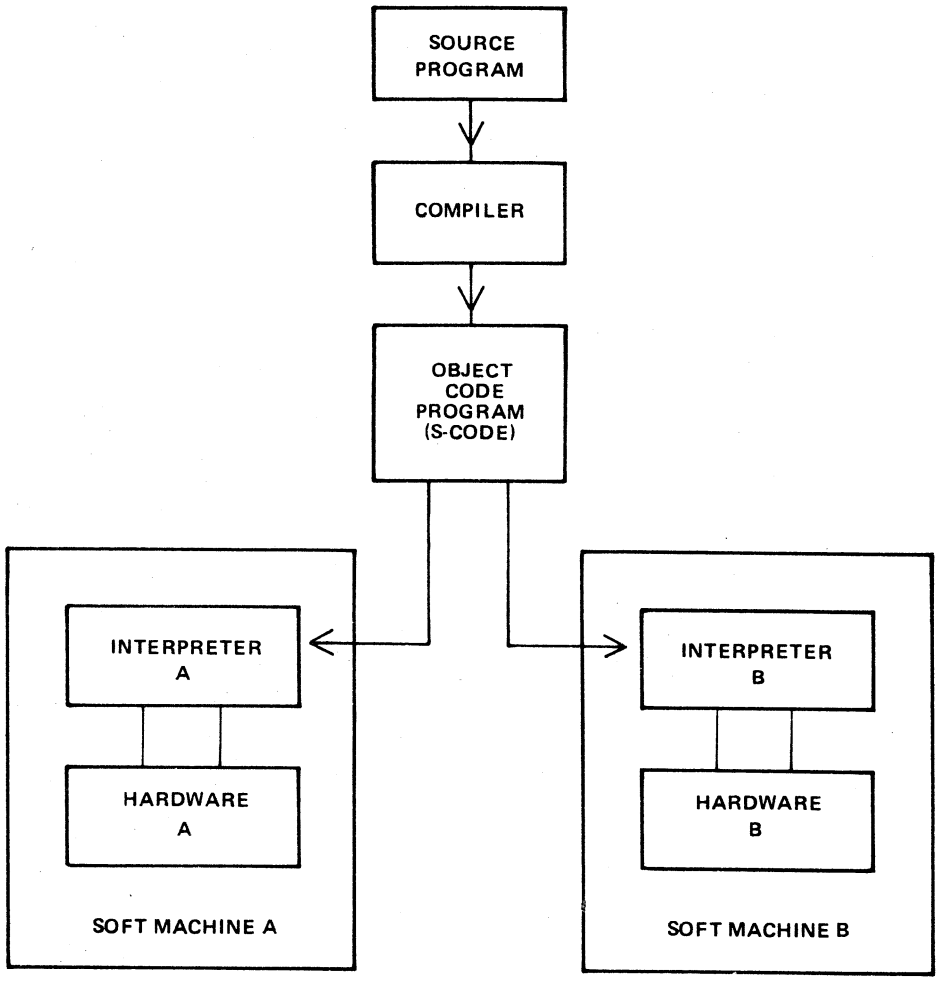


Figure 1-1. CMS Portability

The programmer writes a program in a high-level language. The CMS programming languages are:

COBOL

RPG (including RPGII)

MPL (CMS Message Processing Language)

NDL (Network Definition Language).

This program is writen in 'source code'. This is then input to one of the CMS compilers which converts it to "object code" or "S-code". This is the executable program. The "S-code" is similar in design to the "machine code" of earlier generations of computer.

In earlier generations of computer this ' 'S-code" would be executed by hard-wired instructions. With the advent of fast micro-processor computers, however, it is possible to build a set of micro-instructions which interprets each "S-code" and executes it. The set of micro-instructions is therefore called an "interpreter". The combination of interpreter and micro-processor hardware is sometimes termed a "soft machine".

Now as the "S-code" is independent of any particular hardware, it is possible (and has been achieved in CMS) to build several soft machines which will execute a "object program" in a similar manner. Hence the CMS object programs are portable across the different CMS machines.

These machines include:

B 90
B 1800
B 1900

There are different CMS interpreters on each system. For example, on the B 90 the interpreters are:

BILINTERPX
COBOLINTX
NDL.INTERPX

BILINTERPX is used to execute programs written in MPL and in BIL (an implementation language used for compiler-writing which is so similar to MPL that they share the same S-code format). COBOLINTX is used to execute programs written in COBOL and RPG (these two languages share the same S-code format). NDL.INTERPX is used to interpret data communication controller programs written in NDL.

Certain common features needed in all programs (such as the handling of peripheral devices) have been collected together into a Master Control Program (MCP). The MCP is a micro-code program and is therefore specifically written for each hardware product. Thus there is a B 90 MCP, a B 1800 MCP and a B 1900 MCP. The MCP also controls the operator interface (which is standard across the CMS range) and mainhains overall control of the system, providing complete resource management including multi-programming, I/O device handling and memory management.

CMS software also provides a number of utility programs. As these are written in MPL, they also are portable across the CMS range, limited only by hardware feature availability.

To cover the complete features of each CMS product line, certain aspects of the software are written for a specific product. These additional features include important operational characteristics, and are described in sections 8 through 10. Sections 2 through 7 of this manual cover items which are applicable to any CMS product.

SOFTWARE RELEASE LEVELS

Each item on a CMS software release is identified by a three-part number, as follows:

```
X. XX. XX
|  |  |
|  |  |____ patch number
|  |_______ level number
|__________ mark number
```

The mark and level numbers constitute the release number. For example, the COBOL compiler 3.01.08 is the COBOL compiler included in the 3.01 release of system software, with patch number 08.

Software items from different releases should not be used together. For example, an interpreter from release 3.01 should not be used with an MCP from release 3.00.

This book describes system software relative to the 3.03 release.

Software Patches

Within a particular release, patches to individual items may be issued. For example, an MCP identified by 3.02.12 contains certain improvements over an MCP identified by 3.02.11. A patch always increases the patch number. It is always advisable to use the highest patch versions within any one release. All system software items within a given release (mark and level numbers) may be used together, regardless of the patch number, unless explicitly stated otherwise at the time of release of the item.

Certain items may be patched by the user. The details are machine-dependent and are described in the relevent section (8 through 10).

SOFTWARE SUPPORT

Throughout this book, suggestions are made for corrective action where possible, following a particular output message or symptom of failure. Sometimes the phrase "request technical assistance" has been used. This should be interpreted as a recommendation to contact your immediately higher support level if you are not sure of what to do or do not feel justified in attempting further action without competent advice.

All problems with the system should be recorded. This is for two purposes: to report the problem; and to avoid similar problems in the future. The report should contain the date and time and list the systems. As a minimum it is recommended that the SPO hard-copy printout or SPO log is kept for future reference.

TO THE READER

This book is written as reference material. It is a guide to be consulted during operation of any CMS machine.

This book explains how to start and to stop the system software. As this is normally hardware-dependent, the relevant section (8 through 10) should be consulted.

Once the system software has started (that is, the system is under MCP control), the operator may interface with the MCP via the SPO (Supervisory Printout) device in order to execute programs. The type of device may vary with the hardware product, but input and output messages are standardized.

Section 2 of this book explains some general terms which should be understood in order to make full use of the CMS features. It explains how to cause programs to be executed. This section also explains how to read the diagrams used throughout the book to describe the format of input messages and other details.

Details of input messages are given, in alphabetical order, in sections 3 and 4. The items in section 4 are utility programs which are executed in the same manner as other programs. The items in section 3 are embedded features in the MCP. Refer to section 2 for a fuller explanation.

Sections 5 and 6 describe the sort/merge feature and the compilation feature respectively, and will be of special interest to programmers. Section 5 includes a functional description of the sort/merge feature.

Section 7 lists the messages which may be output to the SPO by the system software during execution of the system. As each message is identified on the SPO by a number, reference to this book can be made by this number.

For other items such as hardware and system software failures, refer to the particular hardware section (8 through 10) for details.

SECTION 2
BASIC CMS OPERATION

INTRODUCTION

All CMS operation has two basic principles: it is disk-based; and operator communication is with the MCP by a SPO device. Other peripherals may be present, depending on the configuration. This section introduces some basic principles which should be understood by all CMS operators. The material in this section is common to all CMS products. Other details that are machine-dependent are given in the relevant section.

PERIPHERALS

Each peripheral is referenced by a three-character abbreviation, where the first two characters give the type of peripheral and the third character refers to the particular peripheral by the letter A, B, and so on. For example, LP is the abbreviation for a line printer, so the first line printer is referred to as LPA, and the second is LPB.

The peripheral types are listed below:

AC – console with any output device
AM – any multi-function card unit
AP – any (serial or line) printer
AR – any card reader
AT – any magnetic tape
CP – any card punch
CT – cassette tape
DC – data communications controller
DF – fixed disk
DI – industry-compatible mini-disk (ICMD)
DK – disk cartridge (any type of speed)
DM – Burroughs super mini disk (BSMD and BSMDII)
DP – disk pack
KB – Keyboard
LP – line printer
MT – magnetic tape (reel)
M8 – 80-column multi-function card unit
M9 – 96-column multi-function card unit
PC – console with serial printer
P8 – 80-column card punch
P9 – 96-column card punch
R8 – 80-column card reader
R9 – 96-column card reader
RS – Reader Sorter
RT – Real Time Clock

SC – console with SELF-SCAN ® device

SD – Screen Display

SP – serial printer (on console)

SS – **SELF–SCAN** ® device

If the configuration contains more than one device of the same type, the designation (A, B, and so on) depends on the location of the peripheral controller in the hardware. If there is only one dual-drive cartridge controller, the upper drive is DKA and the lower drive is DKB. If there is only one dual-drive Burroughs super-mini-disk controller (for example, on a small B 80 with in-built mini disk), the upper drive is DMA and the lower drive is DMB.

The three-character references are used in all operator communication with the MCP (refer to section 3).

SYSTEM AND USER DISKS

The MCP resides on a disk unit. At warmstart time (when the system is started up and the MCP begins to function) the MCP notes the disk containing the executing MCP code. This is called the "system disk".

During operation there is only one system disk. Other disks may contain a copy of the MCP code, but only the disk from which the MCP is running is the system disk.

All other disks on the system during machine operation are called "user disks".

There is one restriction on the portability of system disks between different CMS products. A system disk may not be taken to a different CMS product and used there as a system disk. It may, however, be used on the second system as a user disk. It may also be used on the first system as a user disk. User disks may always be interchanged between different systems.

DISK FORMAT

A disk consists of one or more platters, one or both surfaces of which may be used to record data. The recording area of disks is divided into the following physical items:

Track:

An area of one surface of a disk which is at the same distance from the center of the disk. The entire track can be accessed without moving the position of the read/write head.

Sector:

The basic unit of disk address, size 180 bytes on all Burroughs disks, and 128 bytes on ICMD. A physical read or write uses a complete sector. There are several additional bytes in each sector, used only by the hardware and not accessible to user programs. The sector is also called a "segment".

Cylinder:

If there is more than one surface, each track at the same distance from the center makes a cylinder. The entire cylinder may be accessed without altering the position of the read/write heads.

Figure 2-1 illustrates these terms.

Disk Initialization

Each disk must be initialized before use on a CMS machine. Initialization creates correct sector addresses throughout the disk recording surface, then writes certain data in the low-address part of the disk. The first sector is numbered sector zero, and the first track is numbered track zero. A disk with a bad track cannot be initialized. The method of initializing the disk is machine-dependent (refer to the appropriate section).

Sector zero contains the disk label. This includes the name of the disk, or "disk-id". Every disk has a disk-name. This disk-name can be from one to seven characters, using the set A to Z, 0 to 9, and the dot (".") and hyphen ("–").

SELF-SCAN ® is a registered trademark of Burroughs Corporation.

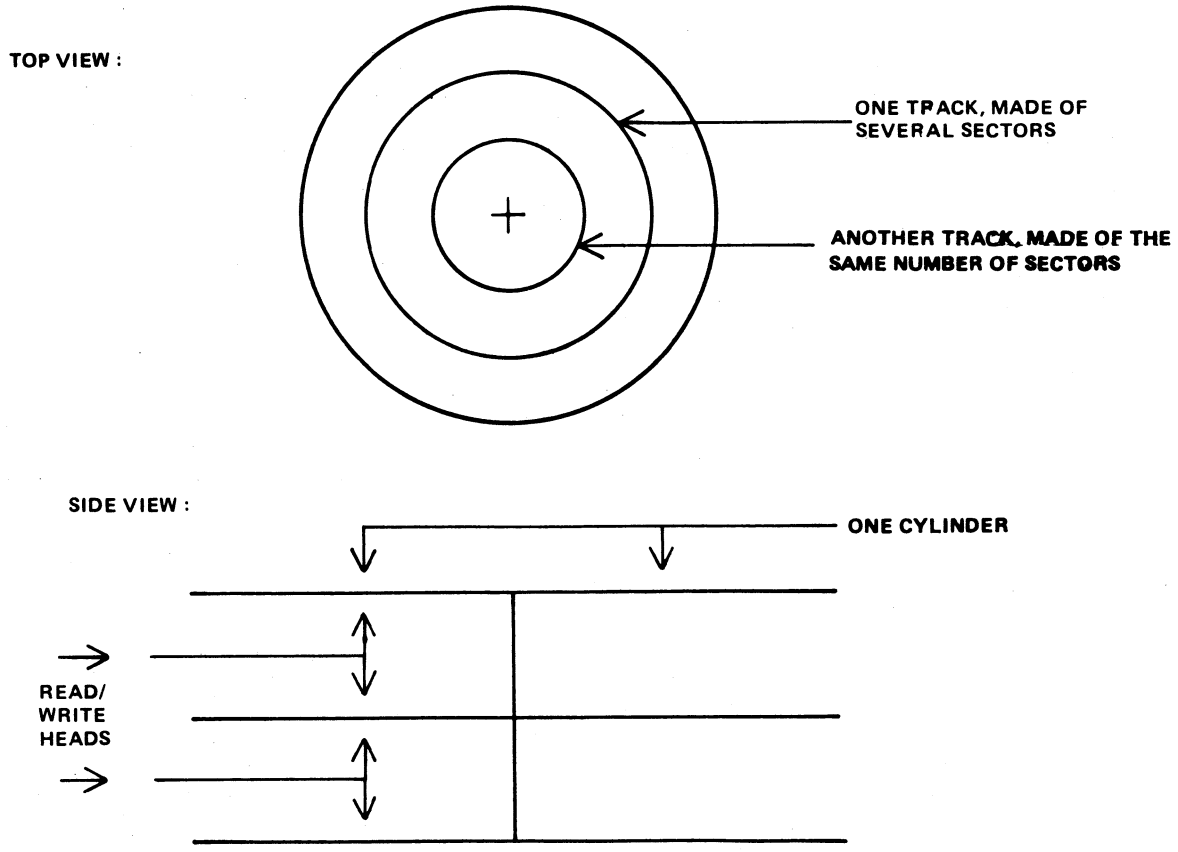


Figure 2-1. Physical Disk Structure

Disk Files

Information is stored on a disk in a "disk file". There may be many files on one disk. Each file is referenced by a "file name". A file name can be from one to twelve characters, using the set A to Z, 0 to 9, and the dot and hyphen. Each disk contains a directory of the files on that disk. This directory is accessed by utilities such as KA and PD (see section 4).

Information can be of different types: normal data, accessed by programs; special data, accessed by the MCP; and programs themselves. The MCP is itself a program, and so are other "system files" such as the interpreters. System files have special restrictions in that a control is placed on their removal (see RM section 4).

Disk File Names

On any system, every disk file (whether data or a program) is accessed by a two part reference, as follows:

disk-name/file-name

For example, the disk file M101A/REP200 is a file with a file-name REP200 to be found on the disk with a disk-name M101A.

It is not necessary to give the name of the system disk when referring to files residing on the system disk. Alternatively, a disk-name of 0000000 by convention refers to the system disk. For example, the disk file REP200 or 0000000/REP200 is a file with a file-name REP200 to be found on the system disk.

It is not allowed to have two disks of the same disk-name in use at the same time. It is not allowed to have two files of the same file-name on the same disk. However, it is quite permissible for two different disks to contain a file with the same file-name. For example, the files M100A/REP200 and M101A/REP200 refer to two different disk files (although one may be a copy or update of the other).

Disk File Group Names

In many utilities (see section 4) it is convenient to refer to groups of files, depending on common starting characters of their file-names.

All files on a disk may be referenced by the equals symbol ("="). For example, the reference M101A/= refers to all files on the disk with disk-name M101A.

All files beginning with, say, the characters REP may be referenced by REP=. For example, the reference M101A/REP= refers to all files on disk M101A with file-names of REP200, REPA, REP678P, and so on.

In general, a group-name consists of an equals symbol ("=") optionally preceded by up to ten symbols which are the first part of the file-names of each of the files in the group.

Example:

Consider a disk M101A containing files with file-names:

PR200,REP100,REP200,REP250,RQ510,CRCOPY

Then the following group-names refer to the files indicated:

M101A/=

PR200, REP100, REP200, REP250, RQ510, CRCOPY

M101A/REP=

REP100, REP200, REP250

M101A/R=

REP100, REP200, REP250, RQ510

Disk Directory

The disk directory is a table on every CMS-initialized disk which enables the MCP to locate any disk file by name. Full details of the directory layout are given in the CMS MCP manual.

The directory is a fixed size determined at disk initialization time, based on the maximum number of files to be placed on the disk. An attempt to create more files than there are entries in the directory will give an appropriate MCP run-time error message.

The directory consists of three parts:

the name-list

the disk file headers for each file

the available table

The relationship between these parts are given in figure 2-2. The name-list is a list, by file-name, of each file existing on that disk. A search through this name-list will reveal if a file is present or not: if present, the name-list entry points to the disk file header for the file. This is a table giving the location of each part of actual data in the file (the file may be divided into up to sixteen separate physical areas on the disk). In the figure only one area is indicated. The available table is a list of the disk areas not in use by a file. When a new disk file is created, an available space is found from this table and an entry made in the name-list, then the space is used to write the file information. When a disk file is removed its entry is deleted from the name-list and the areas specified in the disk file header are entered in the available table.

If there is insufficient space on a disk to allocate new disk file areas, a "NO USER DISK" message is given by the MCP. The operator may remove a file (see RM) to make more space available. The KA utility (see section 4) and KX function provide information on the available space on a disk. The Stand-Alone utility LS will also give the available space on a disk whose files have been listed by an LS/=.

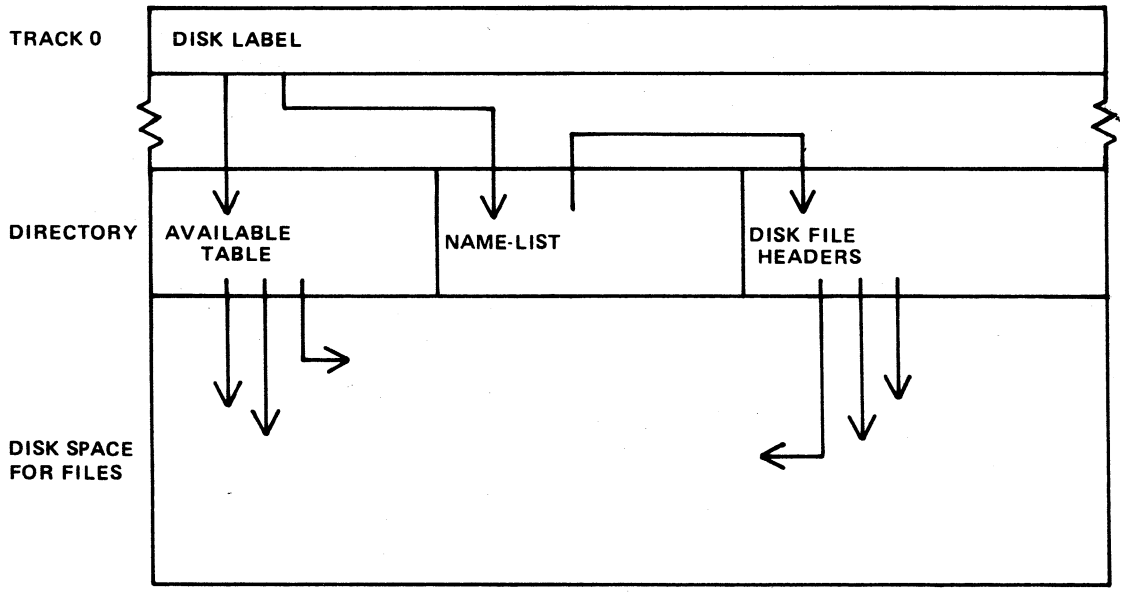


Figure 2-2. Disk Directory Structure

As a simplification, it may be stated that when a disk is initialized the directory is rebuilt with no entries, indicating that the entire disk space is available apart from the directory itself. In fact, any bad areas on the disk are marked in the directory so that they cannot be allocated to files (see also the XD utility); also, there is a special entry called "SYSMEM" which enables certain programs such as PD and RM (which access the directory) to operate successfully.

Indexed Files

Indexed files are in fact a pair of files, the"'key file" and the "data file". They may reside on the same or separate disks. Each file in the pair has a separate entry in the disk directory of the disk on which it resides. A special table at the beginning of the key file (the"'key file parameter block") gives, among other information, the disk-name and file-name of the associated data file. See figure 2-3 for a diagram of the relationships between the two files.

The purpose of indexed files is to simplify access to data in the data file by using a set of keys (such as account number) in each record of the data file. These keys are placed in the key file. A key file may be created by the SORT utility and intrinsic (see section 5, where examples are given).

Special consideration must be given to copying indexed files, due to the link between the key file and data file. This is especially true when copying from one disk to another. Details are given in each relevent section (see COPY utility, section 4; also the machine-dependent copy facilities).

Dual Pack Files

As mentioned before, a disk file may be divided into up to sixteen separate areas. If these areas are located on two separate disks the file is known as a"dual pack file". Such files may be created by the AD intrinsic in response to a"NO USER DISK" message (see section 3).

There is an entry in the directories of both disks for a dual pack file, together with the disk-name of the other disk. Each disk directory has a copy of the disk file header for this file, but the table of locations for each file area also indicates if the area is located on"this" disk or the"other" disk. This is shown diagrammatically for a file with four areas in figure 2-4. In most applications it is necessary for both disks of a dual-pack file to be on-line at the same time.

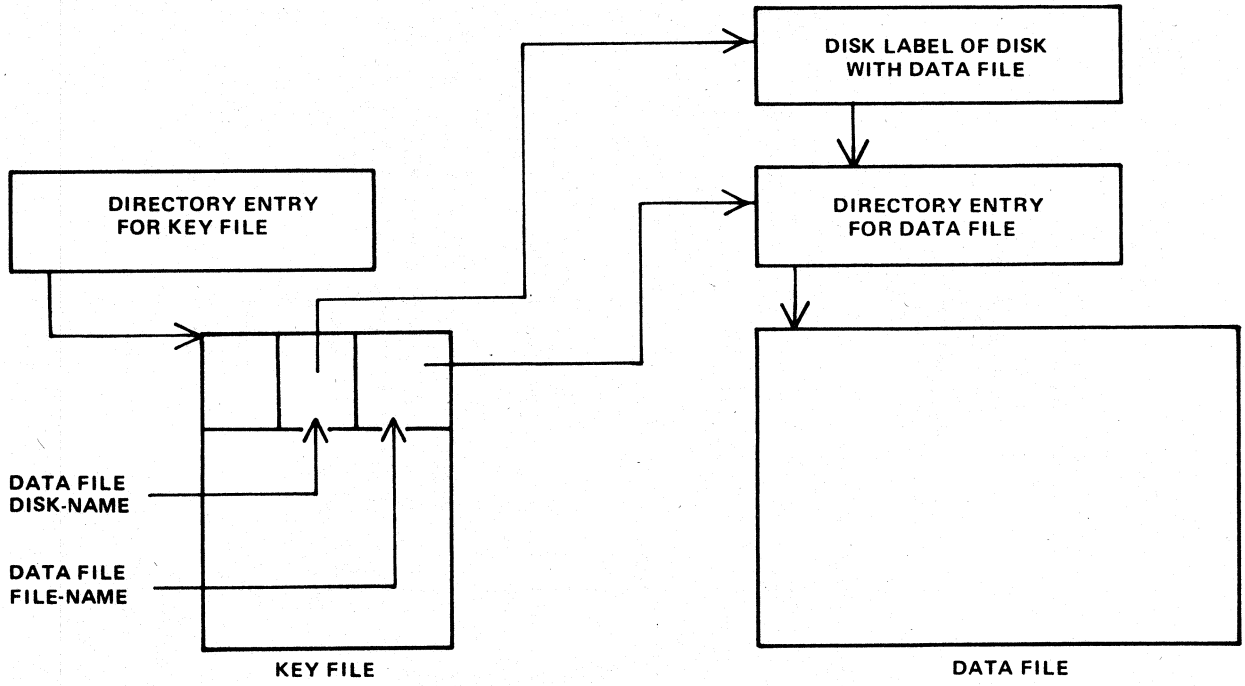


Figure 2-3. Indexed Files

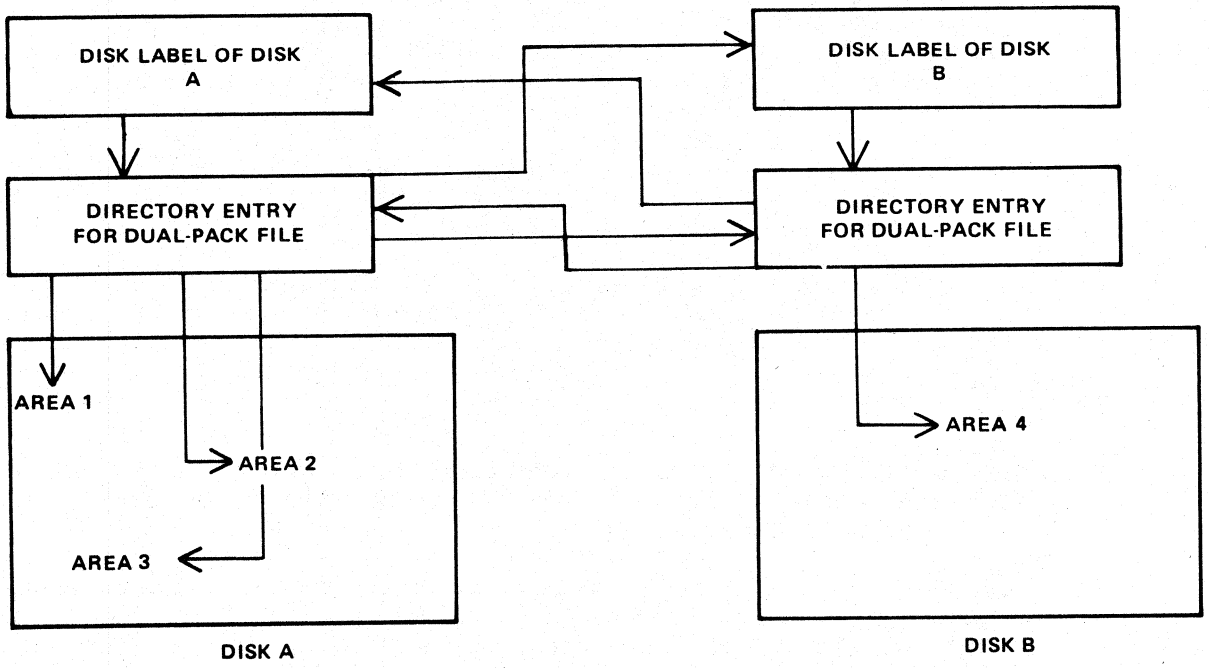


Figure 2-4. Dual-Pack Files

MAGNETIC TAPE FILE NAMES

Note: this includes tape cassette.

A tape may be used to store data either on one file (a "single-file tape") or as a "multifile tape". Each file is separated by a tape mark. Additionally, each file normally has a beginning and an ending label. A multifile tape has also a special beginning ("volume") label.

On loading a tape, the MCP reads the first label to determine the tape name. Tape file names are in two parts:

multifile-name/file-name

For a single-file tape, the multifile-name will be "0000000". The format of the multifile-name is the same as for the disk-name of a disk file.

The COPY utility (section 4) produces a single-file tape when copying to tape. The LD utility (section 4) always produces multifile tapes called "library tapes". Library tapes are referenced by the multifile-name: there is a standard convention for labelling all the files on a library tape. For full details of tape formats, refer to the CMS MCP manual.

Tapes (multi-file or single-file) may be unlabelled. Such tapes must always be accessed via the AD intrinsic (section 3) because there is no label that the MCP can recognize when the tape is loaded. Tapes containing labels that are non-standard are also treated as multifile unlabelled tapes.

PRINTER FILES

There are two types of printer: a wide line printer and a console printer, depending on available hardware. The console printer is also known as a "serial printer". These hardware devices are also referred to as "files" and are given file-names of up to seven characters. When the file is opened and closed, an identifying print line is given to indicate the name of the file. This file-name is also used in MCP messages. Refer to the CMS MCP manual for full details.

It is possible to designate a file type of "any printer". Such a file will be written to a wide line printer if this peripheral is available. If not available, this file will be written to the console printer if available. If there is no console printer either, the MCP will display a "NO FILE" or "DEVICE REQUIRED" message.

OTHER PERIPHERALS

All peripherals are treated as files for input, output or a combination of input/output, depending on the hardware type. The use of any peripheral device is governed by the file-name of up to seven characters, which will appear in any related MCP messages. Refer to the OL intrinsic (section 3) for other details.

PROGRAMS

An executable program is information stored on disk as a disk file. It is referenced in the same way as any data file: that is, through the disk-name and file-name (or just the file-name if the program resides on the system disk). The rules for the program name are the same as for any disk file name.

A "utility" is a program provided for general use by all CMS operators, for house-keeping and other general purposes. For example, the LD utility enables operators to load and dump disk files from disk to magnetic tape for backup purposes.

Executing Programs

In order to execute a program, part or all of the information in the disk file must be brought into memory and placed under control of the MCP. This is called "program load', and takes a certain interval of time.

Programs may be loaded and executed by merely providing the name of program file to the MCP. If so desired, the keyword "EX" may be place before the program name. For example, suppose one wishes to execute a program that resides on a disk PR200A in a file called DCS. Either the input

EX PR200A/DCS

or just

PR200A/DCS

will cause the program to be loaded and executed.

Depending on the system, a BOJ (beginning-of-job) message may be displayed by the MCP after the program has been loaded, and a EOJ (end-of-job) message may be displayed by the MCP at the end of the program. The display of these messages may be turned on or off for individual programs by the MODIFY utility (see section 4).

Failures may occur when attempting to load a program. For example, the requested program may not be on disk. A list of load failure messages is given in section 7.

Many programs enable the operator to enter further information after the program name. This is known as an "initiating message" and the contents are entirely dependent on the program. Nearly all the utilities in section 4 allow further information, the format of which is given in the description of each utility program. For example, the input

COPY REP202 TO RPTAPE

consists of the command to load and execute the program called "COPY" (found on the system disk in this example), followed by the information "REP202 TO RPTAPE" which is passed to the program. There are two types of error which can be made: either there is a load failure (because, for example, the COPY program is not on the system disk), when the MCP would issue an appropriate message; or the following information is an incorrect format for the program, when the program itself would issue a message. In the former case, the MCP message is described in section 7. In the latter case, the output message is described under each utility.

Note that if the utility resides on, say, the disk PR2, the input message would be

PR2/COPY REP202 TO RPTAPE

or

EX PR2/COPY REP202 TO RPTAPE

In section 4 this additional information is omitted in the interest of clarity. It is, however, common for utilities to reside on a disk other than the system disk, in which case the disk-name must be provided.

It is also possible for programs to be automatically executed by another program. In this case, the first program is said to "zip" the second program. No operator input is used in this case, but the BOJ message may be displayed for the zipped program.

INTRINSICS

There is an important type of operator input that does not involve a command to execute programs or utilities. These messages are calls on "intrinsics" which are part of the MCP. Those intrinsics which are common to all CMS machines are described in section 3. Other intrinsics are given in the relevent machine-dependent section.

Because an intrinsic is part of the MCP, there is no separate program corresponding to the name of the intrinsic. Therefore the keyword "EX" is not allowed in a call on an intrinsic, neither can a user disk-name be specified. There is no program load time because the MCP is already executing. For example, the input

RY DMA

is a request to the MCP to ready (RY) the disk peripheral designated by DMA. This input message to the MCP must not be preceded by the keyword "EX".

MIX NUMBERS

As a program is loaded, the MCP assigns it a number from its table of executing tasks. This is the "mix-number" and is used in any messages output by MCP relating to this task. The mix-number is also used in all messages input by the operator for this task. Some input messages also require the corresponding program name as well as the mix-number. The MX intrinsic (see section 3) may be used to determine the current mix of tasks.

The allocation of mix-numbers is dependent on the CMS product. Refer to the corresponding section for more details.

OUTPUT MESSAGES

As mentioned earlier, messages may be output on the SPO either by the MCP and other system software or by the program. It is important to distinguish between the two types of output messages in order to look up the message in the appropriate place.

Messages output by the MCP are of two kinds: short responses to intrinsics, and longer descriptions of any event to be brought to the attention of the operator. The short descriptions are self-explanatory: for example, the input message

```
OL LPA
```

(an intrinsic to inquire of the status of line printer LPA) may result in the response

```
LPA READY
```

Similarly, the short message

```
LPA NOT READY
```

will be displayed if LPA is stopped by the operator or through any fault. The longer descriptions are always referenced by an "event number" enclosed in brackets. The format of these messages is given in section 7, and operators should be generally able to recognize that such a message has been output by the MCP.

For example, the message

```
10/LIST <17> WAITING UNLAB LISTPRT AP NO FILE
```

indicates an MCP message with event number 17, and reference should be made to section 7 for information on possible causes and suggested actions to take.

Messages with event numbers may also be output by other parts of the system software such as interpreters and the sort-intrinsic, although the overall format is similar. After recognizing the event number, reference should be made to section 7 (or section 5 for sort-related messages).

Messages output by all other programs are known as "displays" and may be preceded by the keyword "DISP". Note, however, that utility programs may display messages without this preceding keyword.

All messages output by the utility programs described in this manual are listed under the respective utility. For example, messages displayed by COPY utility are listed under the COPY utility. Messages may additionally be displayed by the MCP for events related to the execution of the COPY, DCR and SCR utilities (for example, if the COPY utility needs space on a particular disk, a "NO USER DISK" message will be output) but these MCP messages will always be distinguished by the event number.

Messages displayed by other programs are not discussed in this manual. Reference must be made to the appropriate manual or operating instructions for that program.

Figure 2-5 illustrates a sample SPO list giving a mixture of messages described in this section. Note in this example that the utility programs LIST and LR do not give rise to BOJ and EOJ or DISP messages. The user program PROGA shows all three messages. These messages may be turned on for utilities by using the MODIFY utility (section 4).

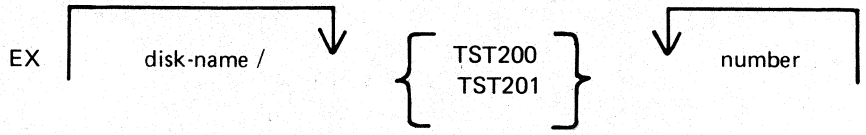
```
input commanc to run LIST --> LIST COLLETTE
MCP output message event 10->   01/LIST <10> WAITING COLLETTE OK NO FILE
input commanc to run PROGA -> PROGA
MCP message for PROGA BOJ -->   02/PROGA BOJ PR IS A
input commanc to run LR ----> LR =
next line is PROGA cisplay ->   02/PROGA DISP:
actual display information ->     PROGRAM A VERSION 3.01.05
input recuest OL intrinsic -> OL LPA
MCP response to OL message ->   LPA LRPRINT IN USE BY 03/LR
input request MX intrinsic -> MX
MCP response to MX message ->   01/LIST SUSPENDED WAITING ON NO FILE
                           ->   ...CONDITION
                           ->   02/PROGA A EXECUTING
                           ->   03/LR B EXECUTING
MCP message for PROGA EOJ --> 02/PROGA EOJ
input recuest ST intrinsic -> ST 3
MCP response to ST message ->   03/LR STOPPED
```

Figure 2-5. Sample SPO List

FORMAT DIAGRAMS

Most of the descriptions of input messages in this book are given as simple format diagrams with corresponding descriptive text and examples. An example will illustrate how to read such format diagrams.

Example:



In this format, items in lower-case ("disk-name" and "number" in this example) are to be replaced by actual values (such as "PR2" and "27"). Other items are included in the input message as they are found. Spaces are required whenever necessary to avoid ambiguity. In the example, it is not strictly necessary to separate the disk-name and the slash ("/") with a space because the slash cannot be part of the disk-name according to the rules for disk-names. Extra spaces may however, be added for legibility. If an arrow in the left-to-right direction is encountered, the items under the arrow may be omitted. Curly brackets are used to denote alternatives. The alternatives are placed in a list underneath each other. (Each alternative item may be more complex than the example quoted: it may contain optional parts and further alternatives). If an arrow in the right-to-left direction is encountered, one may return to the point underneath the arrow and continue building up a valid input message In the example quoted, after adding a valid number (say "27") one may return to add a second number (say "52"). In fact, the format diagram does not specify how many times one may continue to do this, but details are given in the text.

Here are several valid input messages which can be generated from the example. (Note that a disk-name can consist of up to seven characters, see earlier):

EX TST200 57

EX TST201 259

EX PR2/TST200 36

EX PR2/TST200 2 52 574 361

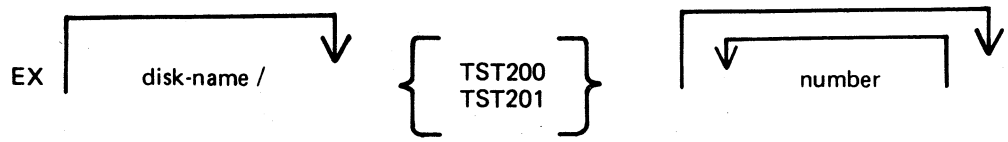
EX M101A/TST201 1 2

Here are several invalid input messages according to the example:

EX PR2/TST200
EX PR2 TST200 36
EX TST202 36
TST201 259
EX PR2/M101A/TST201 1 2

Here is a slightly more complicated example, which makes the number or list of numbers optional:

Example:



The input messages

EX PR2/TST200
EX PR2/TST200 56
EX PR2/TST200 27 56

are now all valid.

These simple format diagrams are easy to understand in conjunction with descriptive text and examples, but cannot be used if the format becomes too complex. In the latter case a rigorous notation known as "railroad diagrams" is employed (see below). In some case in the text of this book, the format has been deliberately simplified for the sake of clarity, with further details given in the text. More complex features have been described by railroad diagrams (see, for example, the COPY and LIST utilities in section 4). Appendix B gives complete railroad diagrams as a handy reference for those who need the exact definition of any input message.

RAILROAD DIAGRAMS

The equivalent railroad diagram to the first format diagram is given in figure 2-6.

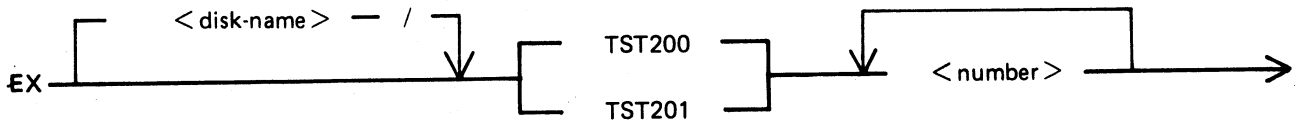


Figure 2-6. Railroad Diagram Sample 1

To form valid input, follow the railroad "track" from left to right or in the direction of the arrows. A junction in the track indicates that alternative paths may be followed. Items enclosed in angled brackets "<" and ">") must be replaced with actual values, as before. Each item not enclosed in angled brackets is included as it is found. Spaces are added where necessary, as in format diagrams.

The equivalent railroad diagram to the second format diagram is given in figure 2-7.

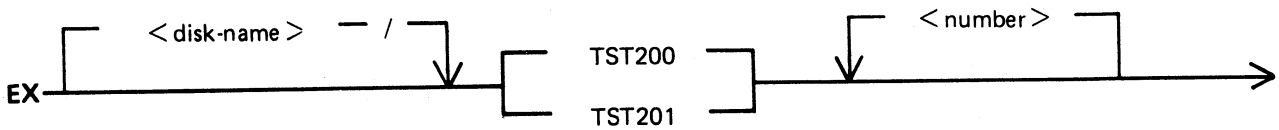


Figure 2-7. Railroad Diagram Sample 2

There are two other features available in railroad diagrams to make possible the exact specification of any input message. These are illustrated in figure 2-8. Firstly, the maximum number of times around a loop may be controlled by including the number

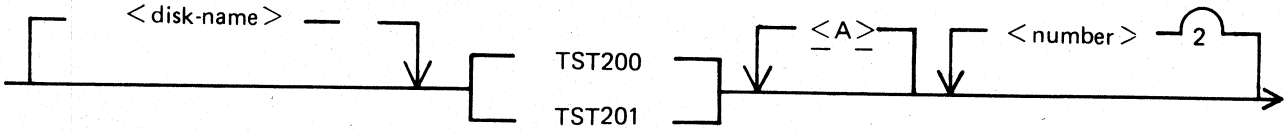


Figure 2-8. Railroad Diagram Sample 3

in the track of the loop. In the example, it is possible to omit the <number>, or to include either one or two values of <number>. Secondly, if angled brackets are to be included as part of the message, these must be underlined. In the example, there is an optional part of the message which consists of the three characters "<A>". The following messages would then be valid:

EX PR2/TST200
EX PR2/TST200 27
EX PR2/TST201 27 56
EX PR2/TST201 <A>
EX PR2/TST200 <A> 56

but the following would be invalid:

EX PR2/TST200 27 56 243
EX PR2/TST201 A
EX PR2/TST201 A 73

Note also that if a number under a loop is preceded by an asterisk (" * "), then that loop must be included in the syntax at least the number of times specified. For example, if the loop included the characters " *1", then the loop must be included at least once.

SECTION 3
CMS-COMMON INTRINSICS

INTRODUCTION

This section describes, in alphabetical order, those input commands which are embedded in ("intrinsic to") the MCP, and which are common to all CMS products.

As discussed in section 2, it is not valid to precede these messages with "EX", because the intrinsics are not separate programs to be loaded and executed. The intrinsics cannot be executed from a user disk, because by nature they are part of the MCP which is on the system disk.

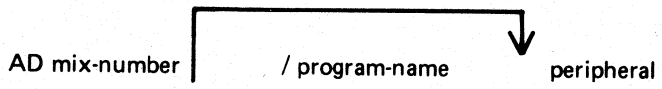
The response to these intrinsics may vary slightly between different CMS products, due to different hardware being used. These variations have been noted in the text where applicable.

AD (Assign Peripheral Device)

This intrinsic allows the operator to assign a particular peripheral to a program that has called for an "unlabelled input file", or that requests a particular output device.

It may also be used to allow file overflow onto a second disk if no disk space is available

Format:



Example 1:

Copy utility requires another disk:

```
COPY INIST TO INDISK3/INIST
10/COPY <12> WAITING FILE 10 NO
... USER DISK
AD 10 DMB
```

(The first message is output by the MCP and the operator responds with the AD message by assigning DMB as the disk to which the remainder of file INIST will be copied. This creates file INIST as a "dual-pack file').

Example 2:

Program "COBOL7", mix number 03, requires a line printer type device:

```
03 COBOL7 <17> WAITING LP NO FILE
AD 03 LPA
```

(The first message is output by the MCP, and the operator responds with the AD message by assigning LPA to mix number 03).

Example 3:

The LIST utility requires an unlabelled tape:

```
LIST TAPE1 MTP NO.LABEL
01/LIST <14> WAITING UNLAB SPURIUS/TAPE1 AT
  ...DEVICE REQUIRED
AD 01 CTB
```

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix number/program AD INVALID	Specified program was not suspended waiting for a device assignment.	Check with MX for name of suspended program.

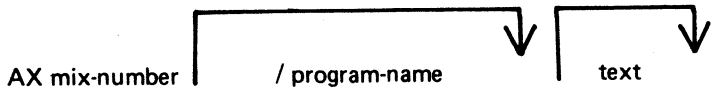
AX (Accept a message for a program)

This intrinsic allows the operator to communicate with a program in the mix. The program must already be suspended waiting for an "accept" (ACPT).

The MCP will prompt the operator for input by printing "mix number/program-name ACPT" on the SPO.

The maximum length of the "text" or operator input is 50 characters. Operating intructions for individual programs will provide the operator with valid "text" responses.

Format:



Example:

The program BM001 displays a message asking for a file name to be entered. The operator responds with the appropriate text, in this case ARSCHG, by the AX message.

```
BM001
01/BM001 BOJ
ENTER BM202 FILE NAME
01/BM001 ACPT
AX 01 ARSCHG
```

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix-number / program-name AX INVALID	Specified program was not waiting for an "accept" or mix number and specified program name do not match.	Check with MX for proper mix-number and program-name combination.

CL (Clear Peripheral)

This intrinsic allows the operator to clear the peripheral from the program and bring the program to End of Job (EOJ). It breaks the "links" between the program and the peripheral.

For example, if the line printer "hangs" during the printing of a report and an attempt is made to DS the program, it will not be possible to discontinue the program unless the line printer is made ready or CL is used to break the "link" between the program and the line printer.

Format:

```
CL { printer peripheral
     tape peripheral
     self-scan peripheral
     ICMD peripheral }
```

Examples:

```
CL LPA
CL SSA
```

Output messages:

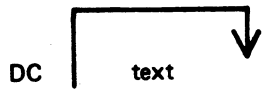
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
CL peripheral INVALID	Program is not waiting on "hung" peripheral.	Check input.

DC (Data Communications operator input)

This intrinsic enables the operator to enter messages from the SPO to the Message Control System (MCS) if data communications activity is in process.The message text, after being stripped of the "DC" characters and the following blank character, is transferred to the MCS input message queue and marked as "operator input".

The interpretation of the message text is defined by the particular MCS.

Format:



Example:

To enter the text "MAKE STATION 2 READY':

DC MAKE STATION 2 READY

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
none	MCS input successful	none
DC INVALID	no MCS in the mix	check input; execute the MCS
DC NOSPACE	There is no available message space in memory for this message	wait a short time then re-input message; if unsuccessful several times, request techrical assistance.

DP (Discontinue and Dump)

This intrinsic is similar to the "DS" intrinsic. The difference is that the disk work space (Virtual Memory on Disk, Virtual Disk) is not freed up and returned to an available status.

The disk work space is, instead, updated from memory with all the most current information about the program. The disk backup is then made into a file (locked) and given a name, "DMFILnn" ('nn" is the mix number for user programs, utilities, and MCP intrinsics).

The peripherals and memory in use by the specified program are made available to other programs.

DP is used when a technical analysis of a particular program is required following a failure during its operation.

Format: **DP mix-number/program-name**

Example:

DP 01/GL060

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix number/program-name DP'ed	DP successful	none.
input INVALID	mix number does not correspond to program name	Check input (reinput if necessary). or Check with MX for proper mix number and program name combination.
input INVALID - NEEDS PROGRAM-ID	program-name is missing	Check with MX and re-input.

DS (Discontinue Program)

This intrinsic causes the orderly termination of the specified program. All peripherals in use by the program are made available to other programs.

Format:

DS mix-number / program-name

Example:

To terminate the program AR040 which has mix number 2:

DS 02/AR040

Output messages:

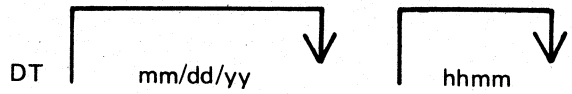
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix/prog DS'ed	DS successful	none
input INVALID	mix number does rot correspond to program name; or program is an MCS.	check with MX, reinput
input INVALID-NEEDS PROGRAM ID	program name not specified	check with MX, reinput

Note: if the program is waiting on a "hung" peripheral device, try the CL intrinsic.

DT (Systems Date and Time)

This intrinsic allows the operator to inquire about or change the system date and time maintained by the MCP.

Format:



Examples:

To inquire about the system date (and time if the system contains a real time clock)

DT

To change the system date:

DT 01/01/78

To change the system date and time:

DT 03/23/78 1234

(March 23, 1978 is the new date. 12:34 is the time).

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
"DD MON YY YYDDD HHMM DOW" where DD = day of month MON = 3 letter abbreviation of month. YY = year YYDDD = Julian date HHMM = time (hours and minutes) DOW = day of week.	Normal response to "DT".	none
<INVALID DATE>	An error was made in one of the following fields: MM DD YY For example = a MM entry of "0" or greater than 12 is invalid. The entire date is rejected, but a valid time entry in the same message will be accepted.	Reinput date portion of message
<INVALID TIME>	A time greater than 2359 was entered. The time is rejected. A valid date entry in the same message will be accepted.	Reinput time portion of message.
<NO CLOCK>	Time entry was made, but system has no real-time clock. Valid date entry will be accepted in same message.	none
MM/DD/YY HH:MM	Normal response to DT inquiry (B80C)	none

FD (Form Define)

This intrinsic allows the operator to define a logical page for a serial printer (SPA) or set top of page for the SPA.

Unless the operator indicates otherwise, the current position is taken as the top of the page.

If the three parameters (HEIGHT, WIDTH, and OFFSET) are specified, then they are used to define a logical page on the SPA. HEIGHT specifies the number of lines on a logical page; WIDTH the maximum number of characters in one line; and OFFSET the number of characters that the printing area is to be offset from the left. An OFFSET of zero specifies the left-most physical position.

WIDTH and OFFSET added together must not be greater than the number of physical print positions on the serial printer. For example, if the physical printer has 255 columns the maximum printing area is given by a WIDTH of 255 and OFFSET of zero. The logical page will remain the same as defined by FD or next warm-start.

Format:

FD SPA [height, width, offset]

Examples:

1. FD SPA 66, 120, 5

defines a logical page on SPA where height is 66 lines and the printing area is 120 characters wide offset 5 columns from the left (that is, from columns 6 through 125, numbering the left-most column as column 1).

2. In order to change the top-of-page position, move the paper to the desired position then execute

FD SPA <empty>.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FD SPA numbers INVALID	FD specifications for height, width, and/or offset are not acceptable. Attempt to print beyond SPA capabilities.	Check input and re-enter.

GO (Restart a Stopped Program)

This intrinsic allows the operator to restart a program which has been stopped with the "ST" command.

Format:

GO mix-number [/ program-name]

Examples:

To restart program whose mix-number is 3:

GO 3

To restart program PR020:

GO 3/PR020

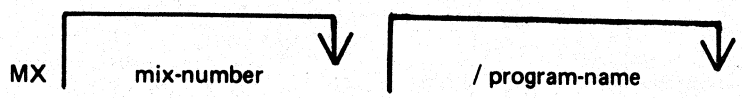
Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix-number/prog-name NOT STOPPED	Specified program was not waiting for a "GO" command.	Check with MX for suspenced program waiting for "GO" reinput.
mix-number/prog-name INVALID	Optional program name was usec anc it did not match the mix number specified.	Check with MX for correct mix number anc matching program rame. Reinput.

MX (Display Current Mix)

This intrinsic allows the operator to inquire about the status of any program(s) currently processing.

Format:



Examples:

To inquire about all programs currently processing:

MX

To inquire about a particular program:

MX 03/PR020

or MX 03

Output messages:

MESSAGE	PROBABLE CAUSES	SUGGESTED ACTION
INVALID MIX	Specified program is not currently running.	Check input (re-input if necessary).
NULL MIX	No programs are currently processing.	None.
INVALID PROGRAM ID	Optional program name was used and it did not match the mix number specified.	Re-input

For each program specified, the following information is provided:

MIX NUMBER

a number assigned by MCP to this program as it was loaded into memory.

PROGRAM NAME

PROGRAM PRIORITY – "A", "B" or "C"

A = lowest priority (that is, application program)

B = medium priority (that is, system utility)

C = highest priority (that is, data communications)

STATUS OF PROGRAM

EXECUTING – program processing normally

SUSPENDED WAITED ON – program processing was temporarily halted. For reasons, see chart below.

SHORT WAITED ON – program is waiting on a resource (that is, Virtual Memory or I/O buffer) which the system can guarantee will be made available in a relatively short time.

SWAPPED OUT WAITED ON – portions of this previously suspended program were temporarily removed from real memory and returned to disk. Memory space was required for other programs in the mix. (Reasons for "swap outs" are same as for program suspension).

Possible messages are summarized by the chart below:

```
 mix   / program  priority──┬─ EXECUTING
number    name              ├─ SHORT WAITED ON ─┐
                            ├─ SUSPENDED
                            │        WAITED ON ─┤
                            └─ SWAPPED OUT
                                     WAITED ON ─┤
                                                ├─ O/C
                                                ├─ VM
                                                ├─ SCL TASK
                                                ├─ OPERATOR INPUT
                                                ├─ ACCEPT
                                                ├─ ZIP
                                                ├─ I/O OR EVENT
                                                ├─ NO DISK FILE (NO FILE)
                                                ├─ DUPLICATE FILE
                                                ├─ SYSMEM FILE
                                                ├─ NO USER DISK
                                                ├─ DIRECTORY SPACE
                                                ├─ DEVICE (NO FILE)
                                                ├─ GO COMMAND
                                                └─ EVENT TIMER
```

Output message examples:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
04/PR060 A EXECUTING	Program processing normally	None
04/PR060 A SUSPENDED WAITED ON O/C	Program is waiting or a file open or close.	None : program will be resumed when file has been opened or closed.
04/PR060 A SUSPENDED WAITED ON VM	Program is waiting on Virtual Memory.	None : do not try to execute too many programs at this time.

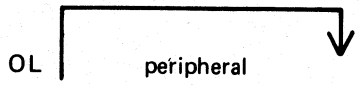
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
04/PR060 A SUSPENDED WAITED ON SCL TASK	Program is waiting for a "command" from the MCP to be completed (such as response to an "OL" input).	None : program will be resumed when SCL task has completed.
10/LR B SUSPENDED WAITED ON OPERATOR INPUT	Program is waiting for some input from operator. (EX: A program previously suspended by ST requires a GO command to continue).	Provide program with appropriate input. Program will continue processing.
08/GL060 A SWAPPED OUT WAITED ON ACCEPT	Program has displayed an "ACPT" message on SPO and is waiting for appropriate response.	Refer to this program's operating instructions for suggested responses to ACPT. Then enter AX, mix number and selected response.
05/AP020 A SUSPENDED WAITED ON ZIP	Program requested assistance of another program in order to complete this job. MCP will automatically load into memory the necessary program(s).	None.
04/PR060 A SHORT WAITED ON I/O	Usually indicative of normal processing, involving I/O activity to disk or peripheral.	None.
05/PR020 A SUSPENDED WAITED ON NO DISK FILE	Program needs (and has not found) a particular file in order to continue processing.	Check SPO for message indicating name of file this program is seeking. Then supply missing file (COPY from backup medium or create it). If in doubt refer to program instructions.
02/PR020 A SUSPENDED WAITED ON DUPLICATE FILE	Program is attempting to place a file of a certain name on disk. However, another file by the same name currently resides on disk. A disk may not contain 2 files with the same name.	Normally, remove the existing file from disk with RM. If in doubt, refer to program instructions.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
10/COPY B SUSPENDED WAITED ON DIRECTORY SPACE	When the disk was initialized the disk directory was constructed to contain a fixed number of file names. The directory has now reached its capacity.	Remove with RM any unnecessary files and program will continue; or DS the suspended program. Replace disk with another disk having sufficient directory space, and re-execute the program.
10/COPY B SUSPENDED WAITED ON NO USER DISK	There is no more available space on disk; or space available is insufficient to hold the file the system is attempting to write; or disk is "checkerboarded".	With KA, analyze amount of available space remaining. If disk is filled remove with RM any unnecessary files; or if disk is filled and a dual-pack file is desired, assign a different disk to this program (see AD intrinsic); or if disk is checkerboarded, use SQ utility to consolidate disk space, then re-execute program that encountered suspension.
10/LIST B SUSPENDED WAITED ON SYSMEM FILE	SYSMEM file cannot be located.	Request technical assistance.
04/PR060 A SUSPENDED WAITED ON NO FILE B90	Device that a program needs in order to continue processing is either unavailable or not ready; or	RY required device; or assign program to alternaive device (see AD intrinsic).
B900	Program needs (and has not found) a particular file in order to continue processing.	Check SPO for message indicating name of file program is seeking. Supply missing file (COPY from backup medium or create).
04/PR060 A SUSPENDED WAITED ON DEVICE B800	Device that a program needs in order to continue processing is either unavailable or not ready.	RY required device; or assign program to alternate device (see AD intrinsic).
02/LR B SUSPENDED WAITED ON GO COMMND	Program was suspended by ST command.	Type "GO" plus mix number of suspended program.

OL (Request for Status Information of Peripherals)

This intrinsic allows the operator to request the status of peripherals on the system.

Format:



Examples:

To display status of all system peripherals:

OL

To display status of a particular peripheral:

OL DKB
OL LPA

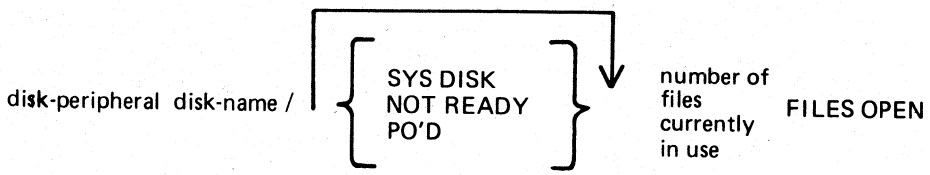
Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
peripheral NOT READY	Peripheral is not on the system; Peripheral may have been "saved"; peripheral may not be correctly loaded.	Check input (reinput if necessary) Ready peripheral if necessary.
OL peripheral NOT ON SYSTEM	peripheral is not configured on machine	none
OL peripheral INVALID	A non-existant device has been specified (that is, OL CCC)	Check input (reinput if necessary).

Other output messages produced by OL depend upon type of peripheral. Refer to the following examples for details.

Examples of disk device output:

The general format of the output message is:



Examples:

DKA AR1/0 FILES OPEN
DKB AR2/SYS DISK 2 FILES OPEN
DMA PRA/NOT READY 0 FILES OPEN
DKA AR1/PO'D 0 FILES OPEN

Examples of magnetic tape device output:

The general format of the output message is:

```
                                  NOT READY
                                  multi-file name
magnetic tape device   {          file name          }
                                  UNLABELLED
                                  PURGED

or

                                  multi-file name
magnetic tape device   {          file name          }  [ { NOT READY } ]  IN USE...
                                  UNLABELLED              { SAVED     }

                                                                          ...BY mix-number / program-name
```

Examples:

CTA NOT READY
CTA ARTAPE
CTA ARTAPE/IN USE BY 10/TAPELR
CTA ARTAPE/NOT READY IN USE BY 10/TAPELR

Examples of output from any other device:

The general format of the output message is:

```
peripheral  [ NOT ]  READY

or

                                  multi-file name
peripheral   {                    file name          }  [ { NOT READY } ]  IN USE...
                                  UNLABELLED              { SAVED     }

                                                                          ...BY mix-number / program-name
```

Examples:

LPA NOT READY
LPA NOT READY IN USE BY 04/PR020
SSA SAVED
SPA SAVED IN USE BY 06/PR060

PG (Purge Tape)

This intrinsic allows the operator to purge (erase) magnetic tape and cassette tape files, thus labelling them as available for output.

Format:

PG tape or cassette peripheral

Examples:

To purge a cassette tape on drive CTA :

PG CTA

To purge a magnetic tape on drive MTC :

PG MTC

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
peripheral * PURGED *	PG successful.	None.
PG INVALID	peripheral not specified in message.	Re-input message
PG peripheral INVALID	Tape could not be purged, as it is "write inhibited", or peripheral is not on the system.	Make certain red tabs on top of cassette are turned outward; make certain "write permit ring" is inserted in tape. Retry PG.

Note: if an attempt is made to purge a tape which is in use, then the response to the OL message for that peripheral is displayed.

PO (Power Off a disk)

This intrinsic allows the operator to "logically" power off a disk (instruct the MCP that the disk is no longer required). At any time when the MCP is idle it is valid to logically power off the system disk with the PO command. This will cause the MCP to terminate. All systems disk files will be closed and SYS-SUPERUTL will go to End of Job (EOJ).

No disk should be removed from the disk drive, no disk units should be powered down, nor should the main cabinet be switched off, until disks have been logically powered off with PO. Failure to observe this practice might cause disk problems at a later date.

Format:

PO disk peripheral

Examples:

PO DKA (disk cartridge)
PO DMB (mini disk)
PO DFA (fixed disk)

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
disk peripheral OK or disk peripheral POWERED OFF	Disk was logically powered off.	It is now permissible to physically power off and remove the disk from the disk drive.
disk peripheral REMOVED WITHOUT PO	Disk was physically removed before being logically powered off.	Check disk for possible corrupted data before re-use.
PO disk peripheral INVALID	specified disk peripheral is non-existant. (ex: PO DKW)	Reinput.
PO disk peripheral NOT ON SYSTEM.	Specified disk is not currently on line.	Reinput.
CANNOT POWER OFF SYSTEM. MIX NOT EMPTY. or PO disk peripheral INVALID	Attempt has been made to PO the system disk while a program is processing.	Allow program to go to End of Job (EOJ), then reinput.

If an attempt to Power Off a disk is made while files on that disk are in use, the OL message for the disk is printed. No further program 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. If the disk is in use, it will not be powered off immediately after giving the PO command though it will print "disk peripheral PO'ED X FILES OPEN", for all disks.

If a disk is removed without being logically powered off, any program using files on that disk will eventually terminate with an error condition indicating hardware failure.

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

PR (Assign Program Priority)

This intrinsic allows the operator to alter the priority of a program by moving it to the highest priority position in the class specified.

Priority "A" is low or normal priority, used for regular work. Within this class, programs which perform more physical I/O operations are given precedence.

Priority "B" is medium priority, used for utilities or programs which may be expected to do emergency work. The priority within this class is reverse historical: that is, a program of this priority placed in the mix will take precedence over previous programs of the same priority.

Priority "C" is high priority, used for data communications programs that are transaction-driven. These are normally dormant, awaiting a transaction, but when required to process a transaction they take high priority to minimize response times. Within this class, programs which do more physical I/O are given precedence. This intrinsic is not implemented on B 90 systems.

Format:

PR mix-number / program-name { A | B | C }

Example:

To change the priority of mix-number 3 (program REP506) to B:

PR 03/REP506 B

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix-number/program-name PR IS priority	Input accepted	None
mix-number/program-name PR INVALID	Mix-number and program-name do not match or priority value incorrect	Check with MX for proper input, and re-enter

RY (Ready a Peripheral)

This intrinsic is used to "ready" a peripheral so the MCP can use it as a resource. When warmstarting, the system will automatically ready all peripherals on the system that are powered on. RY may also be used to Ready a previousy PO'd user disk.

Format:

RY peripheral

Examples:

To ready a self-scan:

RY SSA

To ready a line printer:

RY LPA

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
RY peripheral INVALID	Attempt was made to Ready a non-existant peripheral (that is, RY LLP); Attempt was made to ready a device alreacy "ready".	Check input (reinput if necessary)
RY peripheral NOT ON SYSTEM	Attempt was mace to ready a peripheral on-line to the computer.	Check input (reinput if necessary).

SF (Substitute Disk File)

This intrinsic allows the operator to direct a program to a particular disk file if it is waiting on a "NO FILE", "NO PACK", "DUPLICATE FILE", or "BAD FILE NAME" condition.

This command causes temporary modification of the program's file parameter block. The modification remains in effect for the current execution only, or until it is remodified by the program during the current execution.

The command can only be used when the program is suspended waiting on one of the above conditions. It is not possible to anticipate the program's requirements and modify the file parameter block in advance.

This intrinsic is not implemented on B 90 and B 900 systems.

Format:

```
                                  disk-name /
SF mix-number [ / program-name ] { disk-name / file-name }
                                  file-name
```

Examples:

Program AP10 (mix number 01) requests a disk file called APD2T on disk APD. To direct the program to use file APD2S on the same disk:

```
01/AP10 <10> WAITING APD/APD2T DK NO FILE
SF 01/AP10 APD2S
```

(the first line is the MCP output message; the second is the input SF message in response to the "NO FILE" condition).

To direct the same program to use file APD2T on disk APD1:

```
SF 01/AP10 APD1/
```

or

```
SF 01 APD1/
```

To direct the same program to use file APTEMP on disk ARTD:

```
SF 01 ARTD/APTEMP
```

Output message:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
mix-number / program-name SF INVALID	Program is not waiting on a "no file" or other condition, or mix-number and program-name do not correspond.	Check with MX and re-enter.

ST (Temporarily Suspend a Running Program)

This intrinsic places a temporary halt on a program that is running. The program still appears in the mix. The data needed to restart the program exactly where it stopped is transferred from memory and stored on disk. The memory that was being used by the "stopped" program is now made available to the MCP for other use. The GO command must be used to restart the program.

Format:

ST mix-number [/ program-name]

Examples:

To stop the program whose mix-number is 3:

ST 3

To stop the program PR020:

ST 3/PR020

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
mix-number/program name STOPPED	ST successful	none
mix-number/program name INVALID	Program has already been stopped or program is not in the mix.	Check with MX for status of program; (reinput if necessary).

SV (Save Peripheral)

This intrinsic allows the operator to "logically" power off any input/output device (except disks, see PO intrinsic) in order to prevent their use by any program.

"Tape peripherals" include magnetic tape (MT) and cassette tape (CT).

"Printer peripherals" include line printer (LP) and serial printer (SP).

Format:

```
      ┌                                  ┐
      │   tape peripheral                │
      │   printer peripheral             │
SV    {   self-scan peripheral           }
      │   card-reader peripheral         │
      │   card-punch peripheral          │
      └                                  ┘
```

Examples

```
SV LPA
SV SSA
```

It is possible to "save" a device that is being used by a program. This will allow the program presently assigned to this device to continue using it, but will prevent any subsequent programs from using the device. For example:

```
SV LPA
LPA SAVED IN USE BY 06/PR060
```

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

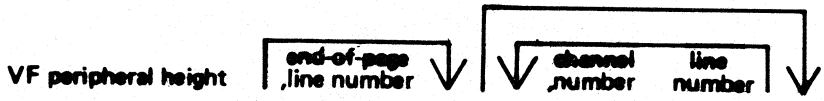
Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
peripheral OK	SV successful	None.
SV peripheral INVALID	Attempt was made to save a disk peripheral or device has already been saved.	Reinput (if necessary using correct peripheral.
SV peripheral NOT ON SYSTEM	Specified peripheral is not on-line to the computer.	Check input; reinput if necessary.
peripheral POWERED OFF	SV successful.	None.

VF (Vertical Format on Printer)

This command allows the operator to define the actions to be taken by the printer when certain vertical format commands are sent. This command applies only to printers which have soft vertical format control.

Format:



The height field specifies the page height in lines. The channel number and line number fields are optional but when specified they must both be present as a pair. The channel number should be 2-11 and page height should not be more than 94.

Example:

VF LPA 66, 60, 2 10

where page height = 66
end of page = 60
channel number = 2
line number = 10

NOTE: For details see LOAD.VFU utility.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
VFU LOAD FAIL - ILLEGAL PARAMETER LIST	Typing error	correct the input and re-enter
VFU LOAD FAIL - peripheral NOT ON LINE	The specified peripheral is not ready	RY the peripheral and re-enter
VFU LOAD FAIL - peripheral IN USE	The specified peripheral is in use by a program	Wait until program has closed the printer file, then re-input
VFU LOAD FAIL - peripheral HAS NO SOFT VFU	The peripheral is not a B9249-30/50 line printer	None

SECTION 4
CMS-COMMON UTILITIES

INTRODUCTION

This section describes all standard CMS utilities that form part of a CMS system software release. The applicability of any utility depends on the type of hardware available. For example, utilities requiring console files cannot be executed on machines without a console: as an example, CREATE, AMEND and UPDATE cannot be run on a B 1800.

Table 4-1 gives a list of all required peripherals for each utility. In this table, required peripherals are denoted by the letter "R", and optional peripherals by the letter "O". One asterisk ("*") indicates that out of all the options, at least one is required. In particular, those utilities requiring a line printer may use a console printer by default if the line printer is not present on the system. Two asterisks ("**") indicate that out of all the options, at least two are required.

STAR FILES

The star-file facility permits the initiating message parameters of most utilities to be specified on a disk file which is referenced in the initiating message.

The utilities which do not support this feature are:

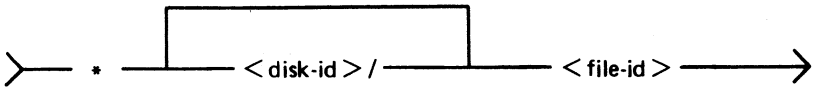
CP – Compute
DA – Disk Analysis
KEY.CHECK – Key Validity
XD – Delete Bad Disk Sectors
PPID – Pseudo-Pack Identifier
WL – Which Log
ARCS – Automatic Run Control System
BF – display Backup File information
RB – Remove Backup files

as well as the following B 90-only utilities:

GEN.DUMPFL – Create empty B 90 Memory Dump File
PATCHMAKER – Patch B 90 machine code, and object program files
CONFIGURER – Configure B 90 System Software

CO supports a star-file facility which is slightly different from the general utility star-file facility. Refer to CO, section 6, for details.

The syntax (as inserted in the initiating message) is:



The utilities which support the star-file feature have the limitation of 400 significant characters within their initiating message. (The initiating message for LD may be up to 600 significant characters in length).

Star files may contain any number of records, with any record size. A single space is considered a significant character, and any double space encountered is considered to be a single space (and hence only one significant character). Spaces at either end of the message are ignored.

Where star files are a feature of a particular utility, the star file may be placed at any point in the initiating message after the utility name. The initiating message may contain any number of star files but these may not be nested: that is, the information within a star file must not contain a call on any other star file.

If the specified file cannot be found, a "<file-name> NOT FOUND" message is displayed by the utility.

Table 4-1. Peripherals Required by CMS-Common Utilities

Utility	Console	Disk	Serial Printer	Self-Scan	Line Printer	Cassette or Mag.Tape	Card Reader	Card Punch	Paper Tape	ICMD
ADD		R				R				
AMEND *	R	R	O	O						
BF	R	R								
CH		R								
CHECKADUMP		R				R				
CHECK.DISK		R								
CO *		R	O		O	O	O			
COMPARE **		O	O		O	O	O		O	
COPY **		O				O	O	O	O	
CP										
CREATE *	R	R	O	O						
DA	R	R	R							
DCR	R	R								
DD		R								
DSKUTL	R	R	O		O					
DUMP		R				R				
FL	R	R		R						
FS		R								
ICMD *		R	O		O					R
IR		R								
KA *		R	O		O					
KEY.CHECK	R	R	O		O					
KX		R								
LB		R								
LD		R				R				
LF		R								
LIST **		O	O		O	O	O		O	
LOAD		R				R				
LR *		R	O		O					
MODIFY *	O	R	O		O					
PB	R	R	O		O					
PD		R								
PL *		R	O		O					
RB	R	R								
RL	R	R								
RM		R								
SCR	R	R	O	O	O	O				
SQ		R								
SYCOPY	R					R				
TAPELR *			O		O	R				
TAPEPD						R				
TL		R								
UNLOAD		R				R				
UPDATE *	R	R	O	O						
XD		R								

Examples:

1. RM *M101A/RMFILE
   where RMFILE is a disk file on disk M101A containing one record with the contents REP200, REP562, RQ=, RCOPY
2. DA *DISK1/F
   where F is a file containing a list of filenames.

3. COPY *DISK2/B
   where B is a file containing the remainder of the initiating message – "FILEA <BOTH> TO DISK3/FILES"

SYS-SUPERUTL

This system utility provides the following functions:

CH – change the name of a file or group of files
KX – interrogate disk space
PD – interrogate disk directory
RM – remove a file or group of files
IR – initiate recall of SPO log messages
LB – look back in SPO log
LF – look forward in SPO log

It executes automatically if the program file is on the systems disk when one of these functions is required. This program is also automatically executed at warmstart time and co-ordinates logging functions at that time.

SYS-SUPERUTL supports the following filetypes:

Filetype	Description
00	Normal Data
01-0E	Source Language
0F	Source Library
10-12	Normal Code
13	Protected Code
14-16	Interpreter/SORTINTRINS/MCPX/SAU
21	SYSLANGUAGE local language
22	SYSCONFIG – a required file
30	Virtual Memory/Dump
31	System Log
40-41	MPLII Compiler Work
81	Key
A0	Printer Backup

Of these, the following filetypes are system files:

13	Protected Code
14-16	Interpreter/SORTINTRINS/MCPX/SAU
21	SYSLANGUAGE local language
22	SYSCONFIG
30	Virtual Memory/Dump
31	System Log

A request for the removal of a system file causes the RM utility to output

```
<file-name> IS A SYSTEM FILE
AX <mix-no>/RM ACPT
```

Then, to remove a system file:

```
AX <mix-no> <file-name> OK
```

The utility has some features which can cause the operator confusion. The utility does not appear in the response to the MX command unless it is actually performing one of its functions, when it appears as 12/PD or 12/CH etc., according to the function which it is currently performing. If an attempt is made to execute one of the SYS-SUPERUTL functions when it is already busy, then a response of "<64> LOAD FAILURE UTILITY BUSY" is returned.

SYSLANGUAGE

All CMS utilities which output SPO messages, with the exception of PATCHMAKER, GEN.DUMPFL, BF, RB and SYCOPY, display these messages from a common SYSLANGUAGE file. THIS FILE MUST BE PRESENT IN ORDER FOR ANY OF OF THE UTILITIES USING IT TO EXECUTE. Associated with this file is the message:

"INVALID DICTIONARY ENTRY <entry-no>"

which denotes that the utility executing has attempted to display a message which is not contained in the dictionary file SYSLANGUAGE.

PSEUDO-PACKS

Pseudo-packs allow fixed disk units containing multiple disk platters to be handled as one large contiguous disk. This enables the MCP to address the space on all disk platters as one large available area.

The disk structure that makes this possible is the Pseudo-Pack Identifier Table (PPIT) and is only relevant to fixed disk directories. The fixed disk directory generated at disk initialization for systems using pseudo-packs consists of four parts:

The name list.
The disk file headers for each file.
The available table.
The Pseudo-Pack Identifier Table (PPIT).

The relationship between these parts is shown in figure 4-1.

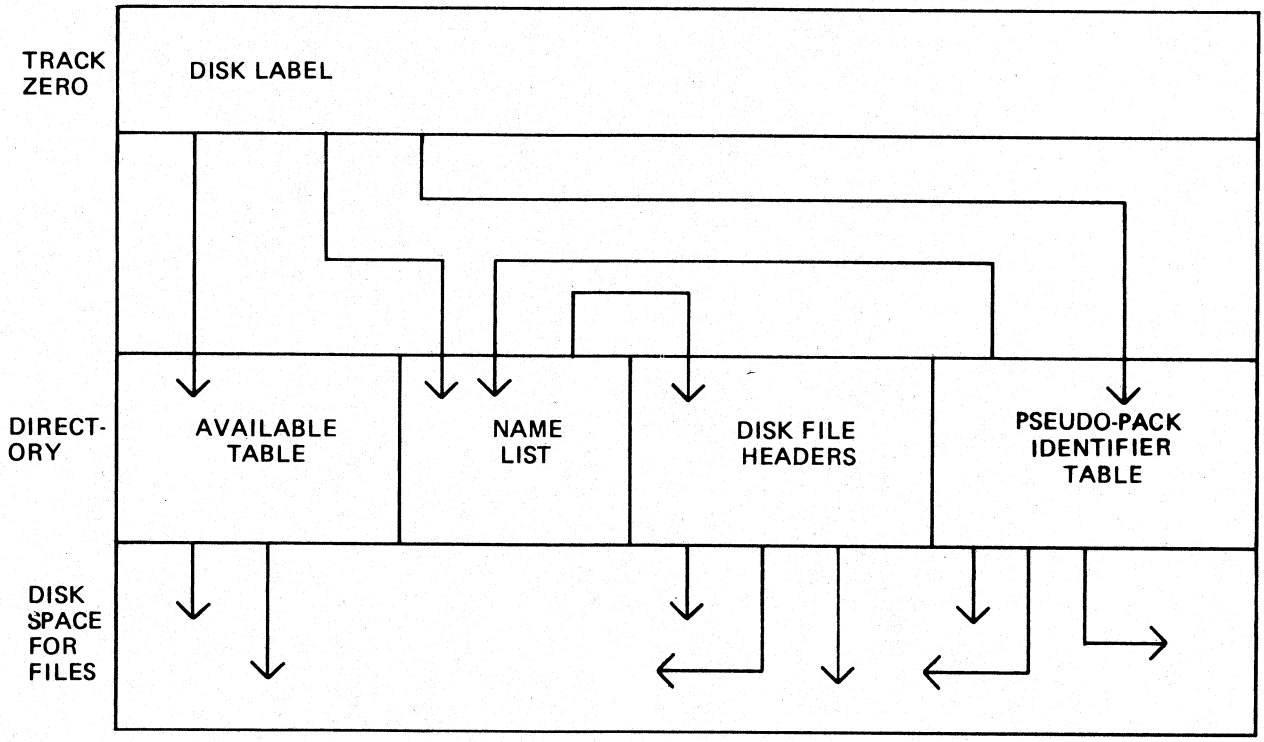


Figure 4-1. Fixed Disk Directory Structure

The available table and disk file headers generated at initialization time have entries which reflect the logical structure of the entire disk unit.

The name list now contains a list of each file on the entire disk unit and a unique identifier, which is used to distinguish files contained on one pseudo-pack from files contained on another pseudo-pack.

The PPIT is a list of all the pseudo-packs declared on the system. An identical PPIT exists on each fixed disk unit.

The ADD PACK intrinsic enables the operator to declare a new pseudo-pack to the system. It also allows the operator to declare the pseudo-pack as restricted or unrestricted. Once a pseudo-pack has been declared restricted or unrestricted, this designation cannot be changed.

When a new file is created on an unrestricted pseudo-pack, areas for the file may be allocated on any of the fixed disks. When a file is created on a restricted pseudo-pack (or with physical unit pack-id), areas for the file are allocated only on the designated unit.

When searching for a file on an unrestricted pseudo-pack, the PPIT and file directories on all fixed disk units are searched, otherwise, only the PPIT and file directory on the designated unit are searched.

The Pseudo-Pack Identifier Display (PPID) utility allows the operator to list the Pseudo-Pack Identifier Table on the operator display terminal (ODT) or line printer.

LOGGING

When the system is warmstarted, the SYS-SUPERUTL utility is initiated and SYS-LOG files are created. The information about the number and size of log-files is stored in a file called "SYSCONFIG" (see CONFIGURER). The MCP then initiates a function of SYS-SUPERUTL, which starts up the "TL" utility, and the transfer of log-files to a "SYS-LOG-HOLD" file begins. When all the log-files are transferred and TL goes to End of Job, SYS-SUPERUTL removes the old SYS-LOG files and creates new SYS-LOG files.

During a session, all the console input/output messages that normally appear on the SPO are stored in SYS-LOG files SYS-LOG-01 through SYS-LOG-nn, where "nn" is 03 to 16 (see CONFIGURER). When one log-file is full, the messages are directed to the next log-file. When all the log-files are full, the logging is directed to the first file again. This overwrites the information held in the SYS-LOG-01 file unless the utility "TL" is begun beforehand, which will transfer all the transferable log-files and keep them in the "TRANSFERRED" state (see TL and WL).

The system will automatically transfer all log-files only at warmstart time.

COMMON UTILITY OUTPUT MESSAGES

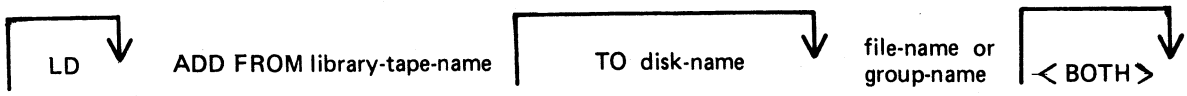
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
INVALID CHARACTER IN IDENTIFIER	Disk name or file name contains character(s) not permitted by the system. Valid characters are: A-Z, 0-9, . (dot), - (dash).	Check input and re-input if necessary.
ILLEGAL PARAMETER LIST	Typing error.	Check input. After words "ILLEGAL PARAMETERS LIST" system will display portion of input message that contains error.
file-name NOT FOUND or file-name NOT ON LINE	Specified file name is not on disk.	Check input and re-enter if necessary; check for correct disk; supply specified file (COPY file from back-up medium or create file).

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO SPECIFICATIONS GIVEN	Input message is incomplete.	Check input and re-enter.
DISK disk-name NOT OPENED NOT ON LINE or DISK disk-name NOT AVAILABLE or DISK disk-name FOR XD NOT AVAILABLE	Specified disk is not on-line to computer.	Check input and re-enter if necessary; Check for correct disk; Ready disk;
FILE LIST MAY BE INCOMPLETE	A group of files on an unrestricted pseudo-disk was requested, but the disk is off-line.	
PSEUDO-DISK pseudo-disk-name ON DSK disk-name NOT AVAILABLE	The pseudo-disk specified has not been found.	
DISK disk-name IS A PSEUDO-DISK	The disk specified is not a physical disk but a pseudo-disk: probably the input disk name is incorrect. This message is printed by those utilities which cannot handle pseudo-disks.	Correct the disk name and reinput.

ADD (Add Files From Library Tape to Disk)

This function, a part of the utility LD, allows the operator to copy files from a library tape to a disk.

Format:



If the <BOTH> option is used immediately after a request to add a keyfile, the data file will also be copied, provided it does not precede the keyfile on the library tape. The keyfile will then refer to the disk which now holds the data file (rather than the disk from which the data file was dumped to the library tape).

A file is copied only if no other files on the specified disk have the same name.

Examples:

To copy all files from ARTAPE to the system disk:

ADD FROM ARTAPE=

To copy a file called PRFILE from PRTAPE to a disk called PRBU:

ADD FROM PRTAPE TO PRBU PRFILE

To copy files called GL300 and GL200 from GLTAPE to the system disk:

LD ADD FROM GLTAPE GL300 GL200

To copy a keyfile called PR240K and its data file from a tape called PRTAPE to a system disk=

ADD FROM PRTAPE PR240K <BOTH>

Since "ADD" is a part of the utility LD, "LD" is actually what will appear in a mix message. To discontinue the ADD function, "DS mix-number/LD" must be used.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name LOADED	ADD successful	None.
library-tape-name NOT A RECOGNIZED DUMP TAPE	Specified tape does not have a valid CMS label, or has not been created by the LD utility (for example: tape is a COPY tape).	Provide correct tape and retry; or DS LD utility.
NO FILES IN THE FAMILY group-name ON TAPE library-tape-name FOR ADD	Specified group was not found on library tape.	Check input and re-input if necessary; Check for correct library-tape.

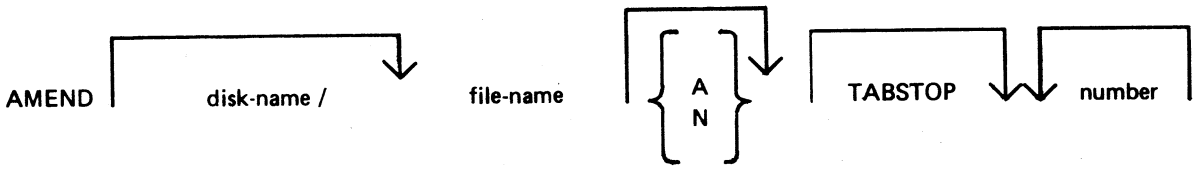
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO FILE file-name ON TAPE library-tape-name FOR ADD	Specified file was not found on library tape.	Check input and re-input if necessary; check for correct library tape.
file-name LOAD/ DUMP DISCREPANCY	End of File reached before expected. Disk File Header may be corrupted. Possibly due to mis-reading of tape.	Try tape on different drive in case the drive is at fault.
NO FILES TO LOAD	No files were found on this tape to copy to disk.	Check input and re-input if necessary; Check for correct tape.
file-name NOT LOADED - ALREADY ON DISK. ALTHOUGH WITH DIFFERENT ATTRIB-UTES.	File not copied as a file with the same name already exists on disk. If the 2 files differ in record, block, and file sizes, the "DIFFERENT ATTRIB-UTES" message prints.	Normally remove with RM the duplicate file and re-attempt the ADD.
file-name DATA FILE NOT FOUND ON TAPE FOR LOAD.	Data file for given keyfile does not follow on tape. Data file cannot be copied. Keyfile is copied.	None.

Note: Refer to "Common Utility Output Messages" for additional aid.

AMEND (Disk File Amending)

This utility is used to modify records within an existing data or source file. The CREATE and UPDATE utilities use many similar features. It is only available for use on systems which have console files. The utility supports the star-file feature. If a file is AMENDed, the generation number of the file is incremented by one.

Format:



Input may be either alphanumeric (A) or hexadecimal (N) (see CREATE for details). The default is A.

Entering TABSTOP in the initiating message causes AMEND to set up TAB positions coinciding with the end of the console line as well as any other tabs specified. Tabs must be used with AMEND, despite the fact that no new records are being entered.

Default tab positions have been chosen to allow a maximum number of characters to be inserted on one line. AMEND uses nine for the record number and 110 for the contents of the record. In addition, manually selected tabs may still be used. The end-of-console line-tabs (depending on record size and file-type) are as follows:

Source or Data alphanumeric : 111 221 331 441
Data Hexadecimal : 56 111 166 221 276 331 386 441 496

These tab positions are the same as those set for CREATE with regard to record input, although AMEND has no facility to input new records.

Examples:

	Tabs Set at:
AMEND FILEA TABSTOP	111 221 331
AMEND FILEB A TABSTOP 51 61 221	51 61 111 221 331 441
AMEND FILEC N TABSTOP	56 111 166 221 276 331 386 441

In these examples, FILEA is a sourcefile with a record size of 410,
FILEB is a datafile with a record size of 500,
FILEC is a datafile with a record size of 450.

The "number" option may be used to set tab positions for character input (see CREATE for details).

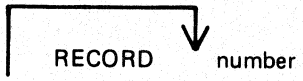
The utility operates in two modes: "Record Modify" (PK2) and 'Record Select" (PK3).

PK1	PK2	PK3	PK4	PK5	PK6
write last & get next	modify	select	---	---	EOJ

An OCK4 "help" option is provided, which will output the above options when pressed in either Modify or Select mode. In order to show which mode the utility was in when OCK4 was pressed, an asterisk (*) is printed next to that mode on the Help display.

PK1 is used to select the next sequential record in the file to be printed. The use of PK1 terminates "Record Modify" and "Record Select" modes, therefore a re-selection of mode must be made before continuing.

If PK3 ("Record Select" mode) is used, the required record is identified by logical record number using this format:



The "number" may take any value from 1 to the number of records in the file.

PK2 is used to make corrections to existing records. This PK operates as PK2 in CREATE utility (see CREATE for details).

Example:

To amend a source file called MYFILE, record size 40 bytes, tab set at 5, 10, 15, 20:

AMEND MYFILE 5 10 15 20

First select a record by pressing PK3, and then enter "20" for logical record 20 in MYFILE. The utility selects and prints the contents of record 20:

20 ABCDEFGHIJKLMNOPQRST

To replace characters, press PK2 and type the replacement

D : ZZZZ : OCK1

resulting in "20 ABCDZZZZIJKLMNOPQRST"

Or if insertion of characters is desired, type the characters to be inserted into the record:

Z : XXXXXX : OCK2

resulting in "20 ABCDZXXXXXXZZZOPQRST"

NOTE

The insertion from character six to eleven will result in the shifting of characters "ZZZIJKLMN" from byte position 12 to the boundary of the next tab position, which is 15. Therefore only 3 characters "ZZZ" are shifted from 12 to 14 and "IJKLMN" are lost. The text from the next tab position 15 onwards is not affected.

Starting Byte for Modification

A starting byte can be specified for the modification of a record.

If both the identifying string and the start position are specified, the utility scans from the start position for that identifying string. The portion of the record before the start position is ignored. If that identifying string does not exist, "BYTE WITHIN RECORD SPECIFIED NOT FOUND" is displayed and the utility then awaits re-input.

Format:

identifying string	:	amending character string	:	start position
or				
	:	amending character string	:	start position
or				
identifying string	:	amending character string	:	

Example:

The following file (named FILEA) is to be amended.

Rec. No.	Contents of Record
1	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
2	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
3	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
4	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
5	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
6	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
7	AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE

1. This first example illustrates modification of each record by replacement.

   Enter: "AMEND FILEA"

   The following is displayed:

   ?DATA AMEND

   PK1 PK2 *PK3 PK6
   NEXT MODIFY SELECT END

   Press PK1. This causes the next record to be displayed (in this case, the first):

   1 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE

   Enter the amending command and press OCK1 (for replacement):

   BBBBB:XXXXX: (OCK1)

   AMEND now displays the AMENDed record:

   1 AAAAABBBBBXXXXXDDDDDAAAAABBBBBCCCCCDDDDDEEEEE

Press PK1.

2 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
:XXXXX:11 (OCK1)
2 AAAAABBBBBXXXXXDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
Press PK1
3 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
BBBBB:XXXXX:2 (OCK1)
3 AAAAABBBBBXXXXXDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
Press PK1.
4 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
BBBBB:XXXXX:26 (OCK1)
4 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXDDDDDEEEEE
Press PK1
5 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
:XXXXX:31
5 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXDDDDDEEEEE
Press PK1
6 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
BBBBB:XXXXX:7
6 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXDDDDDEEEEE

?END AMEND

2. In this next example, insertion is being performed:

Enter: AMEND FILEA

As for replacement, the following is displayed:

```
PK1      PK2      *PK3 PK6
NEXT MODIFY SELECT END
```

Press PK1 to display the next record:

```
1 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
```

Enter the amendment followed by OCK2:

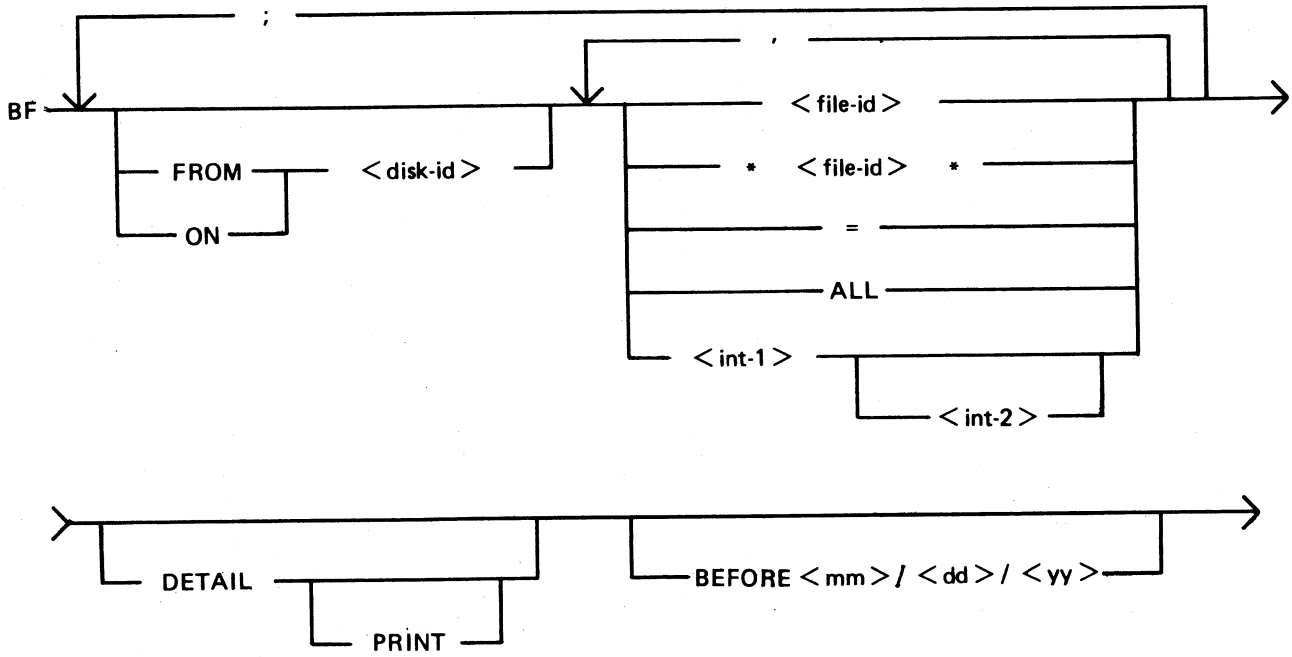
```
BBBBB:XXXXX:  (OCK2)
  1 AAAAABBBBBXXXXXCCCCCDDDDDAAAAABBBBBCCCCCDDDDD
Press PK1
  2 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
 :XXXXX:11 (OCK2)
  2 AAAAABBBBBXXXXXCCCCCDDDDDAAAAABBBBBCCCCCDDDDD
Press PK1
  3 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
 BBBBB:XXXXX:2 (OCK2)
  3 AAAAABBBBBXXXXXCCCCCDDDDDAAAAABBBBBCCCCCDDDDD
Press PK1
  4 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
 BBBBB:XXXXX:26 (OCK2)
  4 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXCCCCCDDDDD
Press PK1
  5 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
 :XXXXX:31 (OCK2)
  5 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXCCCCCDDDDD
Press PK1
  6 AAAAABBBBBCCCCCDDDDDAAAAABBBBBCCCCCDDDDDEEEEE
 BBBBB:XXXXX:7 (OCK2)
  6 AAAAABBBBBCCCCCDDDDDAAAAABBBBBXXXXXCCCCCDDDDD

?END AMEND
```

Output Messages:

Refer to the section on the CREATE utility for output messages.

BF

The BF utility enables the user to display information about backup files residing on one or more disks. Entering BF or BF HELP displays the syntax diagram for BF, which is as follows:



where:

<disk-id>	specifies the disk on which the utility seeks backup files; the default is the backup disk designate.

The following specifications designate the files or groups of files about which the information is required:

<file-id>	designates a backup file or a family of backup files. Either the external or the internal file name may be specified.
* <file-id> *	is used when <file-id> alone could cause confusion (for example, BF * 5 * designates the backup file named 5 instead of the one named PB00005).
=	specifies all the backup files PBxxxxx, where 1 ≤ xxxxx ≤ 65535.
<int.1>	specifies the backup file PB<int.1>.
<int.1> – <int.2>	specifies all the backup files from PB<int.1> to PB<int.2>.
ALL	specifies all backup files irrespective of their names.
DETAIL	option causes BF to display for each backup file specified: – the external name of the backup file – the internal name of the backup file – the file's name in the backup file – the filesize – the creation date and time – the name of the creator program. The default for DETAIL is to display <file-id> ON LINE for each file found within the range specified.

BEFORE <mm/dd/yy> option causes all backup files created before the specified date to be listed.
MM and DD (month and day respectively) may be either 1 or 2 digits, while YY must be 2 digits.

Examples:

To specify the file PB00006 from the system disk:

BF 6

To specify the file PB00026 and all backup files in the family MPL=, on the disk USEDSK:

BF FROM USEDSK 26 , MPL=

To specify PB00016, PB00019 on the system disk, all the backup files named PBxxxxx on the disk TASK and all backup files in the range PB00040 to PB00063 on the disk ARDSK, each one of these files subject to the date given:

BF 16,19 ; FROM TASK = ; FROM ARDSK 40-63 BEFORE 1/6/81

Messages:

The syntax error messages issued by the utility are self-explanatory. The utility goes to EOJ after having displayed:

"***** BF ABORTED *****"

and having advised the user to use the BF HELP utility.

In the cases where the utility issues a message which includes quotes, the user should read "asterisk" instead of "quote".

If a disk cannot be opened, BF continues executing, going on to the next disk specified, after having displayed the following message:

UNABLE TO ACCESS DISK <disk-id>

If a file cannot be found, or no files are found in a family, BF goes on to the next file specified, after having issued appropriate messages concerning the absent files.

On completion, BF issues the message:

** BF COMPLETE **

Limitations:

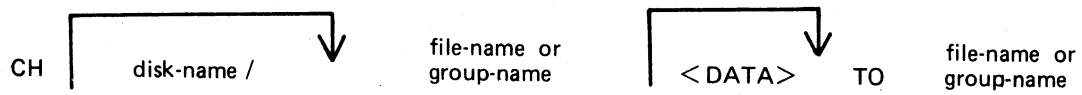
Although the filetype for printer-backup files is supported by the B 90, the B 90 system has no current facilities for creating printer-backup files.

CH (Change File Name)

(a function of SYS-SUPERUTL)

This utility allows the operator to change the name of a file or group of files on disk. The <DATA> option allows the data file of an indexed pair to be changed, and it will also cause the keyfile to refer to the new data file name (the keyfile name does not change).

Format:



Examples:

To change the name of a single file:

CH BPS320D/DCSTSK36K TO DCSTSK
CH DCSTSK TO INDISK3TSK

To change a group of files:

CH BPS320A/AR= TO BP=
CH PRB= TO PR=

To change several different files or groups of files:

CH DCSTSK TO INDISK3TSK, BPS320A/AR= TO BP=

To change the name of the data file of an indexed pair:

CH AR200K <DATA> TO AR200BU

Note: if a change of group file name is specified with the <DATA> option then the data file should appear in the directory after the keyfile. If this is not the case then the name of the data file is changed first, and when the attempt to change the key file name is made, a "data file-name NOT FOUND" message will be displayed. This will not occur when changing the name of a single indexed file.

Output messages:

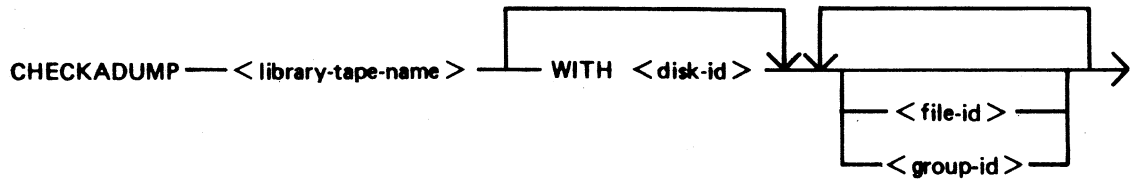
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
"file name" CHANGED TO "file name"	File name successfully changed.	None.
"file name" NOT CHANGED - NOT FOUND	Specified file name is not on disk.	Check input or (re-input if necessary). Check for correct disk.
NO FILES FOUND FOR CHANGING IN THE FAMILY. "group-name"	Specified group name is not on disk.	Check input (re-input if necessary) Check for correct disk.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
"file name" NOT CHANGED IN USE	File's name cannot be changed because it is currently being used by system.	Wait until file is no longer in use, then re-input.
"name" FILE IDENT-IFIER TOO LONG	Attempt has been made to change a file name to more than 12 characters in length.	Re-input.
"file name" NOT CHANGED - ILLEGAL REQUEST	Attempt has been made to change the name of a file to "SYSMEM" (a name reserved for system use) or all spaces.	Re-input.
"file name" NOT CHANGED - "file name" ALREADY ON DISK	Attempt has been made to duplicate the name of a file already on disk.	Re-input.
KEYFILE "file name" NOW POINTS TO DATA FILE "file name".	Successful completion of data file name change.	None.
<64> LOAD FAILURE UTILITY BUSY	Another function of SYS-SUPERUTL is being executed.	Wait until SYS-SUPERUTL is free, then re-input.

CHECKADUMP (Compare Library Tape with Disk)

This utility allows the operator to compare information in files on a library tape with corresponding files on disk. It is used to verify that a library tape is correct after files have been DUMPed or UNLOADed, or that disk files are correct after files have been ADDed or LOADed. Specified tape is processed sequentially, file by file, and the disk is searched for corresponding files. The utility will notify the operator 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 tape.

Format:



Examples:

To compare files on the tape called PRTAPE with the corresponding files on the system disk:

CHECKADUMP PRTAPE

To compare files on the tape called ARTAPE with the corresponding files on a disk called ARDISK2:

CHECKADUMP ARTAPE WITH ARDISK2

To compare the file TESTFL on the tape called BRTAPE with the corresponding file on the disk called ARDISK3:

CHECKADUMP BRTAPE WITH ARDISK3/TESTFL

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
library-tape-name NOT A RECOGNIZED DUMP TAPE	First record of tape not recognized by CHECKADUMP. Tape may not have been created by LD utility.	None. Utility ends.
library-tape-name INVALID DIRECTORY ON TAPE	More or fewer entries in directory on tape than specified in the first record of tape.	None. (See CMS MCP Reference Manual for additional information on tape formats.
COMPARISON ERROR ON library-tape-name ON DISK FILE HEADERS.	Header in body of tape is not identical to respective header in disk directory. The error count for the file is increased by 1.	Recreate dump tape.
COMPARISON ERROR ON file-name FILE NOT FOUND FOR CHECK.	Corresponding disk file cannot be found for file on tape.	Recreate dump tape.
DISK EXPIRING BACKUP RECOMMENDED	Self Explanatory	Backup Disk

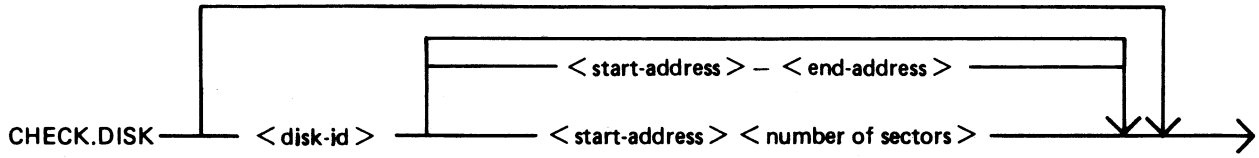
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
COMPARISON ERROR ON file-name FILE NOT AVAILABLE FOR CHECK	Corresponding disk file cannot be read for this file on tape.	Recreate dump tape.
COMPARISON ERROR ON file-name (AROUND RECORD number)	Discrepancy between disk file and tape file. Record number in vacinity of error in file is printed, if possible. One is added to error count for this file.	Recreate dump tape.
COMPARISON ERROR ON file-name AROUND END OF FILE	Difference in length of tape and disk files. One is added to error count for that file.	Recreate dump tape.
COMPARISON ERROR ON file-name DIFFERING FILE SIZES	Difference in sizes of disk and tape files.	Recreate dump tape.
COMPARISON ERROR ON file-name DIFFERING FILE TYPES	Difference in file types of files being compared.	Recreate dump tape.
COMPARISON ERROR ON file-name DIFFERING RECORD SIZES	Difference in record sizes of the files being compared.	Recreate dump tape.
COMPARISON ERROR ON file-name DIFFERING BLOCK SIZES	Difference in block sizes of files being compared.	Recreate dump tape.
NO DISCREPANCIES BETWEEN DUMP TAPE library-tape-name AND DISK disk-name.	CHECKADUMP successful.	None.
DISCREPANCIES FOUND BETWEEN DUMP TAPE library-tape-name AND DISK disk-name.	Discrepancy discovered between disk file and tape file.	Recreate dump tape.

NOTE: Refer to "Common Utility Output Messages" for additional messages.

CHECK.DISK (Check all Sectors of a Disk)

This utility checks either a specified area, or the whole area, of the specified disk by checking blocks of 32 sectors. If an error occurs in a block, each sector within that block is checked individually.

Format:



It is possible that, because of a hardware failure, an error may be detected when a block read is being performed but no errors are detected during the subsequent sector-by-sector read of that block. This is referred to as an "inconsistent error".

When inconsistent errors are encountered, CHECK.DISK continues execution. It is therefore possible to have several read error messages output by the MCP while the utility displays a "NO ERRORS" message on completion of the check.

Checking of only defined areas is permitted by specifying sector ranges in the initiating message (start and end addresses or number of sectors).

These sector ranges may be up to 8 decimal numbers or up to six-digit hex numbers delimited by the characters @ (AT) or . (PERIOD).

Example:

```
CHECK.DISK ARBK 32-128
or CHECK.DISK ARBL @000020@-@000080@
```

The disk-name must be specified if sector ranges are required. If sector ranges are not specified, the utility will default to checking the complete disk from sector zero to the end address.

The utility determines whether a sector with a read error is denoted as BAD in the directory available table and displays the messages on encountering such an error:

```
                ERROR NOTIFIED ON READING DISK disk-name
followed by     SECTOR <address> DENOTED AS BAD IN DISK DIRECTORY
         or     SECTOR <address> NOT DENOTED AS BAD IN DISK DIRECTORY
```

If one or more inconsistent errors occur, the message:

"INCONSISTENT ERROR(S) NOTIFIED – POSSIBLE MEDIUM/DRIVE FAULT"

is displayed on completion of a disk or disk area check.

If no consistent errors are found, but one or more inconsistent errors occur, then "NO CONSISTENT ERRORS" is displayed with the above message.

If one or more consistent errors are encountered, one of the following two messages is displayed on completion of a disk or disk area check:

```
    ONE CONSISTENT ERROR NOTIFIED
or <n> CONSISTENT ERRORS NOTIFIED
```

If no errors – consistent or inconsistent – occur, the message "NO ERRORS" is displayed.

On completion of checking an area or the whole disk, if any read errors were encountered on sectors which are not denoted as BAD in the available table, the message:

nnn BAD SECTOR(S) NOT DENOTED IN DIRECTORY

is displayed following the summary and error count messages.

If the utility detects that it is being run on 1 megabyte floppy disk, it will give an additional message at End of Job if circumstances dictate, as follows:

If 15-30 bad sectors are found, the following message is displayed:

DISK disk-id SHOULD BE REINITIALISED SOON

If more than 31 bad sectors are found, the following message is displayed:

DISK disk-id EXCEEDS BAD SECTOR LIMIT
PLEASE POWER OFF DISK disk-id

If this message is given, the disk should not be used again.

Output messages:

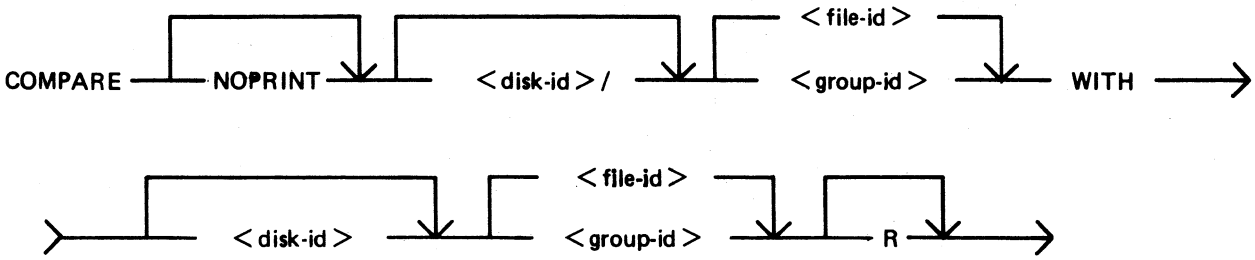
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
CHECK.DISK ON disk-name COMPLETED - ONE CONSISTENT ERROR NOTIFIED.-	This message is displayed after all sectors have been read and one error has been detected.	None.
CHECK.DISK ON disk-name COMPLETED - number ERRORS NOTIFIED	The utility has found more than one bac sector.	None.
READ ERROR ON DCL OF disk-name.	An error was encountered while reading sector zero of the specifiec disk.	None.
ERROR NOTIFIED ON READING SECTOR n	Normal output message. Utility continues.	None.
CHECK.DISK ON disk-name COMPLETED -ERRORS NOTIFIED or CHECK.DISK ON disk-name COMPLET-ED-NO ERROR	Normal EOJ messages.	None.
CHECK.DISK ON disk-name SECTOR address - NO ERROR	Only one sector specified for checking and found to be readable.	None.
CHECK.DISK ON disk-name AREA address-address COMPLETED	An area of sectors checked.	None.
CHECK.DISK ON disk-name COMPLETED	Every sector of the disk checked.	None.
SECTOR start-address BEYOND END OF DISK disk-name	The utility will continue with the next sector range specified.	None.

Note: refer to "Common Utility Output Messages" for additional messages.

COMPARE (Compare Files)

This utility compares corresponding records in two files, or in pairs of files within two groups. A realignment feature is also available as an aid to detecting missing records.

Format:



The NOPRINT option results in suppression of the full printed error listing. Instead, the following is displayed on the SPO only when the first error occurs:

FIRST DIFFERENCE FOUND BETWEEN FILES –

file-name-1 RECORD record-number AT BYTE offset
file-name-2 RECORD record-number AT BYTE offset

Examples:

To compare file PQ60R on the system disk with file PQ60RS on disk PRB3:

COMPARE PQ60R WITH PRB3/PQ60RS

To compare the groups of files beginning with AR and the files A27Q on disk ARBK1 and ARBK2:

COMPARE ARBK1/AR= WITH ARBK2/AR=,
ARBK1/A27Q WITH ARBK2/A27Q

To compare the file IV20F on the system disk with the file of the same name on disk I32, with realignment:

COMPARE IV20F WITH I32/IV20F R

If corresponding records are different, the following is printed on a line printer file (or console printer if the line printer is not available).

DIFFERENCE DETECTED AT BYTE @nnnn@

where n is the number of the byte in the record, starting from 0. The two records are then printed, using more than one line if necessary,with the following format:

byte-offset
32-byte groups in hexadecimal
32-byte groups in ASCII

(A null character (00) in hex is represented by "..", and a non-printable character in ASCII represented by a blank).

Comparison of groups of files works as in the following example:

Assume DISK1 contains the files A, B, C, D, AB, AC, ABC, BC.

Assume DISK2 contains the files A, B, C, D, AB, AC, ABC, BC, BD, EF.

Then

COMPARE DISK1/= WITH DISK2/= compares all files on DISK1 with the corresponding files on DISK2.

But

COMPARE DISK2/= WITH DISK1/= compares files on DISK2 with the corresponding files on DISK1, and will fail to find DISK1/BD and DISK1/EF.

Similarly,

COMPARE DISK1/A= WITH DISK2/A= compares files A, AB, AC and ABC on DISK1 with the corresponding files A, AB, AC and ABC on DISK1 with the corresponding files on DISK2.

Also,

COMPARE DISK1/A= WITH DISK2/AB= compares the following pair of files:

DISK1/A with DISK2/AB,

DISK1/AB with DISK2/ABB, (not found)

DISK1/AC with DISK2/ABC,

DISK1/ABC with DISK2/ABBC (not found)

The realignment option works in the following manner:

If three consecutive records fail to compare then an attempt is made to compare the third record of the second file with the next two records of the first file.

If all these five comparisons fail then an attempt is made to compare the fifth record of the first file with the fourth, fifth, sixth and seventh records from the second file.

If this comparison fails, then the comparison is terminated with an appropriate message (see later).

If a correct comparison occurs at any stage, then the compared records are used as synchronization for restarting normal comparisons.

For example, consider FILE1 containing 10 records A, B, C, D, E, F, G, H, I and J, and FILE2 containing twelve records K, L, M, N, O, P, Q, R, S, T, U, and V.

The utility compares record A with record K, then B with L, then C with M. If all these comparisons fail, then if realignment is specified record M is compared with records D and E. If this also fails, record E is compared with records N, O, P and Q. If none of these compare, the comparison is terminated.

Note that if there is a missing record in one file, and realignment is NOT specified, a comparison error will arise on every succeeding record until end-of-job.

Additional Capabilities

Further features in this utility are summarized in the railroad chart given in figure 4-2, which gives the complete input specifications.

For B 900 systems, the utility attempts to open SYSMEM on all PPIT listed units for directory scanning and searches for a PPIT entry with a tag of @20@ for the system pseudo disk-name.

Non-disk devices:

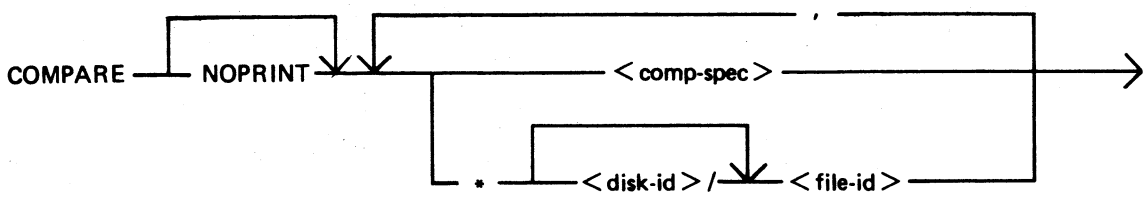
Files on devices other than disk may be compared by following the file name by one of the following keywords:

CRD – any 80-column or 96-column card device

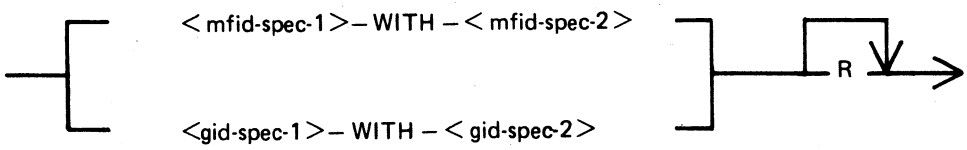
PTR – any paper tape input device

MTP – any magnetic tape or cassette device

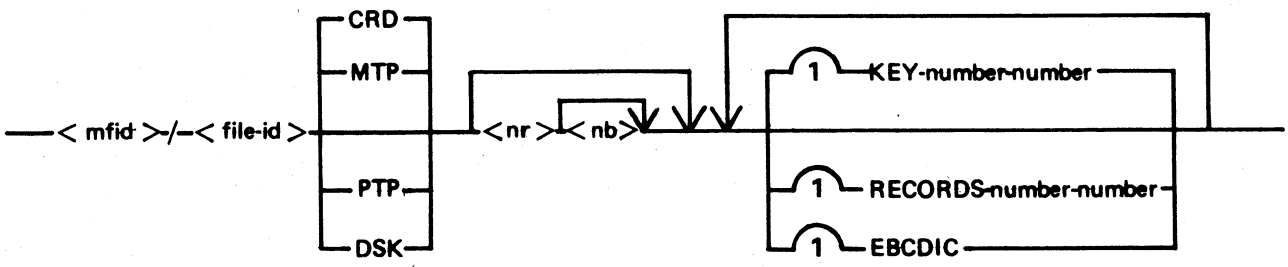
DSK – any disk device (the default; this keyword is for documentation only)



<comp-spec> is defined as :



<mfid-spec> is defined as :



<gid-spec> is defined as :

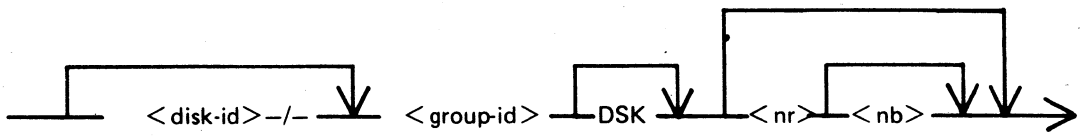


Figure 4-2. Railroad Chart for Compare Utility

Examples:

To compare records on a cassette file ARDUMP/FILE020 with a file AR578QQ on the disk WDSK:

COMPARE ARDUMP/FILE020 MTP WITH WDSK/AR578QQ

(note that the two-part name is valid for multi-file tapes or cassettes, refer to section 2 for naming conventions).

To compare two card files DAT1 and DAT2:

COMPARE DAT1 CRD WITH DAT2 CRD

Record and block sizes:

The record size (and the number of records per block) may be specified after the file name and device keyword if applicable.

Examples:

To compare a system disk file CU265 with a magnetic tape file TPF, treating data blocks on the tape as 80-byte records blocked 9 records to a block:

COMPARE CU265 DSK WITH TPF MTP 80 9

To compare a system disk file SCR01 containing 90-byte records with a system disk file SCR02 containing 180-byte records, but reblocking the second file as 90-byte records:

COMPARE SCR01 WITH SCR02 90 2

Note that if the records to be compared are of different lengths, and reblocking is not specified, then only the number of characters in the shorter record are compared.

If EBCDIC is used the file will be translated from EBCDIC on input. The option KEY allows the comparison to be done only on the field defined, the remainder of each record will be ignored. The first number is the offset of the field within the record, the second is its length. If two files have keys of different lengths, the shorter length will be assumed for both the files.

NOTE: The EBCDIC option is applicable when one of the devices is tape.

Examples

Compare fields starting at byte 11 for 4 characters of FILE1 with FILE2

COMPARE FILE1 KEY 10-4 WITH FILE2 KEY 10-4

The option RECORDS allows the comparison to be done only on the records specified. The first number is the starting record number and the second number is the total number of records available for comparison. No other record will be read from that file.

Example:

Compare records 12, 13, 14 of FILE1 with records 10, 11, 12 of FILE2.

COMPARE FILE1 RECORDS 12 3 WITH FILE2 RECORDS 10 3

Limitations:

The maximum record size is 1024 bytes. If a file exceeds this record size, it may be compared by reblocking. For example, a file with record size of 1200 can be compared by reblocking as 600 bytes blocked 2, or as 300 bytes blocked 4. The higher the blocking factor, the slower will be the comparison. (If the record size is a prime number P, it can be reblocked as 1-byte records blocked P).

The use of a star-file terminates the list of pairs of files to be compared. For example,

COMPARE A= WITH DK2/A=, X= WITH DK2/X=,
STFILE, B= WITH DK2/B=

will compare A=, X= and all files mentioned in the file STFILE, but will ignore the comparisons of B=

Output messages:

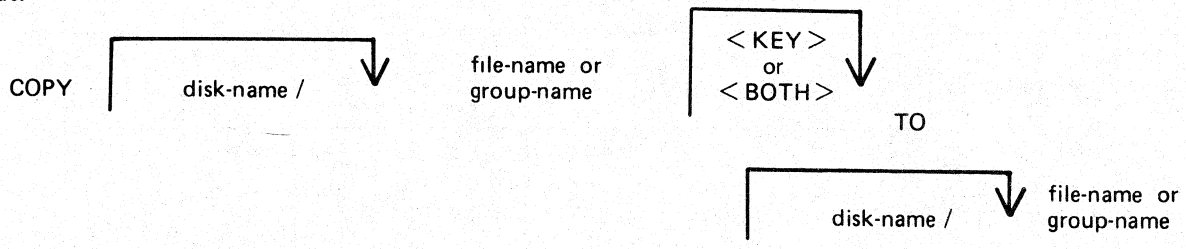
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
END OF FILE filename BEFORE filename - n ERRORS	End of one file is detected before the end of the other file	None.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
filename WITH filename COMPARED - n ERRORS	Normal ending message if both files are same size.	None.
ILLEGAL SYNTAX FOR ITEM input	Initial input mis-typed.	Check input and re-enter.
CANNOT REALIGN filename WITH filename-n ERRORS	Ending message if realignment has been specified but has failed.	None.
INCOMPATIBLE FILESPECS input	A file is specified to be compared with a group of files, or vice versa.	Re-input correct message.
ITEM TOO LONG input	Input message greater than 256 characters.	Divice input into separate parts, and re-erter.
DIFFERENCE DETECTED AT BYTE @nnnnn@	See example earlier	See example earlier
CANNOT READ RECORD n OF filename	Parity error on disk file.	Use backup copy of file concerned, if possible.
ILLEGAL KEY FOR file-name: number-number	A key has been specified with a length zero or which does not lie completely within the record. The utility will proceed with next item.	Check the key length and re-input if necessary.
ILLEGAL RECORDS FOR file-name: number	The record number specified for the starting point of comparisons is not present in the file.	None.
CANNOT COMPARE PAST POSITION number IN file-name	The utility limitation came into effect due to a request to compare beyond an offset of number bytes.	None.
file-name EXHAUSTED AT number	The file had a record range specifiec which ran beyond the end of file. The range has been truncated.	None.

COPY (File Copy)

This utility allows the operator to copy files from one medium to another.

Format:



If as a result of the copying a file a duplicate filename would be created, the original file on the destination disk is removed automatically.

If the file being copied is a keyfile and the <KEY> option is used, the keyfile is copied and the new keyfile refers to the original data file.

If the file being copied is a keyfile and the <BOTH> option is used, the keyfile and the corresponding data file are copied. The data file is given the keyfile name with the letters, "QQ" appended. The new keyfile is made to refer to the new data file name.

If the file being copied is a keyfile and neither <KEY> nor <BOTH> options are used, only the corresponding data file is copied. The records of the new data file are created in keyfile order.

Examples:

To copy a file called AR200 from the system disk to a disk called ARBU:

COPY AR200 TO ARBU/AR200

To copy files called AR200 and AR300 from the system disk to a disk called ARBU:

COPY AR200 TO ARBU/AR200 AR300 TO ARBU/AR300.

To copy a file called APTASK from the system disk to APBU, changing its name to APTASKB:

COPY APTASK TO APBU/APTASKB.

To copy all files beginning with letters "PR" from disk PR2 TO disk called PRBU:

COPY PR2/PR= TO PRBU/PR=

Copying Keyfiles

Assume there is a keyfile called PR200K which refers to a data file called PR200.

The statement

COPY PR200 <KEY> TO PRB/PR200K will create a new keyfile PR200K on disk called PRB which references the original data file, PR200, on the system disk.

The statement

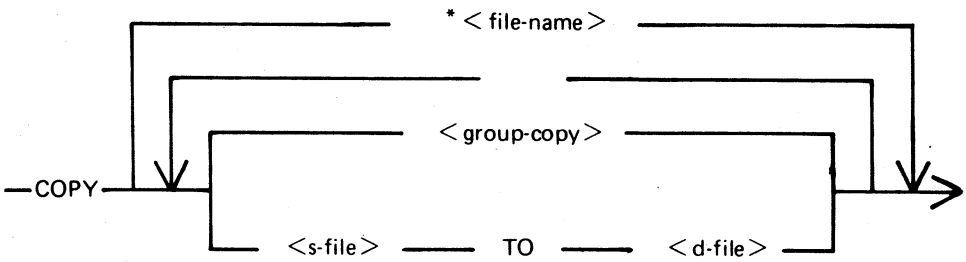
COPY PR200K <BOTH> TO PRB/PR200K will create a new keyfile and data file on disk called PRB. The name of the new data file will be PRB/PR200KQQ and the keyfile (PRB/PR200K) will refer to this new data file.

The statement

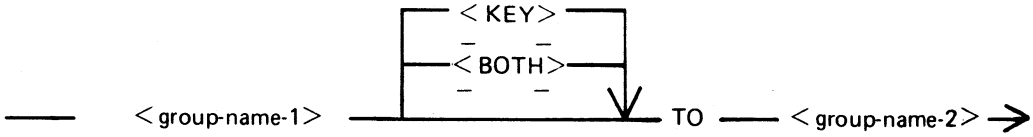
COPY PR200K TO PRB/PR200K will create a new datafile PR200K on the disk PRB. No new keyfile will be created but the records in the new data file will be created in key order according to the keyfile.

Additional Capabilities

Further features in this utility are summarized in the railroad chart given in figure 4-3, which gives the complete input specifications.



< group-copy > is defined as :



< s-file > is defined as :

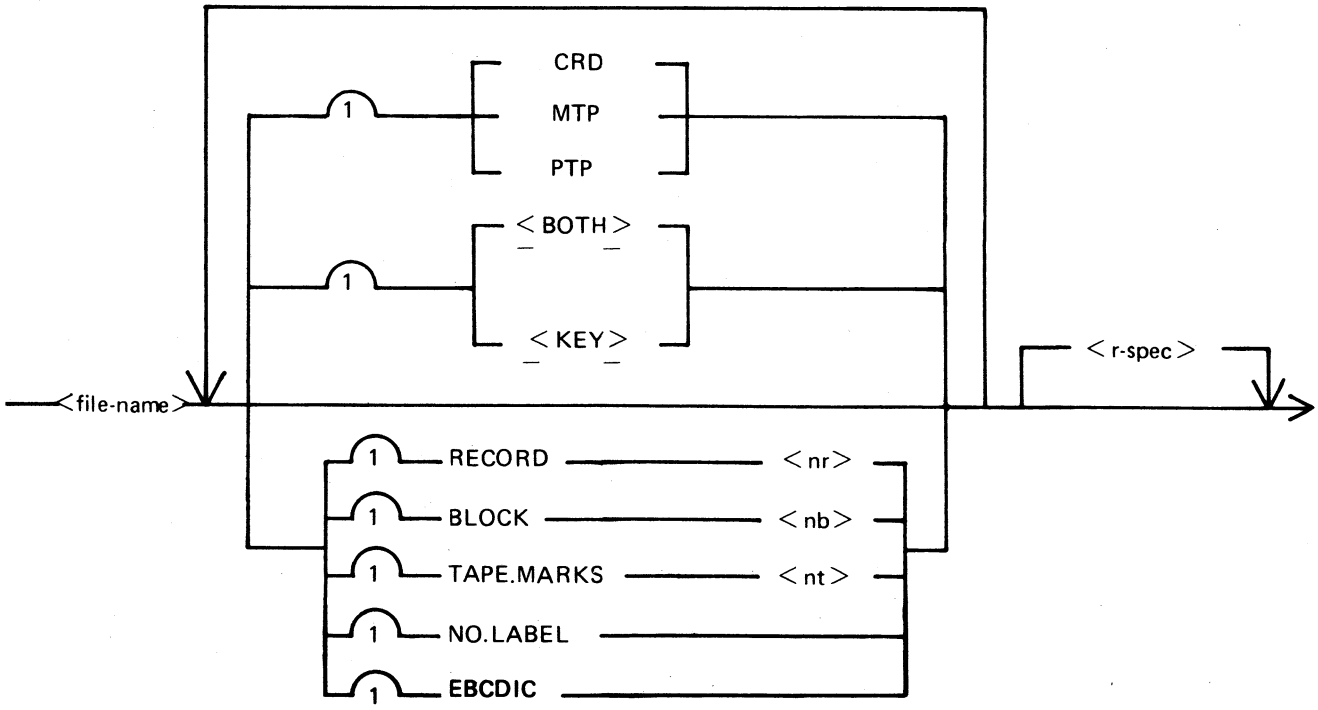
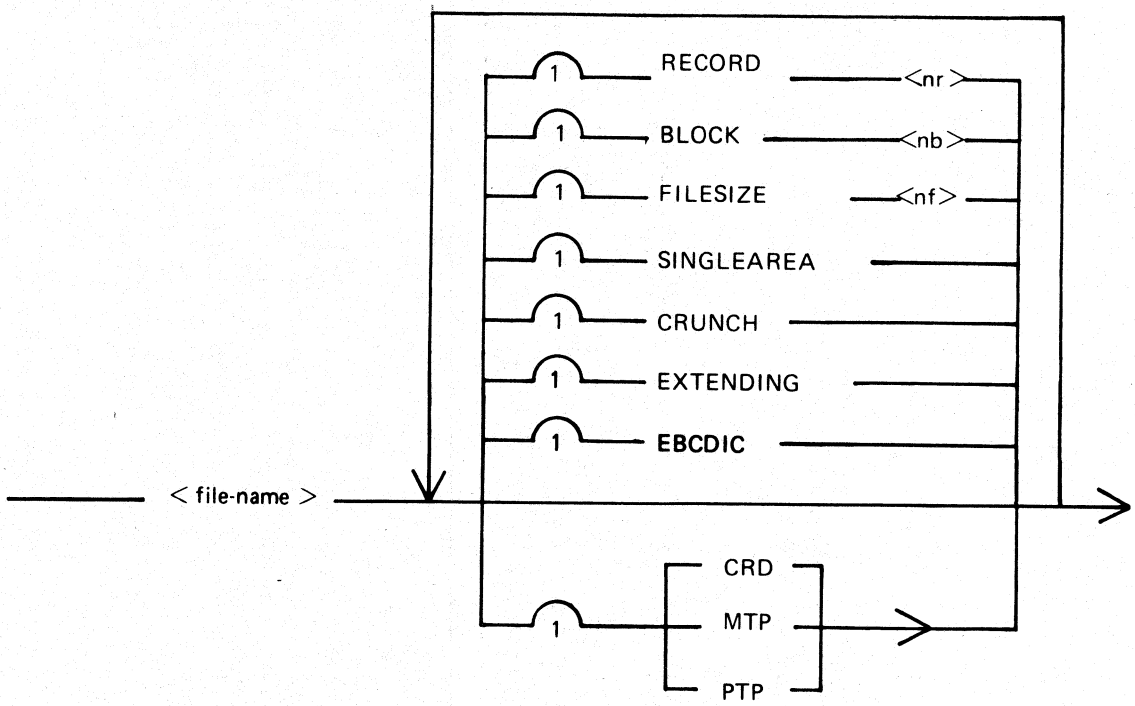


Figure 4-3. Railroad Chart for Copy Utility (Sheet 1 of 2)

< d-file > is defined as :



< r-spec> is defined as :

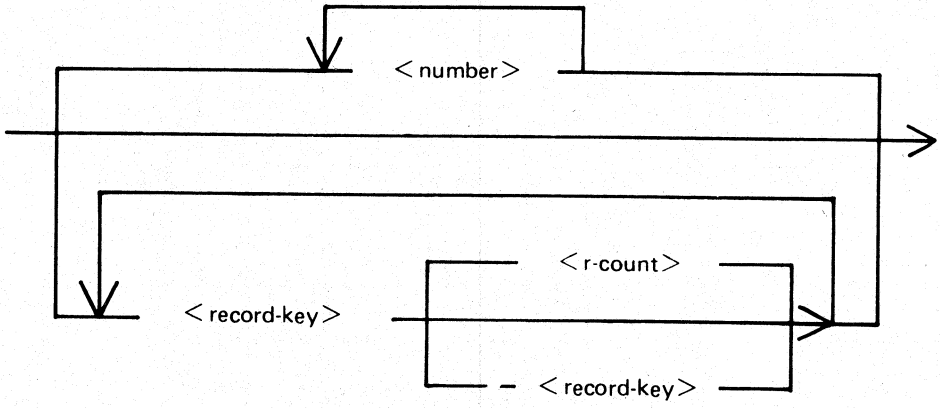


Figure 4-3. Railroad Chart for Copy Utility (Sheet 2 of 2).

NOTE

The facility to copy from or to an EBCDIC file is not supported for disk files.

Non-disk devices

Files may be copied to and from media other than disks. Abbreviation for the valid devices are as follows:

MTP – magnetic tape or cassette

CRD – punched card

PTP – paper tape

Examples:

To copy a cardfile called PRFILE to a disk called PRBU:

COPY PRFILE CRD TO PRBU/PRFILE

To copy a disk file called PR300 to a single-file magnetic tape:

COPY PR300 TO PRTAPE MTP

NOTE

This tape is in "COPY" tape format, not "LOAD/DUMP" format. To access this tape file again it will have to be placed on appropriate device by "COPY" utility, not "' LOAD/DUMP".

To copy a cardfile called PRFILE to paper tape:

COPY PRFILE CRD TO PTFILE PTP

Note: Paper tapes are always "unlabelled", and when accessing it, MCP will issue appropriate message requiring an "AD" intrinsic response from operator. See "AD" intrinsic.

Unlabelled tapes

Input tapes having no CMS labels ("unlabelled" tapes) may be accessed by the COPY utility.

The NO.LABEL option allows the copying of unlabelled files. Upon recognizing an unlabelled file, the MCP will print a "DEVICE REQUIRED" message. The operator must then respond with an appropriate "AD" input message (see "AD") to identify the unlabelled file.

The end of file recognition for unlabelled files is determined by tapemark count. The TAPE.MARKS option allows the operator to specify the total number of tapemarks 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 contribute to this total. Therefore, a standard labelled CMS file will be copied up to, but excluding, the trailing label if NO.LABEL is specified by itself.(The standard CMS labelled tape format is "label, tape mark, data, tape mark, label", see CMS MCP manual). The operator must be aware of the format of any file which is to be copied when using the NO.LABEL option.

If the RECORD size is not 180 bytes, refer to the section on Record/Block modifications.

Example:

To create a disk file called EMPL from first file of a magnetic tape with non-standard label (the format being: LABEL, TAPEMARK, DATA, TAPEMARK):

COPY TP MTP NO.LABEL TAPE.MARKS 2 TO EMPL

Note: MCP will issue a message asking for unlabelled tape TP. Operator must respond with "AD" input. Additionally, the first record of file EMPL will contain a copy of the non-standard label.

Record and block sizes

Record and/or Block sizes may be modified for all file types, input and output.

The number of bytes in the record or block is specified using the corresponding "numbers". The record and block sizes of input disk files are always taken from the file itself (**Disk** File Header). Record and block sizes of non-disk input files are determined as follows:

Record Size:

If RECORD is specified, "number" becomes the new record size.

If RECORD is not specified record size defaults (see below).

Block size:

If BLOCK is specified, "number" becomes the new block size.

If no BLOCK specified, but RECORD is specified, record size becomes new BLOCK size.

If neither BLOCK nor RECORD is specified, Block Size defaults (see below).

Default Values:

Output disk = same as input disk.

Input labelled tape/cassette = from tape label

Input unlabelled tape/cassette = 180 bytes

Cards = 80 or 96 bytes, depending on device.

If the record size is increased then the additional bytes will be filled with spaces if the input file is a source or data file, or with binary zeros for any other type of file.

Example

To copy an 80-column card file labelled PROGSRC to a disk file called PROGSRC on a user disk "USR", and make the record size and block size of the disk file 80 bytes and 720 bytes respectively:

```
COPY PROGSRC CRD TO USR/PROGSRC RECORD 80 BLOCK 720.
```

To copy a disk file PRBU/PR300 to magnetic tape **with** large blocks suitable for tape media:

```
COPY PRBU/PR300 TO PRTAPE MTP RECORD 180 BLOCK 1800
```

File size

The "FILESIZE attribute" of a disk file may be specified for the output disk file. Note that only assigned areas are copied. This feature does not increase disk space at the time of copying, but allows programs to add further records if required. At that time extra disk space may be needed.

Example:

To copy FILE1 and increase its "FILESIZE" to 1500, replacing the original by the copy:

```
COPY FILE1 TO FILE1 FILESIZE 1500
```

Single area

The "SINGLEAREA attribute" may be specified for the output disk file. This ensures that the new file will occupy a single disk area.

Example

```
COPY FFLE2 TO FILE2 SINGLEAREA
```

Crunching files

The "CRUNCH attribute" may be specified for the output file. This causes any unused disk space at the end of the file to be returned to the system.

Example:

COPY PRB/PR200 TO PRB78/PR200 CRUNCH

WARNING

A file cannot be "uncrunched" once it is crunched. This means it cannot be extended. It can only be used for inquiry. This option is therefore useful for storing history files.

Extending disk files

Records can be added to the end of an existing disk file with the option "EXTENDING". The existing file must have identical attributes to the file being copied.

Example:

A data file called DFTUES was created with Tuesday's data. To add this data to the end of a file called DFMON (containing Monday's data):

COPY DFTUES TO DFMON EXTENDING

(Note the size of DFMON must be large enough to contain all required records.)

Selected file copy

Selected record numbers from the input file may be copied.

Example:

To copy 500 records starting at record #1200 from file FILE1 to file FILE2:

COPY FILE1 1200 500 TO FILE2

Note:

pairs of numbers may be specified within each pair; the first number specifies a relative record number and the second specifies number of records to be copied. If an extra number is specified, the last number specifies copying from that record to the end of the file.

Example:

To copy records 100 to 149, 300 to 499, and 1000 to end of file:

COPY FILE1 100 50 300 200 1000 TO FILE2

Selected index file copy

For indexed files, copying of records can be selected based on content of the key. There are 2 options: the number of records can be specified, or an ending key value.

Examples:

PQR is a keyfile containing personnel records. To copy 15 records from the corresponding data file starting from the record with personnel #01786 to a data file, PSNL:

COPY PQR 01786 15 TO PSNL

Using same keyfile, to copy all data records from personnel #01786 to 18000 to data file, PSNL:

COPY PQR 01786 - 18000 TO PSNL

Note:

The second option is specified by the hyphen in the COPY statement. Note that at least on space is required before and after all key values (personnel # in this case).

Save Factor

New magnetic tapes are given a save factor of 999.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
number, number IN file name NOT COPIED	Error found in pair of numbers for selected record copy option. One record number in the pair incicates a section of the file address than a previously specified section. This pair is ignored by COPY.	Check input and re-erter.
file-name EXHAUSTED	End-of-file was encountered while the section of the file indicated was being copied.	None
filename TO filename BAD ATTRIBUTES	Particular attribute specifications is meaningless or inconsistent. The inconsistencies will be in relationship between output device, record size, and block size; or attempt was made to add records to a file whose attributes differ from those of input file.	Check input and re-enter.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FILE IDENTIFIER file-name TOO LONG	Copy of a group of files was requested. At least one file-name that was to be generated by the copy would have been more than 12 characters long, maximum length of file-names is 12 characters.	Check input and re-enter.
filename TO filename COPY DISCREPANCY	End-of-file reached before expected. Disk File Header may be corrupt.	
filename TO filename COPIED	COPY successful.	None.
FAMILY family-name-a TO family-name-b COPIED	COPY successful.	None.
filename TO filename ILLEGAL REQUEST	Copy of a group of files was requested. An individual file-name of "SYSMEM" or "(spaces)" was to have been produced. "SYSMEM" (a file-name reserved for system use only) and "(spaces)" are not acceptable file-names.	Check input and re-enter.
filename NOT ACCEPTABLE RECORD SIZE OF m EXCEEDS MAXIMUM FOR THIS RUN - RESUBMIT	Copy has encountered a file with a record size greater than expected. This can happen if a magnetic tape file with a record size greater than 1024 characters is submitted to the utility without the record size being properly specified in the initial input.	Check input and re-enter.
filename EXHAUSTED DURING RANGE record-key number number	End-of-file encountered while section of file indicated by "record-key" "number" is being copied.	None

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
filename EXHAUSTED DURING RANGE "record-key" "record-key"	No records were found in the range "record-key" "record-key".	Check input and re-enter.
NO RECORDS FOR COPYING FROM filename	Specified file contains no records for copying.	Check for correct filename.
INPUT RECORD number OUTPUT RECORD number PERMANENT ERROR ON INPUT FILE	Error encountered-utility cannot continue.	Do selective copies of the parts of the file before and after the bad record, as part recovery
INPUT RECORD number OUTPUT RECORD number PERMANENT ERROR ON OUTPUT FILE	Error encountered-utility cannot continue.	Remove the disk area in error with XD utility and rerun the COPY
INPUT RECORD number OUTPUT RECORD number OUTPUT FILE TOO SMALL	Error encountered-utility cannot continue.	Re-input the COPY with suitable FILESIZE option.
filename TO file-name COPY FAILURE	COPY unsuccessful irr-ecoverable error was encountered. The reason should be displayed before this message.	None.
NO RECORDS IN THE KEYFILE - file-name	<BOTH> option was used. Utility was not able to access to a data file through some failure in the keyfile.	None.
file-name NOT FOUND FOR EXTENDING	Specified file was not found.	Check input and re-enter if necessary; Check for correct medium;
filename REMOVED	COPY successful. If any duplicate files are encountered by COPY they are aut-omatically removed.	None.
filename TO filename SELECTION CRITERIA IGNORED	Confirms that select-ion criteria were specified when copy-ing a pair of files, and have been discard-ed.	

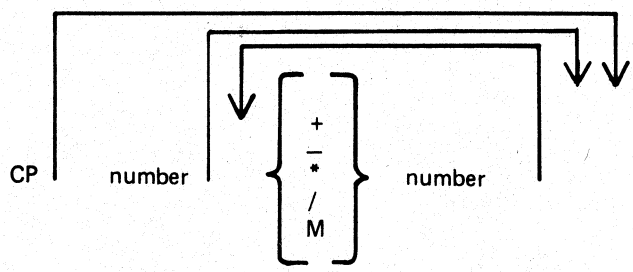
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
BAD ATTRIBUTES SPECIFIED	Inconsistent attributes were specified for input file.	Check input and re-enter.
filename TO filename EXTENDING FLAG IGNORED	EXTENDING option was ignored by utlity when pair of files was copied.	

CP (Compute)

This utility allows simple computations to be made, with the answer displayed in decimal and hexadecimal. Input may be either decimal or hexadecimal. Hex values must be enclosed in @ symbols.

The utility may be initiated with a single computation to perform, in which case it will do the calculation and terminate. If no calculation is initially provided, the utility issues an ACCEPT to enable the computation to be entered. In this case the utility will do the calculation and then issue further ACCEPTS until a null input to the ACCEPT is given.

Format:



The numbers accepted are any decimal or hex values in the range

0 999999999999999[@38D7EA4C67FFF@]

or the negative equivalent.

Parentheses are not allowed. The calculation is performed on a strictly left-to-right basis. The operators +, −, *, /, and M are for addition, subtraction, multiplication, division and modulus division (the result is the remainder) respectively.

Examples:

To compute the hexadecimal value of the decimal number 12345:

CP 12345

To compute the value of a complex expression:

CP 555 * 3 + 2-100 *2/5

(Note: the result of the above is 626, because the calculation is done strictly left-to-right).

Output messages:

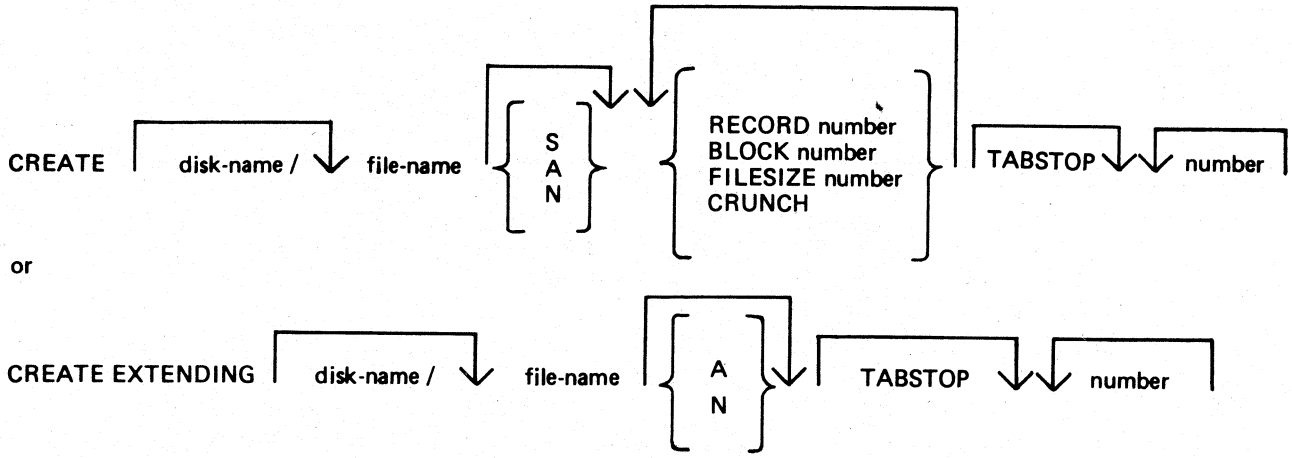
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
CP : NUMBER TOO LARGE	At least one number is out of range.	Re-input.
CP : MISSING OPERAND	Two consecutive operators (+, etc) have been entered with no number between.	Re-input.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
CP : HEX. NO. WITH MISSING "a"	Illegal symbol; or missing a symbol at front or back of a hex value.	Re-input.
CP : INVALID OPERATOR	Illegal symbol; or an operator (+, etc) has been omitted.	Re-input.
CP : OVERFLOW	An intermediate value in the computation is greater than the maximum value.	Re-input.
CP : DIVISION BY ZERO	Self explanatory.	Re-input.
CP : ILLEGAL NUMBER	Self explanatory.	Re-input.
CP : NUMBER TOO LARGE	Self explanatory.	Re-input.
CP : MISSING NUMBER	Self explanatory.	Re-input.
CP : ILLEGAL HEX NUMBER	Self explanatory.	Re-input.

CREATE (Create Disk File)

This utility allows the operator to create or extend data or source disk files. The "AMEND" and "UPDATE" utilities use many similar features.

Format:



When creating a new disk file, certain attributes may be specified.

If the S option is selected, a source file will be created using alphanumeric input. If the A option is used a data file will be created using alphanumeric input. The N option creates a data file with hexadecimal (numeric) input. If none of these is selected, S is assumed. Alphanumeric input is accepted as typed, but numeric (hexadecimal) input requires two characters (0-9, A-F) for each byte of the record.

The RECORD option allows the operator to specify the number of characters per record of the new file. If no record size is specified, a record size of 80 bytes is assumed for source files, and 180 bytes for data files.

The BLOCK option allows the number of characters per block of a new file to be defined. The defaults are as follows:

If RECORD size was specified but no BLOCK, BLOCK size will equal RECORD size.

If neither RECORD nor BLOCK is specified, RECORD size will be 80 bytes for source files and 180 bytes for data files; BLOCK size will be 160 bytes for source files, 180 bytes for data files.

The FILESIZE option allows the maximum number of records likely to be written to the new file. This is useful in allocating only as much disk space as required by the file. Once the FILESIZE has been specified for a file, that file can never be extended beyond that number of records. However, the COPY utility may be used for increasing the FILESIZE of an existing file. The default is 2,048 records.

The CRUNCH option allows the operator to specify that the new file should occupy the minimum area of disk, but never be extended.

The numbers specified for the "numbers" option may be used to set "tab" positions within the record (similar to setting "tabs" on a typewriter). If tabs are set, the operator may input data, press OCK1, and the utility will reposition the print mechanism to the next tab position within the record, and await data input. During this repositioning CREATE will fill all character positions left unspecified in the record with a "filler" (ASCII space for source input, ASCII zero for alphanumeric input, and binary zero for numeric zero). The record length plus one will be used as a termination tab position, whether or not other tab positions are specified.

CREATE 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). A file of 180 byte records requiring hexadecimal input will require a minimum of three tab positions (for instance at positions 50, 100 and 150). The maximum tab size is 111 in alphanumeric input and 54 in hexadecimal input. That is, the difference between two consecutive tab positions should be less than or equal to 111 in alphanumeric input and less than or equal to 54 in hexadecimal input.

By specification of TABSTOP in the initiating message, CREATE sets up tab positions coinciding with the end of the console line as well as any other tabs specified. Also, when in Record Input Mode (PK1), CREATE accepts information only up to the next tab position.

Default tab positions have been chosen to allow a maximum number of characters to be inserted on one line. CREATE uses nine for the record number and 110 for the contents of the record. In addition, manually-selected tabs may still be used.

Default tab positions for the end of the console lines for CREATE are as follows:

Source or data alphanumeric : 111 221 331 441
Data hexadecimal : 56 111 166 221 276 331 386 441 496

Examples:

	Tabs Set at:
CREATE FILEA RECORD 410 TABSTOP	111 221 331
CREATE FILEB A RECORD 500 TABSTOP 51 61 221	51 61 111 221 331 441
CREATE FILEC N RECORD 450 TABSTOP	56 111 166 221 276 331 386 441 496

The EXTENDING option is used to add records to an existing file. The attributes, such as RECORD and BLOCK sizes, are taken from the old file. The file type is also taken from the existing file. The operator may specify "A" for alphanumeric input or "N" for hexadecimal input. If neither "A" nor "N" is specified, "A" is assumed. If a file is CREATE EXTENDed, the generation number of the file is incremented by one.

Examples:

To create a source file called "ICFILE", record size 100 bytes with 5 records per block, tab position set at 65:

CREATE ICFILE RECORD 100 BLOCK 500 65.

To create a source file called "ICFILE" with Record Size 80, block 3, and a maximum of 20 records in the file:

CREATE ICFILE RECORD 80 BLOCK 240 FILESIZE 20.

To extend a source file called "ICFILE" (note: the utility will automatically prompt the operator for the next sequential record number to be created):

CREATE EXTENDING ICFILE

To create a data file called "CFILE" for hexadecimal input with tab positions set at 50, 100 and 150. (Note: Default record size is 180, block 1):

CREATE CFILE N 50 100 150

The utility operates in two modes: "RECORD INPUT" (entered through PK1) and "RECORD MODIFY" (entered through PK2).

PK1	PK2	PK3	PK4	PK5	PK6
input	modify	—	—	—	EOJ

An OCK3 option is included, to display the current tab position.

An OCK4 "help" option is provided, which will output the above options when pressed. In order to show which mode the utility was in when OCK4 was pressed, an asterisk (*) is printed next to that mode on the Help display.

The "Record Input Mode" (PK1) is used to enter new records through the keyboard. Characters are input followed by OCK1 for each TAB position.

The "Record Modify Mode" (PK2) is used to make corrections to the last record input. The point in the record at which alterations are to be made is selected by typing an identifying group of characters immediately preceding the byte(s) of the record to be altered. The portion of the record to be replaced or inserted follows the identifying characters, delimited by a colon (:). If alterations are to be made at the beginning of the record, no identifying characters are necessary. A starting byte position for the identifying character string search may be specified in the console input (see AMEND for details).

If OCK1 is used to terminate input, the characters to be altered will replace the corresponding number of characters in the record.

For example, for a record containing "ABCDEF", the amendment C:XY:OCK1 will result in "ABCXYF".

If OCK2 is used to terminate input, the characters delimited by colons (:) will be inserted at the indicated point. 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, a record specified with tab positions at 4 and 8, contains "ABCDEFGHIJ". The amendment C:WXY: OCK2 will result in "ABCWXYDHIJ".

Initially the utility will be in the "Record Input Mode", and on completion of an entry in any mode, it will allow the operator to select the mode not in use, or terminate the utility (with PK6). Unless otherwise instructed, it will continue in the existing mode.

If the FILESIZE is specified and records are entered beyond the given filesize, then the error message is displayed after (filesize + 2) records have been entered. The last two records will not be written, due to the blocking of the output file by CREATE. For example, if the request

CREATE ICFILE FILESIZE 15

is followed by more than 15 records entered, then the message 'OUTPUT FILE TOO SMALL" will be given after the seventeenth record is entered; and the utility will go to EOJ without writing records 16 and 17 to the output file.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FILETYPE IS NOT SOURCE OR DATA	A file other than a "source" file or a "data" file was specified.	Check for correct file; check input and re-enter if necessary.
ILLEGAL PARAMETER LIST - BAD ATTRIBUTE	Specified record and block sizes are incompatible.	Check input and re-enter;

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ILLEGAL PARAMETER LIST - TABS ERROR	Tab positions beyond end of record were specified; or input fields larger than capabilities of console.	Check input and re-enter.
NOT HEXADECIMAL CHARACTER INPUT - RESUBMIT	Character other than 0-9 ad A-F was input.	Check input and re-enter.
ODD NUMBER OF HEXADECIMAL CHARACTERS INPUT	Warning message indicating that an odd number of hexadecimal characters was input. When input mode is "hexadecimal", utility expects even number of input characters.	None. Utility accepts the input, but adds a zero onto the end (right) of the input, to ever it out.
INPUT ERROR - RESUBMIT RECORD MODIFICATION	Input error during "Record Modify Mode".	Check input and re-enter record modification.
BYTE WITHIN RECORD SPECIFIED NOT FOUND	The identifying string of characters for record modification could not be found in the record specified.	Check input and re-enter the record modification.
UNWANTED KEY PRESSED - PLEASE RE-INPUT.	Invalid OCK was pressed.	Re-enter input and terminate the entry with the correct OCK.
INPUT IMMEDIATELY BEFORE PK6 HAS BEEN LOST	Characters were input immediately before PK6 was pressed to terminate the utility. These characters will not be written to the specified file.	Restart the utility using modify mode to correct this record if desired.
OUTPUT FILE TOO SMALL	Attempt was made to add records to specified file beyond its maximum filesize. File is closed with lock.	List the file to check which records have been entered: then use CREATE EXTENDING to add the desired records. Alternatively use UPDATE with the FILESIZE option.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PERMANENT ERROR ON OUTPUT FILE	Utility is not able to write any more records to the new disk file because of errors on the disk. File is closed with lock.	List the file to check which records have been entered: then COPY to a different file and continue using CREATE EXTENDING.
RECORD SELECTION ERROR	Attempt was made to select a non-existant record.	Check input and re-enter if necessary.
ILLEGAL RECORD NUMBER SPECIFIED	Attempt was made to select record greater than the number of records in file, or zero.	Check input and re-enter if necessary.
RECORD REQUESTED IS BEYOND E.O.F.	Attempt was made to select record beyond end of file.	Usually indicates update is complete: utility terminates normally, no action need be taken.
E.O.F. REACHED DURING DELETIONS	Attempt was made to delete a record beyond end of file.	Usually indicates file update is complete: utility terminates normally, no action need be taken.
FILE TYPE IS NOT DATA	Input type A or N has been specified and the type of file given for extension is not data.	Check the file type and re-input correct file type.
END OF OUTPUT FILE REACHED	The last logical record of the output file has been stored. (applies to the utility "UPDATE").	None.

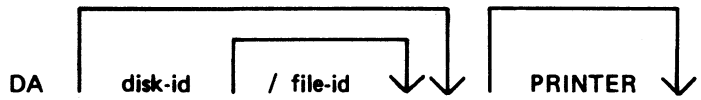
Note: Refer to "Common Utility Output messages" for additional messages

DA (Disk Analysis)

This utility allows the operator to read any portion of a CMS-format disk. It is an interactive program, with the operator entering a range of commands via the console (if the PRINTER option is not specified), or the SPO by means of an accept (AX) command if PRINTER is specified. Output is directed to a line printer when the PRINTER option is invoked.

The utility is commonly used to print the contents of the disk directory. In general, if the PD utility operates correctly for a specific disk, then DA should also run successfully. Specifically, the disk cartridge label, the name list entry and disk file header for SYSMEM must be intact. (Refer to the CMS MCP manual for details of the disk format and directory structures).

Format:



The utility operates in two modes "disk mode" and "file mode". If no file name is specified the utility operates in "disk mode". If no disk-name is specified, the system disk is analyzed.

Disk Mode

In this mode the operator can enter via the keyboard a number of commands. These commands can be abbreviated according to the table provided at the end of this section. The format and meaning of each command in disk mode is given below.

END

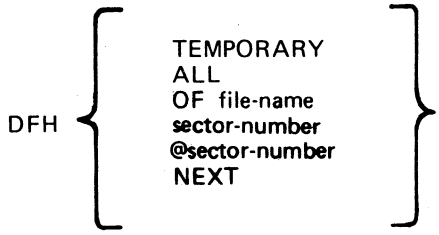
Terminates the utility.

LABEL

Reads and formats the contents of the disk cartridge label.

DFH

Reads and formats the contents of selected disk file headers. This command is followed by other details, given here:



The "TEMPORARY" option displays the headers of all temporary files.

The "ALL" option displays the headers of all files, and their contents if in use.

The "OF" option displays the header of the specified file: all headers will be checked and duplicates will be displayed if found.

The "sector-number" option displays any sector in disk file header format, where the number is a decimal sector address. If preceded by an @ symbol, the sector-number is in hexadecimal. This feature can be used after displaying other parts of the directory, which include sector addresses for disk file headers in hexadecimal.

The "NEXT" option displays the header of the next file in the directory.

AVAIL.TABLE

Reads and formats the contents of selected parts of the disk available space table. This command is followed by other details, given here:

```
                 ALL
                 AVAILABLE
AVAIL.TABLE {    BAD          }
                 INVALID
                 number
                 @ number
```

The "ALL" option displays the entire available table.

The "AVAILABLE" option displays entries for available area only.

The "BAD" option displays entries for bad sectors only.

The "INVALID" option displays all entries in the available table which are invalid, in that the "length" entry does not equal the difference between "start address" and "end address".

The "sector-number" option displays any sector in available-table format, where the number is a decimal sector address is in hexadecimal.

NAME.LIST

Reads and formats the directory name list, including the sector addresses of associated disk file headers. This command may be followed by other details, given here:

```
                 NAMES
                 TEMPORARY
NAME.LIST   {    OF file-name }
                 number
                 @ number
```

If no further details are given, then the entire name list is displayed.

The "NAMES" option displays entries for old (existing) files only.

The "TEMPORARY" option displays entries for temporary files only.

The "OF" option displays the entry for the specified file : all entries will be checked and duplicates will be displayed if found.

The "sector-number" option displays any sector in name-list format, where the number is a decimal sector address. If preceded by an @ symbol, the sector-number is in hexadecimal.

Format headings for Name.List available table or Pseudo Pack Identification Table will not be printed if no entries have been found relating to the requested option.

READ

Reads and displays the contents of any specified sector in hexadecimal and ASCII format. This command may be followed by other details, given here:

```
                 NEXT
READ   ONLY  {   number      }
                 @ number
```

The "ONLY" option inhibits the display of the information.

The "NEXT" option will read the next sector. Note that after some operations which involve a search, the "next" sector may be indeterminate. After a READ of sector n, a READ NEXT will read sector n+1. A READ command with no further details is taken as a READ NEXT.

The "number" option reads the sector whose address is the number. If preceded by an – symbol, the sector-number is in hexadecimal.

DISPLAY

Displays the current sector address and contents of the program's sector-buffer. The first DISPLAY command must be preceded by a READ command. A READ ONLY followed by a DISPLAY is equivalent to a normal READ.

KFPB

The Key File Parameter Block of a keyfile can be printed.

KFPB OF file-name

If the file specified is a key file, then record 1 of that file is selected and the information is printed.

PPIT

The Pseudo Pack Identification Table can be found from the Disk Cartridge Label and requested information may be printed:

```
PPIT { number
       NAMES
       ALL
       OF disk-name }
```

The "number" option displays the sectors from the PPIT.

The "NAMES" option displays the used entries of the PPIT.

The "ALL" option displays every sector of the PPIT.

The "OF disk-name" option displays the entry for that disk only.

File Mode

In file mode, the utility can be used to read selected records. Only the following commands are valid:

READ
DISPLAY
END

The READ command has the same format as in disk mode, except that the "number" refers to the logical record number in the file, and a READ NEXT will read the next logical record in the file. The amount of information displayed is equal to the file's record length. In DISPLAY option, if the file is a key file, then the KFPB of that key file is displayed.

General Notes

In disk mode the sector number starts from zero; that is, "READ 0" will read the first disk sector.

In file mode the record number starts from one; that is, "READ 1" will read the first logical record.

Any I/O error causes the "fetch value" to be displayed, with the current sector address if in disk mode, or current record number if in file mode.

Abbreviations

For ease of use, input commands and other keywords may be abbreviated, as in the following table:

READ	R
ONLY	O
NEXT	N
DISPLAY	D
END	E
LABEL	L
DFH	DF
ALL	AL
OF	OF
TEMPORARY	T
AVAIL.TABLE	A
AVAILABLE	A
BAD	B
INVALID	I
NAME.LIST	N
NAMES	NA

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
OUTWITH format AREA ADDRESS: number LENGTH number	In disk mode the sector which has been specified to be read in DFH, NAME LIST, AVAILABLE TABLE or PPIT format is outwith relevant area.	None.
FILE IS NOT A KEYFILE	The KFPB option has been requested and the file which was specified is not a keyfile.	None.
NOT FIXED DISK SYSTEM	The PPIT option has been requested or a non-Fixed Disk system and no Pseudo Pack Identification Table exists.	None.
DA TERMINATED	The END command is successful.	None.
ILLEGAL PARAMETER input	No valid disk name of file name has been specified in the initiating message.	Re-enter
ILLEGAL PARAMETER character-string	More than one option has been specified.	Re-enter.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SUCCESSFUL READ REQUIRED FIRST	A DISPLAY command has been entered before a successful read.	Enter a READ command, then a DISPLAY will work.
NOT VALID IN FILE MODE	A command other than READ or DISPLAY has been used in file mode.	Check input.
NONE FOUND or NOT FOUND	No information has been found in answer to a command.	None.
READ ERROR ON DCL OF disk-name	An error was encountered while reading sector zero of the specified disk.	None.
PARAMETER REQUIRED	The option which has been specified is incomplete.	Re-input the complete option.
LAST OPTION UNSUCCESSFUL	The DISPLAY option has been requested before a sector or record has been read or after an option has been unsuccessful.	Perform a read before using display option.
ERROR ON READ-RECORD:----	The disk parity error has been encountered while reading a record in file mode.	None.
ERROR ON READ-ADDRESS:----	The disk parity error has been encountered while reading a record in disk mode.	None.
BEYOND END OF FILE-RECORD:----	A record beyond end of file has been requested to be read while in file mode.	None.
INVALID SECTOR-ADDRESS:----	A sector address beyond the last physical sector of a disk has been specified to be read while in disk mode.	None.
MTR MAY BE RUN ON THIS DISK	The disk has sectors reserved in the available table for MTR purposes.	None.

DCR (Disk Confidence Routine)

The DCR utility is used to establish a level of confidence in the ability of the CMS system hardware/software interface to service disk I/O. In doing so, it exercises the system as fully as possible to provide as much information as possible and practicable about faults detected. With respect to hardware, DCR is designed to accommodate any CMS disk, and attempts to maximize disk head movement, demands on the common electronics module, and demands on the directory search hardware.

The DCR utility includes five separate tests. These are:

READ.ALL
OPEN.FILES
READ.CYLINDERS
WRITE.INDEX
READ.RANDOM

All, or a subset of these, may be selected by the user in a given run of the utility. Integrity of files already existing on disks being tested is maintained: except for one test, all I/O operations are read only, and all files created by the utility are closed and removed at or before EOJ.

The utility can be automatically invoked as an option within the SCR utility (System Confidence Routine).

For any run of DCR, one or two disks may be used. The use of two disks is intended for dual disk drives and thus maximizes demands on the common electronics module. Both disks (when two are used) must be of the same capacity.

READ.ALL reads all sectors on a disk, maximizing head movement by reading alternately from low-numbered and high-numbered sectors. The user may place greatest stress on the head by specifying that a system buffer with the capacity of one record be used.

OPEN.FILES stresses the directory search hardware by opening and closing, without intervening I/O operations, up to twenty files on a disk.

READ.CYLINDERS reads one sector from every cylinder on a disk, addressing sectors in such a way as to maximize head movement while ensuring that sectors are read from all tracks (all heads are activated).

WRITE.INDEX stresses the directory search hardware by writing an index file (of a length specified by the user) to disk, and then reading the file back again, checking the data records for accuracy.

NOTE

This test requires a high proportion of execution time (approximately two minutes for a single file of 25 records; approximately thirty minutes for a single file of 500 records).

READ.RANDOM stresses the head movement but under conditions more closely approximating those of a real-life environment than those of test 1, in that records to be read are randomly selected. Also, unlike test 1, the duration of the test is not dependent on disk size, as the user specifies the number of records to be read.

After initiating the Disk Confidence Routine (DCR), prompts are displayed and the replies to these are expected via an ACCEPT message (see section 3 of this manual, AX input).

Where an asterisk (*) appears in the following operating instructions, the message:

"DCR TERMINATED BY THE USER"

will be displayed and the utility will terminate.

Operating Instructions

Enter: DCR

PROMPT 1 is displayed:

"ENTER DISK.ID TO CONTINUE.
ENTER SPACE TO TERMINATE."

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	*	Utility has been terminated.	Restart DCR.
AX <mix-no> <disk-name>	DISK <disk-name> NOT AVAILABLE <PROMPT 1>	The MCP has been unable to open <disk-name>	Make <disk-name> available and re-input.
	READ ERROR ON <disk-name> LABEL <PROMPT 1>	MCP input error on reading the disk label.	Replace disk or try another disk drive.
	READ ERROR ON <disk-name> AVAILABLE TABLE. <disk-name> AVAIL. TABLE WILL NOT BE CHECKED.	MCP input error on reading Available Table.	
	<PROMPT 2>	<disk-name> has been opened successfully and disk-label has been read.	Proceed to the instructions for PROMPT 2.

PROMPT 2:

"SECOND DISK TO BE TESTED? <Y or N>"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <neither Y nor N>	REPLY NOT "Y" OR "N" <PROMPT 2>	Reply not "Y" or "N".	Enter Y or N.
AX <mix-no> N	<PROMPT 3>	No second disk was specified.	Continue, or DS the utility.
AX <mix-no> Y	<PROMPT 1>	Second disk is to be tested.	Enter disk-name or space.

If the answer to prompt 2 is "Y" (denoting that a second disk is to be tested), prompt 1 is displayed. Refer to the following table for possible replies.

Possible replies if a second disk has been requested:

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	*	DCR has been terminated.	Restart the Utility if desired.
AX <mix-no> <disk-name>	DISK <disk-name> NOT AVAILABLE <PROMPT 1>	The MCP has found an error on opening <disk-name>.	Make <disk-name> available and re-input.
	READ ERROR ON <disk-name> LABEL <PROMPT 1>	MCP input error on reading the disk label.	Replace disk or try another disk drive.
	CAPACITIES ON 2ND DISK NOT SAME AS ON 1ST <PROMPT 1>	Indicates inequalities in the sizes of the SYSMEM files or in the number of sectors per buffer between the two disks.	Disk identifiers must be re-specified and the disk-name re-entered.
	READ ERROR ON <disk-name> AVAILABLE TABLE. <disk-name> AVAIL. TABLE WILL NOT BE CHECKED	MCP input error on reading the available table.	
	<PROMPT 3>	Disk capacities have been compared and are equal, disk-name has been successfully opened and the disk-label has been successfully read.	Proceed to the operating instructions for PROMPT 3.

PROMPT 3:

"ENTER TEST NUMBERS OR ENTER HELP
ENTER SPACE TO TERMINATE"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	*	DCR has been terminated.	Restart utility
AX <mix-no> HELP	ENTER ON 1 LINE NUMBERS FOR TESTS WANTED 1 = READ.ALL 2 = OPEN.FILES 3 = READ.CYLINDERS 4 = WRITE.INDEX 5 = READ.RANDOM ALL = ALL TESTS WANTED SPACE = TERMINATE DCR <PROMPT 3>	Help function has been requested.	Select tests required and input else terminate the utility.
AX <mix-no><non-numeric entry>	NON-NUMERIC INPUT CHARACTER <PROMPT 3>	No valid test numbers input.	Re-input correctly.
AX <mix-no><test number list>	<PROMPT 3>	Test numbers not in the range 1-5 and not "ALL".	Re-input correctly.
	<PROMPT 4>	Test 1 or "ALL" tests have been selected specified.	Proceed to input for PROMPT 4.
	<PROMPT 5>	Test 4 or "ALL" tests have been selected.	Proceed to input for PROMPT 5.
	<PROMPT 6>	Test 5 or "ALL" tests have been selected.	Proceed to input for PROMPT 6.

If tests 1, 4 or 5 (or ALL) have not been requested, the utility begins to execute test 2 and/or test 3 if selected.

PROMPT 4:

"MULTI-SECTOR BUFFERING WANTED? <Y or N>"

This prompt is displayed if test 1 or ALL has been requested.

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <neither Y nor N>	REPLY NOT "Y" OR "N" <PROMPT 4>	Reply was not Y (for yes) or N (for no).	Enter Y or N.
AX <mix-no> Y		When performing test 1 (READ.ALL) a buffered read will be invoked, with 32 sectors being read in each read operation.	
AX <mix-no> N	MULTI-SECTOR BUFFERING NOT WANTED	When performing READ.ALL (test 1), single sector reading will be invoked, thus maximizing head movement.	

PROMPT 5:

"ENTER NUMBER OF RECORDS IN INDEX.FILE OR SPACE TO CANCEL TEST"

is displayed if test 4 or ALL has been requested. The number specified is used as the size of the index file written and read in test 4.

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	Test 4 (WRITE.INDEX) will not be performed.		
AX <mix-no> <decimal no>	NUMBER EXCEEDS DISK CAPACITY <PROMPT 5>	n is greater than the number of sectors on the disk.	Enter a number smaller than or equal to the number of sectors on the specified disk.
AX <mix-no><non-numeric entry>	NON-NUMERIC INPUT CHARACTER <PROMPT 5>	The reply was not a decimal number.	Re-input the desired (decimal) number of records.

PROMPT 6:

"ENTER NUMBER OF RANDOM SECTORS TO BE READ FROM EACH DISK OR SPACE TO CANCEL TEST"

is displayed if test 5 or ALL has been specified.

The number entered must be less than 16 777 216 and will be used as the number of sectors to be read in test 5 (READ.RANDOM).

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>		READ.RANDOM (Test 5) will not be performed.	
AX <mix-no> <decimal no>	INPUT NUMBER > 16777215 <PROMPT 6>	Self explanatory.	Re-input number smaller than 16777216.
AX <mix-no><non-numeric entry>	NON-NUMERIC INPUT CHARACTER <PROMPT 6>	Self explanatory.	Re-input desired (decimal) no.

TEST 1 (READ.ALL) may output any of the following error messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
READING ALL n SECTORS ON <disk-name> MULTI-SECTOR BUFFERED (or) BUFFERED BY SECTOR	Number of records in SYSMEM, for each disk being tested is obtained and that number displayed in this message (n). The number of records in SYSMEM may be fewer than the number of physical sectors on a disk.	
MULTI-SECTOR BUFFERING NOT AVAILABLE	If OPEN fails on a 32 sector buffered SYSMEM this message is displayed and input (for both disks, if two are used) will be via the single-sector buffers.	
ERROR ON <disk-name>	If multi-sector buffering is being used and the MCP reports an error when a given track is input to the buffer this message is displayed. DCR re-reads the track via the single sector buffer in an attempt to isolate the bad sector.	
BUT NO ERRORS WHEN SECTORS READ	The MCP has reported no errors when records of the track have been re-read.	
READ ERROR ON <disk-name>, <sector address>	The MCP has reported a read error, other than when a multi-sector buffer is first filled and DCR displays this message.	
NO ERROR ON MULTI-SECTOR READ	Multi-sector buffering is being used and a read error was not reported when the buffer was first filled.	
LISTED AS BAD IN AVAIL.TABLE (or) NOT LISTED AS BAD IN AVAIL.TABLE	The available table has been checked, and appropriate message output. The count of sectors for which read errors have been reported is maintained.	

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
n SECTORS READ n BAD SECTORS READ ON <disk-name>	After every 10% of sectors on each disk being used have been read, DCR displays this message.	
READING ALL SECTORS ABORTED	More than 300 read errors have been reported on one disk - READ.ALL will be terminated if this message is output.	
READING ALL SECTORS COMPLETED. <n> SECTORS READ ON <disk-name>	All sectors have been read and, for each disk, a summary of read errors reported. DCR proceeds to next specified test.	

TEST 2 (OPEN.FILES) may display any of the following messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
OPENING UP TO 20 FILES ON <disk-name>	This is displayed for each disk.	None.
ERROR READING FDNL ON <disk-name> OPEN. FILES TRUNCATED	The file directory name-list (FNDL) is searched to obtain a maximum of 20 file identifiers. This error is reported while reading the FDNL. DCR will attempt to open only those files whose identifiers were obtained before the error was reported.	None.
FILE <file-name> ON <disk-name> OPENED	DCR attempts to open conditionally each of the files. Names were obtained from the FDNL's. This message indicates that the open is successful and closes the file and removes it without performing any other I/O operations.	None.
<file-id> ON <disk-name> DID NOT OPEN	Attempt to open a file is unsuccessful.	None.
NO FILES ON <disk-name>	The address of a FDNL is incorrect or the FDNL is corrupt in a way that does not produce MCP read error reports.	None.
<n> OPENS TRIED ON <disk-name>	Attempts have been made to open all possible (up to 20) files on a given disk	None.
<n> SUCCESSFUL, <n> UNSUCCESSFUL	Summary of the results of all possible opens (up to 20).	None.

TEST 3 (READ.CYLINDERS) may display any of the following messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
READING <n> CYLINDERS ON <disk-name>	For each disk DCR obtains the number of cylinders on the disk and displays this message.	None.
READ ERROR ON <disk-name>, SECTOR <sector address>	If the MCP reports an error on a read, DCR displays this message.	None.

After every 10% of the sectors (cylinders) read on each disk, DCR displays:

"<n> SECTORS READ"

and from each disk:

"<n> BAD SECTORS READ ON <disk-name>"

n being a count being maintained of MCP error reports during this test.

When all cylinders have been read, DCR displays:

"READING CYLINDERS COMPLETED"

and for each disk a summary of the read errors reported by the MCP:

"<n> BAD SECTORS READ ON DISK <disk-name>"

and proceeds to the next test specified by the user.

TEST 4 (WRITE.INDEX) may display any of the following messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
WRITING INDEX FILE ON <disk-name>	This is displayed for each disk being tested.	None.
FETCH VALUE= <hex.no> OPEN FAILED ON <disk-name> WRITE.RANDOM ABORTED	DCR attempts to open an arbitrarily named ("YELNOC") index file on each disk. If the open fails this message is displayed and the test is abandoned.	None.
FETCH VALUE= <hex.no>	This is displayed by DCR if the MCP reports a conditional failure on any write operation.	
DATA SPACE NOT AVAILABLE ON <disk-name>	There is not enough user disk for the utility to operate.	Remove some backed up files to make room and re-execute the utility.

If the MCP reports an error on any write operation during WRITE.INDEX, the following message is displayed:

"WRITE ERROR ON <disk-name>, KEY = <key-value>"

The utility compares the sector number against the list of bad sectors in the disk's available table, displaying one of the following messages accordingly:

"LISTED AS BAD IN AVAIL.TABLE"
or
"NOT LISTED AS BAD IN AVAIL.TABLE"

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
WRITING INDEX FILE TRUNCATED	A conditional failure or an error on any write operation. No more records are written (to either file if two disks are being tested).	AX <mix-no> <empty>
READ ERROR ON <disk-name>, KEY = <key-value>	DCR reads back index files in the order in which they were written. If the MCP reports an error on any read operation this message is displayed by DCR, which maintains a count of bad sectors for each disk.	
READING INDEX ABORTED	More than 300 read errors are reported on a file. If test 5 (READ.RANDOM) has been specified the utility will proceed to this test, if not the utility will go to EOJ.	
DATA MISMATCH ON <disk-name> RECORD <hex.number>	When each record is read in, the data is compared against the value determined by the records ordinal number. If the values are not identical DCR displays this message.	
WRITING INDEX FILE FINISHED	All records have been written.	None.
READING INDEX COMPLETED	All records have been read.	None.
<n> BAD SECTORS READ ON <disk-name>	Summary of read error reports.	None.
<n> DATA MISMATCHES ON <disk-name>	Summary of data mismatches.	None.

TEST 5 (READ.RANDOM) may display any of the following messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
READING <n> RANDOM SECTORS ON <disk-name>	This is a response in reply to number of random sectors requested in PROMPT 6.	None.
READ ERROR ON <disk-name>, SECTOR <sector address>	If the MCP reports a read error, DCR outputs this message and maintains a count of read-errors.	
READING RANDOM SECTORS ABORTED	If more than 300 errors are reported from a disk, DCR displays this message and abandons the test.	
<n> SECTORS READ	After every 10% of the specified number of sectors have been read from each disk being tested, this message is displayed.	None.
<n> BAD SECTORS READ	After every 10% of the specified number of sectors have been read from each disk being tested, this message - a summary of the error error count n - is output.	
READING RANDOM SECTORS COMPLETED	When the total number of requested sectors have been read from each disk. This message is displayed.	
<n> BAD SECTORS READ ON <disk-name>	Total number of bad sectors read.	

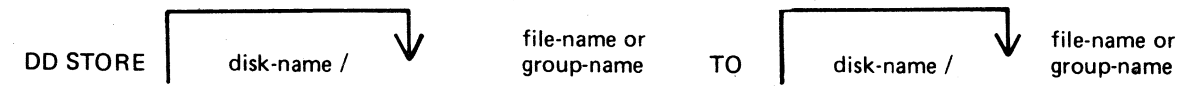
DD (Disk Dump)

This utility allows the operator to back-up ("STORE") one or more files from a disk to one or more disks (cartridge or BSMD) whose capacity is less than the originating media, and later "RESTORE" these files to their original state.

Note that DD will not be issued for any releases after 3.03. Any back-up disks created with DD must be converted to disks created with the DUMPADISK utility. This may be accomplished by restoring the files using DD, and dumping them using DUMPADISK.

Store Function:

Format:



The STORE function allows the operator to store a file or group of files between two disk media, where the size of the file may be greater than the physical size of the disk to which the file is being copied (destination disk). The files to be stored are specified by "file-name" or "group-name" and are taken from the system disk unless "disk-name" is specified. Additional files or groups of files may be specified, separated by a comma.

When the original destination disk is filled, the utility will request the name of the next disk to be used. The disk name is accepted from the operator who should then power off the drive using the "PO" intrinsic, insert the new disk into the drive, and use the "RY" intrinsic to allow the utility to continue. When all the files have been copied, the original destination disk must be re-inserted before DD will go to End of Job.

STORE re-computes block sizes for optimization and to allow a file to be copied to multiple disks. As a result, these files will not be usable as in their original form. The files must be RESTORED.

Examples:

To store a file called PR200 from the disk PRI to the disk PRBU:

DD STORE PRI/PR200 TO PRBU/PR200

To store a group of files beginning with the letters "PR" from the disk PR2 to the disk PRBU:

DD STORE PR2/PR= TO PRBU/PR=

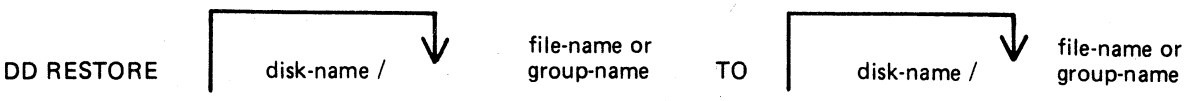
To store a file called PRCLEF from the system disk to the disk PRBU:

DD STORE PRCLEF TO PRBU/PRCLEF

The information needed to restore these files is kept in a file called "DDSTRESTORE" which is created when DD begins processing. Because of this, DD may not be run twice to the same destination disk or the information required to restore the first set of files will be lost.

Restore Function

Format:



This option allows the operator to restore a file or group of files between 2 disks where the disk from which the files are being copied (source disk) was generated by "file-name" or "group-name". Files are copied to the system disk unless "disk-name" is specified. Additional files or groups of files may be specified, separated by a comma.

When all files have been restored, the utility will inform the operator what disk must be inserted next to continue the transfer. Power off the disk drive using the "PO" utility, insert a new disk, and use the "RY" intrinsic to continue the utility.

Examples:

To restore a file called PR200 from the disk PRBU to the disk PRI:

DD RESTORE PRBU/PR200 TO PR1/PR200

To restore a group of files beginning with the letters "PR" from the disk PRBU to the disk PR2:

DD RESTORE PRBU/PR= TO PR2/PR=

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
group-name - NO FAMILY MEMBERS FOUND	Specified group is not on disk.	Check input and re-enter if necessary; Check for correct disk.
FILE NAME TOO LONG	Specified file name is longer than 12 characters.	Re-input correctly.
file-name NOT STORED - ILLEGAL REQUEST	Attempt was made to store a system file (that is, MCP, interpreters, etc).	None.
ENTER NEW PACK ID FOR CONTINUATION OF TRANSFER	Program requests next disk for STORE continuation ACPT disolay.	Enter AX mix number of program and new disk name to which STORE will continue to copy.
INSERT PACK disk-name FOR CONTINUATION OF TRANSFER INSERT NEWPACK disk-name FOR CONTINUATION OF RESTORE	Continuation disks are required.	Respond to ACPT message with AX mix number disk-name. Power off (PO) disk currently filled, insert new disk, type "RY" to contirue processing.
NOT FIRST PACK OF A STORED SERIES PACK disk-name REQUIRED	Source disk specifieo in RESTORE is not first disk of a stored series of disks.	Power off (PO) inappropriate disk, insert proper disk.
file-name STORED TO file-name file-name RESTORED TO file-name	STORE or RESTORE successful	None.

NOTE: Refer to "Common Utility Output Messages" for additional aid.

DSKUTL

The DSKUTL utility provides the following functions:

RF	Reformats a previously initialized disk according to parameters specified in the initiating message.
LIST	This function provides the facility to list any range of disk sectors (absolute addressing) which are accessible to the system software.
COPY	The COPY function provides the facility to duplicate disks of the same type. Note that only SDI-type disks are supported by this function. The entire disk contents are duplicated including the CMS-reserved areas of the disk. Non-CMS format disks can also be duplicated.

NOTE

This utility has the facility to handle pseudo-disks. This feature is not implemented on B 90 systems. In addition, DSKUTL can only be used by systems which implement the "Open Disk Unlabelled" function in the MCP.

These make the following Stand Alone Utility (SAU) functions available under MCP control:

IN, FE (for SDI disk types only)
RF (for any SDI disk type and pre-initialized Caelus disks)
PDX (for any initialized disk)

The disk types which are supported by the DSKUTL RF and LIST functions are detailed in table 4-2.

Table 4-2. DISKUTL – Supported Disk Types

DISK/PACK	CONTROLLER	MNEMONIC	DEVICE	CYL.	TRACKS/ CYL.	SECTS/ TRK	ALLOC. UNIT	NO. OF SECTORS
1MB MINI	c	DM	C7	88	2	32	1	@001600@
CARTRIDGE	(100TPI) c	DK	CB	203	2	32	1	@0032C0@
CARTRIDGE	(200TPI) c	DK	CB	406	2	32	1	@006580@
201-I FIXED	(single) c	DF	CC	406	2	64	1	@00CB00@
201-I FIXED	(dual) c	DF **	CC	406	4	64	2	@019600@
211 FIXED	(single) s	DF	CD	335	4	80	2	@01A2C0@
211 FIXED	(dual) s	DF	CD	335	8	80	4	@034580@
211 FIXED	(quad) s	DF	CD	335	16	80	8	@068B00@
BSMII MINI	(3MB) s	DM	CE	142	2	59	1	@004120@
BSMII MINI	(4.7MB) s	DM	CE	221	2	59	1	@0065C0@

s = SDI and c = caelus in the above table
** = B900 ONLY

DSKUTL operates under MCP control and provides some of the functions of the Stand Alone Utilities (SAU) necessary to support a system without a console printer.

In order to use this utility, the system must be running under an MCP that supports the OPEN/CLOSE disk unlabelled function and a SYSLANGUAGE file must be present. These functions are provided by the B 90 MCP version 3.03 and above.

Before executing DSKUTL, the operator must reserve the disk drive or unit on which the function is being performed by means of the RD intrinsic. When the utility begins execution, it opens an unlabelled disk and the operator is requested to assign the required disk drive or unit mnemonic which has been previously reserved by using the AD command.

The disk to be reserved for access by DSKUTL must not have any files open, thus the disk currently in use as the system disk may not be accessed by this utility.

NOTE

In order to use this utility, the disks in use must be in a format recognizable by the MCP as follows:

SDI disks are factory initialized, although not necessarily in LIVM CMS format, and are supported by this utility whether or not they have been previously initialized or formatted on a CMS system.

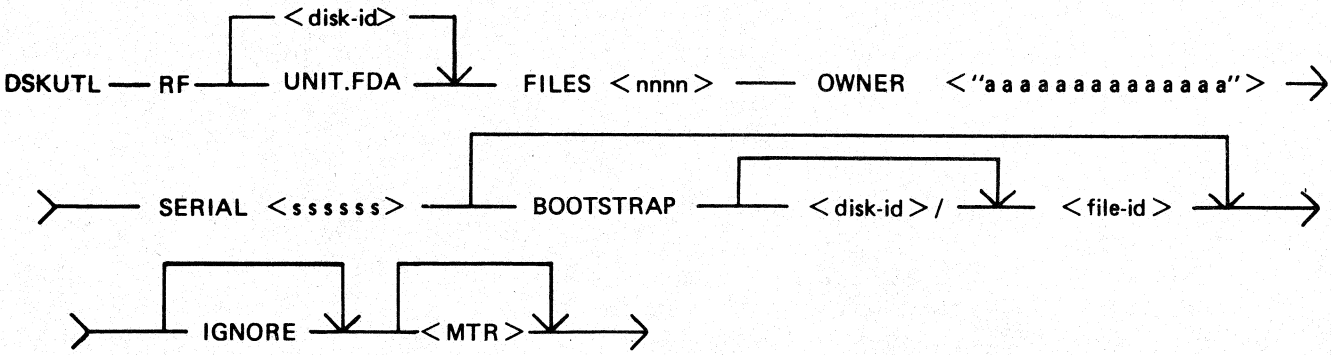
Unused Caelus disks require sector marks to be written throughout the disk surface before use under MCP control. In order to access any Caelus disk using this utility, the disk must previously have been initialized using the SAU 'IN'' function on a system with a console printer, or an SAU.PARAM file created by the utility SAU.INIT for use on a system without a console.

Format:

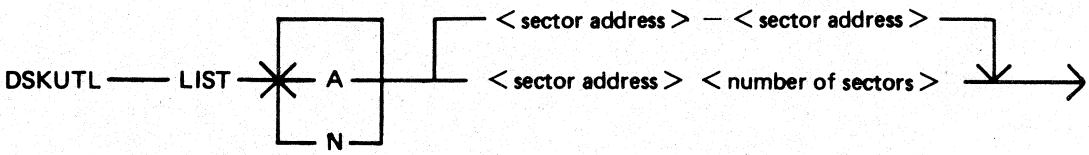
The following format provides the reformatting (RF) function.

NOTE

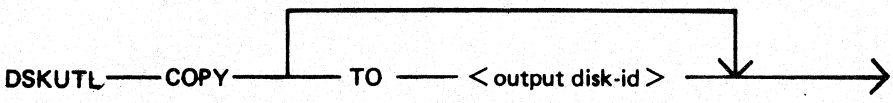
The diamond braces <> must be included in the initiating message for MTR tracks to be reserved.



The following format provides the list (LIST) function.



The following format provides the disk duplicating (COPY) function.



Format 1, RF:

disk-id This field specifies the name written to the disk-label of the disk being reformatted.

UNIT.FDA By specifying UNIT.FDA instead of <disk-name> in the initiating message, DSKUTL will reformat a 211 or 201I fixed disk as a physical unit with a PPIT, provided the system is running as a fixed disk system. The system PPIT will be scanned for the first disk-name listed as a physical unit which is not on-line. The disk being reformatted will be assigned this disk-name and associated logical unit-number when the utility writes the disk-label.

If all physical units listed in the system PPIT are found to be on-line, the disk will be assigned a default disk-name of "00000xy" where "xy" is the logical unit-number (in decimal characters) of the first available unit slot in the system PPIT.

FILES nnnn

This entry specifies the maximum number of files to be held on the disk. <nnnn> is a decimal number in the range 1 to 2804.

OWNER "aaaaaaaaaaaaaa"

This field is used to specify the owner field of the disk-label. It must consist of an alphanumeric entry of 1 to 14 characters delimited by quote characters ('').

SERIAL ssssss

The serial number field of the disk-label is specified by this field. It must be a six-digit decimal number and must include significant zeros.

BOOTSTRAP file-name

The bootstrap file-name, if specified, specifies the name of the bootstrap file to be copied to the disk being reformatted. If the BOOTSTRAP option is NOT entered, the utility will search for the default file-name as shown in table 4-3.

IGNORE

If the IGNORE option is specified, the utility will ignore any information already contained in the CMS reserved areas of the disk. This option must always be specified if the disk is not already in CMS format.
If IGNORE is not specified, the utility will expect the disk-label and available table to be in CMS format. When re-writing the available table to the disk, the utility will accommodate up to 90 bad entries from the original available table. If more than 90 bad entries are encountered on reading the available table of the disk, the utility will be terminated.

NOTE
The IGNORE option can be used to reformat a disk which has a corrupted label, directories or available table. However, care must be exercised if using this option to reformat Caelus disks (see table 4-2).
SDI disks, due to the fact that bad sectors are "hidden" by relocation, should never contain bad entries in the available table other than those reserved for MTR purposes. Caelus disks, however, often contain a number of unusable sectors "hidden" by means of the entries in the available table. Use of the IGNORE option will ignore these entries when constructing the new available table. Subsequent access of these previously noted bad sectors by the MCP can give rise to I/O errors.

<MTR>

The MTR option in the initiating message causes sectors to be reserved in the available table for MTR purposes. The number and location of these sectors depends on the system and disk type. This option is equivalent to the SAU FE function.

NOTE
The diamond braces <MTR> must be included in the initiating message in order to reserve MTR tracks.

Table 4-3. BOOTSTRAP Table

System	B 90	
Default file-name	0000000/CMSBOOT	
Records 2-31	CMS BOOTSTRAP	
Records 32-61	MTR1 CAELUS BOOTSTRAP	
Records 62-91	MTR2 SDI BOOTSTRAP	
For MTR Reformat	**BOOTSTRAP Selected**	**Areas Reserved**
1MB Mini	MTR1	0A80-0ABF 0B00-0B3F 1540-157F
Cartridge (100 tpi)	MTR1	1540-157F 2A80-2ABF 2B00-2B3F

(continued)

Table 4-3. BOOTSTRAP Table

Cartridge (200 tpi)	MTR1	2A80-2ABF 2B00-2B3F 5540-557F
201-I Fixed (S)	CMS	0080-017F 5200-527F CA80-CAFF
211 Fixed (20MB)	CMS	None
211 Fixed (80MB)	CMS	None
BSMII Mini 3MB	MTR2	1321-140C

NOTE

Area addresses are in allocation units (hexadecimal). On the 211 fixed disk, there are MTR sectors but these are located outside the MCP addressable area of the disk.

Operation

The utility immediately opens the specified or default bootstrap file and verifies that the appropriate checksum is correct. The operator is requested to reserve and assign the required disk drive or unit.

If the label of the disk is already in CMS format, the utility displays the name read from the disk-label and requests the operator to confirm that reformatting is required.

The operator must enter:

AX <mix-number> OK

in order for the utility to continue.

The new or existing available table and directory entries and a single SYSMEM entry are written to the output disk and the bootstrap file is copied to the disk.

If UNIT.FDA was entered in the initiating message, a 13-sector PPIT is written to the disk with the assigned disk-name, and the logical unit-number and pseudo-pack tag are entered in the physical unit slot. All other entries are made available. When the unit is subsequently made ready by the operator on termination of the utility, the PPIT will be modified to match the system PPIT by the AVR routine in the MCP.

Finally, the utility writes the disk-label to sector zero (0).

In order to access the disk which has been reformatted, the user must made the disk ready by use of the RY command when the utility has gone to successful EOJ .

Example:

To reformat a BSMII (3/6 mbyte) disk ("TESTA"), already in CMS format with the following parameters:
Disk-name – DISKA
Number of files – 1000
Owner – USER-FRED
Serial Number – 123456
Bootstrap file is default

Insert disk called "TESTA" in drive DMA.
System displays "DMA TESTA/ 0 FILES OPEN"
Enter "DSKUTL RF DISKA FILES 1000 OWNER "USER-FRED" SERIAL 123456"
System displays "READY AND RESERVE REQUIRED DRIVE FOR DISK REFORMAT"
"<mix-number>/DSKUTL <14> WAITING UNLAB REFORMAT DK DEVICE REQUIRED"
Enter "RD DMA"

System displays "DMA RESERVED 0 FILES OPEN"
Enter "AD <mix-number> DMA"
System displays "DMB IS CMS DISK TESTA - CONFIRM <RF> REQUIRED"
Enter "AX <mix-number> OK"
System displays:
"CMS BOOTSTRAP VERSION XX.XX.XX USED ON DMA DISK DISKA"
"0 SECTORS ARE UNAVAILABLE ON DMB DISK DISKA"
"DSKUTL <RF> OF DMB DISK <16672> AS DISK DISKA ON B90 COMPLETED"
Enter "RY DMA"

The reformatted disk can now be accessed as normal.

The following messages can be displayed by the RF function:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PARAMETER SPECIFICATION FOR <RF> INCOMPLETE	At least one of the parameters required in the initiating message for the reformat option is missing.	Check initiating message and re-input.
NO PPIT ON SYSTEM	INIT.FDA was specified in the initiating message and the system running is not a fixed disk assemblage.	Check system configuration and initiating message.
BOOTSTRAP FILE <file-name> NOT CMS STANDARD	The check-string of the specified or default BOOTSTRAP file is incorrect.	Check specified BOOTSTRAP file for integrity. Copy a new version if necessary.
READ ERROR ON BOOTSTRAP FILE <file-name>	A read error has been encountered on the specified or default BOOTSTRAP file.	Copy a new BOOTSTRAP file to the system.
NO UNIT SLOTS AVAILABLE IN SYSTEM PPIT	UNIT.FDA was specified in the initiating message. The utility has found all physical units listed in the system PPIT to be on-line, and there are no available physical unit slots in the system PPIT.	Check initiating message and system configuration.
READ ERROR ON PPIT OF <disk-name>	A read error has been encountered on the system PPIT while scanning for physical unit disk-name.	Re-initialize disk.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
READY AND RESERVE REQUIRED DRIVE FOR DISK REFORMAT	This message will be displayed by the utility before the MCP prompts the operator to AD the required disk drive.	Ensure disk being reformatted is reserved (RD).
OPEN DISK UNLABELLED NOT SUPPORTED	The utility is not able to open unlabelled the disk as the level of MCP in use does not support this feature.	Check version number of MCP.
CAPACITY OF Dxy NOT RECOGNISED - <number of sectors> SECTORS	The disk-size <number of sectors> returned by the MCP does not match any of the disk types supported by the utility.	Check the system configuration.
Dxy IS NOT A FIXED DISK	The disk assigned for RF UNIT.FDA is not a 201I or 211 fixed disk.	Check device assigned.
CANNOT READ SECTOR 0 OF Dxy	The utility has opened Dxy successfully but is unable to successfully read sector 0.	If the disk is a Caelus disk, ensure that it has been previously initialized. If this is the case, the disk will require re-initializing using SAU IN. For SDI disks, use the IGNORE option.
DISK LABEL OF Dxy NOT IN CMS FORMAT	IGNORE was not specified in the initiating message and the disk being reformatted does not contain a CMS format label in sector 0.	Re-enter initiating message using the IGNORE option.
Dxy IS CMS DISK <disk-id> - CONFIRM RF REQUIRED.	The disk label is already in CMS format.	Enter AX <mix-number> OK to continue.
DSKUTL <RF> OF Dxy NOT CONFIRMED	The utility has gone to EOJ without completing the reformat function. OK not entered as above.	Re-run the utility.
TRACK 0 OF Dxy BAD	The utility is unable to read from or write to one or more sectors of track 0.	Check integrity of media and drive. Replace media if necessary.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
BAD MTR SECTOR <z> ON Dxy.	The utility is unable to read from or write to one or more of the disk MTR sectors.	Check integrity of media and drive. Replace media if necessary.
DFy WILL BE REFORMATTED AS PHYSICAL UNIT <disk-id>	UNIT.FDA was specified in the initiating message and this disk name will be assigned to the fixed disk being reformatted.	Information only.
READ ERROR ON OLD DIRECTORY AT SECTOR <z> OF Dxy	The utility has encountered a read error while reading the directory of the disk being reformatted.	Re-enter the initiating message using the IGNORE option. WARNING This should be used with caution for Caelus disks.
OLD DIRECTORY OF Dxy IS CORRUPT	The utility has encountered an invalid entry while scanning the available table for bad sector entries.	Re-enter the initiating message using the IGNORE option. WARNING This option must be used with caution for Caelus disks.
TOO MANY BAD ENTRIES IN THE OLD AVAILABLE TABLE of Dxy	More than 90 bad entries have been found in the available table of the disk being reformatted.	Use IGNORE option. If the disk is Caelus disk, it must be initialized with SAU.
CANNOT ACCOMMODATE DIRECTORY FOR Dxy	The utility cannot find an available area large enough to accommodate the directory as specified.	Check and re-enter initiating message with smaller number of files to reduce size of new directory. Use IGNORE if SDI disk being used.
AVAILABLE TABLE TOO SMALL FOR Dxy	The new available table is not large enough to accommodate all required entries. Bad entries maintained from old available table, MTR sectors, and any bad sectors found during RF will not fit in area.	Re-enter initiating message with a larger number of files to increase size of new available table.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
WRITE ERROR ENCOUNTERED ON Dxy AT SECTOR <z>	The utility has encountered a write error while writing Bootstrap records MTR patterns Available table Directory. If the error is encountered while writing the directory, DSKUTL will mark these as bad in the available table and continues execution. For all other write errors it will abort.	Check integrity of disk drive and media. Replace media if necessary.
MORE THAN <z> BAD SECTORS FOUND ON DISK Dxy	More than 31 read/write errors encountered during reformatting of a 1mb disk, or more than 49 errors for all other disk types.	Check integrity of disk drive and media. Replace media if necessary.
DSKUTL <RF> ABORTED	Utility has terminated prematurely.	See message displayed immediately prior to abort message for reason.
<file-name> BOOTSTRAP VERSION <version-num> USED ON Dxy DISK <disk-id>	Confirmation of BOOTSTRAP file used during reformat.	Information only.
<z> SECTORS ARE UNAVAILABLE ON Dxy DISK <disk-id>	Confirmation of number of bad sector entries placed in the available table of the reformatted disk excluding MTR sectors.	Information only.
PLEASE RUN CHECK.DISK ON Dxy DISK <disk-id>	For 1mb disk only.	It is recommended that CHECK.DISK is run on all reformatted 1mb disks to ensure that surface integrity checks are performed.
DSKUTL <RF> OF Dxy DISK <disk-size> AS <disk-id> ON <system-name> COMPLETED	Normal termination on successful EOJ.	Information only. Reformatted disk must be RY'd before it can be accessed normally.

Format 2, LIST:

A OR N — Alpha character listing or numeric (hex) listing can be specified by the use of this option in the initiating message. The default is both alpha and numeric in the same format as for the CMS LIST utility.

sector address – sector address — This field specifies the sectors to be listed and consists of two decimal numbers separated by a hyphen. The sectors listed will include the starting and ending sector number.
Example: <1 – 20> causes sector numbers 1 through 20 to be listed.

sector address number of sectors — This option also specifies the number of sectors to be listed and can be used as an alternative to the above format. The number of sectors specified by the second decimal number will be listed, starting at the sector address specified by the first decimal number.
If sector zero (0) is specified for listing, then the following is written to the printer following the content of sector 0:

SECTOR ADDRESS	LENGTH	DIRECTORY
32/@0020@	65/@0041@	AVAILABLE TABLE
97/@0061@	91/@0058@	NAME LIST
188/@00BC@	1001/@03E9@	DISK FILE HEADERS
0/@0000@	0/@0000@	PSEUDO PACK IDENTIFICATION TABLE

The above sample output is from the disk reformatted with the parameters given in the example DSKUTL RF function above. If the disk-label is not in CMS format, this portion of the listing is omitted.

Operation:

The utility causes a disk to be opened unlabelled, and prompts the operator to reserve and assign the appropriate disk drive or unit. On assigning the device, the utility lists the specified sectors and goes to EOJ. The drive must now be made ready by the operator before normal access can be made to the disk.

Example:

To list the first 10 sectors of the disk reformatted in the sample DSKUTL RF function above:

```
Insert disk "DISKA" in drive DMA.
    The system displays "DMA DISKA/  0 FILES OPEN"
Enter "DSKUTL LIST 0 10"
      <or>
      "DSKUTL LIST 0 - 9"
    The system displays:
    "PRESENT DISK FOR LISTING NOW"
    "<mix-number>/DSKUTL <14> WAITING UNLAB LIST DK DEVICE REQUIRED"
Enter "RD DMA"
    The system displays "DMA RESERVED 0 FILES OPEN"
Enter "AD <mix-number> DMA"
```

The utility now lists the specified sector numbers and goes to EOJ.

The following messages can be displayed by the LIST function:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PRESENT DISK FOR LISTING PLEASE	The utility displays this message prior to the MCP. Prompts the user to RD and AD the required disk.	RD and RY the required disk.
OPEN DISK UNLABELLED NOT SUPPORTED	The utility is unable to open unlabelled the disk as the MCP being used does not support this feature.	Check version number of MCP in use.
ILLEGAL SECTOR ADDRESS <sector-address> FOR Dxy	The sector addresses specified in the initiating message are outside the physical capacity of the disk.	Check initiating message and re-enter.
Dxy EXHAUSTED DURING SECTOR RANGE <sector address - sector-address>	The last physical sector address of the disk in drive Dxy has been read. The utility will continue with the next specified sector range.	Check initiating message.
READ ERROR ENCOUNTERED ON Dxy AT SECTOR <sector-address>	A read error was encountered at the specified address. The utility will continue.	Information only. Check KA listing of the disk for listed BAD sectors.
DSKUTL LIST ABORTED	The utility has prematurely terminated.	See message displayed immediately prior to abort message for reason.

Format 3, COPY:

TO disk-name — For the COPY function, this option in the initiating message causes the disk-name specified to be written to the disk-label of the output disk if the input disk is in CMS format.
If the input disk is not in CMS format or is a fixed disk with a pseudo-pack PPIT, this name is ignored when writing the disk-label.
If no disk-name is specified in the initiating message, the disk-label is copied from the input disk to the output disk.

Operation:

On commencing execution, the utility displays the message:

"PRESENT INPUT DISK FIRST PLEASE"

The operator must now reserve and assign the disk-drive required for input. The utility displays the message:

"PRESENT OUTPUT DISK NOW PLEASE"

The operator must now reserve and assign the disk drive or unit required for output.

The utility checks that both input and output disks are SDI disks of the same type and capacity. Having established this, the utility proceeds to copy each sector of the input disk to the output disk.

If an error which cannot be corrected is detected, the operator is informed of the sector(s) in error and the utility continues until the entire disk is copied.

NOTE

This function is only available for duplication of SDI type disks. Any attempt to duplicate Caelus type disks will result in the utility displaying the message:

"Dxy DEVICE TYPE NOT SUPPORTED
DSKUTL (COPY) ABORTED"

Example:

To copy the disk "DISKA" created in the above example to a second disk "DISKB".

Insert "DISKA" into drive DMA and the output disk into drive DMB.
Enter "DSKUTL COPY TO DISKB"
The system displays "PRESENT DISK FOR INPUT FIRST PLEASE"
"<mix-number>/DSKUTL <14> WAITING UNLAB COPY DK DEVICE REQUIRED"
Enter "RD DMA"
The system displays "DMA RESERVED 0 FILES OPEN"
Enter "AD <mix-number> DMA"
The system displays "PRESENT DISK FOR OUTPUT NOW PLEASE"
"<mix-number>/DSKUTL <14> WAITING UNLAB COPY DK DEVICE REQUIRED"
Enter "RD DMB"
The system displays "DMB RESERVED 0 FILES OPEN"
Enter "AD <mix-number> DMB"

The utility now copies the disk and goes to EOJ.

The time required to copy a BSMII disk is approximately 12 minutes. This time may be significantly extended, however, if any errors are encountered during the copy.

The following messages can be displayed by the COPY function:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PRESENT INPUT DISK FIRST PLEASE	The utility will display this message just prior to the MCP prompting the operator to RD and AD the input disk.	RD and AD the disk required for input.
PRESENT OUTPUT DISK NOW PLEASE	The utility will display this message just prior to the MCP prompting the operator to RD and AD the output disk.	RD and AD the disk required for output.
OPEN DISK UNLABELLED NOT SUPPORTED	The utility is unable to open the disk as unlabelled, since the level of MCP in use does not support this feature.	Check the version number of the MCP.
Dxy and Dxz HAVE DIFFERENT CAPACITY	The two specified disks are incompatible for this function.	Check correct devices have been AD'd. Only like disk types can be COPIED.
TRACK 0 OF Dxy BAD	The utility is unable to read from or write to one or more sectors of track 0.	Check integrity of media and drive. Replace media if necessary.
Dxy DEVICE TYPE NOT SUPPORTED	An attempt has been made to copy a disk which is not an SDI type disk.	Check AD command for correct device type and re-execute utility.
DSKUTL <COPY> ABORTED	The utility has terminated prematurely.	See message displayed immediately prior to abort message for reason.
SPECIFIED DISK-ID <output-diskname> FOR COPY IGNORED. INPUT Dfy <disk-name> HAS A PPIT	The output disk will be assigned the same disk-as the input disk. The utility will continue.	Information only.
SPECIFIED DISK-ID <output disk-name> FOR COPY IGNORED Dxy DISK NOT IN CMS FORMAT	The input disk is not in CMS format. The specified output disk-name is ignored but the utility will continue.	Information only.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
READ ERROR ENCOUNTERED ON Dxy AT SECTOR <z>	A read error has been encountered on the input disk. The utility will continue.	See recommendations for read/write errors below.
WRITE ERROR ENCOUNTERED ON Dxy AT SECTOR <z>	A write error has been encountered on the output disk. The utility will continue.	See recommendations for read/write errors below.
DSKUTL COPY FROM Dxy TO Dxz COMPLETED	Successful terminating message when the input disk is not in CMS format.	PO disk and remove from drive.
DSKUTL COPY FROM Dxy DISK <disk-name> TO Dxz DISK <disk-name> COMPLETED	Successful terminating message when the input disk is in CMS format.	RY disk to enable normal access.

Recommendations for Read/Write Errors

If a read or write error is detected during the COPY function on either the input or output disk, the following procedure is recommended prior to using the output disk:

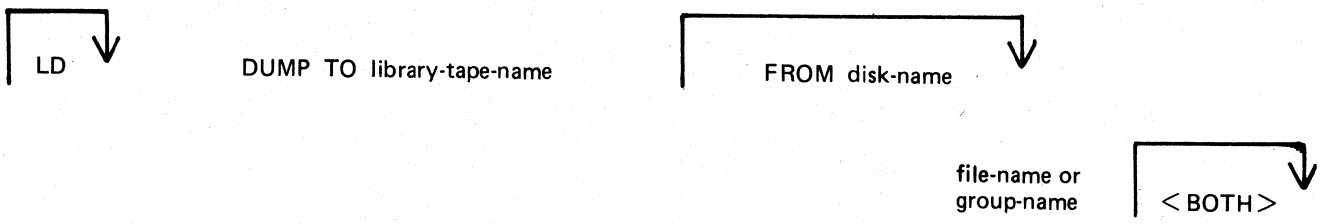
1. Produce a KA listing of the output disk.
2. From the sector addresses displayed by the utility for the read/write errors, determine which files contain the errors. These files should be removed from the output disk.

If the read/write error occurs in the disk directory area, the disk should not be used without re-initialization (or re-formatting if it is an SDI disk).

DUMP (Dump Files to Library Tape from Disk)

This function, a part of the LD allows the operator to copy files from disk to library-tape.

Format:



If the <BOTH> option is used following a request to dump a keyfile, the associated data file will also be dumped following the keyfile on tape, provided the data file is on disk.

Examples:

To dump a group of files beginning with the letters "AR" from the system disk to ARTAPE:

DUMP TO ARTAPE AR=

To dump a file called ARTASK and a group of files beginning with the letters "DCS" from a disk called ARDISK1 to a tape called ARTAPE:

LD DUMP TO ARTAPE FROM ARDISK1 ARTASK DCS=

To dump a keyfile called AR200K and its data file from a system disk to ARTAPE:

DUMP TO ARTAPE AR200K <BOTH>

Since "DUMP" is a part of the utility "LD", "LD" is actually what will appear in a mix message. To discontinue the DUMP function, 'DS" mix number/LD must be used.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO FILES IN THE FAMILY group-name ON DISK disk-name FOR DUMP	Specified group name not found on disk.	Check input and re-enter if necessary; Check for correct disk.
NO FILE file-name ON DISK disk-name FOR DUMP	Specified file was not found on disk.	Check input and re-enter if necessary; Check for correct disk.
file-name NOT DUMPED - IN OUTPUT USE- DUMP ABANDONED TAPE BEING PURGED.	File to be copied is in use and cannot be copied. If "DUMP ABANDONED" message is given, tape is purged and utility goes to EOJ.	None: utlity will not dump this file but will continue with next file to be dumped.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name NOT DUMPED - HAS BEEN REMOVED. DUMP ABANDONED - TAPE BEING PURGED.	File to be copied was removed between start of DUMP and time when file was to be copied to tape. Tape is purged and utility goes to End of Job.	None
file-name NOT DUMPED - HAS BEEN ALTERED - DUMP ABANDONED - TAPE BEING PURGED.	Contents of file to be copied were altered between the start of the dump and the time the file was to be copied to tape. Tape is purged and utility goes to End of Job.	None
file-name LOAD/ DUMP DISCREPANCY	End of File reached before expected. Disk File Header may contain errors.	None.
NO FILES TO DUMP	No files were found on this disk to copy to tape.	Check input and reenter if necessary; Check for correct tape.
file-name DUMPED	DUMP successful	None.
DUPLICATE file-name ALREADY BEING DUMPED	More than one recuest was made to DUMP this file.	None.
file-name NOT DUMPED - DATA FILE NOT ON LINE	<BOTH> option was specified, but data file was not found on disk.	If specified data file dump is required, supply DUMP with backup copy of file if it exists.

Note: Refer to "Common Utility Output Messages for additional messages.

DUMPADISK (Disk Dump)

The DUMPADISK utility is provided to enable periodic back-up and retrieval of user files to disk. It is provided as an alternative to the B 90 Stand Alone Utility (SAU) Copy, which cannot be used on systems without a console (B 93). The utility executes under MCP control and requires the open unlabelled disk facility in order to run. The B 90 MCP level 3.03 and above supports this function. DUMPADISK also requires the presence of the local language file (SYSLANGUAGE).

Note that all reference to pseudo-pack is for information only. The B 90 system does not support pseudo-pack. The B 90 is the only CMS system which supports "Open Unlabelled disk", which it does from the CMS 3.03 release.

NOTE

DUMPADISK will replace the current DD utility. The disk format created by DUMPADISK is unique and builds a non-standard disk directory. Disks created with the DD utility are NOT recognized by the DUMPADISK utility. As DD will not be issued for any releases after 3.3, users with disks created with DD must convert their back-up media to disks created with DUMPADISK.

The files created with the DUMPADISK utility can only be accessed using the DUMPADISK LOAD or ADD function. These disks can, however, be duplicated using the DSKUTL COPY function (see the description of DSKUTL contained earlier in this section). Disks used for back-up must be pre-initialized in CMS format and must not contain any user data. Any files existing on a disk written to by DUMPADISK will be lost when the DUMPADISK disk directory is written.

The utility also creates a security record when DUMPing to back-up disks. This record must be present on any disk used for a LOAD function.

DUMPADISK provides the facility to create one or a numbered sequence of back-up disks during one dump session.

Example:

In dumping files from a 2011 disk to 1mb disk, several 1mb disks may be required to accommodate all the required data. During the DUMP or UNLOAD function, replacement disks will be called for as each back-up disk becomes full. The directory of each disk will contain entries for all the files contained on that disk, and disks which have been created previously in that dump session. Thus in a series of three back-up disks created in the same session:

The directory of disk 1 has entries for all files on disk 1.
The directory of disk 2 has entries for all files on disks 1 and 2.
The directory of disk 3 has entries for all files on disks 1, 2 and 3.

When retrieving files from back-up disks, all files DUMPed may be LOADed successively, or a selection of individual or groups of files may be LOADed. In all cases, it is advisable to initiate any LOAD from the last disk of the sequence. In this way, if any files required for LOAD are not resident on that disk, the utility will call by name for the disk which contains those files.

When creating back-up disks, the utility will request a four-character prefix for disk identification purposes. Each disk created will be labelled with this prefix concatenated with a three-character decimal number, progressing from 001 to 999. During subsequent retrieval, all disks will be called for by their seven-character label (user-specified prefix plus decimal suffix). Back-up disk labels (sector 0) are created in CMS format.

Files can be DUMPed to and LOADed from any CMS supported removable disk media (except ICMD), and DUMPed or LOADed from and to any CMS supported disk.

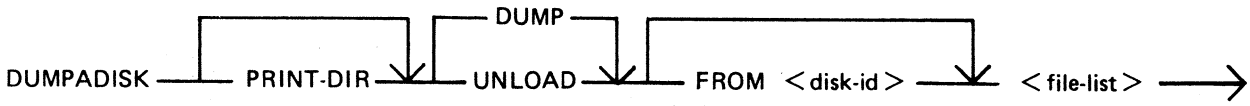
Format:

FORMAT 1

```
DUMPADISK PRINT.DIR
```

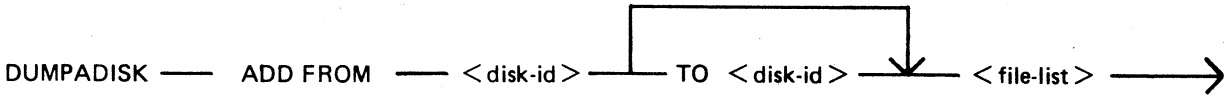
This function lists the disk directory.

FORMAT 2



This function enables back-up disks to be created together with the option to print the directory of the complete dump session.

FORMAT 3



This function enables retrieval of backed-up files to any named disk.

For formats 2 and 3, <file-list> is defined as a comma-list of individual file-ids, family-names (for example, ABCD=), or just = to specify all files on a disk.

NOTE

If indexed pairs are to be DUMPed or LOADed, then the <BOTH> option must be included in the file-list following the indexed pair file-name.

Format 1, PRINT.DIR

By specifying only PRINT.DIR in the initiating message, the utility prints the directory of the disk assigned to the utility.

Operation:

The utility will open-unlabelled a disk and prompt the operator to reserve (RD) and assign (AD) the disk drive containing the relevant disk.

The utility prints the directory and goes to EOJ.

Format 2, DUMP or UNLOAD

PRINT.DIR	This option in the initiating message causes the utility to print the directory of all files DUMPed during the session when it goes to EOJ.
FROM disk-name	This option allows the operator to specify the disk-name from which the files are to be DUMPed or UNLOADed. If this option is omitted from the message, the utility searches the system disk for the specified files.
File-list	This entry specifies the individual files or groups of files to be DUMPed or UNLOADed. The form of this entry can be a comma-list of individual files, a comma-list of file families (FILE=), or just = to DUMP all files from the specified or default disk.

Operation:

On executing, the utility first builds an internal directory or list of files to be DUMPed, and then displays the message:

"ENTER A FOUR CHARACTER DUMP IDENTIFICATION PREFIX"

The operator must now enter the prefix using an accept (AX) command.

The operator is now prompted to Reserve (RD) and Assign (AD) the disk drive in which the output disk is located. If the current disk is filled before the DUMP is complete, the operator is informed that it is OK

to PO drive <Dxy>. The utility then prompts the operator to replace the disk by displaying the following message:

"OVERFLOW DISK REQUIRED TO CONTINUE DUMP"

The operator must PO the existing disk and replace it with another disk.

On completing the DUMP, if the PRINT.DIR option was entered in the initiating message, the directory for the dump session is output to the line printer before the utility goes to EOJ.

Format 3, ADD or LOAD

FROM disk-id	This entry specifies the back-up disk-id from which files are to be LOADed or ADDed. When entering this name for a sequence of back-up disks, the name of the last disk in the sequence should be used. If the specified files contained in the <file-list> option of the initiating message are not on this disk, the utility will call for the required disk by name.
TO disk-id	This option allows the operator to specify the disk to which files are LOADed or ADDed.
file-list	This entry specifies the individual files or group of files to be LOADed or ADDed. The form of this entry can be a comma-list of individual files, a comma-list of file families (FILE=) or just = to indicate a LOAD of all files on the specified disk.

Operation:

The disk from which the ADD or LOAD is to take place must be READY before entering the initiating message.

If any of the specified files for LOADing are not resident on the disk, the available files are LOADed and the utility prompts the operator to supply the required disk. When all files have been LOADed or ADDed, the utility goes to EOJ.

NOTE

Files will only be ADDed if there is no file of that name already on the disk being ADDed to. A LOAD will replace existing files.

Error Recovery

If a read/write error is detected during DUMPing, the utility automatically attempts to reallocate disk space for the file currently being DUMPed. This process is repeated up to ten times for each file on any one disk. If a parity error is detected when attempting to write to a newly re-allocated area of disk, then the utility attempts to write to another area and ignore the previous attempt. Therefore, the utility attempts to re-allocate up to ten non-contiguous areas per file on each disk. If this limit is reached then the dumping of the current file is terminated (see the error messages listed below). The utility continues by dumping the next file, if there are any more to be dumped.

If an error is detected during a multifile ADD or LOAD, the utility terminates the LOAD for that file and continues to ADD or LOAD remaining files.

If a read/write error occurs in the directory area of a disk, the utility terminates.

The following messages can be displayed by DUMPADISK:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
<disk-id> IS NOT A RECOGNISED BACK-UP DISK	The disk being accessed for a LOAD or ADD is not a disk produced by DUMPADISK.	Check that the correct disk was inserted in the drive.
<file-name> NOT LOADED - ALREADY ON DISK.	The initiating message specified files to be ADDed which were already resident on the output disk.	RM existing files and re-run the ADD.
ALTHOUGH WITH DIFFERENT ATTRIBUTES	This message may follow and qualify the above message.	See above message.
<file-name> LOAD DISCREPANCY or <file-name> DUMP DISCREPANCY	DFH file-size and number of records DUMPed or LOADed are not equal.	Investigate discrepancy and re-execute the utility.
NO FILES TO DUMP	The utility is not able to find any of the files specified in the initiating message.	Check initiating message. If correct execute PD and check existance of files on the disk being backed-up.
FILE <file-name> NOT DUMPED followed by - - HAS BEEN REMOVED - IN OUTPUT USE - HAS BEEN ALTERED	The utility is unable to DUMP the specified file for the reasons stated.	Check status of files not dumped, and re-execute the utility.
(NO FILES IN THE FAMILY) <file> or (FILE) ON (BACKUP) or (DISK) FOR (LOAD) or (ADD) or (DUMP) or (UNLOAD)	This generalized message format is used to inform the operator when files are not found during DUMP or UNLOAD, LOAD or ADD.	Check initiating message for correct file-names and re-run utility.
DUPLICATE <file-name> ALREADY BEING (DUMPED) or (LOADED)	The initiating message contained multiple requests to DUMP or LOAD the same file.	None. The utility will continue to execute. Only one copy of the these <file-names> will be DUMPed or LOADed.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
NO FILES TO LOAD	The utility is unable to find any of the specified files on the back-up disk or sequence of back-up disks specified.	Check initiating message and disk-id of back-up disk being used.
<file-name> NOT DUMPED - DATAFILE NOT ON LINE	The data-file of an indexed pair specified in the DUMP initiating message cannot be found. The utility will continue. The key-file is not DUMPed.	Check the input disk for the presence of the specified files. Re-run the utility.
<file-name> - DATAFILE NOT FOUND ON DISK FOR LOAD	The utility cannot find the data-file on the back-up disk for the specified keyfile. The utility will continue but the keyfile is not LOADed.	Check the initiating message and print the back-up disk directory.
DISK <disk-id> NOT AVAILABLE or DISK <disk-id> LOCKED	Disk specified is either not on-line or is in use by another utility using SYSMEM LOCK on this disk.	Check <disk-id> of back-up disk in use or wait until other utility has gone to EOJ.
FILE LIST FOR <family-name> MAY BE INCOMPLETE or FILE LIST MAY BE INCOMPLETE	If DUMPing from an unrestricted pseudo-disk and a disk belonging to this pseudo-pack is not found, then the files DUMPed may not be a complete list.	Check PRINT.DIR listing and re-run utility if required.
UTILITY LIMIT REACHED AT FILE <file-name>	When DUMPing from an unrestricted pseudo-pack the number of files has reached the maximum,2804. DUMPing will now commence.	Information only.
OK TO PO DRIVE Dxy or OK TO PO DISK <disk-id>	Informs the operator that it is safe to PO a disk when DUMPing or LOADing and the next disk in a sequence is required.	PO specified disk.
OVERFLOW DISK REQUIRED TO CONTINUE DUMP	Follows first of above pair of messages to denote that another disk is required to continue the DUMP in progress.	Select required disk and continue.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
DISK <disk-id> REQUIRED TO CONTINUE LOAD	Informs operator of the next disk in sequence required to continue the LOAD.	Select the required disk and continue.
OPEN DISK UNLABELLED NOT IMPLEMENTED	The utility is unable to open the disk unlabelled as the level of MCP in use does not support this function.	Check version number of MCP in use.
DISK <disk-name> IS NOT REQUIRED FOR THIS LOAD	The files to be LOADed do not reside on this disk.	See next message.
MAKE DISK <disk-id> AVAILABLE AND ENTER "OK" or MAKE ALTERNATE DISK DISK AVAILABLE AND ENTER NAME	Follows the above message to inform operator that an alternate disk is required to continue the LOAD.	Select appropriate disk and continue.
ENTER A FOUR CHARACTER DUMP IDENTIFICATION PREFIX	Informs the operator to enter the four character disk-id prefix before starting the DUMP.	Enter the required four character prefix using the AX command.
DISK LABEL OF Dxy NOT IN CMS FORMAT	Sector 0 of the disk specified for DUMPing to is not in CMS format.	Check device assigned or check the disk located in that device.
(WRITE) or (READ) ERROR ENCOUNTERED ON DRIVE Dxy RANGE <hex> - <hex>	The utility has encountered a read/write error while DUMPing or LOADing.	See next message.
PROCESS CONTINUING	The utility has been able to recover.	No action necessary.
I/O ERROR DETECTED IN WRITE TO FILE <file-name> or I/O ERROR DETECTED IN READ FROM FILE <file-name>	The utility has detected an irrecoverable I/O error. The LOAD or DUMP of this file will terminate but the utility will continue with the next file.	At EOJ, check the file in error and re-run the utility.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ERROR AT RECORD NUMBER<decimal-number>	Follows the above message to inform operator of the record number in error.	
<file-name> NOT (DUMPED) or (LOADED) DUE TO I/O ERROR	Follows above messages to inform the operator that the DUMP or LOAD is incomplete due to an I/O error.	Check missing files and LOAD or DUMP from previous level of back-up.
DIRECTORY ERROR ON DISK Dxy	The utility has detected an I/O error in the directory of a back-up disk during LOAD or DUMP. The utility terminates.	Replace back-up disk if DUMPing. If LOADing, the previous level of back-up disk will be required.
READ ERROR ON PPIT OF Dxy	A read error has been encountered in the PPIT of a pseudo-pack. The utility will terminate without LOADing or DUMPing.	RF or IN pseudo-pack if necessary.
DISK <disk-id> DOES NOT BELONG TO THIS LOAD	A disk from the wrong DUMP sequence has been used. The utility will terminate.	Select the correct disk and re-run the utility from the disk just selected.
FILE <file-name> NOT LOADED - (ABNORMAL TERMINATION or (IN USE) or (REMOVED) or (INPUT PARITY) or (OUTPUT PARITY) or (ALTERED) WHEN DUMPING	During a LOAD the utility is unable to LOAD a file. The back-up disk does not contain the specified file. During the DUMP which created the back-up disk in use, the DUMP of that file was terminated because of the reason given.	If a copy of the file in question is required, it will be necessary to LOAD from an earlier level of back-up.

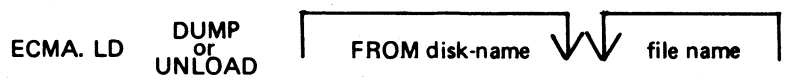
ECMA.LD (LOAD/DUMP OF ECMA TAPE FILES)

This utility allows the operator to structure tape files according to ECMA BASIC and ECMA COMPACT systems as specified in the STANDARD ECMA-41 publication.

The ECMA tapes are treated as unlabelled tapes in the CMS system. The utility is initiated in two different ways for BASIC system and COMPACT system.

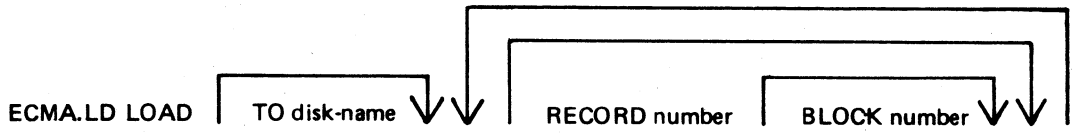
BASIC INITIATION

Format (Disk to Tape copy)



The files are copied from the disk to a purged tape. If the option UNLOAD is specified then the files copied are removed from the disk.

Format (Tape-to Disk copy)



The files are copied from an ECMA BASIC tape to the disk specified by disk-name. The option RECORD followed by a number specifies the record size in bytes on the tape and BLOCK followed by a number is the blocking factor, that is, number of records per block.

If the RECORD and BLOCK options are used then the first file on tape is read according to the first attributes, the second file is read according to the second attribute etc. If the block size (that is, record size x number of records per block) exceeds 256 then an error message is issued and loading is not performed.

If RECORD and BLOCK options are not used then tape files are loaded as 256 byte records blocked 1.

The names of the ECMA tape files loaded to the disk become ECMA001, ECMA002,....ECMA00n.

Examples:

To copy files IN001, IN002 from the systems disk to a tape in ECMA BASIC format:

ECMA.LD DUMP IN001, IN002

To copy file AR030 from the disk ARDISK to a tape and remove after copy:

ECMA.LD UNLOAD FROM ARDISK AR030

To copy files PR200, record size 80 block 160, PR210, record size 60 block 180, to the system disk from ECMA BASIC tape on device CTA:

ECMA.LD LOAD RECORD 80 BLOCK 2, RECORD 60 BLOCK 3

The utility will display the following message:

mix no/ECMA.LD <14> WAITING UNLAB ECMATAP/NONE AT DEVICE REQUIRED

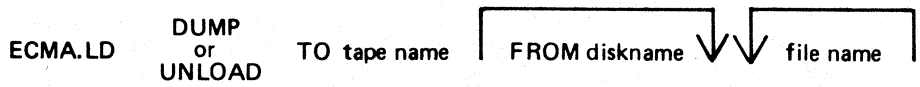
Then enter:

AD mix number CTA

The files PR200 and PR210 will be called ECMA001, ECMA002 on the disk.

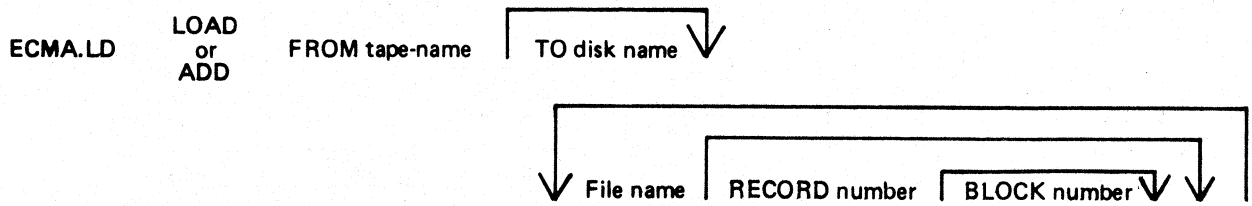
COMPACT INITIATION

Format (Disk to Tape copy)



The files are copied from the disk to tape. If the option UNLOAD is used then files copied are removed from the disk.

Format (Tape to Disk copy)



The files are copied from an ECMA COMPACT tape to the disk specified by disk name. If any file is already present on the disk, it will be removed before the same named file is copied. The option RECORD followed by a number specifies the record size in bytes on the tape and BLOCK followed by a number is the number of records in a block.

If RECORD and BLOCK options are used then the first file on the tape is read according to the first attribute and the second file is read according to the second attribute. If block size (that is, record size x number of records per block) exceeds 256 then an error message is given and utility will terminate.

If RECORD and BLOCK options are not used then tape files are loaded as 256 byte records blocked 1.

The ADD option will copy a file to the disk only if that file is not already present.

Examples:

To copy files IN001, IN002 from the disk INDISK to a tape FRED in ECMA COMPACT format:

ECMA.LD DUMP TO FRED FROM INDISK IN001, IN002

To copy files IN010 from the system disk to a tape FRED and remove from the disk after copy:

ECMA.LD UNLOAD TO FRED IN010

To copy files from tape FRED on CTA which is created in ECMA COMPACT format files IN001, IN002 to the system disk:

ECMA.LD LOAD FROM FRED IN001 RECORD 80 BLOCK 2, IN002

The utility will display the following message:

Mix number/ECMA.LD <14> WAITING UNLAB ECMATAP/NONE AT DEVICE REQUIRED

The response should be:

AD mix number CTA

Output Messages:

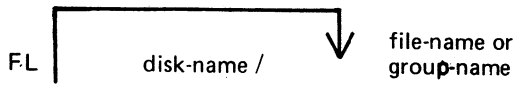
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NOT A BASIC ECMA TAPE	Loading is requested from a BASIC tape but the tape is not a BASIC format	Check the tape and re-input
NOT A COMPACT ECMA TAPE	Loading is requested from a COMPACT tape but the tape is not a COMPACT format	Check the tape and re-input
NO FILES IN THE FAMILY group-name ON TAPE tape-name FOR DUMP/LOAD/ADD/UNLOAD	Specified files are not present on the tape	Check the group name and/or the tape, and re-input
NO FILES IN THE FAMILY group-name ON DISK FOR DUMP/LOAD/ADD/UNLOAD	The specified group of files is not present on disk	Check the group name and re-input
NO FILE file-name ON TAPE tape-name/DISK disk-name FOR LOAD/DUMP/ADD/UNLOAD	The specified file is not present on tape or on disk	Check the file name and re-input
file-name REMOVED	This message is displayed for each file removed by a LOAD or UNLOAD	none
file-name LOADED	This message is displayed for each file added or loaded	none
file-name DUMPED	This message is displayed for each file dumped or unloaded	none
file-name NOT LOADED-ALREADY ON DISK	Self-explanatory	
file-name NOT FOUND	The file is not found on disk	Check input and re-enter
file-name HAS BEEN REMOVED - DUMP ABANDONED - TAPE BEING PURGED	The specified disk file to be copied to tape has been removed before the utility was able to copy it	none

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name HAS BEEN ALTERED - DUMP ABANDONED - TAPE BEING PURGED	The contents of the file to be copied to tape were altered after the start of the dump and before e the utility was able to copy it	none
file-name IN OUTPUT USE - DUMP ABANDONED - TAPE BEING PURGED	The file cannot be dumped to tape as it is being used as an output file	none
BLOCK COUNT ERROR	This is displayed when the BLOCK COUNT entry in the end-of-file label, end-of-track label, or end-of-volume label does not match with the actual number of blocks read during a LOAD or ADD	possible bad tape: retry with a different drive or with a backup tape
INCORRECT TAPE NAME	For a COMPACT format tape, the names in the initiating message and the tape label do not agree	Check the tape label andre-input
NOT FIRST PART OF FILE	A section number other than 1 was found when reading the first label on tape	
DATA BLOCK TOO LARGE	The block size on tape for a LOAD or DUMP is greater than 256 bytes	none
file-name DATA FILE NOT FOUND ON TAPE FOR LOAD	File specified for COMPACT format LOAD cannot be found	Check file name and re-input if necessary

FL (Display File Attributes on Self-Scan)

This utility allows the operator to display detailed information about particular files or groups of files on disk. The information given is similar to the LR utility, and is displayed on a console screen, or console printer if no console screen is available.

Format:



The utility uses the following PKs when more than one file is specified:

PK1	PK2	PK3	PK4	PK5	PK6
page to next screen	—	—	—	—	EOJ

Examples:

To display information about all entries on the system disk:

FL =

To display information about a file called PR200 found on disk called PR2:

FL PR2/PR200

To display information about a group of files beginning with the letters "PR" found on the system disk:

FL PR=

Output Format:

Information from this utility can be output in one of two forms depending on the size of screen being used. FL uses the 1920 character screen in this six-line format:

```
Line 1    FILE ddddddd/fffffffffff filetype @xy@
Line 2    ACTUAL SIZE xxxxxxx : RECORD SIZE xxxxx
Line 3    MAXIMUM SIZE xxxxxxx : RECS/BLOCK xxxxx
Line 4    DATE CREATED yyddd : ACCESSED yyddd
Line 5    AREA MAP : *************** UNIT : uuuuuuu
Line 6    OVERFLOW DISK : ooooooo FILE INCOMPLETE
          UNRESTRICTED PSEUDO
```

The 256 character screen is used in the following manner:

```
Line 1    FILE NAME: ddddddd/fffffffffff
Line 2    FILE TYPE: filetype @xy@
Line 3    SIZE: ACTUAL xxxxxxx, MAX xxxxxxx
Line 4    CREATED ACCESSED REC.SZ RECS/BLK
Line 5    yyddd yyddd xxxxx xxxxx
Line 6    AREA MAP: ***************
Line 7    OVERFLOW DISK: ooooooo (or UNRESTRICTED PSEUDO)
Line 8    FILE INCOMPLETE UNIT: uuuuuuu
```

Note that the OVERFLOW ON DISK will not be displayed if the file has no overflow areas allocated.

The first line contains disk name specified by DDDDDDD on which the file specified by FFFFFFFFFFFF resides.

The FILETYPE entry will contain one of the following:

FILETYPE	ENTRY IN LISTING/DISPLAY	
@00@	DATA	@00@
@01@ – @0E@	SRCLANG	@0x@
@0F@	SRCLIBR	@0F@
@10@ – @13@	CYYMMDD	@1x@
@21@	SYSLANG	@21@
@22@	SYSDATA	@22@
@30@	VIRTMEM	@30@
@31@	SYSLOG	@31@
@81@	KEY	@81@
@A0@	PRNTBKP	@A0@
Any other	SYSTEM	@xy@

ACTUAL SIZE	The actual file size specified for the file.
MAXIMUM SIZE	The maximum file size specified for the file.
RECORD SIZE	Number of characters per record.
RECS/BLOCK	Number of records per block.
DATE CREATED CREATED	Date of creation.
ACCESSED	Date the file was last accessed.
AREA MAP	16 characters to show the allocation of the 16 areas into which a file may be broken.

Entries for AREA MAP are as follows:

ENTRIES	AREA ALLOCATED ON:	POSSIBLE DISK TYPES
*	Unallocated	1 2 or 3
B	Base disk	1
O	Overflow disk	1 or 2
U	Physical unit	2 or 3
A	Another physical unit	3

Disk Types:
1 = Non-pseudo disk
2 = Restricted pseudo disk
3 = Unrestricted pseudo disk

The two component disks of a dual-pack file can be:

Two non-pseudo disks
or
One non-pseudo disk and one restricted pseudo disk.

"UNIT : uuuuuuu"	is only displayed if ddddddd is a pseudo disk; uuuuuuu being the physical unit on which this file directory information was found.
"UNRESTRICTED PSEUDO"	is displayed only if ddddddd is an unrestricted pseudo disk.
"FILE INCOMPLETE"	is only displayed if the file is a dual-pack or an unrestricted pseudo disk file, and one or more of the disk or units on which areas and/or directory entries for this file are not available.

"OVERFLOW DISK : oooooo" is only displayed if the file is a dual-pack.

Output messages:

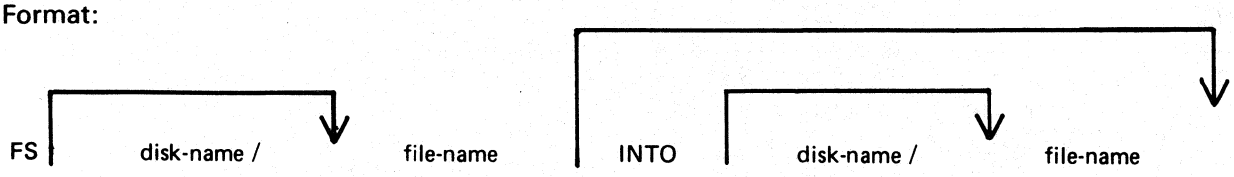
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FILE NAME = disk-name/file-name NOT FOUND IN DISK DIRECTORY	Specified file not found on this disk.	Check input and re-input if necessary; Check for correct disk.
disk-name/file-name NO FILES FOUND IN DIRECTORY FOR FAMILY	Specified group was not found on this disk.	Check input and re-input if necessary; Check for correct disk.

Note: Refer to "Common Utility Output Messages" for additional aid.

FS (File Squash)

This utility allows the operator to remove all deleted records from a data file. Records are normally "deleted" (that is, hexadecimal@FF@ are written over the records) through an appropriate application program. The FS utility will remove these previously deleted records, allowing additional records to be added to the file.

Format:



The "file-name" identifies either a data file or a keyfile. If a keyfile has been specified, the name of the data file is obtained from the information held in the keyfile.

If a keyfile is specified, then the utility will reconstruct this keyfile so that it relates to the modified data file.

While the utility is processing, no other program may access the data file (or the keyfile if one is specified).

If only one file-name is specified and no other options are used, the file squash will be carried out in place, and no new data file will be created. If a keyfile was specified, then a new keyfile with the same 'file-name" will be recreated by the SORT.

If 2 file-names are specified, the data file will be squashed into a new file, and the keyfile (if specified) will be recreated by the SORT. If the first file-name specifies a keyfile, then the new keyfile will have the name indicated by second file-name, and the new data file will have the name of the new keyfile name, with the letters "QQ" attached to the end of the name.

Examples:

To squash the file, PR200:

FS PR200

To squash the file, PR200 and create a new file, PR200B:

FS PR200 INTO PR200B

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name SQUASHED FROM n RECORDS TO m RECORDS	FS successful. Original and resulting fliesizes of the data file indicated. "n" and "m" are decimal numbers up to 7 digits each.	None.
KEYFILE file-name RECONSTRUCTED	"file-name" indicated by operator input identified by keyfile and FS has successfully squashed the data file and reconstructed the keyfile.	None.

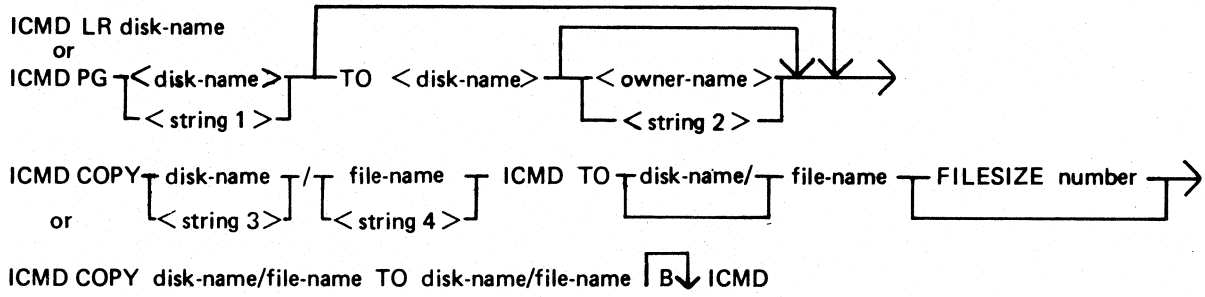
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
KEYFILE SORT FAILURE	SORT utility was not able to properly construct a keyfile.	None.
RECORD SIZE TOO GREAT	An attempt has been made to squash a file with a record size greater than 10000 bytes.	None.
ZIP TO SORT FAILURE - END OF JOB	When attempting to reconstruct the keyfile, the utility failed to execute SORT.	Ensure that SORT and SORTINTRINS are present on the disk, and re-input.

Note: Refer to "Command Utility Output Messages" for additional aid.

ICMD (Industry Compatible Mini Disk Access)

This utility allows the operator to access industry compatible mini disks (ICMD).

Format:



The first function of the utility (LR) allows the operator to print the disk directory of the ICMD. The utility will print a line of information for each file on disk.

The second form of the utility (PG) allows the operator to purge (erase) all files from an ICMD. The utility will replace all files by a single zero-length file called "DATA" to which is assigned all the disk space available to the user. By specifying "TO", the operator may change the disk's name to the second name, and in this case the owner's name may be set or made blank. The facility to re-label disks with non-standard characters contained within the disk-name is incorporated. When "TO" is specified, the operator must "RY" the drive containing the disk to enable it to be accessed by its new name.

<string 1> is a string of up to 6 characters enclosed in quotes, and
<string 2> is a string of up to 14 characters enclosed in quotes.

This allows a disk-id (volume I.D.) containing non-standard characters, spaces or nulls to be re-labelled for use on a B 90 system. Burroughs-initialized ICMD disks contain all EBCDIC spaces in this field.

To re-label an ICMD with non-standard characters in the disk-name:

1. Enter OL. This shows each device I.D. (for example DIA) followed by the disk-name (volume I.D.). The ICMD disk-name is printed. Blanks appear as null output.
2. Execute ICMD PG as defined above. The string contained within quotes must be identical to the name printed during the OL function. If this is blank, the string must be two quote marks with no intervening characters.

Example:

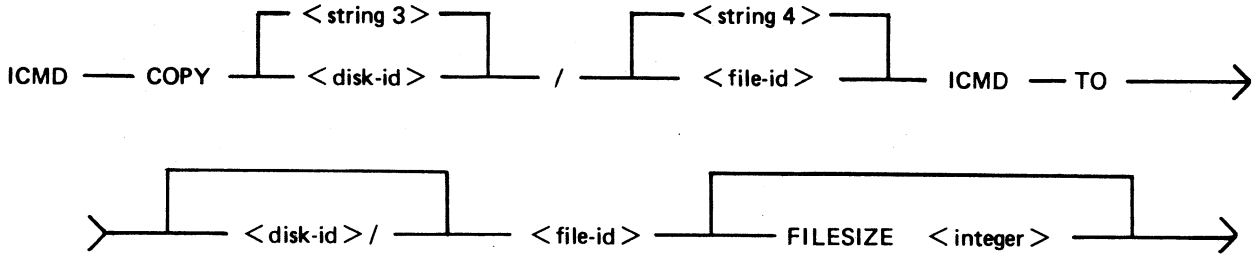
Where "FRED*2" is a non-standard label and DIB has a blank label:

```
OL
    DIA FRED*2
    DIB
- to re-label DIA, enter ICMD PG "FRED*2" TO CMSID1
- to re-label DIB, enter ICMD PG ""TO CMSID2
```

The third form of the utility allows the operator to copy an ICMD file to a CMS file. The CMS file will have the largest block not exceeding 180. If the FILESIZE of the CMS file was not specified, it will be calculated from the header of the ICMD file. Note that when copying a Multi-Volume File (file which resides on more than one disk) the FILESIZE option will be required.

It is also possible to copy files from an ICMD with a non-CMS name. The files on the ICMD may also have non-CMS names.

Format:



<string 3> is a string of up to 6 characters enclosed in quotes, and
<string 4> is a string of up to 8 characters enclosed in quotes.

The fourth form allows the operator to copy a CMS file to an ICMD file. The maximum record size of the CMS file must be 128 bytes (the ICMD sector size). The option 'B' allows the copying of any file without the loss of attributes, but it may be copied back by this utility only.

Examples:

To print the disk directory of an ICMD disk called PR2:

ICMD LR PR2

To purge all the files from the ICMD called PR2 and name it as PR3:

ICMD PG PR2 TO PR3

To copy a file called PR200 from the CMS disk called PRI to an ICMD disk called PRBU:

ICMD COPY PRI/PR200 TO PRBU/PR200 ICMD

To copy a file called PRFILE from the ICMD called PRBU to the CMS disk called PRI:

ICMD COPY PRBU/PRFILE ICMD TO PRI/PR200

To copy a file called !! from the ICMD called FRED* to the file called CMSFILE on the CMS disk called CMSDISK:

ICMD COPY "FRED*"/"!!" ICMD TO CMSDISK/CMSFILE

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ENTER VOL-ID FOR "file-name" n	Program is about to disolay an ACPT requesting the next disk-name containing the Multi-Volume File "file-name". "n" is the sequence number to be used, or blank if none.	Enter appropriate cisk-name.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
VOLUME n OF "file-name" OUT OF SEQUENCE	Sequence number "n" of "file-name" was not expected at this point. (Ex: program was expecting disk with sequence number of 2 and operator supplied disk with sequence number 4).	Remove inappropriate disk from disk drive (PO cannot be used with ICMD). Replace with disk having correct sequential number for this file.
DUPLICATE FILE "file-name"	File cannot be copied to ICMD as a file of the same name already exists. Disk cannot contain 2 files with the same name.	Copy this file while changing its name. For example, if file-name MYFILE is already on the ICMD, enter ICMD COPY DISK/MYFILE TO ICDISK/YOURFILE ICMD.
NO FREE LABEL ON "disk-name"	File cannot be copied to ICMD as its disk directory is full.	Replace ICMD with another ICMD having directory space. Re-attempt ICMD utility;
NO SPACE FOUND ON "disk-name"	File cannot be copied to ICMD as no unused space could be found.	FG ICMD and re-attempt copy; or select another disk on which to copy.

Note: If any of the above messages (including "disk-name" NOT FOUND and "file-name" NOT FOUND) is output in response to the initiating message, the utility will go to End of Job. If in response to an ACPT, the utility will repeat the message: ENTER VOL-ID FOR "file-name" "n".

BAD INDEX TRACK ON disk-name SECTOR number	ICMD has an error on Track 0 and cannot be used by this utility. Utility goes to EOJ.	Select another ICMD for this function.
RECORD SIZE OF file-name EXCEEDS 128	The file cannot be copied to the ICMD as its record size is more than 128, the maximum for an ICMD. Utility goes to EOJ.	Use MCP COPY utility to change record/block sizes. (EX: COPY CMSFILE TO CMSFILE RECORD 128, BLOCK 128) Then use ICMD COPY to complete transfer.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NEXT SEQUENCE NUMBER OF file-name EXCEEDS 99	No more disks may be used, as the maximum for an ICMD is 99. Utility goes to EOJ.	Use MCP COPY utility to break file (see COPY utility). Then use ICMD COPY to complete transfer.
INPUT ERROR ON file-name SECTOR number	Irrecoverable error was found on reading sector "number" from the ICMD. Utility goes to EOJ.	File cannot be copied from the ICMD; use backup ICMD if available.
OUTPUT ERROR ON file-name SECTOR number	Irrecoverable error was found on writing sector "number" to the ICMD.	Disk cannot be used. Use another ICMD.
file-name TO file-name COPIED	ICMD COPY function successful.	None.
disk-name PURGED	ICMD PG function successful.	None.
file-name IS NOT TYPE SOURCE OR DATA	The requested file has the wrong file type for this use.	None.
file-name REMOVED	The CMS file was removed by the utility to make way for another file being copied from an ICMD.	None.

Note: Refer to Common Utility Output Messages for additional aid.

IR (Initiate Log Recall)

(a function of SYS-SUPERUTL)

This function will initiate recall and go back in SYS-LOG files after skipping the number of entries specified by the operator (that is, 5 digit "offset") and display the required message.

Format:

IR offset

Examples:

To initiate recall after 12 entries and display the message on the console:

IR 12

To initiate recall of the message just given:

IR 1

Output messages

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
CANNOT LOCATE DESIRED LOG ENTRY	The log files have been transferred using TL utility	None.
	Offset is greater than number of entries in log file	Decrease the value of offset and re-enter.

KA (Analyze Disk Space Assignment)

This utility provides the operator with a map of all space used on disk by specific files, or available for other use. The printout is in ascending disk address order in terms of areas and their assignment.

KA is capable of analyzing space assigned to one or more files, one or more groups of files, or the available areas.

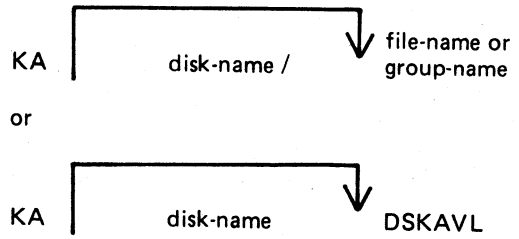
Special reporting is given if the group identifies all files on the disk (that is, disk-name/=). In addition to an analysis of the areas allocated to each file, this report will show the space assigned to the disk directory, temporary, available, bad, and missing areas. The temporary areas are those which are allocated either to temporary files or to the virtual memory.

If files are created, extended, or deleted by the system during the processing of KA the map will not be accurate. It is therefore necessary that KA be run only when no other programs are in the mix.

The analyzed output will be to a line or console printer, and will print the areas in ascending disk address order associating with each area its first sector address, its length in sectors, and its status. The status will be either allocated, available, temporary, bad, or missing. If the area is allocated, the file name of the file to which the area is assigned will also be listed. If a particular file or family is not on-line, then this is indicated on the printout.

If the option DSKAVL is selected, then an analysis of the available areas on the disk specified by "disk-name" (or system disk if no "disk-name" was specified) will be printed.

Format:



Examples:

To analyze disk space assignments of all files on system disk:

KA =

To analyze disk space assignments of all files on the disk called PR2:

KA PR2/=

To analyze disk space assignments for a group of files beginning with the letters, "PR" on the system disk, and a file called PR200 on a disk called PR2:

KA PR= PR2/PR200

To analyze available areas on the disk called PRBU:

KA PRBU DSKAVL

Output format:

Six columns of information are output. The column headings, the format of the values these columns contain, and the significance of these values are as follows:

HEADING	VALUE	SIGNIFICANCE
AREA ADDRESS	8 digits 6 hex. digits	Sector address of start cf area
AREA LENGTH	8 digits 6 hex. digits	Number of sectors in this area
STATUS	9 characters	See Note 1
FILE NAME	12 characters	See Note 2

Note 1: The status will be one of AVAILABLE, 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 is ASSIGNED, then this field will contain the identifier of the file residing in the area. If a file belongs to a pseudo disk its disk name is also listed. Otherwise it will be blank.

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 processing of KA.

If fixed disk is being used, three areas are reserved for MTR purposes with the status marked as "BAD". The area lengths are 256, 128, and 128 sectors respectively.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
AREA APPARENTLY ASSIGNED TWICE	Area is cortaineo partly or completely in an area previously listed.	Notify field engineering. DO NOT USE DISK. It may be possible to recover any file not involved, usirg COPY.
AREA ASSIGNED BEYOND MAXIMUM ADDRESS	Area is assigned beyond the addressable space.	Notify field engineering. DO NOT USE DISK.
NOTE: OUTPUT FILES ON DISK WERE OPEN DURING THIS EXECUTION OF KA	Files were open on disk while KA was processing. Other programs may have been in the mix when KA was processing.	KA printout may not be completely accurate. Begin KA when no other programs are in the mix.
input NOT FOUND IN DIRECTORY ON THIS DISK	Specified file is not on disk.	Check input and reinput if necessary; Check for correct disk.

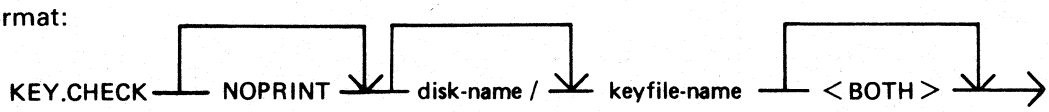
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
input NO FILES IN DIRECTORY FOR THIS FAMILY	Group of files specified is not on disk.	Check input and re-input if necessary; Check for correct cisk.
TABLE SIZE EXCEEDED	Number of lines of output requirec is greater than KA permits. Maximum permittec is about 141 pages of output. KA ends.	None.
NO OUTPUT GENERATED BY KA	Disk directory cont-ains no file names to print.	Check for correct cisk; Check input ano re-inpnt if necessary.

Note: Refer to "Common Utility Output Messages" for additional aid.

KEY.CHECK

This utility allows the operator to check and print the information on the validity of keys in an indexed pair of files.

Format:



This utility does not provide the * <file name> option in the initiating message.

During the execution of the utility the operator is informed if the data file has been updated via another key file or parity errors have occurred on the data file since creation of the specified key file. The checking is performed in two ways, Key file to Data file check, and Data file to Key file check.

In Keyfile to Data file checking, the key value field of each entry in the key file are compared with the key field in the corresponding record of the data file. This comparison will detect any changes to the keys in the data file records and disused entries in the key file.

In Data file to Key file checking, the key field of every non-deleted record in the data file will be checked to have an entry in the key file. The record written to the data file via another key file and records with invalid keys will be detected, as they will have no entry in the key file.

The NOPRINT option is provided to permit execution of the utility on a system with no printer. It is invoked by specifying NOPRINT at the beginning of the initiating message.

SPO messages are provided for resulting output from KEY.CHECK with the NOPRINT option specified. A message is displayed for only the first discrepancy the utility finds between the key file and the data file (see Output Messages).

If the <BOTH> option is specified in the initiating message, then Key file to Data file and Data file to Key file will be checked otherwise only Key file to Data file will be checked.

The utility will terminate if a parity error is encountered on the key file.

If the Generation number of the key file differs from that of the data file then a warning is printed. (Generation number is a field in the File Parameter Block, refer to MCP manual).

The Generation number of the key file will be modified to that of the associated data file on completion of a Key file to Data file check provided that the following conditions are satisfied:

1. Every key entered in the key file matches the key field in the associated data file record.
2. Every non-deleted data file record has a key entry in the key file.
3. The number of matched key entries in the key file is equal to the number of data file records with a key entry in the key file.
4. There are no parity errors on either the key file or the data file.

On completion of a printer listing execution of this utility, the total number of discrepancies found on checking each file is printed on completion of the key file and data file check.

Output Format:

The output format is self-explanatory.

Examples:

```
KEY.CHECK PQR
```

Check the key file PQR, performing Key file to Data file check only.

```
KEY.CHECK PQR <BOTH>
```

Check the key file PQR, performing Key file to Data file check as well as Data file to Key file check.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name IN USE	Another program is using the keyfile	Wait until job using keyfile has closed file, then re-input
file-name IS NOT A KEYFILE	Self-explanatory	Check the file-name and re-input
file-name IS A NULL-KEY KEYFILE	The specified file is a tag-file	See SOG section on the SORT for meanings: this utility cannot check tagfiles
file-name <DATAFILE> NOT FOUND	The data file corresponding to the keyfile cannot be found	(1)Enter LR disk-name/keyfile-name. (2)Check the disk name of the data file in LR listing. (3)If the disk file name is the same as the keyfile then copy data file from backup and re-input. (4)If the dis name is different then load that disk and re-input.
file-name <DATAFILE> IN USE	The corresponding data file is being used by a program	Wait until the other job has closed the data file, then re-input
file-name INCOMPATIBLE WITH KEYFILE	The key-offset plus key-length in the KFPB of the specified keyfile is greater than the record size of the data file	none
READ ERROR ON KFPB OF file-name	Parity errors have been encountered: the utility cannot continue.	(1)Attempt to copy files to another disk (2)Run CHECK.DISK to check the integrity of the disk
READ ERROR ON KEYFILE file-name	same as above	same as above

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
UNABLE TO OPEN file-name <INDEXFILE>	<BOTH> has been given in the initiating message and the utility is unable to open the keyfile and datafile as an indexed pair after completion of the keyfile-to-datafile check	none
=INVALID KEY= KEY IN DATA RECORD: invalid key RECORD datafile record number	A data record has occurred with an invalid key.	None.
file-name FILETYPE IS NOT DATA	The file identified in the KFPB of the specified keyfile as the associated datafile does not have a filetype of 2002.	Check the file-name and re-enter.

Note: Refer to "Common Utility Output Messages" for additional aid.

If NOPRINT is specified in the initiating message, the following messages can be displayed by the utility:

NOTE

As the following messages are for information only, no suggested corrective action is given.

MESSAGE	POSSIBLE CAUSES
KEYFILE keyfile-name PERTAINS TO DATAFILE file-name	Shows which data-file is related to which key-file.
WARNING - GENERATION NO.MISMATCH BETWEEN KEYFILE keyfile-name AND DATAFILE data-file-name	The generation numbers of the file are inconsistent.
FIRST DISCREPANCY FOUND BETWEEN KEYFILE keyfile-name DATAFILE datafile-name	The first error has been encountered between the key and data-files.
x DISCREPANCIES FOUND WITH KEYFILE keyfile-name	The check on the keyfile has been completed and x number of discrepancies have been found between the keyfile and its associated datafile.

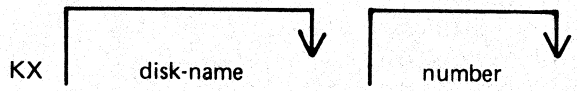
MESSAGE	POSSIBLE CAUSE
KEY.CHECK ON keyfile-name COMPLETED	<BOTH> was not specified in the initiating message. No checking will be invoked on the datafile and the utility will go to End of Job.
KEY.CHECK COMMENCING ON DATAFILE filename	<BOTH> was specified in the initiating message and the datafile check has been invoked.
y DISCREPANCIES FOUND WITH DATAFILE filename	The check on the datafile in relation to the keyfile has been completed and y discrepancies have been found.
KEYFILE keyfile-name IS CONSISTENT WITH DATAFILE file-name	No discrepancies have been found on either the data-file or the keyfile. If the generation numbers of the files did not match that of the datafile then either "GENERATION NO. OF keyfile name MODIFIED TO Z" or "UNABLE TO MODIFY GENERATION NUMBER OF KEYFILE keyfile-name" will be displayed and the utility will go to End of Job.
KEYFILE keyfile-name IS NOT CONSISTENT WITH DATAFILE datafile-name	Discrepancies have been found on one or both files. The utility will go to End of Job.

KX (Disk Allocation Information)

(a function of SYS-SUPERUTL)

This function will allow the operator to display the names of all the files found on the disk specified by "disk name" (or on the system disk if no "disk-name" is specified) whose total number of sectors allocated is equal to or greater than "number" (assumed zero if not specified). The disk concerned may not be a pseudo pack.

Format:



After each display, which will include the information of the current numbers of temporary and available sectors, the KX function of SYS-SUPERUTL remains available, waiting for one of the following input responses:

A call on any other function of SYS-SUPERUTL: this will terminate KX.

KX or KX NEXT To display the next file name, if any, otherwise KX will go to END.

KX RM or KX REMOVE To remove the file whose name has just been displayed.

KX END To terminate KX.

Examples:

To display the name of the first file on the system disk whose size is equal to or greater than 250 sectors:

KX 250

To display the name of the next file whose size is equal to or greater than 250 sectors:

KX NEXT

To remove the file just displayed:

KX RM

To terminate KX:

KX END

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
END KX	KX successful	None.
filename - number SECTORS; AV. number, TEMP (or TEMPORARY) number	Normal KA display showing current number of sectors, available and temporary on disk.	None.
input IS AN UNACCEPTABLE RESPONSE FOR KX	Typing error.	Check input and re-enter.

See SYS-SUPERUTL messages also.

LB (Look Back in Log)

(a function of SYS-SUPERUTL)

This function will Look Back to continue recall in the direction of earlier messages with a screenful of messages. If the serial printer (SPA) is used as the console, then the function will display a number of messages calculated by the length of messages and width of console.

Format:

LB

Example:

To look back and display the messages:

LB

LB can be initiated only after IR, LB, and LF.

Output messages

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
IR REQUIRED BEFORE LB CAN BE DONE	IR function was not initiated before LB.	Initiate IR then re-input LB.
CANNOT RECALL LB BEYOND THIS POINT	LB has reached the beginning of this log file	None

LD (Tape Library Utility)

This utility allows the operator to maintain library tapes. It is divided into the following four separate "sub-programs" (functions):

ADD (tape-to-disk file copy)
LOAD (tape-to-disk file copy; duplicates are removed from disk)
DUMP (disk-to-tape file copy)
UNLOAD (disk-to-tape file copy; files are deleted from disk after being copied to tape)

LD will attempt to open SYSMEM on all PPIT listed units for directory scanning. It will check the PPIT entry for System Pseudo disk-name.

On the B 80 these four functions can be invoked directly: the MCP will recognize that they are part of the LD program and load LD from the system disk, passing LD the appropriate information. For example, the input

DUMP TO ARTAPE AR= causes the same action as
LD DUMP TO ARTAPE AR=

If LD does not reside on the system disk, the user-disk name and 'LD" must be specified.

Should the operator request a "mix message" (see MIX intrinsic) while any of the 4 functions are running, "LD" (not the name of the specific function) will appear in the mix.

Similarly, to discontinue any of the four functions, a message of: "DS mix number/LD" will be required.

Detailed descriptions of ADD LOAD, DUMP, and UNLOAD and associated output messages are provided under the name of the function.

DUMP to [TAPE NAME] TAPE

LF (Look Forward in Log)

(a function of SYS-SUPERUTL)

This function will look forward to continue recall in the direction of later messages with a screenful of messages. If the serial printer (SPA) is used as the console, then the function will display a number of messages calculated by the length of the messages and the width of the console.

Format:

LF

Example:

To look forward from last recall and display messages:

LF

LF can be initiated only after IR, LB, and LF.

Output messages

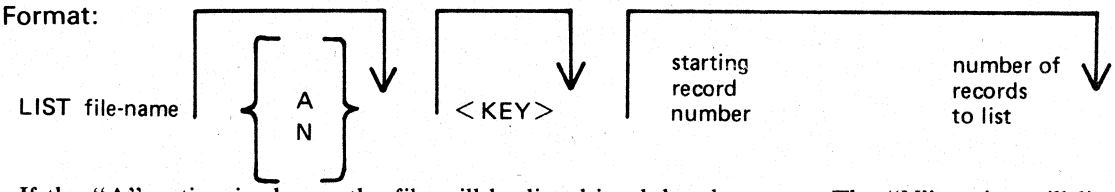
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
IR REQUIRED BEFORE LF CAN BE DONE	IR was not initiated before LF	Initiate IR and then re-enter LF
CANNOT RECALL LF BEYOND THIS POINT	LF has displayed the most recent entry in the log file	None

LIST (File List)

This utility allows the operator to list in whole or in part files on any CMS device. Output will be either to the line printer or to the console printer.

LIST will attempt to open SYSMEM on all PPIT listed units for directory scanning. It will check the PPIT entry for System Pseudo disk-name.

Format:



If the "A" option is chosen the file will be listed in alpha characters. The "N" option will list the file entirely in hexadecimal. If neither the "A" nor "N" options are selected, the file will be listed in both alpha and hexadecimal.

If the file to be listed is a keyfile, the utility will list the associated data file in the order of the keyfile unless the <KEY> option is specified. When the <KEY> option is used, the utility will list the keyfile itself.

The operator may also list selected parts of a file by specifying the relative record number at which printing should begin and the number of records to be printed from that point.

Examples:

To list the records of a file called PROGSRC as alpha:

LIST PROGSRC A

To print the first record only of a file called PR200 in hexadecimal:

LIST PR200 N 1 1

To list records 100 through 149 of PROGSRC as alpha:

LIST PROGSRC A 100 50

To List Keyfiles:

Assume there is a keyfile called PR200K which refers to a data file called PR200.

The statement

LIST PR200K N <KEY> will list all records of the keyfile PR200K in hexadecimal.

The statement

LIST PR200K N will list all records of the data file, PR200 in keyfile order in hexadecimal.

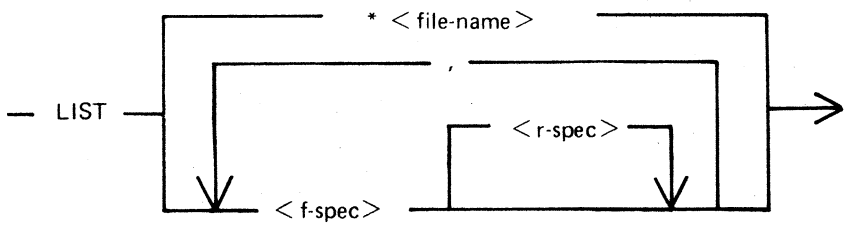
Additional Capabilities

Further features of this utility are summarized in the railroad chart given in figure 4-4, which gives the complete input specifications.

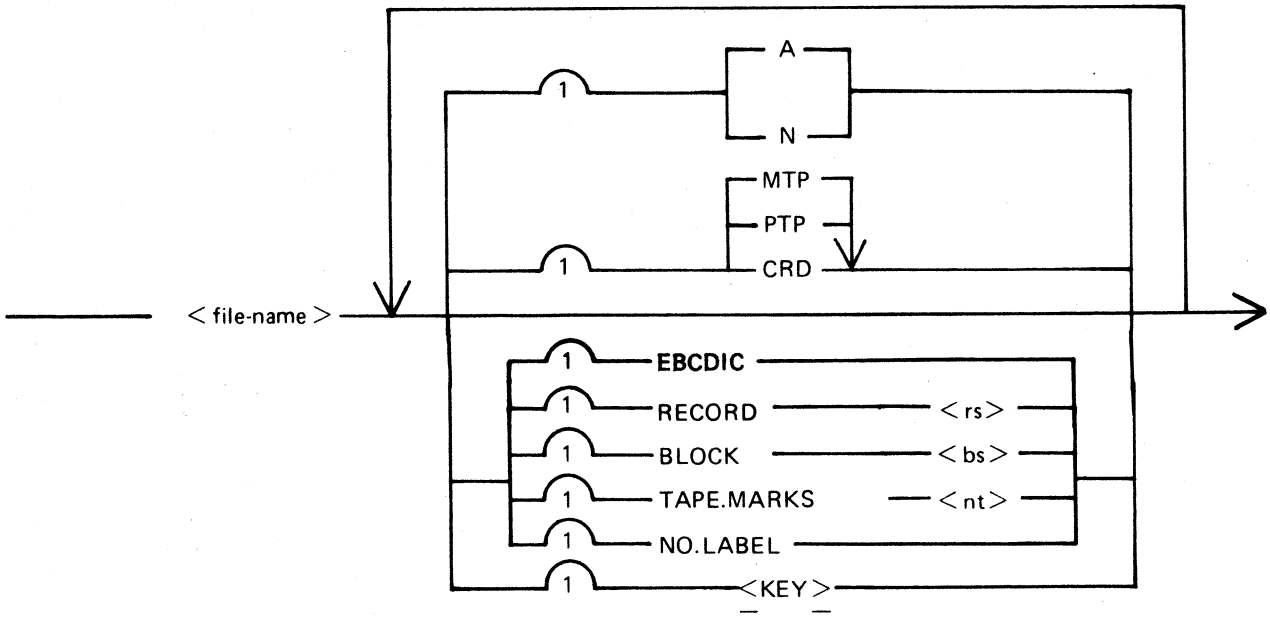
Non-disk files

Files on media other than disk may be listed. Abbreviations for valid devices are as follows:

MTP – magnetic tape or cassette
CRD – punched cards
PTP – paper tape



< f-spec > is defined as :



< r-spec > is defined as :

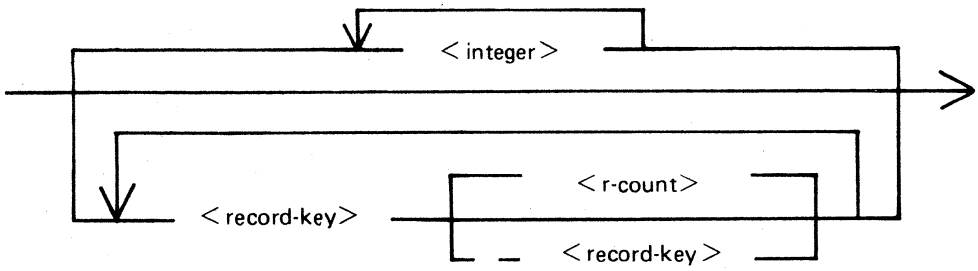


Figure 4-4. Railroad Chart for List Utility

Examples:

To list a cardfile called PRFILE in alpha:

LIST PRFILE CRD A

To list the first 10 records of a CMS labelled magnetic tape called PRTAPE:

LIST PRTAPE MTP 1 10

(Note: this assumes record size of 180 bytes). The tape or cassette to be listed should be a tape created by the COPY utility. Library tapes and non-CMS tapes should be treated as unlabelled (see below).

Unlabelled tapes

Input tapes having no CMS labels ("unlabelled" tapes) may be accessed by the LIST utility.

The NO.LABEL option allows the listing of unlabelled files. Upon recognizing an unlabelled file, the MCP will print a "DEVICE REQUIRED" message. The operator must then respond with an appropriate "AD" input message (see "AD") to identify the unlabelled file.

The end of file recognition for unlabelled files is determined by tapemark count. The TAPE.MARKS option allows the operator to specify the total number of tapemarks 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. Therefore, a standard labelled CMS file will be listed up to, but excluding, the trailing label if NO.LABEL and 2 tapemarks are specified. (A labelled CMS file consists of "Label, Tape mark, data, tape mark, label'). The operator must be aware of the format of any file which is to be listed when using the NO.LABEL option.

If the RECORD size is not 180 bytes, refer to the section on Record/Block modifications.

Example:

To list the first file of a magnetic tape with non-standard label (the format being: label, tapemark, data, tapemark):

LIST TP MTP NO.LABEL **TAPE.MARKS** 2

Note: MCP will issue a message asking for unlabelled tape TP. Operator must respond with "AD" input. Additionally, the first line of the listing contains a list of the non-standard label.

Record and block sizes

The listing is record-oriented. The following record sizes are assumed:

Disk = (Disk File Header) from file itself

Labelled tape/cassette = from tape label

Unlabelled tape = 180 bytes

Cards = 80 or 96 bytes depending on device.

If different values are required Record and Block sizes may be specified.

Example:

To list an unlabelled tape containing 10-byte records with 10 records per block:

LIST TP MTP NO.LABEL TAPEMARKS 2
RECORD 100 BLOCK 1000

If EBCDIC is specified, the input will be translated from EBCDIC coding, otherwise ASCII is assumed.

For magnetic tape or cassette files the record size must be specified if it is greater than 1024 characters, otherwise the utility will not be able to read this file and therefore no list will be produced. If the record size is specified and no block size is specified then the block size will be set to the same as the record size. For unlabelled files the default record and block sizes are 180 each.

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 specified must be an integer multiple of the record size. The utility will attempt to identify inconsistencies when using labelled CMS files. Any inconsistency not isolated by the LIST will cause MCP to discontinue (DS/DP) the utility.

Selected file list

More than one selected portion of the input file may be listed. Pairs of numbers may be specified within each pair the first number specifies a relative record number and the second specifies number of records to be listed. If an extra number is specified the last number specifies listing from that record to the end of file.

Example:

To list records 100 to 149, 300 to 499, and 1000 to end of file.

LIST FILE1 100 50 300 200 1000

Selected indexed file list

For indexed files, listing of records can be selected based on content of the key. There are 2 options: the number of records can be specified or an ending key value.

Examples:

PQR is a keyfile containing personnel records. To list 15 records from the corresponding data file starting from the record with personnel number 01786:

LIST PQR 01786 15

Using same keyfile to list all data records from personnel number 01786 to 18000:

LIST PQR 01786 – 18000

Note: the second option is specified by the hyphen in the LIST statement. Note that at least one space is required before and after all key values (personnel numbers in this case).

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
number number IN filename NOT LISTED	Error found in pair of numbers for selected record list option. One record number in the pair indicates a section of the file at a lower file address than a previously specified section. This pair is ignored by LIST.	Check input and re-enter.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
filename EXHAUSTED DURING number number	End-of-file was encountered while the sectior of the file indicated was being copied.	Check input and re-erter if necessary.
filename EXHAUSTED DURING RANGE record-key record-key or filename EXHAUSTED DURING RANGE record key number	End-of-file encountered while section of file indicated is being listed; or no records were found in range of the 2 record keys.	Check input and re-erter if necessary.
NO RECORDS FOR LISTING FROM filename	Specified file contains no records to list.	Check for correct file-name.
filename NOT ACCEPTABLE - RECORD SIZE OF number EXCEEDS MAXIMUM FOR THIS RUN - RESUBMIT	LIST has encountered a file witin a record size greater than expected. This can happen if a magnetic tape file with a record size greater than 1024 characters is submitted to the utility without the record size being properly specified in the initial input.	Check input and re-enter.
KEYFILE file-name OR DATA FILE NOT FOUND	Utility cannnot locate keyfile or data file pertaining to the keyfile specifications.	Check for correct medium;
BAD ATTRIBUTE SPECIFIED	The number specified after BLOCK is not an integer multiple of the number specified after RECORD.	Correct the input and re-input.
PERMANENT ERROR ON INPUT FILE file-name (NO. number or KEY key)	The irrecoverable error, e.g. parity error has been found.	None.

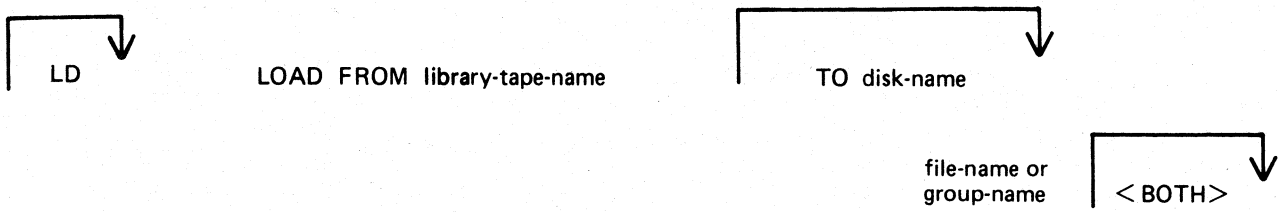
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SELECTIONS OF RECORDS FROM number IGNORED	The file reached end before the record selected.	None.
RECORD SIZE TOO GREAT	The specified record size is greater than 10000.	None.
INVALID CHAR. IN IDENTIFIER file-name - WARNING	A tape with an illegal name is specified.	Correct the name and re-input.
KEYFILE file-name OR DATA FILE IN USE	The specified keyfile and its data file are on line, but one or both are in use by a task which will not permit sharing.	Wait until the task using keyfile or data file goes to end of job.

Note: Refer to "Common Utility Output Messages" for additional aid.

LOAD (Load Library Tape Files to Disk)

This function, a part of the LD utility, allows the operator to copy files from a library tape to disk. As files are copied to disk, any duplicate files will be automatically removed from disk by the function.

Format:



If the <BOTH> option is used immediately after a request to copy a keyfile, the associated data file will also be copied, provided that the data file does not precede the keyfile on the library tape. The keyfile will be altered to refer to the disk which now holds the data file (rather than the disk from which the data file was dumped).

Examples:

To copy all files from a tape called PRTAPE to a system disk:

LOAD FROM PRTAPE =

To copy the file called AR300 from a tape called ARTAPE to a disk called ARBU:

LOAD FROM ARTAPE TO PRBU AR300

To copy files called DCSTSK, and PRTASK from a tape called PRTAPE

LD LOAD FROM PRTAPE DCSTSK PRTASK

To copy from a tape called PRTAPE the keyfile called PR200K and its associated data file to the system disk:

LOAD FROM PRTAPE PR200K <BOTH>

Since "LOAD" is a part of the utility "LD", "LD" is actually what will appear in a mix message. To discontinue the LOAD function, "DS mix number/LD".

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name LOADED	LOAD successful	None.
file-name REMOVED	LOAD successful: original file on disk was removed.	None.
library-tape-name NOT A RECOGNIZED DUMP TAPE	Tape does not have a recognizable header.	Provide utility with correct tape; DS the LD utility

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO FILES IN THE FAMILY group-name ON TAPE library-tape-name FOR LOAD	Specified group was not found on disk.	Check input and re-enter if necessary; Check for correct tape.
NO FILE file-name ON TAPE library-tape-name FOR LOAD	Specified file was not found on tape.	Check input and re-enter if necessary; Check for correct tape.
file-name LOAD/DUMP DISCREPANCY	End of File encountered before expected. Disk File Header may contain errors.	None.
NO FILES TO LOAD	No files were found on this tape to copy.	Check input and re-enter if necessary; Check for correct tape.
file-name DATA FILE NOT FOUND ON TAPE FOR LOAD	Data file for this keyfile does not follow on tape. It cannot be copied. Keyfile was copied.	Check if data file appears before key file on tape (use TAPEPD); if so, load full tape to disk.

Note: Refer to "Common Utility Output Messages" for additional messages.

478
102

LOAD.VFU (Load Vertical Format Unit)

This utility allows the operator to define the page format on a line printer that contains a soft vertical format unit.

Format:

```
                 DEFINE   format-name number-1 [, number-2 [number number]]
                   or
                 FORMAT   LPx      WITH format-name
LOAD. VFU          or
                 LIST     [ALL] [format-name]
                   or
                 DELETE   format-name
```

The utility may be used to define vertical format unit formats and store them in a library file SYSVFU.LIB. These formats can be subsequently selected by name to be loaded to the specified printer (type A 9249-30/50). The utility zips the VF intrinsic which performs the actual loading of a format string.

When using the DEFINE function, if the file SYSVFU.LIB does not already exist, the utility will create this file and an entry will be made for the format being defined. Otherwise, the file will be updated to contain the newly defined format. The number-1 is the page height in lines and number-2 is the end of page line. The pair of numbers consists of a channel number (2-11) followed by a 'skip to line' number (not greater than the page height).

Example:

LOAD.VFU DEFINE PAYROLL 66, 60, 2 10, 4 20

This format is equivalent to:

Page Height = 66
End of Page = 60
Channel 2 has associated line number 10
Channel 4 has associated line number 20

This format is also equivalent to figure 4-5, which illustrates a paper vertical format tape on other line printers.

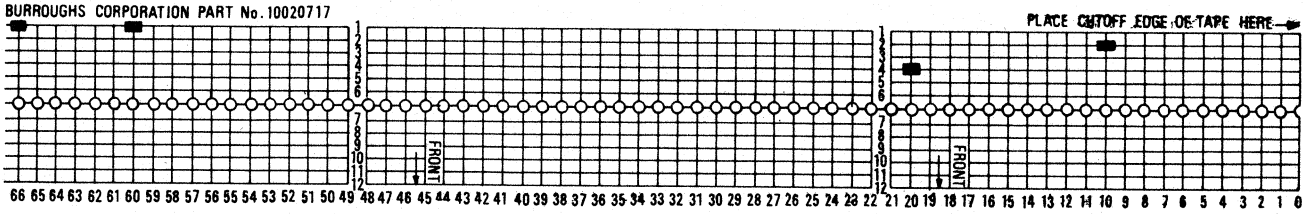


Figure 4-5. Paper Vertical Format Tape

Limitations:

The utility does not check the following:

1. For each channel, no more than 4 line numbers can be specified.
2. Page height defined should not be greater than 94 lines.

The FORMAT function allows the operator to load a predefined VFU format to a specified printer. The printer must be ON LINE and NOT IN USE.

Example:

To load the format PAYROLL defined earlier on LPA:

LOAD.VFU FORMAT LPA WITH PAYROLL

The LIST function has three possible options:

LOAD.VFU LIST
will list all format-names defined in SYSVFU.LIB;

LOAD.VFU LIST format-name
will list the format string of specified format-name as defined in SYSVFU.LIB;

LOAD.VFU LIST ALL
will list all format-ids and strings as defined in SYSVFU.LIB.

The DELETE function is used to delete a predefined VFU format from the SYSVFU.LIB file. If the format being deleted is the only defined format in SYSVFU.LIB, then the SYSVFU.LIB file will be removed.

Example:

To delete predefined VFU format PAYROLL from the SYSVFU.LIB file:

LOAD.VFU DELETE PAYROLL

NOTE: The default values of page height and end of page are 66 and 60 respectively. Non-default values require that the format should be loaded prior to the execution of the program.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SYSVFU.LIB NOT FOUND	Self-explanatory	Use DEFINE function to create library file
SYSVFU.LIB IN USE	Another copy of LOAD.VFU is running	Use only one copy of the utility
READ ERROR ON SYSVFU.LIB	Parity error on the library file	
WRITE ERROR ON SYSVFU.LIB	Parity error on the library file	
format-name NOT FOUND IN SYSVFU.LIB	Requested DELETE, FORMAT or LIST cannot find specified format-name	Check input and re-enter if necessary: use DEFINE function to create format-name
format-name ALREADY DEFINED IN SYSVFU.LIB	Cannot DEFINE two formats of the same name	Re-input using different format-name
format-name DEFINED IN SYSVFU.LIB	DEFINE function has been successful	none
format-name DELETED FROM SYSVFU.LIB	DELETE function has been successful	none

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SYSVFU.LIB DELETED	A DELETE function has deleted the only remaining format in the library file, and the file has been removed	none
PRINTER peripheral FORMATTED WITH format-name	FORMAT function has been successful	none
PRINTER peripheral NOT FORMATTED WITH format-name	The utility has found the specified format in the library, but the zip of the VF intrinsic was unsuccessful (this will be preceded by output from VF)	refer to the VF error message

LR (List Directory)

This utility allows the operator to print detailed information about particular files or groups of files on disk.

If a file has areas on an associated overflow disk, then the disk name of the overflow disk is printed beside each relevant area address and size. Note that the addresses for the areas on an overflow disk are not necessarily correct.

If a particular file or group is not found on a specified disk, this is indicated on the listing.

If "<ASCENDING>" or "<A>" is selected, the utility will print the information requested in ascending order of file-names.

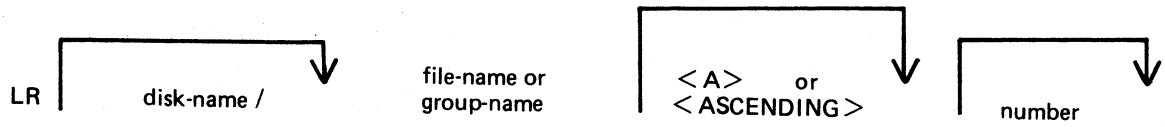
If "number" is specified after the LR of an entire disk (that is, LR ARDISK2/=), then LR will only print information about those files whose total number of sectors allocated is greater than "number'; this will be followed by a listing, with totals, of all available and temporary areas on the disk.

An output line concerning a keyfile will normally be followed by a second line showing the name of the data file to which this keyfile points and the key offset and length.

The heading lines printed at the top of each page will provide a good deal of information about the disk itself.

When handling Pseudo-pack disks, the utility will attempt to open SYSMEM on all PPIT listed units for directory scanning and will search for a PPIT entry with a tag of 20 for system Pseudo disk-name.

Format:



Examples:

To print the entire directory of the system disk:

LR =

To print the entire directory of ARDISK2 in ascending order:

LR ARDISK2/= <A>

To print information about the file called "AR200" and a group of files beginning with the letter "C" only:

LR C=, AR200

To print information only about files on the system disk which have been allocated greater than 1000 sectors:

LR = 1000

Output format:

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

HEADING	VALUE	SIGNIFICANCE
FILE NAME	12 Characters	File name
ACTUAL SIZE	7 digits	Number of records currently contained in file
MAXIMUM SIZE	7 digits	Maximum number of records which file may contain
RECORD SIZE	5 digits	Number of bytes per record
RECORDS BLOCK	5 digits	Number of records in each block
CREATED	5 digits	File creation date (Julian YYDDD)
ACCESSED	5 digits	Last access date (Julian YYDDD)
GEN. NO.	3 digits	
FILE TYPE	7 characters, 2 hex. characters	See Note 1
NO. AREAS	2 digits	Number of areas currently allocated
AREA LOCATIONS	8 digits 6 hex. characters 7 characters	See Note 2
AREA SIZES	8 digits 6 hex. characters	See Note 2

Note 1: The actual filetype of each file is displayed with type identification as follows:

FILETYPE	ENTRY IN LISTING	
@00@	DATA	@00@
@01@ – @0E@	SRCLANG	@0x@
@0F@	SRCLIBR	@0F@
@10@ – @13@	CYYMMDD	@1x@
@21@	SYSLANG	@21@
@22@	SYSDATA	@22@
@30@	VIRTMEM	@30@
@31@	SYSLOG	@31@
@81@	KEY	@81@
@A0@	PRNTBKP	@A0@
Any other	SYSTEM	@xy@
	(YYMMDD = the compile date and SYSDATA could be SYSCONFIG or SYSVFU.LIB)	

Note 2: For each file the area addresses and sizes of allocated areas will be printed in these columns. For areas on an overflow disk the overflow disk name will follow the area location.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
TABLE SIZE EXCEEDED	The number of file-names to be sorted by LR in an ASCEND-ING or A request has exceeded the maximum permitted (254).	None.
input - NOT FOUND IN DIRECTORY OF THIS DISK	File specified is not on the disk	Check the correct disk and re-enter if necessary
input - NO FILES IN DIRECTORY FOR THIS FAMILY	Group of files specified is not on disk	Check input and re-enter if necessary. Check for correct disk.

Note: Refer to "Common Utility Output Messages" for additional aid.

MODIFY (Program Code File Modification)

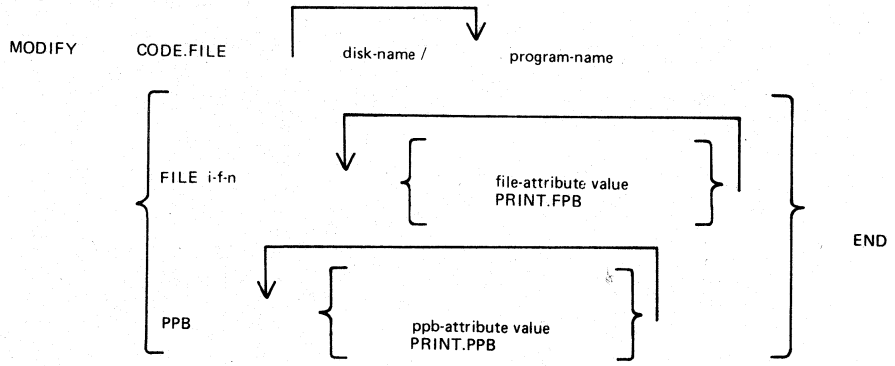
This utility allows the changing of a number of file attributes within the file parameter block (FPB) and program attributes within the program parameter block (PPB) of a code file. It should not be used unless the meaning of each attribute is thoroughly understood. Refer to the CMS MCP manual for more information on FPB and PPB formats.

The utility operates in an interactive manner using a console file if no further information is provided when initating the utility, thus:

MODIFY

For details of the interactive mode, see later. Specifications can be entered when starting the utility. The name of the code file to be modified is preceded by the keyword "CODE.FILE". The word "CODE.FILE" can be omitted from the first element of the initiating message. Following the code file name is either the keyword "FILE" to enable file attributes to be modified, or the keyword "PPB" to enable program attributes to be changed. The file whose attributes are to be changed is specified by the internal file name (i-f-n) as given by the program source code listing. The i-f-n is determined by the programmer. Additional keywords are "PRINT.FPB" and "PRINT.PPB" to print the complete FPB and PPB respectively. The complete specifications to the utility are terminated by the keyword "END".

Format:



The commas are optional, but may be used to improve readability. See later for the list of attributes and allowable values.

When a PPB is being printed, MODIFY checks the release level of the code file, printing the S-LANGUAGE for pre-3.3, and MCP level and HARDWARE type for post-3.2. It should be noted that the MCP level printed for certain released code files is not that of the host system MCP, but that of the system on which the code files were compiled.

If an unknown machine type is encountered, "UNKNOWN" is printed.

The HARDWARE type is indicated as follows:

BYTE VALUE	TYPE
1 – 10	B 80, B 90
11 – 20	B 776
21 – 30	B 800
31 – 40	B 900
41 – 50	B 1700
51 – 60	B 1800

The second byte of the field in the PPB is machine-dependent and is therefore not printed).

Examples:

To modify the value of FID (file-id) and change the device kind of a file whose internal name is INFILE in a program code file COPY on disk SYS2:

MODIFY CODE.FILE SYS2/CCPY, FILE INFILE FID CARDS DEVICE CR, END

To change the value of CONTROL.STACK to 50 in code file AR768 on disk AR1, and print the resultant PPB:

MODIFY AR1/AR768 CONTROL.STACK 50 PRINT.PPB END

Interactive Mode

If no initiating specifications are given, PKs 1 to 6 are lit for various functions.

PK1	PK2	PK3	PK4	PK5	PK6
help	modify PPB	modify FPB	specify code file	print FPB or PPB	ECJ

Pressing PK1 gives a display of the meanings of the 6 PKs, as shown above, followed by the request

CODE.FILE?

Enter the code file name, followed by OCK1. The utility requests

SELECT FUNCTION

and lights appropriate PKs. While any relevent PK is lit, the corresponding function can be started.

If PK2 (modify PPB) or PK3 (FPB) is depressed, the utility requests

PPB ATTRIBUTE or

FPB ATTRIBUTE

Enter the name of the attribute, as given in the table later. The utility displays the current value, then requests

NEW VALUE.

Enter the new value required. The utility then returns to the select-function loop.

File Attributes

Table 4-4 gives the keywords for each file attribute that can be changed by the MODIFY utility, together with allowable values for each attribute. Table 4-5 gives the keywords of each PPB attribute that can be changed, and allowable values for each.

Note that each modification is performed in turn, so that the keywords PRINT.FPB and PRINT.PPB will reflect the FPB and PPB after any modifications specified previously in the message to MODIFY, but before any modifications are made that are specified after the print request.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ATTRIBUTE VALUE MISSING	Value is either missng or incorrect; other modifications carried out.	Check current values by PRINT.FPB or PRINT.PPB; then use utility for the attribute in error.
KEYWORD IN ERROR	Self-explanatory; other modifications carried out.	As above.
ATTRIBUTE-VAL INCONSISTENT	The attribute being assigned cannot take the value being given; other modifications carried out.	As above.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
INCORRECT ATTRIBUTE	A value is being assigned to a value, rather than to an attribute; other modifications carried out.	As above.
DEVICE - MYUSE INCONSISTENT	Incompatible values of these file attributes. This is a warning that the program may give an error when executed.	Use MODIFY to correct either or both of these fields.
FILE-SIZE TOO LARGE	Value for FILE.SIZE is incorrect. This is a warning.	Check with PRINT.FPB if necessary and correct the attribute.
TOO MANY BUFFERS	Value for NO.BUFFERS is incorrect. This is a warning.	As above.
REC. NOT INTEGRAL OF BUF.	The buffer size is not an exact multiple of the record size. This is a warning.	Use MODIFY to correct one or both of these attributes before running program.
CODE FILE NAME IN ERROR	Self-explanatory; all modifications are ignored.	Re-input.
FILE NAME NOT FOUND	The internal file name is not in code file; all modifications are ignored.	As above.
CURRENCY SYMBOL EXPECTED	Self explanatory	Re-input.
NUMERIC ATTRIBUTE - VAL REQD	Non-numeric characters were input where a numeric value is needed.	Check with tables 4.2 and 4.3 and re-input.
FILE NOT SPECIFIED	Missing keyword "FILE"	Re-input.
PPB NOT SPECIFIED	Attempt to modify PPB while not in PPB mode.	Re-input.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NOT AN INDEXED FILE	Attemped to use an indexed-file attribute on a non-indexed file.	Check initial input to MODIFY
NOT A COBOL PROGRAM FILE	Wrong PPB attribute used (see table)	Re-input.
NOT AN MPL PROGRAM FILE	Wrong PPB attribute used (see table)	Re-input.
CODE FILE - BAD FILE TYPE	Attemptec to modify a coce file wich was not of type CODE.	None.
CODE FILE IN USE	The specifiec code file is in use.	Wait until the task using code file goes to end of job.
CANNOT OPEN THAT CODE FILE	Attempted to modify a code file which was not available for some reason.	None.
NUMBER TOO BIG	Attempted to assign a value greater than 65535.	None.
MUST HAVE 0 < KEY LENGTH < 29	Attempted to assign a KEY LENGTH value out of range.	None.

Table 4-4. File Attributes Accessible by MODIFY

file attribute name	allowable values
MFID	1-7 alphanumeric characters
FID	1-12 alphanumeric characters
REEL	3 decimal cigits in range 000-999
DEVICE	one of the mnemonics given in Table 4-6
RECORD	1-5 decimal digits in range 0-65535
BUFFER	1-5 decimal digits in range 0-65535
FILESIZE	1-7 decimal cigits in range 0-1048560
NO.BUFFERS	1-2 decimal digits in range 1-16
CYCLE	2 decimal digits in range 00-99
FORMS	ON, OFF
SET.UPDATE	ON, OFF
NO.LABEL	ON, OFF
CONDITIONAL	ON, OFF
SINGLEAREA	ON, OFF
GEN.CHECK	ON, OFF
NO.REWIND	ON, OFF
REVERSE.CHECK	ON, OFF
CLOSEMODE	LOCK, PURGE, REMOVE, RELEASE, HALF.CLOSE
CRUNCH	ON, OFF
MERGE	ON, OFF
OTHERUSE	FREE, LOCK.ACCESS, LOCKED or SHARED
MYUSE	INPUT, OUTPUT, IO
EXTEND	ON, OFF
ACCESSMODE	SEQUENTIAL, STREAM, RANDOM
GEN.NO	1-5 decimal digits in range 0-65535
LAST.ACCESS	5 decimal digits in Julian date format, YYDDD
SAVE	1-3 decimal digits in range 0-999
FILE.DEFAULT	TYPE1 thru TYPE29 (see MPL Reference Manual)
NON.STANDARD	ON, OFF
D.MFID	1-7 alphanumeric characters (indexed files only)
D.FID	1-12 alphanumeric characters (indexed files only)
ROUGH.TABLE	1-5 decimal digits in range 0-65535 (indexed files only)
KEY.LENGTH	1-2 decimal digits in range 1-28 (indexed files only)
KEY.OFFSET	1-5 decimal digits in range 0-65535 (indexed files only)

Table 4-5. PPB Attributes Accessible by MODIFY

PPB attribute name	allowable values
INTERP.PACK	1-7 alphanumeric characters
INTERP.NAME	1-12 alphanumeric characters
CLASS	A, B, C
EOJ.SUPPRESS	ON, OFF
DATA.STACK	1-5 decimal digits in range 0-65535 (MPL/BIL programs only)
CONTROL.STACK	1-5 decimal digits in range 0-65535
CURRENCY.SYMBOL	one character (COBOL/RPG programs only)

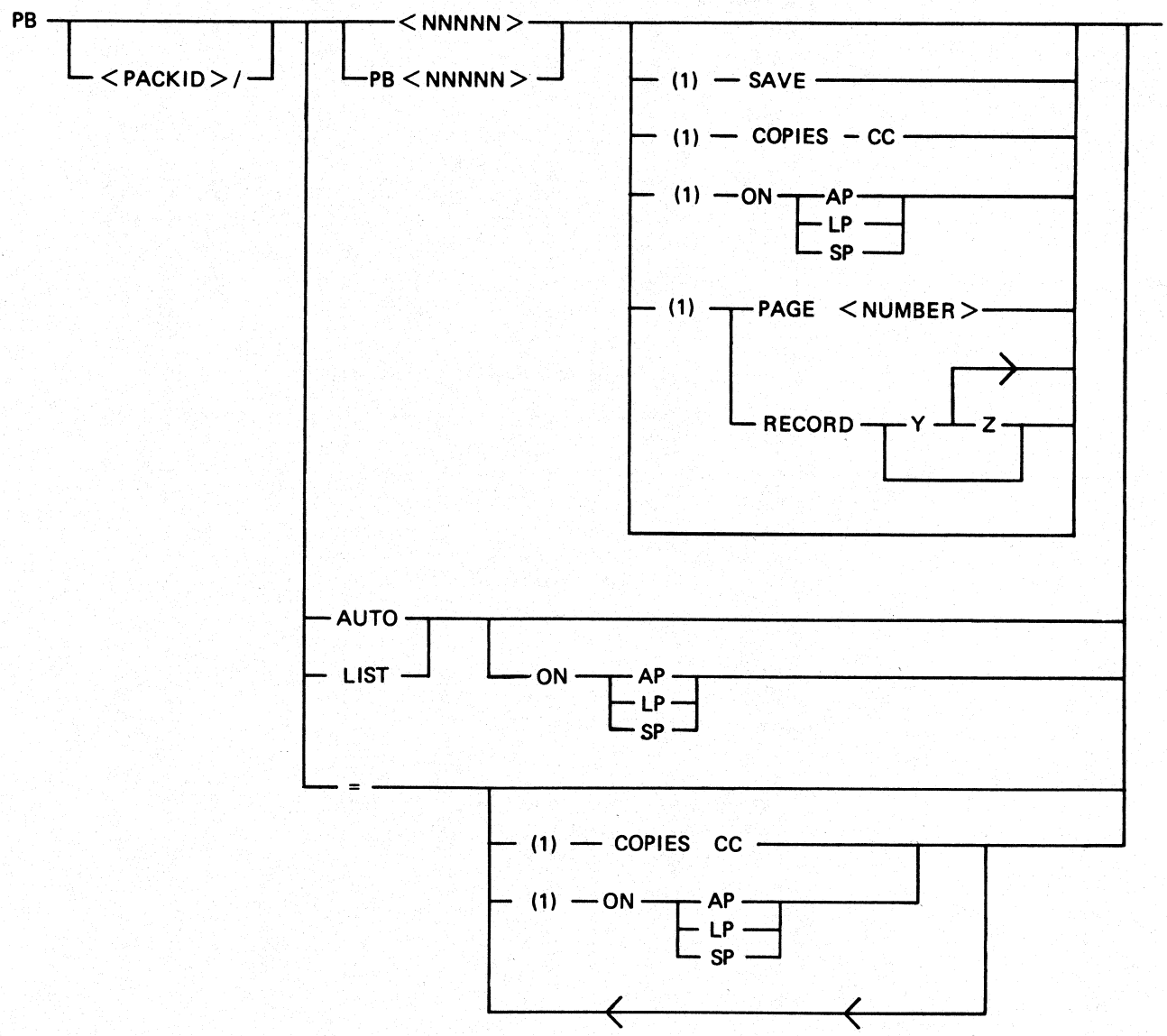
Table 4-6. Mnemonics for Device Attribute for MODIFY

mnemonics	meaning
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
CSIN	magnetic tape cassette without write permit
DC	any disk
DM	Any mini-disk

PB (List Printer Backup Files)

The CMS Printer Backup utility (PB) is an MPLII program which allows files created in accordance with the CMS Printer Backup facility to be printed by CMS users.

The syntax of the PB initiating message is:



PB Initiating Message Parameters

<PACKID>/ — Identification of pack on which the backup file resides. Contains one through seven characters. If no <packid> is specified, the system status communicate is invoked, determining the name of the Printer Backup Designate Disk (PBDD) as specified by the SO SCL. If no PBDD is in use, default is to the system disk. The <packid> must always be terminated with a slash.

PB<nnnnn> — Filename of printer backup file to be printed. Contains one to five decimal digits. If the user omits the letters "PB", they are appended to the front of the digits, forming the filename. In either case, the file must be of printer backup file-type to be printed by this utility. Leading zeros are not required in the PB<nnnnn> filename.

=	Specifies that the family of all files identified by a "PBnnnnn" filename will be printed. Only a file which is of printer backup file-type can be printed by this utility.
SAVE	Prevents the removal of the file from disk after printing is completed. If "SAVE" is not specified, removal occurs.
COPIES	Specifies a non-zero decimal number containing at most two digits. "COPIES cc" specifies the number of copies to be printed. The default value is one.
RECORD	Selects specific parts of the file for printing by giving a starting record number (that is, "Y") and the number of records to be printed (that is, "Z"). More than one range of records may be specified, and overlapping of record number pairs may exist. NOTE: When there is no quantity (that is, "Z") in a record number pair, printing continues until end of file. This will only occur, however, in the last pair when an odd-numbered amount of record pairs is entered. The record option numbers, regardless of how many are entered, are always interpreted in pairs. An attempt to print from record 25 to endfile followed by printing from record 30 to endfile would, if given as "25 30", result instead in the printing of the thirty records beginning at record 25. Start-number, quantity values may be from one to seven decimal digits. A value of zero is invalid. Record number pairs do not have to be entered in pairs of increasing value. The number of record number pairs is limited to 100.
PAGE	Specifies the page of the document at which printing is to begin. The PB utility begins from the start of the file, counting "Top-of-Forms" instructions: page numbers within the file text mean nothing. When it reaches the number of pages specified, (up to seven decimal digits may be used), it begins printing. The value of the page option may be from one to seven decimal digits. A value of zero is invalid.
AUTO	Results in the printing of all available PB files. The utility then enters an "Idle" state until one or more PB files become available for printing. NOTE: The AUTO option cannot be used on systems with no real-time clock.
ON	Specifies the device on which the printing is to be done. "AP" means "Any Printer". It instructs the utility to use the first printer device of any type to become available. "LP" indicates that the line printer is to be used; "SP" indicates the serial printer. If no ON option is entered, default is to the line printer.
LIST	Causes the printing of a list of all printer backup files of the form "PBnnnnn". The following information is printed for each file: 1. "Packid" (followed by that file's <packid>). 2. "External backup filename PBnnnnn" 3. "Internal backup filename PBnnnnn" 4. "Print file name" (followed by the name used for the file by the program that generated it, for example "LISTPRT"). 5. "Program Name" (followed by the name of the program that generated the file, for example, "LIST"). 6. "Creation time HH:MM" (the time, in hours and minutes, at which the file was first opened; the range is 00:00 through 23:59). 7. "Creation date MM/DD/YY" (the date on which the file was first opened). 8. "Device kind" (followed by "LP" (line printer), "AP" (any printer) or "SP" (serial printer)). 9. "Block size" (followed by the size of the block in bytes as specified in the file's header record).

10. "First record" (followed by the number of the record at which printing begins).
11. "FPB flags" (followed by verbal explanations of the flags, as follows:)
    a. "Forms" or "No forms"
    b. "Labelled" or "Unlabelled"
    c. "Translation" or "No translation"
12. "Last record" (followed by the number of the last record printed).

Examples:

PB PB02401

One (1) copy of file "PB02401", which resides on the disk identified by the system status communicate, will be printed. After printing, the file "PB02401" will be removed from the disk.

PB MYDISK/2401

One (1) copy of file "PB02401" residing on the disk named "MYDISK" will be printed. (Note that leading zeroes are not required.) After printing, the file "PB02401" will be removed from "MYDISK".

PB MYDISK/2411 COPIES 22 SAVE

Twenty-two (22) copies of file "PB02411" residing on the disk named "MYDISK" will be printed. After printing, the file will remain on the disk.

PB 01238 RECORD 25 50 300 100 SAVE COPIES 5

Five (5) copies of file "PB01238", residing on the disk identified by the system status communicate, will be printed. In each copy, the only records printed will be 50 records starting at record 25, followed by 100 records starting at record 300. After printing, the file will remain on disk.

PB MYDISK/PB412 ON LP COPIES 3

Three (3) copies of file "PB00412" residing on the disk named "MYDISK" will be printed on a line printer. After printing, the file will be removed from disk.

PB MYDISK/=

All PB files residing on the disk "MYDISK" and belonging to the family "PBnnnnn" will be printed. The PB utility will then go to EOJ.

PB MYDISK/PB00719 RECORD 28 200 SAVE COPIES 10 ON SP

Ten (10) copies of file "PB00719" residing on disk "MYDISK" will be printed on the serial printer. Each copy will contain only the 200 records which begin at the twenty-eighth record of the file. After printing, the file will remain on disk.

PB 02413 RECORD 170

One (1) copy of file "PB02413", residing on the disk identified by the system status communicate, will be printed. This one copy will contain the records beginning at record 170 and continuing to the end of the file. After printing, the file will be removed from disk.

PB 2413 PAGE 15 SAVE ON LP COPIES 30

Thirty (30) copies of file "PB02413", residing on the disk identified by the system status communicate, will be printed on the line printer. Each copy will contain all records found after the fifteenth "Top-of-forms" in the file. After printing, the file will remain on disk.

PB = ON SP

All printer backup files which belong to the "PBnnnnn" family and reside on the disk identified by the system status communicate will be printed on the serial printer. After printing, the files will be removed from disk.

PB AUTO

Print all available "PBnnnnn" files residing on the disk identified by the system status communicate. Then, enter an "Idle" state, printing each PB file that becomes available. Do not automatically go to EOJ.

PB MYDISK/LIST

Produce a listing which identifies all "PBnnnnn" files residing on the disk "MYDISK". List the basic characteristics of each backup file.

PB Messages

The banner is followed by an automatic page throw. Printing of the data contained in the backup file then begins, controlled by the forms-control data accompanying each record.

Files Requiring Special Forms

For files which have the "Special Forms" bit set, the banner is not printed.

The PB utility displays the following message to the user:

SPECIAL FORMS REQUIRED FOR <MFID>/<FID>
CREATED AS PBNNNNN <PRINT FILE NAME> FID BY <PROGRAM FID>

The user must perform an "AX" to inform the utility whether to continue execution. The user enters:

AX <MIX NUMBER> Y
or
AX <MIX NUMBER> N

If the user enters "AX <MIX> Y", printing is performed. If the user enters "AX <MIX> N", that particular PB file is not printed. Entering anything else in the "AX" message causes the PB utility to send an error message to the user:

AX INVALID REQUIRES Y OR N

The user then resubmits the correct response.

After printing a file requiring special forms, the utility displays the message:

REMOVE FORMS

The user enters another "AX" to inform the system to continue:

AX <MIX NUMBER> OK

Entering anything else in the "AX" message causes the PB utility to respond:

AX INVALID REQUIRES OK

The user then resubmits the correct response.

NOTE

If a line printer is already opened, then the utility closes it with lock and re-opens the printer with the forms bit set. At this time, the MCP displays a message on the screen requesting that the user "AD" the device. When a special-forms job concludes, the printer is closed with lock.

Output Messages:

ERROR	POSSIBLE CAUSES	SUGGESTED ACTION
AX INVALID REQUIRES Y OR N	A "Y" or "N" was not entered in response to AX SCL.	Resubmit correct response.
AX INVALID REQUIRES OK	An "OK" was not entered in response to AX SCL.	Resubmit correct response.
I/O ERROR	While reading or writing the named file, an error condition was detected.	Request technical assistance.
INVALID FILE-NAME	This message appears if a "Star File" has been used for the initiating message. One of the following conditions may exist: 1. The packid portion of a filename is greater than 7 characters in length. 2. The fileid portion of a filename is greater than 12 characters or is empty. 3. The filename contains invalid characters.	Check "Star File" input and re-initiate PB.
INVALID PACKID	1. The packid specification is empty. (ie blank characters immediately preceding a slash). 2. The packid contains invalid characters.	Check input and re-enter.
PB DESIGNATE DISK NOT AVAILABLE	A packid was not specified by the user and the system status communicate returned binary zeroes in the printer backup designate field.	Check input and re-enter if necessary.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
TOO MANY DIGITS	1. The number of digits in a number following the page option is greater than 7. 2. The number of digits in a number in a record pair following the record option is greater than 7.	Check input and re-enter.
INVALID NUMBER	While syntaxing a numeric field, a non-numeric character which is not a blank or a binary zero has been detected.	Check input and re-enter.
INVALID START	An error condition is detected when executing a conditional start.	Check record parameters in initiating message; re-enter.
EOF BEFORE PAGES SKIPPED	When attempting to position to the page option. An End Of File condition is detected before the specified page has been reached.	Determine actual filesize and re-enter input.
INVALID OPEN	The fetch value returned on a file open indicates the open was not successful and the file was not in use.	None.
FILE IN USE	The fetch value returned on a file open indicates the open was not successful and the file was in use.	Re-enter input when PB is no longer in use.
INVALID FILE TYPE	The file type of the printer backup file opened was not "@A0@".	None. The file is not a printer backup file.
READ ERROR	While attempting to read the header record, an error condition was detected.	Request technical assistance.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
INVALID ID RECORD	The first 36 characters of the header record were not ASCII.	None. The file is not a printer backup file.
INVALID BLOCK SIZE	The block size from the header record was greater than 132 or equal to zero.	None. The file is not a printer backup file.
INVALID PB NUMBER IN ID RECORD	The number of the PB file as specified in the header record is greater than 65535.	None. File corrupt.
INVALID COPIES NUMBER	The number of copies specified in the initiating message is zero.	Check input and re-enter.
DUPLICATE OPTION	The option has been specified more than once in the initiating message.	Check input and re-enter.
INVALID PAGE PARAMETER	The page number specified in the initiating message is zero.	Check input and re-enter.
TOO MANY RECORD ENTRIES	More than 100 record pairs have been specified in the record option.	Check input and re-enter.
INVALID LENGTH	The stop entry (number of records to be printed) in a record pair is zero.	Check input and re-enter.
TOO MANY DIGITS IN PB NUMBER	The number of digits in the PB number specified in the initiating message is greater than five.	Check input and re-enter.
PB NUMBER > 65535	The PB number specified in the initiating message is greater than 65535.	Check input and re-enter.
INVALID EQUALS OPTION	The option following "=" is not COPIES or ON.	Check input and re-enter.

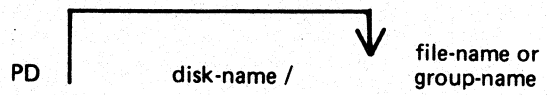
MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
INVALID AUTO OPTION	The option following AUTO is not ON.	Check input and re-enter.
AUTO OPTION REQUIRES CLOCK ON SYSTEM	The auto option is not valid for systems which do not have a real-time clock.	Re-enter input without AUTO option.
INVALID ON OPTION	The parameter following ON is not LP, SP or AP.	Check input and re-enter.
INVALID FILE FORMAT	The least digit of the PB number as recorded in each PB record does not equal the least digit of the PB number as calculated from the header record.	None. File corrupt.

PD (Print Disk Directory)

(a function of SYS-SUPERUTL)

This utility allows the operator to verify the presence on disk of a particular file or a group of files.

Format:



Examples:

To find out if a particular file is on disk:

PD PR210
PD PR2/PR020

To find out if a group of files is on disk:

PD PR2/PR0=
PD PR3=

To find out if several different files or groups are on disk:

PD PR3= , PR2=
PD GL2GL0= , GL2/GL30= , GL250

To inquire about all files on disk:

PD=
PD PR2/=

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ON LINE program-name	File found on disk if single file requested.	None.
group-name ON disk-name CONTAINS:	Group of files found on disk	None.
NOT ON LINE program-name	File not found on disk	Check input (re-input if necessary), Check for correct disk.
NO FILES FOUND IN THE FAMILY "group-name"	Group not found on disk.	Check input (re-input if necessary), Check for correct disk.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
END PD	Successful End of Job.	None.
"file-name" REQUIRES OVERFLOW DISK "disk name"	Remainder of specified file resides on another disk. PD cannot complete until appropriate disk is supplied.	Supply appropriate disk.

Note: See "Common Utility Output Messages" for additional aid.

PL (Print Log Files)

PL lists the contents of log files present during any particular session. Attempts to list other filetypes will fail, since a check is made on the file entries themselves, and these must be of a compatible format. Log files created on previous release levels are incompatible.

The utility incorporates optional facilities to analyze B 900 and B 1800 error entries, analyze statistics for 211 and BSMII disks, and backup files.

Format:

```
PL  <filename>  ↓  <option list>
```

```
where <option list> =  SYSTEM or S
                       ERROR or E
                       STATS
                       DISPLAY
                       BACKUP
                       FROM <time spec>
                       TO <time spec>
                       ENTRY <number>
                       MIX <number>
                       INPUT
                       OUTPUT
                       <empty>
where <time spec> =    <date>
                       or <time>
                       or <date><time>
        and <date> =   MM/DD/YY
            <time> =   HH:MM:SS
```

The option "SYSTEM" (or "S") is specified to list only system messages from the log-file.

The option "ERROR" (or "E") is specified to list only error messages.

The option "FROM <time spec>" is used to list the logged message from the specified date and time. If time is not specified, then 00/00/00 is assumed.

The option "TO <time spec>" is used to list the messages up to that date and time. If time is not specified, then the last date and time in the log are assumed.

If the option "ENTRY" is used, the utility will print starting from the record number specified by the operator.

The "MIX number" option is used to print all messages related to specified mix (number(s)).

The "INPUT" and "OUTPUT" options allow the operator to print either input or output messages.

Any combination of SYSTEM, ERROR and STATS is permitted, but, if DISPLAY or BACKUP is requested, only ERROR and STATS are permitted.

The default options that are set are:

SYSTEM, ERROR and STATS messages;
INPUT and OUTPUT messages;
FROM 00/00/00 00:00:00;
TO <last date and time>;
ENTRY 1;
Output direct to printer.

All entries are displayed irrespective of their mix numbers. Any of these defaults can be reset at run time. If no real-time clock was available when the file was created, then no check will be made on the "time" portion of the operator input, and "N/A" will be printed under the "TIME" heading on the report. The default for DISPLAY/BACKUP is both ERROR and STATS.

STATS request will only output entries logged as statistics entries for BSMII and 211 disks, and only if those disks are used with the Standard Disk Interface (SDI) Common Controller.

If DISPLAY is requested, then error and statistics entries will be displayed on a screen with dimensions not less than 80 characters wide and 10 lines deep.

BACKUP will cause all output to go to a disk file of type "data". The format of this file will be exactly the same as if the output had gone directly to a printer. The file will have 120 byte records, and three records per block. Page throws will be replaced by four space-filled records. The default size of the file will be 4096 records, but this can be altered using MODIFY. The internal file name and resultant output FID will be MLOGBACKUP; the destination disk will be the system disk unless MODIFYed; CLOSEMODE will be LOCK, but may also be MODIFYed.

Entries with multiple records will only have the record number and record contents displayed; all other columns will be blank since the contents of these records will all be of the same type, and created at the same time.

Only entries which conform with either the defaults, or operator input specifications will be displayed, all others will be ignored.

The range of values for ENTRY and MIX numbers are 1-65535 and 1-254 respectively. Checks at run-time are made on the values entered and messages issued if they are in error.

Examples:

To print the contents of the log-file called SYS-LOG-HOLD:

PL SYS-LOG-HOLD

To print the error messages logged in the log-file called SYS-LOG-01:

PL SYS-LOG-01 ERROR

To print entries in SYS-LOG-HOLD file from record 100, related to mix number 12 from January 1, 1979 until latest date:

PL SYS-LOG-HOLD ENTRY 100 MIX 12 FROM 01/01/79

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO SPECIFICATION GIVEN	The initiating message contained no parameters.	Re-enter correctly.
NO FILE NAME FOUND IN PARAMETERS	The first parameter found in the initiating message was not the log file name.	Check input and re-enter correctly.
ILLEGAL OPTION <parameter>	The parameter displayed is not a valid one.	Check input and re-enter correctly.
ILLEGAL VALUE <integer>	The integer entry after either ENTRY or MIX is not within the allowable range of values.	Check input and re-enter correctly.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
NO TIME SPECIFICATION AFTER <time option>	Input after either "TO" or "FROM" is incorrect or inappropriate.	Check input and re-enter correctly.
NO <param> NUMBER GIVEN	A number was expected after "MIX" but was not found.	Check input and re-enter correctly.
<option> OPTION ALREADY SET	One of the parameters set contradicts another.	Check input and re-enter correctly.
ILLEGAL <date/time> SPECIFICATION <time-spec>	The <time-spec> is not of the correct format. <time> comes before <date> if both are specified.	Check input and re-enter correctly.
FILE <file-name> NOT FOUND or, FILE <file-name> IN USE	The desired file is not on the specified disk, or is not available.	Make file available and re-enter.
<file-name> IS NOT A LOG FILE	The file being processed is not of log-file type or format.	
ILLEGAL PARAMETER LIST <param>	A parameter error has been found. The parameter in error is displayed.	Check input and re-enter correctly.
ILLEGAL FILE NAME <file-name>	The file-name does not conform to CMS format.	Check input and re-enter correctly.
INVALID CHARACTER IN IDENTIFIER <identifier>		Check input and re-enter correctly.
DISK <pack-id> NOT AVAILABLE	The given <pack-id> is not on-line.	Put the desired disk on line, and re-enter correctly.
**** UTILITY LIMIT REACHED ****	Invalid information contained in the log file.	None.
UNRECOGNISED DEVICE IN ERROR ENTRY	No device can be found in the Hardware Configuration Table which corresponds to the device type in the error descriptor, and hence no error counts can be updated from this information.	None.

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
NO ENTRIES FOUND WITHIN PARAMETERS	End of File has been reached without any entries being found and printed.	None.
DISK <disk-id> LOCKED	The file requested is on a disk for which a utility is using SYSMEM LOCK.	Re-enter when the utility has finished.
SCREEN SIZE TOO SMALL TO RUN UTILITY	When using the DISPLAY option the utility found that the SPO screen size was less that 80 characters wide and 10 characters deep.	Re-input without the DISPLAY option.
ENTER <END> TO FINISH OR <NEXT> TO CONTINUE	After displaying a screenful, the utility will request whether or not more information is required.	Enter NEXT or END
INVALID REQUEST <param>	A response other than the expected END or NEXT was entered. The utility will again request whether or not more information is required.	Check input and re-enter correctly.
OUTPUT FILE <file-name> TOO SMALL	When using the BACKUP option, the utility reached the end of the output file.	None - the file will be closed and the utility will go to EOJ.
<decimal no> RECORDS WRITTEN TO BACKUP FILE <file-name>	Writing to the backup file has been completed and the file has been closed.	None.

As well as these messages, a check on parity errors is made during the processing of the file. If a read/write parity error is incurred, then the message:

I/O ERROR DETECTED IN READ FROM <file-name>
or
I/O ERROR DETECTED IN WRITE TO FILE <file-name>

is displayed, followed by the record number at which the error occurred. The program will either attempt to continue processing or will go to End of Job depending on where the error occurred. If it does continue, then "CONTINUING PROCESS" will be displayed.

However, if the ENTRY option has been used, and the error occurs while processing this command, then the results might not be those requested.

On reaching the end of the processing, the utility goes to End of Job and closes all files, leaving the input file in exactly the same state as it was at the start.

PPID (Pseudo Pack Identifier Display)

The Pseudo Pack Identifier Display utility may be used to list the Pseudo Pack Identifier Table (PPIT). This utility does not provide the *<file-name> option in the initiating message. The syntax for this utility is:

```
PPID ──┬──────────┬──
       └─ PRINT ──┘
```

If PRINT is specified, the output will be listed on a printer, otherwise the output will be displayed on the Operator Display Terminal (ODT).

One of the following messages will be output for each used entry in the PPIT (Logical Unit No. = XX):

<PACKID> - PHYSICAL UNIT XX<OL> (Physical unit on line)
<PACKID> - PHYSICAL UNIT XX<OFF> (Physical unit not on line)
<PACKID> - PSEUDO ON <UNIT PACKID> (Restricted pseudo pack)
<PACKID> - PSEUDO UNRESTRICTED (Unrestricted pseudo pack)

Physical unit entries are displayed in order of logical unit number, each being immediately followed by entries of pseudo packs restricted to that particular unit. Lastly, unrestricted pseudo pack entries are displayed.

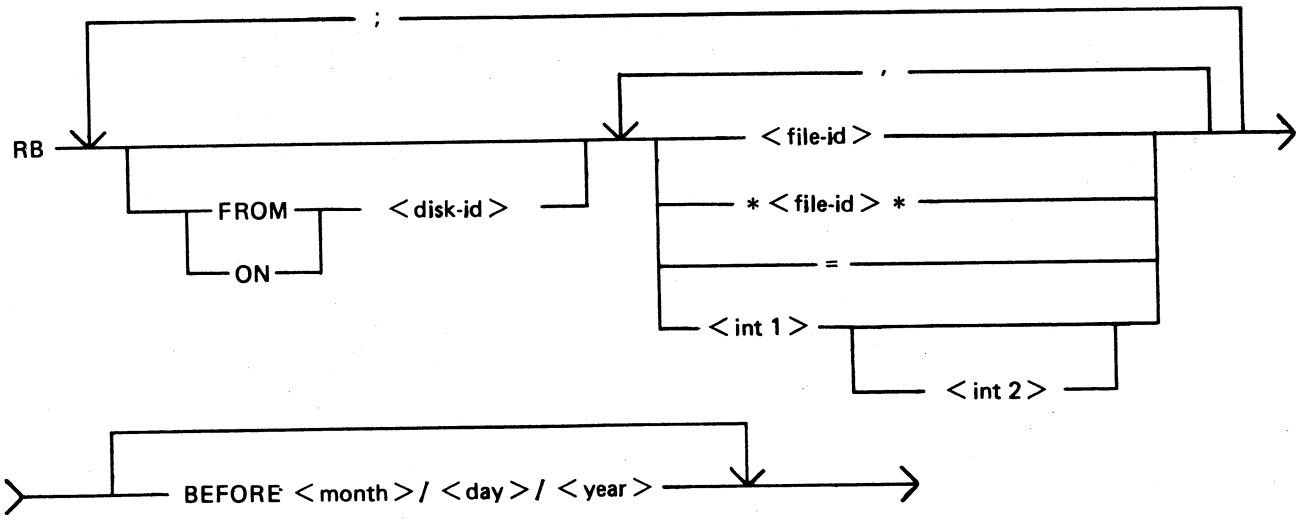
Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO PPIT ON SYSTEM	No pseudo pack identifier table was found on the system disk.	None.
READ ERROR ON PPIT OF "DISKID"	An error was encountered while reading the PPIT.	None. Irrecoverable error.

RB (Remove Printer Backup Files)

This utility enables the user to remove printer backup files from one or more disks. It can remove backup files singly, by family, or by age.

Entering RB or RB HELP displays the syntax diagram for RB, which is as follows:



where:

<disk-id>	specifies the disk on which the utility seeks backup files; the default is the backup disk designate.
<file-id>	designates the backup file or the family of backup files to be removed. Either the external or the internal file name may be specified.
<file-id>	is used when <file-id> alone could cause confusion (for example, RB *5* removes the backup file named 5 instead of the one named PB00005).
=	specifies that all the backup files PBxxxxx are to be removed, where 1 ≤ xxxxx ≤ 65535.
<int 1>	specifies that the backup file PB<int 1> is to be removed.
<int 1>-<int 2>	this specification causes all backup files from PB<int 1> to PB<int 2> to be removed.
<month>	is a 1 or 2-digit integer.
<day>	is a 1 or 2-digit integer.
<year>	is a 2-digit integer.
BEFORE <date>	option causes all backup files created before the specified date to be removed.

Examples:

To remove the file PB00038 from the system disk:

```
RB 38
```

To remove the file PB00017, and all backup files in the family PR=, from the disk USDSK:

```
RB FROM USDSK 17, PR=
```

To remove PB00032 and PB00034 from the system disk, all the backup files named PBxxxxx from the disk TASK, and all backup files in the range PB00011 to PB00019 from the disk ARDSK, each one of these files subject to the date given:

```
RB 32,34 ; FROM TASK = ; FROM ARDSK 11-19 BEFORE 3/3/81
```

Messages

If syntax errors occur, the utility displays messages that are self-explanatory, and goes to End of Job.

If a disk cannot be opened, or no files are found in a family, RB continues execution after displaying the appropriate messages.

The user is also informed when a file is removed, or when it is not removed (because it is still in use, or because it was created on or after the "BEFORE" date).

After completion, RB issues the message:

** RB COMPLETE **

RL (Relabel Disk)

This utility is used to re-label physical CMS disks. It may not be used to re-label pseudo disks, nor may a system disk in use be re-labelled, as the utility requires all files on disk to be closed.

The utility has the form:

RL ——— < old disk-id > ——— AS ——— < new disk-id > ———→

A disk-id of "0000000" may be specified as the old disk-id, but is invalid if specified as a new disk-id.

The utility opens an unlabelled disk and the operator is prompted to AD the required disk drive which has been previously RD'ed.

If the disk label is in CMS format and the disk-id matches the old disk-id specified in the initiating message, the disk will be re-labelled with the specified new disk-id.

If a Pseudo Pack Identification Table (PPIT) resides on an AD'ed fixed disk, the disk will not be re-labelled: alteration of the disk-id of a Physical Unit of a Fixed Disk assemblage would cause PPIT conflicts when the disk was readied and AVRed by the MCP.

Output Format:

The following messages may be displayed by the RL utility:

OPEN DISK UNLABELLED NOT IMPLEMENTED

The utility has been unable to invoke an unlabelled disk open as the facility is not implemented in the MCP.

CANNOT READ SECTOR 0 OF Dxy

The utility has opened the disk successfully but cannot read sector 0. This occurs if a brand new disk (other than an SDI disk) which has never been initialized by Stand-Alone methods is AD'ed to the utility. It may also occur on badly corrupted disks.

DISK LABEL OF Dxy NOT IN CMS FORMAT

The utility has found that the disk label is not in CMS format.

LABEL OF Dxy IS <disk-id> - NOT <old disk-id>

The specified old disk-id does not match the actual disk-id in the disk label.

Dxy IS FIXED DISK DFy WITH PPIT

The fixed disk AD'ed to the utility has a PPIT.

NOTE

If any of the above messages are displayed, the disk-id in the label remains unaltered and the utility goes to End of Job.

Dxy DISK <old disk-id> RELABELLED AS <new disk-id>

The disk has been relabelled successfully. The utility will go to End of Job.

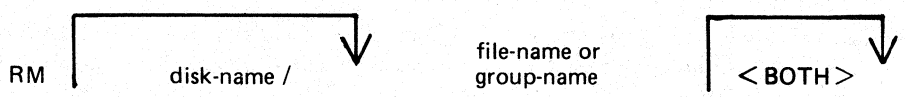
RM (Remove Files from Disk)

(a function of SYS-SUPERUTL)

This utility allows the removal of individual files and groups of files from disk. The disk areas associated with those files are returned to the available table.

If the utility detects that a keyfile is to be removed and the <BOTH> option has been specified, then it will remove both the keyfile and the associated data file if both are on disk. If <BOTH> is not specified then only the keyfile will be removed.

Format:



Examples:

To remove a single file:

RM AR300
RM PR1/PR300

To remove a group of files:

RM AR=
RM INDISK2/IN3=

To remove several different groups and/or individual files:

RM IC230, IN076, INDISK1/IN2=

To remove a keyfile and associated data file:

RM PR200K <BOTH>

A request for the removal of a system file will cause the utility to output the following:

file-name IS A SYSTEM FILE
AX "mix number"/RM ACPT

Then, to remove a system file:

AX mix-number/RM file-name OK (mix-number is the mix number of RM).

If the operator types any other sequence the system file will not be removed.

Example:

```
RM NDL=
NDL.INTERP IS A SYSTEM FILE
12/RM ACPT
AX 12/RM NDL.INTERP OK
```

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
"filename" REMOVED	File was removed.	None.
"filename" NOT REMOVED NOT FOUND	Specified file was not removed.	Check input (re-input if necessary); Check for correct disk.
"filename" NOT REMOVED - IN USE	Specified file was not removed because it is currently being used by the system.	Wait until file is no longer in use, then re-input.
"filename" NOT REMOVED - SYSTEM FILE	Specified file was not removed because it is a "system file" (for example, MCP, an interpreter, etc).	If file is to be removed, type "AX mix-number/RM file-name OK". If file is not to be removed, type AX mix/RM NO
INDEXED PAIR "file name" "file-name" REMOVED	Keyfile and associated data file were removed.	None.
INDEXED PAIR "filename" "filename" NOT REMOVED	Keyfile and associated data file were not removed. This message is followed by the reason.	Check input (re-input if necessary) or Check for correct disk.
INDEXED PAIR "filename" "filename" NOT REMOVED - IN USE	Specified keyfile and data file were not removed because at least one is currently being used by the system.	Wait until files are no longer in use, then re-input.

SCR (System Confidence Routine)

The SCR utility is used to establish a level of confidence in the overall performance of the software/hardware interface of a CMS system. It is not intended to perform the function of an MTR program in rigorously testing isolated hardware components, nor is it intended as a proof of correctness for system software.

The SCR operates under MCP control, and relies upon the MCP for basic error detection. In addition, SCR makes certain comparisons between actual results and expected results, and performs checks on output results.

Operator input required is minimal, and is restricted to initiation of the utility and replies to prompts displayed immediately thereafter. Information concerning errors detected and program performance are displayed at the SPO throughout execution of the SCR. The SCR may be DS'ed by the user at any time.

The integrity of disk files is maintained: all disk files created by SCR are CLOSEd with RELEASE. Thus, when the utility goes to EOJ, those files do not remain on disk.

The SCR utility is written in MPL. One of its major subsections (CPU.IO) ZIPs to two independent programs, one of which is written in MPL and the other in COBOL.

The utility comprises four major subsections. The user can specify which of these subsections are to be performed for any given run of SCR. To assist him in making his selection, a HELP display is provided. The four subsections are:

1. DCR — This subsection ZIPs the independently executable Disk Confidence Routine, which tests CMS disk input/output operations.
2. CPU.IO — This subsection tests the interface between the MCP, IO, the MPL and COBOL Interpreters by ZIPping to an MPL program which outputs the result of calculations to disk, and by ZIPping to a COBOL program which prints the edited results of its calculations.
3. MT.IO — This subsection writes files of binary information to user-selected magnetic devices, reads these files back, and compares the output and input records.
4. OTHER.IO — This subsection outputs files of information to user-selected non-magnetic devices, reads the files back and checks them when possible. Currently, this subsection only applies to printer, and therefore relies on the user to perform a visual check of the information output.

Any or all of these subsections may be selected by the user for a given run of the utility. The order of performance is determined by the order of selection; however, if selected, DCR will always be performed first and it will be performed once only. The number of iterations for each of the subsections other than DCR may be specified by the user; all iterations of a given subsection are performed before execution of another subsection is begun.

It is recommended that care is taken when specifying the size of file for the DCR WRITE-INDEX test. Since the records are written and read randomly, access time becomes proportionately greater as the filesize increases.

Operating Instructions

Enter: SCR

Prompt 1 is displayed:

"ENTER TEST NUMBERS OR ENTER HELP"
"ENTER SPACE TO TERMINATE"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	SCR TERMINATED BY USER	Utility has been terminated.	
AX <mix-no> HELP	ENTER ON ONE LINE NUMBERS FOR TESTS WANTED. 1 = DCR 2 = CPU.IO 3 = MT.IO 4 = OTHER.IO ALL=ALL TESTS WANTED SPACE=TERMINATE SCR <PROMPT 1>	Help function has been requested.	Select tests required and input test numbers, or input space to terminate the utility.
AX <mix-no> <test list>	INVALID INPUT CHARACTER <PROMPT 1>	Test list input did not consist of valid test numbers or "ALL".	Re-input test list, or terminate the utility.
	<PROMPT 2>	Test 1 (DCR) or "ALL" has been selected.	Refer to possible replies to Prompt 2.
	<PROMPT 3>	Test 2 (CPU.IO) or "ALL" has been selected.	Refer to possible replies to Prompt 3.
	<PROMPT 4>	Test 3 (MT.IO) or "ALL" has been selected	Refer to possible replies to Prompt 4.
	<PROMPT 5>	Test4 (OTHER.IO) or "ALL" has been selected.	Refer to possible replies to Prompt 5.

If the reply contains the string "ALL", any valid digits contained in the reply are ignored.

NOTE

If the reply contains the digit 1, or if "ALL" tests are requested, SCR will display prompts requesting parameters for DCR after parameters for other specified subsections have been requested.

Prompt 2:

For a description of prompts and replies establishing DCR parameters, see the description of the utility DCR, contained earlier in this section.

Prompts requesting parameters to DCR are issued as a result of execution of the DCR utility. Hence, when the DCR utility is executed under control of SCR, these prompts are issued following prompts requesting parameters to any other SCR subsection (which are issued prior to execution of the subsections themselves).

NOTE

Replies to prompts requesting DCR parameters must be preceded by "AX <DCR mix-no>", and NOT by "AX <SCR mix-no>".

Prompt 3:

"ENTER NUMBER OF ITERATIONS FOR CPU.IO – DEFAULT = 1."

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>		The CPU.IO test will be performed once.	
AX <mix-no> <decimal no>		CPU.IO will be performed the number of times specified, up to a maximum of 99 times. Only the first two digits are read.	
	INVALID INPUT CHARACTER <PROMPT 3>	A non-decimal character was input.	Re-input number of iterations.

Prompt 4:

"SPECIFY MT.IO PARAMETERS –
ENTER SPACE FOR DEFAULT VALUE"
"ENTER NUMBER OF MT DEVICES – DEFAULT = 1"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	<PROMPT 4.1>	SCR will write to and read from one magnetic tape reel.	Refer to possible replies to Prompt 4.1.
AX <mix-no> <decimal no>	<PROMPT 4.1>	SCR will write to and read from the number of magnetic tape reels specified-up to a maximum of five reels. Only the first character input is read.	Refer to possible replies to Prompt 4.1
	INVALID INPUT CHARACTER <PROMPT 4>	An invalid non-decimal has been input	Re-input no. of reels required.

Prompt 4.1:

"ENTER NUMBER OF CS DEVICES – DEFAULT = 0"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	<PROMPT 4.2>	SCR has used the default value of zero, and will not write to or read from cassette.	Refer to possible replies to Prompt 4.2
AX <mix-no> <decimal no>	<PROMPT 4.2>	SCR will attempt to write to and read from the specified number of cassettes, up to a maximum of five cassettes.	Refer to possible replies to Prompt 4.2
	ERROR: MORE THAN 5 DEVICES REQUESTED. <PROMPT 4>	The total number of magnetic tape reels and cassettes requested, specifically or by default, exceeds five.	Re-input the parameters for MT and CS devices, taking care that the total does not exceed five.

Prompt 4.2:

"ENTER NUMBER OF RECORDS IN FILE – DEFAULT = 256"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	<PROMPT 4.3>	Each file written to and read from magnetic tape reels and/or cassettes will contain 256 records.	Refer to possible replies to Prompt 4.3
AX <mix-no> <decimal no>	<PROMPT 4.3>	The size of all magnetic media files will be the (3-digit) decimal number specified.	Refer to possible replies to Prompt 4.3

Prompt 4.3:

"ENTER NUMBER OF ITERATIONS – DEFAULT = 1"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	READY SCRATCH TAPES ON <m> MAGNETIC TAPE DRIVES, <n> CASSETTE DRIVES	The MT.IO subsection will be performed once only.	Make ready the specified number of tapes on their drives
AX <mix-no> <decimal no>	READY SCRATCH TAPES ON <m> MAGNETIC TAPE DRIVES, <n> CASSETTE DRIVES	The MT.IO subsection will be performed the number of times specified.	Make ready the specified number of tapes on their drives

Prompt 5:

"ENTER NUMBER OF OTHER.IO ITERATIONS – DEFAULT = 1"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> <empty>	<PROMPT 5.1>	The OTHER.IO subsection will be performed once only.	Refer to possible replies to Prompt 5.1
AX <mix-no> <decimal no>	INVALID CHARACTER <PROMPT 5>	The input string contains a non-decimal character.	Re-input the required no. of iterations correctly.
	<PROMPT 5.1>	The OTHER.IO subsection will be performed the number of times specified by the first two digits of the input string (ie up to 99 times).	Refer to possible replies to Prompt 5.1

Prompt 5.1:

"IF PRINTER TO BE TESTED, ENTER Y"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> Y	<PROMPT 5.2>	SCR will write files (the number of files being specified by the reply to Prompt 5) of a pre-determined content to the first printer available at execution time.	Refer to possible replies to Prompt 5.2
AX <mix-no> <not "Y">	<PROMPT 5.2>	The printer test will not be performed.	Refer to possible replies to Prompt 5.2

Prompt 5.2:

"IF CARD PUNCH AND READER TO BE TESTED, ENTER Y"

POSSIBLE REPLY	POSSIBLE RESULTING OUTPUT	POSSIBLE CAUSE	SUGGESTED ACTION
AX <mix-no> Y	CARD PERIPHERALS NOT IMPLEMENTED	No card tests will be performed.	
AX <mix-no> <not "Y">		No card tests will be performed.	

Execution Details

DCR

If the DCR subsection has been requested, SCR executes the DCR utility via a ZIP (PAUSE, DISPLAY) communicate. If the ZIP fails, the fetch value returned by the ZIP communicate is examined. If the failure was because of a full mix, the ZIP is repeated until either it is successful or a different fetch value is returned. If the fetch value indicates failure for any other reason, one of the following messages is displayed:

1. "ZIP FAILURE DUE TO PROGRAM FILE NOT FOUND FOR DCR"
2. "ZIP FAILURE DUE TO INTERPRETER NOT FOUND FOR DCR"
3. "ZIP FAILURE DUE TO NO MEMORY FOR DCR"
4. "ZIP FAILURE DUE TO NO USER DISK FOR DCR"
5. "ZIP FAILURE DUE TO USER COUNT ERROR DCR"
6. "ZIP FAILURE DUE TO DUPLICATE PACK DCR"
7. "ZIP FAILURE DUE TO INVALID LOAD REQUEST DCR"
8. "ZIP FAILURE DUE TO MCS ALREADY PRESENT DCR"
9. "ZIP FAILURE DUE TO DISK ERROR DCR"
10. "ZIP FAILURE DUE TO CODE FILE ERROR DCR"
11. "ZIP FAILURE DUE TO ILLEGAL DATA REQUEST DCR"
12. "ZIPPED PROGRAM DS'ED DCR"
13. "ZIPPED PROGRAM DP'ED DCR"
14. "ZIP FAILURE, REASON UNKNOWN DCR"

Regardless of the reason for failure, SCR proceeds to execute the next requested subsection.

For details of DCR, refer to the description of DCR contained earlier in this section.

CPU.IO

For each iteration of the CPU.IO subsection, the programs SCR.MPL and SCR.COBOL are each executed once, each via a ZIP (DISPLAY). If a ZIP fails, the fetch value is examined. If the failure was because of a full mix, the ZIP is repeated until either it is successful or another fetch value is returned. For all other fetch values after a ZIP failure, the appropriate message from those listed for DCR is displayed, the program name "DCR" being replaced by "SCR.MPL" or "SCR.COBOL". Messages 12 and 13 do not apply to SCR.MPL or SCR.COBOL, as the ZIP is without PAUSE.

As SCR.MPL and SCR.COBOL are executed via ZIPs without PAUSE, multiple copies of each (produced by successive iterations of CPU.IO) may run concurrently. However, once control has been transferred to a copy of either SCR.MPL or SCR.COBOL, its status is not checked by SCR.

The ZIP to execute SCR.MPL includes a 2-character initiating message which is the ASCII value of the current CPU.IO iteration number.

SCR.MPL

The 2-character initiating message is used as the last two characters in the 8-character FID, "SCR.M.cc", of a random access disk file on the System Disk. This file is opened conditionally. If the open is not successful, the following messages are displayed:

"FETCH VALUE = <hex no>"
"SYSTEM DISK NOT AVAILABLE"
"SCR.MPL COPY <2-digit no.> TERMINATED"

(where the <2-digit no.> is the 2-character initiating message) and goes to End of Job.

If the file is opened successfully, the following comparative expression is computed iteratively:

$$(((I+1)*5)*((I+1)*5))/((I+1)*5) = (I+1)*5$$

where I is incremented by 1 from 0 to 50 and is equal to the ordinal number of the computation iteration.

At each computation, if the expression is evaluated as false, the following message is displayed:

"P1 COMPUTATION ERROR, I = <decimal no>"

If the expression is evaluated as true, a fixed value for I is computed and output as the first 2 bytes of a 180-byte record. If the WRITE operation results in a conditional fail and the last two bytes of the fetch value are hexadecimal "0000", the WRITE operation is repeated. If the last two bytes of the fetch value on a conditional fail are not hexadecimal "0000", the following messages are displayed:

"SCR.MPL COPY <decimal no>"

where <decimal no> is the 2-character SCR.MPL initiating message, and

"FETCH VALUE = <hex no.>"

If the WRITE operation causes an error, the MCP error message is the only message displayed.

Immediately after a record is written, it is read back into memory and the first two bytes compared with the current value of I.

If the READ operation causes a conditional fail and the last two bytes of the fetch value are hexadecimal "0000", the READ operation is repeated. If the last two bytes of the fetch value are not hexadecimal "0000", the following messages are displayed, as for the WRITE operation:

"SCR.MPL COPY <decimal no>"
"FETCH VALUE = <hex no>"

If the READ operation causes an error, the MCP error message is the only message displayed. If the comparison between the input value and I fails, the following message is displayed:

"INPUT/OUTPUT MISMATCH, I = <decimal no>"

After 51 compute-write-read-compare iterations have been performed, SCR.MPL displays the following messages:

"SCR.MPL COPY <decimal no> TERMINATED"
"WRITE CONDITIONAL FAILS = <decimal no>,
WRITE ERRORS = <decimal no>."
"READ CONDITIONAL FAILS = <decimal no>,
READ ERRORS = <decimal no>"

SCR.MPL then closes the disk file with RELEASE, and goes to End of Job.

SCR.COBOL

Using the initial values:

A = 10
B = 1
C = .1
D = .01
E = .2

SCR.COBOL calculates:

X = (A+B) * (C–D)/(E*E)

using the COMPUTE statement, and calculates the same expression in an alternative manner, using the individual arithmetic statements ADD, SUBTRACT, MULTIPLY and DIVIDE, assigning the results to variable X1. The value X2 = –1 * X is also calculated.

The values X, X1 and X2 are moved to fields of a line image, using the editing symbols : % Z B – * CR DB, and the line is printed by the first available printer on a page headed "PROGRAM CMTHS". Sub-headings are printed for each of the fields.

The values X, X1 and X2 are calculated and printed five times, the values of A, B, C, D and E being multiplied by 2 after each calculation.

Each execution of SCR.COBOL produces one printed page, as shown here:

COMP. VALUE	CALC. VALUE	EDIT 1	EDIT 2	EDIT 3	EDIT 4	EDIT 5	EDIT 6	EDIT 7
002475	002475	$ 24.75	$ 24.75	.75	**24.75	$002475	**24.75CR	$002475DB
009900	009900	$ 99.00	$ 99.00	.00	**99.00	$009900	**99.00CR	$009900DB
039600	039600	$ 396.00	$396.00	.00	*396.00	$039600	*396.00CR	$039600DB
158400	158400	$1584.00	$584.00	.00	1584.00	$158400	1584.00CR	$158400DB
633600	633600	$6336.00	$366.00	.00	6336.00	$633600	6366.00CR	$633600DB

After the last line of the page has been printed, SCR.COBOL goes to End of Job.

MT.IO

For each iteration of the MT.IO subsection, a file is opened for output on each of the magnetic devices requested by the user.

To each device on which a file was successfully opened, a file of the size specified by the user (or the default size of 256 records) is written. Each record is 120 bytes long and contains 60 fixed point values, where each fixed point value is the current record number. The current record is written to each of the open files before the record is updated. If the WRITE operation causes a conditional fail and the last two bytes of the fetch value are hexadecimal "0000", the WRITE is repeated. If the WRITE operation causes a conditional fail and the last two bytes of the fetch value are not hexadecimal "0000", or if the WRITE operation causes an error, the following messages are displayed:

"SCR MT.IO; ITERATION <decimal no>:"
"WRITE FAIL ON <device> FILE <file no> RECORD <record no>"
"FETCH VALUE = <hex no>"

and the file is half-closed.

After output of all files has been completed, all open files are half-closed and re-opened for input. If an open fails, the following messages are displayed:

"SCR MT.IO; ITERATION <decimal no>:"
"FILE <file no> DID NOT OPEN FOR INPUT ON <device>"
"FETCH VALUE = <hex no>."

All open files are read, the current record being read from all open files before the record number is updated. dated. If the READ operation causes a conditional fail and the last two bytes of the fetch value are hexadecimal "0000", the READ operation is repeated. If the READ operation causes a conditional fail and the last two bytes of the fetch value are not hexadecimal "0000", or if the READ operation causes an error, the following messages are displayed:

"SCR MT.IO; ITERATION <decimal no>:"
"READ FAIL ON <device> FILE <file no> RECORD <record no>"
"FETCH VALUE = <hex no>"

and the file is closed and purged.

As each record of each file is input, the 60 fixed point values it contains are compared with the current record number. In the event of a mismatch, the following messages are displayed:

"SCR MT.IO; ITERATION <decimal no>:"
"WRITE/READ MISMATCH ON <device> FILE <file no>, RECORD <record no>"

After all files have been input, all open files are closed and purged.

The output-input operations described above are repeated for the number of MT.IO iterations specified by the user.

OTHER.IO

The OTHER.IO subsection of SCR is currently limited to the first printer available at execution, using MPL Line Controls.

Each iteration of the subsection outputs six pages of print, each print page containing 20 pairs of print lines. Pairs of lines are separated by a space line. Each of the first pair of lines contains 120 X's, with spaces replacing X's in the leftmost character positions of line pairs, the number of spaces being incremented by 1 for each pair of lines, as shown in figure 4-6.

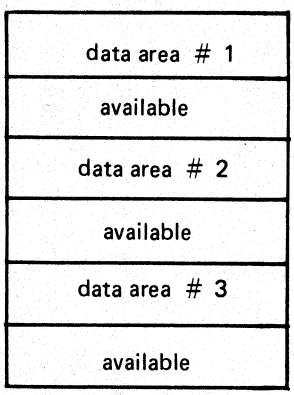
At the end of each iteration, the PRINTER file is closed with RELEASE and re-opened at the start of the next iteration. The number of iterations that are performed are determined by user input.

```
                                                  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
                                  1 1 1 1 1 1 1 1   0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7....5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
-----------------------------------------------------------------------------------
X X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X
X X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X

  X X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X
  X X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X

    X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X
    X X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X

      X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X
      X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X

        X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X
        X X X X X X X X X X X X X X X....X X X X X X X X X X X X X X X X

                                  ..".....
                                  ..".....
                                  ..".....
                                    "
                                    V
                                                        X X X X X X X X
                                                        X X X X X X X X

                                                          X X X X X X X
                                                          X X X X X X X

                                                            X X X X X X
                                                            X X X X X X

                                                              X X X X X
                                                              X X X X X

                                                                X X X X
                                                                X X X X

                                                                  X X X
                                                                  X X X

                                                                    X X
                                                                    X X

                                                                      X
                                                                      X
```

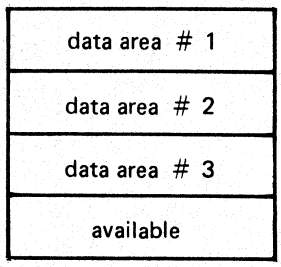
Figure 4-6. OTHER.IO Sample Printer Output

SQ (Squash Disk)

When a disk unit is used extensively with a high degree of file activity involving creation and removal of files then it is possible for the available space on the disk to become so fragmented that it is increasingly difficult to find enough space in one single area to satisfy requests for disk space. This results in a degradation of system throughput with an increasing incidence of "NO USER DISK" failures and extra time needed to search through available areas. This situation is known as "checkerboarding" of the disk. In the extreme case each area of disk in use is separated by an available area, as shown in the diagram below:



The SQ utility is designed to eliminate checkerboarding of disk, either for the whole disk or part of the disk. This process is called "squashing" disk and is accomplished by moving each data area in turn to the first available area at a lower address. If an entire disk is squashed then all available areas are merged into one area at high-address end of the disk, as in the next diagram:



The options available within the SQ utility are:

Squash of a complete disk.

All data areas are moved to successively lower addresses until only one available area is left (as in diagram above).

Partial squash

Only data areas within a default section of the disk are moved to lower addresses within the section.

Fast squash

The aim of a fast squash is to create an available area of disk of a requested size. Only those data areas are moved which will allow an available area of a sufficient size to be created.

Economic squash

In this case, data areas are only moved if the gain in terms of available space justifies the time spent in movement of the data area. As an example, consider the following case:

data area # 1
available # 1
data area # 2
available # 2

where data area #1 is 100 units, data area #2 is 200 units and both available areas are 1 unit each. If available areas are merged the available area gained would be 2 units. However, to acquire these 2 units, the 200 units of data area #2 would have to be moved. Therefore an "economic squash" would not move data area #2. In general terms, an economic squash will ignore small available areas that are interspersed in large data areas. However, in some cases an economic squash will have the same effects as a full squash.

With all options of SQ a further option is available to print a map of the entire disk in disk-address order both before and after squashing action.

Input is as follows:

Format:

```
                 ┌──────────────────────────────────────────────────────────┐
                 │  ┌──────────────────────────────────────────────────────┐
                 │  │ VERIFY                                               │
                 │  │ ALL                                                  │
SQ disk-name     │  │ FAST   number                                        │
                 │  │                                                      │
                 │  │   FROM { BEGIN       } TO { END       }              │
                 │  │        { start-address } { end-address }             │
                 │  └──────────────────────────────────────────────────────┘
                 │                                                          │
                 │                                                  LIST    ↓
```

note: the number is in the range 1 to 65535; the start-address and end-address are 6-digit hexadecimal disk addresses, for example, 000AB3, 01A375.

Examples:

To perform an economic squash of disk PR2:

SQ PR2

To check the integrity of disk PR2:

SQ PR2 VERIFY

To perform a full squash of disk PR2 and list the disk map:

SQ PR2 ALL LIST

To move data areas to provide one available area of 1000 sectors on disk PR2:

SQ PR2 FAST 1000

To perform a partial squash on sectors 0 through 512 of disk PR2:

SQ PR2 FROM BEGIN TO 000200

To perform a partial squash on sectors 512 through 4096 of disk PR2:

SQ PR2 FROM 000200 TO 001000

To perform a partial squash on sectors 4096 to the last addressable sector of disk PR2:

SQ PR2 FROM 001000 TO END

Before performing any function which involves physically moving data areas, the integrity of the disk is checked. Integrity checking involves analyzing disk assignment to verify that the entire area of the disk is described in the file directories and available table, checking the directories themselves and attempting to resolve anomalies (for example, missing areas or overlapping areas). Only after the integrity is verified are areas of disk physically moved.

Certain areas of disk will not be moved in any circumstance. These are areas of disk currently marked as in use, and any system log files. In addition, SQ can only be run in a suitable mix, as defined by the MCP to safeguard the integrity of the disk. No user program can be run with SQ. During execution of SQ the MCP will reject any attempt to execute any utility or user program.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
LARGEST AVAILABLE AREA IS number SECTORS. TOTAL AVAILABLE IS number SECTORS IN number AREA(s). ***SQ COMPLETED***	Given on successful completion of SQ.	None.
NON-FILE DIRECTORY FULL - "PARTIAL" SQUASH REQUIRED	Display during verification phase if there are no free entries when attempting to add entries to the available table if missing areas are detected.	Run SQ with VERIFY LIST options. Examine the disk map to discover a section of the disk that can be squashed to create free entries in the available table Example: The pattern File Area Available File Area Available if squashed will create one free entry in the available table. In general the # of free entries to be created equals the # of missing areas that are not contiguous with any available area.
SQ INVALID - NO INITIATING MESSAGE	Self-explanatory	Re-input.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SQ INVALID - INVALID PACK-ID	Pack-id specified is longer than 7 characters.	Re-input.
SQ INVALID - INVALID SYNTAX	Self-explanatory	Re-input.
SQ INVALID - INTEGER MUST NOT BE GREATER THAN 65535	The maximum # of sectors requested for the FAST option is 65535.	Re-input.
SQ INVALID - INVALID ADDRESS	No address or an invalid address (for example, 004G15) was specified for the FROM or TO address with partial squash.	Re-input.
*** BAD SECTOR AND file-name OVERLAP - NO WAY TO SEPARATE THEM. SAVE AND/OR PURGE AND AT LEAST RETURN SQ VERIFY	Area marked in available table as "bad" overlaps with area allocated to file.	SQ cannot resolve this. Integrity of file is suspect. Remove the file after copying it to another disk for examination if necessary and run SQ VERIFY.
*** file-name AND file-name OVERLAP - THERE IS NO WAY TO SEPARATE THEM. SAVE ONE OR BOTH, PURGE AND AT LEAST RERUN SQ VERIFY	Area allocated to file overlaps with area allocated to another file.	The two areas cannot be separated. Copy each file individually to another disk for later examination if required, remove them both, rerun SQ VERIFY
SQ ABORTED - REQUESTED AREA ALREADY EXISTS	Request was made with FAST option for an available area which already exists.	None.
*** SQ ABORTED - INVALID DISK ALLOC. UNIT = 0	Disk label is probably corrupted.	Disk must be assumed useless and should be re-initialized.
*** DIRECTORY FID NEQ HEADER FID FOR FILE file-name. CORRECT USING CH AND RESTART SQ	Name of file in file directory name list does not match disk file header. The file-name displayed is that in the name list.	Enter "CH <FILE-ID> TO <FILE-ID> to correct the anomoly (this rewrites the disk file header).

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SQ INVALID - SPECIFIED DISK NOT AVAILABLE	Self-explanatory.	Make disk present and rerun SQ; check input.
*** SQ ABORTED NOTHING TO SQUASH IN THAT AREA	A partial squash was requested and SQ found nothing to do in that section.	None.
SQ ABORTED - NO WAY TO GET REQUIRED AREA	Area size specified in the FAST option cannot be obtained either because it is larger than the total available space or because certain areas cannot be moved to release available space. For example: Area # 1 100 units Available # 1 1000 units Area # 2 100 units Available # 1 cannot be used if areas # 1 and # 2 are in-use or system log files because areas # 1 and # 2 cannot then be moved.	Attempt to remove unwanted user files: re-input SQ.
*** MEMORY INCONSISTENCY OR SOME OTHER IRRECOVERABLE PROBLEMS - RERUN SQ	Internal work-tables in memory (used by SQ) are corrupted.	Rerun SQ VERIFY. If problem persists, request technical assistance.
*** ADDRESS MISMATCH - SAVE AND REINITIATE THE DISK	Some addresses in disk directory are probably corrupted.	Try to dump or copy files from the disk. Disk must be reinitialized before re-use.
*** IRRECOVERABLE ERROR ON DISK - SAVE AND/OR REINITIALIZE	Disk is corrupt.	Try to dump or copy files from the disk. Disk must be reinitialized before re-use.
*** TOO MANY FILES OPEN AND/OR BAD AREAS - NO WAY TO SQUASH THE DISK	Self-explanatory.	Save required files from disk, then reinitialize the disk.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
*** DISK INTEGRITY SUSPECT - USAGE MAP OF THE DISK WILL BE PRINTED (IF LIST IS NOT SET)	SQ has detected anc resolved an overlap situation; but disk remains suspect.	Scrutinise disk map, request technical assistance.
*** AREAS STILL MISSING - RERUN SQ VERIFY	A FAST squash has detected that areas of disk are not accounted for.	SQ VERIFY will return missing areas to the available table.
*** LAST SQ EXECUTION WAS ABNORMALLY TERMINATED WHEN MOVING FILE - file-name.INTEGRITY OF FILE SUSPECT. EXECUTION CONTINUES	System crash occurred while file areas was being moved.	Remove suspect file.
*** SQ ABORTED - I/O ERROR AT DISK ADDRESS @NNNNNN@	Hard error on disk persists after 10 retries.	If address is outside directory area, remove offerding area with XD utility.
*** I/O ERROR ON file-name FILE SKIPPED - EXECUTION CONTINUES.	Hard disk error encountered. File is not moved.	None.
*** WRITE ERROR IN NEW LOCATION WHEN MOVING FILE AREA. DISK ADDRESS - @NNNNNN@. THIS AREA SKIPPED - EXECUTION CONTINUING	Hard error encountered when moving file area to available area.	Available area is left as availale and should be XC-ed after squash EOJ.

NOTE

Error messages marked with "***" indicate that a hardware or system software error has occurred or that the disk itself is suspect. If these persist, request technical assistance.

General Guidelines

If the information contained on a disk is important always ensure that backup exists before attempting to squash it.

Always run "SQ VERIFY" before running an actual squash. This will give an indication of the state of the disk.

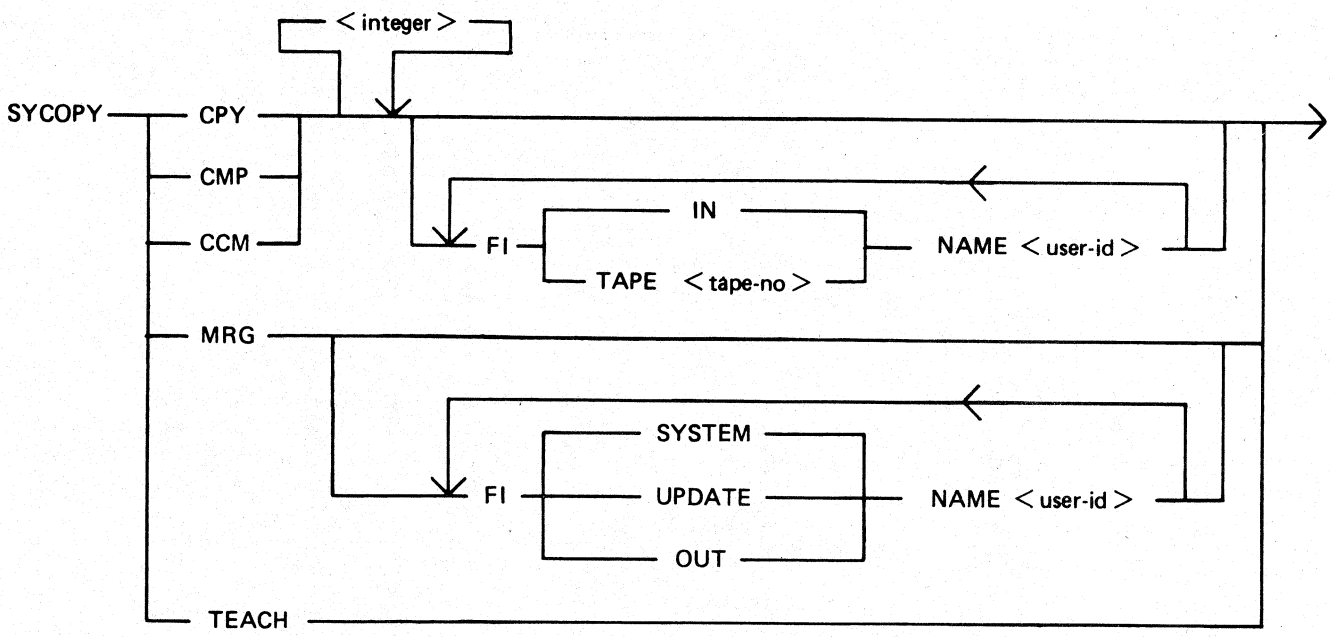
Do not allow disks to become too fragmented before squashing them. A full squash can be a lengthy process and can be avoided by running "SQ VERIFY" on a regular basis and running partial squash when the disk starts checker-boarding.

"SQ VERIFY" is a means of checking the integrity of any disk and if run on a regular basis may help pinpoint sooner rather than later any degradation in hardware performance or system software bugs. For disks that are in constant use "SQ VERIFY" should be run immediately after the first clear start of the day. This can help prevent catastrophic losses of information.

SYCOPY (Copy Library Tapes)

The utility SYCOPY provides the user with a means of duplicating, comparing or merging library tapes (or cassettes) with multiple copy capabilities (except for the merging option).

The initiating message has the following syntax:



where <integer> = 1-7 (defaults to 1)

The meanings of these options are:

file-equate is defined as one or more of the following entries:

FI default-name NAME actual-name

The default-names for input tape is "IN" and for output tapes "TAPE.1" through "TAPE.7' xcept in MRG function where the input tape names are "SYSTEM" and "UPDATE" and the output tape name is "OUT". All output tapes are locked after job termination.

CPY — makes <integer> copies of a library tape (created with the LD utility) with no compare. At end, the output tapes are locked, and the input tape is released.

Examples:

To copy one input tape labelled "IN" to two output tapes, labelled "TAPE.1" and "TAPE.2":

SYCOPY CPY 2

To copy one input tape labelled "FRED" to one output tape, labelled "FRED.OUT":

SYCOPY CPY FI IN NAME FRED FI TAPE.1 NAME FRED.OUT

CMP — compares <integer> tapes with one input tape. On completion, all tapes are released.
For the FILE000, the comparison works as follows:
– first records: only bytes 1-20 (byte 1 being the first) and bytes 28-38 are compared (check if library tape, and number of files).
– other records: only bytes 1-12 (filename) are compared.

The record sizes of input and output tapes must be the same. This restriction does not apply to buffer sizes, which may be different for input and output tapes.
Differences between the filesizes of the input tape and the output tape with which it is being compared will be detected; in this case, a comparison error for End-of-File or Not End-of-File will be displayed (see Output Messages).

Examples:

To compare one tape named "IN" with three tapes named "TAPE.1", "TAPE.2" and "TAPE.3":

```
SYCOPY CMP 3
```

To compare one tape named "FRED" with one tape named "TAPE.1":

```
SYCOPY CMP1FI IN NAME FRED
```

CCM performs CPY then CMP successively. On completion, the output tapes are locked, and the input tape is released.

Examples:

To copy one input tape named "IN" to three output tapes named "TAPE.1", "TAPE.2", "TAPE.3" and then to compare "IN" with "TAPE.1", "TAPE.2" and "TAPE.3":

```
SYCOPY CCM 3
```

MRG merges two input tapes (called SYSTEM and UPDATE by default) to one output tape (OUT by default), then compares the merged tape with the original tapes.
Duplicate files will be removed, with the UPDATE file taking precedence over the SYSTEM file.
The files will appear on the output tape in the following order:
- Duplicate Key Files (from second input tape)
- Non Duplicate Files (from first input tape)
- Remaining Files (from second input tape)

The record sizes of the input tapes must be the same, but buffer sizes may be different. The buffer size of the first input tape will be used for the output tape.
If more than one copy of the newly-merged tape is required, use CPY.
When a comparison error occurs, the faulty tape is either purged (for CCM or MRG) or released (for CMP) and an error message displayed. Refer to Output Messages for possible error messages.

Examples:

To merge two input tapes named "SYSTEM" and "UPDATE" and output one tape named "OUT":

```
SYCOPY MRG
```

To merge two input tapes labelled "A" and "UPDATE" to one output tape named "B":

```
SYCOPY MRG FI SYSTEM NAME A FI OUT NAME B
```

TEACH displays the syntax of the initiating message.

Example:

```
SYCOPY TEACH
```

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
number TAPE(S) COPIED	Successful copy function.	None.
number TAPE(S) COMPARED - NO ERROR	Successful compare function.	None.
number TAPE(S) COPIED AND COMPARED - NO ERRORS	Successful copy and compare function.	None.
2 TAPE(S) MERGED - NO ERROR	Successful merge function.	None.
number TAPE(S) COMPARED number TAPE(S) DELETED	One or more, but not all, of the output tapes has been deleted.	None.
number TAPE(S) COPIED AND COMPARED number TAPE(S) DELETED	One or more, but not all, of the output tapes has been deleted.	None.
INVALID SYNTAX - NO INITIATING MESSAGE	Specified invalid specification in initiating message.	Check input and re-enter if necessary.
INVALID SYNTAX - INIT MSG NOT COMPLETE		
INVALID SYNTAX - INVALID PARAMETER LIST <string>		
INVALID SYNTAX - TOKEN TOO LONG <string>		
COMPARISON ERROR ON <tape.id>/<file.id> - RECORD NO <number>	Compare function failed, possibly due to a hardware malfunction.	Re-run SYCOPY with good tapes and drive.
COMPARISON ERROR ON <tape.id>/<file.id> - NOT END OF FILE		
COMPARISON ERROR ON <tape.id>/<file.id> - END OF FILE		

If all output tapes are deleted, the following message will be displayed before the utility is aborted:

TAPES NOT IDENTICAL - SYCOPY ABORTED

Other Messages:

NOT LIB TAPE <tape.id> - SYCOPY ABORTED
RECORD SIZE DIFFERENT ON <tape.id> (for CPY, CMP, CCM)
RECORD SIZE DIFFERENT - SYCOPY ABORTED (for MRG)
MK NUMBERS DIFFERENT - SYCOPY ABORTED (for MRG)

TAPELR (List Library Tape Directory)

This utility allows the operator to print detailed information about the library tape files. Output will appear either on the line printer or the console printer.

Tapes about which information is required are identified by "library-tape-name". More than one tape name may be requested during a single run of TAPELR.

Format:

```
TAPELR library-tape-name
```

Examples:

To print detailed information about the files on a tape called PRTAPE:

```
TAPELR PRTAPE
```

To print detailed information about the files on tapes called PRTAPE and ICTAPE:

```
TAPELR PRTAPE ICTAPE
```

Output format:

Eight columns of information will appear for each library tape indicated. The column headings, the format of the "values" these columns contain, and the significance of these "values" is as follows:

HEADING	VALUE	SIGNIFICANCE
FILE NAME	12 character	File name
ACTUAL SIZE	7 digits	Number of recorcs in this file
MAXIMUM SIZE	7 digits	Maximum # of records this file may contain.
RECORD SIZE	5 digits	# of characters in each record
RECS/BLOCK	5 digits	# of records in each block
CREATED	5 digits	Date file was createc (Julian YYDDD)
ACCESSED	5 digits	Date file was last accessed by a program (Julian YYDDD)
FILE TYPE	8 characters	See Note below

Note: FILE TYPE will be one of the following:

DATA – normal data file
CODE – object program file
KEY – key file
SYSTEM – system file (for example, MCP,interpreters)
SRCELANG – source language file
SRCELIBR – source library file

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
library-tape-name NOT A RECOGNIZED DUMP TAPE	This tape was not created by either DUMP or UNLOAD functions of LD utility. It is ignored by the TAPELR utility.	None.

Note: Refer to "Common Utility Output Messages" for additional messages.

TAPEPD (Print Name of a Library Tape)

This utility allows the operator to print the names of files found on a library-tape. More than one tape name may be requested during a single run of TAPEPD.

Format:

TAPEPD library-tape-name

Examples:

To print the names of files found on a tape called PRTAPE:

TAPEPD PRTAPE

To print the names of the files found on tapes called PRTAPE, ICTAPE, and GLTAPE:

TAPEPD PRTAPE ICTAPE GLTAPE

Output format:

For each tape requested, the following information is displayed:

MT library-tape-name DUMPED ON day of week DD month YY contains:

This message precedes the names of files found on each tape. The list itself contains 3 files per line.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
library-tape-name NOT A RECOGNIZED DUMP TAPE	This tape was not created by either the DUMP or UNLOAD function of LD utility. It is ignored by the TAPEPD utility.	None.
END TAPEPD	End of job message.	None.

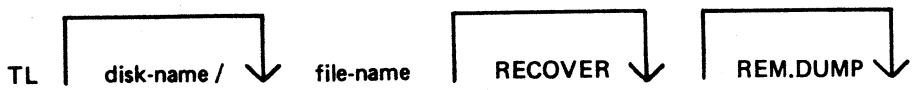
Note: Refer to "Common Utility Output Messages" for additional messages.

TL (Transfer Log Files)

If logging is enabled following any warmstart, a number of "log-files" are in use. The purpose of these files is to maintain a record of all input/output messages that appear on the SPO within this given period of time.

In order to produce easy access to all the files, they are consolidated into one large file. This is done through the use of the TL utility.

Format:



"File-name" is the name the user wishes the consolidated file to be called.

In order to permit a correct consolidation, there may be NO other programs running at the time when TL is initiated.

The utility will determine the number of files to be consolidated and also the size of the consolidated file. It will then transfer each "ready-to-transfer" , closing the consolidated file after each log-file has been transferred, until it reaches the file which was in an "active '" state at time of execution of TL.

The "RECOVER" option exists to allow the transfer and consolidation of the existing "active" log file which was in use when the system was last powered off. This transfer and consolidation is needed when logging fails (or is deliberately inhibited) during the succeeding warmstart. The option will only function if the "active" log file is not in use, that is, logging is inhibited. This may either be deliberate on the part of the operator, or because of a failure of TL to be automatically zipped during warmstart by the MCP. (A "no user disk" condition during the initial zip of TL can also cause this.) At all other times (that is, if logging is "active") any attempt to execute TL <file name> RECOVER will cause the utility to display:

"ILLEGAL USE OF RECOVER PARAMETER, ACTIVE FILE NOT CONSOLIDATED"

and it will consolidate only the "ready-to-transfer" files.

If the "RECOVER" option has not been specified, consolidation will end when the "active" file is reached.

All log-files transferred will be left in a "transferred" state.

TL is able to handle a "NO USER DISK" condition while consolidating the log files. If such a condition is encountered, TL will close any files it has open, display the following message, and go to End of Job:

"INSUFFICIENT DISK SPACE TO CONSOLIDATE LOG FILES"

If REM.DUP is specified in the initiating message, and a duplicate file condition exists during consolidation, the existing "old" file will be removed, a new one created and the following message displayed:

"<file-name> REMOVED"

If REM.DUP is not specified in the initiating message, then TL will wait with a duplicate file condition.

The file type for TL object code is @13@.

When the system is warmstarted, any existing SYS-LOG-FILES which were in the READY-TO-TRANSFER state are transferred into one file named SYS-LOG-HOLD. As they are being transferred, these files are scanned for entries with a type "E" (maintenance entry), or type "D" (statistics entry). These are transferred into both SYS-LOG-HOLD and into a newly created file named SYS.MLGjjjnn. Unlike SYS-LOG-HOLD, this file is not overwritten the next time file consolidation occurs. Thus a new maintenance entry file is created each warmstart and is made unique by the identifiers jjj (the number of days part contained in the Julian date) and nn (a numeric progression where nn is incremented by one for each maintenance file produced on that current day).

Examples:

To transfer all "ready-to-transfer" SYS-LOG files to LOGHOLD:

TL LOGHOLD

To transfer all "ready-to-transfer" SYS-LOG files to LOGHOLD on the disk called ARDISK1:

TL ARDISK1/LOGHOLD

Output messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
DISK disk-name NOT AVAILABLE	The disk disk-name is not on-line. TL will go to EOJ.	Put disk on-line or re-input initiating message
NO MAINTENANCE LOG FILE WILL BE PRODUCED	The file SYSCONFIG is not present or some problem occurred while accessing SYSCONFIG. TL will go to EOJ.	Logging of maintenance entries will take place. Check the file SYSCONFIG.
I/O ERROR DETECTED IN READ OF FILE file-name or I/O ERROR DETECTED ON WRITE TO FILE file-name	A read or write parity error has been detected. Following this will be a message indicating the record number at which the error occurred.	
NO SPECIFICATION GIVEN	No file-name parameter was present.	Re-input init. message.
ILLEGAL FILENAME file-name	Specified file-name contains too many characters.	Check input and re-input.
UNABLE TO OPEN TRANSFER FILE file-name	Utility cannot "create" a consolidated file with the name requested (for example, no disk space, no disk media).	Check with KA on available space: use RM if necessary and re-input. Check drive media.
FILE file-name NOT FOUND	The utility was unable to find the specified file-name	Correct the file-name and re-input.
FILE file-name IN USE	The specifiec file is in use.	Wait until the task using the specifed file goes to end of job.

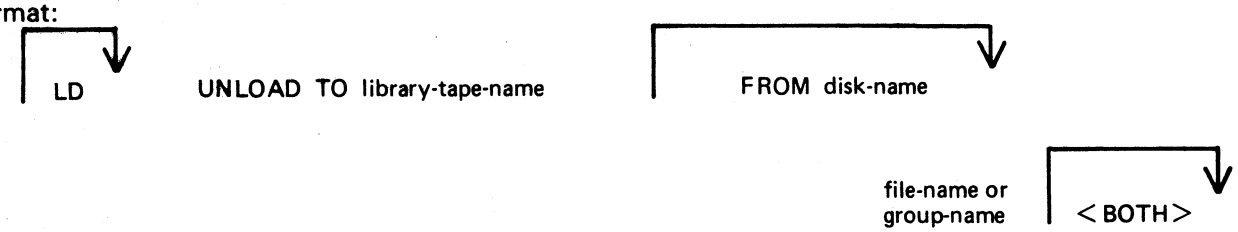
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO LOG FILES FOUND FOR CONSOLIDATION	This is displayed if the utility, on attempting to determine the number and sizes of log files to be transferred, was unable to find any with a file-id of the form "SYS-LOG-NN"	None.
ILLEGAL USE OF RECOVER PARAMETER - ACTIVE FILE NOT CONSOLIDATED	The utility is executed with RECOVER option. The active file can only be consolidated at warmstart.	None.
INVALID CHARACTER IN IDENTIFIER input	Some invalid character has been typed in the input.	Correct the input and re-input.
TRANSFER COMPLETED	Successful termination of the utility.	None.
NO READY-TO-TRANSFER FILES FOUND	No log-files in either an "active" or "ready-to-transfer" state were found. No consolidation will occur.	None.
NO ACTIVE LOG FILE FOUND	No file in the "active" state was found.	None
TRANSFER COMPLETED	IL successful End of Job.	None
PARITY ERROR IN READ OF FILE file-name or PARITY ERROR ON WRITE TO FILE file-name	Read or write parity errors found for specified files. The record number at which error occurred will be displayed. If utility continues processing, then the message "CONTINUING PROCESS" will display. Otherwise IL will go to End of Job and leave partially consolidated file, depending on where error occurred.	None

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO FILE NAME FOUND IN PARAMETERS	The "RECOVER" option was used and a file name was not given.	Correct the input and re-enter.
CONTINUE PROCESS	This message followed the parity error message if utility able to continue process.	None.

UNLOAD (Unload Files from Disk to Library Tape)

This function, a part of the utility LD, allows the operator to copy files from disk to a library tape. The files will be deleted from the disk after they have been copied to the tape.

Format:



If the <BOTH> option is used immediately after a request to dump a keyfile, the associated data file will also be dumped, provided it resides on disk.

Examples:

To dump all files from disk beginning with the letters, "PR" to a tape called PRTAPE; and then remove them from disk:

UNLOAD TO PRTAPE PR=

To dump the keyfile called AR200K and its data file from a disk called APBU to a tape called ARTAPE, removing them from disk after dumping:

UNLOAD TO APTAPE FROM APBU AP200K <BOTH>

To dump from the system disk files called AP020, AP030, and APTASK to a tape called APTAPE; removing them from disk after dumping:

LD UNLOAD TO ARTAPE AP020 AP030 APTASK

Since "UNLOAD" is a part of the utility LD, "LD" is actually what will appear in a mix message. To discontinue the UNLOAD function, DS mix-number/LD must be used.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO FILES IN THE FAMILY group-name ON DISK disk-name FOR UNLOAD	Specified group was not found on this disk.	Check input; re-input if necessary. Check for correct disk.
NO FILE file-name ON DISK disk-name FOR UNLOAD	Specified file was not found on this disk.	Check input and re-input if necessary. Check for correct disk.
file-name NOT DUMPED - IN OUTPUT USE DUMP ABANDONED - TAPE BEING PURGED	Specified file cannot be dumped as it is in use. If "ABANDONED" message is given, tape is purged and UNLOAD goes to EOJ.	Wait until the file is not in use and then re-enter the UNLOAD message for all files.

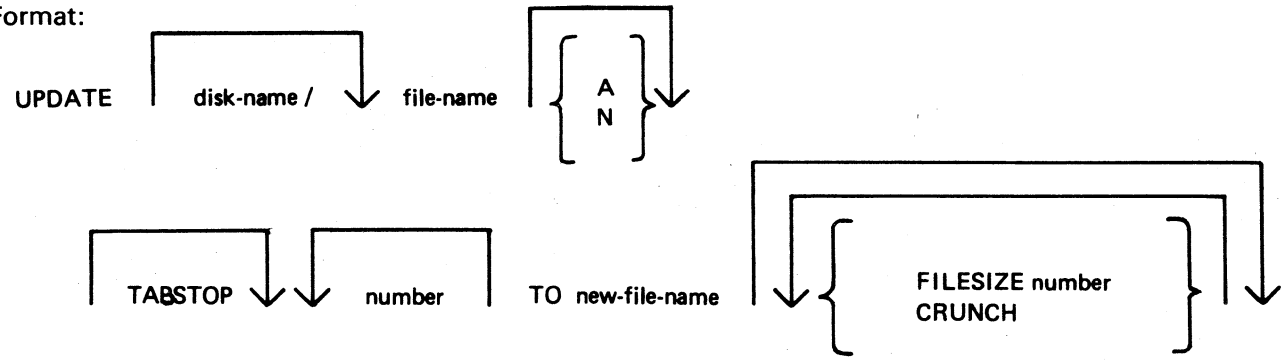
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name NOT DUMPED - HAS BEEN REMOVED. DUMP ABANDONED - TAPE BEING PURGED	Specified file was removed between the start of UNLOAD and the time when it was to be dumped to tape. File cannot be dumped. Tape is purged an UNLOAD goes to EOJ.	None.
file-name NOT DUMPED - HAS BEEN ALTERED> DUMP ABANDOND - TAPE BEING PURGED.	Contents of specifiec file were changed between the start of UNLOAD and the time when it was to be dumped to tape. File cannot be dumped.	Check input and re-enter if necessary.
file-name LOAD/DUMP DISCREPANCY	End of file has been reached before expected. Implies erroneous Disk File Header.	
file-name NOT DUMPED - DATA FILE NOT ON LINE	<BOTH> option was specified, but data file was not found on disk.	If specified data file dump is required, supply utility with backup copy of file if exists.
DUPLICATE file-name ALREADY BEING DUMPED	More than one recuest was made to UNLOAD same file.	None.
file-name REMOVED	UNLOAD successful; original file on disk was removed.	None.
file-name DUMPED	UNLOAD successful.	

Note: Refer to "Common Utility Output Messages" for additional messages.

UPDATE (Disk File Update)

This utility allows the operator to construct new disk files from existing files. "CREATE" and "AMEND" use many similar features.

Format:



The existing file must be a source or data file. Attributes such as Record Size will be taken from this file and used for the "new" file.

Input may be specified as "A" (alphanumeric) or "N" (hexadecimal). (see CREATE utility for details).

By specifying TABSTOP in the initiating message, UPDATE will set up tab positions coinciding with the end of the console line as well as any other tabs specified.

The "number" option may be used to set "tab" positions for character input. (see CREATE utility for details).

The maximum number of records likely to be written to the new file may be specified using the FILESIZE option. If no total number of records is specified, the number will be taken from the old file.

The CRUNCH option allows the operator to specify that the new file should occupy the minimum area of disk, but never be extended.

The utility operates in three modes: "Record Modify" (PK2), 'Record Select" (PK3) or "Record Insert" (PK4).

PK1	PK2	PK3	PK4	PK5	PK6
write last & get next	modify	select	insert	delete	EOJ

OCK3 will display the current tab position anywhere in the record (except at the start of a record where the record number is displayed).

OCK4 will output a display of the above options. If OCK4 is selected while in MODIFY, SELECT or INSERT mode, an asterisk (*) will be displayed preceeding that mode.

PK1 is used to write the last record processed to the new file and 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.

PK5 is used to 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.

If PK3 is used, the required record is identified by logical record numbers using this format

The "number" cannot be less than 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. When found the selected record will be printed, with its record number in th old file followed by the record number that the next record written to the new file will take. "Record Modify" (PK2) or "Record Insert" (PK4) 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 is used to make alterations to existing records. This PK operates as PK2 in the CREATE utility (see CREATE for details).

PK4 allows the operator to insert additional records in the new file after the last selected record of the old file. Input may be made in accordance with the specified tab stops. UPDATE will accept input only up to the next tab position. Fill characters will be displayed up to the next tab position if that tab is not reached. 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. 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: to insert a record at the beginning of a new file, Record "0" should be selected in Record Select Mode, prior to Selecting Record Insert Mode.

Default tab positions have been selected to allow a maximum number of characters to be entered on one line. UPDATE uses 19 positions for the old and new record numbers and 100 for the record contents.

Tabs may be selected manually in addition to the default tab positions.

Default tab positions for UPDATE are as follows:

```
(source or data alphanumeric) : 101 201 301 401
(data hexadecimal)            :  56 101 151 201 251 301 351 401 451
```

Examples:

```
UPDATE FILEA TABSTOP TO FILED                    101 201 301 401
UPDATE FILEB TABSTOP 151 200 TO FILEF            101 151 200 201 301 401
UPDATE FILEC N TABSTOP 46 420 TO FILEG            46  51 101 151 201 251
                                                 301 351 401 420 451
```

If the input file-name (OLD.FILE) is the same as the output file-name (NEW.FILE), the old version of the file will be removed and the generation number of the new version of the file will be set to one more than that of the old version.

Examples:

To update a source file called "APFILE" of record size 40 bytes into a file called "APFILE2".

```
UPDATE APFILE 5 10 15 20 TO APFILE2
```

The utility will illuminate PK1 and PK6. By pressing PK1, next sequental record will be selected and printed.

As the utility is already in the Record Select Mode, by typing a record number, the specified record number and its contents are printed.

```
4    4    ABCDEFGHIJKLMNOPQRST
```

Note that the first "4" is the sequence number in the old "APFLE" and the second "4" is the sequence number in the "APFILE2" file.

At this point the following PKs are available for selection:

PK1 – select next sequential record and print

PK2 – modify the selected record

PK4 – insert new record after selected record (that is, "4")

PK5 – delete the last selected record by selecting next record

PK6 – terminate the utility

To replace characters within a selected record, press PK2 and type the replacement

D:ZZZZ: OCK1

resulting in

4 4 ABCDZZZZIJKLMNOPQRST

To insert characters within a selected record, type

Z:XXXXXX: OCK2

resulting in

4 4 ABCDZXXXXXXZZZOPQRST

To insert a record after record 7 of the existing file, press PK3 (Record Select Mode) and type a record number.

7 OCK1

Note: At this point the record selection number given cannot be less than the last selected record, for example, records from 1 through 3 cannot be selected).

Press PK4 (Record Insert Mode) and utility will print last selected record number on left and next record number after that and allows operator to key in record to be inserted.

7 8 AAAAAA

The record inserted will have a sequence number of "8" in the file "APFILE" and will contain "AAAAAA".

Output messages:

Refer to the section on the "CREATE" utility for output messages.

WL (What Log file)

This utility allows the operator to determine the number of log files present and their status.

Format:

WL

The utility displays the following information:

FILE STATE	SYS-LOG-
ACTIVE	nn
READY-TO-TRANSFER	nn nn etc
TRANSFERRED	nn nn etc
NEXT-ACTIVE	nn (nn)
NOT USED	nn nn etc

where nn represents values between 01 and 16.

When the ACTIVE file becomes full, the MCP executes the WL utility and hence the operator is informed of the states of log files. The user can then decide whether or not to execute TL so as to preserve the log files before any overwriting of log files occurs.

Output messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
NO SYS-LOG-NN FILES FOUND	Logging has not been initiated	Request technical assistance if logging is required (refer to appropriate section of the SOG)

XD (Delete Bad Disk Sectors)

This utility allows the disk directory to be marked such that selected portions of the disk will not be used. The utility will normally be used after recurrent errors of the message

DK...ERROR

where the dots indicate further information. Refer to section 7, MCP output messages, for the following numbered messages:

2 PARITY ERROR
3 TIMEOUT ERROR
4 ADDRESS ERROR
45 PARITY ERROR (fatal to program)
46 TIMEOUT ERROR (fatal to program)
47 ADDRESS ERROR (fatal to program)

The further information will indicate the disk address at which the failure occurred

The utility is initiated as follows:

Format:

XD disk-name address length

The disk-name is the disk-id of the disk from which sectors are to be deleted. The area to be deleted is given in hexadecimal by the starting address and length.

Example:

To delete sixty-four sectors starting from hex 395F from disk PR2B:

XD PR2B 395F 40

NOTE

The specified sectors must not be in use as part of a file. The area must be made available by first removing any file if necessary.

Warnings:

Once sectors are deleted via XD from a disk, they can be restored to use only by a disk initialization. Do not therefore XD a larger area than required.

As XD alters the disk directory, do not run any other programs with it.

Do not execute XD from the same disk as the one from which sectors are to be deleted: for example, it is recommended that XD is always executed from the system disk and always deletes sectors from a user disk.

Output messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
@length@ SECTORS FROM @address@ DELETED	Successful termination of XD.	None.
DISK disk-name FOR XD NOT AVAILABLE	Specified disk is not available.	Check input: make disk ready.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PLEASE RUN SQ ON DISK disk-name	Available table is full and available entries are lost. SYSMEM will be opened on the specified disk with otherwise LOCKED.	Run SQ.
ONLY @length@ SECTORS CAN BE DETECTED	Only the given length can be XD-ed because the amount specified is greater than the available space at that point.	Use the KA utility to determine if any files can be removed to increase the available space surrounding the bad area.
AVAILABLE TABLE FULL - ENTRY @address@ @length@ LOST	No entries left in available space table for XD to complete properly.	The disk may still be used, but a KA will indicate some sectors which cannot be accessed; these may only be retrieved by initializing the disk.
SECTORS FOR XD NOT AVAILABLE	Requested sectors are allocated to a file, missing, or previously XD-ed.	Refer to a KA listing to determine which file, if any, can be RM-ed to make the sectors available.
CANNOT DELETE SECTORS FROM @address@	Only the part of the area specified for deletion is available. The utility will remove the available area only.	None
ADDRESS @address@ BEYOND END OF DISK disk-name	The sector address specified is greater than the address of the last physical sector on the disk.	None

Note: Refer to "Common Utility Output Messages" for additional messages.

SECTION 5
THE SORT/MERGE

INTRODUCTION

This section describes the capabilities of the SORT facility. There are two modules: the sort itself, known as the "sort intrinsic" (file-name SORTINTRINS), and an interface to this intrinsic which allows the user to specify particular sorts and merges. The latter module is sometimes called "the sort", but is more properly called the "sort language processor" (file name SORT). The sort intrinsic is implementation-dependent, as it uses specific hardware features where possible (although output messages are standardized), while the sort language processor is a CMS common item.

This section first describes the user interface to the sort, and then covers the various facilities in some detail.

The interface to the sort from COBOL programs is described in the COBOL language reference manual.

GENERAL FEATURES

The following capabilities are provided:

The records within a file may be sorted on a series of specified keys, each key ascending or descending, using a regular sort or an in-place sort.

A tagfile (suitable for use as an ADDROUT file in RPG or for limited access in COBOL) may be created from a file using a series of keys, each ascending or descending.

A key file (suitable for full indexed access) may be created using a specified unsigned key (ascending only), with an optional check for duplicate keys.

A number of files may be merged using a series of keys, each ascending or descending.

The regular and index sort intrinsics are able to sort files up to the maximum allowable size in CMS, which is 1,048,560 records.

The inplace intrinsic has a B 90 implementation limit of 400,000 records.

The merge intrinsic will produce an output file of up to 1,048,560 records.

The memory required for a sort is calculated dynamically, with a default size of 15K bytes.

The workfile buffer default size for index sorts and regular sorts is 720 bytes.

INVOKING THE SORT

The sort is executed by entering the name SORT preceded by disk-name if not on the system disk, and followed by the sort language specifications or an asterisk plus "star-file" name. The "star-file" contains the sort-language statements, and may reside on card, cassette, or disk. The star-file name may be omitted: in which case the sort statements must be on a system disk file named SORTSPEC.

Examples:

To invoke the sort using a star-file named SRTLANG on the system disk:

SORT *SRTLANG

To execute the sort from disk PB4 using a star-file SORTSPEC on the same disk:

PB4/SORT *PB4/SORTSPEC

If the sort specification is given in the initiating message it cannot be longer that 255 characters. If the sort statement is zipped from a user program it cannot be longer than 716 characters. If it is not possible to specify a complex sort or merge within these limitations, a star-file should be used.

For a one-part star-file name of 7 characters or less, the file will be searched for first on cards, then on cassette, then on the system disk. For a two-part name the file will be searched for on a user disk. For a one-part name of more than 7 characters, the file will be searched for on the system disk.

If the required file is not found, the sort displays

FILE filename UNAVAILABLE
FIX AND REPLY "OK" ELSE <NULL>

and waits on an ACCEPT.

There are two alternative responses:
make the file present and enter OK to the ACCEPT, to resume execution, or
enter a null response (terminator only) to the ACCEPT, to cause EOJ.

If the specification statements are provided in the initating message, control characters such as carriage return and line feed are treated as space characters.

A star-file on cassette must be created by the COPY utility, not the LD utility.

A star-file on disk must be of type data or source, and should not be in use by other programs.

Input statements may be printed on the printer, unless inhibited by a user option (see later), or if provided in the initiating message.

THE SORT LANGUAGE

The specification for a sort consists of three statements:

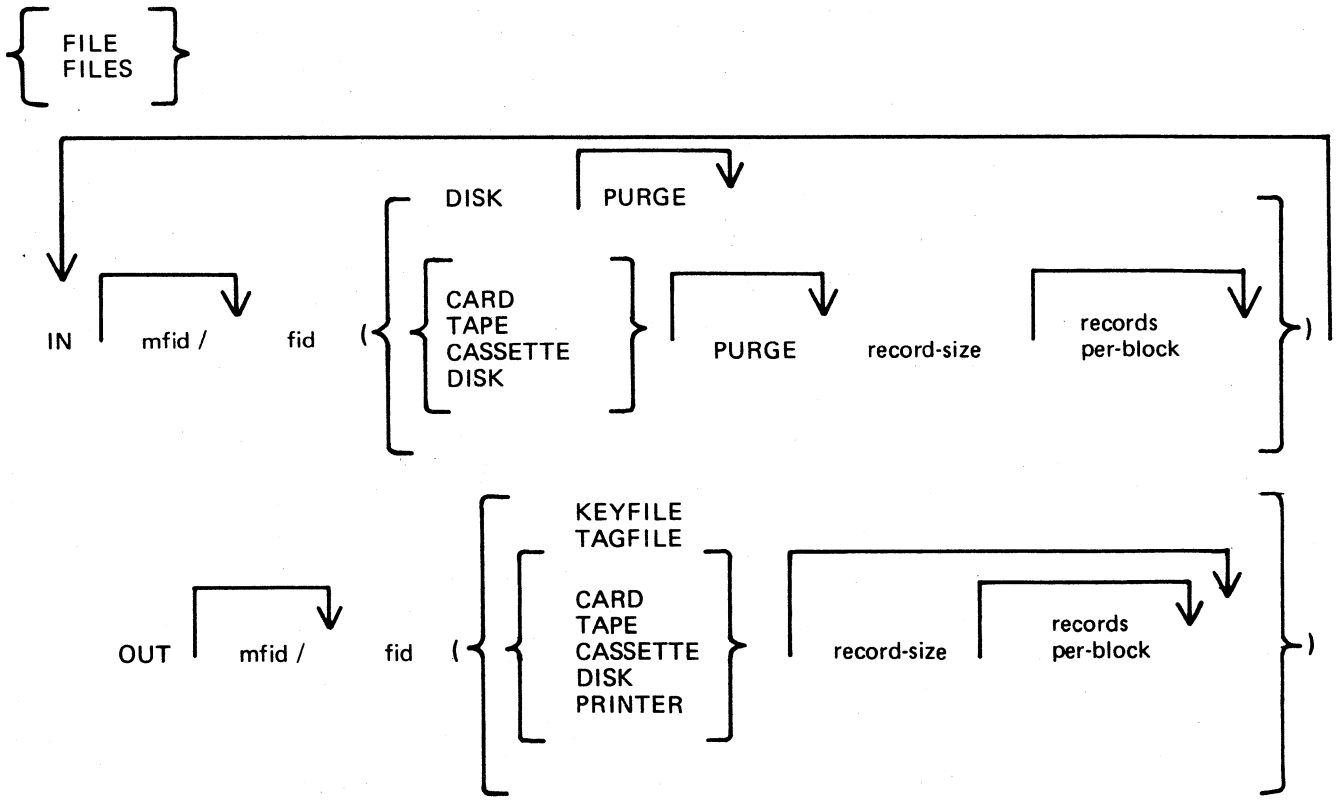
the FILE statement
the KEY statement
the USER-OPTION statement

There must be one file statement, one key statement, and, optionally, one or more user-option statements, in any particular sort invocation. All keywords are reserved: that is, they can only be used in the place specified below and cannot be used for other purposes such as filename.

The File Statement

This consists of two parts; the first describes the input file(s) and the second describes the output file. Multiple input files are used only for the merge, which is specified as a user-option (see later). A sort must have only one input file; a merge may have up to 16 input files.

Format:



The parentheses "(" and ")" may be replaced by the characters "<" and ">" respectively.

Rules for the file statement are as follows:

The medium for the specified input file(s) and output file is indicated by the keyword DISK, CARD, etc. When the medium is DISK, the absence of a disk-name (mfid) indicates the system disk. CARD refers to 80-column card only. TAPE refers to magnetic tape only. CASSETTE refers to magnetic tape cassette only. DISK refers to any kind of disk-device. The input file for a tagfile or keyfile creation must be on disk.

The PURGE option indicates that the input file(s) are to be purged after use.

The record size and records-per-block values are numeric values. When the input medium is DISK, the record size and records-per-block may be omitted. For a merge specification, input disk file descriptions with record size specifications may be interspersed with descriptions without such specifications.

If the records-per-block is omitted and record size is given, a blocking factor of 1 is assumed.

In all cases (except an index sort), input and output files must have the same record sizes.

The values of record size and records-per-block may be omitted for output files. For a sort, the values assumed are those of the input file. For a merge, the values assumed are those of the first specified input file.

For a keyfile creation sort, the output specification enclosed in parentheses must be the single word KEYFILE. The output will be on disk and record and block sizes are not user definable.

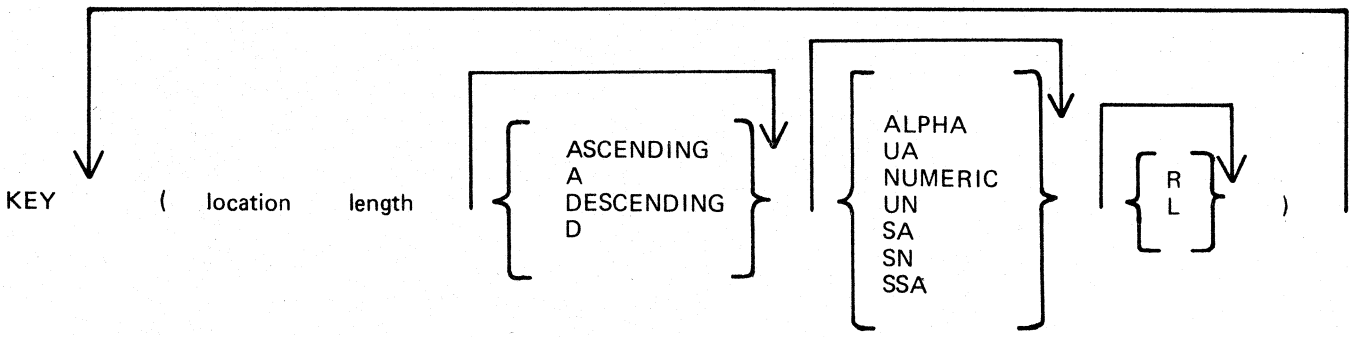
For a tagfile creation sort, the output specification enclosed in parentheses must be the single word TAGFILE. The output will be disk and record and block sizes are not user definable.

The Key Statement

This statement defines the record key(s) that are used for the sort or merge.

A number of keys may be specified, each key description being enclosed in parentheses. The first key will be the major key and additional keys will be minor keys of decreasing significance.

Format:



The "location" is a numeric value specifying the position of the key relative to the start of the record, in 4-bit units. The first 4-bit unit has a location of 1. The key location is given by the position of the left-hand 4-bit unit in the key (which, depending on the key format), may be a character or a sign. The key should start on a byte boundary unless a record sort with a numeric key is performed.

The "length" is a numeric value specifying the key length, in 4-bit units. This must include the sign, for signed keys.

The keywords ASCENDING and DESCENDING determine the order of collation. These keywords may be abbreviated to A and D respectively. If omitted, the default is ASCENDING.

The format of the key is specified by one of the following keywords:

ALPHA or UA – unsigned 8-bit alphanumeric
NUMERIC or UN – unsigned 4-bit numeric
SA – signed 8-bit alphanumeric
SN – signed 4-bit numeric
SSA – 8-bit alphanumeric with separate sign

The default is ALPHA.

For a signed key, the position of the sign is specified by one of the following keywords

R – right-hand (least significant) end of key

L – left-hand (most significant) end of key

The default is L.

For a description of key types and sign zone interpretation, see later under "KEYS".

The User-Option Statement

These statements have three functions:

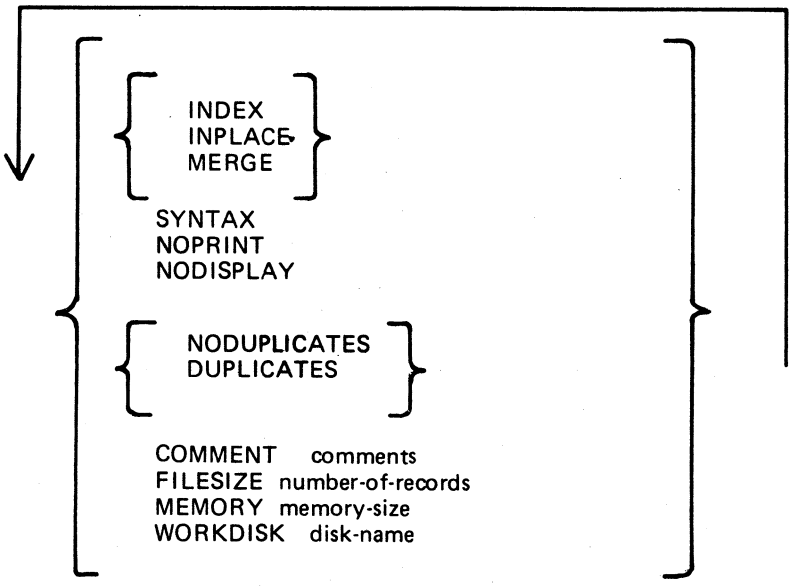
to specify which function is required

to tailor a sort or merge to the particular machine configuration (memory, printer availability, etc).

to add comments

The user-option statements are optional; if more than one are used they may appear in any order relative to each other or to the file and key statements.

Format:



The type of sort is given by one of the keywords INDEX, INPLACE, or MERGE. If one of these does not appear, a regular full record sort is assumed. The keyword INDEX specifies the creation of a keyfile or tagfile, depending on the output file details (see FILE statement). The keyword INPLACE specifies a full record sort using a minimal amount of disk work space. The keyword MERGE specifies a merge of several input files.

Note that if the INPLACE option is specified on a system that has not implemented the inplace sort, then a regular full record sort will be performed.

The keyword SYNTAX specifies that a check on the correctness of the sort statements is to be made without the sort actually being performed.

The keyword NOPRINT stops the listing of the sort statements on the printer. If used, this keyword should be the first entry. If the statements are input via the SPO they are not printed and so this keyword is not required in this case. The NOPRINT option also affects the printing of error and warning messages (see later).

The keyword NODISPLAY controls the display of messages on the SPO during the sort. This option can be used both in initiating messages and from file-oriented statements. It suppresses startup and termination messages. It does not affect the display of error and warning messages. Error and warning free sorts and merges will show no SPO activity if this option is used.

The keyword NODUPLICATES specifies that duplicate keys are not allowed in a keyfile creation. The keyword DUPLICATES specifies that duplicate keys are allowed in a keyfile creation. Both options are valid only when creating a keyfile. If neither is specified, the default is NODUPLICATES.

The keyword COMMENT introduces comment text. The end of the comment text is either the end of the input or the end of a record if the input comes from a starfile. Comments may appear between user-option statements and between file descriptions and key descriptions.

The keyword FILESIZE provides the following capabilities:

specification of sort disk work space where the input file is not on disk (this use is not required if the input is from disk).

specification of maximum size of the output file if on disk.

allowance for future expansion of the output disk file where the sort/merge will not by default create a large enough file.

This keyword should be followed by a number giving the specified maximum number of records. For non-disk output files, the value is used for optimization purposes. If not used, default values are assumed where necessary. This option is not applicable to the inplace sort or to keyfile or tagfile creations.

The keyword MEMORY specifies the amount of non-overlayable work area to be used by the sort. This option is not applicable to the merge or to the inplace sort. If this is not enough for a successful sort, then this option is overridden. The memory size is in bytes; for example, MEMORY 1024.

The keyword WORKDISK enables the regular sort to utilize disk space in an efficient manner. It is not applicable to the merge or inplace sort. When the work-disk is specified, the sort locates up to half the work space on that disk, with the rest on the system disk. If this option is not used, but the input or output file is resident on a user disk, the work space is shared between that disk and the system disk. In all other cases the work space is located entirely on the system disk. The named disk may be any type of disk applicable to the system in use.

Examples

To sort the system disk file INP.FILE using the key starting at character 5 of length 3 characters, creating a system disk file OUT.FILE:

```
SORT FILE IN INP.FILE (DISK) OUT OUT.FILE (DISK)
        KEY (9 6)
```

To create a keyfile OUTKEY.FILE on disk PR2 from a data file INP.FILE1 on disk PR2, using a 5-byte key starting at the first byte:

```
SORT FILE IN PR2/INP.FILE1 (DISK)
        OUT PR2/OUTKEY.FILE (KEYFILE)
        KEY (1 10)
        INDEX
        COMMENT DUPLICATES NOT ALLOWED
```

To merge the three system disk files FILE1, FILE2 and FILE3 into an output file MERGE.OUT:

```
SORT
FILE IN FILE1 (DISK) FILE2 (DISK) FILE3 (DISK)
OUT MERGE.OUT (DISK)
KEY (5 10)
MERGE
```

FUNCTIONAL DESCRIPTION

The five functions of the sort are described here:

Regular record sort.
Inplace record sort.
Keyfile creation.
Tagfile creation.
File merge.

Regular Record Sort

All the records contained within the specified input file are ordered using one or more keys. Deleted records (see later) are not included in the output file. See later for details of the keys. Refer to figure 5-1 for an example of a regular record sort, where the key is starting in byte 3 and is 5 characters long, and the sort is in ascending order. The X's refer to any other characters.

The input file must be wholly contained on one hardware type, although it may be a multi-reel or dual-disk file. No other programs may write to this file during the execution of the sort.

The sort uses non-overlayable memory during execution. The amount is calculated according to the input file and key sizes. The amount may be specified as a user option, in which case the specified amount is used unless it is less than enough for a successful sort. In the latter case the specified value will be overridden.

The sort uses disk work space, of up to 2.2 times the size of the specified input file. For the location of the work disk space, refer to the WORKDISK user option (see earlier). This work space is returned to the system at end-of-job.

Inplace Record Sort

This is the same as the regular record sort, except that the records are sorted within the input file. No new output file is created. The time taken is substantially greater than a regular sort, for the same input specifications. If deleted records are present in the file before the sort, they are removed: hence the number of records in the file may decrease after it has been ordered.

The inplace sort uses non-overlayable memory during execution. The size of this area cannot be specified at initiation.

The input file must be on disk. No other programs may access this file during execution of the inplace sort. The output file must be the same as the file specified for input.

If a particular system does not implement an in-place sort, a regular sort will be performed instead.

The inplace sort uses disk work space, of 0.2 to 0.3 times the size of the input file. When the input file is resident on a user disk, up to one-half of the work space is located on that disk, otherwise all work space is located on the system disk. This work space is returned to the system at end-of-job.

Keyfile Creation

A new file (the "keyfile") will be created containing one record for each record of the input file (the "data file"). The keyfile is sorted in order of the specified keys, and each keyfile record contains the key and a pointer to the corresponding record in the data file. Any deleted records in the data file are not referenced in the keyfile. Note that the records in the data file are not re-ordered and deleted records in the data file are not removed. Refer to figure 5-2 for an example of a keyfile creation, where the key is starting in byte 3 and is 5 characters long, and the sort is in ascending order. The X's refer to any other character.

Duplicate keys are not allowed unless specified (see the user-option statements DUPLICATES and NODUPLICATES). If they occur, then the record number is displayed on the SPO for each such occurrence, and the sort will continue but the output keyfile will be purged at end-of-job.

The keyfile creation uses disk work space, of up to 2.2 times the size of temporary file created by the sort in this case. This file is large enough to contain one record with the key value and record number for each record in the input file. For the location of the work disk space, refer to the WORKDISK user option (see earlier). This work space is returned to the system at end-of-job.

Certain key values are not allowed during a keyfile creation. The key must not consist of all binary zeroes, or must not contain any byte whose value is hex FF. If such a key is encountered, the record number is displayed on the SPO, and the sort will continue but the output keyfile will not refer this record in the data file.

Tagfile Creation

A tagfile creation is similar to a keyfile creation, except that the output file contains only the record pointers, and not any key values. The tagfile records, however, are ordered in key value order, as specified by the sort. Any deleted records are not referenced in the tagfile. Refer to figure 5-3 for an example of a tagfile creation, corresponding to the keyfile creation in figure 5–2.

A tagfile is a null keyfile. It is suitable for use as an ADDROUT file in RPG, and for limited indexed access in COBOL (the tagfile is read sequentially).

Disk space requirements are the same as for keyfile creation.

Merge

The merge merges up to 16 input files, using one or more specified keys, producing one output file. Deleted records in the input files are not included in the output file. If there are duplicate keys values, the order in which they are placed in the output file is given by the order in which the input files are specified.

Each input file must be wholly contained on one hardware type, although it may be a multi-reel or dual-disk file. No other programs may write to these files during the execution of the merge.

Each input file must have the same record size and the same position and length for each key. Each file must be already correctly ordered on the specified keys. If this is not the case, the merge will terminate prematurely after displaying a message on the SPO.

Refer to figure 5-4 for an example of a merge of two files, with a key starting at byte 3 which is 5 characters long. The X's and Y's refer to any character.

The merge uses non-overlayable memory during execution. The size of this area cannot be specified at initiation: it will be approximately equal to the sum of the block sizes of the input files and the output file.

The merge does not use any disk work space.

Details Of Sort Keys

A "key" is the field within each record that is used for sorting or merging. If several distinct field within a record are specified, then each field is a separate key. The relative order of importance of the keys is determined by the order in which they are specified. Figure 5-5 illustrates this with a two-key sort, using the KEY statement.

KEY (5 6 ALPHA) (15 2 DESCENDING ALPHA)

The X's indicate any character. In this example the three-byte field is the major key, sorted in ascending order: the one-byte key is a minor key sorted in descending order within the order of the major key.

For a keyfile creation, only one key may be used. This key must be a maximum of 28 bytes long, must be a whole number of bytes in length, and must start on a byte boundary.

For all sorts except keyfile and tagfile creation, there can be up to 10 keys. The sum of the length of all keys (including signs) must be a maximum of 29 bytes.

The available key types are discussed here, under the keyword specified in the KEY statement (see earlier):

ALPHA (or UA)

Unsigned 8-bit alphanumeric field, containing ordinary ASCII characters. Note that this may consist of the 8-bit ASCII digits "0" to "9" but still be termed alphanumeric. This key type is the default.

NUMERIC (or UN)

Unsigned 4-bit numeric field, where each 4-bit unit is a binary coded decimal digit, 0000 to 1001 (0 to 9).

SA

Signed 8-bit alphanumeric field. Each byte is an ordinary ASCII character (including the digits 0-9), except that either the first or the last character indicates the sign. Whether the sign is the first or last character is specified by the keyword L (left) or R (right). The default is L (first character; leading sign). The convention for coding the sign character is given in Table 5-1. These characters are termed "overpunched signs" by analogy with historical punched card systems.

SN

Signed 4-bit numeric field. Each 4-bit unit is a binary-coded decimal digit, 0000 to 1001 (0 to 9), except that either the first or the last 4-bit unit indicates the sign. Whether the sign is the first or last 4-bit unit is specified by the keyword L (left) or R (right). The default is L (first 4-bit unit); leading sign . The convention for coding the sign is given in Table 5-2.

SSA

8-bit alphanumeric field with separate sign. Each byte is an ordinary ASCII character (including the digits 0 to 9), with the sign given by an ASCII character in either the first or last character. Whether the sign is given by the first or last character is specified by the keyword L (left) or R (right). The default is L (first character); leading sign . The convention for coding the sign character is given in Table 5-3.

The position of a sign within a signed key (left or right) must be the same throughout all occurrences of the key. Signed keys are ordered so that negative values come before zero and positive values

8-bit keys may start on 4-bit unit boundaries, unless the separate sign type (SSA) is used, or the key is to be used in keyfile or tagfile creation.

Deleted Records

A deleted record is denoted by every byte in the record (including the key) containing the value hex FF. The action taken by the various sort options is discussed earlier. Deleted records may be physically removed by the FS utility.

Output Messages

Output messages cover warnings and errors. Messages are generated by both the sort intrinsic and the sort language processor. The intrinsic messages are numbered by event numbers in the same way as MCP output messages (see section). The sort language processor messages are numbered in a similar way.

Messages can be divided by number as follows:

0-99

Sort language processor messages, displayed on the printer. Such messages appearing in the list below that are followed by a series of dots (...) should be read with the phrase NEAR COL XXX (with XXX replaced by an appropriate column number) in place of the dots.

0-34

Warnings, where corrective action is attempted.

35-39

Warnings, where no corrective action is attempted.

40-59

Errors in syntax (that is, the format of the sort statements is incorrect).

60-99

Errors in semantics (that is, an inconsistency has been detected in the statements, such as a key position greater than the record size).

170-200

Sort intrinsic messages, displayed on the SPO.

Certain messages may be suppressed by the NOPRINT and NODISPLAY keywords in the sort statements.

The NOPRINT option suppresses listing of the sort statements on the printer by the sort language processor. If this option is set, a maximum of five errors and four warning messages are directed to the SPO, with only the error or warning number being given (no explanatory text). The NOPRRNT option has no affect on sort-intrinsic-generated messages.

The NODISPLAY option suppresses display on the SPO of start-up and termination messages by the sort intrinsic. Messages in the list below that are marked with an asterisk (*) are those that are suppressed when this option is set. Note that it is not possible to suppress individual messages; every applicable message is suppressed if the option is set. The NODISPLAY option has no affect on sort language processor messages.

Number	Message
0	EXPECTED SLASH NOT FOUND, "/" INSERTED ...
1	EXTRA "FILE IN"...
2	MERGE INTRINSIC IGNORES <WORK-DISK OPTION>
3	OVERLENGTH PART OF <LABEL NAME> IGNORED ...
4	INPLACE INTRINSIC IGNORES <WORK-DISK OPTION>
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	OUTPUT BUFFER SIZE TOO BIG, <BLOCK FACTOR> REDUCED ...
10	<USER OPTION> ALREADY INVOKED, LATEST USE ...

Number	Message
11	MERGE <SORT TYPE OPTION> NOT SPECIFIED
12	OVERLENGTH PART OF <DISK NAME> IGNORED ...
13	MISSING "FILE IN" ...
14	INDEX <SORT TYPE OPTION> NOT SPECIFIED
15	EXTRA "KEY" ...
16	<FILE SIZE OPT> VALID FOR MERGE/REGULAR SORT ONLY
17	MISSSNG "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 used
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- AND OUT-FILE RECORD SIZES MADE EQUAL
28	<BLOCK FACTOR> TOO LARGE, MAXIMUM ASSUMED ...
29	INPLACE SORT MUST HAVE IDENTICAL IN- AND OUT-FILES
35	IDENTICAL IN/OUT - FILES WILL PRODUCE DUPLICATE FILE
36	NOT NECESSARY TO PURGE CARD FILE ...
37	ALPHANUMERIC KEY DOES NOT START ON BYTE BOUNDARY ...
40	<KEY STATEMENT> ALREADY PROCESSED, NOW ...
41	<DIGIT STRING> EXPECTED ...
42	<CHARACTER STRRNG> EXPECTED ...
43	<SEPARATOR STRING> EXPECTED ...
44	<RCRD-BLCK PAIR> MUST BE GIVEN FOR NON-DISK IN-FILE ...
50	NO <FILE STATEMENT> SPECIFIED
51	ILLEGAL WORD ...
52	<LETTER STRING> EXPECTED ...
53	MISSING <LABLE NAME> ...
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	FINAL STATEMENT INCOMPLETE ...
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	KEY LENGTH 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 MULTIPLE TAPE ...
71	MERGE INSTRINSIC 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

Number	Message
75	ALPHANUMERIC KEY LENGTH NOT EVEN NUMBER OF 4-BITS ...
76	<MEDIA> MUST BE DISK FOR IN-PLACE SORT
77	IN- AND 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 ...
86	CURRENT SUM OF KEY LENGTHS OUT OF RANGE ...
170	DUPLICATE RECORD <record number>
171	ILLEGAL INDEX KEY IN RECORD <record number>
172	RECORDS LOST / GAINED BY SORT-MERGE
173	<number> DUPLICATE RECORDS
174	<number> RECORDS CONTAINING INVALID INDEX KEYS
175*	<number> DELETED RECORDS
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>> NEAR RECORD <record number> ON <file name>
185	UNORDERED MERGE INPUT FILE <file name> NEAR RECORD <record number>
186	TOO MANY RECORDS FOR SORT-MERGE
187	DUPLICATE RECORDS-KEYFILE NOT BUILT
188	INITIATING MESSAGE INVALID
189*	SORT-MERGE VER x.y.z INITIATED FROM <mix number>/ <program name>
193	INPUT RECORD SIZES UNEQUAL - BAD FILE <filename>
194	IN/OUT RECORD SIZES BAD - OUTPUT SIZE CHANGED
195	BAD RECORD/BLOCK SIZE FOR OUTPUT DEVICE
196	KEY OVER-RUNS RECORD END
197	CANNOT SPLIT INDEX FILE
198	<number> PARITY BLOCKS
199	INDEX INPUT FILE NOT TYPE DATA

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 – parity on output file
5 – sort workfile error
6 – input file error
7 – output file error
8 – parity on sort workfile
9 – parity on input file (block ignored)

data file input	record number	data file output
X X 1 0 5 2 0 X X X	1	X X 0 1 0 0 3 X X X
X X 1 0 0 3 5 X X X	2	X X 0 2 1 4 7 X X X
X X 0 1 0 0 3 X X X	3	X X 1 0 0 3 5 X X X
X X 6 9 7 1 2 X X X	4	X X 1 0 0 3 6 X X X
X X 6 9 8 4 3 X X X	5	X X 1 0 5 2 0 X X X
X X 1 0 0 3 6 X X X	6	X X 3 2 4 0 0 X X X
X X 3 2 4 0 0 X X X	7	X X 6 9 7 1 2 X X X
X X 0 2 1 4 7 X X X	8	X X 6 9 8 4 3 X X X

Figure 5-1. Regular Record Sort

data file input	record number	keyfile output
X X 1 0 5 2 0 X X X	1	0 3 0 1 0 0 3
X X 1 0 0 3 5 X X X	2	0 8 0 2 1 4 7
X X 0 1 0 0 3 X X X	3	0 2 1 0 0 3 5
X X 6 9 7 1 2 X X X	4	0 6 1 0 0 3 6
X X 6 9 8 4 3 X X X	5	0 1 1 0 5 2 0
X X 1 0 0 3 6 X X X	6	0 7 3 2 4 0 0
X X 3 2 4 0 0 X X X	7	0 4 6 9 7 1 2
X X 0 2 1 4 7 X X X	8	0 5 6 9 8 4 3

Figure 5-2. Keyfile Creation

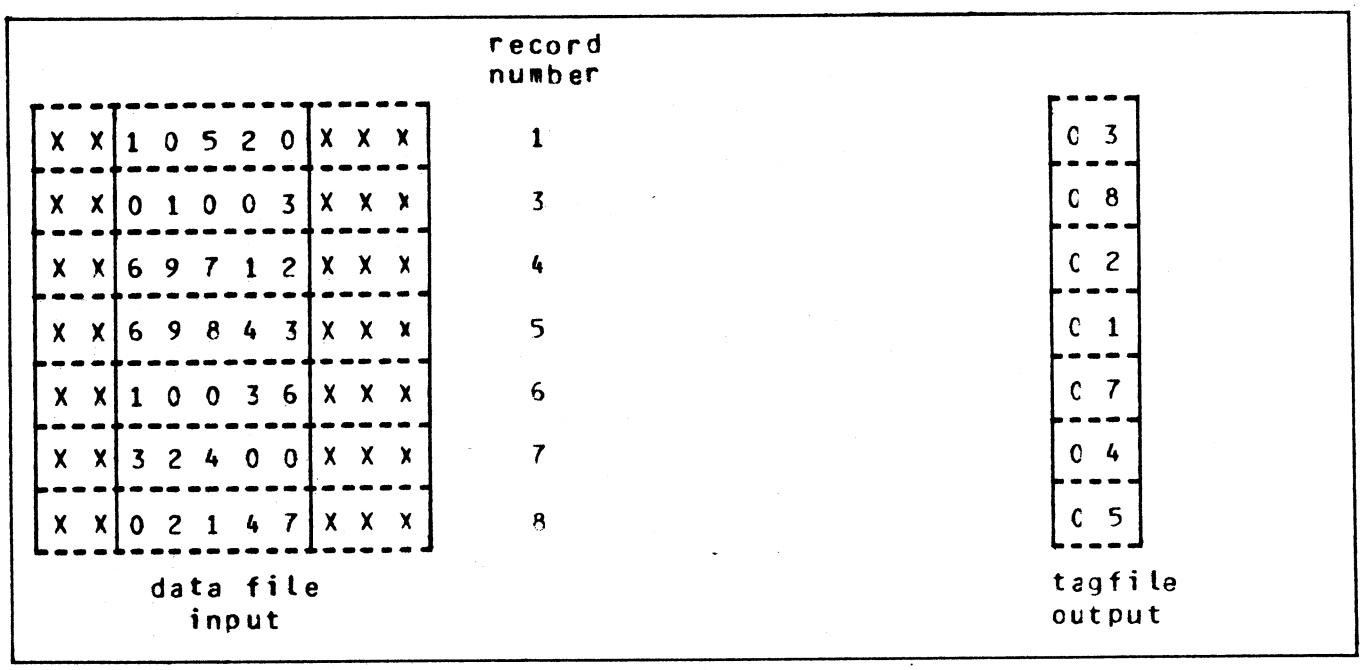


Figure 5-3. Tagfile Creation

X X	0 1 0 0 3	X X X
X X	0 2 1 4 7	X X X
X X	1 0 0 3 5	X X X
X X	1 0 0 3 6	X X X
X X	1 0 5 2 0	X X X
X X	3 2 4 0 0	X X X
X X	6 9 7 1 2	X X X
X X	6 9 8 4 3	X X X

Y Y	0 8 7 2 6	Y Y Y
Y Y	1 0 0 3 8	Y Y Y
Y Y	2 0 0 0 1	Y Y Y
Y Y	4 2 1 6 8	Y Y Y
Y Y	4 9 1 3 7	Y Y Y

input files

X X	0 1 0 0 3	X X X
X X	0 2 1 4 7	X X X
Y Y	0 8 7 2 6	Y Y Y
X X	1 0 0 3 5	X X X
X X	1 0 0 3 6	X X X
Y Y	1 0 0 3 8	Y Y Y
X X	1 0 5 2 0	X X X
Y Y	2 0 0 0 1	Y Y Y
X X	3 2 4 0 0	X X X
Y Y	4 2 1 6 8	7 Y Y
Y Y	4 9 1 3 7	Y Y Y
X X	6 9 7 1 2	X X X
X X	6 9 8 4 3	X X X

output file

Figure 5-4. File Merge

X	X	3	6	4	X	X	5	X	X
X	X	1	0	2	X	X	4	X	X
X	X	9	7	3	X	X	3	X	X
X	X	0	2	0	X	X	7	X	X
X	X	1	0	2	X	X	0	X	X
X	X	3	6	4	X	X	8	X	X
X	X	5	2	7	X	X	9	X	X
X	X	6	3	0	X	X	0	X	X

data file input

X	X	0	2	0	X	X	7	X	X
X	X	1	0	2	X	X	4	X	X
X	X	1	0	2	X	X	0	X	X
X	X	3	6	4	X	X	8	X	X
X	X	3	6	4	X	X	5	X	X
X	X	5	2	7	X	X	9	X	X
X	X	6	3	0	X	X	0	X	X
X	X	9	7	3	X	X	3	X	X

data file output

Figure 5-5. Multiple Key Sort

Table 5-1. Sign Convention For Signed 8-Bit Alphanumeric Fields

Key Value	Hex Code	ASCII Character
-0	2D	-
-0	7D	}
+0	30	0
+0	7B	{
-1	4A	J
-2	4B	K
-3	4C	L
-4	4D	M
-5	4E	N
-6	4F	O
-7	50	P
-8	51	Q
-9	52	R
+1	31	1
+2	32	2
+3	33	3
+4	34	4
+5	35	5
+6	36	6
+7	37	7
+8	38	8
+9	39	9

Note: any other hex code in the sign character is interpreted as positive, with the key value given by the binary value of the right-hand 4 bits of the character.

Table 5-2. Sign Convention For Signed 4-Bit Numeric Fields

Key value	Binary Code (BCD character)
negative	0101 (5)
positive	0011 (3)

Note: any value other than 0101 (5) is interpreted as positive.

Table 5–3. Sign Convention for Separate Sign Character with 8-bit Alphanumeric Fields

Key value	ASCII character (hex value)
negative	"-" (2D)
positive	"+" (2B)

Note: any character other than "-" is interpreted as positive.

SECTION 6
COMPILATION FACILITIES

INTRODUCTION

Compilation of programs written in CMS COBOL, RPG and MPL can be performed with the CO (compile) utility. CO is a normal utility program residing on disk. It is used to co-ordinate the various parts of the CMS COBOL, RPG and MPL compilers. Each compiler consists of several object code files (called "passes") and produces a number of workfiles to pass information between each pass. The CO utility allows initial input to be made to the compiler by specifying such things as input and output file names.

Additionally, CO allows multiple executions of each compiler by storing compiler workfile information in a master file called CO.MASTER on the system disk. The compiler passes have access to the information in this disk file. Information in this file also allows the CO utility to perform restarts if the system halts during a compilatin. This restart facility eliminates the need to rerun a compilation from pass one if one or more passes have already completed successfully.

CO uses some standard names for input and output files, which can be changed by the inital CO message. The basic CO operation is given in Figure 6-1.

Initial input to CO is either from the initiating SPO message or through macro (star) files or through a disk file called 'CO.STARTUP" on the system disk. CO generates the CO.MASTER file used to co-ordinate the compiler passes. There is an option to produce a CO listing. Information provided to CO enables the user to describe the following:

input patch file
input source file
output source file
output object program
output compilation listing
compiler workfiles

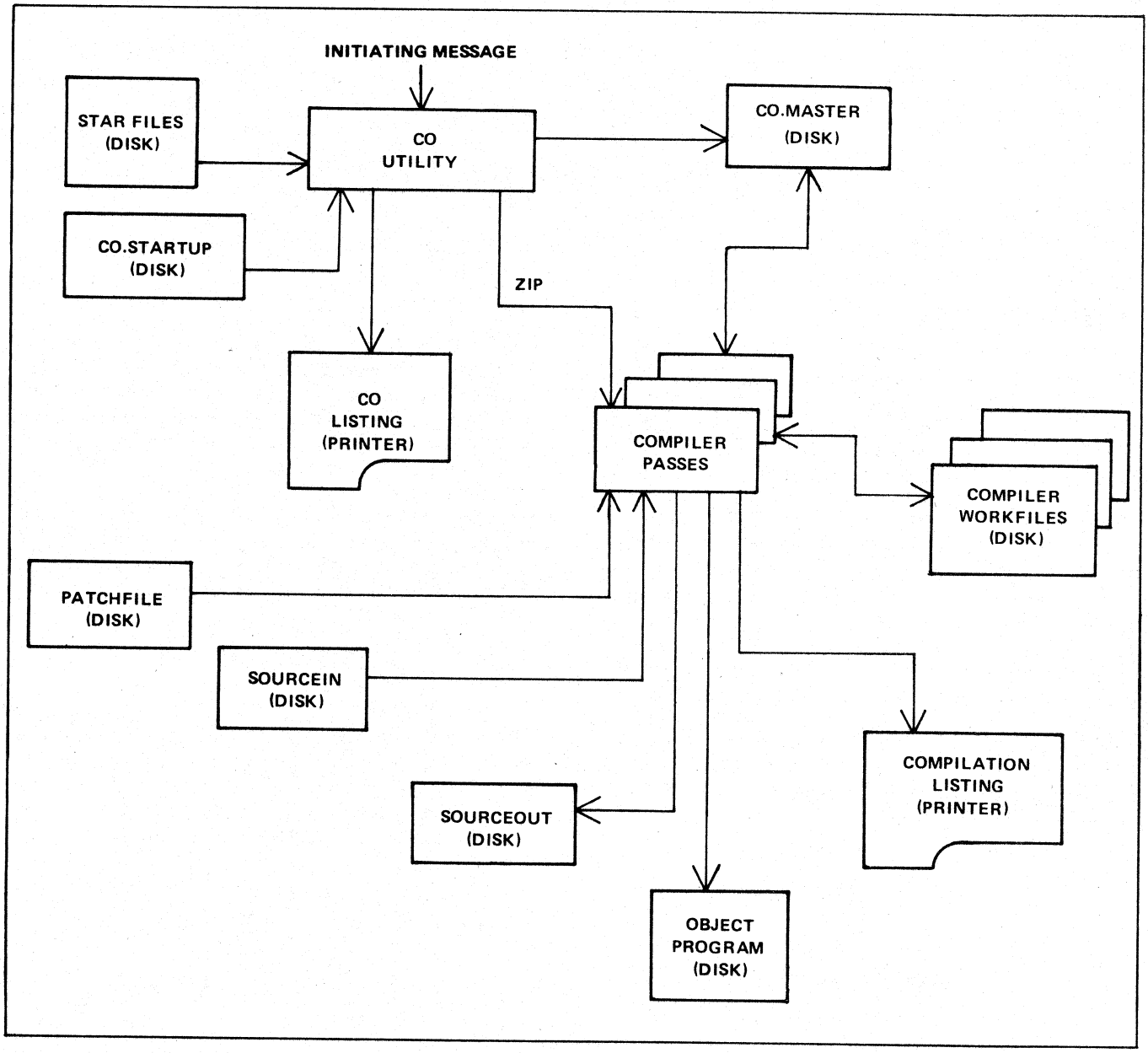


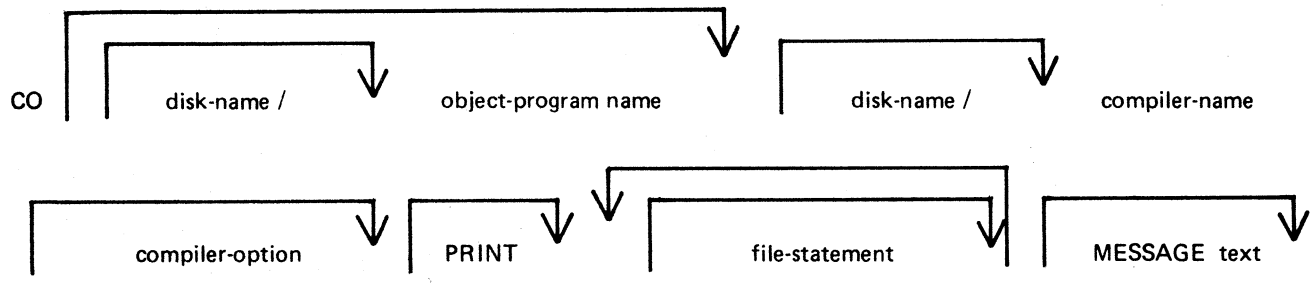
Figure 6-1. Operation Of CO Utility

CO provides the ability to set "dollar options" for the compilation.

CO operates in two basic ways
to initiate and control a single compilation.
to interrogate compilation status, and restart or clear an aborted compilation.

TO INITIATE A SINGLE COMPILATION

Format:



This initial input can be entered on the SPO when executing CO. CO can also be started with no additional information from the SPO: in this case the information is either on a card file or on a disk file called "CO.STARTUP" on the system disk. Selected parts of the input can be provided as star files see later in this section. All disk files used for inputting information to CO can be 80 byte card-image records created by CMS CANDE or CREATE, with all information in the first 8 records and the first 72 characters of each record.

Semi-colons may separate any clauses after the compiler-option, for readability.

The object-program-name is optional. If not specified, it will be created on the system disk with a name as follows:

"COBOBJECT" – for COBOL compilations
"RPGOBJECT" – for RPG compilations
"MPLOBJECT" – for MPL compilations

The name of the disk for the generated object program may also be specified. If not specified, the system disk is assumed.

The compiler-name may be one of the following:

COBOL	– the COBOL compiler
RPG	– the RPG compiler
MPL	– the MPL compiler
MPL.1	– the MPL.1 compiler
MPL.2	– the MPL.2 compiler
MPL.BINDER	
COBOLXREF	– the separate COBOL cross-reference program
RPGXREF	– the separate RPG cross-reference program
OPTLIST	– the separate COBOL/RPG optimizer

If the optional disk-name is given before the compiler-name, then the compiler may be executed from a user disk so long as all passes are on the specified disk. If no disk name is given then the compiler must be on the system.

The compiler-option may be one of the following:

SYNTAX (abbreviation SY)
LIBRARY (abbreviation LI)
GO (alternative SAVE)

With a "compile for syntax", no object program is generated. For COBOL and RPG, COBSVERTER is not executed; for MPL, the compilation stops at the end of pass 3.

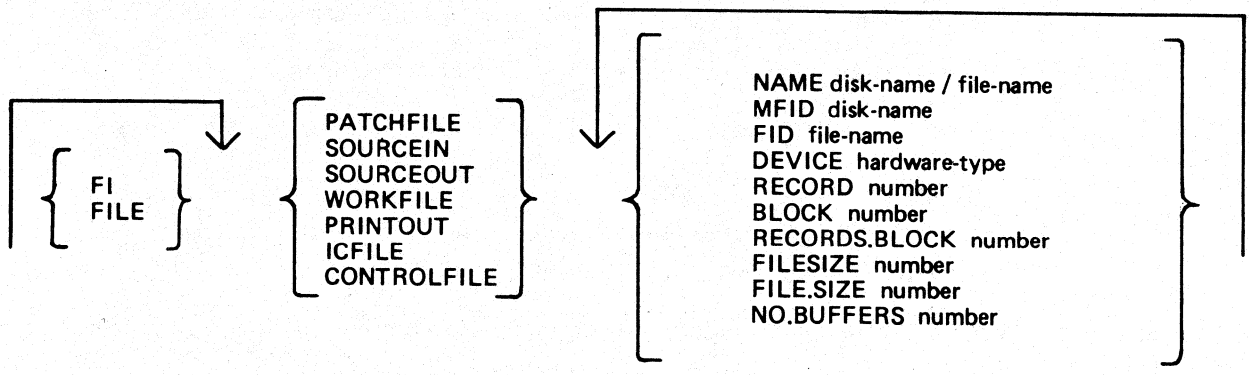
With a "compile to library", an executable object program is created if there are no syntax errors, and saved on disk with the object-name as specified in the CO statement. A compile to library is the default compiler-option.

A "compile and go" is the same as a compile to library, with the additional feature that the object program is executed. The CO utility goes to EOJ after zipping the object program.

If the PRINT option is specified, an edited listing of the startup message is output, along with a list of settings for all the file parameters which are modifiable through CO. A list of default dollar card settings for the compilation is output along with any dollar options entered via the CO message. This object listing also contains a log of all error messages displayed.

The file statements allow names and other attributes to be set for the compiler input and output files. The general form of these statements is given here, but not all attributes may be set for all files: these will be discussed in turn.

Format:



The meaning of the file attributes are as follows:

NAME

This specifies the CMS file name plus disk name and may be of the form MFID/FID.

MFID

This specifies the disk name of the given file. The default is the system disk. The MFID must be a maximum of 7 characters.

FID

This specifies the file name of the given file. The default is the system disk. The FID must be a maximum of 12 characters.

(Note: FID and MFID may be used together, or NAME may be used instead of the MFID/FID combination).

DEVICE

This specifies the type of peripheral of the input or output file. The default for the PRINTOUT file is line printer. The default for all other files is any disk. The device is specified by a 2-character mnemonic as follows:

```
card readers:
        any card reader                     AR
        any multi-function unit             AM
        80-column reader                    R8
        80-column reader/punch              M8
        96-column reader                    R9
        96-column reader/punch              M9
card punches:
        any card punch                      CP
        any multi-function unit             AM
        80-column punch                     P8
        80-column reader/punch              M8
        96-column punch                     P9
        96-column reader/punch              M9
tapes and cassettes (NRZ or PE)
        any tape                            AT
        write-disabled reel                 MT
        write-disabled cassette             CT
                (alternatives: CS, CASSETTE)
        Note: if a write-disablec oevice is specified
        for an output file, CO will substitute a
        write-enabled device in the specification, and
        issue warning...
printers:
        any printer                         AP
        serial printer (console)            SP
        line printer                        LP
disks:
        any disk, default catridge          DK
                (alternatives: DC, DISK)
        Burroughs Super Mini                DM
        Fixed disk                          DF
        pack                                DP
```

FILESIZE

This attribute specifies the maximum number of records in the output file. If used with PATCHFILE, the number specifies the total number of source lines to be compiled, including dollar cards and "COPY statements" in the case of COBOL compilation. The number may be followed by the letter K to denote thousands. For example, the statement

FI SOURCEOUT FILESIZE 4 K

specifies a filesize of 4000 records. A space must separate the "K" from the number.

An alternative spelling of the keyword is "FILE.SIZE".

The files are as follows:

NO.BUFFERS

This attribute specifies to the compiler the number of buffers to be used for the file in question. It is applicable to PATCHFILE, SOURCEIN, SOURCEOUT, PRINTOUT, CONTROLFILE and ICFILE statements. The valid range of values is 1-16. The default is zero, which will cause the individual compiler defaults to be used.

PATCHFILE

This is the primary source input file, and contains dollar records and source records which may optionally be merged with a secondary input file to produce an output source file. Attributes which may be set, and their default values are:

NAME	PATCHFILE
MFID	system disk (0000000)
FID	PATCHFILE
DEVICE	DK
FILESIZE	0

SOURCEIN

This is the optional secondary input file. Attributes which may be set, and their default values, are:

NAME	SOURCEIN
MFID	system disk (0000000)
FID	SOURCEIN
DEVICE	DK

SOURCEOUT

This is the optional source output file produced by merging PATCHFILE and SOURCEIN. Attributes which may be set, and their default values, are:

NAME	SOURCEOUT
MFID	system disk (0000000)
FID	SOURCEOUT
DEVICE	DK
FILESIZE	4 K
RECORD	80
RECORDS.BLOCK	9
BLOCK	720 for disks 240 for cassettes 2000 for magnetic tapes

WORKFILE

This refers to the intermediate workfiles produced by the compiler during the compilation. The only attribute that can be specified, and its default, is:

MFID	system disk (0000000)

PRINTOUT

This refers to the listings produced by the various compiler passes during the compilation. The only attribute that can be specified, and its default, is:

DEVICE	LP

ICFILE

This is only applicable when executing MPL.2 directly, and identifies the intermediate code file (or semi-compiled file).

CONTROLFILE

This is only applicable when executing MPL.2 or MPL.BINDER directly, and identifies the file containing the necessary control information.

The MESSAGE statement

The reserved word MESSAGE indicates the start of the list of dollar options for that compile. The text consists of a list of one or more dollar cards, taken from the list given below, separated by spaces.

For example, for an MPL compilation the following would be valid:

MESSAGE $LIST $XMAP $SEGMENT FR20

Use of this feature is not valid for COBOL compilations.

Examples:

To compile the COBOL source program AR678S, and create the object program AR678, both on the system disk:

```
CO AR678 COBOL LI FI PATCHFILE FID AR678S
```

To merge the RPG patch file RQ20P with the source file RQ20S, and create a new source RQ20SN and object RQ20NEW, all on disk RDEV:

```
CO RDEV/RQ20NEW RPG LI; FI PATCHFILE NAME RDEV/RQ20P;
FI SOURCEIN NAME RDEV/RQ20S; FI SOURCEOUT NAME
RDEV/RQ20SN; MESSAGE $MERGE
```

To compile the MPL source program MTEST.S from the disk USR1 to produce object MTEST on disk USR1, with CO listing, and compiler listing on the console:

```
CO USR1/MTEST MPL LI; FI PATCHFILE NAME USR1/MTEST.S;
PRINT; FI PRINTOUT DEVICE SPA; MESSAGE $LIST
```

To compile source PATCHFILE with COBOL and create object COBOBJECT on the system disk:

```
CO COBOL
```

To patch COBOL source CS500 (found on tape) with cardfile CRDPATCH and produce an object program CNEW on disk F1, putting compiler workfiles on disk FSCRATCH:

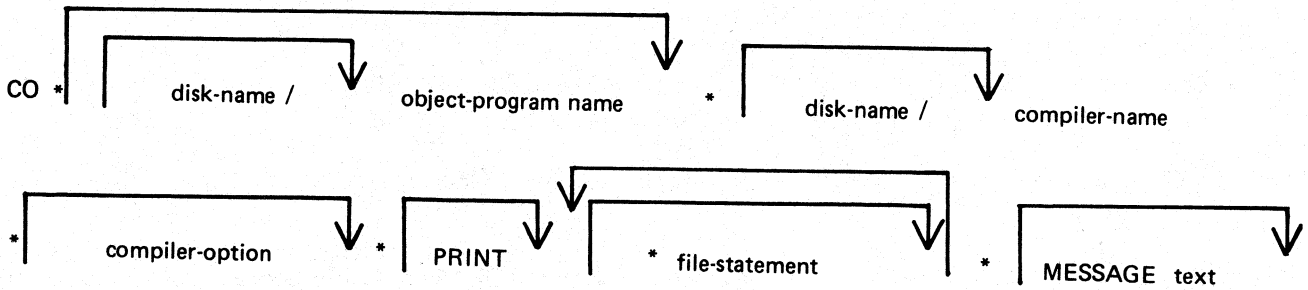
```
CO FI/CNEW COBOL LI;
FI PATCHFILE FID CRDPATCH DEVICE AR;
FI SOURCEIN FID CS500 DEVICE AT;
FI WORKFILE MFID FSCRATCH;
```

USE OF MACRO CALLS

All or part of the initiating message to CO may be provided as data in disk files. This is indicated by an asterisk (star) in the message.

Following the asterisk must be a valid file name, including the disk-name if not on the system disk. When the initiating message is scanned, the contents of a star-file are included in the scan. At the end of the star-file contents, scanning continues with the primary message. The complete initiating message, or individual clauses, may be included in the star-files. The message format for CO has been repeated below with an asterisk where a star-file may be used:

Format:



Example:

Consider four star-files, with names and contents as follows:

name	contents
FILE1	AA RPG GC
FILE2	PATCHFILE NAME BB
FILE3	DEVICE AR
FILE4	SOURCEOUT FID CC

Then the following two initiating messages to CO are valid:

CO *FILE1 PRINT *FILE2 DEVICE AR

CO *FILE1 PRINT *FILE2 *FILE4

but the following two messages are invalid, because the contents of FILE3 is wrong, as it starts in the middle of a file-statement:

CO *FILE1 PRINT *FILE2 *FILE3

CO *FILE1 PRINT PATCHFILE NAME BB *FILE3

COMPILER DOLLAR OPTIONS

In the following list, (D) indicates the options which are set by default. This notation is also used in the output listing from CO, to distinguish default settings from deliberate dollar card settings.

For COBOL, no dollar options may be set or reset by CO. For a fuller description of the available RPG and MPL options, consult the relevent compiler manual.

RPG

```
$ LIST        (D)        $ NLIST
$ SEQ                    $ NSEQ
$ XMAP                   $ NXMAP
$ SUPR                   $ NSUPR
$ LOGIC                  $ NLOGIC
$ MERGE                  $ NMERGE
$ SEVERE      (D)        $ NSEVERE
$ MAP                    $ NMAP
$ NAMES                  $ NNAMES
$ NEW                    $ NNEW
```

Any RPG dollar options specified in the startup message for CO will override any occurrences of that option appearing in either the PATCHFILE or SOURCEIN input files.

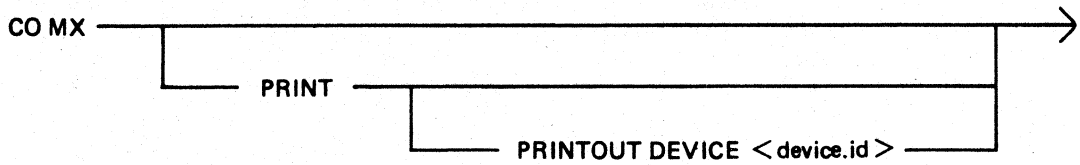
MPL

```
$ LIST                   $ CODE
$ NOLIST                 $ NEWTAPE
$ LISTE                  $ FORMAT
$ LISTP                  $ SEGMENT file-name
$ SEQUENCE               $ SEGSIZE integer
$ XMAP
$ NOWARNING
```

Any MPL dollar options specified through CO are used only as initial settings, and may be overridden by dollar cards appearing in either of the source input files.

TO INTERROGATE THE STATUS OF COMPILATIONS

CO may be used to determine the status of any compilations. This is done by interrogating the CO.MASTER file. Output may optionally be directed to serial or line printers. The initiating syntax is of the form:



where:

CO MX directs output to the SPO
CO MX PRINT directs output to the serial printer
CO MX PRINT PRINTOUT DEVICE <device.id> directs output to the named output device (for example, LP will direct output to a line printer).

This function may be used at any time, provided that the mix is not already full. It yields information in the following form:

```
CO          MIX-NUMBER                    n
            COMPILER NAME                 compiler-name
            CURRENT COMPILER PASS         compiler pass name
            PROGRAM NAME                  object-file-name
            WORKFILE CHARACTER            "A"
```

(where "A" is a character used by CO in the meaning of workfiles so that different compilations can precede with separate workfiles). In addition to the above information, if a compilation is either awaiting Restart or Clear (see later), or has been Restarted but has not yet gone to end-of-job, the following will be displayed:

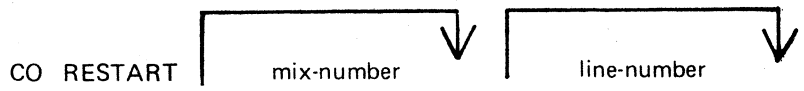
(THIS CO IS IN RESTART MODE)

Note that the "CO MX" function will give information about all current compilations marked in the CO.MASTER file, even if they were prematurely stopped by a system failure. This is different from the MCP "MX" intrinsic, which will only give the jobs currently in the multiprogramming mix.

TO RESTART AN ABORTED COMPILATION

This function may be used to restart a compilation which has been terminated prematurely due to a system failure such as a clear start or ZIP failure. The message is:

Format:



If only one compilation was being done, then the simple message

CO RESTART

is sufficient. The compilation is resumed at the beginning of the pass in which the failure occurred. If more than one compilation was being done, the mix-number must be supplied, which is the mix-number of the particular CO that was running at the time of failure. This number is determined either from the SPO log, or the CO MX facility.

If the failure occurred during the printing phase of a COBOL compilation (pass COBOL7), then this pass can be restarted at a specified line-number. The line number must be in the range 000000 to 065535. For example, the message

CO RESTART 2 010000

will cause the CO with mix-number 2 to be restarted, which will cause COBOL7 to start printing at line 010000.

Once a RESTART has been initiated, no new compilations can be started until the restarted compilation has gone to normal EOJ. If that is not possible, for instance because a file is not present, then the restarted job should be CLEARed (see later) to allow other compilations to be initiated.

Other CO versions executing are undisturbed by a RESTART operation.

When a restarted job terminates, whether naturally or as a result of a CLEAR (see later), the message

"RESTART COMPLETED"

is displayed on the SPO. If a new CO, having the same mix-number as the one that failed, is started up before the failed CO can be restarted or cleared, then the block of information in CO.MASTER for the failed CO is lost and that compilation cannot be restarted. In this case the followwng message is displayed:

"CO MIX-NUMBER n CANNOT NOW BE RESTARTED
COMPILATION BLOCK RE-USED".

TO CLEAR AN ABORTED COMPILATION

If a clear start or other failure has occurred, and it is decided not to Restart compilations, then on or all of the compilations may be cleared. References to one or all of the compilations are deleted from the CO.MASTER file, and workfiles belonging to the compilation are removed unless the compiler requested that they be saved. The message is:

Format:

```
CO CLEAR [ { mix-number } ]
           { ALL      }
```

Providing a mix-number clears only the compilation whose controlling CO had the specified mix-number. The keyword ALL clears all compilations known to CO. If only one compilation was being done, then the simple message

CO CLEAR

is sufficient.

Other CO versions executing are undisturbed by a CLEAR operation.

Any CO can be CLEARed at any time whether or not a restart is in operation or pending.

Example of aborted compilation:

Assume that a system failure occurred while doing a COBOL compilation. The controlling CO had a mix-number of 6 and the compiler was in the OPTLIST pass. Assume also that, since the failure, an RPG compilation had been initiated.

The message

CO MX

may result in the following information:

```
CO MIX NUMBER                          4
        COMPILER NAME                  RPG
        CURRENT COMPILER PASS          RPGPHASE1
        WORKFILE CHARACTER             D

CO MIX NUMBER                          6
        COMPILER NAME                  COBOL
        CURRENT COMPILER PASS          OPTLIST
        WORKFFLE CHARACTER             C
                (THIS CO IS IN RESTART MODE)
```

Then to restart the COBOL compilation, enter

CO RESTART 6

to cause the compilation to start at the beginning of OPTLIST. If the COBOL compilation is not required to be restarted, then enter

CO CLEAR 6

Note that in both cases the RPG compilation is unaffected.

ZIP FAILURES

If a zip failure occurs, or a particular compiler pass is DS'ed or DP'ed, CO displays one of the messages in table 6-1 indicating the reason for the failure, then takes one of the following two actions:

If no Restart is in operation or pending.

One of the following messages is displayed:

'CO SHOULD BE RESTARTED OR CLEARED".

'CO MIX-NUMBER n MUST BE RESTARTED OR CLEARED".

The CO utility is forced into "Restart mode" which prevents any new COs or any other RESTARTS being performed until the CO in question has been either restarted and completed, or cleared.

If a Restart is in operation or pending.

One of the following messages is displayed:

"CO SHOULD BE RESTARTED OR CLEARED"

"CO MIX-NUMBER SHOULD BE RESTARTED WHEN EXISTING RESTART COMPLETE".

The CO utility is not forced into "Restart mode" in this case. It is required to RESTART the job as soon as possible after the existing restart is complete.

Table 6–1. Zip Failure Messages

```
ZIP FAILURE DUE TO PROGRAM FILE NOT FOUND FOR program-name
ZIP FAILURE DUE TO INTERPRETER FILE NOT FOUND FOR program-name
ZIP FAILURE DUE TO NO MEMORY FOR program-name
ZIP FAILURE DUE TO NO USER DISK FOR program-name
ZIP FAILURE DUE TO MIX FULL FOR program-name
ZIP FAILURE DUE TO USER COUNT ERROR program-name
ZIP FAILURE DUE TO DUPLICATE PACK program-name
ZIP FAILURE DUE TO INVALID LOAD REQUEST program-name
ZIP FAILURE DUE TO MCS ALREADY PRESENT program-name
ZIP FAILURE DUE TO DISK ERROR program-name
ZIP FAILURE DUE TO CODE FILE ERROR program-name
ZIP FAILURE DUE TO ILLEGAL DATA REQUEST program-name
ZIP FAILURE DUE TO REASON UNKNOWN program-name
COMPILER PASS DS'ed program-name
COMPILER PASS DP'ed program-name
```

RESERVED WORDS

All keywords used in initiating messages to CO are reserved words; that is, they cannot be used for file names or other user-defined parts of the initiating message. A complete list of the reserved words is given in table 6-2.

Table 6–2. CO Reserved Words

MPL	SOURCEOUT
COBOL	WORKFILE
RPG	PRINTOUT
RPGXREF	NAME
OPTLIST	MFID
SAVE	FID
GO	DEVICE
SYNTAX	FILE.SIZE
SY	FILESIZE
LIBRARY	RECORD
LI	BLOCK
PRINT	RECORDS.BLOCK
MESSAGE	MX
FILE	RESTART
FI	CLEAR
PATCHFILE	ALL
SOURCEIN	

ERROR MESSAGES

The printout file for CO may contain error messages from the CO utility itself. These are listed in table 6-3, and are largely self-explanatory.

If errors have been detected in CO's initiating message, a message will be displayed to indicate whether or not the compiler will be zipped. If no errors have been flagged, it is assumed that the compiler will be zipped, and no message is displayed.

Table 6–3. Error Messages from CO

ERROR NO	ERROR MESSAGE
1	First word in Initiating Message is not a valic disk or file name.
2	Invalid object program name or compiler rame specified.
3	Warning - Compiler Pack Id. truncated to 7 characters.
4	No compiler specified.
5	Warning - Program Pack Id. truncated to 7 characters.
6	Warning - Program Id. truncated to 12 characters.
7	Reserved Worc not found when expected.
8	FI/FILE not followed by one of PATCHFILE, SOURCEIN, SOURCEOUT, CONTROLFILE, ICFILE, WORKFILE, PRINTOUT.
9	Reserved Worc used out of context.
10	Warning - Compiler Option already mocified - previous mod. ignored.
11	Warning - Print Option specified more thar orce.
12	Record size * Blocking factor > 65535, Block size set equal to Record size.
13	MESSAGE not used by COBOL - Message Text ignored.
14	Warning - Message Text already processed - this Message Text ignored.
15	Modification not valid for COBOLXREF - ignored.
16	$SEGMENT file-name parameter not a valid file name.
17	CO.MASTER no longer on disk - job terminted.
18	Warning - $SEGMENT pack-id truncated to 7 characters.
19	Warning - $SEGMENT program-name truncated to 12 characters.
20	$SEGSIZE parameter not numeric.
21	Modification not valid for SOURCEIN - modification ignored.
22	Warning - MESSAGE contains no valic dollar cards.
23	Warning - $card not valid for this compiler.
24	Warning - Invalid $card in Message Text.
25	Warning - PATCHFILE modifications already dore - these mods ignored.
26	Warning - PATCHFILE modifications incomplete.
27	Modification not valid for PATCHFILE - modification ignored.
28	Warning - No PATCHFILE modifications.
29	MFID/FID modification already done for this file - mod ignored.
30	Pack name/File name in NAME specification invalid.
31	Pack name in NAME specification truncated to 7 characters.

32 File name in NAME specification truncatec to 12 characters.
33 Pack name in MFID clause invalid.
34 Pack name in MFID clause truncated to 7 characters.
35 File name in FID clause invalid.
36 File name in FID clause truncated to 12 characters.
37 RECORD size of zero specifiec - mod ignored - cefault used.
38 DEVICE modification already done for this file - this modification ignored.
39 Specified Device Type invalid - default device type used.
40 FILESIZE already modified for this file - this mod ignored.
41 FILESIZE parameter not numeric.
42 BLOCK size already modified - this mod. ignored.
43 RECORDS.BLOCK/BLOCK parameter not numeric.
44 RECORD parameter not numeric.
45 SOURCEIN modification already done - these mods ignored.
46 Warning - SOURCEIN modification incomplete.
47 No SOURCEIN modifications.
48 SOURCEOUT modifications already done - these mods ignored.
49 Warning - SOURCEOUT modifications incomplete.
50 Warning - No SOURCEOUT modifications.
51 Specified RESTART mix-number not numeric.
52 Specified RESTART mix-number out of range.
53 Cannot initiate another RESTART - previous RESTART incomplete.
54 No CO with specified mix-number available for RESTART.
55 Null Initiating Message - no file CO.STARTUP - job terrinated.
56 Nested Macro Call found.
57 Invalid file name in Macro Call.
58 Pack name in Macro Call truncated to 7 characters.
59 File name in Macro Call truncatec to 12 characters.
60 Record size for card device > 96, RECORD and BLOCK size set to 96.
61 File CO.STARTUP empty - No Initiating Message - job terminated.
62 Cannot run CO on this system, SYSTEM STATUS Communicate not supported.
63 CO.MASTER not found - job terminated.
64 Compiler not zipped - errors in Initiating Message.
65 Specified CLEAR mix-number not numeric.
66 Specified CLEAR mix-number out of range.
67 No CO's to RESTART/CLEAR - job terminatec.
68 No CO with specified mix-number available for Clearing.
69 SYSTEM STATUS Communicate failure - job terminated.
70 WORKFILE modification already done - this moc ignored.
71 WORKFILE not followed by MFID - mod ignored.
72 Specified RESTART line-number for listing not rumeric.
73 Specified Restart Line-nmber > 65535 - Line-number set to 0.
74 RESTART mix-number not specified.
75 No parameter specified for CLEAR function.
76 Line-number parameter only valid for COBOL - ignored.
77 Illegal character encountered in Initiating Message.
78 Cannot continue compilation - Compilation Block

initialized.

79 Non-input Device specified for PATCHFILE - mod. ignored.

80 Non-input Device specifid for SOURCEIN - mod. ignored.

81 Macro file not found - <file-name>

82 Non-output Device specified for SOURCEOUT - mod ignored.

83 BLOCK size for SOURCEOUT is not a multiple of the RECORD size.

84 Specified BLOCK size for Tape Device too large - default value used.

85 Specified RECORD size for Tape Device too large - default used.

86 80 col card device RECORD size > 80, RECORD and BLOCK size set to 80.

87 96 col card device RECORD size > 96, RECORD and BLOCK size set to 96.

88 Warning - PRINTOUT modification already done - this mod ignored.

89 Device specified for PRINTOUT not Printer Type - ignored.

90 PRINTOUT not followed by DEVICE or NO.BUFFERS - ignored.

91 BLOCK size or Blocking-factor of zero specified - default used.

92 SOURCEOUT FILESIZE specified as zero - mod. ignored - default used.

93 Specified PATCHFILE FILESIZE > 65535 - default value used.

94 Specified SOURCEOUT filesize > 65535 - default value used.

95 Specified RECORD size > 65535 - default value used.

96 Specified BLOCK size or Blocking-Factor > 65535 - default used.

97 When compiling MPL on Cumbernauld Support System, level number required.

98 If 80 Col. card output device used, record truncation will occur.

99 RECORD size already modified - this mod. ignored.

100 Macro file present but empty - <file-name>

101 NO.BUFFERS already modified - this mod. ignored.

102 NO.BUFFERS parameter not numeric.

103 Value specified for NO.BUFFERS not in range 1 to 16 - default used.

104 Warning - NO PRINTOUT modifications.

105 Warning - PRINTOUT modifications incomplete.

106 Modification not valid for PRINTOUT - ignored.

107 ICFILE modification not valid for this compiler - ignored.

108 CONTROLFILE modification not valid for this compiler - ignored.

109 CONTROLFILE modification(s) already done - these mods ignored.

110 No CONTROLFILE modifications.

111 CONTROLFILE modifications incomplete.

112 Modification not valid for CONTROLFILE - ignored.

113 ICFILE modification(s) already done - these mods. ignored.

114 No ICFILE modifications.

115 ICFILE modifications incomplete.

116 Modification not valid for ICFILE - ignored.

```
117  Non-input device specified for CONTROLFILE - ignored.
118  Non-input device specified for ICFILE - ignored.
119  PATCHFILE modification not valid for this compiler - ignored.
120  SOURCEIN modification not valid for this compiler - ignored.
121  SOURCEOUT modification not valid for this compiler - ignored.
```

RESTARTING EXECUTING CO VERSIONS

Note that CO does not prevent restarting or clearing a currently-executing compilation. However, when the original version of the compiler pass, that was executing when the RESTART was done, terminates and returns to CO, CO will recognize that the block of information in the CO.MASTER file has been overwritten. The message

"CANNOT CONTINUE COMPILATION-COMPILATION
BLOCK INITIALIZED"

is displayed and the job is terminated. This may result in problems such as the occurrence of duplicate files. If an executing compilation is CLEARed, there is an added complication that any workfiles for that compilation are removed.

Restarting or clearing any currently-executing compilation is therefore not advised.

SECTION 7
NUMBERED SYSTEM SOFTWARE OUTPUT MESSAGES

INTRODUCTION

The CMS software maintains a table of output messages used by the MCP, the interpreters, and other items like the SORT intrinsic. Each of these messages is given a number called the "event number". The event number is always displayed, enclosed in brackets, as part of the message.

This section lists each message in event number order.

EVENTS # 1-9

Software Information

These are information indicating error conditions. If any action is required, other message(s) will follow immediately.

Message format:

mix number/program-name <EVENT#> file-name peripheral function message ERROR WHILE IN verb status.

The peripheral is given by the mnemonic for example, DK for disk. The function is INPUT or OUTPUT. The message is given in the table below. The verb is OPEN, CLOSE, READ, WRITE, etc. The status gives additional information such as the disk address for disk failures.

Examples:

12/PD <46> MYDISK/SYSMEM DM OUTPUT TIMEOUT ERROR
... WHILE IN OPEN 0033

10/MYPROG <2> FR203D DM READ PARITY ERROR
... WHILE IN READ 14A5

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
1	TAPE FORMAT	Ending label on labelled tape is missing or invalid.	None. Program has indicated it can handle the error itself.
2	PARITY <STATUS>	Hard parity error found on this device.	None. Program has indicated it can handle the error itself. <STATUS> shows disk sector address for disk errors.
3	TIMEOUT <STATUS>	For tape device, no data was found for quite a distance on tape (exact length depends upon unit); For disk, cylinder specified could not be used.	None. Program has indicated it can handle the error itself. <STATUS> shows disk cylinder address for disk errors.
4	ADDRESS <STATUS>	Sector could not be found on disk (cylinder was found). Either software has indicated an invalid sector or sector address on disk is corrupt.	None. Program has indicated it can handle the error itself. <STATUS> shows disk sector address for disk errors.

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
5	REWIND	Problem encountered on tape rewind.	Retry the program.
6	DESCRIPTOR	Identifies I/O error that is not PARITY, TIMEOUT, ADDRESS or REWIND.	None. Program has indicated it can handle the error itself.
7	NON-FILE FULL ON MFID	It is possible for all entries in available table (non-file directory) 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. Disk is checkerboarded.	Run SQ utility and then retry the program.
8	DEVICE LOCKED	A non disk device is closed with lock.	None.
9	DEVICE NOT PURGED	An attempt is made to close purge a write disable magnetic tape drive.	None.

EVENTS # 10-19

Software Suspensions

When a program that is running encounters a condition that prevents it from continuing, MCP will suspend the program and inform the operator as to the reason for the suspension. When the condition is cleared, MCP will normally allow program to continue running. If program does not continue automatically then the operator should issue a "GO" command (see "GO" intrinsic) to the program.

Message formats:

mix number/program-name <EVENT #> WAITING file-name device message

Examples:

10/LIST <17> WAITING UNLAB LISTPRT AP NO FILE
10/COPY <17> WAITING TL REEL 001 AT NO FILE
02/NGD03 <12> WAITING FILE 255 NO USER DISK
05/MPL.SEG <10> WAITING CARDFL DF NO FILE

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
10	NO FILE	Disk or tape file this program needs to use cannot be found.	Check for correct disk or tape; Supply program with backup copy of file requested;
			Program will continue when file is supplied.
11	DUPLICATE FILE	While attempting to place a certain file on disk, program has discovered a file with the same name already exists on disk. Program halts, as 2 files with same name cannot reside on one disk.	Normally remove with RM the existing file from disk. Program will continue. If in doubt refer to program instructions.
12	NO USER DISK (cont'd)	There is no more available space on disk; or space available is insufficient to hold file this program is attempting to write; or disk is "checkerboarded".	With KA, analyze amount of available space remaining. If disk is filled remove with RM any unnecessary files and program will continue; or

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
	NO USER DISK (cont'd)		If disk is filled and a dual-pack file is desired, assign different disk to this program (see AD intrinsic); or if disk is checker-boarded use SQ utility to consolidate disk space, then re-execute program that encountered suspension.
13	DIRECTORY FULL	When the disk was initialized the disk directory was constructed to contain a fixed number of file names. The directory has now reached its capacity.	Remove with RM any unnecessary files and program will continue; or DS the suspended program. Replace disk with another disk having sufficient directory space, and re-execute the program.
14	DEVICE NOT AVAILABLE	Output device that program needs in order to continue processing is either unavailable or not ready.	RY required device; or assign program to alternate device (see AD intrinsic).
15	FORMS REQUIRED	Program is waiting for operator to insert correct forms into the outout device before it will continue processing.	Insert correct forms into output device. Then use "AD" intrinsic to assign device to program (see AD intrinsic).
16	GENERATION NUMBER MISMATCH	While opening an old indexed file, a discrepancy has been found between the generation number fields of the keyfile and the data file. The operating system will cancel the suspension when the discrepancy no longer exists.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
17	NO FILE	Program needs a device that is either unavailable, busy, or not ready.	RY required device; or allow program currently using the device to go to End of Job and suspended program will then automatically be able to use device;
18	DUPLICATE FILE	2 or more disks (or tapes) having the same names have been found on-line. Only one disk (or tape) of a given name may be on-line at a time.	Check for correct disk or tape. Replace (or relabel) one of the duplicates.
19	FILE IN USE	The specified file cannot be opened because it has already been opened by the maximum number of users allowed for the desired move i.e. input/output.	Wait until the file is not in use.

EVENTS # 20-40

INVALID REQUEST ON CLASS A OR B COMMUNICATE TO MCP

These messages normally indicate program errors. Program in error should be DS'ed or DP'ed (see DS and DP intrinsics), if necessary. Then operator should attempt to run the program again. If the same error is encountered, request technical assistance.

Message format:

mix number/program-name <EVENT#> CANNOT verb text parameter

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
20	(NULL)	Illegal verb.	
21	NOT FIB	Byte 1 of Priority A or B communicate should contain Data Segment Table Index of a File Information Block. Data Segment Table does not contain File Information Block.	
22	FILE LOST	The medium which contained the file has been illegally physically removed when it was still required (e.g. half close).	None
23	ATTRIBUTE MISMATCH	Each class A communicate is checked against the files attributes. Attribute includes such things as device, myuse, access mode etc. This message indicates failure on this check e.g. read on an output file, start on a non-disk file, write on a closed file etc.	
24	BAD SEQUENCE	Priority A error may be noted when OPENED I/O with Sequential Access 1. REWRITE was not immediately preceded by successful READ.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
		2. OVERWRITE was immediately preceded by OPEN or START communicate. 3. OVERWRITE or REWRITE was preceded by conditional READ which failed.	
25	BAD WORK AREA	Work Area Segment specified within File Information Block cannot be used, because it either indicates FIB segment or it is Read Only segment - yet communicate requests data transfer to that segment.	
26	ILLEGAL KEY	Key requested on Priority A communicate for a disk was equal to zero.	
	NON-RESIDENT COMMUNICATE HANDLER		
27	DEVICE NOT ON SYSTEM	There is no device of the required kind on the system.	
29	DUAL PACK MISMATCH	The two parts of a dual pack file have been found to be inconsistent.	
30	ALREADY OPEN	Communicate requested an OPEN on already opened file.	
31	ALREADY CLOSED	Communicate requested a CLOSE of already closed file.	
32	BAD ADVERB	Adverb to OPEN was determined to be illegal for any use of the following: 1. MYUSE equal to "0"	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
		2. MYUSE incompatible with device (for example, I/O for LP). 3. Access mode Random for non-disk device. 4. Access mode not equal to Sequential or Random.	
33	BAD BLOCK OR RECORD SIZE	Record and/or Block Size been determined to be incompatible or illegal for any of 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 not multiple of 180 if old file is opened with specified Buffer length other than zero. 5. DFH Buffer length not multiple of FPB Buffer length if old file is opened with specified Buffer length.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
34	BAD FILE SIZE	Maximum file size exceeds 1048560 or 65535 if single area File.	
35	BAD NUMBER OF BUFFERS	Number of buffers specified exceeds 16.	
36	BAD DEVICE	Device requested not supported by CMS. Device code is illegal.	
37	BAD FILE TYPE	1. An OPEN has been requested on "SYSMEM" file by unprivileged user program. 2. Unprivileged user program requested a CLOSE with Lock or Purge of file with SYSMEM VMFILE or Firmware File Type.	
38	PROTECTION ERROR	Privileged user attempted to CLOSE with Lock or Purge a VMFILE or Firmware file while the file was still in use.	
39	BAD FILE-NAME	OPEN or CLOSE has detected an illegal special character imbedded in file-name.	
40	BAD KEYSIZE	During OPEN of a new index file, MCP has discovered in File Parameter Block (FPB) a key length of zero or one that exceeds 28 bytes.	

EVENTS # 41-49

Fatal Device Errors

These messages indicate fatal hardware errors. The program encountering the error should be DS'ed or DP'ed (see DS and DP intrinsics) if necessary. Then the operator should attempt to run program again. If the same error is encountered the media (disks, tapes, etc) involved should not be used further until field engineering has been notified and media has been checked.

Message format:

mix number/program-name <EVENT#> file-name peripheral function message ERROR WHILE IN verb status

The format of these messages is the same as the format for event numbers 1-9.

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
41	BAD LEVEL NUMBER	The implementation number in a keyfile KFFB is greater than this implementation.	
42	DISK LOCKED	Disk write error occurred at file close time. This could be either while writing last block, or updating Disk File Header or available table.	
43	END OF TAPE	Unexpected end of tape encountered.	
44	TAPE FORMAT	Ending label or labelled tape is missing or invalid.	
45	PARITY <STATUS>	Hard priority error discovered on device. <STATUS> shows disk sector address for disk errors.	
46	TIMEOUT <STATUS>	For tape device, no data was found for quite a distance on tape (exact length depends upon unit); For disk, cylinder specified could not be found. <STATUS> shows disk cylinder address for disk errors.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
47	ADDRESS <STATUS>	Sector could not be found on disk (cylinder was found). Either Software has indicated invalid sector or sector address or disk is corrupt. <STATUS> shows disk sector address for disk errors.	
48	REWIND	Problems encountered on rewinding a tape.	
49	DESCRIPTOR	Identifies I/O error that is not PARITY, ADDRESS, or REWIND.	

EVENTS # 50-69

Load Failures

These messages indicate that the MCP failed to begin the processing of a particular program for some reason. Operator should correct the condition that caused the load failure and then try running the program again.

Message format:

[EVENT#] LOAD FAILURE message

Examples:

[53] LOAD FAILURE NO USER DISK
[51] LOAD FAILURE PROGRAM NOT FOUND

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
50	DISK NOT FOUND	Specified disk is not on-line or ready.	Check for correct disk; RY disk if not ready
51	PROGRAM NOT FOUND	Specified file was not found on disk.	Check input and re-input if necessary; Check for correct disk.
52	FULL MIX	System cannot accept any additional programs of this Priority Class at the moment; or program presently running will not permit any other programs to enter the mix.	Allow one program to come to End of Job (EOJ). Then begin required program again; or DS unwanted program(s), if applicable. Then begin required program again.
53	NO USER DISK	There is not enough room on disk for this program's Virtual Memory file. (VMFILE).	Remove with RM any unnecessary files;
54	INTERPRETER NOT FOUND	For B 90:	
		COBOLINT or BILINTERP were not found on system disk or have become corrupted.	Supply, with COPY, the missing interpreters from backup medium; Replace, with COPY, the corrupted interpreters from backup medium;

(cont'd)

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
	INTERPRETER NOT FOUND (cont'd)	For B 800:	
		MCP may be corrupted; program operator is attempting to run may be corrupted.	Replace, with COPY, corrupted MCP from backup medium; Replace, with COPY, corrupted program with backup medium;
55	USER COUNT ERROR	Too many programs are trying to use a file. Limit is seven.	Wait until one or more programs goes to EOJ. Then try again.
56	CODE FILE ERROR	Data file name was mistaken for program name, and an attempt was made to run the file instead of a program.	Check input and re-enter.
57	INVALID LOAD REQUEST	Typing error. (EX: too many characters in program name).	Check input and re-enter.
58	INSUFFICIENT MEMORY	There is not enough room in memory (to hold this proram's Task Control Block and Program Control Block).	Request technical assistance.
59	MCS ALREADY PRESENT	Only one MCS may be in the mix.	None.
60	DUPLICATE PACK	2 or more disks with the same name are currently on-line, and this system does not permit this.	Re-name one of the disks with RL; check for correct disk.
61	NULL MIX REQUIRED	Specified program may be run only if no other programs are in the mix e.g. SQ	Wait until all programs go to EOJ before running this program.

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
62	ILLEGAL DATA COMM LOAD REQUEST	An NDL task can be loaded only as part of the data comm loading sequence. Any other attempt to load an NDL task is failed.	
63	DISK ERROR	There has been an irrecoverable disk error while attempting to load a program.	
64	UTILITY BUSY	The operator has tried to initiate a super-utility function while another is in the mix.	Wait until the first function ends.
65	INSUFFICIENT REAL STORE	The program run structure cannot be constructed in the amount of memory specified in the real store field of the EX command.	
66	DUAL ALPHABET NOT SUPPORTED	The program requires interpreter and MCP facilities to support dual alphabet or reverse escapement and the current system does not.	

EVENTS # 70-99

System Errors

These messages indicate system errors. The program encountering the error should be DS'ed or DP'ed (see DS and DP intrinsics) if necessary. Then the operator should attempt to run the program again. If the same error is encountered, technical assistance should be requested.

Message format:

mix number/program-name <EVENT#> SLICE# message

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
70	SEGMENT OUT OF RANGE	Segment number exceeds number of segments declared in the program.	
71	SEGMENT SIZE ERROR	Offset and length value exceeds declared size of segment.	
72	STACK OVERFLOW	Amount of Control Stack requested during a run has exceeded declared Stack Size.	Use MODIFY utility to increase control stack size of program, and rerun program.
73	STACK UNDERFLOW	Designated code has attempted to retrieve more information from Control Stack than is present.	
74	INSUFFICIENT REAL STORE	The program has attempted to exceed the memory size specified at load time in the real-store field.	
75	PROGRAM TOO LARGE	The program has attempted to exceed the physical memory size.	
80	BAD KEY POSITION	The offset of key position is more than record size.	

EVENTS # 100-169

Program Errors

These messages indicate program errors. The program in error should be DP'ed (see DP intrinsics) if necessary. The source program, object program, dumpfile produced by DP, and any data files used by this program should be saved. The operator should try to run the program again. If the same error is encountered, request technical assistance, and supply all relevant data saved to the technician.

Message format:

mix number/program-name <EVENT> SOURCE REFERENCING program segment # segment address program counter address message

Examples:

03/MYPROG <121> 0 SEGMENT 0 ADDRESS
... 15 SUBSTRING ERROR

04/PROGB <140> 2639 SEGMENT 18 ADDRESS
... 436 CODE FILE ERROR

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
	ALL INTERPRETERS :		
100	COMMUNICATE ERROR	MCP returned @80@ in Byte 0 of Fetch Message on a communicate.	
101	COMMUNICATE EOF ERROR	MCP returned an End of File indication in Fetch message (@20 20 00@) and the user has not specified any action if EOF occurs.	
102	COMMUNICATE I/O ERROR	MCP returned I/O error indication in Fetch message (@20 30 00@) and the user has not specified any action to take if error occurs.	
103	SEGMENT NUMBER ERROR	Interpreter detected an invalid code on data segment number.	
104	WRITE ERROR	Interpreter has detected attempt to WRITE into a Read-Only Segment or literal. Code file has become corrupt or an error exists in compiler or interpreter.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
105	SEGMENT BOUNDARY VIOLATION	Interpreter, in calculating an address, has discovered that the address of the data or code is out of range. Code file has become corrupt or an error exists in compiler or interpreter.	
INTERPRETER for MPLII			
110	INVALID OP	Code file has become corrupt or error exists in MPLII compiler or interpreter.	
115	DESCRIPTOR ACCESS CODE	Program tried to store the fetch value to a non-character field, or to convert to a non-character field, or to store to a self-relative descriptor, or an error in the SETNAME procedure, or a decimal add error.	
116	SEGMENT SIZE ERROR	Segment length is too long or is set while segment is in core.	
117	ADDRESS ERROR	SETNAME extent error; identifier has become out of scope	
118	MESSAGE REFERENCE ERROR	Message reference field not divisible by 4, or illegal access to message reference space.	
119	STRING STORAGE ERROR	Illegal destination in store string instruction.	
120	REMAP ERROR	Program tried to remap a bit descriptor.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
121	SUBSTRING ERROR	Attempt made to substring into non-character area.	
122	INDEX ERROR	Program tried to index a self-relative descriptor, or to bit index a self-relative descriptor.	
123	EXIT ERROR	Local data returned from a function.	
124	CPA ERROR	Error in communicate parameter area: for example, message referenced expected, or size of character field at least 3 bytes expected.	
125	DIVIDE ERROR	Divide by zero attempted.	
126	ZIP ERROR	Error returned after ZIP (not used on B80)	
127	BIT DESCRIPTOR ERROR	Bit field overlaps more than one byte boundary.	
128	FPB ERROR	Error in file parameter block encountered - subfield of a non-F.P.B. segment requested.	
129	CONTROL STACK ERROR	Control Stack overflow or underflow.	
130	DATA STACK ERROR	Illegal descriptor encountered.	
131	DECLARATION MODE ERROR	Size of character field greater than 255.	
132	DATA STRUCTURE ERROR	Insufficient room for structure nesting or size of character field greater than 255, or insufficient room	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
	DATA STRUCTURE ERROR (cont'd)	for array bound evaluation, or non-character descriptor encountered when character descriptor expected.	
	INTERPRETER for COBOL and RPG		
140	INALID S-OP CODE	Invalid S-Op-Code. Code file is corrupt or error exists in COBOL/RPG compiler or interpreter.	Same as event 143
141	INVALID COPX - GREATER THAN SIZE OF COP TABLE	Invalid COPX - greater than size of COP table. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143
142	ALPHANUMERIC FIELD TYPE NOT 8-BIT UNSIGNED	Alphanumeric field type not 8-bit unsigned. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143
143	INVALID EDIT MICRO OPERATOR	Invalid EDIT micro Operator. Code file corrupt or error in COBOL/RPG compiler COBOL/RPG compiler or interpreter.	Re-run program from backup copy. If still in error, request technical assistance.
144	INLINE EDIT MASK NOT CORRECTLY TERMINATED	Inline EDIT MASK not correctly terminated. Coce file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143
145	EXAMINE SOURCE FIELD ERROR	EXAMINE source field error. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143
146	EXAMINE PARAMETER FIELD NOT 8-BIT UNSIGNED, ONE CHARACTER	EXAMINE parameter field not 8-bit unsigned character. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
147	EXAMINE CONTROL BYTE ERROR	EXAMINE control byte error. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same a event 143
148	COMPARE FOR CLASS-CLASS AND FIELD TYPE INCOMPATIBLE	COMPARE for CLASS-CLASS and FIELD type incompatible. Code file corrupt or error in COBOL/RPG compiler or interpreter.	Same as event 143
149	SUBSCRIPTED OR INDEXED SUBSCRIPT OR INDEX	Code file corrupt or error in COBOL/RPG compiler or interpreter.	
150	INDEXED/ SUBSCRIPTED VARIABLE IS INDEXED/ SUBSCRIPTED BY MORE THAN 3 VARIABLES	The array has more than three subscripts or indexes.	
151	FETCH COMMUNICATE RESPONSE FIELD NOT OF LENGTH 3 BYTES		
152	INVALID EXAMINE SPECIFICATION		
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 value was used). If this did not result from a programming error, the Perform Stack should be increased.	The size of the perform stack may be increased by the MODIFY utility, using the CONTROL.STACK option to change the PPB (see section 4).
161	NON-POSITIVE OVERFLOW	Subscripts must be positive.	

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
162	ARRAY BOUND VIOLATION	Subscripts outside upper bound of OCCURS Clause.	
164	TRANSLATION SOURCE ERROR		
166	INVALID SIGN CODE		
167	I/O ERROR	Invalid read/write to a file.	
168	SORT OR MERGE ERROR	An error has been encountered in SORT or MERGE.	
169	ZIP FAILURE	Zip to another task has failed.	

EVENTS # 170-199

SORT Exception Events

Any of these output messages may be displayed while SORT and SORTINTRINS are running. For reporting SORT errors: all SORT errors should be accompanied by either the Sort Spec or the COBOL program that requested SORTINTRINS, as well as the input file(s). Since SORTINTRINS is a machine-dependent program, the method of getting a dump may vary. To get a program dump on the B 80: rerun the program with "GT" on. This will cause SORTINTRINS to dump its run structure on the console printer.

Message format:

mix number/program-name <EVENT#> message

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
170	DUPLICATE RECORD record number	Only for Keyfile creation. Another record in file has same key as this record.	If duplicates are desired, specify "DUPLICATES" in input.
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.	None.
172	RECORDS LOST OR GAINED BY SORT-MERGE	Probably indicates an error in SORTINTRINS.	See introductory paragraph concerning errors.
173	number DUPLICATE RECORDS	Normal message tells the opertor the total number of records that have duplicates. (See Event 170).	None.
174	number RECORDS CONTAINING INVALID INDEX KEYS	Normal message tells operator total number of records with invalid index keys. (See Event 171 for further information).	None.
175	number RECORDS DELETED	Records with hex FF in every byte position will be deleted by SORT. Informs operator how many records were deleted.	None.

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
176	number RECORDS MERGED	Normal message for MERGE only. Tells operator total number of records merged from all files.	None.
177	number FILES MERGED	Normal message for MERGE only. Tells total number of files merged. Should be same as number of files requested in the Sort Specs.	None.
178	SORT-MERGE OUTPUT FILE NOT CREATED	SORTINTRINS was DS'ed; may indicate corrupt SORT or SORTINTRINS programs.	See introductory paragraph concerning SORT errors.
179	SORT-MERGE ABNORMAL EOJ	Early termination due to errors.	
180	SORT-MERGE SOFTWARE ERROR	Error in SORTINTRINS	See introductory paragraph concerning SORT errors.
181	number RECORDS REFERENCED BY KEYFILE-TAGFILE	Only for keyfile-tagfile creation. Tells number of entries of keyfile/tagfile.	None.
182	NO INITIATING MESSAGE	SORT INTRINSIC requires a properly coded Initiating Message. This should be properly formatted by SORT or Sorts within programming languages such as COBOL. Probably indicates an attempt to execute SORT INTRINS directly.	None.
183	number RECORDS SORTED	Normal message tells number of records sorted by the SORT utility. SORT successful.	None.

EVENT	MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
184	FILE ERROR <number> ON SORT-MERGE FILE file-name	The <number> means: 1. EOF on output File 2. PARITY on Input File 3. EOF on Sort Work-file 4. Bad Disk Address 5. SORT workfile Error 6. Input File Error 7. Output File Error Except for 2, this probably indicates error in SORTINTRINS.	See introductory paragraph concerning SORT errors.
185	UNORDERED MERGE INPUT MERGE FILE file-name	Files to be merged were found not to be increasing/decreasing key value. Either the file is incorrect or the key position has been incorrectly specified.	SORT files for order. Then retry.
186	TOO MANY RECORDS FOR SORT-MERGE	Machine-dependent limitation.	See introductory paragraph concerning SORT errors.
187	DUPLICATE RECORDS - KEYFILE NOT BUILT	Was not specified and duplicates exist. The keyfile will not be built and this message will be displayed.	Specify DUPLICATES in input.
188	INIT MESSAGE INVALID	Initiating message to the SORTINTRINS is not in proper format. This could possibly be caused by a fault in the program that zipped the SORTINTRINS.	See introductory paragraph concerning SORT errors.
189	SORT-MERGE INITIATED FROM mix number/program-name	Informs the operator which program used the SORT intrinsic.	None.

ROBERT L. OLSEN
TERRITORY MANAGER
7555 BEACH BLVD., SUITE 116
JACKSONVILLE, FLORIDA 32216
721-1660

SECTION 8
B 90 DEPENDENT SYSTEM SOFTWARE

INTRODUCTION

This section covers those items in the CMS software which are operationally different on the B 90 series from other CMS implementations. These differences are mainly a result of the different hardware features involved. The software covered includes:

- Powering the B 90 on and off.
- The B 90 CMS disk "bootstrap" feature.
- Stand-slone utilities, operating without MCP control.
- Loading the MCP.
- Particular features of the B 90 MCP.
- Taking memory (system) dumps
- Utilities released only for B 90 systems.
- B 90 system errors and symptoms.

POWER ON

Ensure no disks or cassettes are in the system (failure to do this may result in subsequent media corruption).

Turn the system power on. It is then under the control of ROM which performs a mini-test of critical machine components to verify it is capable of starting. The successful completion of this test is verified by the PK lights lighting and then turning off sequentially, with PK1 and PK2 remaining lit.

PK1 permits the loading of a cassette into the system. Some examples of cassettes to load would be (1) AE 500 firmware to cause the B 90 to perform as an AE 500, or (2) ACSYS SL5 emulator firmware cassettes to cause the B 90 to process Series L cassette programs on disk. B 90 cassette loads are not used in CMS.

PK2 permits the loading of information from a disk into the system. Some examples of disks to load would be (1) ACSYS (SL5 emulator) firmware disks, or (2) CMS disks to cause the B 90 to load CMS firmware.

Load the CMS disk in any available disk drive.

A mini-disk is considered loaded immediately the drive unit door is closed, and the blue indicator is lit (disk properly inserted and up to speed). For MCP control the disk must be write-enabled (red indicator lit).

For cartridge disk, wait about 20 seconds for the cartridge to come up to speed (you hear a click as the heads access the disk). Ideally they should be run initially for a few minutes before use to achieve correct running temperature. For MCP control the disk must be write-enabled.

For fixed disk, wait until the 'READY' half of the "BUSY/READY" button is on.

Depress PK2 to enter CMS Bootstrap Mode (see below). The various states, including Initial State and Bootstrap Mode, are shown in figure 8–1.

Possible errors in power-on sequence are given below, System Load Errors.

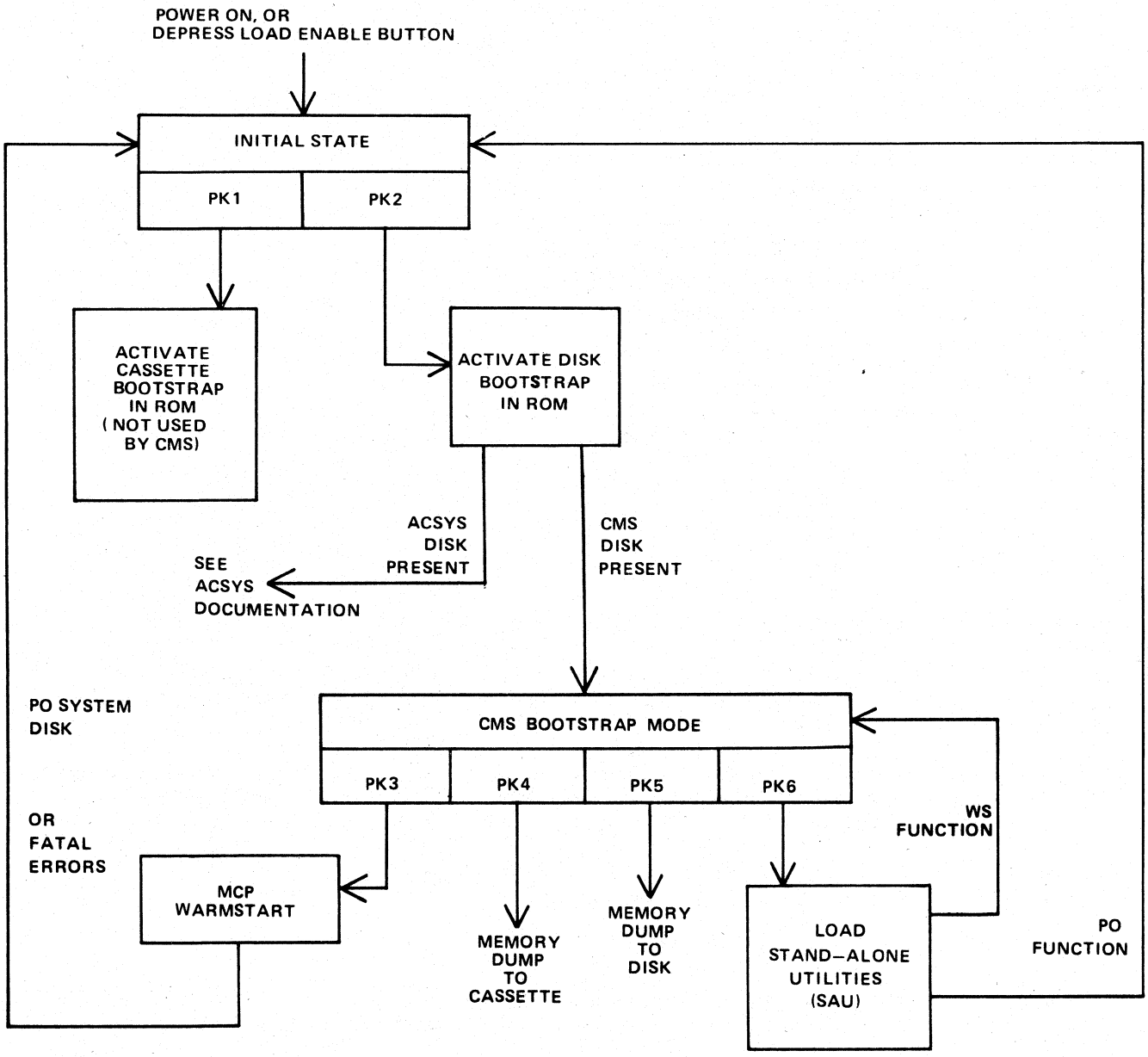


Figure 8–1. B 90 Coldstart and Warmstart

CMS BOOTSTRAP MODE

From the Initial State, depression of PK2 initiates a ROM load routine which searches through all present disks until it finds the bootstrap code. Refer to ROM Scanning Algorithm for details of the search algorithm.

Successful load of CMS bootstrap: PK3 through PK6 will be lit, to provide the following facilities:

PK3 – warmstart MCPX

PK4 – dump contents of memory to cassette.

PK5 – dump contents of memory to disk.

PK6 – enter stand-alone utilities (SAU).

A Note on Forcing System Initialization

When the system hangs (that is, it is not performing any functions or responding to any input from an operator, but has not returned to the initial state), it is necessary to force the system to initialize.

This is done by depressing the Load Enable button in the main cabinet. Never switch the cabinet off, or unload disks or cassettes in use, as this can cause media corruption of various kinds.

If the correct procedure is followed, then although disk or cassette files may be only partially created or updated, the system when recovered should be still able to access the media.

STAND ALONE UTILITIES

The SAU process has the necessary functions to prepare a disk for use on the B 90. This process is used to initially transfer system software (MCP, interpreters, compilers and utilities) to a new disk at a new installation, or to install a new level of MCP at an existing installation. It is also used to condition new disks (IN Initialize) or recondition (RF reformat) existing disks for copying on the B 90.

The SAU functions are brought into memory from disk and operate independent of the MCP.

Loading Stand-Alone Utilities

The system must be in the CMS Bootstrap state. If the system is in the initial state, depress PK2 to get to CMS Bootstrap State (PKs 3 through 6 lit).

Depress PK6. System will search for a disk file called SAU. For details of the search algorithm used, refer to Bootstrap Scanning Algorithm.

For failures in search, see System Load Errors.

A printed message verifies that the SAU has been loaded into memory from disk.

STAND-ALONE UTILITY
VERSION n.nn.nn
REQUEST "HELP" FOR FUNCTION SUMMARY
FUNCTION

Functions Available

initialize a disk: IN
reformat a disk to the initial state: RF disk-name
load files to disk from cassette: LD disk-name FROM cassette-name
copy files from disk to disk: COPY disk-name/file-name TO disk-name/file-name
remove disk files: RM disk-name/file-name
list the disk files and their sizes: LS disk-name/file-name
relabel a disk: RL disk-name
list status of drives: OL
warmstart the MCP: WS
initialize a disk for MTR use: FE (field engineers' use)
terminate stand-alone utility execution: PO
change file-name on disk: CH disk-name/file-name TO disk-name/file-name
clean BSM disk drive read/write heads: CLEAN
compare files on disks: COMPARE disk-name/file-name WITH disk-name/file-name
print disk directories: PDX
duplicate BSMII: DISCOPY drive TO drive
initialize a disk on a console-less system: SAU.PARAM (refer to the utility SAU.INIT)

When the Stand-Alone Utilities are running, only the ON light is lit. Each of the utilities is initiated by a command on the Alpha Keyboard. Only one utility may run at a time and keyboard input is valid only when no utility is running. Keyboard input entered while a utility is running will be lost except for the first four characters. The reset key will clear all keyboard input since the last OCK key, and any OCK key will terminate keyboard input.

Common SAU Output Messages

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FUNCTION ABORTED	Hardware error.	If this message is preceded by another message then follow that message, else notify field engineer.
Dxy DEVICE ERROR DEVICE INOPERABLE	Specified drive was accidentally made Not Ready; or drive unit malfunction.	Ready drive; or notify field engineer.
Dxy DEVICE ERROR OFF CYLINDER	Drive unit malfunction.	Notify field engineer.
Dxy DEVICE ERROR WRITE INHIBITED	Disk in specified drive has its write lockout indicator set.	Ensure that the Write Lockout Hole (BSMD) is covered, or for cartridge - the Write Lockout Plug is flush with surface of cartridge.
Dxy DEVICE ERROR SEEK TIMEOUT	Drive unit malfunction	Notify field engineer.
Dxy DEVICE ERROR CONTROLLER PROBLEM	Drive unit malfunction	Notify field engineer.
FUNCTION	Prompt to request Operator to enter next desired function.	Enter next function, or "HELP" to display list of functions.

Note:

x identifies type of disk

(M = BSMD or BSMDII, F = FIXED, K = CARTRIDGE)

y identifies unit

(A, B, C, etc.)

Disk I/O Errors during SAU

These errors are identified by the messages:

O/P ERROR, I/P ERROR, <unit> DIRECTORY I/O ERROR, <unit> DEVICE ERROR

which may be encountered while running the Stand Alone Utilities.

Disk I/O errors indicate a failure to read from (I/P error) or write (O/P error) disk. Such problems should not be allowed to persist on a disk which is to be used to store important information, especially where the

disk is to be used as a systems disk. Therefore an explanation or fix is required before the drive and disk can be considered acceptable for live use. Even if the Stand Alone Utility continues to run satisfactorily, there may be some form of disk corruption. After any of these errors, the media involved should be checked for any corruption which might cause future system problems (for example, the CHECK.DISK, KA, or DA utilities under MCP control).

A Note On Dual Pack Files

A "dual pack file" is a file which resides on two separate disks or logically identified disks (for example, DFA and DFB).

A dual pack file consists of:

A disk file header on each of the two disks.

At least one and at most sixteen file areas, each of which may be allocated on either disk.

Both parts have the same file name.

Under MCP control, the file may be opened only if both parts are present.

Each file header contains a reference (the pack-id) to the other disk.

Therefore if for any reason one part of the dual pack file is lost, or if the pack-id of one of the disk volumes is changed, the file will be inaccessible under MCP control.

Therefore caution must be exercised when using SAU to initialize, reformat, or relabel any disk containing part of a dual pack file. Dual pack files may be located with the LS function.

In addition to the file-name and area occupied, the LS function will give, for each dual pack file, and overflow pack-id and the total overflow area.

If one part of a dual pack file is lost for any reason, the SAU RM function may be used to remove the remaining part.

The terms "master file" and "overflow file" are sometimes used to distinguish the two parts of a dual pack file, however it should be borne in mind that either part of the file may be regarded as the master file. In this section the term "master" file is used to indicate the part of the file mentioned in a COPY or RM command.

It is necessary that both "master" and "overflow" files reside on disks with the same allocation unit.

CH (Change disk file name)

This function allows the operator to change the name of disk files.

Format:

CH disk-name/ file-name or group-name TO disk-name/ file-name or group-name

Examples:

To change a file AR030 on a disk ARDISK to PR030X:

CH ARDISK/AR030 TO ARDISK/PR030X

To change a group of files starting with GL on a disk GLDISK to a group starting with AP:

CH GLDISK/GL= TO GLDISK/AP=

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
file-name CHANGED TO file-name	function successful	none
file-name NOT CHANGED ILLEGAL FILE-ID	file-name or group-name contains an illegal character	Check input and re-enter
file-name NOT CHANGED O/P LENGTH OVERFLOW	The resulting file-name is longer than 12 characters (possibly during a group change)	Check the group-name and re-input
file-name NOT CHANGED O/P LENGTH UNDERFLOW	When changing a group name to a smaller group name, the resulting file has no characters (for example, changing F= to = when a file called "F" is present)	Correct the group-name and re-input if necessary; or enter another CH for the file not changed
DUPLICATE FILE file-name	An attempt was made to change a file to the name of an already-existing file	Correct the input and re-enter
file-name NOT FOUND - file-name NOT CHANGED INCOMPLETE DUAL PACK FILE	The file is a dual-pack file and the overflow pack is mounted but the file is not present	Investigate why the other part of the file is missing

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PACK disk-name NOT ON LINE - file-name NOT CHANGED - INCOMPLETE DUAL PACK FILE	The file is a dual pack file but the overflow pack is not present	Make the overflow pack present and re-input
file-name NOT CHANGED DUPLICATE FILE ON OVERFLOW PACK	An attempt was made to change the name of a dual-pack file to the name of a file already present on the overflow pack	Correct the input and re-enter

CLEAN (Clean BSM Drive Read/Write Heads)

The read/write heads of the one megabyte mini drive are cleaned by this function. This cannot be used on any other type of drive. Each head is cleaned in turn by the Burroughs head cleaning diskette.

The procedure the system follows to clean one head is as follows:

Load head onto cleaning surface.

Sequentially access the disk from the outermost track to the innermost.

Sweep heads from the outer to the inner-track and back, ten times.

Unload head.

The current postion of the head on the disk is visually displayed by illuminated PK lights.

Example:

CLEAN

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
DRIVE	Prompt requesting operator to insert cleaning disk into proper drive.	Place cleaning disk into proper drive. Make certain disk is write inhibited (write inhibit hole is open and is positioned at top corner of disk).
LOAD WRITE INHIBITED CLEANING DISK AND TRANSMIT "OK" TO CONTINUE	Cleaning disk was inserted into drive.	Enter "OK" plus OCK1 to start cleaning procedure; or press OCK1 to terminate the function.
HEAD 1 CLEANED. FLIP WRITE INHIBITED CLEANING DISK AND TRANSMIT "OK" TO CONTINUE.	Head 1 has been cleaned.	Turn disk upside down (write inhibit hole to bottom) so that cleaning surface faces other side; then enter "OK" and press OCK1 to begin cleaning functions.
HEAD 0 CLEANED END CLEAN	Both heads have been cleaned. Function ends.	Remove cleaning disk and store.

Any attempt to clean head 0 before head 1 will result in the "LOAD/FLIPDISK" message being prompted.

WARNING

Any attempt to use a write-enabled disk or any disk other than the cleaning disk during execution of this function will result in serious damage to the disk drive.

COMPARE (Compare two disk files)

This function allows the operator to compare files present on disk.

Format:

COMPARE	disk-name/	file-name or group-name	WITH	disk-name/	file-name or group-name

Examples:

To compare a file PR200 on disk PRDISK with another file PR200 on disk USER:

COMPARE PRDISK/PR200 WITH USER/PR200

To compare a group of files starting with IN on the disk INDISK with group of files starting with XY on the disk XYZ:

COMPARE INDISK/IN= WITH XYZ/XY=

Before the files are compared a check is made that the file sizes and number of areas are consistent. If any conflicts are found here, the function will be terminated.

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
file-name NOT FOUND	Self-explanatory	Check input
FILE SIZE CONFLICTS - END COMPARE	One file contains more records than the other	none
CONFICTIONS IN NUMBER OF AREAS - END COMPARE	The number of areas assigned to the two files are different	none
disk-name/file-name COMPARED WITH disk-name/file-name - NO CONFLICTS FOUND	successful file comparison	none
disk-name/file-name COMPARED WITH disk-name/file - CONFLICTS FOUND	The compare was unsuccessful	Check SPO and error logs for any errors during copying of either file; re-copy file(s) from backup disks and re-run the COMPARE function
PACK disk-name NOT ON LINE - file-name NOT COMPARED - INCOMPLETE DUAL PACK FILE	Comparison of dual pack file was required and the oveflow pack was not on line	Make the overflow pack present and re-input COMPARE function

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name NOT FOUND file-name NOT COMPARED INCOMPLETE DUAL PACK FILE	Comparison of dual pack file was required and the overflow pack was present but did not contain the required file	Investigate why the file was not present on the overflow pack
END COMPARE	end-of-job of this S.A.U function	none

COPY (Copy files disk to disk)

This function allows the operator to copy files from one disk to another. Overflow files will not be copied.

Format:

COPY disk-name/ file-name or group-name TO disk-name/ file-name or group-name

Examples:

To copy a single file:
COPY PR1/PR200 TO PRBU/PR200

To copy a group of files:
COPY PR1/PR= TO PRBU/PR=

To copy all files on one disk to another:
COPY PR1/= TO PRBU/=

To copy a file and change its name:
COPY PR1/PR200 TO PRBU/PR200B

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
DUPLICATE FILE NAME -	System detected a file name on disk identical to that which the operator is attempting to copy. No two files with the same name may reside on a disk. Therefore specified file was not copied.	Normally, remove the existing file with RM; reattempt the COPY. Check input for correct specification of file names.
file-name TOO LARGE	Insufficient available space on destination disk for the specified file to be copied.	Use a different disk, or first remove some files using the RM function.
NAME LIST FULL (cont'd)	When this disk was initialized, the disk directory was constructed to contain a fixed number of file names. The directory has now reached its capacity.	The COPY function ends automatically. IF possible, remove with RM any unwanted file(s) to make room in the disk directory. Then reattempt to COPY the file which encountered the error;

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
		Replace the disk with a disk that has sufficient room in the disk directory.
Dxy DIRECTORY I/O ERROR	A read or write error was encountered while the system was attempting to access the directory of the specified disk. The directory structure of the input (source) disk may be corrupted.	1. Change the position of the disks and retry. 2. For a group copy, some files may be recovered using single file copy. If these attempts fail then type of error should be analyzed with DA utility and disk re-initialized before re-use.
O/P DISK IS NOT WRITE PERMIT.	Destination disk (disk to which system is copying is write inhibited.	For cartridges: ascertain that write lockout plug is flush with outer surface of cartridge; For BSMD: ascertain that the write lockout hole is covered; Retry the COPY.
file-name NOT FOUND	Specified file-name is not on disk.	Check input - re-enter if necessary; Check for correct disk.
PACK disk-name NOT ON LINE	Specified disk is not on-line to the computer.	Check input - re-enter if necessary; Check for correct disk.
INVALID REQUEST	Typing error.	Check input and re-enter.
END COPY	Copy successful (EOJ); Specified group name to be copied is not on disk.	None if copy successful; Check input - re-enter if necessary; Check for correct disk.
file-name COPIED	COPY successful	NONE
CANNOT ALLOCATE AREAS FOR - file-name	Insufficient available space on disk to which operator is attempting to copy this file.	Using KA utility, analyze disk available space.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name NOT CHANGED - O/P LENGTH OVERFLOW	The resulting file-name is longer than 12 characters.	Correct the group name and re-input.
file-name NOT CHANGED - O/P LENGTH UNDERFLOW	The resulting file-rame has no characters.	Correct the group name and re-input if necessary.
O/P ERROR - file-name	Error encountered while system was attempting to write specifiec file to disk.	Allow COPY to continue copying any remaining files. Then reattempt copying the file in which the O/P error occurred.
I/P ERROR - file-name	System encountered errors in the file from which it was reading.	Allow COPY to continue. Then use back-up copy of the specified file and re-attempt the COPY; switch the disks to the opposite drives.
file-name NOT COPIED: ILLEGAL FILE-ID	Specifiec file-rame contains an illegal character (that is, / ?), therefore file will not be copied. Legal characters are 0-9, A-Z, . (dot), - (dash).	Check input and re-enter if necessary.
file-name COPIED: ILLEGAL FILE-ID	Disk-to-disk copy is being made and this warning is given that the new file is copied but with an invalid file-name.	

Dual Pack Files

The copy function cannot create a dual pack file, however a complete dual pack file may be copied to a single disk.

The master file will be copied first. COPY will then look for a matching overflow file on the overflow pack. If a matching overflow file is found it will be copied and the function will terminate with the message:

file-name COPIED FROM disk-name-1 AND disk-name-2

If the overflow pack is not on line or the overflow file is not found or an overflow file is found which does not match the master file, the function will print the following message, and wait for operator input:

disk-name-file-name IS AN INCOMPLETE DUAL PACK FILE

A MATCHING OVERFLOW FILE ON disk-name-2 IS NOT PRESENT.

PLEASE TAKE ONE OF THE FOLLOWING ACTIONS:

A. SUPPLY THE CORRECT OVERFLOW PACK AND
TYPE "A" TO TRY AGAIN.
B. TYPE "B" TO SKIP THIS FILE.

At this stage the master file pack may be removed if necessary.

The operator should either power on the correct overflow pack and type "A" followed by OCK1 or type "B" followed by OCK1.

If the "A" option is selected the function will repeat its search for the overflow file as above and either terminate normally or repeat the prompt to the operator.

If option "B" is selected the output file will be purged and the function will terminate with the message:

file-name NOT COPIED, PART OF A DUAL PACK FILE

WARNING

If a level of SAU earlier than 3.00 is used to copy a dual pack file or a family containing a dual pack file, the output disk is liable to be seriously corrupted and to require re-initializing.

DISCOPY (Duplicate a BSMII Disk)

This function allows the operator to copy the contents of a Burroughs Super Minidisk II (BSMDII) to another BSMDII.

Format:

DISCOPY D5X TO D5X

where X = A, B, etc.

All information is copied from one BSMII to another, that is, disk label, bootstrap, directories and data. Thus a complete disk can be duplicated for backup purposes without initialization. The automatic relocation of bad sectors allows the duplicated directories to remain valid since all information on the duplicated disk will have the same logical, if not physical, address as that on the master disk.

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
END DISCOPY	end of function	none
NON BSMII DISK SPECIFIED	Either of the disks specified is not a BSMII disk	This function can only be performed on BSMII disks: use COPY for other kinds of disk, including BSM (1MB) disk
PACK	This is a prompt for the operator to give the seven-character disk-name of the input disk	Enter up to seven characters
drive-mnemonic PURGED	An input error has occurred during the copy. The output disk is purged by overwriting track zero with "null" characters.	Investigate cause of error on the input disk, then re-run the DISCOPY function: the purged output disk may be re-used.
FUNCTION ABORTED - drive-mnemonic NOT PURGED	An I/O error has occurred on the output drive during the copy, and the copy is stopped.	Use the LS function to determine which files are successfully copied to the ouput disk: these may be recovered. Otherwise re-run the DISCOPY function using a different output disk.
drive-mnemonic COPIED TO drive-mnemonic	function successful	none

FE (Initialize MTR Disk)

A virgin or a formatted disk is initialized 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 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 called "CMSBOOTxxxxx" which can be contained by any on-line disk or by any cassette labelled "SYSB90" which has "CMSBOOTxxxxx" as its first file.

The "xxxxx" characters are ignored by the utility; only the seven leading characters are compared when the on-line disks are searched (that is, the file specifications searched for is equivalent to CMSBOOT=).

In the case of fixed disk, the CMS bootstrap will be loaded from CMSBOOTxxxxx. All other disks initialized by FE will extract the MTR bootstrap.

The MTR tracks will not be included in the number of bad sectors, but if the disk is reformatted (see RF), the total number of sectors removed from the available table will be referred to as BAD.

Example:

FE

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
DRIVE	Prompt allowing entry of drive where disk to be initialized is located.	Enter Drive (ex: DKA, DMB, etc)
DATE	Prompt allowing entry of initialization date.	Enter date (format MM/DD/YY)
FILES	Prompt allowing entry of maximum number of files for this disk.	Enter maximum number of files this disk will contain (less than 2805).
SERIAL	Prompt allowing entry of disk serial number.	Enter serial number (6 numerals).
PACK	Prompt allowirg entry of disk rame.	Enter disk-name (up to 7 legal characters) Legal characters include 0-9, A-Z, . (dot), - (dash).
OWNER	Prompt allowing ertry of user-cefinec information.	Enter valid information (up to 14 characters).

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
RESTRICTED	Prompt requiring response of "N" at this time.	Enter "N".
BOOTSTRAP VERSION n.nn.nn USED END FE	FE (EOJ)	None.
NO OF BAD SECTORS number	Disk was successfully initialized. If any bad sectors were detected the number is displayed.	None.
TRACK 0 BAD	Sector(s) in Track 0 are bad and cannot be used. FE will end, as a CMS disk must contain reserved information in Track 0 (ex: warmstart bootstrap).	Select another disk to be initialized with FE.
SYSB90 NOT PRESENT LOAD CMS BOOTSTRAP AND HIT OCK1 TO CONTINUE	System failed to find a bootstrap file from either disk or cassette	Load another drive with a disk containing CMSBOOT or load a cassette with name SYSB90 containing CMSBOOT. Then press OCK1. System will re-attempt search for CMSBOOT.
BAD MTR TRACK	Any MTR track contains a bad sector.	
CYL xxx BAD	In cases where certain cylinders are used for MTR purposes but not removed from the available table discovery of a bad sector will terminate the function.	Select another disk to be initialized by FE.

IN (Initialize a Disk)

The MCP (Master Control Program) requires that any disk to be used on the system have a valid CMS disk label, disk directory, and available area table. In addition, each sector of the disk must be initialized with its address.

The IN function performs this disk initialization. When all disk input parameters have been requested, it will check the recording surface of each disk by writing and reading test patterns to each sector of the disk. Any "bad sectors" (unusable or unreadable) will be removed from the disk's available area table, and the number of bad sectors will be displayed to allow badly worn disks to be discarded.

The function will also write a disk label containing information supplied by the operator to the appropriate prompts, below, and create a disk directory of the appropriate size required for the number of files specified, plus a single, SYSMEM, entry. Sectors 1 through 31 of the disk are loaded from a file "CMSBOOTxxxxx" which may be located on any on-line disk or on any cassette labelled "SYSB90" which has "CMSBOOTxxxxx" as its first file. Before loading can take place, the correct identification string must be found in sector one of this file, namely, B90DISKINITVAOF. The disks are searched in descending order, cartridge disks first, then a further scan of Burroughs Super Mini Disks.

Since all fixed disks must be suitable for MTR purposes, the functions IN and FE are identical in this particular case. The relevent MTR tracks will be checked and/or removed and the CMS bootstrap is written.

Example:

IN

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO OF BAD SECTORS number	Disk was successfully initialized. If any bad sectors were detected the number found is displayed.	None.
BOOTSTRAP VERSION n.nn.nn USE) END IN	IN successful (EOJ)	None.
SYSB90 NOT PRESENT LOAD CMS BOOTSTRAP FILE AND HIT OCK1 TO CONTINUE.	System failed to find bootstrap file on disk or cassette.	Load another drive with a disk containing CMSBOOT or load ato cassete with name SYSB90 containing CMSBOOT as its first file. Then depress cassette, then press OCK1. System will reattempt search for CMSBOOT.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
TRACK 0 BAD	Sector(s) in Track 0 are bad and cannot be used. IN will terminate, as a CMS disk must contain reserved information in track 0 (ex: warmstart bootstrap).	Select another disk to be initialized.
DRIVE	Prompt allowing entry of drive in which disk to be initialized is located.	Enter drive (ex: DKB, DKA, etc.)
DATE	Prompt allowing entry of initialization date.	Enter date (format MM/DD/YY for month, day, year; e.g. 01/23/79)
FILES	Prompt allowing entry of maximum number of files for this disk.	Enter number of files disk will contain (less than 2805).
SERIAL	Prompt allowing entry of disk serial number.	Enter serial number (6 numerals).
PACK	Prompt allowing entry of disk name.	Enter disk-name (up to 7 characters).
OWNER	Prompt allowing entry of user-defined information.	Enter any valid information (up to 14 characters).
RESTRICTED	Prompt requiring response of "N" at this time.	Enter "N".

LD (Load Disk)

This function allows the operator to load all files from a dump tape to a disk. A dump tape is one produced by the DUMP or UNLOAD functions of the utility "LD" (which run under MCP control). Each sector of data written to disk is verified. The SAU LD function cannot create a dual pack file.

Format:

LD disk-name FROM library-tape-name

XD user 0020 0060
then copy are files
KA

Example:

LD ARDISK2 FROM ARTAPE

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name LOADED	Displayed for each file loaded and verified.	None.
LOAD REEL number	Dump tape is a reel from a multi-reel dump and the next reel is required.	Supply next reel.
END LD	LD (EOJ)	None.
NOT DUMP TAPE	Specified tape is not a correctly formatted dump tape. Function ENDS.	Supply correctly formatted dump tape; re-initiate LD.
multi-file-name NOT PRESENT. LOAD CASSETTE AND HIT OCK1 OR HIT ANY KEY THEN OCK1 TO TERMINATE.	Specified tape is not installed and ready; Function ends.	Install and ready tape; re-initiate LD; or press OCK1 to retry search for multi-reel name tape; or press any key plus OCK1 to terminate LD.
UNRECOVERABLE CASSETTE ERROR	Error was encountered while system was attempting to read cassette. Normally caused by accidentally opening cassette drive unit. Function ends.	Ready cassette drive unit; re-initiate LD.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PACK disk-name NOT FOUND	Specified disk is not ready; specified disk is not on-line to computer.	Ready the disk; Check for correct disk.
O/P DISK NOT WRITE PERMIT	Disk to which files are to be loaded is write protected. Function ends.	Ensure that write lockout hole is covered (for BSMD), for cartridge - ensure write lockout plug is flush with surface of disk cartridge; re-initiate LD.
DUPLICATE FILE NAME - file-name	File of specified name already exists on destination disk. A disk may not contain 2 files with the same name. LD continues loading other files, ignoring any duplicates.	Remove (with RM) the existing file from disk, then re-initiate LD.
CANNOT ALLOCATE AREAS FOR - file-name.	No appropriately sized available area on the destination disk for specified file.	If specified file is desired, replace this disk with a disk having more available area. Then re-initiate LD.
O/P ERROR - file-name	Disk write error encountered while system was loading specified file.	Place disk in opposite drive and re-initiate LD.
AREA SIZES TOO SMALL FOR - file-name.	Insufficient available area on disk for specified multi-area file.	If specified file is desired, replace this disk with a disk having more available area. Then re-initiate LD.
NAME LIST FULL	No space remaining in disk directory for another file name. Function ends.	Use another disk, or remove unwanted files with the RM function.

LS (List File Sizes)

This function allows the operator to print name and sizes in sectors of files on a specified disk. For dual pack files, both portions of the files and their sizes are printed. If all file sizes on a disk are printed by

LS <disk-name>/=

then all available areas of that disk are printed at the end of the list.

Format:

LS disk-name/ file-name or group-name

Example:

To list file sizes of all the files on the disk ARDISK2

LS ARDISK2/=

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PACK disk-name NOT ON LINE	Specified disk is not on line to the computer.	Check input - re-enter if necessary; Check for correct disk.
END LS	LS successful (EOJ)	None.
Dxy DIRECTORY I/O ERROR	A read or write error was encountered while the system was attempting to access the directory of the specified disk. The directory structure of the disk may be corrupted.	

OL (Print Status of Drives)

This function allows the operator to print the status of all cassette and disk drives.

Example:

OL

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
END OL	OL successful (EOJ);	None.
disk drive disk-name	Specified disk is resident in specified drive.	None.
disk drive NOT READY	There is no disk in this drive; the disk in the drive has not been set to run; the disk in the drive has not come up to proper speed; system does not recognize disk.	If applicable - set disk to run; or allow disk to attain proper speed; make certain disk has a proper "label".
disk drive NOT CMS - STANDARD	Disk in this drive does not have a valid CMS label.	Disk must be initialized with IN to create valid CMS label.
cassette drive NOT READY	There is no cassette in this drive; cassette has not been loaded into drive properly;	Check for proper loading of cassette in drive.
cassette drive UNLABELLED	Cassette in specified drive does not have a valid label.	Use ^G to create valid cassette label.
cassette drive multi-file-name file-name	Cassette in this drive has the specified name.	None.
disk-drive TEMPORARILY NOT AVAILABLE	The door of the disk drive has been opened.	Close the door.

MTR cassettes do not have labels corresponding to the correct CMS format and thus will appear "UNLABELLED".

PDX (Print Disk Directories)

This function allows the operator to print the disk directories, disk label and any sector in hexadecimal.

Format:

PDX

The function operates in interactive mode. The operator is prompted to supply the input.

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
DRIVE	This is a prompt to the operator	Enter the device mnemonic of the disk to be read; for example, DKA
IS DIRECTORY REQUIRED <Y OR N>	This is a prompt to the operator	Enter "Y" if a complete directory print is required, otherwise enter "N"
SECTOR <4 character hex value or null>	The operator is prompted to provide the hex address of a sector to be displayed. This prompt is repeated until a null response (just an DCK key) is given.	Enter the 4-character hex sector address followed by DCK1, or just DCK1 to terminate the function
COMPLETE <Y OR N>	The operator is prompted to state if printing of all sectors in the disk directory is required or only the first sector of each entry	Enter "Y" or "N" as appropriate
LABEL CORRUPT	The check-string "SL9INTERNL" in sector zero has not been found	none: use backup copy of disk for normal use
END PDX	end of PDX function	enter dsired S.A.U function

PO (Power Off)

This function allows the operator to terminate the execution of the Stand-Alone Utilities.

Example:

PO

The utility displays the message

* * * END STAND ALONE UTILITY * * *

and causes the B 90 to return to the initial state (PK1 and PK2 lit, refer to figure 11-1).

RF (Reformat Disk)

This function provides all the capabilities of IN (initialize), except the disk recording surface test. A CMS label and a CMS disk directory are written to the disk. Any information previously contained on the disk will be lost. The disk label will contain information supplied by the operator's responses to the appropriate prompts, below, and the directory will be of the minimum size required for the number of files specified. The original contents of the disk label will be displayed to record the change and to assist re-input of the same data when required. The bootstrap is written back to track "0" in the same manner as for IN.

Format:

RF disk-name

Example:

RF ARDISK2

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
DATE	Prompt allowing entry of RF date.	Enter date (format MM/DD/YY)
FILES	Prompt allowing entry of maximum number of files disk will contain.	Enter number of files disk will contain (less than 2805).
SERIAL	Prompt allowing entry of disk serial number.	Enter serial number (6 numerals).
PACK	Prompt allowing entry of disk name.	Enter disk-name (up to 7 characters).
OWNER	Prompt allowing entry of user-defined information.	Enter any valid information (up to 14 characters).
RESTRICTED	Prompt, requiring a response of "N" at this time.	Enter "N".
BOOTSTRAP VERSION n.nn.nn USED END RF	RF End of Job (EOJ)	None.
PACK disk-name NOT ON LINE	Specified disk-name is not on-line to computer.	Check input (re-input if necessary); Check for correct disk).

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
Dxy DIRECTORY I/O ERROR	A read or write error was encountered while system was attempting to access directory of this disk. Directory of this disk may be corrupted.	Switch disk to alternate drive and reinitiate RF; or
SYSB 90 NOT PRESENT. LOAD CMS BOOTSTRAP AND HIT OCK1 TO CONTINUE.	System failed to find a bootstrap file from either disk or cassette.	Load another drive with disk containing CMSBOOT, or load a cassette with name SYSB90 containing CMSBOOT as its first file. Then press OCK1. System will re-attempt search for CMSBOOT.

RL (Relabel a disk)

This function allows the operator to change a disk's name without affecting the remaining contents of the disk.

Format:

RL disk-name

Example:

RL AP2

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
END RL	RL successful (EOJ).	None.
PACK	Prompt to request operator to enter the new disk name.	Enter new disk name (up to 7 legal characters).
PACK disk-name NOT ON LINE	Specified disk is not on line to the computer.	Check input - re-enter if necessary; Check for correct disk.
Dxy DIRECTORY I/O ERROR	A read or write error was encountered while the system was attempting to access the directory of the specified disk. The directory structure of the disk may be corrupted.	

Warning:

If RL is used on a disk containing part of a dual pack file, that file will be inaccessible under MCP control.

There are two ways to prevent this from occurring:

Remove the file before relabelling the disk, or

Copy the entire file to a single pack file before relabelling.

There are two ways to resolve the problem if it does occur:

Use Stand-Alone Utility (RM) to remove the file, or

Use Stand-Alone Utility (RL) to relabel the disk to its original name.

RM (Remove Disk Files)

This function allows the operator to remove files from disk.

Format:

```
                  file-name
RM  disk-name/      or
                 group-name
```

Examples:

To remove a single file:

RM PR1/PR200

To remove a group of files:

RM PR1/PR=

To remove all files from disk:

RM PR1/=

Unlike the RM utility that runs under MCP control, the Stand-Alone RM will remove system software (MCPX, COBOLINTX, BILINTERPX, etc.) without a warning message, the same as it removes other programs.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
file-name REMOVED	RM successful.	None.
PACK disk-name NOT ON LINE	Specified disk is not on-line to the computer.	Check input - re-enter if necessary; Check for correct disk.
file-name NOT FOUND	Specified file-name is not on disk.	Check input - re-enter if necessary; Check for correct disk.
END RM	RM successful (EOJ); Specified group name to be removed is not on disk.	None if RM successful. Check input - re-enter if necessary. Check for correct disk.

Dual Pack Files

If a file to be removed has an overflow area or another disk volume:

- if both parts of the file are available, both are removed and the following message is printed:

  file-name REMOVED FROM disk-name-1 AND disk-name-2

- if the overflow file is not found for any reason or if a file is found with the same file-id but which does not match the master file, then the following message is printed:

```
disk-name/file-name IS AN INCOMPLETE DUAL PACK FILE
A MATCHING OVERFLOW FILE ON disk-name-2
IS NOT PRESENT. PLEASE TAKE ONE OF THE
FOLLOWING ACTIONS:
    A) SUPPLY THE CORRECT OVERFLOW PACK
       AND TYPE "A" TO TRY AGAIN
    B) TYPE "B" TO SKIP THIS FILE
    C) TYPE "C" TO REMOVE THE INCOMPLETE FILE
```

If option "A" is selected then either the complete file will be removed or the above prompt will be repeated.

If option "B" is selected then the following message is printed:

```
file-name NOT REMOVED, PART OF DUAL PACK FILE
```

If option "C" is selected then the following message is printed:

```
file-name REMOVED ONLY FROM disk-name-1
```

WS (Warm Start)

This function causes a "branch" to the warmstart routine stored in the Read Only Memory (ROM) of the machine. It will cause the CMS operating system (file-name MCPX) to be loaded from disk into memory.

Operating Procedure:

Type "WS" then press OCK1. Wait for PKs 3, 4, 5, and 6 to be illuminated.

Press PK3.

Enter today's date in the format:

MM/DD/YY

when prompted to do so.

For further information, refer to the section headed "Loading the MCP".

INITIALIZATION OF DISKS ON CONSOLE-LESS SYSTEMS

The facility exists to initialize a Caelus disk on a console-less system (Caelus disks are BSM, 201I and Cartridge).

To IN/FE a disk on a console-less system, SAU is loaded into memory as normal by pressing PK2 and then PK6. When it is loaded into memory, it searches through the drives on line to find a file called SAU-.PARAM. This file is created by the utility SAU.INIT.

> WARNING
>
> It is recommended that the user ensure that no other SAU.PARAM files have been copied onto any disk, as the wrong disk could be initialized.

> WARNING
>
> Care should be taken when SAU.PARAM is being executed that all fixed disk assemblages are powered on. The reason for this is that the mnemonics are assigned sequentially to each powered on device. The algorithm searches for 211 disks first, from the highest channel to the lowest, and assigns mnemonics to any it finds that are powered on. It then searches for 201I disks, again from highest channel to lowest, and assigns mnemonics as before.
>
> **Example:**
>
> Consider a system with a 211 disk (DFA) and a dual-platter 201I disk (DFB, DFC). If the 211 disk is now powered off, the platters of the 201I disk will become DFA and DFB. Thus if an SAU.PARAM file is used to initialize DFB, DFC contains data and DFA is powered off, then DFC becomes DFB and is inadvertantly initialized.

If an error occurs, the function will terminate with an error pattern displayed on the MTR keyboard lights as shown below.

If the disk drive specified for initialization cannot be found, the utility halts with PK1 and PK2 lit on Bank 4. The drive to be initialized can now be powered on. When OCK1 is pressed, the utility searches for the drive again and continues execution.

This facility enables SAU IN/FE to be initiated with only one drive on-line, that is, a disk containing only the SAU utility code file, the SAU.PARAM file and CMSBOOT file.

When loaded, the required drive can be made ready and initialized. Thus, disks containing earlier versions of SAU utility can be re-initialized if necessary.

This is the only error condition which enables a retry to be attempted.

The error messages are displayed on the PK lights of the MTR keyboard. The MTR keyboard layout is as follows:

```
Bank Buttons        0 0 0 0 ← Bank 6

      1 2 3 4 5 6 7 8

      0 0 0 0 0 0 0 0     ← Bank 4

      0 0 0 0 0 0 0 0     ← Bank 1/2/3/5
```

The error messages are displayed on Bank 4, with the following PK lights illuminated:

PK1	– Disk has too many bad sectors for CMS use.
PK2	– No Bootstrap present.
PK3	– I/O failure – Bootstrap not copied.
PK4	– A cylinder is bad.
PK5	– Bad MTR track.
PK6	– No. of bad sectors exceeds 50 (on 201I)
PK7	– Track 0 bad.
PK8	– Track unusable – may be drive fault.
PK1 + PK2	– Dxx device error – Drive inoperable – Disk not present

PK2 + PK3 – Write Inhibited.
PK3 + PK4 – Off Cylinder.
PK4 + PK5 – Seek Timeout.
PK5 + PK6 – Controller problem.
PK6 + PK7 – Directory I/O error.
PK7 + PK8 – BSMII or 211 Disk found. Use DSKUTL.
PK2 + 4 + 6 + 8 – No file SAU.PARAM on disk.
PK1 + 2 + 7 + 8 – Error while opening SAU.PARAM.

Only one error condition will be displayed. If the initialize is successful, PK1 and PK2 on Bank 2 will be lit ready for the user to load the MCP, and all SAU.PARAM files found will be removed.

LOADING THE MCP

(This process is also called the "warmstart procedure").

The B 90 MCP code file is named "MCPX".

From the CMS Bootstrap Mode (PK3 to PK6 lit), depress PK3.

The bootstrap will search for a disk file called "MCPX". The search for the MCP proceeds as outlined in Bootstrap Search Algorithm later in this section.

For failures in the search, see below, System Load Errors.

If the MCP search is successful, the MCP is loaded to memory and MCP initialization takes place. The activity during this process can be distinguished on the D-lights.

The initial part of the MCP is loaded (D2 light flickers).

The console printer is initialized (if one exists).

AVR (Automatic Volume Recognition, see below) is performed on all peripherals (D2 flickering with D4 and D7 on).

For mini-disks, you will hear the disk start to click.

For cartridge disks, AVR procedures are very quick.

The system will print an MCP version message on the SPO (console, SELF-SCAN or terminal, according to data in the SYSCONFIG file), followed by a list of on-line peripherals that are powered up.

NOTE

The version of the MCP is identified by mark/level/patch numbers. For example, version 3.02.27 is mark 3, level 3.02, patch 3.02.27. A new software release is denoted by a higher level number (for example, 3.03). Within a release, higher patch numbers indicate improved versions of that level. For example, the application of two patches to 3.02.27 will create an MCP version 3.02.29 (see PATCHMAKER for details of how to patch MCP files).

The system will request the date. Depress the Ready Request button and verify that the ON, READY and ALPHA lights are lit.

Enter the date as requested, followed by OCK1. (Leading zeros are optional). The system prints a date message.

The MCP automatically loads the program SYS-SUPERUTL (D2, D3, D4, D5, D6 and D7 on).

If logging is specified in the SYSCONFIG file, the message

"COMMENCING LOG FILE CONSOLIDATION"

is given, and the TL utility is loaded (D2, D3, D4, D5, D6 and D7 on) and executed (D2 flickering with D4 and D6 on). At end of TL execution, the message

"TRANSFER COMPLETED"

"COMMENCING LOG FILE REALLOCATION"

"LOGGING IS INITIATED ON MM/DD/YY"

is given, and SYS-SUPERUTL creates new log files (D2, D3, D4 and D5 on).

Optionally, (depending on SYSCONFIG), a user program is loaded and executed (D2, D3, D4, D5, D6 and D7 on).

The warmstart is complete and the MCP enters idle state (see below).

It is advisable not to enter system commands until the complete warmstart procedure is over, otherwise confusion can result. Also, such input is not entered in the system log.

Example:

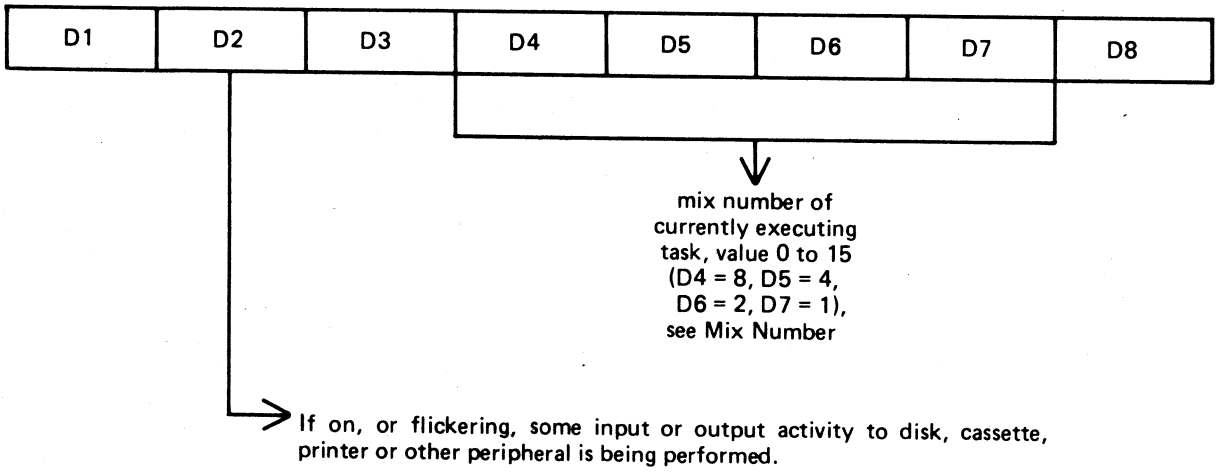
```
    B-90 DCMCP VERSION 03.02.32  79313
     DFA FA/     0 FILES OPEN
     DFB FB/ SYS DISK     0 FILES OPEN
    ENTER DATE AS MM/DD/YY
01/01/80
     01 JAN 80  80001  TUE
    COMMENCING LOG FILE CONSOLIDATION
    TRANSFER COMPLETED
    COMMENCING LOG FILE REALLOCATION
    LOGGING IS INITIATED ON 01/01/80 (MCP VER
...SION 03.02.32)
```

For possible errors, see System Load Errors, later.

BASIC OPERATION UNDER MCP CONTROL

D-Lights

During MCP execution D-lights D1 to D8 give an indication of system activity, as follows:



If the lights D4 to D7 are all off, this indicates system activity not related to a particular program. For example, AVR procedures result in D2, D4 and D7 being lit.

If the lights D4 to D7 are all on (value 15), this indicates processing of an input request or loading of a program.

MCP States

After warmstart is complete, the MCP is either idle or executing a task. The idle state is identifiable by the absence of activity on the D-lights. In this state, three distinct patterns can be encountered:

D2, D3 only lit:	The last activity was a program gone to EOJ.
D1, D3, D4, D5 only lit:	The last activity was a SYS-SUPERUTL function (IR, LB, LF, PD, RM, KX, or CH).
D1, D3, D4, D5, D6, D7 only lit:	The last activity was a system intrinsic.

When the MCP is executing a task, the mix number is evident in D-lights D4 to D7 (see Mix Numbers).

At any time when the system is idle, it is valid to terminate the MCP by a PO of the system disk (see PO intrinsic).

Mix Numbers

The following table indicates the mix numbers for B 90 activities:

mix-number	task	D4-D7 lights
0	* system activity	none
1-8	user programs	various
9	* AVR	D4, D7
10,11	utility programs	various
12	+ SYS-SUPERUTL	D4, D5
13	MCS	D4, D5, D7
14	* NDL	D4, D5, D6
15	* program loader/input requests	D4, D5, D6, D7

* these tasks are not shown in the response to the MX intrinsic.

+ SYS-SUPERUTL is not shown in the response to the MX intrinsic unless a particular function is being performed: in this case, the MX response shows the name of the function, for example, 12/RM , or 12/PD

Normal user programs or compilers are allocated the next available mix numbers in the range of 1-8. Utilities are allocated the lowest available mix number from 10 or 11.

Automatic Volume Recognition (AVR)

This procedure is carried out if any new media is loaded onto the system. If the procedure fails, the device is made not ready. The procedure varies for different media.

For fixed disk, disk cartridge, Burroughs Super Mini (BSM) disk, Burroughs Super Mini II disk, and ICMD, AVR will attempt to read the label.

For cassette, AVR will search for a CMS label or scratch label, otherwise the cassette is treated as unlabelled.

In all cases, if transient errors are suspected, Readying the peripheral (see RY intrinsic) re-initiates the AVR process.

Console Keyboard Under MCP Control

Under MCP control, the console keyboard may be used to enter system commands to the MCP, or solicited data to a program. Commands to the MCP may only be entered when the READY light is lit. Data to a program may only be entered on the alpha keyboard if the ALPHA light, but not the READY light, is lit. Input is terminated by an OCK key, or a PK key only if the corresponding PK light is lit. It can be seen that the keyboard can exist in one of three states; inactive, system enabled, or program enabled.

Since an understanding of the operation and behaviour of the keyboard and its indicators is essential, some important points are noted here.

If a disabled key is pressed at any time, or too much data is entered at once, then an error bell will ring and the ERROR indicator will light, this error condition must be reset before any further keyboard input can be made.

The RESET key has two basic functions, resetting the error condition explained above, or clearing the information keyed since an OCK or PK key was last pressed. This key is necessarily enabled whenever any alpha or numeric key is enabled.

When some incomplete information has been keyed either to the MCP or a program then it is necessary to terminate some information to this destination before entry to the other destination can be allowed. For example, once the ready light is lit an OCK key must be pressed to terminate input to the MCP before any input can be keyed to a waiting program. This restriction stands even if the reset key is used.

If the keyboard is enabled for input then the D-lights are not lit.

In order to enable keyboard input to the MCP the ready enable key must be pressed. This key will be ignored if the SCL/LOADER routine of the MCP is currently executing a system command or loading a program (D4, D5, D6, D7 lit) and a wait of up to 30 seconds may be necessary before any keyboard input can be made. Similarly if the MCP is very busy when a request for input is made there may be a delay of one or two seconds before the keyboard will become enabled for input. A maximum of two characters can be entered in this time, or a keyboard error condition will arise.

If the system is being used with the self-scan screen operating as a SPO device as well as a console file, then the ready enable key has an additional function. This key must be depressed in order to prompt the MCP to display either the complete screenful of messages on the screen, or the last message on the bottom line of the screen (scrolling the other information up one line). When the screen is being used as a console file, the screen will display information from the program using it until the ready enable key is depressed.

If the screen is displaying SPO information, depression of an OCK will return the display to the console file information.

Interrupting the MCP

It is possible to interrupt the MCP while it is running, this will inform the MCP that some action must be taken. The ability to interrupt the MCP is a good indication that the system is running satisfactorily. The only conditions where it may prove impossible to cause an interrupt is when the MCP is processing a system command or loading a program. Typical interrupts and the consequent processing include:

Press READY ENABLE button. This will cause the MCP to enable the keyboard for input.

Opening or closing the serial printer cover will cause the MCP to prevent or allow output to the printer accordingly.

Loading a disk or cassette will cause the MCP to read the label and perform the AVR procedure.

MEMORY DUMP TO CASSETTE

The system must be in the CMS Bootstrap state. If the system had clear-started, depress PK2 to get to the CMS Bootstrap state (PK3 to PK6 lit).

Depress PK4. The numeric light will be lit.

Insert a write-enabled cassette in any cassette drive unit, and wait until fully rewound.

Depress the numeric key (1 to 4) corresponding to the drive unit containing the cassette (CTA=1 etc.).

The contents of RAM will be written to the cassette. During the dump, an indication of the memory address being dumped is displayed on the PK lights. At the end of the dump, PK17 to PK24 lights are lit. The cassette will be labelled "MEMDUMP/MEMORY". The system will return to the bootstrap state (PK3 to PK6 lit).

Remove the cassette, clearly mark it with the date and time, and submit it with details of the fault to your support personnel.

For possible errors, see System Load Errors, below.

MEMORY DUMP TO DISK

The system must be in the CMS Bootstrap state. If the system had clear-started, depress PK2 to get to the CMS Bootstrap state (PK3 to PK6 lit).

Depress PK5. The system will search for a disk file called MEMDUMP. The search proceeds as outlined **in Bootstrap Scanning Algorithm** later in this section.

> The utility GEN.DUMPFL may be used to create a suitable file on the system disk. This must have been done before a dump to disk is attempted. It is recommended that for minidisk-based systems a spare minidisk is kept with a MEMDUMP file on, and this disk is the only disk loaded when the memory dump to disk is taken. This will avoid any possible confusion between MEMDUMP files on different disks.

The contents of RAM will be written to the MEMDUMP file. At the end of the dump, the CMS Bootstrap state will be entered (PK3 to PK6 lit).

For possible errors, see System Load Errors, below.

> NOTE
>
> Valid memory dumps will be produced only if the pertinent error condition was the last event on the system. Invalid memory dumps will occur if the machine main cabinet has just been switched on, or the error condition did not occur under MCP control, or PK1 was depressed while in initial state (PK1 and 2 lit).

ROM SCANNING ALGORITHM

From the Initial State (PKs 1 and 2 illuminated), pressing PK2 initiates a ROM load routine which searches through all present disks until it finds a valid bootstrap code.

For B 92s equipped with free-standing BSMII drives or 211 drives, and all B 93 and B 94 systems, the search consists of two passes: the first looking for removeable devices (1Mb, 3Mb and cartridge disks) and the second looking for fixed devices (201I and 211 disks). Each pass scans from the highest channel to the lowest, and from the bottom drive to the top drive in each channel.

If either a channel or a drive fault is encountered, the Error light is lit, and the inverse of the channel address is displayed on PK1 through PK8. That is, PK1 extinguished and PK2 through PK8 lit indicates a fault on channel 0, PK2 extinguished and PK1 and PK3 through PK8 lit indicates a fault on channel 1, and so on. This type of error will occur, after a two-minute timeout, if the scan encounters a powered-off SDI device (3Mb or 211 disk).

Pressing OCK1 causes the Caelus Primary or SDI Host Status to be displayed on PK1 through PK8, and the Caelus Secondary Status or SDI Device Status Byte 1 to be displayed on PK17 through PK24.

Pressing OCK2 causes the SDI Device Status Byte 2 to be displayed on PK1 through PK8, and the SDI Device Status Byte 3 to be displayed on PK17 through PK24.

Pressing the Reset key causes the scan to resume from the next drive in the case of a drive fault, or the next channel in the case of a channel fault. If both passes of the scan have been exhausted without a valid bootstrap being found, then PK1 through PK8 are illuminated. In order to re-attempt a Cold Start, it is necessary to press the Load Enable button, ensure that a disk containing a valid bootstrap is present on the system and is powered on, and repeat the above procedure.

For all other B 90 systems, the search differs from the above in the following respects:

There is only one search pass, which scans from the highest channel to the lowest, and from the bottom drive to the top drive in each channel.

If either a channel or a drive fault is encountered, the Error light is lit and the inverse of the channel address is displayed on D1 through D8. The Caelus Primary or SDI Host status is displayed on PK1 through PK8, and the Caelus Secondary or SDI Device Status Byte 1 is displayed on PK9 through PK16. Pressing the right-hand Form Feed key causes the SDI Device Status Byte 2 to be displayed on PK1 through PK8, and the SDI Device Status Byte 3 to be displayed on PK9 through PK16. From this state, pressing the left hand Form Feed key causes the Caelus Primary and Secondary Status, or the SDI Host and Status Byte 1, to be displayed.

Pressing the Reset key causes the scan to resume at the beginning of the algorithm (that is, at the bottom drive of the highest channel).

BOOTSTRAP SCANNING ALGORITHM

The successful loading of a CMS bootstrap causes PK3 through PK6 to be illuminated. When PK3, PK5 or PK6 is pressed on any B 90, a scanning algorithm is invoked, which is essentially the same as that described for B 93 ROMs, with the following differences:

In order that it may be readily determined whether a channel or drive fault is encountered by the ROM or the bootstrap, the bootstrap displays the channel address on PK1 through PK8 by illuminating the channel PK and extinguishing the rest. That is, a fault on channel 0 is indicated by PK1 being lit and PK2 through PK8 being extinguished, a fault on channel 1 is indicated by PK2 being lit and PK1 and PK3 through PK8 being extinguished, and so on.

Similarly, when both passes have been exhausted without the desired file being found, PK1 through PK8 are extinguished and only the Error light is illuminated.

The time-out incurred by the bootstrap encountering an SDI device which is powered off is 70 seconds.

When a channel or drive fault is encountered, pressing the Reset key causes the bootstrap to continue its scan in the same manner as described for the B 93 ROM.

SYSTEM LOAD ERRORS

The following table of errors cover all symptoms found during start-up of the B 90, warmstart, memory dump to disk or cassette, or entry to Stand-alone utilities.

SYMPTOM	POSSIBLE CAUSES	SUGGESTED ACTION
Sequential lighting and extinguishing of lights does not occur on entry to Initial State (for example, depressing Load Enable button).	MTR switch in wrong position.	Set MTR switch to "normal". Depress Load Enable button.
From Initial State, PK2 is ignored	Keyboard locked in "shift" mode.	Depress shift key. Depress PK2.
Depression of PK2 causes numeric light to be lit.	PK1 was depressed by mistake (this clears memory).	Depress Load Enable. Depress PK2.
Depression of PK2 causes keyboard lights plus ERROR light, to be lit:		
(a) D1 through D8 lit	No bootstrap found (disk not initialized correctly).	Check on disk used. Reload with correct disk. Press Load Enable. Press PK2.
(b) one D light not lit, plus PK lights lit	Disk media or drive fault.	See below for diagnosis. Note D and PK lights. Check that disk is not at fault by using disk in another drive. Power off faulty disk, replace with backup copy. Request technical assistance.
All keyboard indicators flash	Memory parity	Depress Load Enable, depress PK1 (to clear memory), then depress Load Enable again, and retry. Request technical assistance if not successful.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
Warmstart stops with D2 lit.	Console printer not ready.	Check forms transport closed. Check printer cover down. Check levers in correct position. If system does not continue, depress Ready then DCK1.
Print head does not initialize at start of warmstart.	Console printer faulty.	Open and close cover. Repeat warmstart. Request technical assistance.
Memory dump to cassette: System warmstarts	PK3 was depressed instead of PK4.	Memory dump cannot now be taken. Allow warmstart to complete.
Memory dump to cassette: keyboard ERROR light is lit when numeric key depressed.	Wrong cassette drive has been selected, cassette is not write-enabled, or tape is rewinding.	Correct fault in cassette. Depress Reset. Enter correct numeric key.
Memory dump to cassette: ERROR light is lit after 10 seconds.	Cassette is inserted backwards.	Correctly insert cassette. Depress Reset. Enter correct numeric key.
Memory dump to cassette: ERROR light is lit during dump.	Irrecoverable tape error, or accidental opening of cassette drive unit.	Allow cassette to rewind, or replace cassette. Depress Reset. Enter numeric key.
Memory dump to disk: Depression of PK5 causes one D light plus ERROR, to be lit.	No MEMDUMP file found on disk.	Load a disk with a MEMDUMP file and depress PK2 then PK5 . to retry the dump.
Memory dump to disk: ERROR light is lit during dump	MEMDUMP file on disk is too small to hold memory contents for this machine.	Load disk with larger MEMDUMP file; depress PK2 then PK5 to retry the dump.
Memory dump to disk: system warmstarts	PK3 was depressed instead of PK5.	Memory dump cannot now be taken. Allow warmstart to complete.

SYMPTOM	POSSIBLE CAUSES	SUGGESTED ACTION
Entry to S.A.U: Depression of PK6 causes one D light, plus ERROR, to be lit.	SAU could not be loaded from disk: disk error.	Note D and PK lights. Remove disk, replace with backup copy. Press Load Enable, PK2 and PK5. See below for diagnosis.
Entry to S.A.U: load stops with all keyboard lights lit.	Console printer not ready.	Check forms transport closed. Check printer cover down. Check levers in correct position. Utility should continue.

Diagnosis of Disk Errors at System Load Time

Caelus Device Primary Status

Bit	Interpretation
0	Drive Addressed (0 = Upper, 1 = Lower)
1	0 = Seek complete
2	0 = End of cylinder
3	0 = Search complete
4	0 = Secondary status condition
5	1 = Operational
6	0 = Seek incomplete
7	1 = Status OK

Caelus Disk Secondary Status

Bit	Interpretation
0	Ignored
1	Ignored
2	0 = Illegal seek
3	0 = Write inhibit
4	0 = Sector not found
5	0 = LRC error (parity)
6	0 = Illegal command sequence
7	0 = Device error

SDI Host Controller Status

Bit	Interpretation
0 1 2	Apart from MTR use, these bits should be set to 001, indicating that the Host Controller is ready to accept a Device Controller command or 1st segment data from Host
3	1 = Device Controller Status available
4	1 = Buffer available
5	1 = Buffer not empty
6	1 = Interface timeout (hardware fault)
7	1 = Interface timeout exceeded (hardware fault)

SDI Device Status

Bit	Byte 1	Byte 2	Byte 3
0	Drive No. Bit 0 (LS)	Error	Not Ready
1	Drive No. Bit 1	Search Unsuccessful	Disk Expiring
2	Drive No. Bit 2 (MS)	Corrected	Write Protected
3	Transfer Delay	Command not Accepted	New Disk
4	N Sectors Before Read	Command Error	Danger
5	N Sectors Before Write	Address Error/End of Drive	Confidence Test Completed
6	Operation Complete	Mandatory Int. to Host	Temporarily Unavailable
7	Interrupt	Address not Found	Unassigned

ERRORS UNDER MCP CONTROL

The following table of errors covers many symptoms found while using the B 90 under normal control of the MCPX, with suggested causes and actions to take.

SYMPTOM	POSSIBLE CAUSES	SUGGESTED ACTION
Unidentifiable problem	MCP file corrupted	Use backup copy of MCP
little happening, with D2 on, D3 off most of time	System thrashing; too many jobs or too large jobs in mix.	If console not in use, set CT ON. If light pattern above PK17 to PK20 is : on, on, off, on, then thrashing is confirmed : D lights 4 to 7 indicate mix number of executing task. Try entering MX to clear condition : if not sucessful, take memory dump and do clear start : request technical assistance.
	Extended memory not correctly used.	Run CONFIGURER and warmstart if necessary.
PK lights 17 to 24 flashing with "DF" pattern (on,on,off, on,on,on,on,on)	hardware disk I/O error during MCP task	Take memory dump: request technical assistance to analyze memory dump to find disk address for use in fixing the disk. To help find the fault, warmstart and run the KA utility on the system disk before executing any programs. (See KA, section 4). If shows any "AREAS ASSIGNED TWICE" the disk directory may be bad. DO NOT USE THIS DISK until recovery operations have been completed.

SYMPTOM	POSSIBLE CAUSES	SUGGESTED ACTION
PK lights 17 to 24 flashing with "AC" pattern (on,off,on, off,on,on,off,off) or "AD" pattern (on off,on,off,on,on, off,on)	see below:	(Take recovery action as for "DF" messages above: DO NOT USE THE DISK until recovery action is completed.)
	hardware disk I/O error during MCP disk directory operation.	Request technical assistance to fix the disk.
	logical error in disk directory due to previous fault.	Rerun using backup disk : analyze corrupted disk using S.A.U. PDX function and report.
non-zero retry count on PO of system disk	Deteriorating disk performance.	Ignore if 10 or 20 during normal running in one day. Call field engineer if greater rate than usual rate for the site.
	Improper care of removable disks: possibly failure to re-initialize disks after long use	All frequently used removable disks should be cleaned and re-initialized regularly (for example, on a monthly basis).
Logging not taking place	SYS-SUPERUTL not on systems disk.	Load SYS-SUPERUTL to systems disk.
	TL not on systems disk	Load TL to systems disk
	Logging not specified in SYSCONFIG file.	Rerun CONFIGURER and warmstart.
PD, RM function not available	No SYS-SUPERUTL	Load SYS-SUPERUTL to systems disk.
	SYS-SUPERUTL already performing one of its functions (see discussion in section 4).	Check with MX if SYS-SUPERUTL is performing a function (mix-number 12 gives the name of the function): wait then re-enter command.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
Cannot run data comm	No NDLSYS file on system disk	Load NDLSYS file (you may need to run an NDL compilation.
	NDLSYS file does not correspond to actual network on machine.	Load correct NDLSYS file; or recompile NDL program.
	NDL.INTERP not present on system disk.	Make sure the correct interpreter (for example MICROPOL) is named NDL.INTERP and located on the system disk.
	No MCS program	Ensure that the correct MCS is executing as well as any user data comm program.
Degraded performance	Lack of memory due to incorrect use of Extended memory, wrong SYSCONFIG file on system disk.	Run CONFIGURER, repeat after warmstart.
	Lack of memory due to memory fault at warmstart time.	Call field engineer.
System not active: programs in mix but D-light pattern does not change; keyboard input (for example, MX) is possible.	Program swapped out waiting on memory.	Enter <mix> GO to resume program execution; if symptom persists, program is too large for available memory.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
System not active : programs in mix but D-light pattern does not change; keyboard input (for example, MX) causes error bell to ring	Console file input in process.	Complete input using an OCK key.
	System printer is jammed.	Open and close printer cover to initialize print head. Repeat if not successful first time. If still not successful, perform a warmstart.
	Peripheral (possibly line printer) hang	Enter CL command. If not successful, make peripheral ready or not ready; switch if off or on; insert and unload spare cassette in cassette drive. If this fails, take memory dump, report fault, and warmstart. Note: never load or unload disk media, or cassette media which are in use, until system has returned to to initial state.
	MCP software error	Take memory dump, report fault, and warmstart.
System initializes (PK1 and PK2 lit)	Corrupted MCP code file.	Take memory dump, report error, rerun on backup disk.
	Unexpected hardware error.	Take memory dump, report error, re-warmstart.
	Unknown MCP error.	Take memory dump, report error, attempt to rerun. If still not successful, request technical assistance.
PK lights 17 to 24 flashing (pattern other than above)	MCP has detected an error condition from which it cannot recover.	Refer to section on MCP diagnostic messages for further details.
All keyboard lights flash	Memory parity	Initialize system, and rerun. If still faulty, call field engineer.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
Program load taking long time : keyboard disabled and D lights 4 to 7 flashing.	Program loaded thrashing due to lack of available memory; too many or too large programs in mix.	Try keyboard input. If this fails, initialize system and retry.
Hissing sound from disk drive.	Disk drive heads are damaged.	Stop work. Remove the disk from the damaged drive. THIS DISK IS PROBABLY DAMAGED, DO NOT LOAD ANY OTHER DISK INTO THE DRIVE. Call field engineer.
Non-functional disk drive.	Hardware fault on drive.	Load a good disk on crive (EXCEPT IF DRIVE HEADS HAVE CRASHED), run LS utility. If this produces errors, the drive is faulty: call field engineer.
I/O errors from disk.	Non-function disk drive.	See above.
	Crashed disk head.	See above.
	Physical damaged disk media.	Replace with new disk media.
	Disk media has bad areas.	Run CHECK.DISK to report on parity errors. Run KA to give disk directory analysis: If this shows AREA MISSING or AREA ASSIGNED TWICE, send the KA plus recent SPO message log to field engineer. Run DA to give further analysis of disk.
		Use backup disk for system files. Recover any other files by attempting to COPY them one-by-one from the corrupt disk to a good disk.

B 90 DEPENDENT UTILITIES

The following pages describe those system utilities which run under MCP control but which are relevent only to the B 90 CMS software.

CONFIGURER (Configure B 90 Software System)

This utility sets up data in a disk file called "SYSCONFIG" to determine how the B 90 is to be used. The SYSCONFIG file must be present on the system disk if any of the software options are to be used, other than the defaults given later. The data in this file is examined at warmstart time. Therefore any options specified will only be effective after the next warmstart.

To execute, enter

CONFIGURER

The system will display

"SPO LOGGING REQUIRED – Y OR N"

Enter "Y" if a log of SPO messages is to be kept on disk. This is normally required if the SPO messages are sent to to the SELF-SCAN display or data comm SPO where there is no hard copy. If SPO logging is not required, enter "N".

This option has no effect on B 90 systems – if log files are present, SPO messages will always be logged.

"ENTER NUMBER OF LOG FILES–
<MINIMUM 3, MAXIMUM 16>"

Enter a number between 3 and 16. At warmstart, this number of log files is created on the system disk to accommodate the logs (space permitting). The names of the files are "SYS-LOG-01" through "SYS-LOG-nn" where nn is the number entered. The minimum is 3 files.

"ENTER LOG FILE SIZES IN SECTORS–
<MINIMUM 32, MAXIMUM 16383>"

Enter a number to specify the size, in sectors, of the log files. This number must be within the range 32 – 16383. Thus the minimum disk space for system log files is (32 x 3) sectors.

NOTE

In order to create valid log files, the version of SYS-SUPERUTL contained in this release must be resident on the system disk, as log files are now considered to be system files, and not of type "DATA".

"INFORMATION FOR WARMSTART
ENTER ID OF FILE TO BE ZIPPED–"

If it is required to execute a program at the beginning of each warmstart, enter the name of the program code file (including disk-id if not on the system disk) to be executed together with the initiating message for the program to be zipped. This will only be executed if logging is switched on. If no program is to be run, make a null input (press OCK1).

"ENTER POWER OFF MESSAGE–"

Up to 80 characters can be entered, terminated by OCK1, as the message to be displayed when the system disk is powered off. If no message is required, press OCK1.

This is not implemented for B 90 systems.

"ENTER FID OF REQUIRED MESSAGE FILE–"

This is used to indicate to the MCP which local language message file is to be used. A null input causes the file SYSLANGUAGE to be used.

NOTE

For this release (3.03), SYSLANGUAGE is the only file available.

"DOES THE SYSTEM HAVE A LOCAL SPO <Y OR N>...?"

If the system is configured with a local SPO, enter "Y". If it is not, enter "N". If "Y" is entered, the next message is:

"SPO OPTION – SSA OR SPA"

Enter "SSA" if it is desired to display messages on the SELF-SCAN display. This frees the console-printer to be used entirely by programs with console files. The SELF-SCAN can also be used by programs. When the SELF-SCAN is used as an I/O device and as a SPO, the SPO messages are displayed when the Ready Enable key is pressed. SPO messages are replaced by program output (if any) when an OCK is subsequently pressed.

Enter "SPA" if it is desired to output SPO messages to the serial printer. This is most practical if the console printer is equipped with more than one tractor, in which case only the bottom left tractor is used by the MCP for SPO messages, and the other tractor(s) can be used exclusively by programs with console output files, or as a default line printer.

"DOES THE SYSTEM HAVE A REMOTE SPO <Y OR N...>?"

If "Y" is entered, the following question is displayed:

"ENTER FID OF REMOTE SPO MCS"

This request allows up to 40 characters to be entered. However, up to the first 12 characters will be treated as the MCS object FID. Any invalid character in this field will cause a message to be displayed and the request is repeated. The remainder of the 40 characters are not validated. If any characters are entered after the 40th character position, an error message is displayed and the request re-issued.

"JOB COMPLETED"

This message is displayed when the utility terminates. The utility now makes the file available on the system disk under the name "SYSCONFIG".

If an illegal response is made to a prompt, an error message is displayed and the prompt repeated.

NOTE

The presence of SYSCONFIG is mandatory on the system disk.

Defaults:

If the SYSCONFIG file is not present on the system disk, the following options are assumed:

SPO logging is enabled.

The number of log files is 3.

The size of each log file is 32 sectors.

No message is displayed at startup.

The SPO option is SPA.

Output Messages:

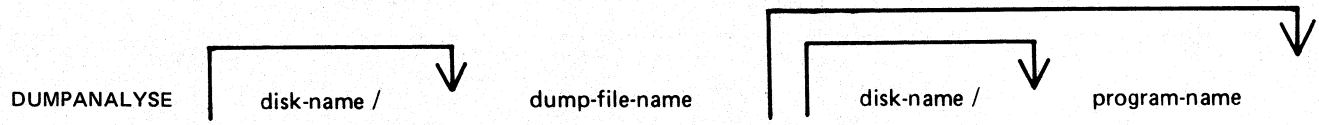
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ILLEGAL NUMBER OF LOG FILES	A number outside the range 3-16 was entered.	Wait for repeat, then enter correct value.
ILLEGAL FILE SIZE	A number outside the range 32-16383 was entered.	Wait for repeat, prompt, then enter correct value.
ILLEGAL -	A file-name entered does not have the correct format.	Wait for repeat prompt, then enter correct value.
INVALID CHARACTER IN IDENTIFIER identifier	Typing mistake	Check input and re-enter when prompted.
INVALID SELECTION - RETRY	Option has been specified which is not one of allowed values.	Wait for repeat prompt, then enter correct information.
PARITY ERROR ON WRITE TO FILE file-name. ERROR AT RECORD NUMBER number.	Parity error occurred while writing information to the SYSCONFIG file. Utility terminates.	Investigate cause of disk error. Use different disk. Repeat execution of program.
I/O ERROR DETECTED ON WRITE TO FILE file-name ERROR DETECTED AT RECORD NUMBER n	A write error has been detected when writing disk - the utility will terminate.	Investigate cause of disk error. Repeat execution of the program.

DUMPANALYSE (Analyze B 90 Program Dump Files)

When a program is DP'ed (see DP intrinsic) a file on the system disk is created with a name DMFILnn where nn is the mix number of the program dumped. This utility provides an analysis of this file, for use by technical analysts, printed on the line printer.

If the dumped program was written in MPL (used BILINTERPX) only the dump-file is required for a full analysis. If the dumped program was written in COBOL or RPG (used COBOLINTX), a fuller analysis can be provided by specifying the program code file as well.

Format:



Note that the program name must be the name specified at compile time and placed in the code file. This is normally the same as the disk file name but the file name could have been changed by the CH utility. if the wrong program name (or none at all) is given for a COBOL or RPG program, then the dump analysis will be incomplete (no COP table data can be analyzed). Because DUMPANALYSE has the capability of opening codefiles OTHERUSE LOCK.ACCESS, multiple executions of the utility are possible on the same file.

Examples:

To analyze the dump file of an MPL program DP'ed when it was mix number 1:

DUMPANALYSE DMFIL01

To analyze a dump file (which had been copied to a new file DPFILE on disk PRB) caused by DP'ing the COBOL program AR678:

DUMPANALYSE PRB/DPFILE AR678

To analyze the dump file created by DP'ing the RPG program RS202P. The program was at mix number 3 and the code file resides on disk RTB:

DUMPANALYSE DMFIL03 RTB/RS202P

On completion of the printed analysis, the dump file is removed from disk. To preserve a copy of the dump file, make a backup copy (using the COPY program) before executing the DUMPANALYSE program.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
WARNING BIL DUMP FL, SECOND PARAMETER IN INIT.MESS IGNORED	Program name has been specified for a dump file from an MPL program.	None.
PACK ID TOO LONG disk-name	Disk-name exceeds 7 characters. Utility terminates.	Check input and re-enter.

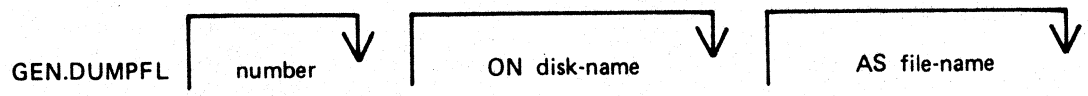
MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
FILE ID TOO LONG file-name	File-name exceeds 12 charactes. Utility terminates.	Check input and re-enter.
INVALID CHARACTERS IN FILE file-name	File-name or program name mis-spelled. Utility terminates.	Check input and re-enter.
file-name NOT FOUND	Dump file or program file cannot be found. Utility terminates.	Check input and re-enter. Or make file present by loading program or dump file from backup copy.
INTERPRETER VERSION NOT SUPPORTED interpreter-name	Dump file is corrupted; Error in compiler creating program file.	Use backup copy of dump file; retain program code file and request technical assistance.
FILE NOT FOUND	Dump file or program file cannot be found. Utility terminates.	Check input and re-enter, or ensure that file is present.
MFID NOT PRESENT FILE IN USE, OTHERUSE CONFLICTS BAD FILE TYPE DISK NOT FOUND DUPLICATE PACK DISK ERROR DISK LOCKED PARITY ERROR	These are reasons for the FILE NOT FOUND condition.	
NO USER DISK	The MCP is unable to load the utility.	Use backup copy of utility.

GEN.DUMPFL (Create Empty B 90 Memory Dump File)

Before the contents of memory can be dumped to disk, an empty disk file called MEMDUMP must be created. This file must be large enough to take the contents of all the memory of the system on which the dump is being taken.

The size of the file is specified by "number" in the range 64 to 11314. If "number" is not given, the size defaults to 11314 records, which corresponds to 1024K bytes of memory. The contents of the created file are initialized, and a repeating pattern of ASCII A through L is written throughout each record of the file.

Format:



Examples:

To create an empty file called MEMDUMP on the system disk for a B 90 with 80K bytes of memory:

GEN.DUMPFL 80

To create an empty file called MEMORY.DUMP on disk PRB for a 128K B 90:

GEN.DUMPFL 128 ON PRB AS MEMORY.DUMP

Note that this file-name must be changed to "MEMDUMP" before a successful memory dump can be taken. Any existing file of the same name will be removed when the new file is created by this utility.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
SPACE RESERVED	Successful EOJ	None.
SIZE TOO SMALL	"number" is less than 64.	Re-input.
SIZE TOO LARGE	"number" is greater than 11314.	Re-input.
DISK NAME TOO LARGE	Disk name is greater than 7 characters.	Re-input.
FILE NAME TOO LONG	File name is greater than 12 characters.	Re-input.

GT (General Trace)

The general trace command is an MCP intrinsic which displays various diagnostic information either on the system console or on the line printer. The general format is:

Format:

```
        ON
        OFF
GT      printer-peripheral
        diagnostic-class        switch-value
        disk-peripheral
```

To turn the trace printing on, with printing on the console (SPA) enter

GT ON

To direct the diagnostic printing to a line printer, specify the printer peripheral. For example

GT LPA

Note that the trace must also be turned ON in this case. The trace will be interleaved with any program printout.

To turn the trace off, enter

GT OFF

To display the cumulative number of retries performed on a disk unit since the last warmstart, specify the desired peripheral. For example,

GT DMA

The general B 90 machine code trace is implemented as follows:

Each trace point is identified by a diagnostic class and a diagnostic value. The class is one of 16, identified by a hex digit (0-F). This identifies the system function being performed, as follows:

0	Open/close file handling routines
1	Indexed file handling
2	Accept/display routines
3-7	Intrinsic functions (for example, SORTINTRINS)
8	Automatic volume recognition (AVR) routines
9	BAILIFF (task handling MCP routine)
A	Disk space allocate/deallocate routines
B	Interpreters (BILINTERPX, COBOLINTX, NDL.INTERPX)
C	MCP communicate handler (MCH)
D	Virtual memory (VM) routines
E	Task control routines (for example, BAILIFF)
F	I/O master interrupt handler (MIP)

The diagnostic value is a hex number (0-F) giving a measure of depth of trace required (0 is least significant, F is critical).

The GT command allows the storage of a switch-value for each diagnostic class. If the trace has been initiated via the GT ON command, then each time a trace point is encountered in the machine code, a diagnostic printout will occur if the switch-value for that class of trace is lower than or equal to the diagnostic value of that trace point. For example, if a particular trace point had a diagnostic value of C, then a trace point would occur if the switch-value for the appropriate class was in the range 0-C inclusive. If the switch-value set by the GT command was D, E or F, no trace print would occur.

The default values set at warmstart are:

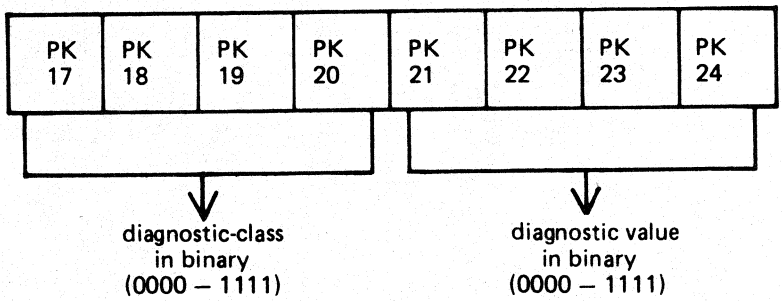
trace option OFF

diagnostic print directed to SPA

all switch-values set to F.

Therefore the only trace prints that will normally occur are the critical ones (diagnostic value F).

Note that if the trace option is ON the PK lights display the current trace point, whether or not the printing takes place, as follows:



For example, a trace point with class D (binary 1101) and severity 7 (binary 0111) will cause PK lights 17, 18, 20, 22, 23 and 24 to be lit.

The switch-value entered in the GT command must contain two hex digits:

first digit – switch-value for register diagnostics

second digit – switch-value for memory diagnostics

If no options are specified for the GT command, then the current switch-values for each of the 16 diagnostic classes is displayed.

Examples:

To find the retry count for DMA and DKA:

```
GT DMA
   DMA        0        RETRIES

GT DKA
   GT DKA NOT ON SYSTEM
```

To set the switch-values to CC for interpreters, then interrogate all switch-values, then turn on printing to LPA:

```
GT     B      CC
GT
     DIAG SWITCHES FF FF FF FF FF FF FF FF FF
.....FF FF CC FF FF FF FF
GT LPA
GT ON
```

Note that only the disk-peripheral option and the GT command with no further options result in an immediate response to the operator.

Format of diagnostic printout:

The format of a register diagnostic message is given here, where each X represents a single hexadecimal digit. The hex string is printed on one line.

register :	AD	BO	B1FL	J	K	L
	xx	xx	xxxx	xxxx	xxxx	xxxx

register	M1	M2	WR	X	Y	MXA	MXB	UMRX	AD,ESCT
	xxxx	xxxx	xxxx	xxxxxxxx	xxxxxxxx	xxxx	xxxx	xxxx	xxxx

The diagnostic class and value are given by the AD register (first 2 digits, repeated in first pair of last four digits). For example, a DF diagnostic with M1 = 1111 would look like:

```
DF  xx xxxx xxxx xxxx xxxx xxxx
                 1111 xxxx xxxx xxxxxxxx xxxxxxxx xxxx xxxx xxxx DF xx
```

where x indicates any hexadecimal digit.

When GT is switched on, system fatal errors (which would normally result in a set of PK lights 17 to 24 flashing) are reported as a diagnostic printed message followed by initialization to the Initial state. A memory dump should normally be taken to find more information. When GT is switched off, the pattern of PK lights 17 to 24 that are set flashing on a system fatal error correspond to the value of the AD register.

ND (New Density)

This intrinsic allows the operator to define the print density on suitable console printers.

Format:

ND peripheral density

The density field is a single character which specifies the density as follows:

A. The greatest number of characters per inch available on the printer.

B. The second highest number of characters per inch.

and so on.

1. The greatest number of lines per inch available on the printer.

2. The second highest number of lines per inch.

and so on.

Examples:

ND SPA 1
ND SPA B

When the number of characters per inch changes, the operating system will adjust the values of page width and offset (previously set by an FD command or by default) so that any subsequent output is restricted to the part of the platten available with the previous density. The page height will be adjusted when the number of lines per inch changes.

The new density, together with the adjusted values of page height or page width and offset, are recorded in the system configuration (SYSCONFIG) file and are remembered across system shutdown and warmstart.

At warmstart, if any inconsistency is found between the system configuration file information and the capability of the current printer, then the system configuration file information is ignored and default values are used. The system configuration file is unaltered.

The densities available on different printers are as follows:

PERIPHERAL	DENSITY	VALUE
120 cps console printer	15 characters per inch	A
	10 characters per inch	B
90 cps console printer	16 2/3 characters per inch	A
	12 1/2 characters per inch	B
	10 characters per inch	C
	8 lines per inch	1
	6 lines per inch	2

Output Messages:

MESSAGE	POSSIBLE CAUSE	SUGGESTED ACTION
input INVALID	(1)the specified density value exceeds the number of options available on the configured console	(1)enter correct input
	(2)the printer is in use	(2)wait until program has closed the console file
	(3)the mnemonic is incorrect	correct the input and re-enter
input NOT ON SYSTEM	The specified peripheral does not exist	Correct the input and re-enter

PATCHMAKER (Patch B 90 Machine-Code Object Program Files)

This utility reads a file of patches from disk or cassette or from console or SPO keyboard input and patches a machine-coded system software item. Stringent conditions are enforced to make the patch, including the necessity to apply each patch in the correct order. Each line of patch data contains an extra byte which is used as a checksum for that line. If a line of data is incorrectly entered, the operator is prompted to re-enter that line – this does not invalidate previously entered lines of data. An overall checksum is also included to verify the entire patch. All previous patches must be applied before making the next patch.

In order to achieve continuity of system code-file names, the following naming convention will be used:

For B 90 MCP	MCPXn.nn.nn
For B 90 BILINTERP	BILXn.nn.nn
For B 90 COBOLINT	COBXn.nn.nn
For B 90 SORTINTRINSIC	SINTXn.nn.nn
For B 90 NDLINTERP	NDLXn.nn.nn
For B 90 MICROPOL	MICXn.nn.nn
For B 90 SAU	SAUn.nn.nn

where n.nn.nn is the software mark, release and patch level (for example, MCPX3.03.10).

It is essential that an unused copy of all micro-coded software items is retained for patching. It is not possible to patch a B 90 MCP that has been used in normal B 90 operation. This is because certain MCP tables included in the code file are modified during operation. This modification would cause the check digit calculations in PATCHMAKER to fail. On any system equipped with a console, the Stand-Alone Utility (SAU) COPY function may be used to create unused copies of all system software for patching purposes, and also to create backup copies of patched software. MCP COPY utility must be used on console-less systems. Files on the system disk cannot be patched: they must reside on a user disk.

To execute, enter

PATCHMAKER

The utility runs in interactive mode.

The utility displays on the SPO:

"IS PATCH FILE TO BE ENTERED FROM CONSOLE"
"<mix-number>/PATCHMAKER ACPT"

and waits on an ACCEPT.

If the operator enters

AX<mix-number>YES

then the console file will be opened. A response of "NO" causes the utility to display on the SPO:

"IS PATCH FILE TO BE ENTERED FROM SPO–"
"<mix-number>/PATCHMAKER ACPT"

If the response is "YES", then the system displays:

"IS PATCH FILE TO BE OUTPUT ON DISK"
"<mix-number>/PATCHMAKER ACPT"

If the patch file is not to be entered from either console or SPO, the system displays:

"IS PATCHES FILE ON CASSETTE–"
"<mix-number>/PATCHMAKER ACPT"

If the response is "YES", a tape file named "PATCHES" is required, while a response of "NO" requires a disk file on the system disk named "PATCHES", and causes the system to display:

"ENTER DISK IDENTIFIER (OF FILES TO BE PATCHED)"
"<mix-number>/PATCHMAKER ACPT"

If a console file is opened, the utility displays:

"?DATA PATCHES
SIGNIFY WHETHER PATCH FILE IS TO BE
OUTPUT ON CASSETTE OR DISK"

The operator may enter, via an ACCEPT, either "CASSETTE" or "DISK". Patches entered subsequently on the keyboard will be written to the specified medium.

The utility displays:

"ENTER PATCHES NOW"

The patches must be entered via the keyboard from the hard-copy provided. The characters must be entered exactly as supplied, although spaces are not significant and may be entered as found convenient. The utility will ask for resubmission of lines which are incorrect.

If correct, the message

"PATCHES HAVE BEEN ENTERED CORRECTLY"

is given, followed by "?END PATCHES" if a console file has been used.

When the correct patch has been entered, the utility displays on the SPO:

"IS PATCHING NOW REQUIRED"

and waits on an ACCEPT.

If the operator enters anything other than

AX<mix-number>YES

then the utility will go to normal EOJ. If the operator enters "YES", then patching will be carried out.

The utility displays:

"ENTER DISK IDENTITY OF FILES TO BE PATCHED"

and waits on an ACCEPT.

The operator must enter the disk name of the disk on which reside all the software files to be patched. This must not be the system disk.

The utility displays:

"FILE <file-name> BEING PATCHED,
TO BE SAVED AS "

and waits on an ACCEPT.

The file-name is the name of the file to be patched, which is generated from the patch itself. This file must be present on the disk specified earlier. The operator must enter the new name for the file after it has been patched. The patched file will be retained only if the patching is successful.

Checksums are computed and verified before and after patching. If the utility goes to EOJ without displaying any error messages, then the patching has been successful.

A sample B 90 CMS System Software Flash is shown in the following paragraphs.

Sample Flash

B 90 CMS System Flash No. 3.03-XX

SUBJECT: B 90 MCP Version 3.03.04

PROBLEM: The FUNCTION F can result in.....

TEMPORARY FIX:

Follow the instructions for PATCHMAKER found in section 8 of the Series B 90/B 900/CP 9500, B 1800/B 1900 Computer Management Systems (CMS) System Software Operation Guide, Form No 2015228.

See also B 90 System Flash No. 3.03-00 and B 90 CMS 3.03 Release Documentation.

The sub-title of the relevant procedure is "PATCH IMPLEMENTATION".

A	MCPX3.03.04					
B	3030	3030	3030	304D	4350	D9
	5833	2E30	332E	3037	2000	E4
	0000	00B5	3D6F	C12C	0132	32
	3820	38C4				
	0332	3837	2033	3020	1C7F	F6
	3120	0A5A	001C	0216	3B0F	E2
	197A					
C	C47A * (ASTERISK)					
	NOTE: The character * must be entered. The word (ASTERISK) is included for clarification only.					
D	MCPX3.03.05					
E	MCPX					

NOTE

This is not a live patch. It is shown only to illustrate the new patching procedure.

The following procedure is designed to make patching easier for the user to understand and implement. The format for each patch, issued in a CMS Flash, will conform to the sample Flash shown previously.

System Flashes which involve patches to system software will consist of a number of "labelled inserts", (A through E).

The instructions contained in the procedure specified below contain "labelled blanks", (A through E). When applying a patch using this procedure, the "labelled blanks" are replaced with the text of the corresponding "labelled inserts" of the Flash.

All patch data is entered through the SPO device keyboard (SELF-SCAN, CRT, Printer or Datacomm SPO). This convention allows one set of instructions to be used for all B 90 systems.

Operating Instructions

Before executing the program PATCHMAKER, perform steps 1 and 2.

1. Ensure that the program PATCHMAKER is resident on the system disk.

2. COPY an unexecuted version of the file __A__ to a user disk. This is the file to be patched. Use SAU COPY if your system is equipped with a console, use MCP COPY utility for console-less systems.

Execute PATCHMAKER as follows:

3. Enter "PATCHMAKER"

   The system displays
   "IS PATCH FILE TO BE ENTERED FROM CONSOLE – "
   "<mix-number>/PATCHMAKER ACPT"

4. Enter "AX <mix-number> NO"

   The system displays
   "IS PATCH FILE TO BE ENTERED FROM SPO – "
   "<mix-number>/PATCHMAKER ACPT"

5. Enter "AX <mix-number> YES"

   The system displays
   "IS PATCH FILE TO BE OUTPUT ON DISK"
   "<mix-number>/PATCHMAKER ACPT"

6. Enter "AX <mix-number> YES"

   The system displays
   "ENTER PATCHES NOW"
   "<mix-number>/PATCHMAKER ACPT"

7. Enter "AX <mix-number> space __B__ for each line of the flash as supplied.

   If entering the patch through a console SPO (SELF-SCAN, CRT, or Printer), terminate each line by pressing OCK1.

   If entering the patch through a Datacomm SPO, terminate each line with the transmit (XMT) key.

   The spaces contained in __B__ can be entered. This makes the data easier to read from the flash and easier to enter from the keyboard. These spaces are ignored by PATCHMAKER.

   If the system displays the message
   "ERROR IN LAST LINE – RESUBMIT"
   "<mix-number>/PATCHMAKER ACPT"
   check the content of the last line entered, and re-enter the data correctly.

8. When all the lines of __B__ have been entered correctly, enter "AX <mix-number> space __C__.

   Ensure that the last character entered of this line is the character * (asterisk or star). This character informs PATCHMAKER that this is the last entry and is the overall checksum for the patch.

   For non-English keyboards, this is the key which produces the ASCII code "2A".

9. When __C__ has been entered correctly, the system displays
   "PATCHES HAVE BEEN ENTERED CORRECTLY"
   "IS PATCHING NOW REQUIRED – "
   "<mix-number>/PATCHMAKER ACPT"

10. Enter "AX <mix-number> YES"

    The system displays
    "ENTER DISK IDENTIFIER (OF FILES TO BE PATCHED)"
    "<mix-number>/PATCHMAKER ACPT"

11. Enter "AX <mix-number> <disk-name>"
where <disk-name> is the name of the user disk on which __A__ resides (see step 2).

The system displays
"FILE <disk-name>/__A__ BEING PATCHED, TO BE SAVED AS – "
"<mix-number>/PATCHMAKER ACPT"

12. Enter "AX <mix-number> <disk-name>/__D__"
where <disk-name> is the name of the user disk on which __A__ resides.

The system checks the integrity of the patches entered above against the old file. If no mismatch is detected, __A__ is copied by PATCHMAKER, the patches are applied and the file is saved as __D__ on disk <disk-name>.

The system displays
"PATCHES TO <disk-name>/__A__ SUCCESSFULLY ACCOMPLISHED,"
" AND SAVED IN <disk-name>/__D__"
and PATCHMAKER will go to EOJ.

The following steps for installing the patched software assume the normal case of one operational system disk per system and several user disks. The actual procedure can be varied if you have differing requirements.

If the system is equipped with a console, perform steps 13 through 18.

If the system is a console-less system, perform steps 19 through 23.

13. Logically PO the user disk containing __D__ and write disable the disk.

14. Load Stand Alone Utilities (SAU).

15. Using SAU RM, remove __A and E__ from each disk in your library (including your system disk).

    Replace this file on each disk with __D__.

16. Retain a copy of __D__ which must never be executed in order that future patches can be applied if necessary.

    The patch implementation is now complete for your library files.

17. On the system disk, using SAU CH, enter
CH <system-disk-name>/__D__ TO __E__.

18. Enter "WS"

    The system can now be warmstarted and will be fully operational with the newly patched software.

For systems not equipped with a console, the procedure is similar except the COPY functions are performed under MCP control using the CMS COPY utility.

19. Using the MCP RM function, remove __A__ from all of your disks. (DO NOT attempt to remove __E__ from your system disk).

    Replace this file on each disk with __D__.

20. Retain a copy of __D__ which must never be executed in order that future patches can be applied if necessary.

21. Using the CMS COPY utility under MCP control, enter
"COPY <disk-name>/__D__ to __E__.

    This will copy the newly patched software to the system disk which is currently running.

22. Logically PO the system.

23. Warmstart the system.

During this warmstart, the old executing version of __E__ will be automatically removed and the newly patched version of __D__ will be made available for execution.

The system is now fully operational with the newly patched software.

NOTE

The above patching procedure illustrates the method of entering patches through the SPO. If the option of entering patches through the console is exercised, each line of part B of the patch should be terminated by OCK1, unless it is either the last line of part B or contains less than 11 bytes, in which case it is terminated by pressing OCK2. Part C of the patch is terminated by pressing PK6, and does not contain the asterisk when entered through the console.

NOTE

If patching of the NDL Interpreter or MICROPOL is required, you are recommended to seek technical assistance from your local Burroughs representative. This recommendation will be included in the text of the Flash.

Output Messages:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
ERROR IN PATCHES ENTERED - RUN ABORTED	Keyboard input has been made in error.	Check keyboard input with hard-copy, re-execute utility.
PATCHES TO file-name-1 SUCCESSFULLY ACCOMPLISHD AND SAVED IN file-name-2	Successful patching run.	Make backup copy of file-name-2 (unused patched file) for future patching.
INITIAL CHECKSUM DISCREPANCY	Wrong software file has been submitted for patching.	Re-execute utility with correct input file.
ERROR IN LAST LINE - RESUBMIT	The last line of patch data entered did not match the checksum entered for the line.	Check data entered for line in error and re-enter data correctly.
Other error messages	Wrong input file; other serious software errors.	Request technical assistance.

Note: the device kind of the utility's console file may be modified via the MODIFY utility to one of KB, KD or KP if required. The internal name of the console file is THREADS.

PMB90 (Analyze B 90 Memory Dumps)

This utility is an interactive program which produces a formatted print of the contents of a memory dump tape or disk file produced by the memory dump feature of the B 90 bootstrap ROM. The tape must be labelled "MEMDUMP/MEMORY" or the disk file must be named MEMDUMP on the system disk unless otherwise specified to PMB90. Neither MEMDUMP file-name nor disk-name may start with a number.

The utility requires the following files on the system disk:

PMB90 – object code file

PMBHELP – data file of prompt messages

PMBERROR – data file of error messages

PMBM.xxxxx – data file for information on MCP xxxxx

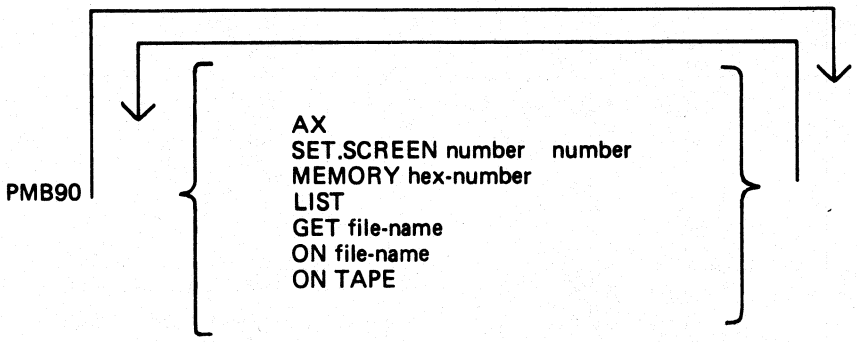
PMBO.xxxxx – data file for more information on MCP xxxxx

The x values vary with each release: the files provided with each release must be used with that release, otherwise incorrect analysis may be made. The utility checks version numbers of its data files and issues a warning if they are not the same as the PMB90 version. All data files include version numbers.

Starting the Utility

The utility can be executed with a number of options in the initial message. The format is as follows:

Format:



The meaning of these options is as follows:

AX

The program will use the SPO via DISPLAYS and ACCEPTS to communicate with the operator. If this option is not specified the console will be used for communication.

SET.SCREEN

This option sets the screen page and line sizes from the numbers specified. These must be set if the DISPLAY option is used (see later). The input medium is the console keyboard but echoing of input is on the self-scan screen.

MEMORY

If this option is specified, the program will read only the first part of the memory dump, up to the byte given by the hexadecimal number. Note that use of this option could cause the program to fail on certain print options if the analysis requires a part of the dumped memory that has been excluded.

LIST
This option will list the contents of the unanalyzed memory dump during the initial reading of the dump by the program. The list is in groups of 4096 (4K) bytes.

GET
This option can be used to specify the name of the memory dump disk file. If not used, the file "MEMDUMP" on the system disk is opened. If this file is not present, a cassette labelled "MEMDUMP/MEMORY" will be opened.

ON
This option can be used to specify the disk on which resides the memory dump file, if not the system disk.

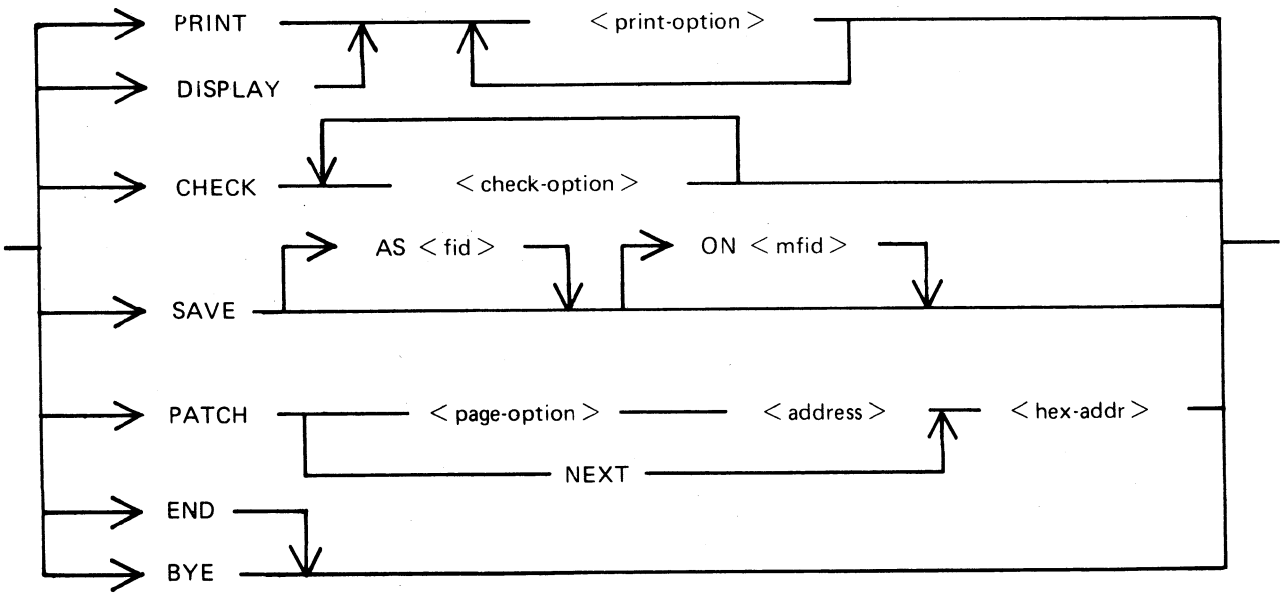
ON TAPE
This option directs the utility to access memory dump files on cassette.

Using the Utility

Where the console is to be used for communication, the user is informed if no console is available, and is given the opportunity to make a console file available. Non-printer console files allow up to 224 characters input.

The analysis takes place in an interactive manner. All the available options and instructions on how to use them are provided in response to the input "HELP". Further details on a particular option are provided in response to the input "HELP option". A knowledge of the MCP is required in order to diagnose the reason for any particular memory dump.

The complete dump options are given here in "railroad diagram" format, with further details later.



PRINT
This option specifies that output from the following list of <print-option>s is to be printed on the console.

DISPLAY
This option specifies that output from the following list of <print-option>s is to be displayed on the self-scan screen. In this case the SET.SCREEN parameter should have been used in the initial message.

CHECK
The two <check-option>s are MEMORY.LINKS and ALL.MEMORY. The MEMORY.LINKS option gives a print of all memory links from the initial pointer (PTRX) until the end of the chain or an inconsistency is reached.

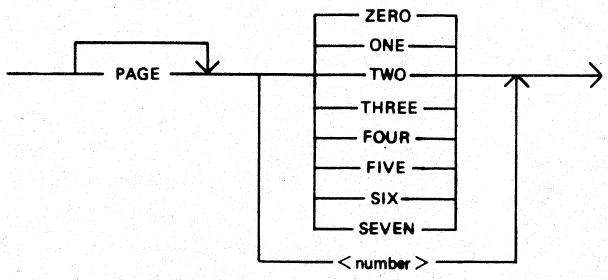
SAVE

This option enables a copy of the dump (patched if required) to be made on the specified disk with the specified file-name.

PATCH

This option enables invalid areas of memory to be patched in the dump, to enable PMB90 to continue its analysis.

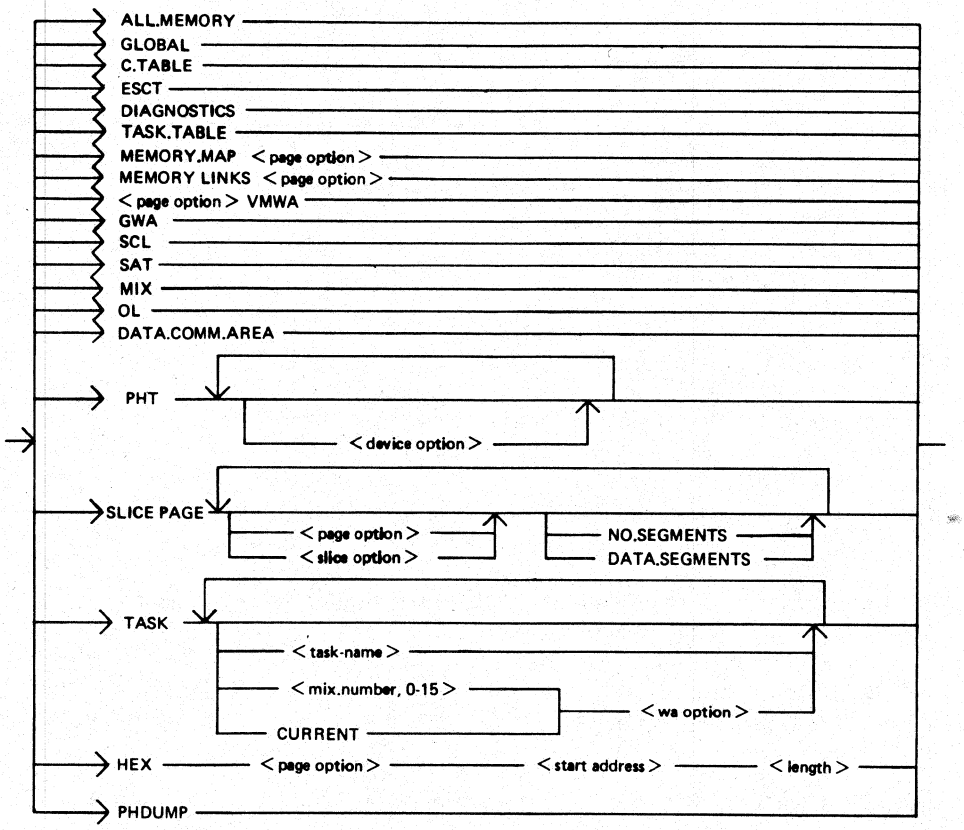
The <address> is four hex digits, and the <hex-addr> may be from 1 to 16 hex digits. The <page option> is as follows:



END, BYE

These alternative options cause PMB90 to go to normal EOJ.

The format of <print-option> is as follows:



ALL.MEMORY

This option creates an analysis of fixed MCP data areas, configuration table and task table if present, all peripheral handler tables, memory links and analysis of all locked slices in memory plus all overlayable code and data segments present in memory at the time of the dump.

GLOBAL

This option creates an analysis of the fixed MCP data areas. It includes the last code executed.

C.TABLE

This option gives a print of the configuration table if it was in memory at the time when the dump was taken.

ESCT

This option gives a print of the mix table (execution scan priority table).

DIAGNOSTICS

This option gives a print of the MCP's diagnostic buffer area.

TASK.TABLE

This option gives further analysis of the mix, if the overlayable task table was present in memory at the time when the dump was taken.

MEMORY.MAP

This option provides an analysis of the layout of memory, and may be output for any specified page.

MEMORY.LINKS

This option analyzes the layout of the overlayable area of memory, and may be output for any specified page.

VMWA

This option gives a print of the virtual memory work area only.

GWA

This option gives a print of the Global work area only.

SCL

This option prints the keyboard buffer only.

SAT

This option gives a print of the Slice Address Table only.

MIX

This option gives a selective analysis of parts of the dump relating to the tasks running at the time of the dump.

OL

This option provides a selective analysis of peripheral configuration information.

DATA.COMM.AREA

This option prints the areas of memory relating to data communications.

PHT

This option gives a print of selected Peripheral Handler Tables. If the <device-option>s are absent, then all peripherals attached to the system at the time of the memory dump are analyzed. Allowable values for <device-option> are:

CX – Channel Expander
LP – Line Printers
SP – Serial (console) Printer
CT – Cassettes
DK – Cartridge Disks
DF – Fixed Disks
DM – BSM, BSMII Disks
KB – Keyboard
SS – Self-Scan
ADC – Asynchronous Data Comm Controllers
SDC – Synchronous Data Comm Controllers
DI – ICMDs
RT – Time-of-Day Clock
DCPP – Data Communication Power Pak

SLICE

This option provides selective printing of locked slices of memory or of data segments. It does not allow for associated segments to be output. The <slice-option> may be either a "slice-number" in the range 0-45 or one of the following names:

DISKDDR
LPDDR
PANDDR
KBDDR
CASSDDR
SENDDR
CONSOLE
INXS
SCREENSN
SUSN
INITIALIZE
ADCDDR
SDCDDR
OPENCLOSE
DCCH
SPO
CONBUFSN
SCLBUFSN
ICMDDDR
OCOMSN
DIAGSN
REMSPOSN

TASK

This option prints the contents of a Task Control Block (TCB).

The <task-name> may be one of

NDL
MCS
BAILIFF
SCL
LOADER

The <wa-option> may be one of

MPLII
BIL
COBOL
RPG
SORT
NDL

HEX

This option provides a print (or display) in hexadecimal and byte format of selected parts of memory. The <start address> is a four-hex-digit number and the <length> is also specified as a four-hex digit number.

PHDUMP

This option causes the peripheral handling dump area to be output.

Example:

To obtain a complete memory dump print on the console printer:

```
PMB90 (OCK)
PRINT ALL.MEMORY (OCK)
```

To obtain a dump of the data comm buffers, plus the data comm controller device-dependent routines, plus the MCS and NDL task tables:

```
PMB90 (OCK)
PRINT DATA.COMM.AREA (OCK)
PRINT PHT ADC SDC (OCK)
PRINT TASK MCS NDL (OCK)
```

Note:

When submitting memory dumps for analysis, it is helpful if some preliminary analysis has already been performed. The following option is recommended:

```
PMB90
PRINT MIX OL MEMORY.MAP MEMORY.LINKS GLOBAL PHT TASK CURRENT
```

Always provide the MEMDUMP file on magnetic media even if this preliminary analysis has been performed.

POWER OFF

Logically power off all user disks (see PO command if under MCP control, or SAU PO command if under SAU control).

Logically power off the system disk (see PO command if under MCP control). Wait until the system returns to the initial state, that is PK1 and PK2 are lit.

If the PO command cannot be used, due to some system error, then the system should be halted by pressing the Load Enable button, causing the system to return to the initial state with PK1 and PK2 lit.

Remove all removable disk media.

A mini disk can be removed immediately the unit door is opened.

A disk cartridge can be removed only when the red stop light is lit, assuming that the drive is functioning correctly.

Power off the disk units (failure to remove disk media before this, may result in subsequent media corruption).

Remove all cassettes from the system.

Power off the main cabinet (this must be the LAST action after all peripherals have been switched off).

Note on disk removal:

There are only two situations when it is valid to remove a disk:

where the MCP is not running and the disk is not in use.

where the MCP is running, but the disk is a user disk which is logically powered off after using the PO command: note that the PO command does not cause a disk to become logically powered off if it is in use, but the PO will be completed only after all activity on the disk is complete.

Note on power failures:

If the main cabinet is switched off accidentally (for example, by power failure), remove all disks and cassettes before it is switched back on.

SAU.INIT

This utility is used to create a parameter file for initializing Caelus disks (BSM, 201I, Cartridge) on consoleless systems. The parameter file is called SAU.PARAM. The Stand Alone Utility will extract parameters from this file. For further information on execution, refer to the description of Stand Alone Utilities earlier in this section.

Format:

```
SAU.INIT —— DEFINE —— ON —— <disk-id> —— [ IN ] ——>
                                         [ FE ]

>—— FILES <number> —— OWNER "<name>" —— SERIAL <serial-number> ——>

>—— DISK <disk-name> —— DRIVE —— [ DKx ]
                                 [ DMx ] ——
                                 [ DFx ]
```

OR

```
SAU.INIT —— DISPLAY —— <disk-id> ——>
```

where	<disk-id>	::= up to 7 legal characters
	<number>	::= up to 4 numeric digits (1-2804)
	<name>	::= up to 14 characters inside quotes (")
	<serial-number>	::= 6 decimal characters

The DEFINE function, which requires all parameters to be specified, creates a file "SAU.PARAM" on the specified disk. The file consists of a single 180-byte record, and has a system data filetype. If another SAU-.PARAM file exists on the specified disk, it will be removed when DEFINE is invoked.

The DISPLAY function displays the parameters defined in an existing "SAU.PARAM" file which has been previously created by the DEFINE function.

The format of the "SAU.PARAM" record is as follows:

Byte Offset	Length	Parameter
0	2	"IN" or "FE"
2	3	"Dxy"
5	8	"mm/dd/yy"
13	4	"0001" – "2804"
17	6	
23	7	Disk-Name
30	14	Owner
44	121	Dummy
165	15	"B90DISKINITCDMU"

Output from the DEFINE function is:

```
IN  ]
    ]—— <disk-id> / SAU.PARAM CREATED
FE  ]
```

If there is no available area for the file on the specified disk, then the following messages are displayed and the utility goes to End-of-Job:

```
                                  ┌─ IN ─┐
DATA SPACE NOT AVAILABLE ON ──┤      ├── <disk-id>
                                  └─ FE ─┘

IN ┐
   ├── <disk-id> / SAU.PARAM NOT CREATED
FE ┘
```

Output from the DISPLAY function for a valid "SAU.PARAM" file is:

```
                                  ┌─ IN ─┐
PARAMETERS IN FILE ──┤      ├── <disk-id> / SAU.PARAM –
                                  └─ FE ─┘
IN ┐
   ├── DISK Dxy AS <disk-id> FOR <number> FILES
FE ┘

SERIAL NO. <serial-number>, OWNER <name>, DATE <date>
```

If a file "SAU.PARAM" is found but has incorrect attributes or checkstring, the following message is displayed and the utility goes to End-of-Job:

```
IN ┐
   ├── <disk-id> / SAU.PARAM NOT CMS STANDARD
FE ┘
```

Both the DEFINE and DISPLAY functions of the utility check the disk which contains SAU.PARAM for the file "CMSBOOT", and inform the user whether or not it has been found.

If "CMSBOOT" is found and has the correct file attributes and checkstring, the utility displays the following message:

<disk-id>/CMSBOOT IS VERSION <number>

If the attributes and checkstring are incorrect, the following message is displayed and the utility goes to End-of-Job:

BOOTSTRAP FILE <disk-id>/CMSBOOT NOT CMS STANDARD

NOTE

It is not necessary for "CMSBOOT" to reside on the same disk as "SAU-.PARAM" when the Stand Alone Utility is executed with parameters from "SAU-.PARAM"

SECTION 9

B 900/CP 9500 DEPENDENT SYSTEM SOFTWARE

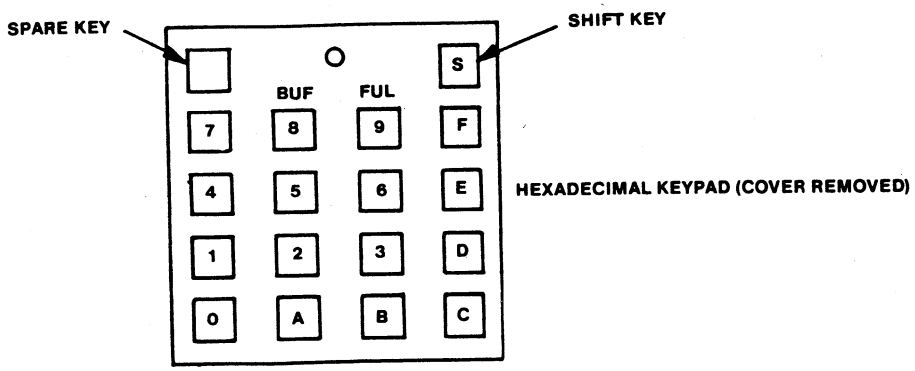
GENERAL

This section covers those items in the CMS software which are operationally different on the B 900/CP9500 from other implementations. These differences are a result of the different hardware features involved. The items covered include:

B 900/CP9500 Control Panel
System Initialization Concepts
System Startup
Coldstart (Attended and Unattended Operation)
Warmstart (Attended and Unattended Operation)
Read Only Memory (ROM) Dump Routine
Re-Label (RL)
Release Level Display (RLD)
Reset Option (RO)
Set Option (SO)
Sysanalyzer
Analyzer
Romanalyzer
Romconvert
Field Patch Program (FPP)
Configurer

B 900/CP9500 CONTROL PANEL

Located on the top front surface of the B 900/CP9500 are the following controls and indicators (left to right):

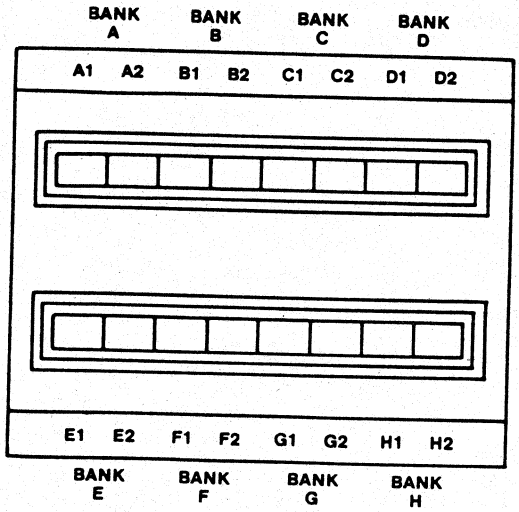


The hexadecimal keypad is not used during normal system operation.

The keypad is used by the field engineer in running Maintenance Test Routines (MTRs). It can also be used to execute the Read Only Memory (ROM) Dump Routine. This routine allows the operator to dump the contents of system memory when the MCP is unable to produce a system dump file.

Hexadecimal Display Lights

The indicator display is used to report messages and errors to the user. At any one time, the system can display up to 16 hexadecimal digits. The display is divided into banks, A through H, with two hexadecimal digits per bank.

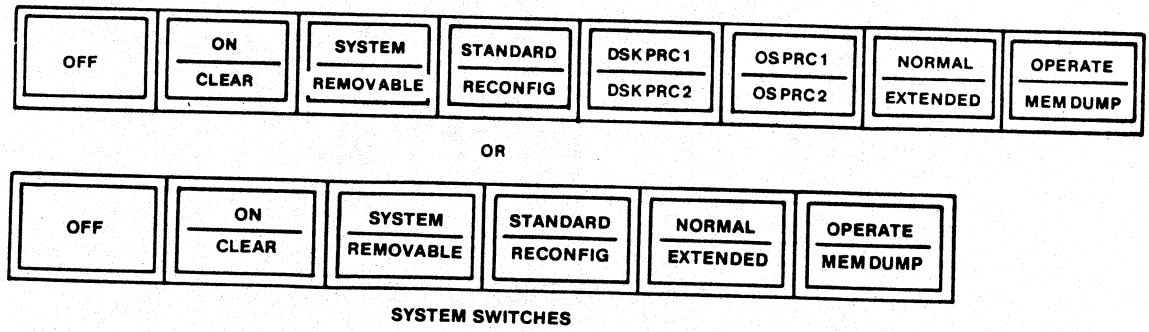


HEXADECIMAL DISPLAY
(2 ROWS OF 8 DIGITS)

NOTE: ANY BANK (BANK N)
CONTAINS 2 HEX DIGITS.
N1 = MOST-SIGNIFICANT DIGIT.
N2 = LEAST-SIGNIFICANT DIGIT.

System Switches

Two styles of switches are available depending on the system type.



SYSTEM SWITCHES

ED2143b

The use of these switches are discussed in the following sections on system startup, coldstart, warmstart, and processor switches.

Data Communications Transmit and Receive Indicators

These four pairs of lights are used by the B 900/CP9500 to indiciate when it is transmitting to or receiving from a terminal on a particular data communications line.

Data Communications Group Select Switch

This seven-position thumbwheel switch is used to select a four line group of data communication lines for monitoring on the four pairs of transmit and receive indicators.

SYSTEM INITIALIZATION CONCEPTS

Before user programs can be executed on the B 900/CP9500 system, two things must be present.

1. The *system disk* must be on line and ready, and must contain those files necessary to support the functions of the Master Control Program (MCP).
2. The system must be warmstarted to load the MCP and system files necessary for operation. The MCP is the program which, once loaded, continues to run as long as the system remains in operation (refer to the CMS MCP Reference Manual, form 2007555).

The functions necessary to create the system disk are provided by the stand-alone utility, coldstart. The function of loading the MCP and the system files are provided by the stand-alone utility, warmstart. The system disk does not need to be created every time the B 900/CP9500 system is powered on. The normal procedure is to power on the system and warmstart to load the MCP into the system.

Execution of the coldstart or warmstart utility can be selected by the operator as a part of the system startup procedure. (Details of coldstart, warmstart, and system startup follow.)

SYSTEM STARTUP

If the B 900/CP9500 is currently powered off, pressing the ON/CLEAR switch provides power to the system.

NOTE

It is strongly recommmended that all disk units be powered off before applying power to the B 900/CP9500. This requirement can be met if the operator adopts the following procedure for powering off the B 900/CP9500:

1. Use the CMS intrinsic PO (power off a disk) to logically power off the disk units and wait for the message from the MCP indicating the successful PO.

2. Physically power off the disks according to the procedures described in the B 900/CP9500 Systems Operators Manual, form 1118270.

After power is applied to the B 900/CP9500, the disk units may be powered on. With power applied to the B 900/CP9500, pressing the ON/CLEAR switch puts the system in startup mode. Successive stand-alone software then takes the system to a state where the MCP can begin to operate. Startup software consists of the following programs:

1. System Startup Read-Only Memory (ROM).
2. CMS Track Zero Bootstrap (SYSINITBOOT).
3. System Dependent Bootstrap (SYSBOOTSTRAP).

As soon as the system enters startup mode, the startup ROM routine (permanently resident in the system) takes control. It then looks for a disk that contains SYSINITBOOT. The startup ROM loads the SYSINITBOOT file, then starts executing the SYSINITBOOT routine (that is, the CMS track zero bootstrap).

The task of SYSINITBOOT is to find a disk that contains SYSBOOTSTRAP, load SYSBOOTSTRAP, and start its execution.

NOTE

Once SYSBOOTSTRAP starts to execute, its action depends upon the setting of the SYSTEM/REMOVABLE switch. If this switch is set to SYSTEM, the SYSBOOTSTRAP program loads the warmstart utility. If the switch is set to REMOVABLE, it loads the coldstart utility (from the disk labelled B 900RL1).

The operator is therefore responsible for selecting whether the system coldstarts or warmstarts when it is put in startup mode by appropriately setting the SYSTEM/REMOVABLE switch.

During startup, bank A of the hexadecimal display lights indicates which startup software is in operation. Within each program's range of values, bank A indicates the program's successful progress, or an error that it may have encountered. To identify the startup program currently in control, use the following table:

Value Range (Bank A)	Program Executing
@00@ – @1F@	SYSTEM STARTUP ROM
@20@ – @2F@	SYSINITBOOT
@30@ – @3F@	SYSBOOTSTRAP
@40@ – @9F@	SYSCOLDSTART
@A0@ – @FF@	SYSWARMSTART

SYSINITBOOT

SYSINITBOOT (the track zero bootstrap) receives control directly from the startup ROM routine. Its responsibility is to locate, load, and pass control to the SYSBOOTSTRAP program. It searches the various disks on the system for the SYSBOOTSTRAP disk. This disk must be ready, have a valid CMS disk label, and contain the SYSBOOTSTRAP file.

The search is done from the highest I/O channel to the lowest, each channel being searched from its lowest disk drive to its highest. Bank A indicates the primary status of the execution of SYSINITBOOT. The values shown in the following table are displayed in bank A under normal execution and termination. For more details on these messages, refer to the Startup Display Reference Tables.

Bank A	Bank B	Bank C	Bank D	Bank E	Bank F	Bank G	Bank H
@20@	---	CNNL/DRV	DISKID	---	---	---	---
@2F@	---	---	---	---	---	---	---

Bank A

Value	Meaning
@20@	SYSINITBOOT is executing normally and is performing a search for the SYSBOOTSTRAP disk.
@2F@	SYSINITBOOT has found a SYSBOOTSTRAP disk. The SYSBOOTSTRAP file has been loaded successfully, and control is about to be passed to the SYSBOOTSTRAP program.

Any value other than 20 or 2F appearing in bank A indicates that SYSINITBOOT has encountered an error condition. For a decode of error codes, refer to the SYSINITBOOT and SYSBOOTSTRAP Error Codes and Startup Display Reference Tables.

SYSBOOTSTRAP

SYSBOOTSTRAP is loaded by and receives control from SYSINITBOOT, and is responsible for:

1. Checking the system's random access memory (RAM).

2. Locating the disk which contains the coldstart/warmstart utilities and the system software file, SYSDSCP (disk processor code).
3. Loading either coldstart or warmstart (depending on the setting of SYSTEM/REMOVABLE switch) and SYSDSCP into the B 900/CP9500.
4. Passing control to coldstart or warmstart.

The coldstart/warmstart is a disk which is found by SYSBOOTSTRAP to be ready, to have a valid CMS disk label, and to contain the required system files.

For coldstart, the required system files are SYSCOLDSTART and SYSDSCP. In addition, the disk must have a packid (that is, the disk name of B900RL1.)

For warmstarting, the required system files are SYSWARMSTART and SYSDSCP. If fixed disk exists on the system, the two files (SYSDSCP and SYSWARMSTART) need not reside on the same fixed disk drive, as long as both are on fixed disk. When no fixed disk is present, both required files must reside on the same removable disk.

SYSBOOTSTRAP uses the same method in searching for the coldstart/ warmstart disk as was used to find the SYSBOOTSTRAP disk; the channels are searched highest to lowest, each channel being searched from its lowest disk device to its highest.

During the SYSBOOTSTRAP phase of system startup, banks A and E are used to indicate the status of SYSBOOTSTRAP. Both banks should be interrogated to determine whether SYSBOOTSTRAP is executing normally or has encountered an error condition.

The values shown in the following table are displayed in bank A under normal execution and termination of SYSBOOTSTRAP. For more details on these messages, refer to the Startup Display Reference Tables.

Bank A	Bank B	Bank C	Bank D	Bank E	Bank F	Bank G	Bank H
@30@	RELOC ID	---	---	RELOC RSLT	---	---	---
@31@	MEM ID	BUS/ PAGE	---	CHKOUT RSLT	---	---	---
@32@	FILE INDEX	CHNL/ DRV	DISK ID	SRCH RSLT	---	---	---
@33@	PROC ID	CHNL/ DRV	DISK ID	LOAD RSLT	---	---	---
@3F@	---	---	---	ERR #	---	---	---

Bank A

Value	Meaning
@30@	SYSBOOTSTRAP has gained control from SYSINITBOOT and is in the process of performing its startup activities.
@31@	SYSBOOTSTRAP is performing its memory checkout procedure on the necessary system RAM memories.
@32@	SYSBOOTSTRAP is performing a file-by-file search in locating the coldstart/warmstart disk.

Value	Meaning
@33@	SYSBOOTSTRAP is loading coldstart/warmstart system firmware.
@3F@	SYSBOOTSTRAP has completed execution.

NOTE

If bank E is blank, then SYSBOOTSTRAP has terminated normally. Otherwise, bank E contains an error number indicating abnormal termination.

For a decode of SYSBOOTSTRAP error codes, refer to the SYSINITBOOT and SYSBOOTSTRAP Error Codes and Startup Display Reference Tables.

SYSINITBOOT and SYSBOOTSTRAP Error Codes

The following tables show all error conditions that SYSINITBOOT and SYSBOOTSTRAP may encounter. Beneath each table is a brief description of the error code found in bank A. The Startup Display Reference Tables may also be used to decode the codes found in banks B through H.

SYSINITBOOT Error Codes

Bank A	Bank B	Bank C	Bank D	Bank E	Bank F	Bank G	Bank H
@21@–@28@	OP	CHNL/DRV	DISK ID	S0	S1	S2	S3
@2E@	—	———	———	—	—	—	—

ERROR CODE	POSSIBLE CAUSES	SUGGESTED ACTION
@21@	SYSINITBOOT has requested that an undefined disk operation be executed.	Verify that proper release level of B900RL1 disk is being used. Try another copy of the release disk. If this does not work, request technical assistance.
@22@	SYSINITBOOT requested an invalid disk operation length.	Same as above.
@23@	The disk drive at the indicated channel (bank "C") cannot be supported by the SYSINITBOOT software.	Same as above.
@24@	The indicated channel (bank "C") cannot be supported by the disk processor.	Same as above.
@25@	The disk drive indicated in bank "C" is not ready.	Ready the indicated drive and press the CLEAR switch to reinitiate system startup.
@26@	A disk device error was encountered on the channel and drive found in bank "C".	Request technical assistance.

ERROR CODE	POSSIBLE CAUSES	SUGGESTED ACTION
@27@	A key search failure has been encoutnered on the channel and drive found in bank "C."	Request technical assistance.
@28@	The disk drive indicated in bank "C" does not exist.	Request technical assistance.
@2E@	No disk drive meeting the requirements for a SYSBOOTSTRAP disk has been found on the system (NOTE: A device error may have been detected on an otherwise valid disk.)	Ensure that the required disk drive is ready. If it is, request technical assistance.

SYSBOOTSTRAP Error Codes

Bank A	Bank B	Bank C	Bank D	Bank E	Bank F	Bank G	Bank H
@3B@	REQ	———	———	———	———	———	———
@3C@	OP	CHNL/ DRV	DISK ID	DKCTL RSLT	———	———	———
@3D@	OP	CHNL/ DRV	DISK ID	S0	S1	S2	S3
@3E@	ERR STAT	OFF 1		XOFF		OFF 2	

ERROR CODE	POSSIBLE CAUSES	SUGGESTED ACTION
@3B@	SYSBOOTSTRAP has been sent to its force state handler. This mechanism is to be disabled during SYSBOOTSTRAP execution and its occurrence is a fatal error.	Verify that proper release level of B900RL1 disk is being used. If it is, request technical assistance.
@3C@	SYSBOOTSTRAP detected a disk exception condition indicating that an invalid parameter was passed.	Same as above.
@3D@	A device error was detected by the SYSINITBOOT software disk controller.	Request technical assistance.
@3E@	SYSBOOTSTRAP has encountered a hardware detected error while executing.	Request technical assistance.

Startup Display Reference Tables

The following table defines the identifiers found in banks B through H in the error tables described previously. When decoding these values, the following tables must (where applicable) be used in addition to the text found with the error condition.

IDENTIFIER	DISPLAY BANK(S)	MEANING
BUS/ PAGE	C	The current RAM memory location where the most significant digit is the physical bus address being accessed and the least significant digit is the page address on bus being accessed.
CHKOUT RSLT	E	The result of a SYSBOOTSTRAP RAM memory check, where a blank indicates no error.
CHNL/DRV	C	The current drive and I/O channel where the most significant digit of display bank "C" is the current channel being accessed and the least significant digit of display bank "C" is the current drive on that channel.

The table below maps a unique "address literal" to each available channel. The "address literal" is used by the disk processor when referencing a channel.

	Channel Index (hexadecimal)	Address Literal
IOS and NON-IOS SYSTEMS	0	@01@
	1	@02@
	2	@04@
	3	@08@
	4	@10@
	5	@20@
	6	@40@
NON-IOS ONLY	7	@80@
	8	@30@
	9	@50@
IOS ONLY	A	@60@
	B	@70@
	C	@90@
	D	@A0@
	E	@B0@

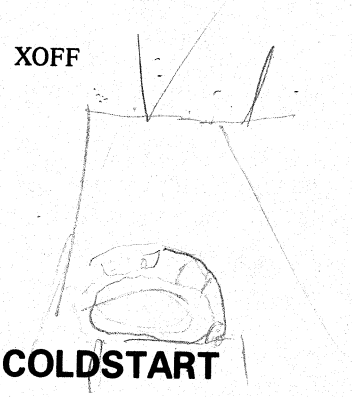
The "IOS" and "NON-IOS" notations indicate how the presence or absence of the "Input/Output Select" option affects a system's ability to address a given I/O channel index. (For example, a system without IOS cannot address channels 8-E).

DISK ID	D	The type of device being accessed. The table below is ordered so that faster devices appear before the slower.

ID	DEVICE	
@54@	9493-80	Fixed Disk
@53@	9493-20	Fixed Disk
@6F@	9493-18	Fixed Disk
@6C@	9481-12	Cartridge Disk
@6A@	9480-12	Cartridge Disk
@6E@	9480-22	Cartridge Disk
@55@	9489-21	3/6 BSMD
@56@	9489-23	3/6 BSMD
@68@	9489-1/11	BSMD
@60@	9489-12	BSMD

IDENTIFIER	DISPLAY BANK(S)	MEANING
DKCTL RSLT	E	SYSBOOTSTRAP has terminated due to an undefined disk operation. The identifier, DKCTL RSLT, may contain @21@–@24@. These codes are explained in the section on SYSINITBOOT and SYSBOOTSTRAP Error Codes where they appear in bank A.
ERR#	E	The code displayed in bank "E" indicates the termination of SYSBOOTSTRAP, where: @00@ – No error has occurred, control has been passed to COLDSTART or WARMSTART depending on the setting of the SYSTEM/REMOVABLE switch. @01@ – No Operating System (OS) processor was found on the system. @02@ – A COLDSTART/WARMSTART disk was not found on the system. @03@ – SYSBOOTSTRAP is not able to put a "processor freeze" on itself.
ERR STAT	B	Processor I.C. error status byte which may indicate that: 1) a micro access is in progress; or 2) a data access is in progress. To determine whether a data or micro access is in progress, refer to the "STATUS" identifier found in bank "E" in the ROM Dump Display Reference Table.
FILE INDEX	B	An Identifier of @00@ indicating that the SYSCOLDSTART or SYSWARMSTART file is being searched for.
LOAD RSLT	E	The status of loading SYSWARMSTART, SYSCOLDSTART and SYSDSCP firmware into memory, where: @00@ – System file load in progress. @01@ – System file load complete, no error.
MEM ID	B	An identifier indicating which part of RAM memory is being checked, where: @00@ – SYSBOOTSTRAP/SYSDSCP memory. @01@ – SYSCOLDSTART/SYSWARMSTART memory. @02@ – COLDSTART/WARMSTART buffer memory.
OFF1	C,D	If the "ERR STAT" code found in bank "B" indicates that a micro access is in progress, OFF1 is the micro program offset (UMR) where the error occurred. or If the "ERR STAT" code found in bank "B" indicates a data access is in progress then "OFF1" and "OFF2" indicate the contents of the M1 and M2 registers of the processor, respectively.
OFF2	G,H	Used in conjunction with "OFF1" to indicate the contents of the M1 and M2 registers of the processor when "ERR STAT" indicates that a data access is in progress.
OP	B	Current disk operator: @80@ – Read @20@ – Key Search @10@ – Read Drive I.D.

IDENTIFIER	DISPLAY BANK(S)	MEANING
PROC ID	B	The identifier used by SYSBOOTSTRAP to identify what firmware is being loaded, where: @00@ – SYSCOLDSTART or SYSWARMSTART load. @01@ – SYSDSCP load.
RELOC ID	B	An identifier used to determine the status of the SYSBOOTSTRAP relocation activities: @00@ – SYSBOOTSTRAP is relocating itself to its executing offset. @01@ – SYSINITBOOT software disk controller is being relocated to its SYSBOOTSTRAP area.
RELOC RSLT	E	The result of relocating SYSBOOTSTRAP or SYSINITBOOT software disk controller to an executing offset. A blank indicates that there is no error.
REQ	B	The values of the processor's I/O channel's request lines.
SRCH RSLT	E	The status or result of the search for the COLDSTART or WARMSTART disk, where: @00@ – Indicates that the search is in progress. @01@ – Indicates that the search has been completed successfully. @02@ – Indicates that the search for the COLDSTART or WARMSTART files has been completed unsuccessfully.
SO-S3	E-H	Identifiers which indicate the status of the hardware disk controller: For Processor disk controllers: S0 = Primary Status S1 = Secondary Status For Host controllers: S0 = Host controller status S1-S3 = Device controller status controllers.
XOFF	E,F	If the "ERR STAT" code found in bank "B" indicates a micro access is in progress, then XOFF indicates the extended micro program offset (UMRX) where the error occurred. or If the "ERR STAT" code found in bank "B" indicates a data access is in progress then XOFF is the extended data access offset (MAX) where the error occurred.



COLDSTART

Coldstart is a stand-alone utility whose primary function is to create a system disk. Once the system disk has been initialized via coldstart, system files may then be loaded from the release disk(s). The system releases may be contained on either cartridge or mini disks that are labeled sequentially with a numeric suffix of 1 for the first release disk, 2 for the second, and so forth. Coldstart is loaded from the first release disk (mini or cartridge disk) labeled B900RL1, located in channel 3.

On systems with an operator display terminal (ODT) device (that is, in operator- attended mode), coldstart displays information to, and requests information from, the operator. On systems without an ODT device, coldstart becomes user-transparent. No information is requested from, or displayed to, the user except via the indicator lights. When necessary, information is derived from default values.

Operator Attended Mode

Operator attended coldstart is comprised of the following functions:

FE = INITIALIZE MTR DISK
HE = HELP
IN = INITIALIZE A DISK
LD = LOAD FILES FROM RELEASE DISK
OL = LIST STATUS OF DRIVES
PT = PATCH SYSTEM FILES
RF = REFORMAT DISK TO INITIAL STATE
RP = REPLACE SYSTEM FILES
WS = WARMSTART

Idle, DS, Resolving Duplicate Packids

The idle, DS, and resolving duplicate packid's features are common to more than one function. These features are described as follows.

Idle Loop

The idle loop is that state of coldstart during which the operator is expected to request a function. Coldstart signals that it is in the idle loop with the message: ENTER FUNCTION.

Cancelling a Coldstart Function (The ?DS Option)

The operator may abort any of the coldstart functions by entering ?DS in place of any response that is expected to contain three or more characters, except when entering the date. When the ?DS option is used, coldstart aborts the function that is in progress, and returns to the idle loop after issuing the message: FUNCTION DS-ED.

Resolving Duplicate Packids

Coldstart is able to resolve situations where one or more packids are not unique to a specific removable disk. Whenever a coldstart function requests a packid, the following sequence of events occurs:

1. The coldstart function issues the message: ENTER OBJECT PACKID <1-7 CHARS>.
2. The operator enters a packid containing from one to seven letters and/or digits. (Example: DUPNAM1.)
3. If there is more than one disk currently on-line with this name, coldstart returns a list of those disk drives with the specified packid.
   SOURCE DISK: DUPNAM1
   DUPNAM1 IS ON DKA
   DUPNAM1 IS ON DME
   Disk drive mnemonics indicate both disk type, (DK for cartridge or DM for mini) and the physical device (A-H).
4. If the packid specified by the operator is unique, the pack sequence ends at step 2 and the particular coldstart function continues. If the packid is not unique, the operator is requested to identify the drive on which the desired disk resides.

   ENTER DRIVE <3 CHARACTERS>

5. The operator then enters the three-character disk drive mnemonic indicating which one of the duplicate named disks coldstart should use. The first two characters identify the disk type; the third identifies the physical drive. The removable disk type mnemonics are: DKx - cartridge and DMx - mini-disk.

The operator's response may not include blanks. The physical drive identifier, (X), is a single letter between A and H, inclusive. Sample responses are: DKB and DMH.

Initiation of Coldstart

To initiate coldstart on systems with an ODT device, the following procedures should be followed.

1. Turn system power on (refer to System Startup).
2. Insert first release disk labeled B900RL1.
3. Set processor's SYSTEM/REMOVABLE switch to REMOVABLE. The remaining switches should be in their primary positions (that is, STANDARD, NORMAL, OPERATE, DSK PCR1, and OS PRC1).
4. Press the system ON/CLEAR button.

Coldstart begins by displaying the following message: BOJ COLDSTART -REV.XX.YY.ZZ.

Coldstart next requests the date: ENTER DATE <MM/DD/YY>.

The operator enters the date, with slashes between month, day, and year. Leading zeroes must be included. (Example: 06/20/80.)

The status of each disk drive and the packid of its resident disk are reported to the operator, using one of the following:

1. <packid> READY ON <device-mnemonic><drive-id>.
2. <packid> NOT READY ON <device-mnemonic><drive-id>.
3. <packid> WRITE INHIBITED ON <device-mnemonic><drive-id>.

The <device-mnemonic> reported identifies the type of drive: DF for fixed disk, DK for cartridge disk, or DM for mini-disk. The <drive-id> is a letter, in the range A through H, identifying the physical unit used. It occurs immediately after the <device-mnemonic> with no intervening spaces.

If all drives do not have a ready message reported, coldstart issues the following message: TYPE "A" WHEN DISKS ARE READY. The operator enters an A when the disk drives are in the states desired.

NOTE

If one or more disks are still NOT READY or WRITE INHIBITED, it makes no difference. This message (and the expected response) gives the user an opportunity to change the status of one or more devices before coldstart functions are used.

If anything other than an A is entered, an <ERR> message is returned and the "TYPE A WHEN..." message is repeated. Coldstart then responds with the message: STORING SYSINITBOOT IN MEMORY.

After successfully loading SYSINITBOOT in memory, coldstart identifies the source disk of the SYSINITBOOT file and its disk mnemonic with the following messages:

SOURCE DISK: B900RL1
B900RL1 IS ON <device mnemonic>

If the first release disk is not on-line when coldstart attempts to load the SYSINITBOOT file, the following message is displayed:

ENTER RELEASE DISK
TYPE A TO CONTINUE, B TO STOP

Coldstart now issues the following messages:

ENTER HELP FOR LIST OF COMMANDS
ENTER FUNCTION

If the operator enters HELP, coldstart presents the user with the available options and their mnemonics, as follows:

FE = INITIALIZE MTR DISK
IN = INITIALIZE A DISK

LD = LOAD FILES FROM RELEASEDISK
OL = LIST STATUS OF DRIVES
PT = PATCH SYSTEM FILES
RF = REFORMAT DISK TO INITIAL STATE
RP = REPLACE SYSTEM FILES
WS = WARMSTART

Otherwise, the operator chooses a function by entering an appropriate mnemonic on the keyboard. Coldstart indicates it has completed one function and is ready for a new function with the message:

ENTER FUNCTION.

FE (Initialize MTR Disk)

This function initializes a disk for MTR usage. Old disk contents are deleted, the disk surface is qualified, and a flag in the disk label is set to indicate that it is an MTR disk.

Six specific tracks on any given MTR disk are reserved for use by MTR routines. This is done by setting bad sector indicators in the disk's non-file (available space) directory. If these six tracks are actually found to be bad at any location, MTR initialize terminates. Success of this function means that those six tracks are flawless and unconditionally reserved for MTR usage.

The first request issued by the FE function is for identification of the MTR disk's drive. The following message appears: ENTER DRIVE <3 CHARACTERS>. Respond with a three-letter drive mnemonic. (Example: DMF denotes that the MTR disk is a mini disk, residing on drive F.) A fixed disk (mnemonic DFx) cannot be used as the MTR disk. Only the mnemonics DKx and DMx, cartridge and mini respectively, are valid.

The disk surface is now qualified. Bad sectors within the six tracks reserved for usage by MTR causes failure of this function, and the message: BAD MTR TRACK. Other tracks, which are to contain MTR software, are allowed a total of 50 bad sectors. The number is reported to the user: NO OF BAD SECTORS = ##.

The user is now offered the option of using default mode. The following message is issued: DEFAULT MODE <Y OR N>.

If default mode is selected, the removable-disk defaults are used to complete the MTR disk's initialization. Refer to Default Assignments listed under Coldstart Messages.

If default mode is not used, MTR initialize continues to request information. (Refer to Initialize function for more details.) When FE initialize ends successfully, the following message is displayed: END INITIALIZE MTR DISK.

HE (Help)

The operator may request, via this function, a short definition and the necessary mnemonic for every coldstart option.

FE = INITIALIZE MTR DISK
IN = INITIALIZE A DISK
LD = LOAD FAILURE FROM RELEASE DISK
OL = LIST STATUS OF DRIVES
PT = PATCH SYSTEM FILES
RF = REFORMAT TO INITIAL STATE
RP = REPLACE SYSTEM FILES
WS = WARMSTART

IN (Initialize)

This function initializes a system disk according to CMS format. There are three steps to this function: 1) requesting of information; 2) qualification of the recording surface; and 3) creation of CMS structures on the disk.

The first request by initialize is for the type of disk: fixed or removable. Removable disks include both cartridge disks and Burroughs Super Mini-Disks (BSM). The system request is as follows: TYPE OF DISK <REM OR FIX>. The operator can specify only REM or FIX.

If the operator specifies fixed disk, all fixed disk drives found powered on and ready are initialized as a single assemblage. If a removable disk is specified, coldstart requests that the disk be identified by issuing the message: DRIVE <3 CHARACTERS>.

After the disk has been identified by the operator response of a three letter drive mnemonic, initialize tests its recording surface by writing and reading test patterns. Any bad sectors found are deleted from the disk's addressable space. The count of bad sectors is reported to the operator before processing: NO. OF BAD SECTORS = ##.

NOTE

When initializing two fixed disks, the "NO. OF BAD SECTORS = ##" are reported to the operator for both drives.

Bad sectors may cause initialize to terminate. In unattended mode, this causes coldstart to terminate. There are two ways this can occur:

1. The disk contains more than 50 bad sectors. This is indicated to the operator: TOO MANY BAD SECTORS.
2. There is a bad sector in track zero. This is indicated to the operator: TRACK ZERO HAS BAD SECTOR.

Initialize now asks the operator whether default mode is to be used. Default mode is similar to operator-unattended mode, in that initialize derives the rest of its information from default values. The format of the query is: DEFAULT MODE <Y OR N>. If Y is selected, (that is, default mode) all further information needed by initialize is derived from the default values. Nothing further is requested from the operator. Refer to the default assignments listed under Coldstart Messages.

If N is selected, initialize generates the disk's directory and label using the information supplied to the following queries:

1. ENTER FILES <1-2804>. Enter the number of files to be allowed on the disk.

NOTE

When initializing two fixed disks, the statement ENTER FILES <1-2804> is displayed once for the entire assemblage. Enter the number of files to be allowed. Also, each assemblage must be initialized with the same number of files.

2. ENTER OBJECT PACKID < 1-7 CHARS>. Enter the disk-id which can be from one to seven alphanumeric characters.

NOTE

When initializing two fixed disks, the statement ENTER OBJECT PACKID <1-7 CHARS> is duplicated for the second spindle.

3. ENTER SERIAL <6 DIGITS> (*). Enter from one to six numeric characters as the disk serial number.
4. ENTER OWNER <1-14 CHARS> (*). Enter the name of the owner of the disk which can be from one to fourteen alphanumeric characters.

(An * appearing next to a message indicates that the message is not displayed when initializing fixed disk.)

When initialize terminates successfully, it issues the message: END INITIALIZE.

A disk is initialized according to CMS formats consisting of the following:

1. Disk label.
2. Directory:
   1) Non-file directory.
   2) File name list.
   3) Disk file headers.

3. SYSMEM file.
4. PPIT Table (Fixed disks only) *.

* 201-I Disk - 211 available slots for user packs.
211 Disk - 211 available slots for user packs.

Increasing Fixed Disk

All fixed disk drives on a B 900/CP9500 that are powered on and ready are initialized as a single assemblage. To add one or more drives into this assemblage, perform the following when running in coldstart mode:

1. Power off all fixed disk drives that are presently running. This is to protect the files on these units from erasure by the initialize function.
2. Power on the fixed disk drive(s) that are being added to the assemblage.
3. Use the initialize function to initialize the drive(s) to be added. Specify the same number of files as those specified on the original assemblage. A unique label will be requested for each drive initialized.
4. Restore power to all units.
5. Rewarmstart the system by following the warmstart procedure.

The MCP automatically adds the new drives to the fixed disk assemblage once the system has been warmstarted.

NOTE

When adding one or more drives into an existing fixed disk assemblage, the additional drives must be placed in an available slot or channel which is higher than any other channel which is being used or which was used.

When replacing a disk on the system, the new disk must be cabled into the same channel from which the old disk was removed. It must also be given the same packid and initialization parameters.

LD (Load)

This function loads the files from the release disks to the system disk. The release disks are labeled sequentially (B900RL1, B900RL2, and so forth) and must be inserted in the drive according to the sequence of their labels.

The following files must reside on the first release disk:

SYSBOOTSTRAP.
SYSCOLDSTART.
SYSINITBOOT.
SYSDSCP.

The remaining files may reside on any release disk, including the first one.

SYSMCP.
SYSWARMSTART.
SYSICP.
SYSMPLII.
SYSCONFIG.
SYSLANGUAGE.
COPY.
SYSTRANSLATE.
SYS-SUPERUTL.
OPTIONAL system files:
  SYSCOBOL.
  NDLSYS.
  NPC900.
  SORTINTRINS.

Dump analyzers:
SYSANALYZER.
ANALYZER.
Any user file.

LD only loads to a initialized disk (that is, a disk containing a SYSMEM file entry) from a release disk and first requests the disk type and packid of the system disk:

TYPE OF DISK <REM OR FIX>
ENTER OBJECT PACKID <1-7 CHARS>

All files on the first release disk are loaded and the name of each file is listed as it is loaded. LD then requests that the next release disk be inserted for loading of more files by issuing the following message:

ENTER NEXT RELEASE DISK
TYPE A TO CONTINUE, B TO STOP

To enter more files, insert the next release disk and type A to continue. If all files have been entered, stop by entering B. The stop command initiates a check for presence of the following system files:

SYSLANGUAGE
SYSMCP
SYSDSCP
SYSICP
SYSBOOTSTRAP
SYSWARMSTART
SYSMPLII
SYSTRANSLATE
SYSCOBOL
SYSCONFIG
SYS-SUPERUTL
COPY

The following message is issued if the check fails: SYSTEM LOAD INCOMPLETE. An incomplete load does not return the disk to an initial state (that is, as it was immediately after initialize); any files loaded remain on disk. The operator is notified when the LD function terminates successfully via the message: END LOAD.

OL (Disks On Line)

The OL function lists all ready disks that are on the system. Each disk is identified by device mnemonic, physical drive, and packid.

Examples

DMB ACCOUNT. A Burroughs Super Mini Disk named ACCOUNT is ready on on drive B.

DFC THISDSK. A fixed disk named THISDSK is ready on drive C.

DKG PACK123. A cartridge disk named PACK123 is ready on drive G.

END DISKS ON LINE.

PT (Patch System Files)

System files are patched by replacing a given system file on the system disk with its counterpart from a removable patch disk (cartridge or mini). The removable disk must be a release disk.

The following files can be patched:

SYSCOBOL
SYSMCP

SYSICP
SYSDSCP
SYSWARMSTART
SYSMPLII
SYS-SUPERUTL
SYSCONFIG
SYSTRANSLATE
SYSBOOTSTRAP

The patch function derives its input from a patch disk, but the disk to be patched must be identified. First, patch requests the disk's type: TYPE OF DISK <REM OR FIX>. Next, if a removable disk is being patched, patch requests its packid: ENTER OBJECT PACKID <1-7 CHARS>.

Patching Restrictions

Each system file and each patch file has a version string associated with it. The format of this string is RRRRPP, where the RRRR field represents the release level and the PP field represents the patch level.

The version string for each file on the patch disk is checked against its corresponding system file's version string before any loading of patch files occurs. One of the following actions results:

1. The release level field of the version string in the patch file and system file must be equal. If this is not true, the patch function terminates immediately.
2. If the patch file has a patch level which is less than the patch level of the corresponding system file, patch terminates and the operator is notified: VERSION MISMATCH.
3. If the patch level of a patch file is equal to the patch level of the corresponding system file, no error results but the patch file is ignored and is not loaded.
4. Any patch file having a patch level greater than the patch level of the corresponding system file is loaded.

Once all patch file version strings have been checked against all system file version strings, patch copies all patch files to the disk being patched. If any error occurs, the patch function terminates and whatever patch files have been copied are deleted. This returns the disk being patched to its original state. If all patch files are copied without error, the corresponding old system files are deleted. This completes the patch function. The operator is notified when the patch function terminates successfully via the message: END PATCH.

RF (Reformat)

This coldstart function returns the system disk to its initial state (that is, as it was immediately after initialize). Reformat, unlike initialize, does not check the integrity of the recording surface by writing test patterns; therefore, the number of bad sectors are not affected by reformat.

First, reformat requests the type and packid of the disk to be reformatted by issuing the following messages:

TYPE OF DISK <REM OR FIX>

If REM is entered, the following message is issued:

ENTER OBJECT PACKID <1-7 CHARS>

Once reformat identifies the disk to be accessed, its label is accessed and used to print the following information:

NO. OF BAD SECTORS <#>
DATE <date of last initialize or reformat>
FILES <maximum number of files presently allowed>
SERIAL <serial number of disk> (*)
PACKID <packid of disk>
OWNER <owner of disk> *

If FIX is entered, the following messages are issued:

NO. OF BAD SECTORS <#> (*)
PACKID <packid of disk> (*)
FILES <maximum number of files presently allowed>

The messages marked with " *" indicate that the message will be duplicated for each spindle in the assemblage.

The operator is now offered the option of using default mode (refer to Default Assignments) by the message: DEFAULT MODE <Y OR N>. If N is selected, reformat generates the disk's directory and label using the information supplied to the following queries:

1. ENTER FILES <1-2804>. Enter the number of files to be allowed on the disk.

NOTE

When reformating two fixed disks, the statement ENTER FILES <1-2804> is displayed once for the entire assemblage. Enter the number of files to be allowed. Also, each assemblage must be initialized with the same number of files.

2. ENTER OBJECT PACKID < 1-7 CHARS>. Enter the diskid which can be from one to seven alphanumeric characters.

NOTE

When reformating two fixed disks, the statement ENTER OBJECT PACKID <1-7 CHARS> is duplicated for the second spindle.

3. ENTER SERIAL <6 DIGITS> (*). Enter from one to six numeric characters as the disk serial number.
4. ENTER OWNER <1-14 CHARS> (*). Enter the name of the owner of the disk which can be from one to fourteen alphanumeric characters.

An " *" appearing next to a message indicates that the message is only displayed when reformating a removable disk.

When RF terminates, it issues the message: END REFORMAT.

RP (Replace System Files)

This function is used to replace certain system files on a previously created disk without version string checks being done, as in the patch function.

The following system files can be replaced:

SYSBOOTSTRAP
SYSDSCP
SYSMCP
SYSWARMSTART
SYSICP
SYSMPLII
SYSTRANSLATE
SYS-SUPERUTL
SYSCOBOL
SYSCONFIG

The replace function obtains its inputs from the release disk. The disk to be updated must be specified. First, replace requests the disk's type: TYPE OF DISK <REM OR FIX>.

If the disk type specified is removable, replace requests the packid of the disk to be updated: ENTER OBJECT PACKID <1-7 CHARS>.

The replace function uses the system files (listed previously) from the release disk to replace the copies of these files on the specified disk. If any error occurs, the replace function terminates, and whatever replacement files have been loaded, are deleted. This returns the disk on which the replace was being performed to its original state.

The user is notified when the replace function finishes successfully via the message: END REPLACE FUNCTION.

WS (Warmstart)

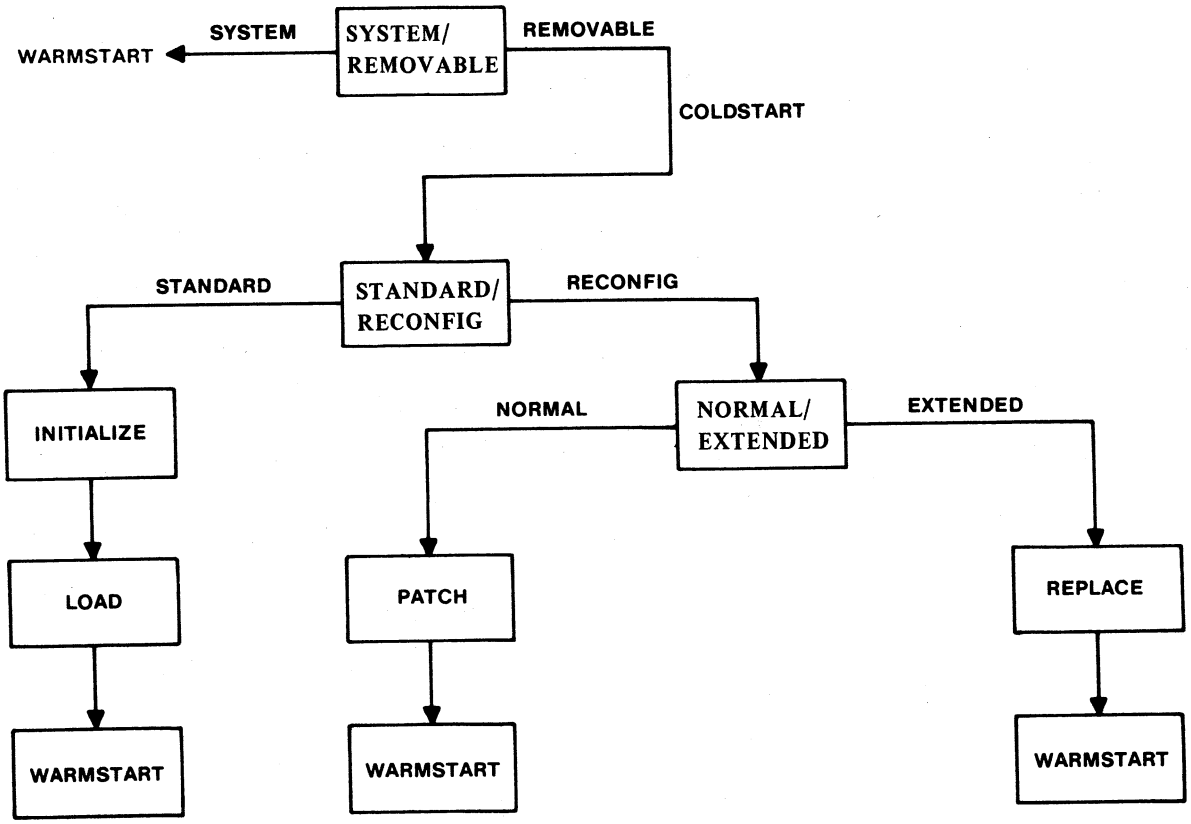
This coldstart function allows the operator to enter the warmstart process directly from coldstart. This function begins by alerting the operator to set the SYSTEM/REMOVABLE switch to SYSTEM by issuing the message: SET SYSTEM/REM SWITCH TO SYSTEM. Once the switch is set, control is passed to the system startup procedures. These startup procedures initiate execution of the warmstart utility.

Operator Unattended Mode

The system automatically enters coldstart unattended mode when the coldstart utility determines that there is not an ODT on the system. In unattended mode, any one of the following series of functions can be executed.

1. Initialize, Load, and Warmstart.
2. Patch and Warmstart.
3. Replace and Warmstart.

The system decides which function, or set of functions, to perform after interrogating the switch settings on the control panel. If the system detects the SYSTEM/REMOVABLE switch to be in the SYSTEM position, warmstart is initiated. If set to REMOVABLE, coldstart examines the STANDARD/RECONFIG and NORMAL/EXTENDED switches for positioning, and executes the appropriate functions. When the functions are completed, coldstart signals the user to set the control panel switches in their primary positions for warmstart to proceed. See the following figure for a summary of switch settings and their corresponding functions. (Please note that before the system can be warmstarted, the switches must be returned to their primary positions.)



ED2599

Initialize, Load, Warmstart

In unattended mode, the initialize/load/warmstart series of functions are initiated when the user puts a release disk on the system and sets the following system switches:

REMOVABLE
STANDARD
CLEAR

The initialize/load functions act upon the fastest speed device that is available to the system. If the user wishes to perform these functions on any device other than the fixed disk, the fixed disk MUST be physically powered down or it is initialized. After the release disk has been inserted and the control panel switches set, the system begins with the initialize function. All the necessary information, such as the number of files allowed, serial number, pack name and owners-id are derived from default values. The user is kept informed of the function's progress via bank D on the display. Bank A normally contains @40@ unless an error occurs. For example, a @55@ in bank A means that the disk has a bad sector in track zero. The user should then get a different disk and initiate the function again. The unattended mode uses the same criteria for qualifying disks as attended mode. If there are too many bad sectors on a disk being initialized, coldstart terminates.

After the initialize function is completed, control is automatically passed to the load function. Bank D contains a @20@ while the load function is in progress. When the load is completed, bank A contains @40@ and bank D contains @50@ as follows:

A	B	C	D
40	—	—	50
—	—	—	—
E	F	G	H

This pattern flashes to indicate that user intervention is needed. At this point, the user has the option of inserting the next release disk to continue the LOAD function, or of setting the SYSTEM/REMOVABLE switch to the SYSTEM position so warmstart can take control. In operator-unattended mode, no check is made for the presence of all required system files (refer to Load under Operator- Attended Mode). If any are absent, warmstart does not operate.

Patch, Warmstart

In unattended mode, patch/warmstart is initiated when the user puts a release disk on the system and sets the following switches:

REMOVABLE
RECONFIG
NORMAL
CLEAR

In deciding which disk to patch, coldstart searches for the fastest device.

As in attended mode, the version string for each patch file is checked against it's corresponding system file's version string before any loading of patch files occurs. If an error occurs, the patch function terminates, displaying an appropriate message on the hexadecimal display lights. For example, a PATCH LEVEL ERROR causes the hexadecimal displays to contain the following:

A	B	C	D
74	—	—	80
—	—	—	—
E	F	G	H

If any error occurs during the patch operation, the entire patch function terminates and whatever files have been loaded are removed. This returns the disk being patched to its original state. Patch flashes the following indicator lights when the function is finished. (These indicator lights mean that warmstart is ready to take control.)

A	B	C	D
40	—	—	50
—	—	—	—
E	F	G	H

The user should:

1. Remove release disk.
2. Set the STANDARD/RECONFIG switch to STANDARD.
3. Set the SYSTEM/REMOVABLE switch to SYSTEM and warmstart begins.

Replace, Warmstart

In unattended mode, the replace/warmstart is initiated when the user puts a release disk on the system and sets the following switches:

REMOVABLE
RECONFIG
EXTENDED
CLEAR

The replace function is used to replace certain system files on a previously created disk. Unlike the patch function, replace does not perform version string checks.

The replace is performed on the fastest speed device. If any load error occurs, the replace function terminates, and whatever replacement files have been loaded are removed. This returns the disk to its original state. This type of error is displayed on Hex display. When the replace function is successful, it terminates by flashing a pattern on the indicator lights shown below.

A	B	C	D
40	—	—	50
—	—	—	—
E	F	G	H

For warmstart to take control, the user should:
1. Remove release disk.
2. Set NORMAL/EXTENDED switch to NORMAL.
3. Set STANDARD/RECONFIG switch to STANDARD.
4. Set SYSTEM/REMOVABLE switch to SYSTEM.

Coldstart Messages

Coldstart primarily uses banks A and D to indicate its status. Bank A is used to display a code which corresponds to coldstart's status or an error message. Bank D contains a code showing what coldstart function is executing. The codes from banks A and D and their meanings are listed in tables 9-1 and 9-2.

Table 9-1. Codes Appearing in Bank A of Indicator Display (Coldstart Status) (Sheet 1)

CODE MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
@40@ None	Normal operation.	None.
@41@ File not found	This error occurs internally to the SYSCOLDSTART program.	Try a RP and/or COPY of required system files.
@42@ Duplicate file-<filename> not loaded	Occurs when loading files to disk. If a file was loaded from a B900RL# disk that has the same file-id as one on a subsequent B900RL# disk, this error occurs.	Warmstart the system and copy the correct file.
@43@ ⟨drive mneumonic⟩ disk controller error	The disk processor has reported to COLDSTART that a controller error has occurred. This is a hardware error in the disk detected by the disk processor.	Request technical assistance.
@44@ Invalid system error number	The number reported to the system error procedure is not a valid COLDSTART error number.	Request technical assistance.
@45@ ⟨drive mneumonic⟩ device error	A device error was reported for the specified disk. This is a hardware detected error.	Request technical assistance.
@46@ Write inhibited	The specified disk is write disabled.	Write enable the disk. Refer to B 900/ CP 9500 Operator's Manual for Instructions, form 1118270.
@47@ Name list full	There are no available entries in the file name list. Therefore, no files can be loaded.	Reformat disk increasing the number of files. (May require use of disk media with greater storage capacity.)
@48@ No user disk	All available disk space on a cartridge has been utilized.	Remove any unwanted file(s) and/or squash the disk.
@49@ No disk space	The available disk space on all of the fixed disks has been utilized.	Remove any unwanted file(s) or squash the disk.
@4A@ Not ready	SYSCOLDSTART is trying to access a not ready disk.	Make the disk ready.
@4B@ Not single area file	The file to be patched is not a single area file.	Copy the file specifying "singlearea."
@4C@ Pack not online	A pack-id of a not ready disk has been entered.	Check input and re-enter or insert requested pack in drive.
@4D@ Incorrect file type	When loading or patching system files, a check is done on their file type. The proper type must be present for the load and patch functions to successfully complete.	B900RL# disk may be corrupted. Try a backup copy of the disk or request technical assistance.

Table 9-1. Codes Appearing in Bank A of Indicator Display (Coldstart Status) (Sheet 2)

CODE MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
@4E@ Invalid drive	A drive which is not on the system has been specified.	Execute OL to see what peripherals are on-line. Check input and re-enter.
@4F@ Invalid label on drive	When performing an OL function, this message occurs, indicating that the label on the specified drive is not a CMS disk label.	Disk must be re-initialized on a CMS system.
@50@ Number of bad sectors = 〈#〉	When initializing or reformatting, this message reports the number bad sectors on the disk.	None, normal operation of initialize and reformat.
@51@ Extended pack files not handled	Files that extend over multiple packs cannot be loaded by COLDSTART.	Copy file to one pack.
@52@ Invalid pack	An attempt was made to initialize a pack named "B900RL#" or when entering drive for "B900RL1", the drive does not contain 'B900RL1" disk.	Insert proper disk.
@53@ No ready fixed disk on system	In response to the prompt "TYPE OF DISK?," "FIX" is entered and no ready fixed disks on the system exits.	Ready fixed disk.
@54@ Too many bad sectors	A disk containing more than 50 bad sectors is not initialized. COLD-START stops initializing when this event occurs.	Initialize another disk.
@55@ Bad track zero	CMS defines that track zero must be error free. The disk is not initialized when an error on this track is discovered.	Initialize another disk.
@56@ Enter release disk	When a B900RL1 disk is needed, this message occurs. It can occur at the beginning of COLDSTART if the SYSINITBOOT file is needed, or during the load, patch or replace functions.	Insert the B900RL1 disk.
@57@ No disk space for directory	During the initialize or reformat function, if the disk has bad sectors there may not be enough contiguous good sectors for the directory size chosen.	Reduce the directory size or use another disk.
@58@ Disk is not in initial state	When performing the load function the object disk must be in an initialized state.	Initialize or reformat the disk.

Table 9-1. Codes Appearing in Bank A of Indicator Display (Coldstart Status) (Sheet 3)

CODE MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
@59@ Duplicate packid	An attempt was made during the initialize or reformat functions to label the disk with the same name as one already on-line.	Power off the "duplicate" pack or specify another name.
@5A@ No bootstrap file	At the beginning of COLDSTART the SYSINITBOOT file is loaded from the B 900RL1 disk into processor memory. If this file is not found, this message will occur.	Warmstart the system with a disk that has the SYSINITBOOT file and copy it to B900RL1.
@5B@ B900 System load incomplete	After the load function has performed the file transfers to the system disk, a check is done to see if the necessary files have been loaded. If all of the required files are not on the system disk, this message occurs.	Check the B900RL1 disk to see what files are missing. Copy those files to B900RL1.
@5C@ Bad MTR track	If one of the tracks to be used for MTR purposes, during the FE initialize function, has been determined to be bad, this message will occur.	Try FE initialize on another disk.
@5D@ Ready	This is not an error message. It indicates the status of the drives during the OL function.	None.
@5E@ Unexpected interrupt	Interrupts are disabled at all times during the execution of COLDSTART. This message will occur if an interrupt is received.	Request technical assistance.
@5F@ Decimal number > 65535	This message occurs when COLD-START detects a decimal number greater than 65535.	Request technical assistance.
@60@ ODT not ready	This error code displayed if the ODT is not ready.	Power on the ODT.
@61@ Filesize error	When computing area sizes for files, if the file area is larger than (65535 minus allocation unit) sectors.	Request technical assistance.
@62@ DP bad prep status	A bad status error was reported (that is, a status is not available for each active device).	Request technical assistance.
@63@ DP-bad status	An invalid command status result was retruned by the disk processor.	Request technical assistance.

Table 9-1. Codes Appearing in Bank A of Indicator Display (Coldstart Status) (Sheet 4)

CODE MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
@64@ DP-unsolicited status	A command complete status was returned by the disk processor for an invalid drive.	Request technical assistance.
@65@ DP-bad ready status	A ready status is returned for a ready drive.	Request technical assistance.
@66@ Fixed disk not ready	The fixed disk was selected as the system disk and it is not ready.	Power on the fixed disk.
@67@ Not extendable	The system entry in the PPIT does not contain a logical unit of @FF@.	Reformat fixed disk.
@68@ PPIT mismatch	Before loading to fixed disks, the fixed disks are checked for identical PPIT's.	Power off all fixed disks except the one being loaded to or re-initialize with all fixed disks powered on.
@69@ Duplicate logical unit	The same logical unit number has been assigned to two fixed disks.	Request technical assistance.
@6A@ Divide by zero	A zero divisor was sent to the divide routine.	Request technical assistance.
@6B@ Memory parity error	A hard read error occurred.	Request technical assistance.
@6C@ No system pack	Load has detected an allocation unit of zero while attempting to load a multi-area file.	Re-initialize or reformat disk.
@6D@ Version mismatch	When patching, a check is done on the version string source file.	Use a more current B900RL1 disk or use replace function.
@6F@ Invalid disk	Invalid disk device.	Request technical assistance.
@71@ Invalid bootstrap file	The SYSINITBOOT OR SYSBOOTSTRAP file contains an invalid check string.	Use another copy of SYSINITBOOT and/or SYSBOOTSTRAP.
@73@ Function DS'ED	This message will occur when ?DS is entered to discontinue the present function.	None, Normal operation.
@74@ Patch level error	When executing the patch function, the patch file must be a later release than the source file.	Try a more current B900RL1 disk or use the replace function.

Table 9-2. Codes Appearing in Bank D of Display (Coldstart Function)

CODE	MEANING
@10@	The initialization function of COLDSTART is in operation. Banks F, G, and H contain an incrementing track or sector location.
@11@	The write-address logic of the initialize function is executing. Banks F, G, and H contain an incrementing address being written to disk.
@12@	The accuracy-test logic of the initialize function is executing.
@13@	The build-Track-0 logic of the initialize function is executing.
@14@	The build-non-file-directory logic of the initialize function is executing.
@15@	The build-non-file-name-list logic of the initialize function is executing.
@16@	The build-disk-file-name-header-list logic of the initialize function is executing.
@17@	The build-SYSMEM logic of the initialize function is executing.
@18@	The build-PPIT logic of the initialize function is executing.
@20@	The load function of COLDSTART is in operation.
@30@	The reformat function of COLDSTART is in operation.
@33@	The build-Track-0 logic of the reformat function is executing.
@34@	The build-non-file-directory logic of the reformat function is executing.
@35@	The build-disk-file-name-list logic of the reformat function is executing.
@36@	The build-disk-file-header-list logic of the reformat function is executing.
@37@	The build-SYSMEM logic of the reformat function is executing.
@38@	The build-PPIT logic of the reformat function is executing.
@40@	The OL function of COLDSTART is in operation.
@50@	The WARMSTART function of COLDSTART is in operation.
@60@	The HELP function of COLDSTART is in operation.
@70@	The FE initialize function of COLDSTART is in operation.
@80@	The patch function of COLDSTART is in operation.
@90@	The replace function of COLDSTART is in operation.

Disk Selection

All disks have been assigned priority according to their speed. The fastest disk device is the fixed disk, then the cartridge disk, and then the mini disk. Because more than one type of cartridge drive is permitted on the system, they have individually been assigned priorities. Refer to the section on Startup Display Reference Tables for a list of disk drives in their highest priority order.

In attended mode, the operator may select which drive/disk the desired function is to act upon. In unattended mode, however, coldstart searches for the highest priority device. If it is to be cartridge, the top drive is chosen over the bottom drive.

Please note that the initialize function can not be run against the release disk. The release disk may be scratched via the reformat function in attended mode. To safeguard against accidental reformating or generally writing to it, the release disk used may be write inhibited.

Default Assignments

Some of coldstart's functions allow the operator the option of assigning disk attributes, or defaulting to system set up values. The default values are always used in the unattended mode. In the attended mode the message "DEFAULT MODE <Y OR N>" is displayed on an ODT. If Y is entered, the system assigns the number of files to be allowed on disk, serial number, packid, and owner-id as shown in the following table:

INFORMATION NEEDED	REMOVABLE DISK DEFAULTS	FIXED DISK DEFAULTS
Number of files to be allowed on system disk.	256 **	2805 **
Serial number of disk used as system disk.	000000 (EBCDIC)	000000 (EBCDIC)
Pack-id of disk used as system disk.	000000A (ASCII)	0000001 * (ASCII)
Owner-id of disk used as system disk.	"REMOVABLE DISK"	"FIXED DISK"

(*) - If more than one fixed disk, the first has the packid as shown; each additional disk's packid is incremented by 1.

(* *) - 2805/256 includes the system file. 2804/255 is the actual number of user files which can reside on the disk.

If default mode is not selected, the function continues to request all necessary information from the user. Refer to the following table for the format and the type of information requested.

INFORMATION NEEDED	SYSTEM QUERY	RESPONSE FORMAT
Number of files to be allowed on system disk.	ENTER FILES ⟨1-2804⟩	Up to four digits.
*Serial number of disk serving as system disk.	ENTER SERIAL ⟨6 DIGITS⟩	Exactly six digits.
Pack-id of disk serving as the system disk.	ENTER OBJECT PACKID ⟨1-7 CHAR⟩	Up to seven letters and/or digits.
*Owner-id of disk serving as the system disk.	ENTER OWNER ⟨UP TO 14 CHARACTERS⟩	Up to fourteen letters and/or digits.

(*) - Removable disk only.

WARMSTART

The warmstart utility starts when the coldstart or SYSBOOTSTRAP utility passes control to it. Warmstart is responsible for:

1. Determining the hardware configuration by associating each processor that it finds with local memory size to the appropriate processor attribute (that is, operating system, task processor, data comm processor, or disk processor). Warmstart uses these attributes to verify the specifications found in the SYSCONFIG file, and to load the proper system firmware files.
2. Determining that all required system files are available by searching all ready disk's on the system. The search proceeds from the fastest drive to the slowest (the order is described in the section Startup Display Reference Tables under DISKID). All required system files must be found to reside either entirely on the system's fixed disk assemblage or entirely on a single removable disk.
3. Assigning appropriate roles in the software system to each component of the hardware system as follows:
   1) If the user configuration specifications in the SYSCONFIG file are satisfied by the hardware currently available, warmstart configures the software system according to the SYSCONFIG file.
   2) If the current hardware system cannot satisfy the user configuration specifications in the SYSCONFIG file, warmstart uses the default configuration (refer to Default Configuration.)
4. Loading system firmware as required by the software system assignments.
5. Passing control to the operating system.

NOTE

If the system has just been coldstarted, the version of SYSCONFIG that is used by warmstart is the version supplied on the release disk(s). If the configurer program has been executed since the last coldstart, the version of SYSCONFIG that is used by warmstart is that produced by the most recent execution of configurer. To restore the original release version to the system disk, either do a new coldstart replace or execute configurer with the proper parameters.

Default Configuration

The B 900/CP9500 warmstarts using a default configuration if the specifications in the SYSCONFIG file conflict at any time during the warmstart process with the actual available hardware configuration.

If this condition exists, then the system is configured as follows:

1 - Operating System Processor.
1 - Disk Processor.
All - Data Comm Processors physically on the system, where logical DCP numbers increase from zero in a decreasing bus address order.
All - Task Comm Processors physically on the system as task processors supporting all interpreters.

Warmstart also allocates the following buffer memory:

1. If the operating system processor memory is greater than or equal to 192KB, then buffer memory size is 64KB.
2. If the operating system processor memory is less than or equal to 192KB, then buffer memory size is total OS memory less 96KB.
3. The data comm buffer memory allocated is one half of the total buffer memory allocated.

WARNING

With the 3.02 release of System Software, there is no way of knowing that the system has warmstarted using a default configuration. A method of determining when a default configuration has been used during warmstart will be provided on a future release.

To tailor the SYSCONFIG file for specific configurations, the configurer program must be used. A repeat warmstart can then be done which loads the needed configuration. (The configurer program is described later.)

Operator Attended Mode

These instructions are for operating warmstart when the system has not just been coldstarted. The coldstart program has the option warmstart which describes the necessary operations.

1. Power on the B 900/CP9500 processor by pressing the ON/CLEAR switch (refer to System Startup).
2. Insert a valid system disk (prepared by the B 900/CP9500 coldstart utility).
3. Set the B 900/CP9500 processor's SYSTEM/REMOVABLE switch to SYSTEM.
4. Press the system CLEAR button.

Warmstart displays the following sequence of messages:

BURROUGHS CMS MCP (version #)
<disk mnuemonic> <packid>/SYS DISK 0 FILES OPEN

ENTER DATE AND TIME USING DT COMMAND

Warmstart has requested the date, with slashes between month, day, and year, and time in hours and minutes. Leading zeroes must be included.

<DATE AND TIME>
TRANSFER COMPLETED COMMENCING LOG FILE REALLOCATION

LOGGING IS INITIATED ON <MM/DD/YY> AT <TIME>
(MCP VERSION #)

Operator Unattended Mode

The system automatically enters unattended mode when warmstart determines that there is no ODT on the system. Because the date and time cannot be entered, warmstart uses the default values of 11/11/11 for the date and 24:00:00 for the time. As in attended mode, the system is configured according to SYSCONFIG. If the SYSCONFIG file is invalid or not found, the warmstart default configuration is used to configure the system. The hexadecimal display does not indicate that the system has warmstarted using the default configuration.

Warmstart Messages

Warmstart has two ways of reporting its status to the operator. The hexadecimal display banks are used by warmstart to display data during its execution. Display banks A, B, and C indicate warmstart's progress through its code and serves as a minimal error report. The other banks, D through H, may supply further data, depending on the error involved.

When an operator display terminal (ODT) is present on the system. Warmstart attempts to direct a message to the ODT device. Banks A, B, and C are periodically updated. Each bank is responsible for a different runtime data report.

Bank A

Errors detected by warmstart are displayed in bank A. Warmstart errors can be broken-down into the following ranges:

VALUE	MEANING
@A0@	Normal execution, no error
@A1@ – @AF@	WARMSTART internal errors
@B0@ – @BF@	System error
@C0@ – @CF@	Hardware error

Bank B

This is a hexadecimal count of successfully executed operations. (Note: This is not a count of executed micro-instructions; it is a count of logical operations.) This bank has a maximum possible value of @FF@ (255 decimal).

Bank C

In the event of an error, this bank represents the last operation attempted during which an error occurred. (During normal operation, it is periodically updated to reflect the operation in progress.) The following table is a decode of the contents of bank C:

VALUE	OPERATION
@00@	Processor Interface Control (PIC) command
@10@	List peripherals
@11@	Locate OS peripheral
@12@	Determine hardware configuration
@13@	List remote peripherals
@14@	Locate system disk
@20@	Prime local device software controller
@21@	Prime disk processor parameters
@30@	Allocate WARMSTART buffer memory
@31@	System configuration
@40@	Handle disk processor ctm prep
@50@	Memory checkout – attached
@51@	Memory checkout – unattached
@60@	Prepare firmware segment load
@70@	Load firmware segment
@72@	Complete WARMSTART table
@80@	Display message to ODT
@E0@	WARMSTART complete – pass control to OS

The following table outlines the display layout for error information shown in bank A and banks D through H.

Bank A	Bank D	Bank E	Bank F	Bank G	Bank H
@A0@		NO ERROR (NORMAL)			
@A1@	– –	CURRENT OFFSET		NEXT OFFSET	
@A2@	– –	CURRENT OFFSET		NEXT OFFSET	
@A3@	– –	– –	– –	– –	– –
@A8@	OP TYPE	DEV TYPE	CHAN/UNIT	DEV DEP PARMS	
@A9@	OP TYPE	DEV TYPE	CHAN/UNIT	DEV DEP PARMS	
@AA@	OP TYPE	DEV TYPE	CHAN/UNIT	DEV DEP PARMS	
@AB@	OP TYPE	DEV TYPE	CHAN/UNIT	OFFSET	

Bank A	Bank D	Bank E	Bank F	Bank G	Bank H
@AC@	OP TYPE	DEV TYPE	CHAN/UNIT	OFFSET	
@AD@	OP TYPE	DEV TYPE	CHAN/UNIT	DEV DEP PARMS	
@B0@	– –	– –	– –	– –	– –
@B1@	– –	– –	– –	– –	– –
@B2@	FILE INDEX	SEG TYPE	– –	– –	– –
@B3@	PROC TYPE	– –	– –	– –	– –
@C1@	– –	– –	– –	– –	– –
@C3@	– –	– –	– –	– –	– –
@C4@	– –	– –	– –	– –	– –
@C8@	OP TYPE	DEV TYPE	CHAN/UNIT	STATUS	
@C9@	OP TYPE	DEV TYPE	CHAN/UNIT	STATUS	

NOTE

In certain cases, a single display bank is not sufficient. Two, sometimes three banks are then used for a single value. When this occurs, the leftmost value digit of the leftmost bank is the most significant digit. The rightmost digit of the rightmost bank is the least significant digit.

The following table explains the meaning and recommended action for warmstart codes appearing in bank A:

CODE	POSSIBLE CAUSES	SUGGESTED ACTION
@A0@	Normal execution, no errors.	None.
@A1@	Current WARMSTART opcode not recognized.	WARMSTART code may be corrupt. Reload the WARMSTART file to the system disk.
@A2@	Current WARMSTART operation not completed successfully.	Same as above.
@A3@	WARMSTART table entry type not recognized.	Same as above.
@A8@	Operation desired is not recognized for the given device.	Request technical assistance.
@A9@	Device not recognized.	Same as above.
@AA@	Invalid unit for the given device.	Same as above.

CODE	POSSIBLE CAUSES	SUGGESTED ACTION
@AB@	Source descriptor is invalid; does not exist or exceeds physical attributes.	Same as above.
@AC@	Destination descriptor is invalid; does not exist or exceeds physical attributes.	Same as above.
@AD@	Device not on system.	Same as above.
@B0@	System disk not found.	Ensure the system disk is active and has the required files.
@B1@	SYSCONFIG file error.	Reload the SYSCONFIG file.
@B2@	A required firmware segment during the system load was not found. A firmware file or the WARMSTART file may be corrupt.	Reload as necessary.
@B3@	Unable to load a required firmware segment. The segment exceeds physical page attributes. System firmware file may be corrupt.	Reload. Possible memory problems. Request technical assistance.
@C1@	Not able to access the OS processor interface control.	Request technical assistance.
@C3@	WARMSTART was not able to find the minimum memory necessary for it to allocate its disk parameters and buffers; or, the minimum memory on the operating system has not been found.	Request technical assistance.
@C4@	The hardware system ID in ROM is not a recognized value.	Request technical assistance.
@C8@	A hardware error has been detected on the given device.	Request technical assistance.
@C9@	Device not ready; the selected device has gone from operational to non-operational.	Ready device and re-initiate WARMSTART.

Warmstart Display Reference Tables

CURRENT OFFSET	CURRENT WARMSTART DRIVER TABLE OPERATION OFFSET.
NEXT OFFSET	NEXT WARMSTART DRIVER TABLE OPERATION OFFSET.
OP TYPE	INTERNAL WARMSTART DEVICE OPERATION. @00@ – COMMAND @01@ – READ @02@ – WRITE @03@ – SEARCH
DEV TYPE	INTERNAL WARMSTART DEVICE TYPE. @00@ – DISK @01@ – ODT @02@ – PROCESSOR (PIC) @03@ – MEMORY
CHANNEL/UNIT	FOR DISK THE MOST SIGNIFICANT DIGIT INDICATES THE LOGICAL DRIVE NUMBER OF THE DEVICE BEING ACCESSED THROUGH THE DISK PROCESSOR. THE LEAST ORDER DIGIT IS NOT USED. FOR A PROCESSOR OR MEMORY THE MOST SIGNIFICANT DIGIT INDICATES THE PHYSICAL BUS ADDRESS AND THE LEAST SIGNIFICANT DIGIT INDICATES THE MEMORY PAGE ON THE BUS.
OFFSET	ADDRESS OF ERROR.
FILE INDEX	SYSTEM FILE BEING ACCESSED. @03@ – SYSMCP @04@ – SYSICP @05@ – SYSDSCP @06@ – SYSMPLII @07@ – SYSCONFIG @08@ – SYSLANGUAGE @09@ – SYSTRANSLATE
SEG TYPE	CURRENT FIRMWARE SEGMENT TYPE. @02@ – RESIDENT CODE @03@ – RESIDENT DATA @06@ – VITAL RESIDENT CODE @0A@ – PRIME RESIDENT CODE
PROCESSOR TYPE	CURRENT WARMSTART ENTRY TYPE. @01@ – OPERATING SYSTEM @02@ – DATA COMM @03@ – TASK @04@ – DISK @05@ – BUFFER @FF@ – TERMINATOR

STATUS

BANK G – GENERAL STATUS.

00	–	CONTROLLER ERROR (INTERNAL TO OP)
01	–	COMMAND SUCCESSFUL
02	–	COMMAND UNSUCCESSFUL
03	–	DEVICE ERROR ABORTED COMMAND
04	–	NOT USED
05	–	DESCRIPTOR ERROR

BANK H – DEVICE ERROR STATUS.

00	–	NO ERROR
01	–	SEEK TIMEOUT
02	–	HEAD OFF CYLINDER
03	–	SEQUENCE ERROR (INTERNAL TO OP)
04	–	PARITY
05	–	SECTOR NOT FOUND
06	–	ILLEGAL ADDRESS
07	–	STATUS WORD ERROR (INTERNAL TO OP)
08	–	DATA ERROR (IN COMPARE)
09	–	WRITE INHIBITED (ATTEMPT TO WRITE TO A "WRITE PROHIBITED" DISK).

OS AND DISK PROCESSOR SWITCHES

These switches are available on certain systems. They are found along with the other system switches on the control panel of the computer. Their purpose is to aid the operator in bypassing a faulty OS (operating system) and/or disk processor. The following procedures enable the user to continue operation until the faulty processor can be fixed. However, the user should not set these switches unless advised to do so by Burroughs technical personnel.

OS PRC1/OS PRC2 Switch

The normal position of the OS switch is OS PRC1. Pressing OS PRC2 allows a task processor to take over the responsibilities of the operating system processor. The old OS processor is assigned as a task processor. The OS processor can be used in conjunction with the disk processor switch described in the next section.

The following procedure should be followed for setting the OS processor switch:

1. Logically power off the system disk using the PO command.
2. Press the OS processor switch to OS PRC2.
3. Rewarmstart the system using the warmstart procedure.

DSK PRC1/DSK PRC2 Switch

The normal position of the disk processor switch is DSK PRC1. Pressing DSK PRC2 allows a task processor to take over the responsibilities of the disk processor. The following procedure should be followed for setting the disk processor switch:

1. Logically power off the system disk using the PO command.
2. Press the disk processor switch to DSK PRC2.
3. Rewarmstart the system using the warmstart procedure.

MEMORY DUMP

GT MD

This feature allows the operator to dump the contents of memory to a dump file at will and does not require the system to be rewarmstarted. The dump file can then be printed or copied to another disk to be printed

at a later time. To invoke this feature, the operator must enter GT MD on the ODT or, if not available, a remote SPO terminal. The MCP temporarily suspends all jobs in the mix, displays the message SYSDMFILENN OPENED, writes the contents of memory to the dump file, and displays the message SYSDMFILENN CLOSED, then resumes normal mix activity.

Read-Only-Memory (ROM) Dump Routine

The ROM dump routine is a software debugging tool which resides in the system's ROM. It is used when the MCP is unable to produce a system dump file for analysis by the system dump analyzer program (SYSANALYZER). It allows the dumping of all contents of system RAM to a CMS-compatible removable disk.

ROM dump also allows the operator to examine selected areas of system RAM, via the hexadecimal display banks.

The operator interfaces with the ROM dump routine via the hexadecimal MTR keypad and the hexadecimal display lights (refer to B 900/CP9500 Control Panel). These interfaces are described in the following paragraphs.

Hexadecimal MTR Keypad

The MTR keypad is a rectangular pad, comprising eighteen keys and a buffer-full indicator light, located at the left end of the B 900/CP9500 processor's control panel. Ten of the eighteen keys are labeled with the decimal digits 0 through 9. Six keys are labeled with the hexadecimal undigits A through F. The two remaining keys are at the top of the keypad. The shift key is located to the right of the buffer-full indicator. It is a momentary shift key, and must be pressed together with the key being shifted. The key to the left of the buffer-full indicator is presently not connected. Keys 0 through F, when unshifted, are used to specify addresses. Keys A through F, when shifted, have various control and option meanings.

NOTE

The buffer-full indicator lights whenever the rate of the operator's key entries overtakes the keypad's rollover capacity. Keys pressed while it is lit do not register a value.

ROM Dump Usage of Displays

The hexadecimal display lights consist of eight banks of two digits each, arranged in two rows, each row containing four banks (refer to B 900/CP9500 Control Panel). The ROM dump routine uses the display lights to display RAM address values (that is, bus/page and/or offset), contents of RAM, and error information.

Operation of ROM Dump

The following paragraphs provide all necessary instructions for operation of the ROM dump routine.

Initiation

The ROM dump routine is initiated as follows:

1. Set the OPERATE/MEMDUMP switch to MEMDUMP.
2. Press the ON/CLEAR switch. This causes the system to come up in the MTR DEBUG routine.
3. If a ROM dump to disk is being initiated, insert the dump disk into the loader device, and continue to step 4.
4. Press the SHIFT key and E together. After this shifted E is entered, the hexadecimal displays blank out. This indicates that the system is in software dump (option mode), waiting for the operator to select an option.

Option Idle State

The ROM dump routine enters an option idle state (that is, waiting on an option specification key) for any one of the following conditions:

1. After a shifted E has been entered.
2. Upon receipt of the reset key during one of the ROM dump options.
3. Upon the successful completion of one of the dump-to-display options.
4. Upon the successful completion of the dump-to-disk option.
5. Upon recovery from a non-fatal error which occured during the execution of one of the dump options.

Option Mode

The following options are available to the operator when the ROM dump routine is in option mode. Shifted hexadecimal characters refer to the simultaneous pressing of the shift key and one of the hexadecimal keys.

Dump-To-Display (Shifted A)

This option allows the user to specify an address in system RAM. The first eight bytes of this address are viewed on the hexadecimal displays.

Continue-Dump-To-Display (Shifted C)

This option allows the user to view the contents of the eight bytes following the eight bytes previously viewed, via dump-to-display or continue-dump-to-display.

Dump-To-Disk (Shifted D)

The operator can use this option to dump all contents of system RAM to a CMS-compatible, removable disk.

Reset (Shifted B)

This option may be used at any time during the ROM dump routine. Entering a shifted B returns the ROM dump routine to its option idle state.

References to an address, in both the previous and in the following material, implies a RAM memory location specified by a four-digit memory offset, a one-digit bus address, and a one-digit page address, all in hexadecimal. The entry format of an address is:

A1	A2	B1	B2	—	—	D1	D2
E1	E2	F1	F2	—	—	H1	H2

Where:
1. Banks E and F contain the previous offset.
2. Digit H1 contains the previous bus.
3. Digit H2 contains the previous page.
4. Banks A, B, and D contain the key value as entered on the MTR keypad.

Dump-To-Display

The dump-to-display option of the ROM dump routine allows the operator to specify an address in system RAM. The contents of the first eight bytes of RAM starting at the specified address are shown in the hexadecimal displays. Dump-to-display first displays the previous address as offset in bank E-F, bus address in bank H1, and page address in bank H2. Specification of a new address to be inspected is then allowed.

Address Specification

The offset is always the first part of the address specified. Enter four hexadecimal digits on the MTR keypad. Display banks A and B, together, reflect the entire offset entries. The latest keystroke is added to the display by moving previously entered digits leftward from bank B2 to B1 to A2 to A1. Therefore, all leading zeros must be included.

The bus address is entered on the fifth keystroke and is displayed in bank D2. The page address is entered on the sixth keystroke moving the previously entered bus address into bank D1 and displaying the page address in bank D2.

Upon entering a complete address, an entry of the terminator key (shifted F) is required for the dump-to-display to proceed. After viewing the contents of the previously entered address, the following options are available:

1. Entering an entirely new address (that is, offset, bus, and page).
2. Entering a new offset while retaining the bus and page addresses previously specified.
3. Entering shifted C (continue-dump-to-display).

To enter an entirely new address, enter shifted A followed by the six digits of address information.

To enter a new offset and retain the bus and page addresses previously specified, enter a shifted A followed by the four-digit offset. Entering a shifted F after the new offset has been entered signals the ROM dump routine to fill the hexadecimal displays with the contents of eight bytes of memory, starting at the offset, bus, and page address specified.

Continue-Dump-To-Display

When the ROM dump routine enters the continue-dump-to-display mode, the previous starting address dumped is shown in the lower four banks of the display, banks E-H. The previous offset plus eight and the previous bus/page address are shown in the upper four banks of the display, banks A-D. The operator now has the following options:

1. Entering shifted F or terminating the entry, whereupon the contents of the address shown in banks A-D are displayed. At this point, shifted C may again be entered to continue the dump-to-display. This technique of alternating between shifted F and shifted C allows the operator to step through memory, displaying eight bytes of data at at time.
2. Entering shifted B returns the ROM dump routine to an idle state.

Dump-To-Disk

The ROM dump routine allows the dumping of all contents of system RAM to a CMS compatible disk on the loader/coldstart I/O channel only.

After entering the dump-to-disk mode, the operator has the option of:

1. Pressing the terminator key (shift F) which causes the dump-to-disk to execute.
2. Pressing the reset key (shift B) to return the ROM dump routine to an idle state.

The following aspects of dumping-to-disk should be noted:

1. The hexadecimal display banks are used to display a continuous update of the RAM address being dumped to disk.
2. The dump-to-disk is complete when all hexadecimal displays are blank.
3. Dumping is not selective. All of system RAM is dumped to the disk.
4. The data being dumped is written onto the dump disk beginning at the first good sector after the highest addressable directory structure. The dumped data skips across any bad sector(s) encountered during the dump process. If a bad sector is found, the first good sector after it is used.
5. The dumping algorithm just described results in the overwriting of the contents of the files already on the disk. The disk's directory is not updated to reflect this corruption.

The first action involved in dumping-to-disk is the location of the dump disk and the location on that disk of the highest addressed directory structure.

Dumping begins in the first byte following this structure's end address. If this causes the dump to begin in a bad sector however, the first byte of the next good sector is used. Data is dumped in 180-byte records.

The first record contains a validity string, which indicates that the disk contains a ROM dump. The validity string is: B900PROMDUMP0.Q.

The first nine bytes of each page of memory dumped contain a validity string of the following format:

Bus Address: One byte
Page Address: One byte
Page Size: Two bytes
Error Status: One byte
IC-Status 1: One byte
IC-Status 2: One byte
Next Page: Two bytes, disk sector address for start of dump of next page.

The respective memory page contents, from its zero address to its limit, are dumped immediately following its descriptor. The descriptor starts on a record boundary. Each memory page dumped takes-up only the number of records necessary for it and its respective descriptor.

The last record written by the ROM dump is either:

1. A terminator record, when the disk is able to contain a complete dump (that is, a dump consisting of the entire system RAM memory dump, the validity record, and the terminator record) indicated by a blank display.
2. The last record that could be dumped for the current memory page, when an end-of-disk condition occured (indicated by 12 displayed in bank D).

The terminator record, if written, consists of the following string: B900PROMDUMPED.

ROM Dump Messages

Hardware detected errors are handled by presenting error relevant information on the hexadecimal displays in one of the following formats:

Bank D	Bank A	Bank B	Bank C	Bank E	Bank F	Bank G	Bank H
@08@, @18@, @28@, @38@, @58@, @88@, @C8@, @FF@	OFFSET		---	STATUS	---	MAX REG	
@04@, @10@–@13@	---	---	---	---	---	---	---
@01@–@03@	SECTOR ADDR		---	DCPS	DCSS	---	---

The following table lists the meanings of error codes found in bank D of the hexadecimal display. For a decode of the identifiers and codes found in banks A-C and E-F, refer to the ROM Dump Display Reference Tables.

ERROR CODE	POSSIBLE CAUSES	SUGGESTED ACTION
08	Hardware detected memory error, either a data read parity error or memory limit occurred during an execution of Dump-To-Display.	Re-enter Dump-To-Display option as keypad will be enabled for entry of next option (option idle state).

ERROR CODE	POSSIBLE CAUSES	SUGGESTED ACTION
18	An error occurred during the setup for or execution of the Dump-to-Disk option.	System clear is necessary to restore the processor to a running state.
28	Dump routine could not find a page in memory while dumping to disk.	Same action as above.
38	Dump routine failed while checking disk label prior to dumping to disk.	Same action as above.
58	A bad area was found on the dump disk.	Same action as above.
88	Read after write error was detected after Dump-to-Disk was complete.	Same action as above.
C8	An write error was detected when dumping to disk.	Same action as above.
FF	A Fatal error. A hardware detected error has occurred which gives an unexpected error status.	System clear is necessary.
04	The selected dump disk has gone from operational to non- operational	Restore dump disk to operation state. Dump will automatically continue from point of interruption.
10	No memory was found to dump	Read/Write memory has not been found to exist on the hardware system. The keypad is left enabled or is in option idle state.
11	A dump disk was not found. Dump-to-disk requires a write enabled, operational disk on the dump disk channel.	Insert a dump disk into a valid drive and power up. The keypad is enabled or is option idle state. It is necessary to respecify the Dump-To-Disk option.
12	The end of the dump disk was reached before the dump was complete.	No continuation is allowed. The keypad is left in an option idle state.
13	A valid CMS disk label was not found on the first write enabled, operational disk.	Power off invalid disk and insert a valid dump disk in the correct drive.
01	A disk seek error has occurred.	The error may have been caused by either the disk or the disk drive unit. Reinitiate the Dump-To-Disk using another disk and/or alternative drives where available.
02	A disk read error has occurred.	Same action as above.
03	A disk write error has occurred.	Same action as above.

ROM Dump Display Reference Tables

IDENTIFIER	DISPLAY BANK(S)	MEANING
DCPS	E	Disk controller primary status.
DCSS	F	Disk controller secondary status.
OFFSET	A, B	Data offset or micro offset where the error occurred.
MAX REG	G, H	The contents of the max hardware register at the time of the failure.
SECTOR ADDR	A, B	The disk sector address where the error was detected.
STATUS	E	The STATUS code found in bank "E" may be decoded as follows:

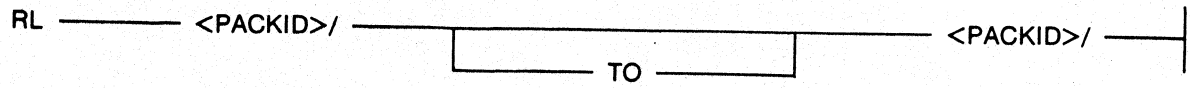
BANK E STATUS BIT	NAME	VALUE	MEANING
0	Write/read access	1	A write operation was being performed when the error occurred.
		0	A read operation was being performed when the error occurred.
1	Data/micro access	1	A data access was in progress when the error occurred.
		0	A micro instruction was being fetched when the error occurred.
2	Memory limit error	1	An attempt was made to access non-existant memory.
3	Boundary error	1	A boundary error has occurred.
4	Write parity error	1	A write parity error has occurred.
5	Read parity error	1	A read parity error has occurred.
6	Error state	1	An error has occurred since the last clear or reset error status word command.
		0	An error has not occurred since the last initialize or reset error command.
7	Second error	1	A second error has been detected before the error state bit was reset. Occurrance of a second error will freeze the processor.
		0	A second error has not occurred.

NOTE

Bank E status bit 0 is the least significant bit. For example, if bank E contains a @22@ this indicates that a read parity error occured while doing a data access.

RL (Relabel)

This intrinsic allows the operator to relabel the packid of any CMS disk without disturbing the contents of the disk. It enables a remote operator to change the packid of the disks that were initialized with default packids on a SPO-less system, but is not restricted to such use. It provides a mechanism for changing the pseudo packids that are defined for the system. The syntax for this command is:



Two conditions are required for this command to be accepted:

1. If a pseudo-packid is being relabelled, then all physical units that have packids listed in the PPIT must be physically ready.
2. A null mix must exist. On a system with a remote operator, the null mix requirement is satisfied if the only tasks in the mix are the MCS that is managing the remote system operation, and a program named SPIM.

When the packid requested to be changed is a pseudo-pack, this command causes the entry for that packid in the PPIT to be changed to the new packid. When the packid requested to be changed identifies a physical pack, this command causes the packid in the disk label field of the designated physical unit to be updated with the new packid and assigns the unit to the system with this new packid.

WARNING

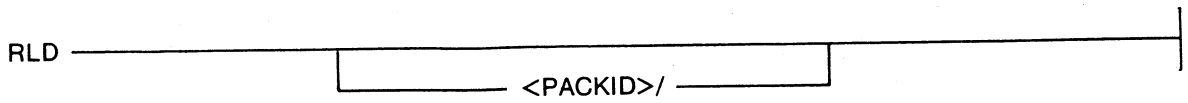
RL does not scan the disk file header list and does not notice or attempt to correct disk file headers of dual pack file components or KFPBs.

ERROR MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
RL INVALID	An attempt has been made to relabel a disk without a null mix.	Re-enter when system has a null mix.
"RL ⟨packid⟩/TO ⟨packid⟩/COMPLETE"	The command was completed successfully.	None, normal operation.
"FAULTY SCL INPUT" NOT ON SYSTEM	The pack-id specified to be changed is not present.	Re-initiate RL with both valid packids.
RL PACKID/DUPLICATE PACK	A duplicate pack condition is detected.	If possible, PO the pack that is not being relabelled.
"FAULTY SCL INPUT" INVALID	The physical unit from the PPIT is not ready.	RY the specified device and reinitiate RL.
"FAULTY SCL INPUT" INVALID	The pack-id specified is not legal.	Re-initiate RL with valid packid.

RLD (Release Level Display)

The release level display (RLD) checks whether any of the following system files are present on a specific disk, and reports the release level of each file found:

SYSBOOTSTRAP
SYSCOBOL
SYSCOLDSTART
SYSDSCP
SYSICP
SYSINITBOOT
SYSMCP
SYSMPLII
SYSWARMSTART



The disk that RLD is to examine for the system files is identified by <packid>. The default is to the first release disk (packid B900RL1). The packid for fixed disk is <0000000>. The RLD utility searches the disk identified by <packid> for system files. Each file search succeeds or fails individually, without affecting others. The files are sought in alphabetical order. For each file, RLD displays one of the following messages:

1. <mfid>/<file-id> UNABLE TO OPEN FILE - indicates that the file was not present on that disk.
2. <mfid>/<file-id> READ ERROR ON HEADER - indicates that the RLD utility was unable to extract release level information from the header of the indicated file.

3. <mfid>/<file-id> <mnemonic> RELEASE LEVEL XX.YY.ZZ - identifies the disk being examined, the file found, and the file's release level. The mnemonic is a two character string that uniquely identifies the file:

FILE NAME	MNEMONIC
SYSBOOTSTRAP	BT
SYSCOBOL	CB
SYSCOLDSTART	CS
SYSDSCP	DP
SYSICP	TP
SYSINITBOOT	BZ
SYSMCP	OL
SYSMPLII	MP
SYSWARMSTART	WS

RLD Messages

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
INVALID PACKID	A packid was entered that contains illegal characters (more than seven characters) is not followed by a slash or refers to an absent disk.	Check input and re-enter.

RO (Reset Option)

This intrinsic allows the operator to reset the printer backup option. Resetting the printer backup option causes files to be directed to a printer device if available. Otherwise, the program attempting to open the line printer is suspended waiting on device.

```
RO ———— PRBK ————————————————————————————————————————————————|
```

When a successful RO has been completed, the system responds with: VALID SCL INPUT COMPLETE.

SO (Set Option)

This intrinsic allows the operator to set the printer backup option. Setting the printer backup option causes files to be directed to a printer device if available. Otherwise, files are directed to disk.

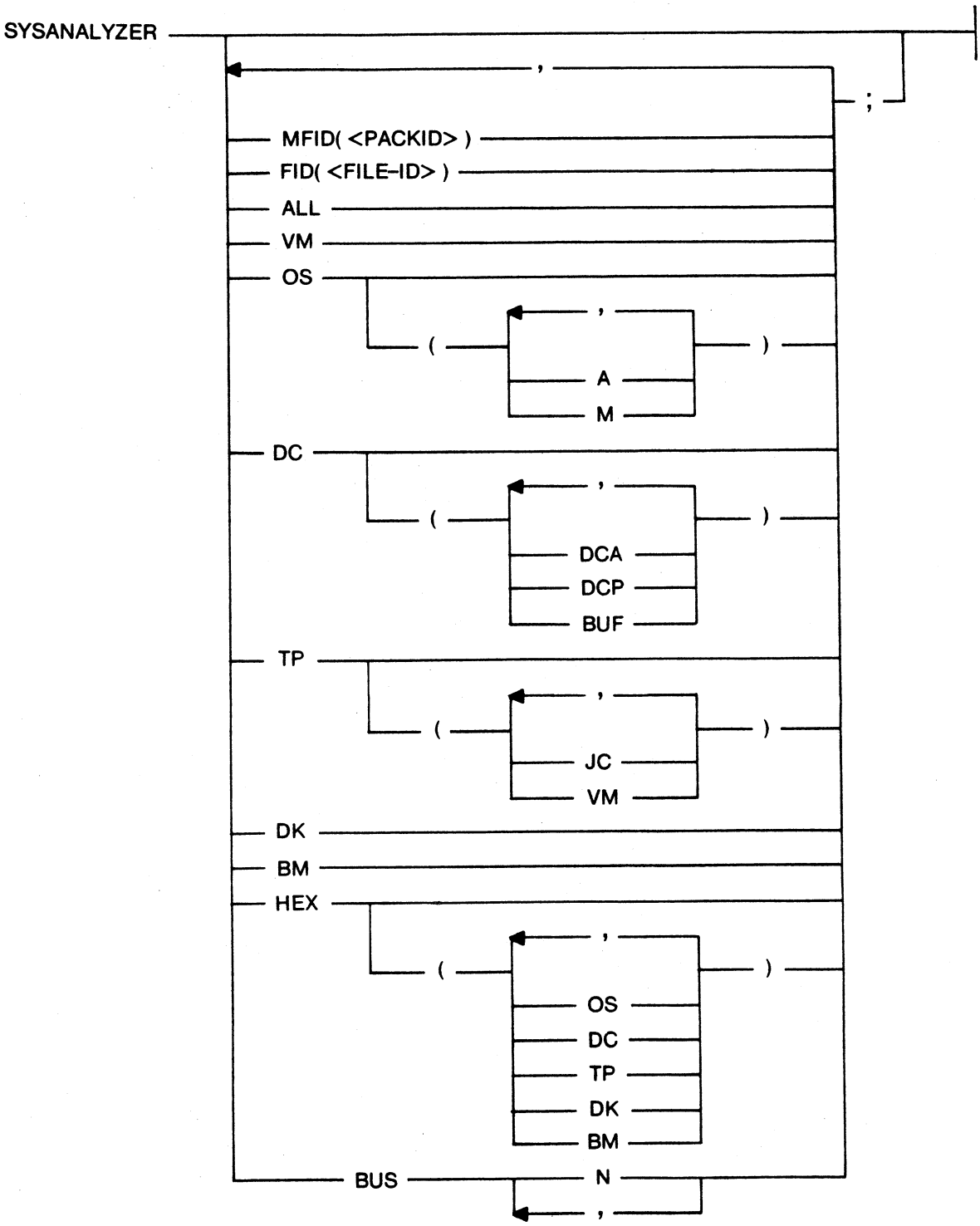
```
SO — PRBK ——————————————————————————————————————————————————|
          |————————————————————————————————————————————|————————————|————————————|
          |                                            | <PACKID>/  |  <FILESIZE>  |
          |————— TO ————|
```

Specifying a packid causes all printer backup files to be directed to that pack. If no packid is given, all such files go to the system disk. The <filesize> option changes the default maximum printer backup disk file size. When a printer file is opened, and the MCP determines that it must go to backup, if the maximum filesize in the FPB is 0, the default value is used; otherwise, the FPB value is used. This default value remains in effect until it is explicitly changed by another SO command, or the system is re-warmstarted. If no filesize is given, the default value remains as specified in bytes 347-349 of the miscellaneous information area of the SYSCONFIG file. Note that a value of 9999 is stored in the SYSCONFIG file unless this field is altered via the B 900/CP9500 configurer utility. When a successful SO has been completed the system responds with: SO COMPLETE.

SYSANALYZER (System Dump Analyzer)

The B 900/CP9500 system dump analyzer, SYSANALYZER, provides a detailed analysis of a system dump file.

Parameter values are supplied to SYSANALYZER via a single BOJ initiating message. The syntax of this message is depicted in the the railroad diagram below; the options may be entered in any order. The analysis still takes place in the sequence in which the diagram depicts those options.



CAUTION

The options ALL and HEX are advised for analyses requested by the user. ALL produces a formatted full system analysis; HEX produces a full system dump. The user is advised to ignore the more specific analysis options, except when otherwise requested.

The following points should be noted:

1. Unless the HEX option is specified, no hexadecimal dump is printed.
2. HEX is not included in the scope of the ALL option.
3. If the user enters the "FID( )" option and/or the "MFID( )" option, with no other options following, only the header page and dump file information are output by the analyzer program.
4. If the initiating message contains no options (that is, consists only of the string SYSANALYZER), the system analyzer uses default values equivalent to "MFID (<system pack>), FID (SYSDMFIL00), ALL, HEX;". (Note that if all options are omitted, the semicolon may not be included in the message.)

OPTION	MEANING
MFID( )	The <packid> specified is the pack on which the dumpfile resides.
FID( )	The <file-id> specified is that of the dumpfile to be analyzed.
ALL	Perform analysis of: Virtual Memory (see VM) Operating System (see OS) Data Comm Processor (see DC) Task Processor (see TP) Disk Processor (see DK) Buffer Memory (see BM)
VM	Perform analysis of virtual memory.
OS	Perform analysis of operating system. If no suboptions are specified, use all suboptions. Suboptions of OS: (A) Activity management. (M) Memory associated with OS processor.
DC	Perform an analysis of each data comm processor. If no suboptions are specified, use all suboptions. To restrict analysis to individual DCP s, use the "BUS" option. Suboptions of DC: (DCA) Include analysis of data comm-related parts of operating system. (DCP) Include analysis of data comm-related parts of disk processor. (BUF) Include analysis of data comm-related parts of buffer memory.
TP	Perform an analysis of each task processor. If no suboptions are specified, use all suboptions. To restrict analysis to individual task processors, use the "BUS" option, described below. Suboptions for TP: (JC) Job Control (VM) Virtual Memory
DK	Perform analysis of disk processor.
BM	Perform analysis of buffer management.
VM	Perform analysis of virtual memory.
HEX	Print an unformatted hexadecimal dump of the memory or memories specified. If no suboptions are specified, print a full hex dump of the system. Suboptions for HEX; (OS) Print a hex dump of memory associated with the operating system. (DC) Print a hex dump of memory associated with the data comm processor(s). (TP) Print a hex dump of memory associated with the task processor(s). (DK) Print a hex dump of memory associated with the disk processor. (BM) Print a hex dump of buffer memory.

OPTION	MEANING
BUS (n)	Each "n" is a hexadecimal character that corresponds to a given physical bus address. When the BUS option is specified, only the specified bus addresses are considered to be in the dump file.

The BUS (n) option allows analysis of the specific task processor(s) or data comm processor(s) that the user wishes to examine. When this option is not used, all task processors are analyzed as a result of TP, and all data comm processors are analyzed as a result of DC.

The OS processor's memory must be available if the DC(DCA) or VM option is specified. To do a complete analysis of a DCP when the BUS (n) option is used, the bus address of the OS processor must be specified as well as the bus address of the given DCP.

The BUS (n) option may be used together with the ALL option. This causes a comprehensive analysis, but only of the bus addresses specified.

SYSANALYZER Example

The following initiating messages result in the same analysis. The actual sequence of analysis is represented by the order of the options in the second message.

BOJ Message 1: SYSANALYZER FID(THISFIL), MFID(THATPCK), HEX, ALL;
BOJ Message 2: SYSANALYZER MFID(THATPCK), FID(THISFIL), ALL, HEX;

SYSANALYZER Output

The analysis printout on the line printer begins with a header page and ends with a trailer page. The following information is given on the header page:

SYSTEM DUMP ANALYZER

Sysanalyzer initiating message.
Packid of disk on which dump file resides.
Fileid of the dump file itself.
Date of creation of dump file (YY MM DD).
Time of creation of dump file (HH MM SS).
Date analysis began.
Time analysis began.
Revision level of analyzer.

NOTE

On systems without a real-time clock, a value of 24 00 00 is supplied in this time field.

The trailer page contains, in blocked-out letters, SYSTEM DUMP ANALYZER FINI.

Analysis

The structure of a full system analysis follows. A full analysis results from the use of the ALL option, either explicitly or by default.

- Dump File Information
  - Dump File Parameters
  - Dump File Map
- Virtual Memory
  - Data Access Tables
  - Mix Table
- Operating System Analysis
  - Status Blocks
  - Queues
  - Memories

DATA COMM ANALYSIS

- PARAMETERS
- ANALYSIS OF DATA COMM-RELATED OS MEMORY
  - ANALYSIS OF LINK BLOCKS
  - DC DATA
- ANALYSIS OF DATA COMM-RELATED DC MEMORY
  - RESERVED POINTER AREA
  - LINE AND STATION TABLES
- ANALYSIS OF DATA COMM-RELATED BUFFER MEMORY
  - AVAILABLE BUFFER POOL
  - RESULT QUEUE
  - REQUEST QUEUE
  - SUBNET QUEUE

⟨1 SET PER DC PROCESSOR⟩

TASK PROCESSOR ANALYSIS

- PARAMETERS
- ANALYZE STATUS BLOCKS
- ANALYZE CRITICAL QUEUE
- ANALYZE NORMAL QUEUE
- ANALYZE LIST HEAD
  - TP MEMORY ANALYZE
    - PARAMETERS
    - NORMAL HEAD ANALYSIS
    - TCB (TASK CONTROL BLOCK) ANALYSIS
    - PCB (PROGRAM CONTROL BLOCK) ANALYSIS
    - ICB (INTERPRETER CONTROL BLOCK) ANALYSIS

    ⟨1 SET PER PAGE OF MEMORY⟩

⟨1 SET PER TASK PROCESSOR⟩

Disk Management Analysis

Parameters
Task Information
Task Stacks
Manager Stack
Port Table
Command Table
Drive Table

Buffer Memory Analysis

File Table
Key File Table
System I/O Table
Available Table

System Analyzer Error Messages

The system dump analyzer reacts to errors in the initiating message by reprinting the syntax it received with all blanks removed. A flagging asterisk is printed underneath, marking the first erroneous option, suboption, or character detected. Analysis stops at the first syntax error.

It can be seen from the syntax diagram that two commas cannot appear consecutively. Misspelling of options, missing parentheses on suboptions, and optionless suboptions are also illegal. The following initiating message contains each of the errors just mentioned.

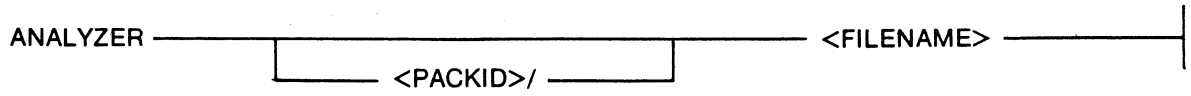
```
SYSANALYZER MFID(A),,FID(AB),(OS),OS(DMP;
            *
```

Only the second comma following MFID(A) is flagged. It is the first error detected and syntaxing stops at that point. An error detected in the dump file by the analyzer, or an error in the analysis, causes an error message to be generated within the analysis printout. These error messages are bracketed by strings of $$$ before and after the message (for example, $$$ ERROR MESSAGE $$$). Messages not bracketed in this way are merely warnings.

ANALYZER (User Program Dump Analyzer)

The B 900/CP9500 user-program dump analyzer, ANALYZER, analyzes dump files from COBOL, RPG, or MPLII user tasks.

The B 900/CP9500 user-program dump analyzer can be initiated in any mix.The syntax of the initiating message is shown in the following railroad diagram.



where:

1. <packid> = name of pack where dump file resides. If none is specified, system pack is assumed.
2. <filename> = name of dump file created when program was DP'ed. It should reside on the same disk as the program's object code.

CAUTION

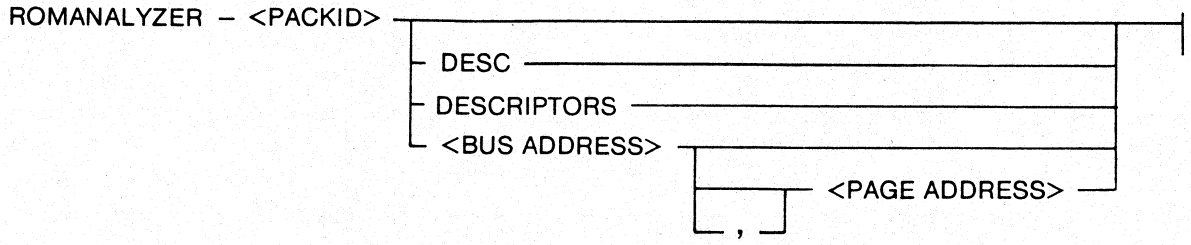
When analyzing a COBOL or RPG user program, if the object code file is removed or changed between the time the program is DP'ed and the time ANALYZER is initiated, it is likely that the analysis is incomplete.

There are several errors possible when entering the initiating message. The following error messages alert the user.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
PACK ID TOO LONG	A packid containing more than seven characters was entered.	Re-enter initiating message, using correct packid.
FILE ID TOO LONG	File name contained more than 12 characters was entered.	Re-enter initiating message, using correct fileid.
INVALID CHAR IN FILE	Initiating message contained character(s) other than alphanumerics commas, or spaces.	Re-enter initiating message, using only legal characters.
*** < FILENAME > NOT FOUND***	One of the following: 1. < packid > was of valid length and contained valid characters, but identified wrong pack. 2. < fileid > was of valid length and content, but dump file is not on disk.	If file is not on disk, install disk containing dump file. Otherwise, re-enter initiating message.
INTERP NOT SUPPORTED	Dump file is from program not handled by COBOL or BIL interpreters.	Cease trying to analyze dump file.
NO SPECIFICATION GIVEN	Error in initiating message not covered by preceding messages.	Re-enter initiating message.

ROMANALYZER (ROM Dump File Analyzer)

The B 900/ CP9500 ROM dump file analyzer, ROMANALYZER, allows the user to print selected parts, or all, of the information written on a removable disk by the ROM dump routine. ROMANALYZER can be run as a user job on any CMS system. ROMANALYZER communicates with the user via system accepts and displays.



where:

1. <packid> = packid of the removable disk containing the ROM dump file. The packid must identify a removable disk on which the ROM dump routine has created a dump file.
2. DESC. Print only the page descriptors.
3. <bus address> = the bus address to whose pages the ROMANALYZER program is to restrict itself (see note).
4. <page address> = the page on the specified bus to which the program is to restrict itself (see note).

NOTE

To specify a <bus address>, use a single hexadecimal digit between 1 and F. To specify a <page address>, use a single hexadecimal digit between 0 and F.

If a bus address appears in the initiating message (with or without a page address), ROMANALYZER processes the page(s) involved and issues an ACCEPT communicate. After processing those pages, the following messages appear on the screen:

<mix #> ROMANALYZER DISP: <BUS><EMPTY>: ,<PAGE>
<mix #> ROMANALYZER ACPT

When the ACPT message appears on the SPO, enter one of following:

AX <mix #> <bus address>
or
AX <mix #> <bus address>, <page address>

ROMANALYZER continues to request additional bus/page data until a null input (AX <mix #>) is entered.

The output generated by ROMANALYZER is directed to the line printer and is in one of the following formats:

1. All descriptors and hexadecimal dumps for all pages as contained in a ROM dump file.
2. All descriptors for all pages, with no hexadecimal dumps.
3. Descriptors and hexadecimal dumps for pages specified from the SPO via bus and/or page addresses.

Page Descriptor Format

The descriptor for each memory page provides the following information:

1. Bus address of dump.
2. Page address of dump.
3. Size (last physical address) of dump. (If the size field has a zero value, this means one of two things: either the disk was exhausted while dumping that page or, the disk was removed from the drive prematurely.)
4. Verbal interpretation of I.C. error status (at address @FFFC@).
5. Verbal interpretation of I.C. status 1 (at address @FFFB@).
6. Verbal interpretation of I.C. status 2 (at address @FFFE@).
7. Address at which dump of next memory page begins.

Format Of Hexadecimal Dump

Each line of a hexadecimal dump that is generated by ROMANALYZER has the following format:

<bus address>/<page address>/<hex memory address><dump><alphaversion>

where:

1. <bus address> = a single hexadecimal digit representing the bus address of the page being dumped (ranging from 1 to F).
2. <page address> = a single hexadecimal digit representing the address of the dumped page on a particular bus address.
3. <hex memory address> = four hexadecimal digits, representing the address within the specified page of the 64 bytes dumped in the given line.
4. <dump> = a hexadecimal dump of 64 bytes.
5. <alphaversion> = the character representation of the same 64 bytes. A byte that does not form a recognized ASCII character is represented by a blank.

Examples

1. ROMANALYZER <packid> - prints the descriptor and the hexadecimal dump for all pages of the ROM dump file.
2. ROMANALYZER <packid> DESC - prints only the descriptors for the pages generated on a particular disk.
3. ROMANALYZER <packid> <bus address> - prints the descriptors and hexadecimal dumps for all pages of a particular bus address.
4. ROMANALYZER <packid> <bus address> <page> - prints the descriptor and hexadecimal dump for a particular page.

ROMANALYZER Messages

While ROMANALYZER is accessing the ROM dump file, the following fatal errors may appear:

ERROR	POSSIBLE CAUSES	SUGGESTED ACTION
**ERROR-NO INITIATING MESSAGE	A packid was not specified in initiating message.	Re-initiate ROMANALYZER with a valid packid.
**ERROR-DISK NOT AVAILABLE	Cannot find disk specified or bad disk or drive.	Move disk to different drive and re-initiate ROMANALYZER.
FATAL ERROR-BUS PARAMETER TOO LONG	Space or comma missing after "DESCRIPTORS" in initiating message.	Re-initiate ROMANALYZER with proper syntax.
FATAL ERROR-PAGE PARAMETER TOO LONG	A single digit bus address (0-F) was not found after "DESCRIPTORS" in initiating message.	Re-initiate ROMANALYZER with valid bus/page address.
FATAL ERROR-INVALID BUS ADDRESS PARAMETER	Bus address not a hex digit (0-F)	Re-initiate ROMANALYZER with valid bus address.
FATAL ERROR-INVALID PAGE ADDRESS PARAMETER	Page address not a hex digit (0-F).	Re-initiate ROMANALYZER with valid page address.

The following messages denote irrecoverable errors encountered during processing of the dump file. Either the dump file was not created properly or has become corrupt.

* *ERROR-EOF READ SYSMEM, 1 * * *

* *ERROR-ERR READ SYSMEN, 1 * * *

* *ERROR-ERR SYSMEN AT.ADDR + 1 * * *

* *ERROR-EOF SYSMEN AT.ADDR 1 + 1 * * *

* *ERROR-ERR SYSMEN INITIAL SEC * * *

* *ERROR-EOF SYSMEN INITIAL SECTOR * * *

* *ERROR-BAD LEADING SECTOR * * *

* *ERROR-LEADING SECTOR NOT ZERO'D OUT * * *

* *ERROR-ERR IN FILE READ * * *

* *ERROR-EOF IN FILE READ * * *

* *ERROR-GET.RECORD READ * * *

* *EOF-GET.RECORD READ * * * DESC FOUND IN GET.RECORD

* *ERROR-TRAILING SECTOR NOT ZERO'D OUT * * *

The following non-fatal messages may be printed at the end of all information extracted from the ROM dump file. These messages represent errors that occurred during processing.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
BUS NOT FOUND	Specified bus not found in dump file.	ROMANALYZER will request a new bus address, and continue execution.
PAGE NOT FOUND	Specified page not found in dump file.	ROMANALYZER will request a new page address, and continue execution.

ROMCONVERT (ROM Dump File Converted)

The B 900/ CP9500 ROMCONVERT utility structures the ROM dump file into a format required by the SYSANALYZER utility. The output file generated by this program can then be used as input to the SYSANALYZER utility. The SYSANALYZER utility requires a system dump file generated by system firmware. It cannot analyze a file generated by the ROM dump routine.

NOTE

SYSANALYZER is only capable of analyzing the converted dump file if: 1) system software was under MCP control prior to the ROM dump being taken; or 2) certain critical memory locations are in the dump file and are not corrupt.

ROMCONVERT can be used on any CMS system. It executes as a normal user job. The syntax for executing ROMCONVERT is:

ROMCONVERT ⟶

ROMCONVERT uses, as input, a removable disk on which a ROM dump file has been generated. The dump file begins at the start of the first sector after the highest addressable directory structure. The presence of a ROM dump file is not reflected in the disk's directory.

The ROMCONVERT utility requests the packid of the source disk. The following messages appear on the SPO:

```
<mix #>/ROMCONVERT DISP:
      ENTER SOURCE PACKID
<mix #>/ROMCONVERT ACPT
```

The user responds, via the AX intrinsic, with a packid (conforming to CMS requirements for a packid).

The utility's output consists of a file that can be analyzed by the SYSANALYZER utility. The file's name is specified by the user.

The packid and file-id are requested separately:

```
<mix #>/ROMCONVERT DISP
    ENTER DESTINATION PACKID
```

<mix #>/ROMCONVERT ACPT
(respond with AX <mix #> <packid>)
<mix #>/ROMCONVERT DISP
ENTER DESTINATION FILEID
<mix #>/ROMCONVERT ACPT
(respond with AX <mix #> <fileid>)

The occurrence of any error sends the ROMCONVERT utility to EOJ.

During the processing of the source disk, one of the following error messages may be displayed:

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
NO CHARACTERS IN INPUT	No packid specified.	Check syntax of packid and re-enter.
INPUT TOO LONG	More than seven characters entered for packid or 12 characters for a fileid.	Check input and re-enter.
INVALID CHARACTER IN INPUT	Invalid packid entered.	Check syntax of packid and re-enter.

The following messages denote irrecoverable error conditions encountered during processing of the ROM dump file:

INPUT READ ERROR
OUTPUT WRITE ERROR
EOF ON INPUT
(*) UNABLE TO OPEN SYSMEM
TOO MANY BAD AREAS ENCOUNTERED
INVALID NEXT DESCRIPTOR
OS PROCESSOR NOT FOUND
BAD DESCRIPTOR LOCATED
INVALID PAGE
INVALID BUS

* This message may mean that the ROMCONVERT utility is terminating because no source disk with the specified packid is present. This is the only condition that the user can rectify.

FPP (Field Patch Program)

The B 900/CP9500 Field Patch Program (FPP) is designed to provide a means to update the B 900/CP9500 release disk (B900LR1) with system patches. It is an interactive MPLII program, which requires no special mix and may be used on any CMS system. The FPP displays messages to the user and accepts responses, via the system AX intrinsic. The input to FPP is a patch supplied by the system house. Once the patch has been applied to the B 900/CP9500 release disk, the patch or replace function of coldstart can be used to incorporate changes on the patched release disk into a system.

Format:

FFP ⟶

To protect system security, the FPP makes no actual changes to the B 900/CP9500 release disk until all of the following conditions are true:

1. All of the following files must be present on the release disk:
   SYSMCP
   SYSBOOTSTRAP
   SYSINITBOOT
   SYSCOLDSTART
   SYSDSCP
   SYSMPLII
   SYSICP
   SYSWARMSTART
2. Every file that is present on the release disk must be at the patch level specified in the supplied patch.
3. The patch must be entered correctly.

NOTE

The COBOL interpreter (SYSCOBOL) need not be on the release disk, but if present, it must be at the correct patch level in accordance with condition 2.

The B 900/CP9500 FPP begins execution by displaying the following message on the SPO:

<mix #>/FPP DISP: ENTER AUTHORIZATION CODE

This is followed by an ACCEPT message requesting the user to enter the authorization code supplied with the patch. The code is entered in the following format:

AX <mix #> <check code> <old version string> <new version string>

where:
1. <check code> = two groups of four hexadecimal digits (nnnn nnnn).
2. <old version string> = fourteen alphanumeric characters specifying the current release level of the files on the release disk.
3. <new version string> = fourteen alphanumeric characters specifying the version string that is to be given to each file on the release disk if all files have been patched successfully.

Every required file on a release disk must have the same version string as the <old version string> supplied in the authorization code. The presence of any version string that does not match the one in the authorization code causes FPP to terminate after displaying the message:

<mix #>/FPP DISP: INVALID RELEASE DISK

Entering The Patch

If the FPP finds that the release disk is valid (every system file is present and has the proper release and patch level), it goes on to accept the patch from the user. To request the first line of the patch, FPP displays the following message on the SPO:

<mix #>/FPP DISP: ENTER PATCH (FILE SEG ADDR PATCH)

This message is not repeated for further patch lines; FPP merely continues to do ACCEPTs until the user responds with AX <mix #> END.

Format Of Patch Line

Each line of the patch is entered in the following format:

AX <mix #> <file> <addresss> <patch>

where:

1. <file> = a two-character file mnemonic, which represents one of the system files:

MNEMONIC	FILE
OS	SYSMCP
BT	SYSBOOTSTRAP
BZ	SYSINITBOOT
CB	SYSCOBOL
CS	SYSCOLDSTART
DP	SYSDSCP
MP	SYSMPLII
TP	SYSSICP
WS	SYSWARMSTART

2. = a segment number composed of two hexadecimal digits. (It must equal zero for files other than SYSMCP.)
3. <address> = four hexadecimal digits, composing an address that is within the range of the segment. (FPP does not check whether it is in range.)
4. <patch> = an even number of hexadecimal digits (no more than forty).

Blanks may be added or omitted to aid in readability. If an incorrect patch line is entered, it is rejected by the FPP.

Patching the Release Disk

Once the FPP has determined that the patch is correct, it updates the version string and enters the patch into the files.

NOTE

To incorporate the patched files onto the system disk, the patch or replace function of coldstart must be used.

FPP Messages

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
AUTHORIZATION CODE ERROR	Incorrect check code.	Obtain correct code from Burroughs representative.
ILLEGAL INPUT CHARACTER	An invalid character was detected in the input.	Check patch and re-enter.
INPUT TOO LONG		Check patch and re-enter.
INPUT INCORRECT LENGTH	Patch format error.	Check patch and re-enter.
INVALID RELEASE DISK	Versions strings do not match (see text).	(Refer to text.)
INVALID PATCH RE-ENTER PATCH		
CANNOT FIND PRIME-RESIDENT SEGMENT	Release disk has become corrupt.	Use backup copy.

MESSAGE	POSSIBLE CAUSES	SUGGESTED ACTION
INVALID SEGMENT NUMBER (MUST BE 00 FOR NON OS)	Error in format.	Check patch and re-enter.

CONFIGURER

When the system is warmstarted, its configuration characteristics are determined from information found in the SYSCONFIG file. The configurer utility allows the user to create the SYSCONFIG file, update it, or list its contents. The SYSCONFIG file is used:

1. By the B 900/CP9500's warmstart code to determine the system's logical configuration as desired by the user.
2. For parameters that must be maintained across warmstarts.

Configurer is an MPLII program that runs on any CMS system. Its code may be executed from any disk. Configurer displays messages to the user, and accepts responses, via the system SPO using the CMS AX intrinsic. These responses determine its actions.

NOTE

The SYSCONFIG file is instrumental in determining how the system's resources are allocated for optimal use and performance. Therefore, it is strongly recommended that only those persons familiar with the particular system's physical characteristics be allowed to alter the SYSCONFIG file. A Burroughs field engineer can assist in identifying the system's physical characteristics.

SYSCONFIG Layout

Configurer views the SYSCONFIG file as having three major areas, each consisting of at least one level of substructure. This substructure may take the form of other areas, fields, or a mixture of both. SYSCONFIG's three major areas are:

1. LOGINFO. The common area containing SPO and maintenance logging related information.
2. MISCINFO. The common area containing startup, runtime, and power off information.
3. B900INFO. The system dependent area containing information on the operating system, peripherals, data comm subsystem and warmstart.

NOTE

A "field" is a container of an elementary data item. An "area" consists of two or more fields and/or areas. All areas and fields have names.

Hierarchy is indicated via indentation. That is, if name X is indented to a given level, every name immediately after it that is indented further, names an area/field contained in area X. If no names are indented following name X, then X is a field.

The following summarizes the structure of SYSCONFIG:

LOGINFO

- LOGGING
- #LOGFILES
- LOGFILESIZE

} System-Independent Area

MISCINFO

- ZIPTEXT
- POWEROFFMSG
- SMDICTNAME
- PBFILESIZE

} System-Independent Area

```
B900INFO
  OSINFO
        REMOTEFLAG
        MAXMIX
        VM
  PERIPHINFO
        PRNTR.TRANSL
        SOFTVFU
  DCINFO
        DCBUFFER
        NDLSYSFILE
        SYSRECONFILE
  B900CONFIG
        BUFFERMEM
        PROCINFO
              TP ASSIGNMENT
                    TPASSIGNS
                    INTERPRETERS
                    MEMSIZE
              DCP ASSIGNMENT
                    DCPBUSADDR
                    LOGICALDCP
```

System-Dependent Area

Sample SYSCONFIG File

Since a SYSCONFIG file is required for operation of the MCP, a sample SYSCONFIG file is provided with the B 900/CP9500 CMS 3.02 Software Release.

This sample is used as is until the user MAKEs a new file or FIXes (that is, updates) the sample.

NOTE

Whenever the SYSCONFIG file is altered, the system must be warmstarted in order to invoke the changes.

The contents of the sample SYSCONFIG file are given under LIST Mode.

Initiation

The initiating syntax for configurer is as follows:

CONFIGURER ⟶

When configurer starts, the following messages appear on the SPO:

<mix #>/CONFIGURER BOJ PR IS X

(where X = configurer's CMS priority: A, B, C)

<mix #>/CONFIGURER DISP

CP9500 CONFIGURER UTILITY - <release level information>

Mode Selection

Configurer has three modes of operation: MAKE, FIX, and LIST. At startup, or when a mode has just finished, configurer displays the following messages:

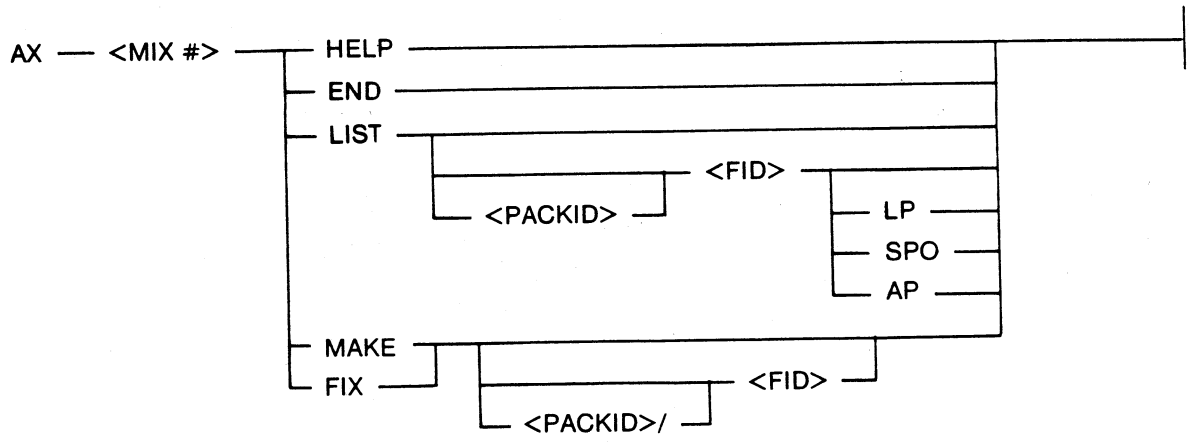
<mix#>/CONFIGURER DISP

CONFIGURER OPTIONS ARE: MAKE, FIX, LIST, HELP, END

SELECT CONFIGURER OPTION <mix#>/CONFIGURER DISP

<mix#>/CONFIGURER ACPT

Any user response that can be derived by following this railroad diagram from left to right is expected:



Entering HELP causes configurer to display syntax diagrams for the selection of MAKE, FIX, or LIST modes. The request for the selection of a configurer option is then repeated. Entering END sends configurer to EOJ.

When selecting LIST, MAKE or FIX, a filename may be specified. If no filename is specified, then the default is SYSCONFIG. If no packid is specified, the system disk packid (seven ASCII zeroes) is used as the default. Therefore, if no packid or filename is specified, then the default is 0000000/SYSCONFIG.

LIST Mode

Selecting configurer's LIST mode causes configurer to find SYSCONFIG file that has the specified (or default) file name, and list its contents on the SPO, the line printer, or whatever printer becomes available.

Only LOGINFO, MISCINFO, and B900INFO are formatted for the user's convenience. Any other areas (that is, system dependent areas for other CMS machines) are dumped in hexadecimal.

An example of LIST mode's output follows. This example shows the values of the Sample SYSCONFIG File discussed previously.

Sample SYSCONFIG File

```
LIST OF          : B900RL1/SYSCONFIG
ON               : 09/02/80
AT               : 15:45:59
SYSCONFIG AREA   : LOGINFO
FIELD            : MEANING/CONTENTS
LOGGING          : YES
#LOGFILES        : 04
LOGFILESIZE      : 00064
SYSCONFIG AREA   : MISCINFO
FIELD            : MEANING/CONTENTS
ZIPTEXT          : 〈 EMPTY FIELD 〉
POWEROFFMSG      : SYSTEM DISK POWERED OFF
SMDICTNAME       : SYSLANGUAGE
PBFILESIZE       : @00270F@
SYSCONFIG AREA   : B900INFO
```

```
B900INFO AREA        : OSINFO
FIELD                : MEANING/CONTENTS
REMOTEFLAG           : YES
MAXMIX               : @10@
VM                   : A
B900INFO AREA        : PERIPHINFO
FIELD                : MEANING/CONTENTS
PRNTR.TRANSL         : NO TRANSLATION
SOFTVFU              : A
B900INFO AREA        : DCINFO
FIELD                : MEANING/CONTENTS
DCBUFFER             : @000000@
NDLSYSFILE           : 0000000/NDLSYS
SYSRECONFILE         : SYSRECON
B900INFO AREA        : B900CONFIG
FIELD                : MEANING/CONTENTS
BUFFERMEM            : @002000@
B900CONFIG           : TP ASSIGNMENTS
                     : #  INTERPS MEM
                       4  COBOL @040000@
                          MPLII
                       1  COBOL @020000@
                          MPLII
B900CONFIG           : DCP ASSIGNMENTS
                     : BUS LOGICAL
                         8      0
```

MAKE Mode

Selecting the configurer utility's MAKE mode causes the following to occur:

1. A workfile is created to serve as a temporary container for the SYSCONFIG file.
2. The configurer program displays messages that lead the user through the areas and fields of the SYSCONFIG file by successively naming areas and fields until a complete SYSCONFIG file is generated.

This causes MAKE mode to name an area, the first area subsidiary to that area, and so on until a field is reached. A value is then requested for the field, and for all subsequent fields within that lowest area. Processing of a field does not stop until a legal value has been supplied for that field; processing of an area is not finished until all fields or areas subsidiary to it have been processed.

3. After all fields have been processed, the workfile is closed with REMOVE. It is given the filename which the user specified (or allowed to default) in the message that selected MAKE mode.
4. The user is returned to the SELECT CONFIGURER OPTION level to select another mode or to select HELP or END.

To name a field or area, MAKE mode displays the following message on the system SPO:

<mix#>/CONFIGURER DISP: CURRENT AREA IS : <field/area name>

To request a value for a field (that is, a container of an elementary data item), MAKE mode displays the following message on the SPO:

<mix#>/CONFIGURER DISP: ENTER <field name> <value choice>

The following describes the fields of SYSCONFIG and the values which may be assigned to them while using either MAKE or FIX mode. Generally, null input is valid, and causes a default value to be assigned to the field. Where valid, the default value is given in parentheses (that is, default =).

LOGINFO

The first area of a SYSCONFIG file is system independent. It contains information that concerns the logging of SPO messages. This area is named LOGINFO.

· The LOGINFO area consists of three fields:

LOGGING
#LOGFILES
LOGFILESIZE

When specifying a value for a field in the LOGINFO area, please observe the following syntax:

LOGGING	Enter YES (Y) or NO (N). The LOGGING field indicates whether SPO messages are to be recorded in log files. (Default = NO.)
#LOGFILES	Enter a decimal integer between 3 and 16. The #LOGFILES field indicates how many log files the CMS super utility should create. The names of the logfiles are SYS-LOG-01 SYS-LOG-02, and so forth, up to SYS-LOG-nn, where nn is the number of logfiles requested. (Default = 3.)
LOGFILESIZE	Enter a decimal integer between 1 and 16383. The LOGFILESIZE field specifies the size of each logfile in sectors. (Default = 32.)

MISCINFO

The second area of a SYSCONFIG file is system independent. It contains several types of information. Its name is MISCINFO. The MISCINFO area consists of four fields:

ZIPTEXT
POWEROFFMSG
SMDICTNAME
PBFILESIZE

When specifying a value for a field in the MISCINFO area, please observe the following syntax:

ZIPTEXT	Enter from 0 to 255 valid ASCII alphanumeric characters. The message in ZIPTEXT must constitute a valid SCL command. The ZIPTEXT field (if not null) contains the message that the CMS super utility is to ZIP as soon as it starts executing (for example, the name of a program to be executed). (Default = <empty>.)
POWEROFFMSG	Enter from 0 to 80 valid ASCII alphanumeric characters. The message supplied in the POWEROFFMSG field is to be displayed on the SPO at the time that the system disk is powered off (it is therefore the last message that the system displays). (Default = <empty>.)
SMDICTNAME	Enter a valid CMS <file-id>. System messages to the SPO are generated by using a system message dictionary file. The SMDICTNAME field specifies the name of the dictionary file that is to be used. (Default = SYSLANGUAGE.)
PBFILESIZE	Enter a decimal integer between 1 and 1048560. All printer backup files generated on a CMS system have the same size limit (in bytes); this limit is specified in the PBFILESIZE field. (Default = 0.)

B900INFO

The B 900/CP9500 system dependent area (B900INFO) must be present in any SYSCONFIG file used to warmstart a B 900/CP9500.

This area consists of four subsidiary areas:

1. Operating System Information (OSINFO).
2. Information about peripherals (PERIPHINFO).
3. Data Comm Subsystem Information (DCINFO).
4. Warmstart information (B900CONFIG).

These areas are described in the following paragraphs.

OSINFO

The B900INFO area's first subsidiary area contains information that concerns the operating system. Its name is OSINFO. The OSINFO area consists of three fields:

REMOTEFLAG
MAXMIX
VM

When specifying a value for a field in the OSINFO area, please observe the following syntax:

REMOTEFLAG	Enter one of the following: YES; Y; NO; N. The REMOTEFLAG field indicates whether the system is to allow a Message Control System (MCS) program to act as the SPO. (This allows a data comm device to be the remote SPO.) (Default = NO.)
MAXMIX	Enter a decimal integer between 1 and 16. The MAXMIX field sets a limit on how many jobs can be in the mix at one time. (Default = 11).
VM	Enter a null entry. This field is not currently used by the CP9500.

PERIPHINFO

The second area subsidiary to B900INFO is named PERIPHINFO. It contains information concerning peripherals (other than data comm and disk devices).

This area consists of two fields:

PRNTR.TRANSL
SOFTVFU

When specifying a value for a field in the PERIPHINFO area, please observe the following syntax:

PRNTR.TRANSL — The PRNTR.TRANSL field names the translation table to be used for the printer. (Default = NO TRANSLATION.) Enter one of the following table names:

VERSION 1

AUSTRALIA
CANADA
KOREA
NEW ZEALAND
USA

VERSION 2

BAHAMAS
BARBADOS
BERMUDA

VERSION 2 (cont.)

CENTRAL AFRICA
CEYLON
EAST AFRICA
FIJI
GHANA
GUYANA
HOLLAND
HONG KONG
INDONESIA
JAMICA

VERSION 2 (Cont.)

KENYA
MALAYSIA
MALTA
NETHERLANDS
NIGERIA
NORTHERN RHODESIA
RHODESIA
SINGAPORE
SOUTH AFRICA
TAIWAN

VERSION 2 (Cont.)

THAILAND
TRINIDAD
UNITED KINGDOM
USA

VERSION 3

ALGERIA
BELGIUM
FRANCE
FRENCH GUIANA
FRENCH WEST INDIES
MALAGASY REPUBLIC
MOROCCO
TUNISIA
WEST AFRICA
ZAIRE

VERSION 4

ITALY

VERSION 5

AUSTRIA
CZECHOSLOVAKIA
GERMANY
SWITZERLAND

VERSION 6

BRAZIL
PORTUGAL
PORTUGUESE WEST AFRICA
PORTUGUESE EAST AFRICA

VERSION 7

ARGENTINA
CHILE
COLUMBIA
COSTA RICA
DOMINICAN REPUBLIC
ECUADOR
HONDURAS
MEXICO
NICARAGUA
PANAMA
PERU
PHILIPPINES
PUERTO RICO
SPAIN
URUGUAY
VENEZUELA
USA

VERSION 8/8A

DENMARK
NORWAY

VERSION 9

YUGOSLAVIA (CROATIAN)

VERSION 10

FINLAND
SWEDEN

VERSION 14A

TURKEY

VERSION 16A

ICELAND

VERSION 24

KATAKANA

The above tables are to be used for the B9249-2/3/4 line printers.

The B9249-30/50 requires translation only for the following countries:

Denmark, Norway	–	Specify version 8A in SYSCONFIG
Turkey	–	Specify version 14A in SYSCONFIG
Iceland	–	Specify version 16A in SYSCONFIG

SOFTVFU — The SOFTVFU field contains parameters that the MCP downloads to the printer's vertical formatting unit (VFU). The VFU is a programmable substitute for a printer's format tape. (Default = A.)

DCINFO

The B900INFO area's third subsidiary area contains information that concerns the data comm subsystem. Its name is DCINFO.

This area consists of three fields:

DCBUFFER
NDLSYSFILE
SYSRECONFILE

When specifying a value for a field in the DCINFO area, please observe the following syntax:

DCBUFFER — Enter a decimal number. It may be an integer (for example, 12), or it may include a fractional part (for example, 12.3). It may not exceed 1024. The DCBUFFER field specifies how many kilobytes (1 KB = 1024 bytes) of the system's buffer memory must be dedicated to data comm. (Default = 0.)

NOTE

This specifies how much of the system's buffer memory is to be reserved for data comm. The total amount of buffer memory is determined by BUFFERMEM (in the B900CONFIG area). Whenever DCBUFFER or BUFFERMEM is specified, configurer checks if both values have been entered, that the amount of DCBUFFER to be reserved is less than the total amount of BUFFERMEM by at least 2 KB (that is, that (BUFFERMEM - DCBUFFER) > 2). If this is not true, configurer requests new values for both fields after displaying the following message on the SPO:

* * INVALID BUFFERMEM SPECS * *

NDLSYSFILE — Enter a valid CMS file name (that is, <packid>/<file-id>). Do not enter more than 20 characters. NDLSYSFILE contains the file name of the NDLSYS file that is to be used in setting up the CMS Data Comm Subsystem. (Default = 0000000/NDLSYS.)

SYSRECONFILE — Enter a valid CMS <file-id>. Do not enter more than 12 characters. SYSRECONFILE specifies the file-id of the system reconfiguration file. Any data comm configuration changes made during operation are recorded in the reconfiguration file. (NDLSYS is a read-only file.) (Default = SYSRECON.)

B900CONFIG

The system dependent area ends with an area that contains information needed to warmstart the physical system. This area specifies which physical resources are needed, and how they are to be used. B900CONFIG begins with a field:

BUFFERMEM — Enter a decimal number. It may be an integer (for example, 62) or it may include a fractional part (for example, 62.3). It may not exceed 1024. BUFFERMEM specifies how many kilobytes (1 KB = 1024 bytes) should be allocated for system buffer memory use.

This is followed by the Warmstart Table, which specifies how the system's task processors are allocated.

The configurer utility handles the Warmstart Table as an area named PROCINFO, containing the following areas and fields:

PROCINFO
- TP ASSIGNMENT
  - TPASSIGNS
  - INTERPRETERS
  - MEMSIZE
- DCP ASSIGNMENT
  - DCPBUSADDR
  - LOGICALDCP

When specifying a value for a field in the PROCINFO area, please observe the following syntax:

PROCINFO

For each task processor, or set of identical task processors, enter TP and specify the necessary values for the fields in the TP ASSIGNMENT area. For each data comm processor, enter DCP and specify the necessary values for the fields in the DCP ASSIGNMENT area. To cause configurer to make a Warmstart Table for the default configuration, enter DEFAULT without specifying TP or DCP. To stop specifying processor assignments, enter PROCEND.

TPASSIGNS

Enter a decimal integer from 1 to 6. Depending on the system, there can be up to six task processors on the system. Note that if there is no requirement for the DCP (data communications processor on the system), the DCP can be used as a task processor. All task processors that are to have identical attributes can be assigned in one pass through the TP ASSIGNMENT area. The TPASSIGNS field specifies how many TPs are being specified; INTERPRETERS names the interpreters that each of these TPs is to support; MEMSIZE specifies the minimum amount (in kilobytes) of local memory that each of these TPs must own. (One page contains 64 KB.)

INTERPRETERS

Enter one of the following: MPLII; COBOL (that is, COBOL and/or RPG); ALL (that is, same as specifying MPLII and COBOL); ENDINTERP. (Additional INTERPRETERS values are requested until ENDINTERP is specified, except for the ALL specification.)

MEMSIZE

Specify a decimal number. It must be between 64 and 1024. It may be an integer (for example, 199) or it

NOTE

The upper limit for the MEMSIZE parameter corresponds to 16 pages of local memory. This reflects the address capacity of the hardware registers. Currently, a task processor is not allowed to own more than four pages, which sets a practical limit of 256.0 for the MEMSIZE parameter. Task processors supporting more than one interpreter must have a minimum MEMSIZE of 128 KB.

The DCP ASSIGNMENT area specifies the bus address of each data comm processor, and the logical DCP number associated with bus address. When the CMS data comm subsystem is loaded, the logical DCP number determines which DCP firmware file is loaded into a given data comm processor. For each DCP, a fresh pass through the DCP ASSIGNMENT area must be made.

DCPBUSADDR. Enter a decimal integer between 1 and 15.
LOGICALDCP. Enter a decimal integer between 0 and 7.

NOTE

When updating the PROCINFO area, both TP and DCP assignments must be made (that is, assigning TPs only causes no DCPs to be assigned).

FIX Mode

Selecting FIX mode causes the following to occur:

1. Configurer finds SYSCONFIG file having the specified (or default) filename.
2. A workfile is opened to serve as a temporary SYSCONFIG container and the contents of the old SYSCONFIG file are copied into the work file.
3. If a SYSCONFIG file does not have a system dependent area, FIX mode creates one. As in MAKE mode, the user is requested to supply responses for all fields primed until a complete system dependent area is generated.
4. If a SYSCONFIG file already has a system dependent area, FIX mode allows the user to select specific fields in any of the three major areas in a SYSCONFIG file for updating.
5. When FIX mode finishes, it closes the work file with REMOVE, giving it the specified (or default) file name.

When updating a complete SYSCONFIG file, the user is allowed to select specific fields. This is done by presenting the user with a choice, one level at a time, of all areas/fields that are subsidiary to the area in which the user currently is.

The choice is offered by displaying the messages:

```
<mix #>/CONFIGURER DISP
        <area-name> OPTIONS ARE: <subsidiary area/field names>
<mix #>/CONFIGURER DISP
        SELECT <area-name> OPTION
```

Selecting an area name causes FIX mode to enter that area, and repeat the message, this time listing the areas/fields subsidiary to the area the user selected.

Selecting a field name causes FIX mode to enter that field and request a value for it. This process is the same as when MAKE mode requests a value for a field; the same message is used:

```
<mix #>/CONFIGURER DISP
ENTER <field-name> <choice of values>
```

When the user is presented with a choice of areas/fields, entering END causes FIX mode to back out to the next most general level. To leave FIX mode, continue responding with END until the message: "CONFIGURER OPTIONS ARE: MAKE, FIX, LIST, HELP, END" is issued. (Entering END in response to this message sends the entire configurer utility to EOJ.)

OS And Data Comm Buffer Memory Validation

In addition to the basic validity checks applied to any other single field, the utility verifies both the operating system and data comm buffer memory specifications against each other to ensure their compatibility. This is done in:

MAKE Mode: At the time the OS buffer specification is received.
FIX Mode: When either the data comm buffer or OS buffer specification is being updated.

Ensure that the difference between the (OS buffer - dc buffer) is greater than or equal to the OS buffer minimum, where the OS buffer minimum is equal to 2000 bytes. Failing this verification results in configurer issuing prompts for the re-specification of both buffer sizes.

The following table lists the maximum amount of buffer memory that can be specified in the SYSCONFIG file based on the amount of memory available on the OS.

MEMORY SIZE	MEMORY USAGE
128KB	96KB required for MCP. 32KB available for buffer memory.
192KB	128KB required for MCP. 64KB available for buffer memory.
256KB	128KB required for MCP. 128KB available for buffer memory.

Processor Configuration

The initial selection list for processor assignment allows:

1. The initiation of a task processor (TP) assignment sequence.
2. The initiation of a data comm processor (DCP) assignment sequence.
3. The specification that the default assignments are to be made at warmstart time.
4. The specification that the user processor assignments are complete (that is, PROCEND).

If the default processor assignments scheme is sufficient, merely entering default completes the processor assignment. This causes the warmstart defaults to be used.

If discrete user assignments are desired, there are two forms of processor assignment types defined: TP and DCP. Each form is implemented in a multi-response form. Each of the response parts has a respective prompt by configurer indicating its attributes, prior to the issuing of the accept for the user input.

The TP and DCP assignments may be utilized in any order, and the required number of times, such that the desired configuration is completely specified. This is the logical system allocated at warmstart time, given that the physical system has the capability of supporting the configuration.

NOTE

If the logical TP and DCP assignments do not match that of the physical system configuration the system will, at warmstart time, come up under its default configuration. Warmstart default is not the same as the SYSCONFIG default. Refer to Warmstart for the exact parameters.

Do a LIST before ending the configuration procedure to verify that changes have been made correctly.

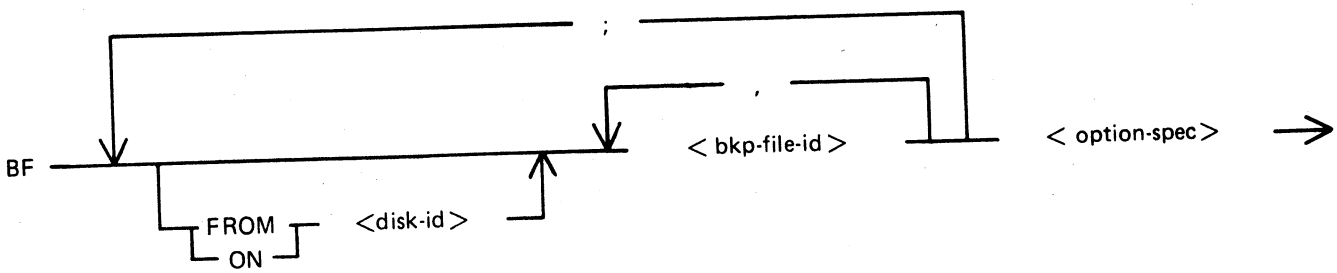
SECTION 10
B 1800/B 1900 DEPENDENT SYSTEM SOFTWARE

CMS-UTILITIES

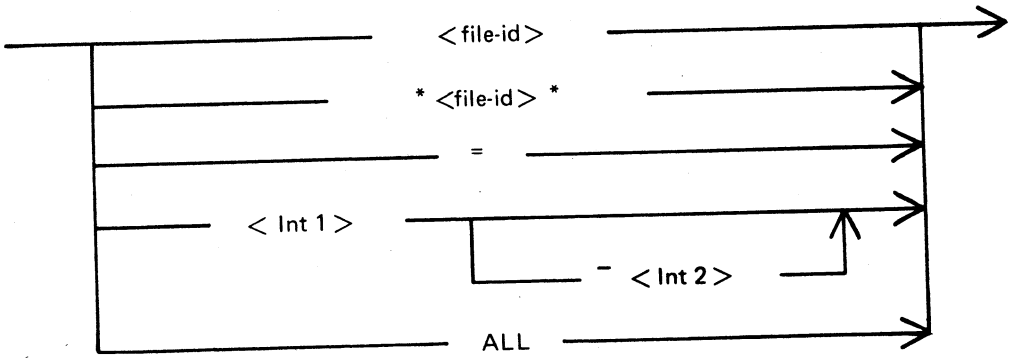
BF

This utility enables the user to display information about printer backup files residing on one or more disks.

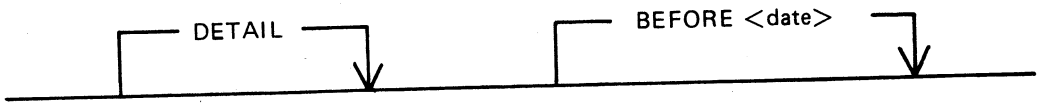
BF HELP will display the syntax diagram for BF, which has the following format:



where <bkp-file-id> is defined as



and <option-spec> is defined as



<disk-id> specifies the disk on which the utility seeks backup files; default is the backup designated disk.

The following specifications designate the files or groups of files about which the information is required:

<file-id> designates a backup file or a family of backup files. Either the external or the internal file name can be specified.

* <file-id>	* is used when <file-id> alone could cause confusion (for example BF 5 designates the backup file named 5 instead of the one named PB00005).
=	specifies the backup files PBxxxxx, where 1 <= xxxxx <= 65535.
<Int 1>	specifies the backup file PB<Int 1>.
<Int 1> - <Int 2>	specifies all the backup files from PB<Int 1> to PB<Int 2>.
All	specifies all backup files irrespective of their name.
DETAIL	option causes BF to display, for each backup file specified: the external name of the backup file the internal name of the backup file the name of the file in the backup file the file size the date and time of creation the name of the creator program The default for DETAIL is to display <file-id> ON LINE for each file found within the range specified.
BEFORE <date>	option causes all backup files created before the specified date to be listed; <date>::= < MM /DD /YY> MM and DD (month and date respectively) can be single digits.

Examples

To specify the file PB00006 from the system disk:

BF 6

To specify the file PB00026 and all backup files in the family MPL=, on the disk USEDSK:

BF FROM USEDSK 26, MPL=

To specify PB00016,PB00019 on the system disk, all the backup files named PBxxxxx on the disk TASK and all backup files in the range PB00040 to PB00063 on the disk ARDSK, with each one of these files being subject to the date given:

BF 16,19 ; FROM TASK = ; FROM ARDSK 40-63 BEFORE 1/6/81

Messages

The syntax error messages issued by the utility are self-explanatory. The utility goes to End of Job after displaying 'BF ABORTED', and advising the user to use the BF HELP facility.

In the case where the utility issues a message mentioning quotes, the user should read 'asterisk' instead of 'quote'.

If a disk cannot be opened, BF continues executing and goes on to the next disk specified, after displaying:

UNABLE TO ACCESS DISK <disk-id>

If a unique file cannot be found, or no files are found in a family, BF goes on to the next file specified, after issuing the appropriate messages concerning the absent files.

After completion, BF issues the message:

* * BF COMPLETE * *

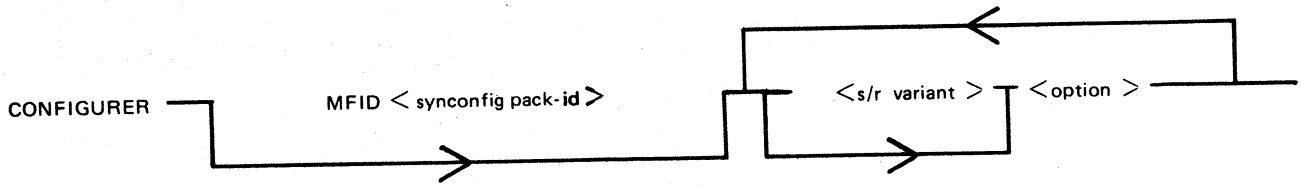
CONFIGURER

Introduction

This utility creates and modifies the file 'SYSCONFIG'. Much of the information stored in this file is used at CLEARSTART time. Therefore, these options do not take effect until the next CLEARSTART. Information related to the Data Communications Subsystem (DCSS) is used at DCSS initiation. The SYSCONFIG file must be present on the system disk in order to CLEARSTART.

Syntax

The format of the initiating message is as follows:



The MFID option specifies the location of the SYSCONFIG file.

<s/r variant>:

SET	Causes an option to be set on.
RESET	Causes an option to be set off.
SO	Equivalent to SET.
RO	Equivalent to RESET.

<option>:

CREATE	This option enables the user to create the file SYSCONFIG. Any existing file on the required pack is then removed, unless its file type value is not the standard @22@ file type value. In this case, a warning message is issued (see messages). The CREATE option is always the first executable option. SET and RESET are not valid for it.
DISPLAY	Causes a description of the SYSCONFIG file to be displayed on the SPO. SET and RESET are not valid for this option.
PRINT	Causes a listing describing the SYSCONFIG file to be generated. SET and RESET are not valid for this option.
WRITE	Causes a backup file to be created with the description of the SYSCONFIG file. SET and RESET are not valid for this option.
LIST	Causes a description of the file SYSCONFIG to be either listed or written into a backup file. SET and RESET are not valid for this option.
TEACH	Causes the syntax diagram for CONFIGURER to be printed. SET and RESET are not valid for this option.
HELP	Causes the syntax diagram for CONFIGURER to be displayed on the SPO.

SET and RESET are not valid for this option.

DATE

TIME

It is advisable to keep these options set on in order that the MCP maintains the correct system date and time.
SO DATE and/or SO TIME cause the date and/or time messages to be displayed on the SPO at CLEARSTART. The values given by the user as replies to these prompts are used as initial values for the system time.
RO DATE and/or RO TIME causes the system time to be initialized with the values contained in the SYSCONFIG file. These values are the actual system data and time at the last resetting of the options.

MESSAGE FILE <file name>

Enables the user to specify the desired message file.

NBLOG <num>

Controls the number of log files to be initiated at CLEARSTART. <num> is a number from 3 to 16. SET and RESET are not valid for this option.

LOGSIZE <num>

Sets size in sectors of each log file. <num> is a number from 32 to 4096. Log file size is set at CLEARSTART. SET and RESET are not valid for this option.

LOG

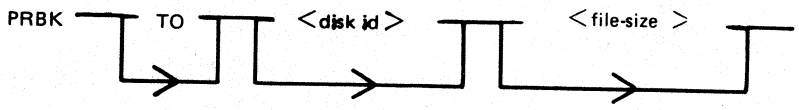
Controls whether the logging facility will be initiated at CLEARSTART.
SO LOG causes logging to be initiated at CLEARSTART.
RO LOG inhibits logging, except the logging of hardware maintenance entries. When this option is reset, log file information is not displayed by a CONFIGURER PRINT or DISPLAY. The message 'SPO LOGGING OFF' appears.

ZIP '<text>'

Indicates a message, <text>, to be ZIPPED after CLEARSTART.
'<text>' denotes the message to be ZIPPED and must be a quoted string. SO ZIP '<text>' sets the message to be ZIPPED.
RO ZIP resets the message to null. When this option is reset, no ZIP message will appear in a CONFIGURER PRINT or DISPLAY.
This facility is very convenient as it can be used to zip an MCS automatically after each CLEARSTART.

PO MESSAGE '<text>'

Controls the message to be displayed at System disk power-off time.
<text> is a quoted string denoting the desired message.
SO PO MESSAGE '<text>' sets the message to be displayed.
RO PO MESSAGE resets the message to null.
When this option is reset, no PO MESSAGE information will appear in a CONFIGURER PRINT or DISPLAY.



Controls how the backup file option is set at CLEARSTART.
<disk-id> denotes the pack on which backup files are to be created and searched for.
<file size> denotes the maximum size of a backup file.
SO PRBK... sets the backup file option at CLEARSTART causing backup files to be created.

RO PRBK... resets the backup file option at CLEARSTART.
When this option is reset, backup file information is not displayed by a CONFIGURER PRINT or DISPLAY.
When creating a printer backup file, the <file size> contained in SYSCONFIG will be used IF AND ONLY IF the file size of the output file originally directed to the printer has a value of 0.
Any other value will be used as a file size for the printer backup file and may cause an unexpected DS/DP of the program when that file size is exceeded.

COUNT — Controls the CLEARSTART count. The only valid operation for this option is RO COUNT, resetting the count to zero.

MASK <@XXXX@> — Where @XXXX@ is a 4 digit hexadecimal number representing the Channel Mask.
RO MASK will supply the value @FFFF@ to the SYSCONFIG Channel Mask (all channels will be enabled).
SO MASK @XXXX@ gives the value XXXX to the Channel Mask. This represents, from right to left, the numbers of the channels which are to be kept.
Example: SET MASK @0203@ enables the MCP to handle only channels 0,1 and 9 (the mask setting corresponding to the binary setting @0000001000000011@).
This setting is stored in the SYSCONFIG file and recalled at each CLEARSTART time unless another mask value has been supplied via the RO MASK option.
Note that setting the maintenance panel toggles at CLEARSTART time, with the value @FA@ in the first two digits, and a mask value in the last four digits, takes precedence over the SYSCONFIG mask.

The remaining options become effective at DCSS initiation.

DCSIZE <num> — Specifies the amount of memory allocated for the DCSS message area in K bytes.
<num> must be in the range 5 to 310.
SO and RO are not valid for this option.

DCPDUMP <num> — Controls whether the memory of the DCP specified by <num> is to be dumped to a file at DCSS initiation. <num> must be in the range 0 to 4. The dump will be contained in the file DUMPDCP<num>, where <num>, as above, is the number of the DCP.
SO DCPDUMP <num> will cause a dump to be created.
RO DCPDUMP <num> inhibits a dump. When this option is reset, no DCP dump information will be displayed in a CONFIGURER DISPLAY or PRINT.

DCCHTRACE — Controls whether the DCSS debugging tool, trace, is to be initialized at DCSS initiation. Setting this option causes a printer file to be opened at DCSS initiation. Debugging information is printed while DCSS is running. SO DCCHTRACE causes the trace to occur. RO DCCHTRACE inhibits the trace.

REMOTEFLAG — Controls whether remote SPO activities will be allowed or inhibited.
SO REMOTEFLAG sets the REMOTEFLAG at DCSS initialization, allowing REMOTE SPO activity.
RO REMOTEFLAG resets this flag at DCSS initialization, inhibiting REMOTE SPO activity.
When this option is reset, the message 'REMOTE SPO OFF' appears in a CONFIGURER DISPLAY or PRINT. (For more information on REMOTE SPO, see DCSS section.)

REMOTECHAR <char> — Specifies the control character for REMOTE SPO activities.

<char> denotes the desired character. SO and RO are not valid for this option.

RECONFIGURATION — This option allows the user to modify the network configuration, using the appropriate MCS commands.

The Save Factor of SYSCONFIG contains the compile date of the version of CONFIGURER which created that SYSCONFIG.

Defaults

The values of the SYSCONFIG file given with the 3.03 release are as follows:

SPO LOGGING	ON
NUMBER OF LOG FILES	03
SIZE OF LOG FILES	00032
LOG PARAMETERS	00000
MESSAGE FILE NAME	SYSLANGUAGE
DATE	ON
TIME	ON
RELEASE LEVEL	3.03.05
DCCH TRACE	OFF
REMOTE SPO	ON
RECONFIGURATION	ON
REMOTE CHARACTER	?
CLEAR START COUNT	0000
DC BUFFER SIZE	0064

All other DCSS information is reset.
ZIP information is reset.
PO MESSAGE information is reset.
The backup option, PRBK, is reset.

Update Messages

DCCH TRACE FLAG SET [RESET]
REMOTE FLAG SET [RESET] (REMOTE CHARACTER: = 'x')
RECONFIGURATION FLAG SET [RESET]
REMOTE CHARACTER 'x' CHANGES TO 'y'
DC BUFFER SIZE CHANGED FROM xx K-BYTES TO yy
NUMBER OF LOG FILES CHANGED FROM xx TO yy
LOG FILE SIZE CHANGED FROM xx SECTORS TO yy
MESSAGE FILE NAME 'xxx' CHANGED TO 'yyy'
SPO LOGGING FLAG SET [RESET] (NUMBER OF LOG FILES: xx, LOG FILE SIZE: yy)
DUMP DCPx SET [RESET]
TRACE BDSx SET [RESET]
BACKUP OPTION SET [RESET] (BACKUP DESIGNATE : 'xxx', BACKUP SIZE yyy RECORDS)
BACKUP DESIGNATE DISK 'xxx' CHANGED TO yyy
BACKUP FILE SIZE CHANGED FROM xxx RECORDS TO yyy
ZIP MESSAGE RESET
ZIP MESSAGE CHANGED TO '<zip text>'

Error Messages

<mix /prog> CANNOT OPEN <mfid /fid> FILE TYPE MISMATCH
<mix /prog> CANNOT OPEN <mfid /fid> PACK NOT PRESENT
<mix /prog> CANNOT OPEN <mfid /fid> DIRECTORY FULL
<mix /prog> CANNOT OPEN <mfid /fid> INVALID FILE NAME
<mix /prog> WAITING OPEN ON <mfid /fid> IN USE
<mix /prog> CANNOT OPEN <mfid /fid> DISK LOCKED
<mix /prog> CANNOT OPEN <mfid /fid> BAD BLOCK OR RECORD SIZE
<mix /prog> CANNOT OPEN <mfid /fid> BAD FILE SIZE
<mix /prog> CANNOT OPEN <mfid /fid> FETCH VALUE: @XXXXXX@
<mix /prog> WAITING CLOSE ON <mfid /fid> DUPLICATE FILE
<mix /prog> CANNOT CLOSE <mfid /fid> FILE TYPE MISMATCH
<mix /prog> CANNOT CLOSE <mfid /fid> FILE IN USE
<mix /prog> CANNOT CLOSE <mfid /fid> DISK LOCKED
<mix /prog> CANNOT CLOSE <mfid /fid> FETCH VALUE @XXXXXX@
<mix /prog> DATA ERROR ON READ FROM <mfid /fid> -RECORD XX
<mix /prog> INVALID KEY ON READ FROM <mfid /fid> -RECORD XX
<mix /prog> CANNOT READ <mfid /fid> -RECORD XX FETCH VALUE @XXXXXX@
<mix /prog> PARITY ERROR ON WRITE TO <mfid /fid> -RECORD XX
<mix /prog> NO USER DISK TO CREATE <mfid /fid>
<mix /prog> CANNOT CREATE <mfid /fid> -DISK LOCKED
<mix /prog> INVALID KEY ON WRITE TO <mfid /fid> -RECORD XX
<mix /prog> CANNOT WRITE <mfid /fid> -RECORD XX FETCH VALUE @XXXXXX@
<mix /prog> INVALID NUMBER OF LOG FILES
<mix /prog> INVALID LOG FILE SIZE
<mix /prog> INVALID DCP NUMBER
<mix /prog> INVALID DC BUFFER SIZE
<mix /prog> INVALID BACKUP FILE SIZE
<mix /prog> INVALID CONFIGURER REQUEST (<option>)

These messages are self-explanatory.

When a SYSCONFIG file already exists on a pack and an attempt is make to CREATE a new SYSCONFIG file on the same pack, the previous one will be removed unless its file type value is different from @22@. In this case the following messages are issued:

<mix /prog> * * WARNING * * <fid> ALREADY PRESENT ON <mfid> FILE TYPE IS @XX@

<mix /prog> ACPT (ANSWER 'OK' OR ELSE)

By answering 'OK', the user gives CONFIGURER the authorization to remove the previous SYSCONFIG file.

By entering any other reply, this prevents the removal of the previous SYSCONFIG file.

Warning Messages

<mfid /fid> REMOVED
<mfid /fid> CREATED
<mfid /fid> UPDATED
<mfid /fid> LISTED
<mfid /fid> OK

DCP.ANALYZER

A data comm problem can arise from a problem in the DCP itself. A DCP dump must then be taken and properly analyzed by DCP.ANALYZER.

How to Take a DCP Dump

There are two ways:

- by setting the appropriate bit in SYSCONFIG (see CONFIGURER notes). The dump(s) will be taken at the next data comm initialization time. This will cause DCCH to go to End of Job, without executing the requested MCS. The bits corresponding to the relative position of the DCP dumped will be reset in SYSCONFIG, and it is the responsibility of the operator to restart the data comm subsystem.

- by using the command 'DC DUMPDCPx', a dump of the DCP whose relative position is given by x will be taken. This does not interfere with the normal execution of the data comm operations on the other DCPs and allows normal use of the concerned DCP after the dump has been taken (although the problem for which the dump was taken still remains).

How to Analyze a DCP Dump

DCP.ANALYZER analyzes dumps created from either DCP-1 or DCP-3. Two modes of operation are provided: interactive mode and batch mode.

Interactive mode is invoked by not specifying an initiating message, and can be used from either the SPO or a terminal. DCP.ANALYZER can be used with either a GEMCOS or TMCS interface.

The user can examine (display) and/or print selected portions of the dump information using the commands provided.

Syntax

DCP.ANALYZER (from the SPO)
<MCS control char> RN DCP.ANALYZER (from a terminal)

Two kinds of reserved words are used by DCP.ANALYZER:

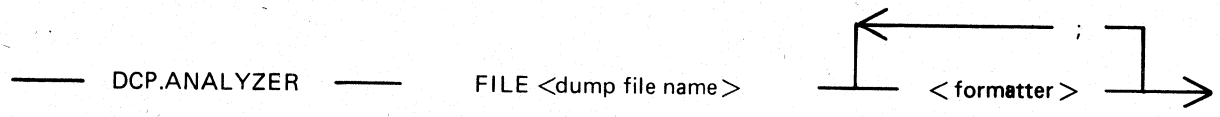
- the Commands used to specify an action to be executed, usually a file related function.
- the Formatters to generate output formats.

Commands and formatters are entered by using AX messages via the SPO or directly from a terminal.

If the output takes more than a page, then '...' appears at the bottom left hand corner of the screen, and the user must send a blank input to get the next page. For a terminal the cursor is positioned to send one space character.

A HELP function will be invoked if an entry, not recognized as a reserved word, is input.

In batch mode, entered by specifying an initiating message, all output is printed. The user can choose which portions of the dump information are to be printed, or the entire dump can be printed. The initiating message syntax is:



where <dump file name> is the name of the DCP dump file to be analyzed, and <formatter> is one of the formatters defined below.

EXAMPLE: To print the entire dump of DCP 0:

```
DCP.ANALYZER FILE DUMPDCP0 DUMP
```

Commands

BYE
CLOSE PRINTER
DCP-1
DCP-3
DISPLAY
END
FILE
FORMS
GET
HELP
PRINT
WHAT

Formatters

DUMP
LIEGE.HDR
LINE
LINE HISTORY
LINE.INFO
LINE.TABLE
LIST.AVAIL
LIST.MSG
LIVM.HDR
NETWORK
RESV.MEM
STATION
STATION.HISTORY
STATION.TABLE
SUMMARY

Each command and formatter is defined below.

Where a numeric parameter or address is required, the value can be specified in two ways:

decimal; for example, 255
hexadecimal, by specifying the hexadecimal value IMMEDIATELY after the special character '@'; for example, @FF

BYE

This command is used to exit the interactive mode.

CLOSE PRINTER

When a print is requested, the analyzer opens a printer file and keeps it open until the end of job. The CLOSE PRINTER command allows the user to close the printer file without terminating the analyzer job.

DCP-1

This command is used when the program cannot determine the type of dump file to be analyzed. The DCP-1 type will be forced.

DCP-3

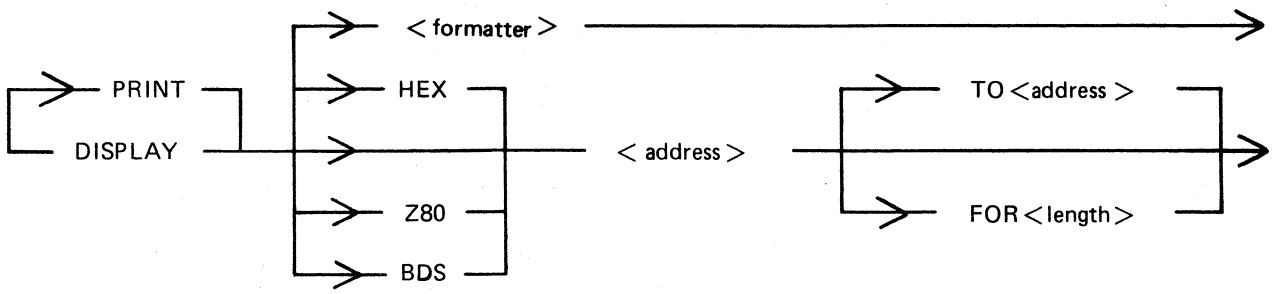
This command is used when the program cannot determine the type of dump file to be analyzed. The DCP-3 type will be forced.

DISPLAY

This command can be used to display the dump file contents in hexadecimal or disassembled micro code.

DISPLAY <formatter> used in interactive mode will display the selected format but is redundant, since entering <formatter> only will give the same results.

The full syntax, common for DISPLAY and PRINT, is:



The type code HEX, BDS or Z80 specifies the format desired:

HEX for hexadecimal memory contents
BDS for disassembled DCP-1 code
Z80 for disassembled DCP-3 code

The default is the last type used or HEX if no type has previously been specified.

The dump range may be specified from a start address to an end address or from a start address for a specified number of bytes. If only a start address is specified, a length of 255 bytes is assumed.

DUMP

This formatter will output all information for the file loaded. It successively executes the following formatters:

SUMMARY
NETWORK
RESV.MEM
LIST.AVAIL
LINE for each line declared in NDL for the related DCP

END

This command is used to exit the interactive mode.

FILE <file-name>

This command is used to open the specified file and select it as a DCP-1 or DCP-3 dump file with the message 'DCP-n DUMP FILE LOADED'. If the dump type cannot be determined, the message 'CANNOT DETERMINE DCP DUMP TYPE' is displayed and the dump type can be selected by the user via the use of the DCP-1 or DCP-3 commands.

If the file cannot be opened, the message 'CANNOT OPEN <fid> [<n>]' is output where <n> is the CMS event number indicating the reason for the failure.

FORMS

This command will display all the formatters available, adding DCP-1 and DCP-3 on the list.

GET <file-name>

This command is similar to the FILE command.

HELP

This command will interactively display explanations on the analyzer program, all the commands and all the formatters available.

LIEGE.HDR <addr>

The Liege header found at the specified address is output. The Liege header is B 1000 system implementation dependent.

LINE <lln>

LINE <memory address>

This formatter takes a parameter which is either a logical line number or the memory address of the line information table.

The line information table, the line table and each attached station table is provided.

LINE.HISTORY <lln>

All message headers found in the available list referring to the logical line number specified are output. Since the buffers are allocated cyclically, these messages provide a history of activity on the line.

Following the messages, the line information table, line table and station tables for each attached station are output.

LINE.INFO <lln>

LINE.INFO <memory address>

The parameter may specify either a logical line number or the memory address of the required line information table. If memory address is specified, the user must ensure that it addresses a valid line info table.

The line information table format printed is B 1000 systems implementation dependent.

LINE.TABLE <lln>

LINE.TABLE <memory address>

The parameter may specify either the logical line number or the memory address of the required line table.

If a memory address is specified, the user must ensure that a valid line table address is specified.

The format of the line table printed is as specified for the NDL S-machine, and is documented in the Data Comm Subsystem Reference manual, Form number 1090909.

LIST.AVAIL

This formatter outputs the list of all available buffers in the DCP memory. When present, the message header of each buffer will be output.

Since the available buffer pool is allocated cyclically, the list of available buffers is a dynamic history of messages in the approximate order that they were processed.

LIST.MSG <addr>

The message header (LIEGE.HDR and LIVM.HDR) and the beginning of the message text found at the specified address is output. A few '...' are present at the end of the line if the message is longer than 61 characters.

LIVM.HDR <addr>

The LIVM header found at the specified address is output. The LIVM header is defined for the NDL S-machine, and is documented in the CMS Data Comm Subsystem Reference manual, Form number 1090909.

NETWORK

This formatter provides information on the line configuration defined in the NDL for the related DCP.

PRINT

This command is similar to the DISPLAY command but the output generated is directed to a printer.

Refer to DISPLAY for the complete syntax.

PRINT <formatter> option used in batch mode will print the selected format but is redundant, since entering <formatter> only will give the same results.

RESV.MEM

This formatter will output information regarding the state of the DCP and HOST interface at the time of the dump.

The reserved memory format printed is B 1000 system implementation dependent.

STATION <lsn>

STATION <memory address>

The parameter may specify either a logical station number or the memory address of a station table.

If a memory address is specified, the user must ensure that it is a valid station table address.

The station table is output together with any messages queued in the station queue.

STATION.HISTORY <lsn>

All message headers in the available buffer pool referring to the logical station specified are output. Since buffers are allocated cyclically, the list of messages provides a history of station activity.

Following the messages, the station table and station queue are output.

STATION.TABLE <lsn>

STATION.TABLE <memory address>

The parameter may specify either a logical station number or the memory address of the required station table.

If a memory address is specified, the user must ensure that it is a valid station table address.

The selected station table is output.

SUMMARY

This formatter outputs information regarding the generation of the DCP code file and also about the physical configuration of the lines on the DCP.

WHAT

This command will give information about the version level of the DCP.ANALYZER used and about the dump file analyzed.

DISKDUMP

The aim of the DISKDUMP utility is to make the B 1000 system more secure and to allow the user to get through files corrupted by software or hardware. For that purpose, it enables the user to have a quick on-line backup of a CMS disk.

This backup is created on another CMS disk, but can transit via a tape.

Any program will execute normally, provided it resides on a disk other than the disk used as input to DISKDUMP.

For the programs residing on the input disk:

- if they are started before DISKDUMP, they can go on executing as long as they do not access SYSMEM, as SYSMEM is locked during the DISKDUMP run (so no access is allowed to the available table or disk file headers for modification of area allocation).

- an attempt to start them during the DISKDUMP execution will fail and a 'DISK ERROR' message will be displayed. The actual meaning of this message is: 'SYSTEM LOCK'.

Four functions are provided:

1. DSKDSK for a copy from disk to disk.
2. DSKMTP for a copy from disk to tape.
3. MTPDSK for a copy from tape to disk.
4. VERIFY.

DSKDSK

The data of every sector of the input disk is moved into the corresponding sector of the output disk.

After the process, both disks are identical, except that they keep their own label.

Therefore, if there is a bad sector on the output disk, the copy of the corresponding sector from the input disk will not be possible, and the utility will go to DS/DP. Ensure that the output disk is not contaminated with bad sectors before initiating a DSKDSK.

It is clear that a copy of a disk to itself (after a transit on a tape) will not be affected by the presence of bad sectors on the disk.

Each sector in error on the input disk is skipped while the corresponding sector on the output disk is filled with binary '1's.

DSKMTP

Every sector of the input disk corresponds to a record on the output tape. A bad sector on the disk produces a tape record filled with binary '1's.

MTPDSK

Every record of the input tape is moved into a sector on the output disk. The first record on tape must be a copy of the CMS disk label.

VERIFY

Every record of the output device is compared with the corresponding record on the input device.

Any discrepancy found between two records causes the utility to display the following message:

COMPARISON ERROR REC XXXXXX

XXXXXX being the (hexadecimal) number of the faulty record.

Approximate duration of the process:

DSKDSK of a 206 disk pack to a 206 disk pack: 11 minutes.
A VERIFY of the operation: 11 minutes.

DSKMTP of a 206 disk pack to two 2400 feet tapes: 8 minutes.
A VERIFY of the operation: 10 minutes.

MTPDSK of two 2400 feet tapes to a 206 disk pack: 9 minutes.
A VERIFY of the operation: 11 minutes.

Program initiation:

DISKDUMP

The utility then displays a series of prompts to which the user must enter replies via:

AX <mix number>

Prompt 1: ENTER OPTION: DSKDSK, DSKMTP, MTPDSK
Prompt 2: ENTER INPUT DEVICE ID
Prompt 3: ENTER OUTPUT DEVICE ID
Prompt 4: VERIFY? ENTER YES, NO, ETX

NOTE

DEVICE ID, in prompt 2 and 3 should be read: PACK ID in the case of a disk pack, or TAPE LABEL in the case of a tape.

Error Handling

Any detected hardware error causes a message to be displayed, indicating the device in error, the Fetch Communicate Message (FCM) returned by the MCP, and, if the process of copying was already executing, the number of the record in error (hexadecimal number).

DISKDUMP goes to End of Job when a hardware failure occurs on the tape or on the output disk.

Limitations

When beginning the write operation on the output disk, DSKDSK and MTPDSK zip the series of messages: 'RY DPA', ..., 'RY DPL', 'RY DKA' ,..., 'RY DKF'.

This sets a limitation of 12 to the number of output disk packs, and of 6 to the number of output cartridges.

WARNING

Copying a disk where files are in use produces comparison errors at VERIFY time.

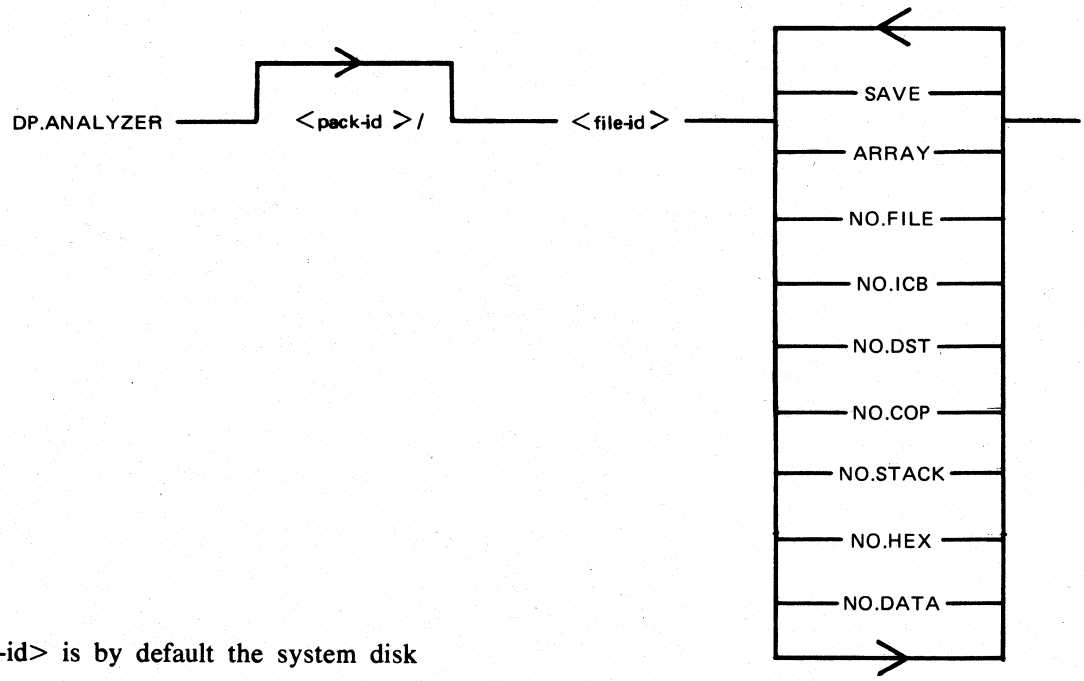
DP.ANALYZER

Execution of a program can lead to a DS/DP condition. When a program is DP'ed, a dump file is created on the same disk as the program file (either by the operator or by the COBOL interpreter when a 'NO USE PROCEDURE' condition is encountered after an error). The name of the dump file is DMFILnn, where nn is the mix number.

The dump file can be analyzed, in relation to the program file itself, to give a listing containing:

- program parameters
- run parameters
- interface control block
- communicate parameter area
- program segment table
- data segment table
- control stack analysis
- file information (FPB, FIB, buffers)
- data segments
- current code segment
- locked slice

The listing can be obtained by invoking DP.ANALYZER with the following syntax:



<pack-id> is by default the system disk

The options, entered in any order, have the following effect:

SAVE	This will cause DP.ANALYZER to retain the input dump file rather than remove it after analysis (which is the default).
ARRAY	This is applicable only to COBOL/RPG dump files. It will cause all elements of all arrays to be printed. The default is to print a maximum of twenty elements of each array.
NO.FILE	This will suppress printing of any file information.

NO.ICB	This will suppress printing of any ICB information.
NO.DST	This will suppress the DST and PST analysis.
NO.COP	This will suppress the COP table analysis (only for COBOL programs).
NO.STACK	This will suppress the Data Stack analysis (only for MPL II programs).
NO.HEX	This will suppress printing of the current Code segment and of the Locked Slice.
NO.DATA	This will suppress the printing of the Data segments.

Examples

1. DP.ANALYZER DMFIL03 SAVE

   DP.ANALYZER will expect to find a dump file called DMFIL03 on the system disk. It will print a formatted dump listing and will retain the dump file.

2. DP.ANALYZER USER/DMFIL02 NO.FILE

   DP.ANALYZER will expect to find a dump file called DMFIL02 on a disk labelled USER. It will print a formatted listing omitting all file information and remove the file USER/DMFIL02 at the end of the job.

3. DP.ANALYZER DMFIL04 ARRAY SAVE

   DP.ANALYZER will expect to find a dump file called DMFIL04 on the system disk. It will print a formatted listing including all the elements of all arrays and will retain DMFIL04 at the end of the job.

NOTE

Sometimes, due to the BIL interpreter, the contents of the end of an MPLII program working stack are unpredictable. In this case, the printing will be limited to the portion which can be analyzed.

LT

Load Line Printer Train

This utility allows the correct chain of characters to be loaded on a 450/750 LPM printer. The first time, after CLEARSTART, a 450/750 LPM printer is switched ON-LINE, the message:

'LT REQUIRED FOR LP<x>'

is displayed, where <x> is the printer mnemonic. The operator must then enter:

'LT LP<x> <chain type>'

where <x> is the printer mnemonic, and <chain type> must be one of the following train names or train numbers:

For 1100/1500 train printers:

Train Name	Train Number	Description
EBCDIC18	001	18-character EBCDIC
FORTRAN48.NON STD	002	48-character FORTRAN.NON STD
B300.B50048	003	48-character B300/B500
EBCDIC48	004	48-character EBCDIC
EBCDIC72	005	72-character EBCDIC
UKB3500.72	006	72-character EBCDIC
UKB6500.72	007	72-character EBCDIC
PORTUGAL.72	008	72-character EBCDIC
SPAIN.72	009	72-character EBCDIC
FINLAND.72	010	72-character EBCDIC
DENMARK.72	011	72-character EBCDIC
BCL72	012	72-character BCL
TURKEY.72	013	72-character EBCDIC
SWEDEN.72	014	72-character EBCDIC
ASCII72	015	72-character ASCII
EBCDIC96	016	96-character EBCDIC
EBCDIC-UPPER.CASE	016	96-character EBCDIC
EBCDIC.UPPER.CASEB	016	96-character EBCDIC
EBCDIC-LOWER.CASE	016	96-character EBCDIC
EBCDIC.LOWER.CASEB	016	96-character EBCDIC
KATAKANA	017	96-character KATAKANA
EBCDIC.A72	018	72-character alphabetized EBCDIC
EBCDIC.N72	019	72-character numericized EBCDIC
RPG48	020	48-character RPG
OCR.A72	021	72-character OCR-A
OCR.B72	022	72-character OCR-B
FORTRAN48	036	48-character FORTRAN
THAI	052	144-character THAI

For 400/750 LPM train printers (PRINTER CONTROL 5 or 6):

For 400/750 LPM Train printers:

Train Name	Train Number	Description
FORTRAN48	130	48-character FORTRAN
FORTRAN48.NON STD	130	48-character FORTRAN.NON STD
B300/B500.48	131	48-character B300/B500
EBCDIC3.48	132	48-character EBCDIC-3
RPG48	140	48-character RPG
EBCDIC96	144	96-character EBCDIC
KATAKANA	145	96-character KATAKANA
EBCDIC3.16	254	16-character EBCDIC-3
EBCDIC3.64	255	64-character EBCDIC-3

The normal END-OF-JOB message is:

'<train name> = <train number> LOADED ON LP<x>'

Two error messages can be displayed:

'INVALID LT REQUEST'

This indicates that the chain type is not valid, or that the specified chain does not match the printer type.

'LP<x> NOT AVAILABLE FOR LT'

This indicates that the line printer <x> is not ON-LINE.

'LT T' command will cause the SPO to display:

the syntax diagram of the LT utility
a list of the train names with their train number

MEM.ANALYZER

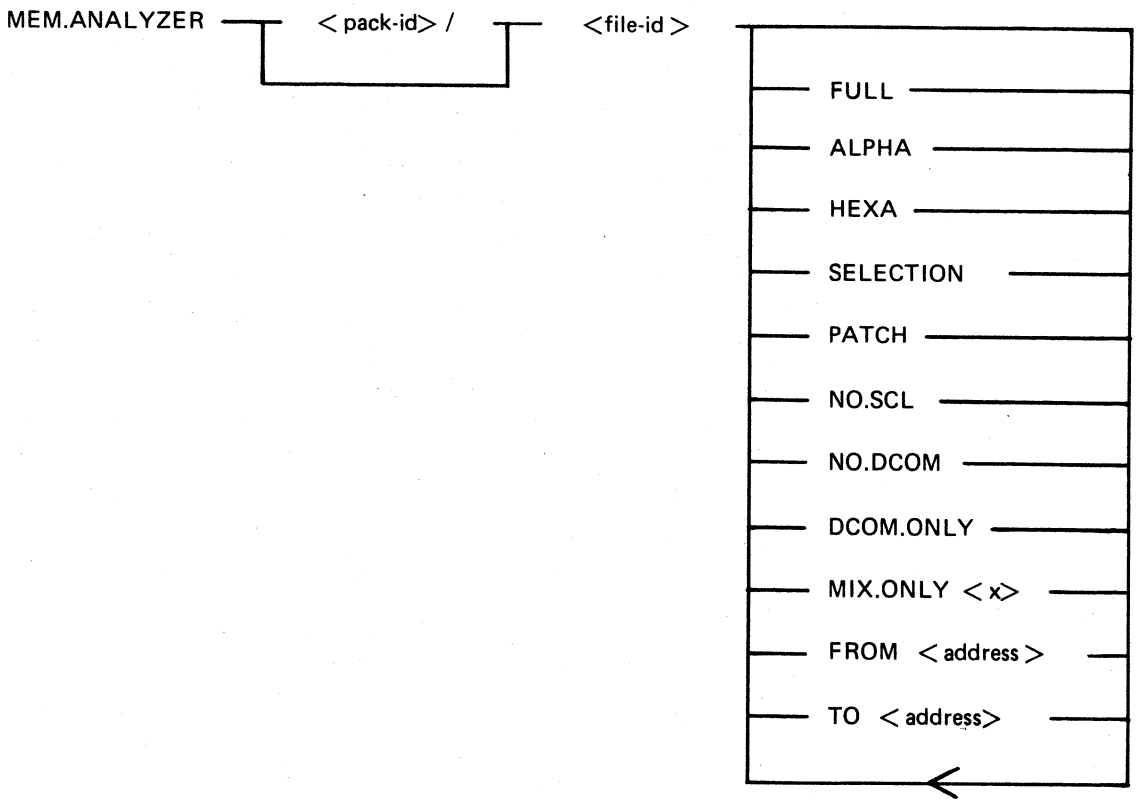
A memory dump can be obtained by executing the appropriate MEM.DUMP cassette in STAND-ALONE mode (for the operating instructions, refer to MEMORY.DUMP in Section 2, STAND-ALONE UTILITIES). This causes the contents of the memory to be written into a file named DMFIL00.

The system must then be restarted, with a CLEARSTART, and the MEM.ANALYZER program must be executed to obtain a formatted, analyzed listing of the memory dump file.

The file DMFIL00 is created on the system disk.

If enough space is not available on that disk, or if memory is corrupted so that the location of that disk is destroyed, the program will halt to give the operator the ability to direct the dump to another disk.

The listing can be obtained by invoking MEM.ANALYZER with the following syntax:



<pack-id>	indicates the name of the device containing the dump file to be analyzed.
<file-id>	indicates the name of the dump file to be analyzed.

The options, entered in any order, have the following effects:

FULL	Print tables and all memory in hexadecimal.
ALPHA	Print memory contents in alpha-numeric.
HEXA	Print memory contents in hexadecimal.
PATCH	Printing of the patch analysis is enabled.
NO.SCL	Disable SCL partition decoding.
NO.DCOM	Disable Data Comm module decoding.
DCOM.ONLY	Disable User-memory analysis.
MIX.ONLY <x>	In analysis of User-memory, only the mix specified by <x> will be considered.
FROM <address>	When ALPHA or HEXA is used (default is 0).
TO <address>	When ALPHA or HEXA is used (default is the maximum memory value).

NOTE

1. The FROM and TO options do not require the specification of ALPHA or HEXA. The default is HEXA.

2. The FROM...TO.. option cannot be repeated to print different portions of memory in one run.

SELECTION

This option causes a series of ACCEPT/DISPLAYs to be executed at the SPO. These allow the operator to select sections of the dump listing to be printed on a printer. The syntax for these sections follows:

MAT indicates the Memory Assignment Table portion of the listing.

VM indicates the Virtual Memory Tables and VM statistics portion of the listing.

PHYSIO indicates the PHYSICAL.IO information portion; that is, Channel Control Words, Device Control Blocks, Channel Control Blocks.

SCL indicates SCL information.

SEGMENT <integer-1> <integer-2>
indicates a data segment to be printed.
<integer-1> is the data segment number in the Data Segment Table.
<integer-2> is the Mix number of the associated program.

FIB <integer-1> <integer-2>
indicates the FIB to be printed.
<integer-1> is the data segment number of the segment containing the FIB in the Data Segment Table.
<integer-2> is the Mix number of the associated program.

MIX <integer>
the entire User Memory Analysis section (that is, ICB, DST, PST, File information) for the Mix indicated by <integer>.

ICB <integer>
indicates the ICB of the Mix indicated by <integer>.

DST <integer>
indicates the Data Segment Table of the Mix indicated by <integer>.

PARTITION <integer>

indicates a hexadecimal dump of the partition indicated by <integer>, where <integer> indicates the Mix number.

END
terminates the selection process.

NOTE

The printer file used in this selection process cannot be diverted to backup. For a selective listing which can be diverted to backup, use the other MEM.ANALYZER options.

NOTE

Each dump submitted with an FTR must use the default option.

This program analyzes the input file specified, and a complete listing will provide:

A form to be filled in by the user.
A form to be filled in by the Burroughs representative.
The contents of the main registers (register A must be noted manually for a B 1830).
The last 12 entries in TAS
Scratchpads
Mix table
Monitor stack
Memory assignment table
Virtual memory tables
Virtual memory statistics
Channel and Channel Control Word tables
Channel control blocks
Device control blocks
File control blocks
SCL partition (mix #0)
Overlay module analysis (mix #62
DATA COMM module analysis, if DCHH was running
Shared files Data segment
A few other system tables
The ICB, PST, DST, FIBs and FPBs for each user mix
A history of the patches inserted since release time

The copy of the interpreters in memory is compared with their initial version on the system disk and the address of the first discrepancy, if applicable, will be displayed.

The Memory Dump file is always retained after MEM.ANALYZER has gone to End of Job. It can be removed with the RM command by the operator, but if its name was left as DMFIL00 it will be replaced when a new Memory Dump file is created.

Therefore, it is suggested that after creation of DMFIL00, the name of this file is changed to a unique name, and it is analyzed as soon as possible by MEM.ANALYZER. It should also be removed as soon as the file is no longer needed.

PATCH.MAKER

This utility is released in order to implement required patches to released firmware. This utility either reads a file of patches from cassette, or generates a file from SPO input.

It is recommended that two system disks are used when the MCP file is to be patched.

This is not required when patching the MPL interpreter (BILINTERP), the RPG and COBOL interpreter (COBOLINT) and the DCP.3 NDL Post Compiler (NPC3.B1000). However, it is possible to patch the MCP file when only one system disk is available (refer to Operating Instructions, later in this section).

It must be remembered that ALL disks containing system software must be updated with the patched software.

Levels of patches are successive. If a user requires patch level 3, for example, then both previously released patches must have been implemented. The utility employs a number of checksum routines which prevent patching of the wrong file and implementation of unauthorized patches.

The utility first asks the user, with displays and accepts, for the identity of the firmware file to be patched in this run (MCP, BILINTERP or COBOLINT) and of the new file to be created.

The user is then asked to specify if the patches are to be entered via the SPO or via a cassette. If the response specifies cassette, a search for a tape file named 'PATCHES' is initiated. When this file is found, the patching is carried out. If the response specifies SPO, the patches are to be entered via the keyboard from the hard-copy supplied. The characters entered must be exactly as supplied. The series of characters can be divided into several series of any length up to 254 characters. The operation is terminated by a null AX.

In order to minimize the risks involved in transmitting patches by TELEX, and possible errors in operator action, an initial sumcheck has been inserted in the patch string itself. Each 12 bytes of patch information are sumchecked. If a mismatch is detected, the wrong 12-byte group will be displayed and these 12 bytes must be re-entered again via the SPO.

A null 'AX' terminates input and allows the patching to begin.

If any message is displayed, one of the checks has failed. The input must be examined and re-entered when the error is found.

Error Messages Displayed

<string of 24 bits> AT OFFSET <nn> IS INVALID: RESUBMIT ON SPO

Initial sumcheck has detected an error within the string of digits entered via the SPO or cassette.

Patch Level Discrepancy

The specified file does not have the patch level expected. All previous patches must be entered.

Initial Sumcheck Discrepancy

The sumcheck of the original file to be patched does not match the value contained in the patch entry. All previous patches must be entered.

Final Sumcheck Discrepancy

The sumcheck of the patched file does not match the value contained in the patch entry. Try again on another disk drive. If the error persists, the patch may be in error. Contact your local Burroughs representative.

Address Error

The address given for a micro is out of range for the file specified. Try again on another disk drive. If the error persists, the patch may be in error. Contact your local Burroughs representative.

Old/New Micro Discrepancy

The old micro contained in the patch entry does not match the old micro in the file to be patched. All previous patches must have been entered.

Example

Assume that a pack with all the released software is available. It is called BKP3.03.

The system is CLEARSTARTED with BKP3.03 as the system disk. The patch consists of a hard-copy (flash with '333355554444', for example).

When CLEARSTART has finished:

```
PATCH.MAKER
02/PATCH.MAKER BOJ PR = A TIME: hh.mm.ss
02/PATCH.MAKER [DISP] ENTER NAME OF FILE TO BE PATCHED
02/PATCH.MAKER [DISP] [PACK-ID>/] <FILE-ID]
02/PATCH.MAKER [ACPT]
AX 02 BKP3.03/MCP
02/PATCH.MAKER [DISP] ENTER NAME OF NEW FILE
02/PATCH.MAKER [DISP] [<PACK-ID>/] <FILE-ID>
02/PATCH.MAKER [ACPT]
AX 02 BKP3.03/MCPNEW
02/PATCH.MAKER [DISP] ENTER INPUT DEVICE: SPO OR CASSETTE
02/PATCH.MAKER [ACPT]
AX 02 SPO
02/PATCH.MAKER [ACPT]
AX 02 33335555
02/PATCH.MAKER [ACPT]
AX 02 4444
02/PATCH.MAKER [ACPT]
AX 02
02/PATCH.MAKER EOJ
```

When this has finished, execute 'COPY MCPNEW TO MCP', then CLEARSTART the system with BKP3.03 as the system disk.

Operating Instructions for Patching the MCP File

The procedure varies according to the number of system packs the user has at his disposal:

1. Two System packs are available (SYS1 and SYS2)

```
Clearstart with SYS1 as System disk
                SYS2 as User disk
Run PATCH.MAKER with the file MCP on the System disk as input file
                     with a file MCP on the user disk as output file
```

Therefore, the name of the file to be patched is: SYS1/MCP, and the name of the new file is SYS2/MCP.

```
Clearstart with SYS2 as System disk
                SYS1 as User disk
Copy MCP to SYS1/MCP
```

2. One System pack only is available

```
Copy the MCP file to MCPxxx
Run PATCH.MAKER with as input file: MCPxxx
                     as output file: MCPyyy
(xxx and yyy being three-digit integers)
```

There are two ways available of using the new MCP file MCPyyy:

either copying it to MCP, and Clear Starting (1)
or Clear Starting directly with MCPyyy (2)

1) Copy MCPyyy to MCP. Ensure that two conditions are satisfied while copying MCPyyy to MCP:

a. A NULL MIX (even SYS-SUPERUTL must be DS'ed)
b. NO DEVICE STATUS CHANGE (No device hardware interrupt)

On completion, enter 'PO' for the system disk.

Clear Start

2) Clearstart with 000yyy loaded in the X register between TAPE mode and RUN mo (see CLEARSTART in this section).

It is always possible to run again with the old MCP file by Clear Starting with the file named MCPxxx.

A COLDSTART selecting only the file MCP on the release tape is another means of loading a valid MCP file.

Example: to apply patch 45 to the MCP file:

Copy MCP to MCP045
Run PATCH.MAKER with

name of the file to be patched: MCP045
name of the new file: MCP046

Then, either

Copy MCP046 to MCP, after having DS'ed all programs in the MIX.
Power off the system disk and CLEARSTART.

or make a CLEARSTART with 000046 loaded in the X register between TAPE mode and RUN mode.

Operating Instructions for Patching the Interpreters or NPC3.B1000

Ensure that the file to be patched is not in use.

- no COBOL or RPG program running for COBOLINT.
- no MPL program running for BILINTERP (all utilities are written in MPL).

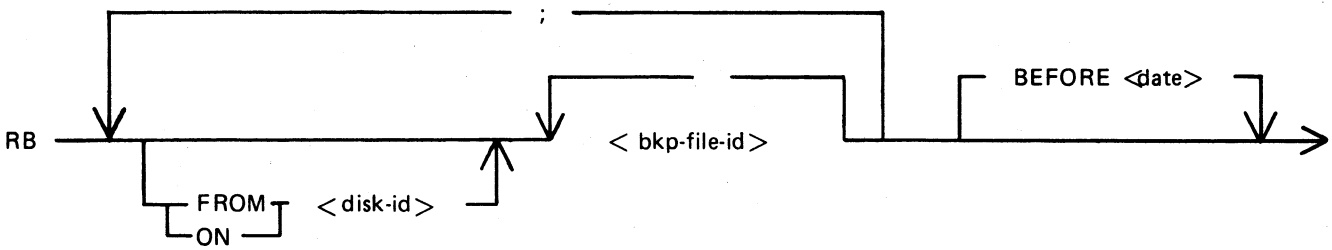
Example: to patch the BILINTERP:

Copy BILINTERP to BILINTERP1
Run PATCH.MAKER with as input file BILINTERP1 and
as output file BILINTERP2
Copy BILINTERP2 to BILINTERP

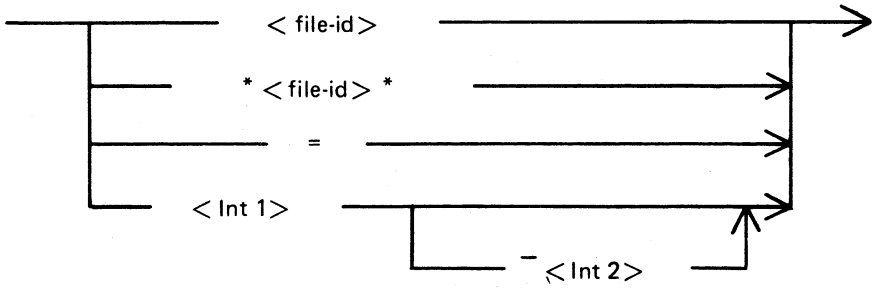
RB

This utility enables the user to remove printer backup files from one or more disks. Backup files can be removed simply, by family, or by age.

RB HELP displays the syntax diagram for RB, as follows:



where <bkp-file-id> is defined as



and <date> ::= <MM /DD /YY>

where MM and DD (month and day respectively) can be single digits.

<disk-id>	specifies the disk on which the utility seeks backup files; default is the backup designated disk.
<file-id>	designates the backup file or the family of backup files to be removed. Either the external or the internal file name can be specified.
* <file id>	* is used when <file id> alone could cause confusion (for example RB * 5 * removes the backup file named 5 instead of the one named PB00005.
=	specifies that all the backup files PBxxxxx are to be removed, where 1 =<xxxxx =< 65535
<Int 1>	specifies that the backup file PB<Int 1> is to be removed.
<Int 1> -<Int 2>	specification causes all backup files from PB<Int 1> to PB<Int 2> to be removed.
BEFORE <date>	option causes all backup files created before the specified date to be removed.

Examples

To remove the file PB00038 from the system disk:

RB 38

To remove the file PB00017 and all backup files in the family PR=, from the disk USDSK:

RB FROM USDSK 17, PR=

To remove PB00032,PB00034 from the system disk, all the backup files named PBxxxxx from the disk TASK, and all backup files in the range PB00011 to PB00019 from the disk ARDSK, with each one of these files being subject to the date given:

RB 32,34; FROM TASK =; FROM ARDSK 11-19 BEFORE 1/6/81

Messages

If syntax errors occur, the utility displays messages that are self-explanatory, and goes to End of Job.

If a disk cannot be opened, RB continues executing, going on to the next disk specified, after displaying:

UNABLE TO ACCESS DISK <pack-id>

If a file cannot be found, or no files are found in a family, RB goes on executing after displaying the appropriate messages.

The user is also informed when a file is removed, or when it is not removed, due either to the fact that it is still in use, or was created on or after the 'BEFORE' date.

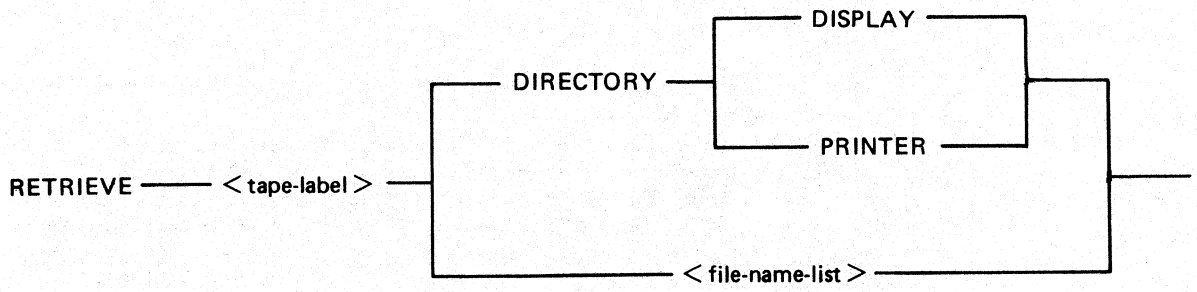
After completion, RB issues the message:

* * RB COMPLETE * *

RETRIEVE

This utility provides the user with a means of copying to the system disk one or more single files from a tape created by the DISKDUMP program.

Syntax: The following syntax diagram will be displayed by entering 'RETRIEVE', followed by a blank or by any single word.



DIRECTORY option gives the user the list of the files which are present on the tape, directed to the device specified by one of the following options:

DISPLAY causes a single list to be displayed on the screen.

PRINTER provides the user with detailed information about each file residing on the tape: that is,

the pack-id (of the originating disk)
the file-name
the actual and the maximum file size
the creation date and the last access date
the file type

<file-name-list>: up to 20 files can be loaded on the system disk in a single run of the RETRIEVE utility. The file names will be separated by a blank, or by a comma surrounded by blanks.

Messages

Most of the messages that are displayed are self-explanatory. However, the following messages need some explanation.

If a file to be loaded is already on the system disk, the messages

FILE <file-name> IS ALREADY PRESENT ON DISK

PLEASE REMOVE IT AND REPLY TO THE ACCEPT

will be displayed on the screen.

The utility will resume as soon as an answer has been given to the Accept, whatever this answer might be (a blank or any word).

STAND-ALONE UTILITIES

The stand-alone utilities perform specific functions which are not under MCP control. The following are the stand-alone utilities available with this release:

NAME	FUNCTION
COLDSTART	Loads system software to disk from a COLDSTART magnetic tape.
CLEARSTART	Determines the system configuration, builds tables and places the system under MCP control.
PACK.INIT	Initializes a disk pack in CMS format.
CART.INIT	Initializes a disk cartridge in CMS format.
MEM.DUMP	Creates a disk file (DMFIL00) containing a complete image of the system memory.

All these stand-alone utilities are initiated with cassettes.

CREATION OF CASSETTES

A utility, CASS.CREATE, has been developed which enables B 1860 series stand-alone cassettes to be generated under CMS MCP control.

Syntax

CASS.CREATE <sau-file-name> TO XXX MTP

where <sau-file-name> is one of:

1860.CLEAR for CLEARSTART
1860.COLD for COLDSTART
1860.PACK for PACK INITIALIZER
1860.CART for CARTRIDGE INITIALIZER
1860.DUMP for MEMORY DUMP

Operating Instructions

1. Place the blank cassette in CTA.
2. Enter 'PG CTA'.
3. System will display: 'CTA PURGED'.
4. Enter CASS.CREATE message.
5. System will display:

   '<mix>/CASS.CREATE WAITING SPURIUS/XXX AT NO FILE'

6. Enter 'AD <mix> CTA'.
7. Cassette will be created and the following message displayed:

   '<sau-file-name> TO XXX COPIED'

8. To write on the other end of the cassette; remove it from the drive, wind on by hand until the tape is exposed in the opening, and insert it the other way round in the cassette drive. Go back to step 2.

INITIATION OF THE STAND-ALONE UTILITIES

B 1000 System

1. Place the cassette in the cassette reader on the control panel. Rewind if necessary so that the BOT light is illuminated.
2. Set the register select switch so that it points to the L register. Set the cassette switch to SYSTEM, if applicable. Ensure that the SINGLE MIC/CONT switch is set to CONT and MICRO SELECT to NORMAL (B 1860), or SINGLE MICRO to NORMAL (B 1900).
3. Set MODE switch to the MTR position. Press the CLEAR and then the START buttons. If the cassette does not start, press HALT and CLEAR simultaneously and then START. After reading the cassette, the system will halt with @AAAAAA@ in the L register and the release level (for example 03.03.00) in the T register.
4. Set the MODE switch to the NORMAL position and press the START button. This will load the program and begin execution.

B 1830 System

1. Place the cassette in the cassette reader. If the cassette is not at BOT is will automatically rewind.
2. Enter 'ST'.
3. Press the CLEAR and START buttons.
4. Enter 'CR23' (this is the L register). '0101 0101 0101 0101 0101 0101 AAAAAA' will be displayed.
5. Enter 'SR' and press the START button. The program will load and begin execution.

The following halts are common to all stand-alone utilities:

CONTENTS OF L	DESCRIPTION
@000004@	cassette parity error
@111111@	no channel found with SPO attached
@222222@	invalid response after a test service request
@333333@	invalid status
@444444@	memory parity error
@555555@	invalid result descriptor
@666666@	SPO not ready

The allocation of channels in CMS is done from channel 0 to channel 15, which is contrary to Native Mode software.
Two pack controls, one on channel 7 and one on channel 9, reference DPC, DPD, DPA and DPB under Native Mode software and DPA, DPB, DPC and DPD under CMS software.

NOTE

Ensure that this does not cause confusion when using CMS Stand-alone cassettes.

COLDSTART loads the system software from a CMS library tape (a single reel tape) to disk (either disk pack or disk cartridge).

It can be either 800 BPI NRZ or 1600 BPI PE, and any tape unit can be used.

NOTE

If a file on disk is found with the same name as one on the tape, the tape file is loaded and the disk file is removed.

However, it is advisable to initialize the disk before using the COLDSTART program, or to re-initialize the disk if COLDSTART aborts.

COLDSTART is only necessary for new installations. Users of the 3.02.00 release need only initialize a user disk and use 'LD LOAD FROM <tape-id> TO <pack-id> ='.

Operation

'B1800/B1900 CMS COLDSTART mark 3.03.00' will be displayed and the cartridge/pack configuraton will also be displayed.

DISK PACK AND CARTRIDGE UNITS

CHANNEL x Dyy Dyy ...

The value x of the channel is given in hexadecimal (2,B,...). The different values of yy correspond to identifications of all the units connected to disk controls (DKA,DPB,...).

Parameters have to be entered by the operator, in answer to the following questions:

QUESTION: CMS DISK DRIVE -<DPX> OR <DKX>

RESPONSE: Enter the disk drive on which the system software is to be loaded; for example, 'DPA'. Any bad specification causes the program to ask for the same information again.

The magnetic tape configuration will then be displayed.

MAG TAPE UNITS

CHANNEL x MTy MTy ...

The value x of the channel is given in hexadecimal (2,B,...). The different values of y correspond to the identification of all the devices connected to magnetic tape controls (A,B,...).

QUESTION: CMS MAG TAPE UNIT -<MTX>

RESPONSE: Enter the mnemonic for the tape unit on which the COLDSTART tape is loaded. Any bad specification causes the program to ask for the same information again.

The tape label is read and the following messages are displayed.

'CREATION DATE OF TAPE: MM/DD/YY'

QUESTION: DISPLAY LOADED FILES? <YES OR NO>

RESPONSE: Enter 'YES' if a list of all files loaded is required. Enter 'NO' if no list is required.

The following information is then displayed:

'RELEASE LEVEL: N'

where N = 2 if the tape was created with a 3.01 version (or any previous version) of LD.
N = 3 if the tape was created with a 3.02 or later version of LD.

This information is obtained from the label of the tape.

QUESTION: 'LOAD ALL FILES <YES OR NO>'

RESPONSE: A 'YES' response directs COLDSTART to load all the files of the tape. A 'NO' response causes the following messages to be displayed:

'ENTER FILE NAME <12 CHAR OR ETX>'

The operator must enter the name of the file or the family of files (a character string appended with an '=') required. This message will be displayed again until a blank entry (ETX only) is made, or the limit of 20 entries has been reached.

NOTE

Introducing a file name or a family name with no entry in the directory of the tape will display the error message:

'THIS FILE IS NOT ON TAPE'

NOTE

The operator will be asked to enter another file name or family name.

Then the message:

'LOADED FILES:'

is displayed, followed (if requested) by the names of the files, five per line, loaded to disk. When all the requested files have been loaded, the message:

'END OF COLDSTART'

will be displayed.

NOTE

The message 'missing etx, try again' is displayed in the case of transmission without an ETX and the user is allowed to re-enter the information again.

NOTE

It is highly recommended to check that the displayed configuration of the disk and magnetic tape devices corresponds to the desired configuration.

Error Messages

The following error messages may be printed. The action to take is noted, except where the message is self-explanatory.

'NOT LIBRARY TAPE'

– attempt to COLDSTART with an incompatible tape. Ensure that the tape unit was specified correctly and that the tape is a LIBRARY tape.

'MAG TAPE NOT READY. CORRECT AND START'
'DISK NOT READY. CORRECT AND START'

– problem with the indicated device. When corrected, press the START button on the console panel.

'PROBLEM WITH MAG TAPE COLDSTART ABORTS'
'DATA ERROR ON READING TAPE LABEL COLDSTART ABORTS'

– problem encountered while reading the tape. Verify its density and try on another device if possible.

'PROBLEM WITH Dxx COLDSTART ABORTS' (xx = Px or Kx)
'DIRECTORY FULL COLDSTART ABORTS'
'NOT ENOUGH SPACE ON CMS DSK COLDSTART ABORTS'
'NOT A CMS DISK COLDSTART ABORTS'

– all these are fatal errors. The disk must be re-initialized before attempting to COLDSTART again.

'CANNOT REMOVE <FILE-NAME> PART OF DUAL PACK FILE'

– COLDSTART continues loading the other files. Only this file is not loaded.

'DISK NOT FOUND'
'MAG TAPE NOT FOUND'

– warning message displayed when a unit outside the controller's range has been specified.

'NOT STANDARD LABEL'

– warning message COLDSTART will try to resume processing as if the tape had a standard label.

'TAPE NO LONGER READY COLDSTART ABORTS'

'NOT A DISK FILE HEADER COLDSTART ABORTS'
TAPE MARK NOT FOUND COLDSTART ABORTS'

– internal messages indicating that COLDSTART is lost somewhere within a file. This can be caused by a hardware problem on the magnetic tape device or a corrupted copy of the System tape.

'COLDSTART INTERRUPTED' (flashing)

– the interrupt switch has been set during the load process.

CLEARSTART

FUNCTION

CLEARSTART determines the exact configuration of the system, builds various tables for the MCP, initializes memory links, executes the AVR of the system disk, loads MCP and finally, passes control to the Operating System.

SELECTION OF THE MCP FILE

At CLEARSTART time, the user has the ability to specify the MCP file he wants to start with.

This MCP file must have the name MCPnn, where nn is a three-digit integer.

The procedure is as follows:

- with the MODE switch set to MTR, press the Clear and the Start buttons.
- load 000xxx in the X register:

  set the register select switch so that it points to the X register.
  set the maintenance panel toggles to 000xxx (for example, to start with MCP123, set the panel toggles so that they represent the binary number: @00000000000 0000100100011@).
  press the LOAD button.

- set the MODE switch to NORMAL and press the START button.

CHANNEL SELECTION

The channels to be handled by the MCP can also be selected at CLEARSTART time via the maintenance panel toggles which must be set before RUN mode.

The toggles are represented here from left to right by six one-digit values (A,B,C,D,E and F); the first two digits (A and B) must be set to the value @FA@ and the last four digits (C to F) must be set to a value representing, from right to left, the numbers of the channels which have to be taken into consideration.

For example, setting the toggles to @FA0203@ corresponds to the following binary setting: @111110100000 001000000011@. This enables the MCP to handle only channels 0,1 and 9. All the other channels are disregarded.

No checks are made by the Operating System to detect mishandlings such as removing the SPO channel, removing the system disk channel or removing a channel belonging to a magnetic tape exchange. Therefore, this feature has to be used with great care.

In order to use a MCP file with a restricted configuration, the user must manipulate the maintenance panel toggles between the TAPE and the RUN mode to load the X register first, then to set up the required configuration.

Disk Initializers (CART.INIT/PACK.INIT)

GENERAL INFORMATION

All disks must be initialized in the CMS format before they can be used under MCP control. Each initializer comprises six basic passes:

1. Write addresses for every sector on the disk.
2. Verify all addresses on the disk.
3. Write a data pattern of @00@ throughout the disk and verify that the pattern has been written correctly.
4. Write and verify a pattern of @FF@.
5. Write and verify a pattern of @55@.
6. Write and verify a pattern of @E5@.

For a disk pack only, the verification in passes 3 through 6 consists of three sub-passes:

1. With normal offset.
2. With offset in.
3. With offset out.

In the event of an error, each I/O operation is retried five times and a record is kept of the number of retries (this appears on a KA or LR listing as the field 'ERROR COUNT').

If an I/O operation is not successful after five retries, the action taken depends on the type of disk and the operation attempted.

For a disk pack an attempt is made to relocate the bad sector (five spare sectors are available per cylinder for this purpose). If the sector cannot be relocated, or five sectors have already been relocated, the track in which the sector lies is completely removed.

For a disk cartridge, the discovery of a bad sector causes the track to be removed. If any sector in cylinder zero is discovered to be bad during passes 1 or 2, the initializer terminates.

ERROR LIMITS

The disk is regarded as unusable if:

1. The number of removed tracks exceeds 25, for a pack.
2. The number of removed tracks exceeds 5, for a cartridge.

The disk label and disk directory are built after initialization. The disk directory starts at the first track after track 0 where the total directory (available table + file name list + disk file header) can be located.

OPERATING INSTRUCTIONS

The operating instructions are similar for both initializers. After the cassette has been loaded, the title is displayed.

For cartridge:

'CMS DISK CARTRIDGE INITIALIZER MARK 3.03.00'

For pack:

'CMS DISK PACK INITIALIZER MARK 3.03.00'

The operator is then asked on which device the initializer parameters will be input.

'INPUT DEVICE <SPO OR CRD>'

If 'SPO' is specified, then the initializer operates in an interactive mode. If 'CRD' is specified, the input parameters are read from cards; one card for each disk.

CARD FORMAT

COLUMN	DESCRIPTION
1	unit (A thru H)
3-8	serial number (6 numeric digits)
10-16	disk identifier (up to 7 alphabetic characters)
21	disk type (R or blank)
23-25	maximum number of files (up to 999)
30-37	date (MM/DD/YY)
40-53	owner's name (up to 14 characters)

The last two fields are optional. All other fields are checked for validity. If an error is discovered, the card is rejected.

NOTE

The cards must be enclosed between '? DATA' and '? END' cards ('?' represents any invalid character for an 80 column card).

SPO INPUT

Parameters are entered interactively. As each field is input, it is checked for validity. If it is not valid, a suitable message is output and the field can be re-entered.

CARTRIDGE

QUESTION: WHICH CARTRIDGE -DK<X>

RESPONSE: DKA, DKB, ... DKH

PACK

QUESTION: WHICH PACK -DP<X>

RESPONSE: DPA, DPB, ... DPH

BOTH

QUESTION: ENTER 6 DIGIT SERIAL NUMBER

RESPONSE: number in the range 000000 -999999

QUESTION: ENTER PACK.ID

RESPONSE: up to 7 characters from the set 'A' thru 'Z', '0' thru '9', '.' or '-'.

QUESTION: ENTER CARTRIDGE TYPE <R or nothing>

QUESTION: ENTER PACK TYPE <R or nothing>

RESPONSE: R or ETX only

NOTE

R means restricted disk which is not yet implemented.

QUESTION: MAX NB OF FILES (UP TO 2805)

RESPONSE: any decimal number between 1 and 2805

NOTE

The default value is 254 for cartridge and 2805 for pack.

CARTRIDGE

When 256 files are requested, 8 sectors are allocated for the Available Table. For every additional 200 files declared, one sector is added until the maximum size of 14 sectors is reached.

PACK

The size of the Available Table is given by the following algorithm:

(number of files requested /32) + 3

The Available Table size will be extended so that the whole directory ends on an allocation unit boundary.

Therefore, on a 206 pack where the allocation unit is 8 sectors:

2805 files will give:

2805 sectors for headers
255 sectors for File Name List
91 sectors for Available Table (2805/32) + 3

3151 sectors for the whole directory.

The Available Table will be extended to 92 sectors to give a total directory length of 3152 sectors (the next integer multiple of 8 sectors).

2803 files will give:

2803 sectors for headers
255 sectors for File Name List
91 sectors for Available Table (2803/32) + 3

3149 sectors for the whole directory.

The Available Table will be extended to 94 sectors to give a total directory length of 3152 sectors (the next integer multiple of 8 sectors).

It must also be remembered that one sector of the disk file header is reserved for each file requested.

QUESTION: ENTER DATE -<MM/DD/YY>

RESPONSE: MM/DD/YY

QUESTION: ENTER OWNER'S NAME (UP TO 14 CH)

RESPONSE: up to 14 characters

The disk will now be initialized. While initialization is executing, entry of <ETX> and pressing the XMT button displays information concerning the initialize:

1. For pass 1 or 2 the message identifies the pass and the current cylinder number.
2. For passes 3 – 6 the message identifies whether a write or verify operation is in process, the pattern used and the current cylinder.

If the message 'DPEC ATTENTION' is displayed, it indicates that an extended result descriptor cannot be cleared from the DPEC. Press START twice on the processor panel.

NOTE

This can happen only with packs.

When initializing a cartridge, after completing the 6th pass, initialization/verification, the message:

'ENTER FINAL PATTERN (4 DIGITS) OR <ETX>'

is displayed. This enables the user determined pattern to be used. If this extra pass is not desired, enter <ETX>.

At the end of initialization, the following information will be displayed:

DRIVE
SERIAL NUMBER
PACK-ID
NUMBER OF REMOVED TRACKS
NUMBER OF COUNT ERRORS (NUMBER OF RETRIES)
NUMBER OF RELOCATED SECTORS (PACK ONLY)

If tracks were removed or sectors relocated, the address of each removed track and relocated sector is given. Finally, the message 'DP<X> INITIALIZED' or 'DK<X> INITIALIZED' is displayed.

It is then possible to re-initialize another disk (if SPO input was selected) by pressing the START button on the console panel.

TIME NEEDED

An estimation of the time needed to initialize packs follows:

35 minutes for a 205
50 minutes for a 206
150 minutes for a 207
70 minutes for a 225

ERROR MESSAGES

All the error messages are self-explanatory and related to a bad syntax detection.

MEMORY.DUMP

The function of the MEMORY.DUMP utility is to get the contents of the memory written into a memory dump file, named DMFIL00 (refer to the MEM.ANALYZER utility for the listing of the dump).

MEM.DUMP CASSETTE

Two cassettes are available; one for B 1830 systems and the other for B 1860 systems. This cassette must be executed in STAND-ALONE mode. If the system is still running at the time a memory dump is desired, it must be stopped with the INTERRUPT switch, or, if this is not successful, with the HALT button.

The operating instructions to run the cassette are as follows:

If running on a B 1830, note the value of the A register and press the CLEAR button.
Mount the appropriate MEM.DUMP cassette.
Switch to TAPE mode.
Press START.
When register L = @AAAAAA@, switch to RUN mode and press START.
When register L = @CCCCCC@, the dump is complete.
Rewind the cassette.
Perform a CLEARSTART.
Print DMFIL00 by executing MEM.ANALYZER.

A list follows of possible halts (in L register):

@111111@	Physical error on an I/O. Pressing the START will display the result descriptor in L.
@555555@	An error occurred while reading or writing the dump file to disk. By pressing START, 10 retries will be performed. To direct the dump to another disk, clear register L and then press START. The halt @999999@ will occur, as described below.
@666666@	Not enough space on the designated disk. Pressing START proceeds to the next halt, which will be @999999@.
@999999@	MCP tables have been destroyed (or one of the previous halts has occurred) and the program cannot find the channel and unit of the system disk. Enter in register L the parameters of the drive on which the file DMFIL00 is to be created. – channel number in the second digit – unit number in the last digit (first unit is 0) For example, @090001@ indicates that the dump file will be created on the second unit on channel 9.
@AAAAAA@	Switch to RUN mode and press START to continue. Information given at the time of halt: – release level in register T – memory load address in register X – dump file size in register Y – channel number in register S0A – unit number in register S1A
@CCCCCC@	Dump is complete. Pressing START will display in register L @TC000D@ where: T is the device type (@F@ for pack, @0@ for cartridge) C is the channel number D is the unit number or the location of the file DMFIL00.
@FFFFFF@	Not enough space in the disk directory. The dump cannot be resumed. Note: In order to regain space already allocated for the dump file, a SQUASH must be performed.

NOTE

If the system pack was seeking when the interrupt occurred, the dump file creation program can enter an infinite loop. To resume processing, press the HALT, then CLEAR, then START buttons.

Known problem: sometimes the dump cannot be taken, even after CLEAR and START. In that case, the cassette must be reloaded, but all status, registers, TAS and Scratchpads are lost.

SYSTEM HALTS DOCUMENTATION

SYSTEM HALTS

Most of these halts are traps for conditions that should not occur. When a halt occurs, a value is displayed in the L register as listed below. Except for the halts listed as recoverable, a dump must be taken and an FTR must be raised.

The system cannot be restarted, unless specified.

CLEARSTART Halts

000004	Cassette parity
C00001	SYSMEM not found
C00002	SYSMEM DFH corruption
C00003	BILINTERP not found
C00004	SYS-SUPERUTL not found
C00005	TL not found
C00006	SYSCONFIG not found
C00007	SYSCONFIG File type /= @22@ or SYSCONFIG machine type not in range @51@ to @60@
C00008	TRACE not found
C00009	MCP not found
C0000A	MCP file type /= @1C@
C0000B	System disk not write enable
C0000C	System disk not ready
C0000D	Other system disk exception
C0000E	TTY SPO exception
C0000F	CRT SPO exception
C00010	Soft console exception (B 1830)
C00011	Invalid data comm configuration (both DCP-1 and DCP-3 present) The system can be restarted with a correct setting of the channel mask using the maintenance panel toggles.
C00012	Memory parity error
C00013	System disk AVR failure (Available Table full)

MCP Halts

	SCL RESIDENT
4A0000	Cannot open SYSLANGUAGE file
4A0001	Invalid fetch value on read (in X).
	VM
4B0002	Recoverable error. Press START to continue.
4B000F	Fatal error, memory corrupted.
	GLOBAL
4D0000	Console interrupt. Press START
4D0001	Memory parity error. T contains: card and chip address decode and syndrome.
4D0002	READ or WRITE out of bounds occurred. See CD register. This is detected in the SCHEDULER. Mix number is S10A.
4D0003	Attempt to schedule a job while its ICB.ADDRESS is zero. This is detected in the SCHEDULER. Mix number is in S10A.

4D0004	Attempt to schedule a job with an empty STACK.LIST (probably an interpreter error). This is detected by the scheduler. Mix number is in S10A and ICB.ADDR in S09A.
4D0005	Attempt to schedule a job with a FIB.REQUEST. ATTENTION bit which does not correspond to a FIB segment. This is detected by the LOGICAL-IO selector. Mix number is in S10A and ICB.ADDR in S09A.
4D0006	Attempt to schedule a job with the FIB.REQUEST. ATTENTION bit set. The FIB segment type is valid (01), but is not in memory. This is detected by the LOGICAL.IO selector. Mix number is in S10A and ICB.ADDR in S09A.
4D0007	Attempt to schedule a job with the FIB.REQUEST. ATTENTION bit set. The FIB type is valid and present in memory, but PHYSIO did not put a fetch value in the FIB.REPLY word. Early return – Wait bit has disappeared. This is detected by the LOGICAL.IO selector. Mix number is in S10A and ICB.ADDR in S09A.
4D0008	ICB.STACK overflow. The error is detected by COMMUNICATE.SWITCH. Mix number is in S10A and ICB.ADDR in S09A.
4D0009	READ or WRITE out of bounds occurred (see CD register). As this error is detected by COMMUNICATE.SWITCH, the faulty module can be identified when analyzing the dump. Mix number is in S10A and ICB.ADDR is in S09A.
4D0010	Attempt to communicate with a non-existent resident module. The X register contains the desired module number. The T register contains: module number – displacement. The faulty module can be identified when analyzing the dump. Mix number is in S10A and ICB.ADDR is in S09A.
4D0011	Attempt to communicate with a non-existent overlay module. The X register contains the module number multiplied by 12. The T register contains: module number – displacement. The faulty module can be identified when analyzing the dump. Mix number is in S10A and ICB.ADDR in S09A.
4D0012	BOOTSTRAP does not find an empty slot in the Mix table. SCL is the faulty module.
4D0013	Master tries to execute Slave reserved micro code.
	SCL HANDLER
630002	Spo.Previous.Op was Test.Wait.Receive
630006	Invalid fetch value after a Load DS (in X)
631000	Pack error (System Pack Lockout)
	SCL INPUT
640002	Invalid fetch value after a load Data Segment (in X)
	PHYSIO RESIDENT
F00001	Communicate with wait variant issued on a program already waiting I/O.
F00002	'Wait only' communicate issued on a non-opened file.
F00004	Disk search communicate issued on a non-opened file.
F00005	Duplicate disk opened.
F00008	Single device Read/Write/Close communicate issued on a non-opened file.
F00009	Invalid single device Comm.Verb.
F0000A	ICMD Read/Write/Close communicate issued on a non-opened file.
F0000B	Open communicate on a file in use.
F0000C	Disk channel neither active nor tested while in result descriptor analysis.
F0000D	FCB queued twice
F0000E	FCB already de-linked
F0000F	FCB not linked
F00010	DCB already de-linked
F00011	DCB not linked

F00012	System disk not ready
F00013	Invalid STC on Test.Op
F00014	ICMD channel neither active nor tested while in result descriptor analysis.
F00017	Invalid disk communicate.
F00018	Disk Read/Write communicate issued on a non-opened file.
F00019	Attempt to issue a disk Read/Write communicate on an active file.
F00020	Attempt to issue a disk Read/Write on a file related to an invalid device pointer.
F00021	Invalid STC detected while in Handle Data Transfer
F00022	Invalid link between FCBs during search.

DISK OPEN

F10001	SYSMEM lock failure
F10002	Invalid overuse
F10003	Invalid EOF pointer
F10004	Invalid reply word
F10005	Invalid lock holder

DISK R/W NEXT AREA

F40002	Wait error
F40003	Wait error
F40004	Wait error
F40005	Wait error
F40006	Wait error
F40007	SYSMEM lock error
F40008	Wait error
F40009	Wait error
F4000A	Wait error
F4000B	Wait error
F4000C	Wait error
F4000D	Wait error
F4000E	Wait error
F4000F	Wait error
F40010	Available table size error
F40011	Attempt to allocate beyond Available table size
F40012	Attempt to allocate beyond Available table size
F40013	Allocation error
F40014	Allocation error

PHYSICAL I/O OVERLAY LOADER

FFFFF1	Disk open overlay problem
FFFFF2	Disk open new file overlay problem
FFFFF3	Disk close overlay problem
FFFFF4	Disk areas overlay problem
FFFFF5	Disk AVR overlay problem
FFFFF6	ICMD overlay problem
FFFFF7	Disk Purge overlay problem
FFFFF8	Disk crunch overlay problem
FFFFF9	Single device open overlay problem
FFFFFA	Single device close overlay problem
FFFFFD	RL overlay problem
FFFFFE	Single device AVR overlay problem

DATA COMMUNICATIONS HALTS

DC0014	Neither Buffer 1 nor Buffer 2 can be transferred. A Read is asked.

SYSTEM DEPENDANT FETCH VALUES

The I/O system failures described in this section should not occur. However, if they do, instead of leading to a System Halt condition, they cause a fetch value to be returned to the running program, as listed. A 'COMMUNICATE ERROR' DS/DP message is displayed. A Dump of the program supplies the Fetch value in the Communicate Parameter Area.

However, it may be impossible to Dump the program if it remains in Terminating status. In this case, the only way to access the Fetch value is to get a Memory Dump when the minimum of programs remain in the Mix.

Physical I/O

	PHYSIO RESIDENT
80F015	Mix waiting I/O bit reset before the end of a communicate with wait variant.
80F016	FIB terminating mask flag not reset after completion of previous I/O.
80F01E	Open communicate issued on an already opened file.
80F01F	Disk close communicate issued on a non-opened file.
80F033	No local SPO on system.
	DISK OPEN OLD FILE
80F120	Other use error
80F122	EOF error
80F12A	SYSMEM lock error
	DISK OPEN NEW FILE
80F201	Directory error
80F22A	SYSMEM lock error
	DISK CLOSE
80F313	User count error
80F31E	FCB structure error
80F321	Record size error
80F322	EOF error
80F32A	SYSMEM lock error
	DISK AREAS
80F401	Program status error
80F402	File status error
80F403	Bad verb
80F404	Disk file header error (NB. of sectors/block = 0)
80F42A	SYSMEM lock error
	DISK PURGE
80F701	Area size error
80F707	Available Table size error
80F72A	SYSMEM lock error
	DISK CRUNCH
80F801	Last area size error
80F802	Area allocation error
80F803	Area length error

80F807	Available Table size error
80F813	User count invalid
80F81E	FCB structure error
80F820	CLOSEMODE error
80F821	Record size error
80F822	EOF error
80F82A	SYSMEM lock error
	SD OPEN
80F91E	Duplicate open
20F9FF	Device cleared
	SD CLOSE
80FA20	Invalid MYUSE

Logical I/O

80FF10	Bit CD(2) or CD(3) is set when entering in the resident
80FF20	Buffer status is invalid in the routine Can I Read Again
80FF30	Buffer status is invalid in the routine Sequential Write
80FF40	Buffer status is invalid in the Get routine
80FF50	Buffer status is invalid in the Put routine
80FF70	Bit CD(2) or CD(3) is set when entering in the Open Segment
80FF80	FIB Data Segment is not found in the IFNB Segment
80FF90	Bit CD(2) or CD(3) is set when entering in the Logical Close Segment
80FFA0	Attempt to divide by zero
80FFF0	Bit CD(2) or CD(3) is set when leaving logical I/O module
80FFFF	72 FIBs are already open

DATA COMMUNICATIONS ERRORS

When one of the following Data Communications errors occurs, the message

<285> DC ERROR xx ON DC y

is issued, displaying xx – the decimal value of the L register last byte
yy – the DCP number

xx	
01	Memory corruption problem. A message is to be added into the input queue. The message link cleared at dequeue available space time is no longer cleared.
02	Cannot lock the Host to send a Nack message after a B 1000 No Space Condition.
03	The message length given by the Liege header is less than the number of bytes contained in the Host.
04	The message length to empty is still greater than 255 bytes, and the transfer in count supplied by the Host is less than 255 bytes.
05	During a Read sequence (B 1000 in receiving mode) a service request is issued by the Host with the corresponding status HTC.READ = TRUE HTC.BUSY = FALSE HTC.BUF1 = FALSE HTC.BUF2 = FALSE
06	During a Write sequence (B 1000 in sending mode) a service request is issued by the Host with the corresponding status

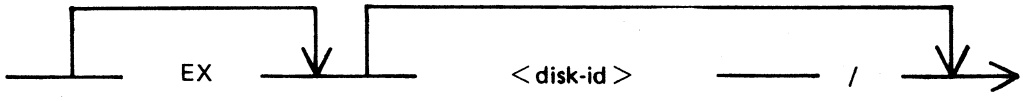
	HTC.WRITE = TRUE HTC.BUSY = TRUE HTC.BUF1 = TRUE HTC.BUF2 = TRUE Therefore, there are no free buffers to continue the Write.
07	When the DCP is in sleeping mode, the only valid sequence is READ. The HTC status is located in S6A.
08	Invalid DCP status.
09	For a new input message, the transfer-in count of the first hardware buffer is less than 6 bytes.
10	The sequence byte is wrong. A buffer may be lost.
11	Invalid HTC status (see S6A).
12	Cannot find MTRINTERP.
13	DCP is in No Space condition and the next interrupt gives an invalid HTC status (see S6A).
14	Channel control word corrupted.
15	Transfer-in count value set to zero during a Nack read phase
16	Invalid Liege Header during the Nack Read phase.
17	During a Read phase the transfer in count is less than 255 bytes while Busy bit is true.
18	During a Read phase the transfer-in count is less than 255 bytes, and two buffers are filled.
19	The HTC CNTL register is null after a test and lock command.
21	Protocol error.
26	Firmware halt (take DCP dump).
27	Unexpected number of bytes received by HOST DMA.
28	Unexpected status received while in ROM mode.
29	DCP Memory parity error.
30	Unexpected Jump in ROM.

APPENDIX A
COMPLETE RAILROAD DIAGRAMS

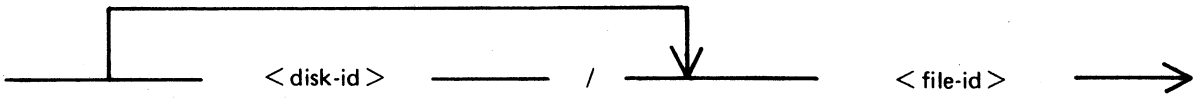
This appendix gives the railroad diagrams for all the CMS-common intrinsics and utilities, including SORT and CO, in alphabetical order. These diagrams give the complete input message formats, for ease of reference.

For details of the meaning of these messages, refer to the text.

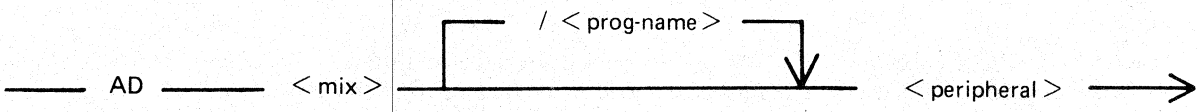
In the following diagrams the <ex-option> is defined as :



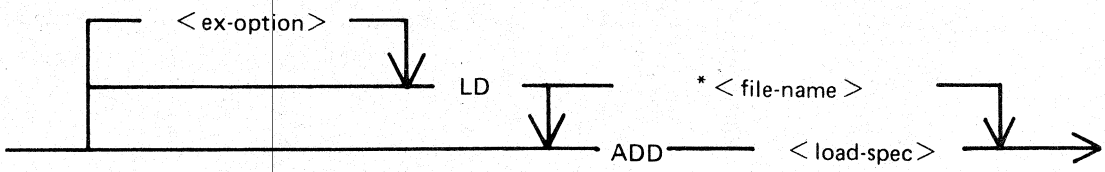
and <file-name> is defined as :



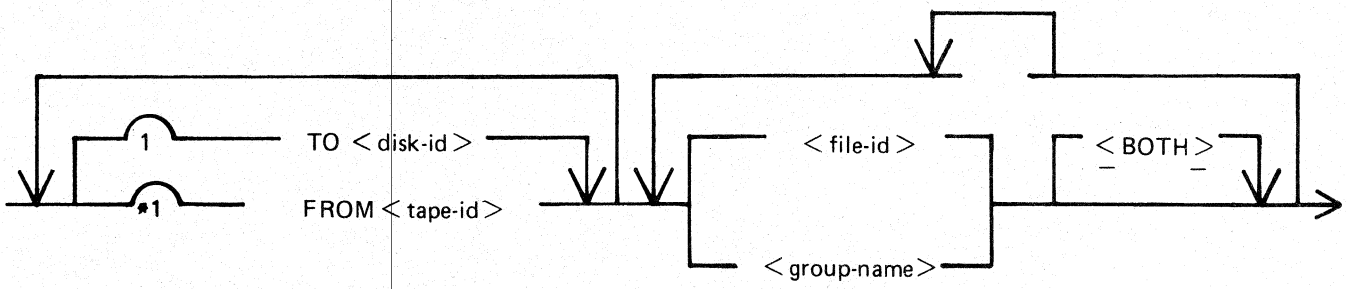
AD intrinsic



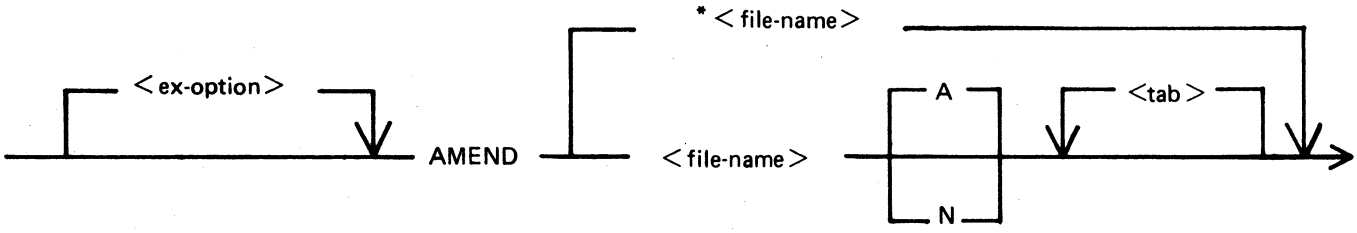
ADD utility



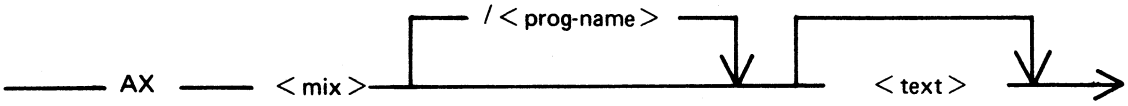
< load-spec > is defined as :



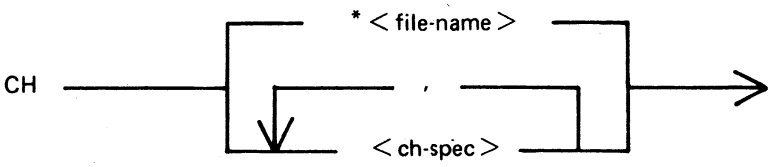
AMEND utility



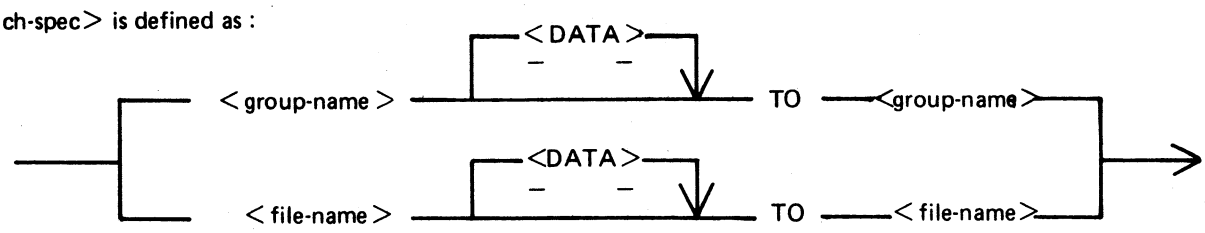
AX intrinsic



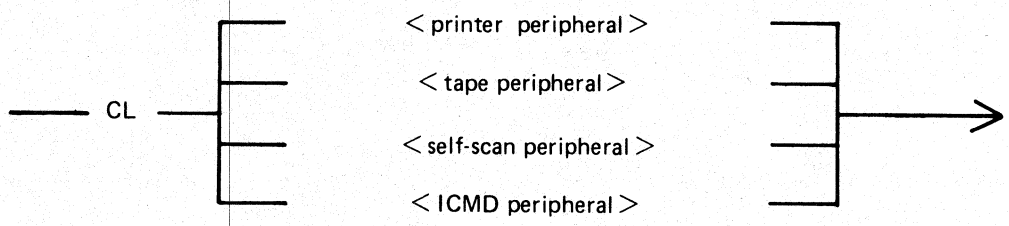
CH utility



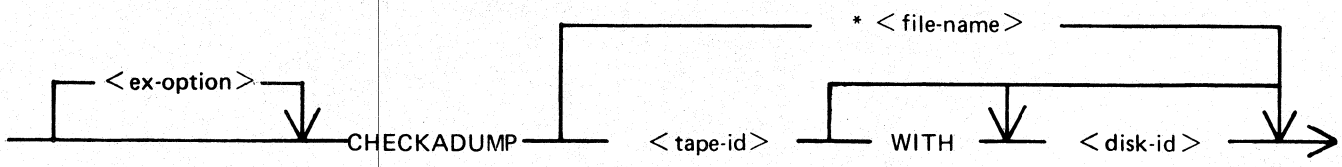
< ch-spec > is defined as :



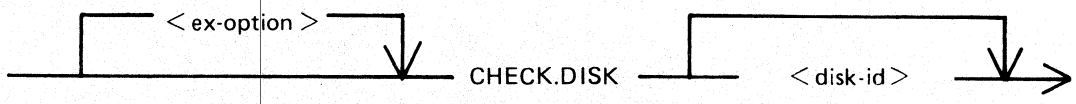
CL intrinsic



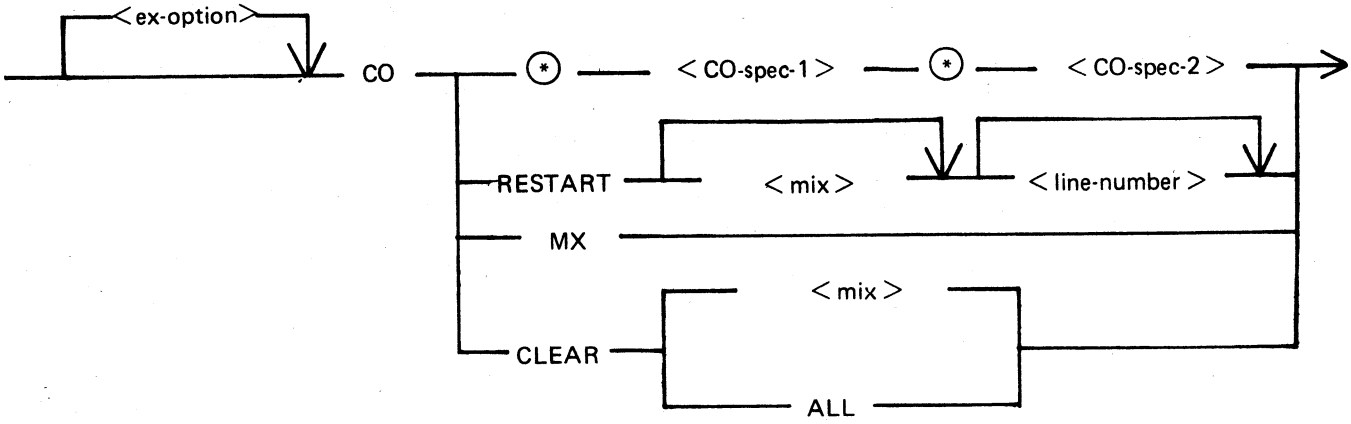
CHECKADUMP utility



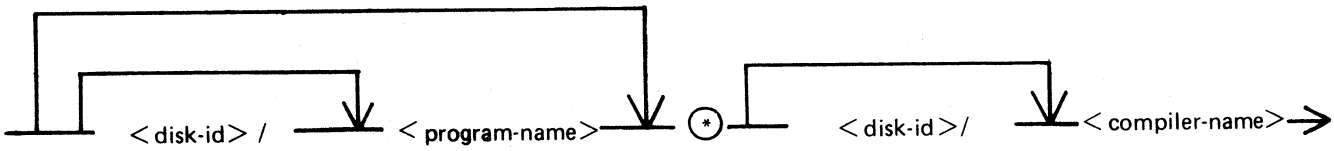
CHECK.DISK utility



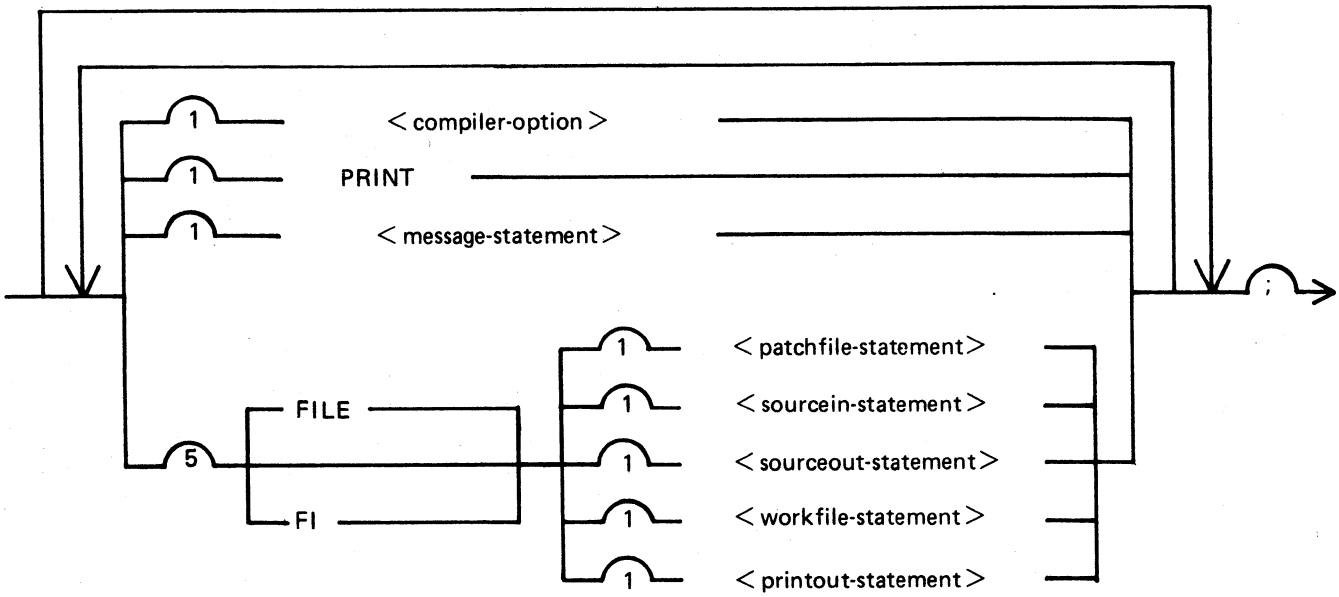
CO utility



<CO-spec-1> ia defined as :

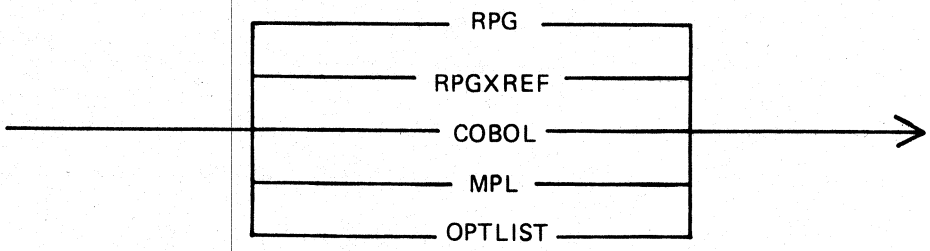


<CO- spec-2> is defined as :

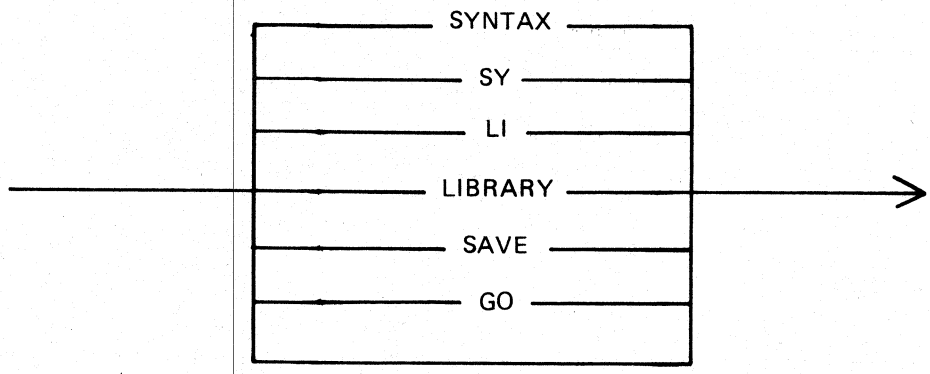


(*) indicates the allowed postions for macro cells

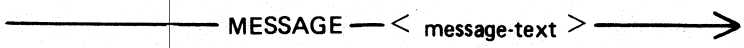
The < compiler name > is defined as :



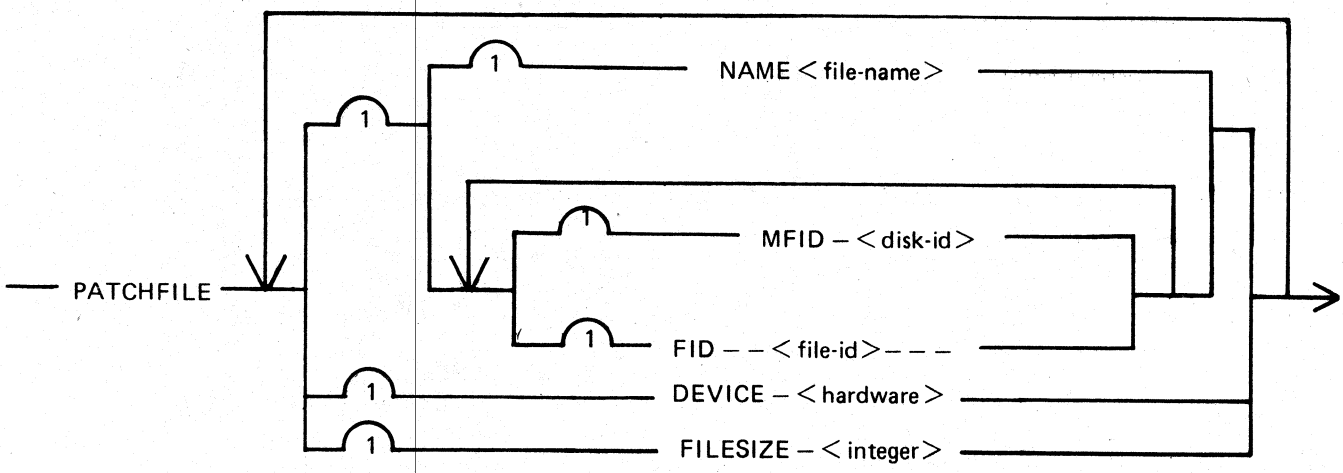
The < compiler-option > is defined as :



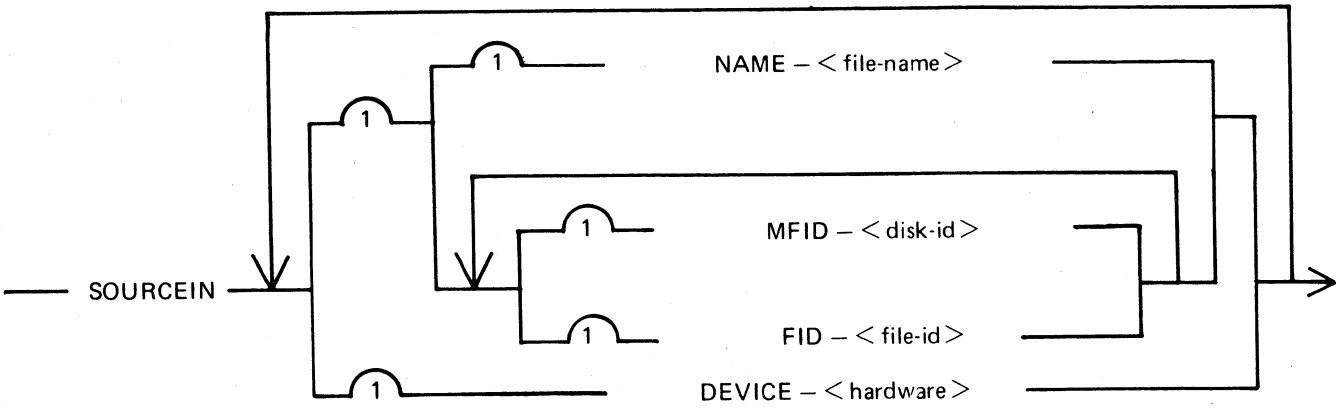
The < message-statement > is defined as :



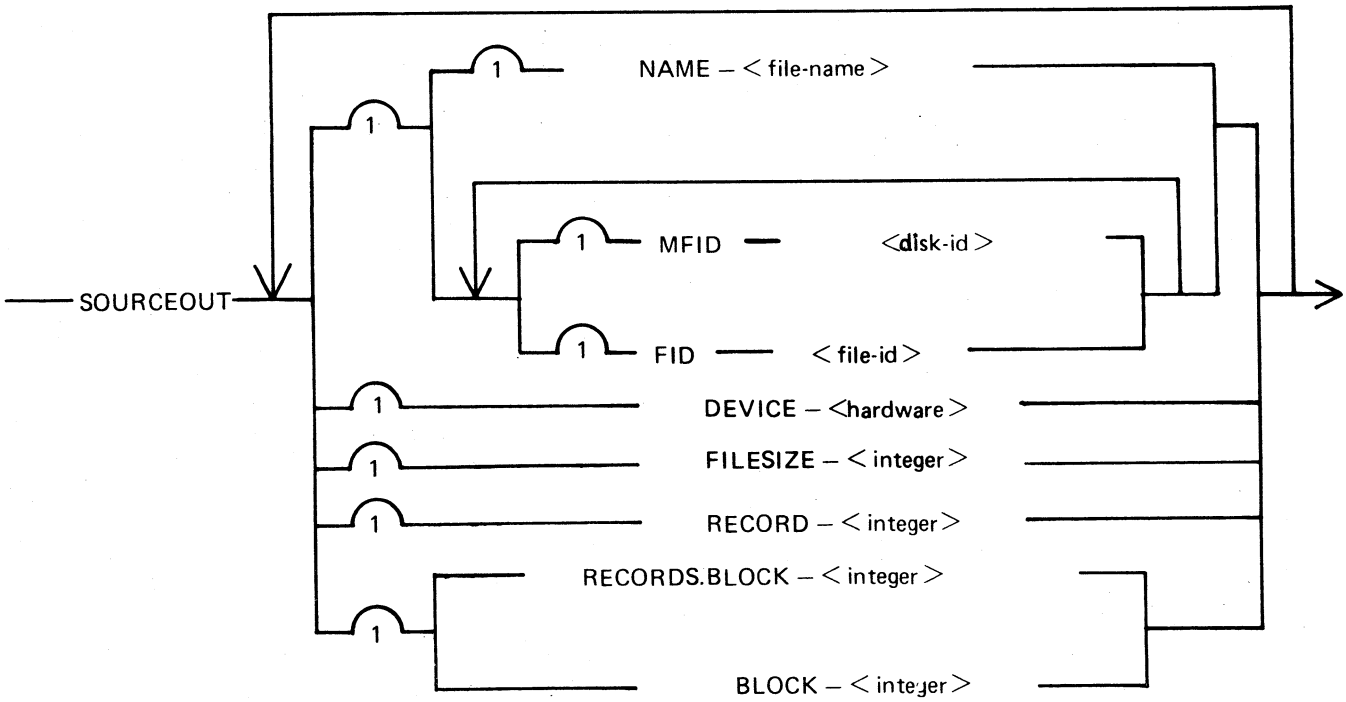
< patchfile-statement > is defined as :



< sourcein-statement > is defined as :



< sourceout-statement > is defined as :



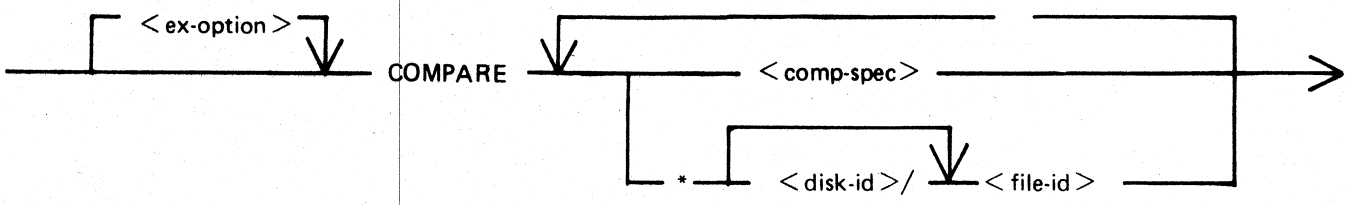
< workfile-statement > is defined as :

—WORKFILE —— MFID —— < disk-id > —

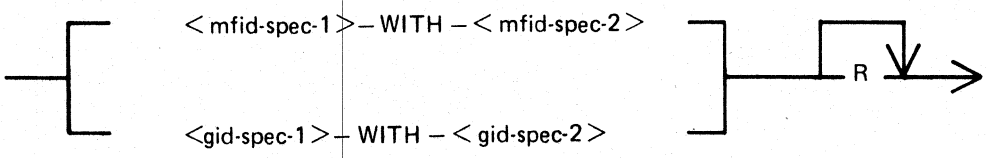
< printout-statement > is defined as :

—PRINTOUT —— DEVICE —— < hardware > —

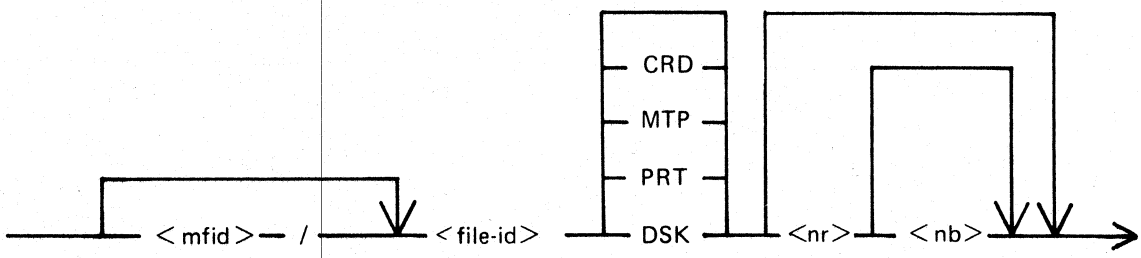
COMPARE utility



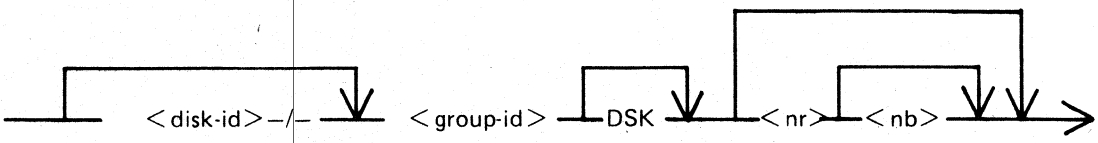
< comp-spec > is defined as :

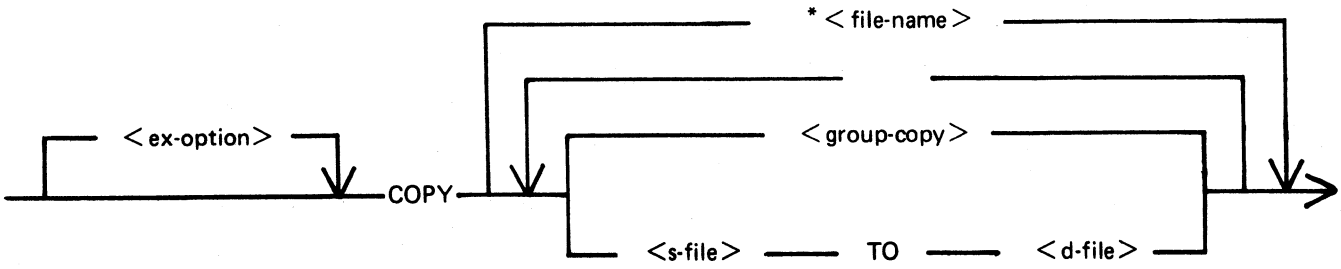


< mfid-spec > is defined as :

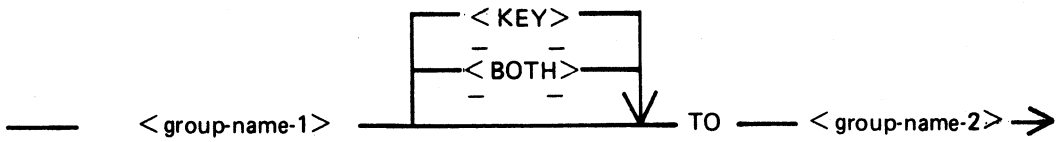


< gid-spec > is defined as :

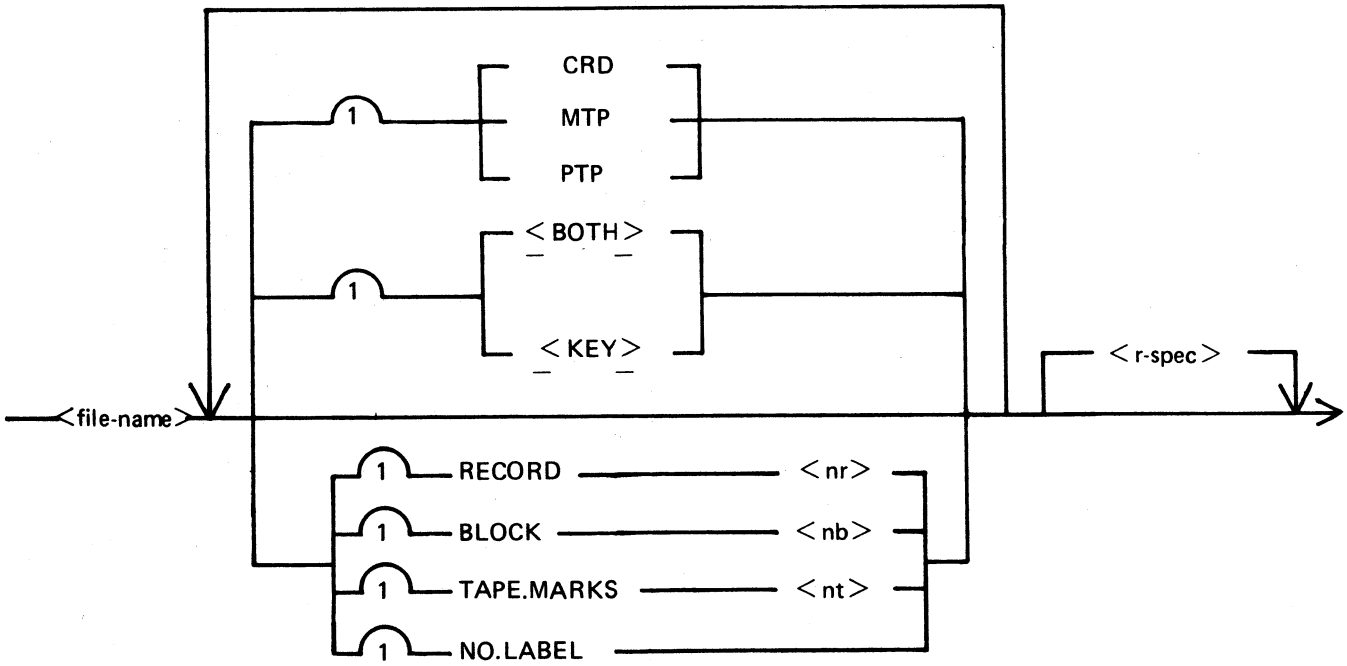




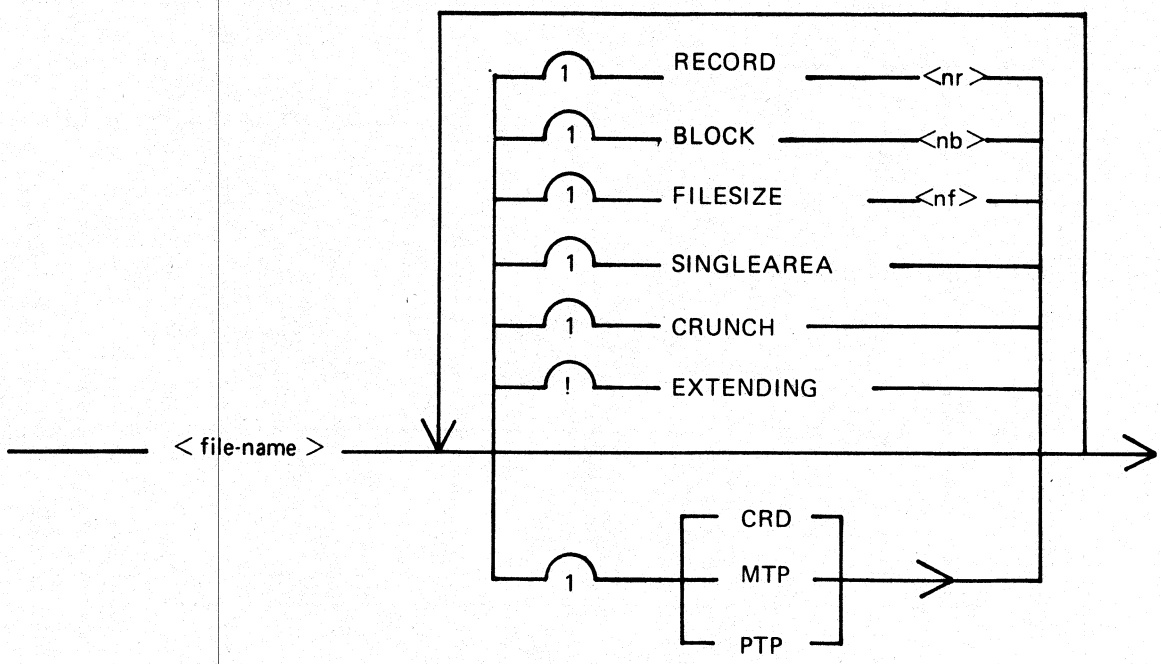
< group-copy > is defined as :



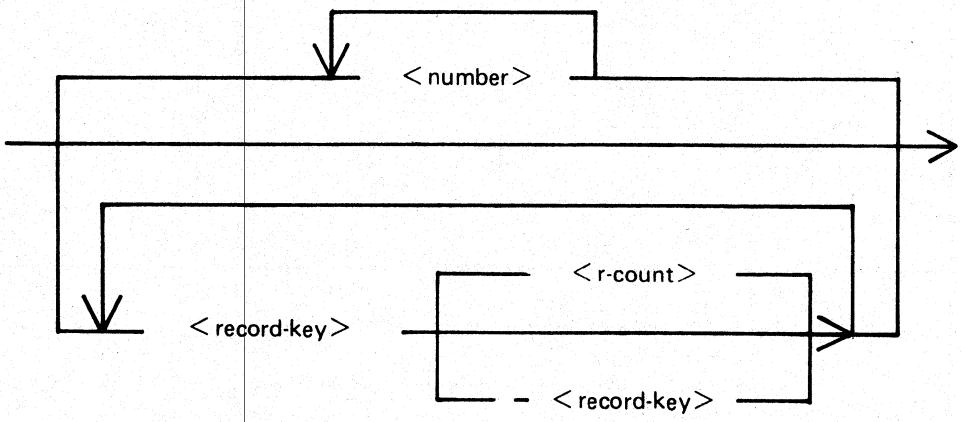
< s-file > is defined as :



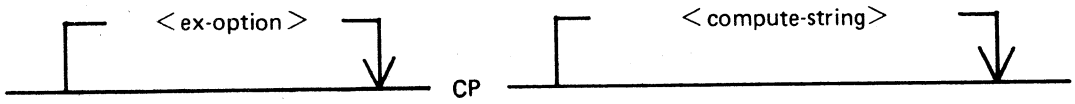
< d-file > is defined as :



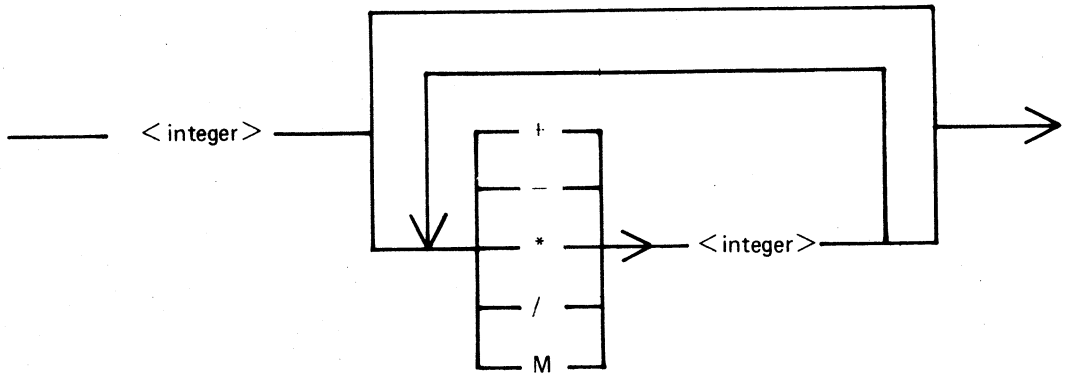
< r-spec> is defined as :



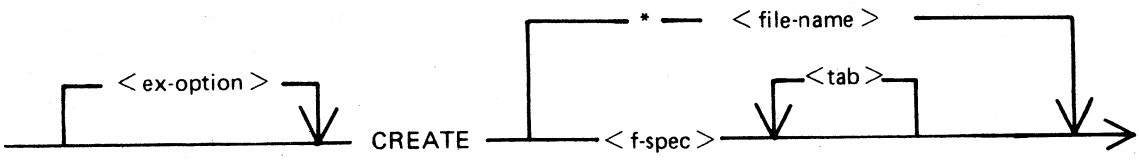
CP utility



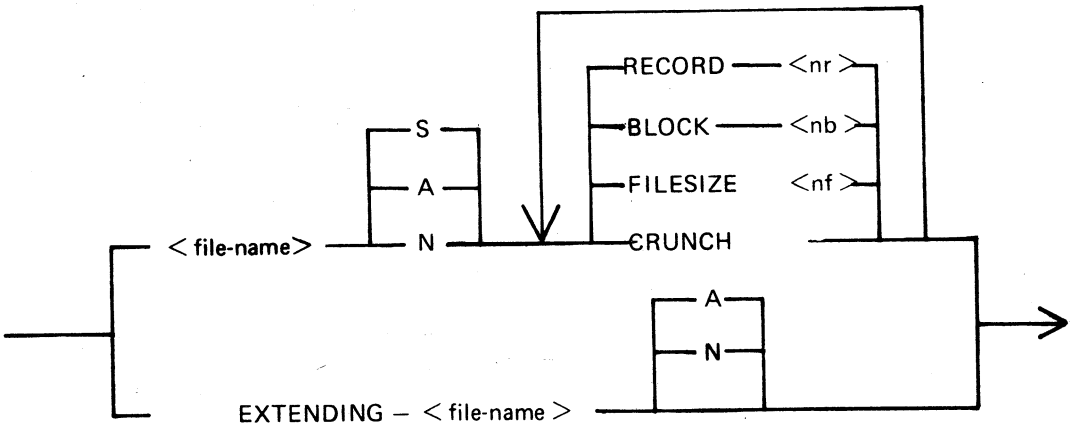
<compute-string> is defined as :



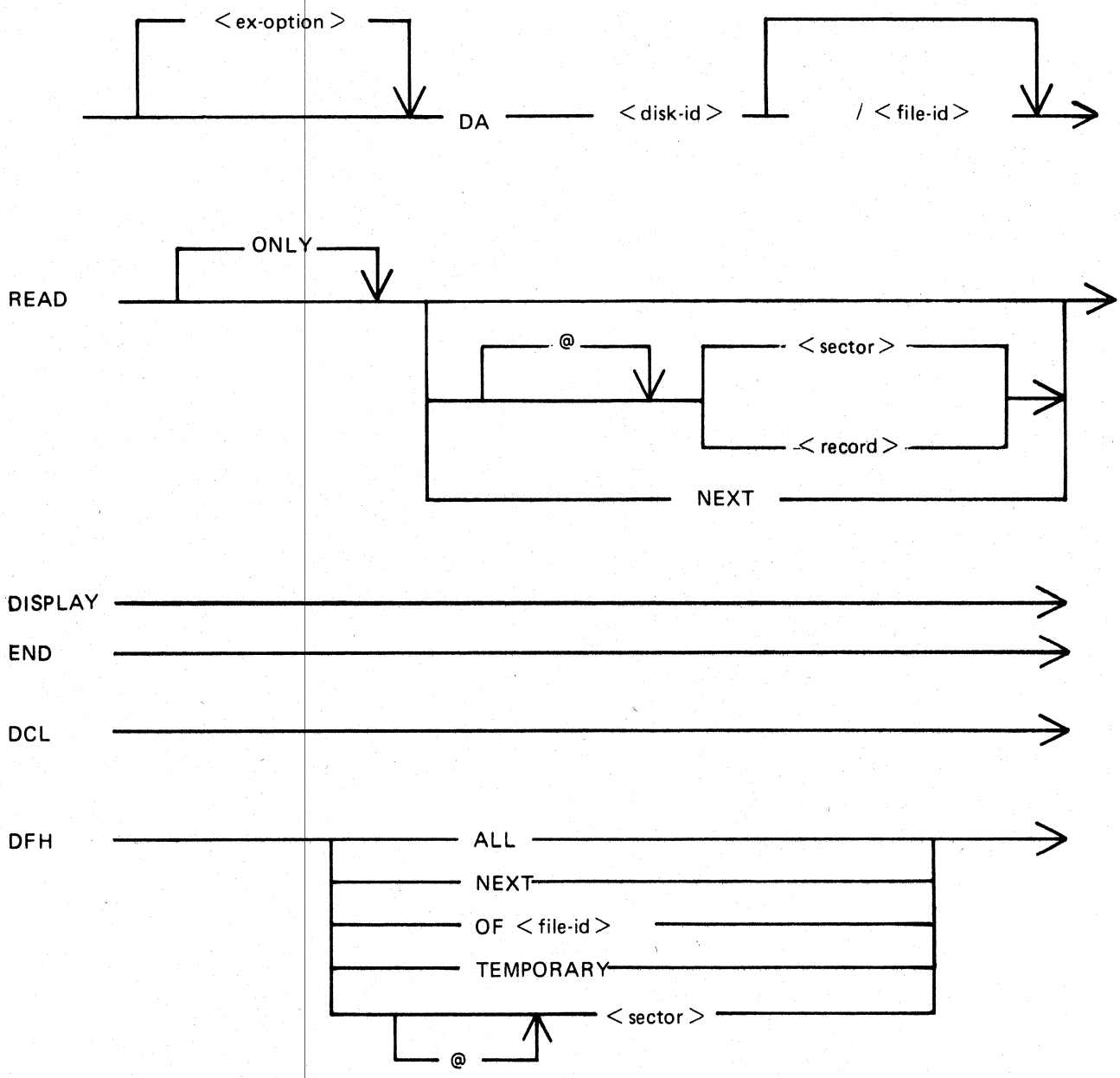
CREATE utility

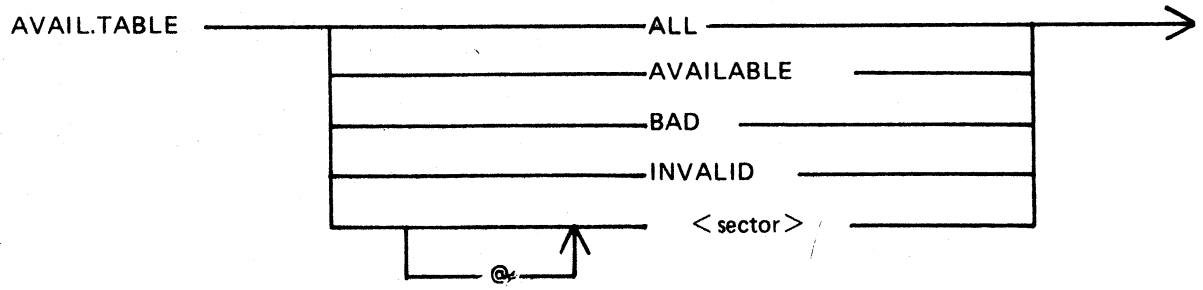


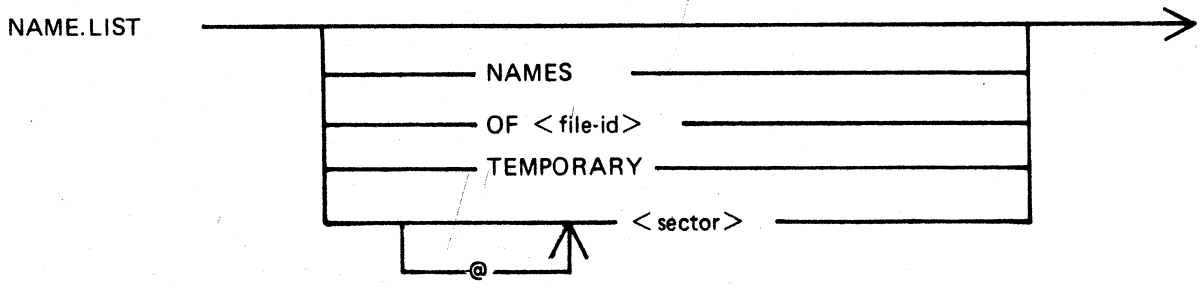
<f-spec> is defined as :



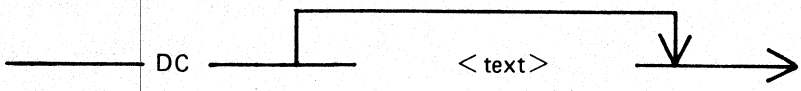
DA utility


< ex-option >
DA
< disk-id >
/ < file-id >
ONLY
READ
@
< sector >
< record >
NEXT
DISPLAY
END
DCL
DFH
ALL
NEXT
OF < file-id >
TEMPORARY
< sector >
@


AVAIL.TABLE
ALL
AVAILABLE
BAD
INVALID
<sector>
@

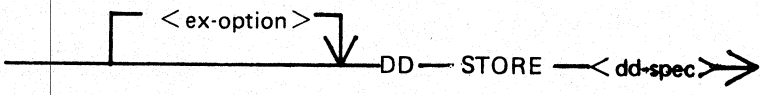

NAME.LIST
NAMES
OF <file-id>
TEMPORARY
<sector>
@

DC intrinsic

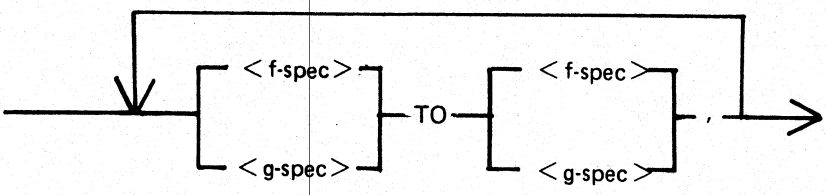


DD utility

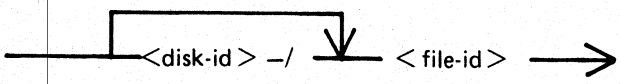
FUNCTION – STORE



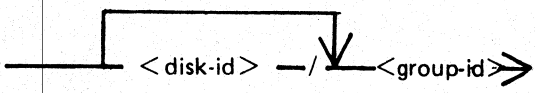
< dd-spec > is defined as :



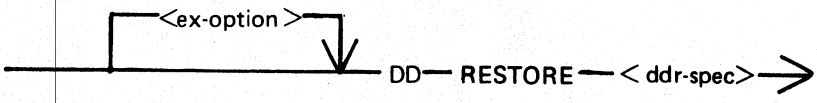
< f-spec > is defined as :



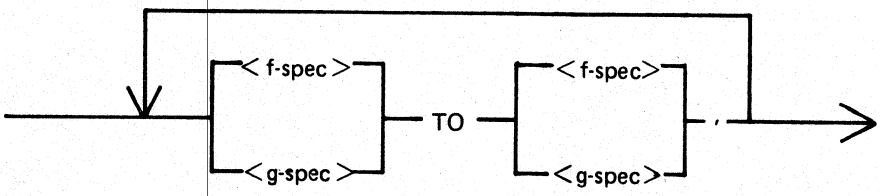
< g-spec > is defined as :



FUNCTION – RESTORE

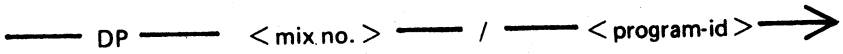


< ddr-spec > is defined as :

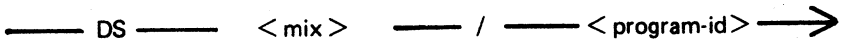


The < f-spec > and < g-spec > are defined the same way as in STORE FUNCTION.

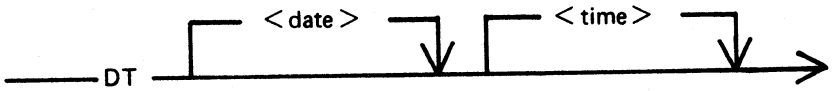
DP intrinsic



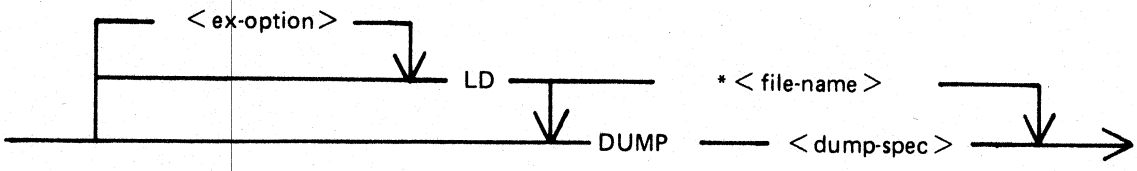
DS intrinsic



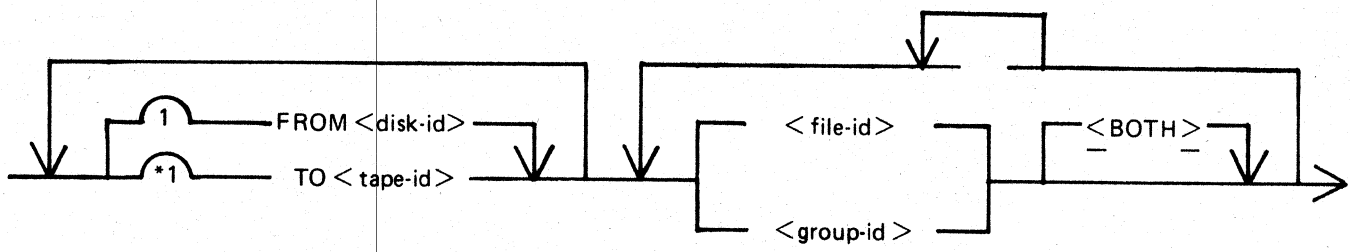
DT intrinsic



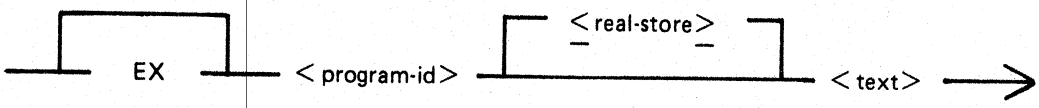
DUMP utility



< dump-spec > is defined as :

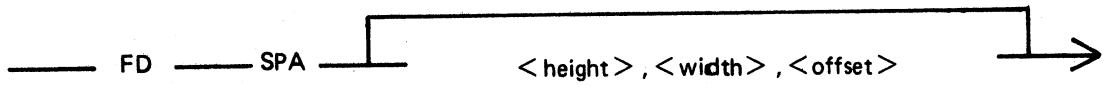


EX intrinsic

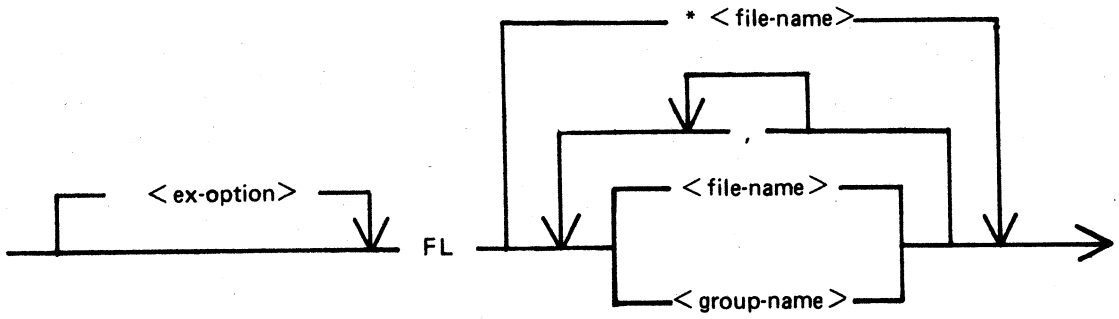


< text >

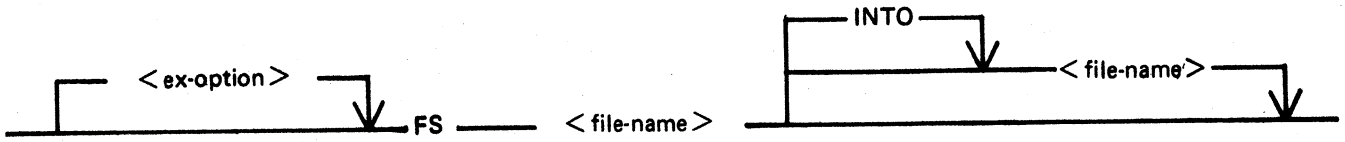
FD intrinsic



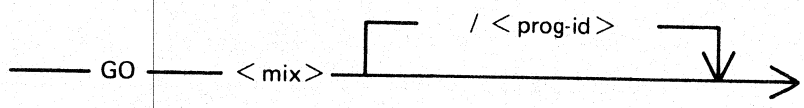
FL utility



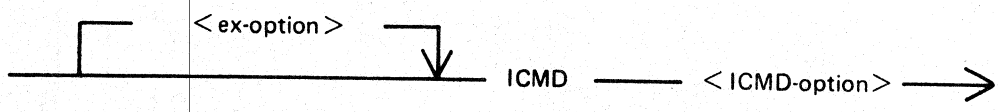
FS utility



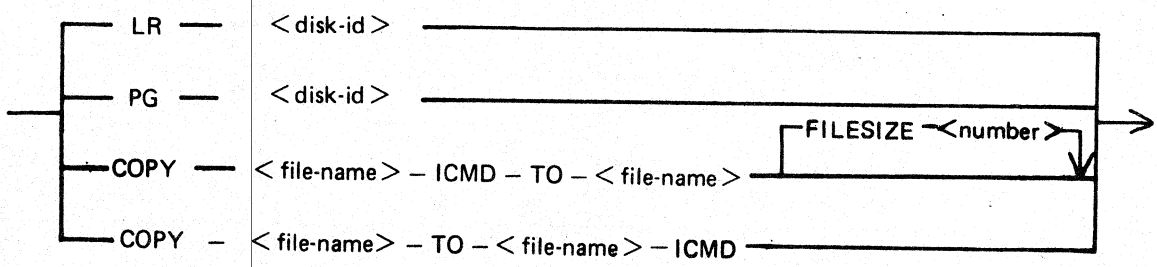
GO intrinsic



ICMD utility



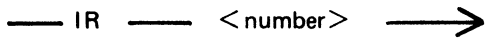
< ICMD-option > is defined as :



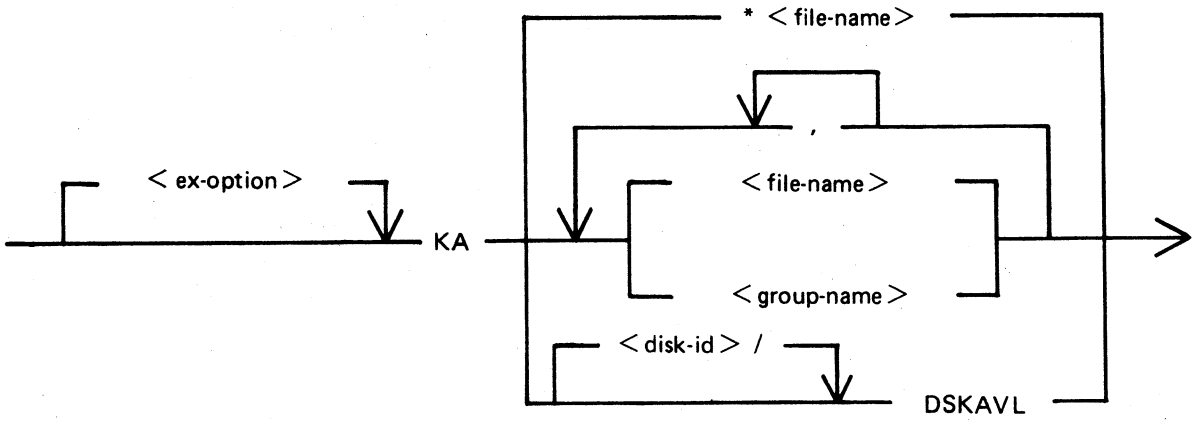
< file-name > for this utility is defined as :

< disk-id > / < file-id >

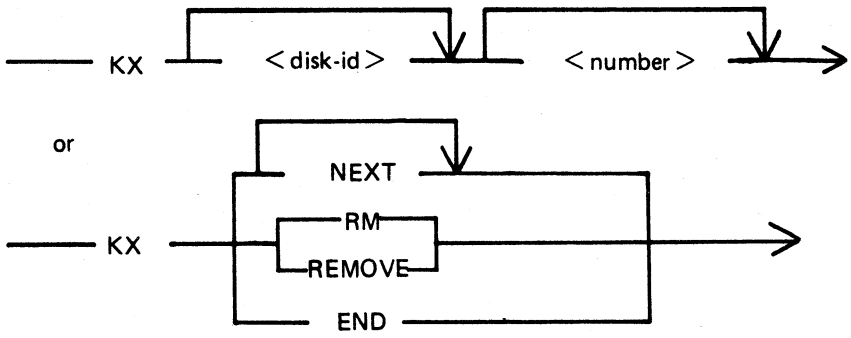
IR utility



KA utility



KX utility



LB utility

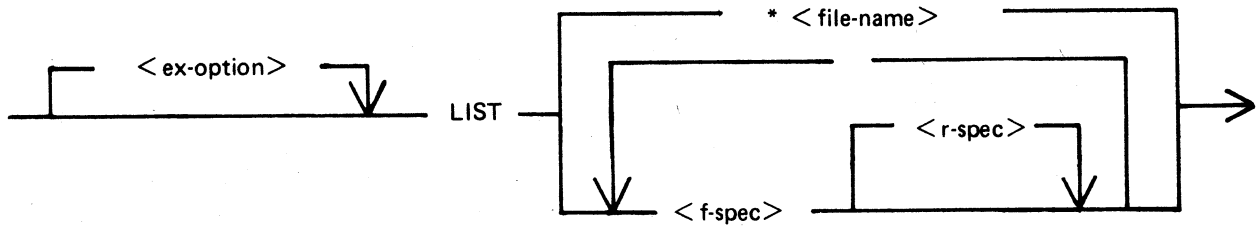
— LB —

LD utility

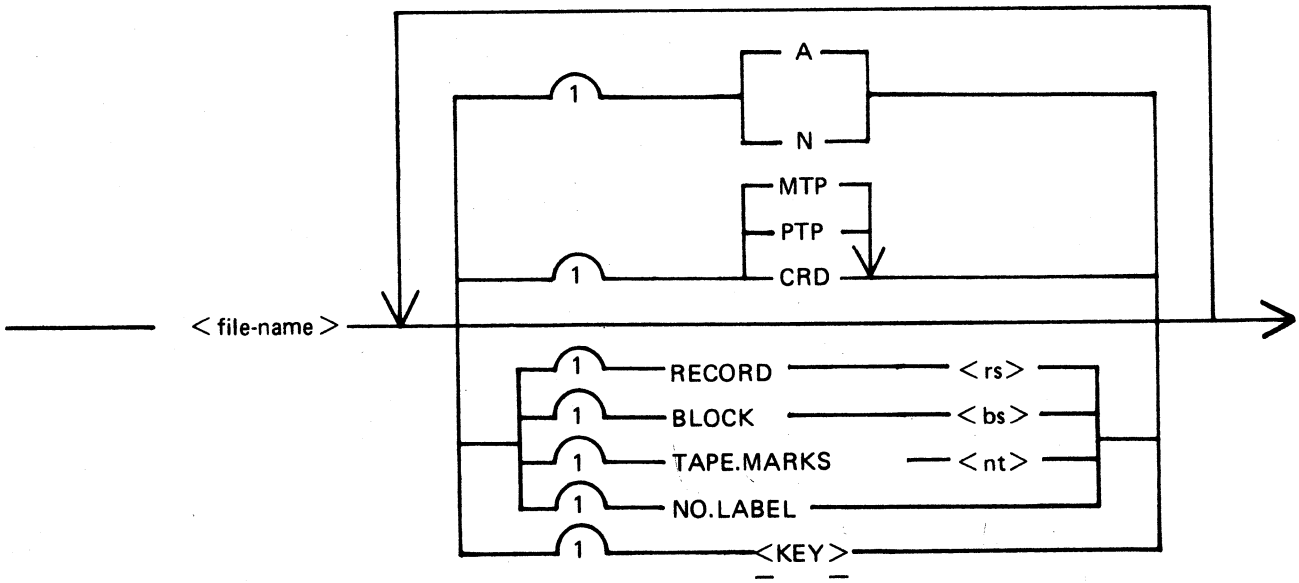
See ADD, DUMP, LOAD, and UNLOAD

LF utility

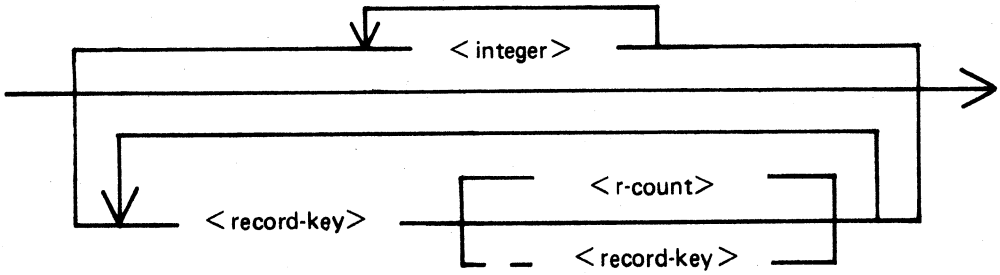
— LF —



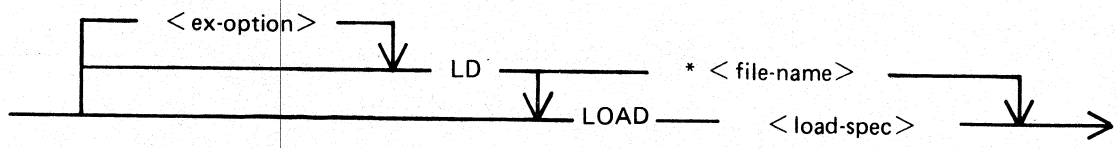
<f-spec> is defined as :



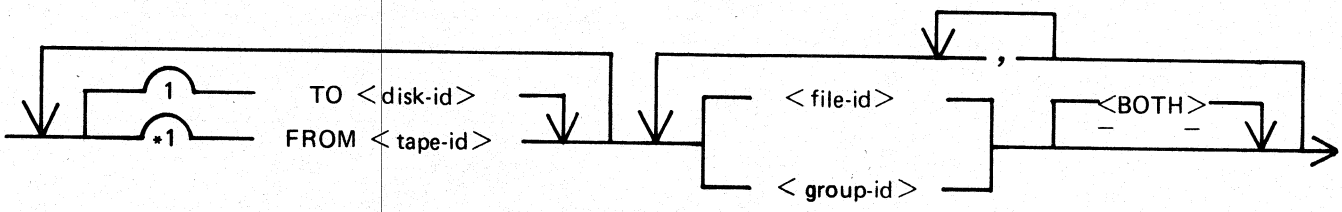
<r-spec> is defined as :



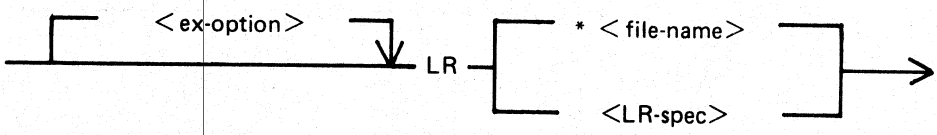
LOAD utility



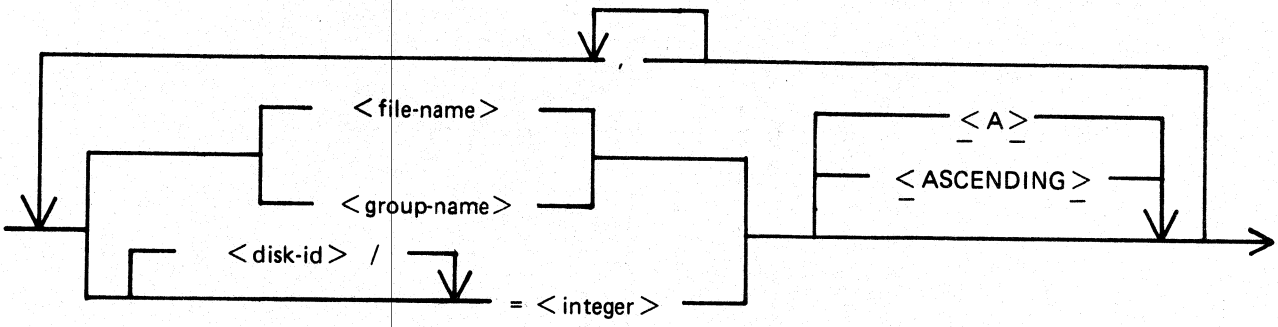
<load-spec> is defined as :

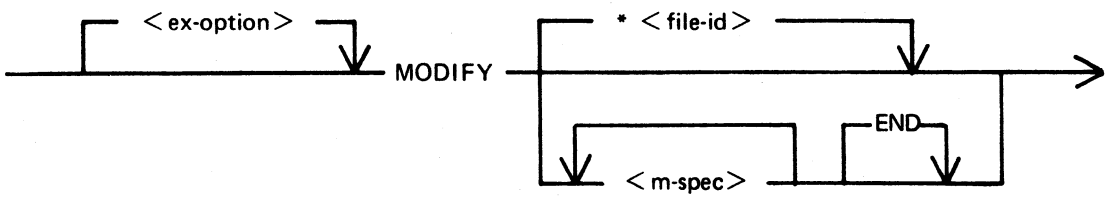


LR utility

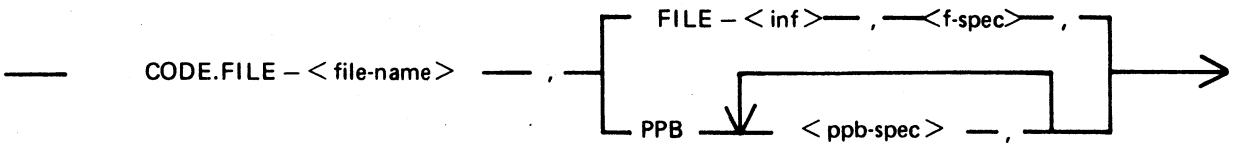


< LR-spec > is defined as :





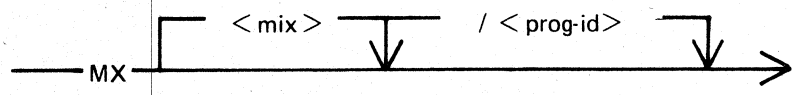
<m-spec> is defined as :



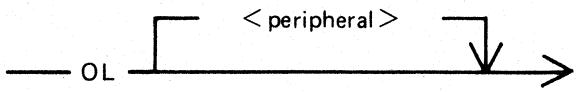
<f-spec> is defined as :

<ppb-spec> is defined as :

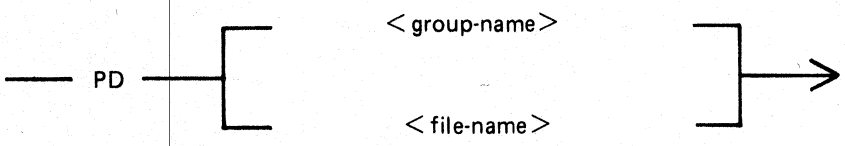
MX intrinsic



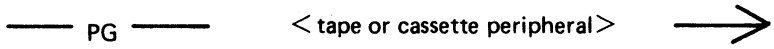
OL intrinsic



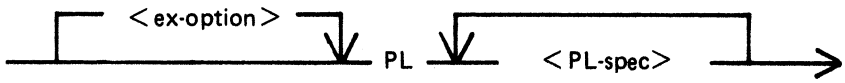
PD utility



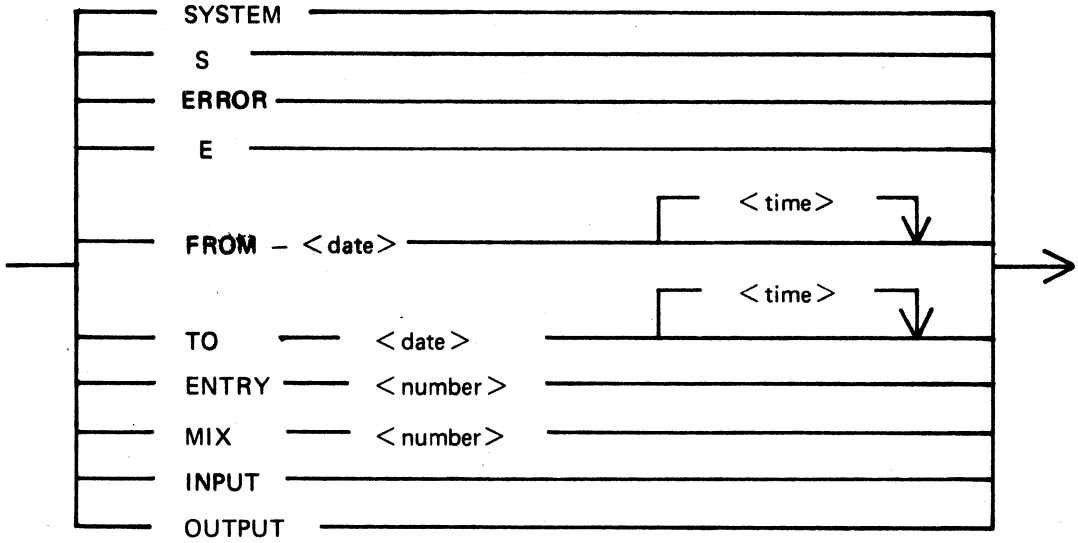
PG intrinsic



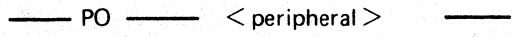
PL utility



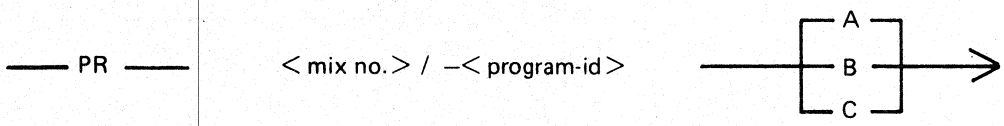
< PL-spec > is defined as :



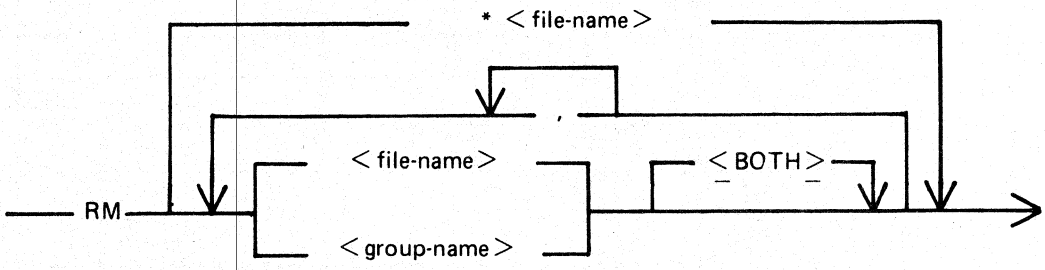
PO intrinsic



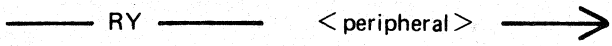
PR intrinsic



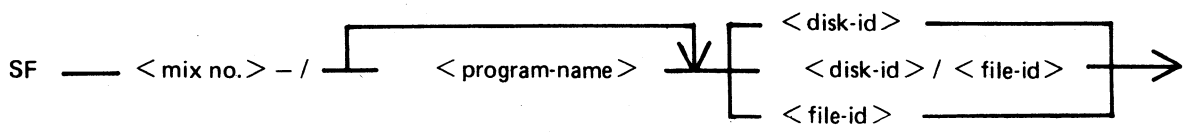
RM utility



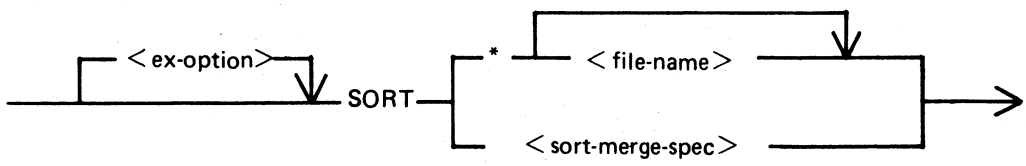
RY intrinsic



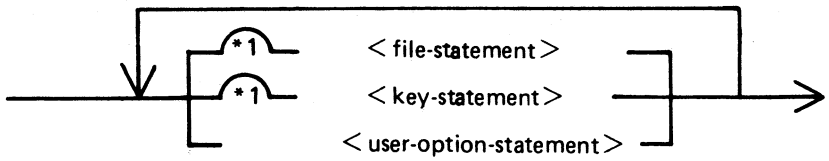
SF intrinsic



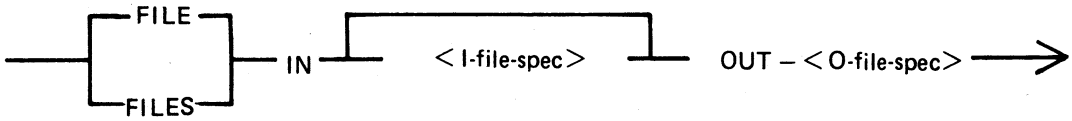
SORT utility



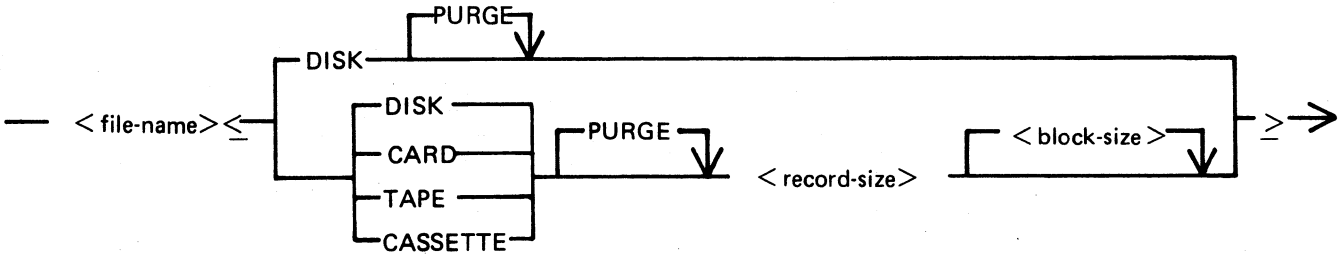
< sort-merge-spec > is defined as :



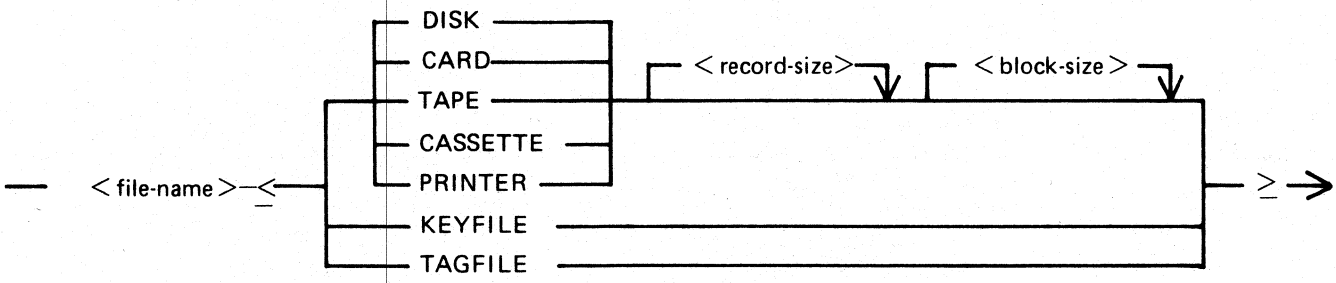
< file-statement > is defined as :



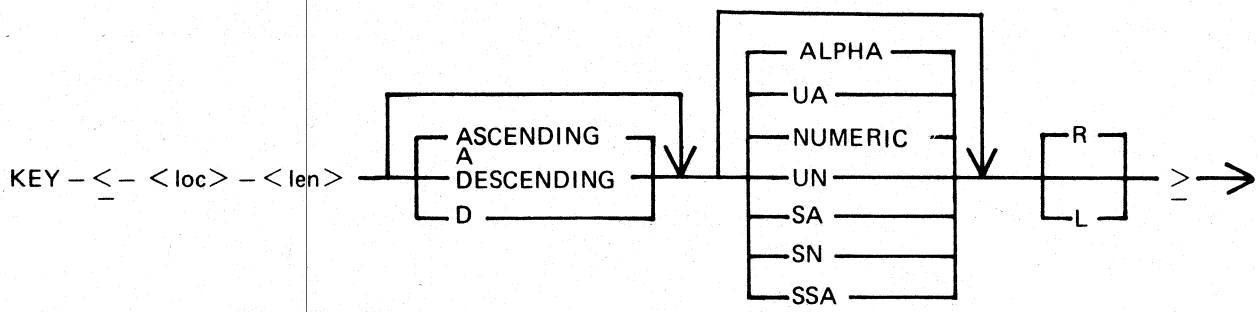
< I-file-spec > is defined as :



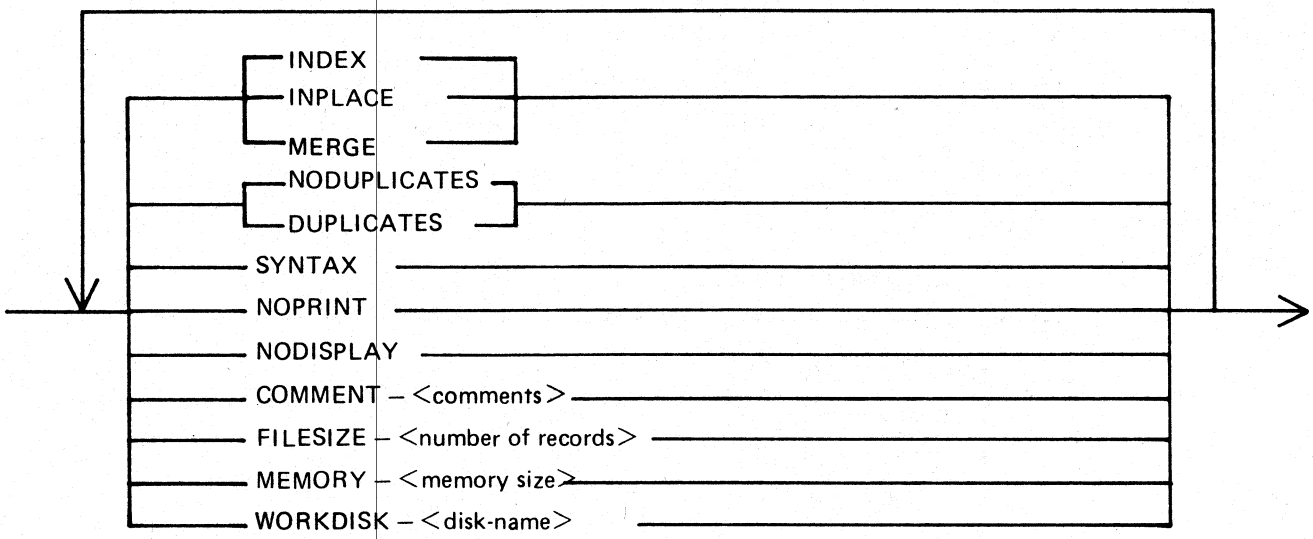
<O-file-spec> is defined as :



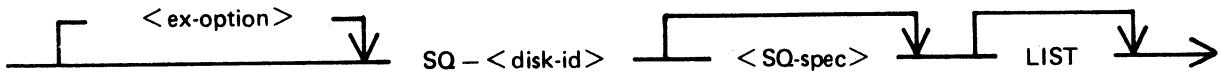
<key-statement> is defined as :



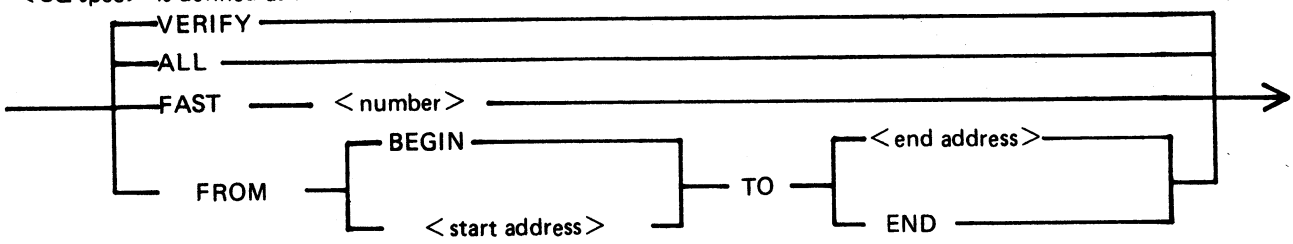
<user-option-statement> is defined as :



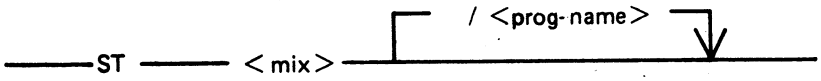
SQ utility



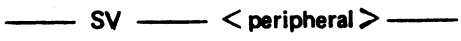
<SQ-spec> is defined as :



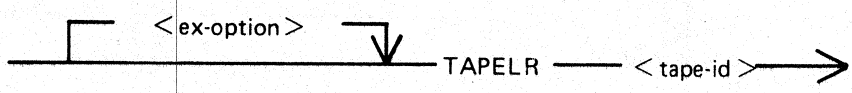
ST intrinsic



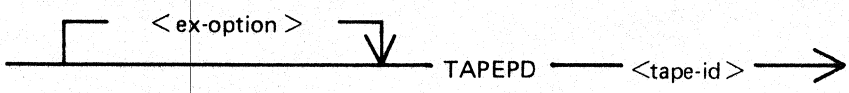
SV intrinsic



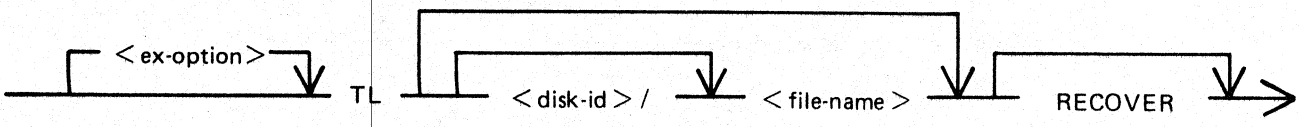
TAPELR utility



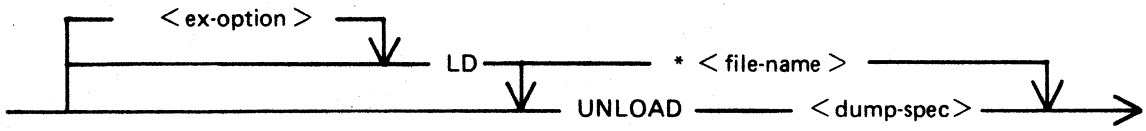
TAPEPD utility



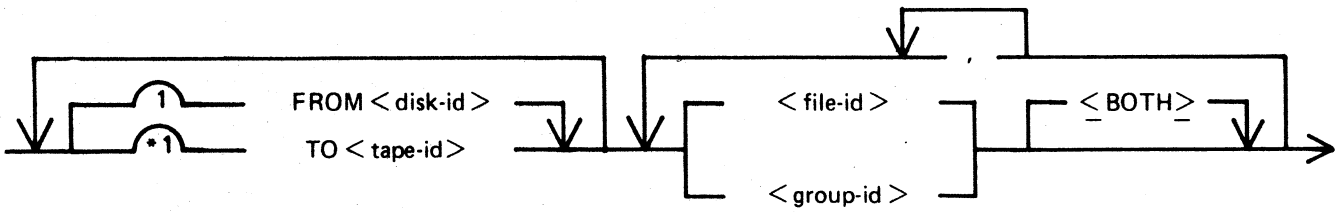
TL utility



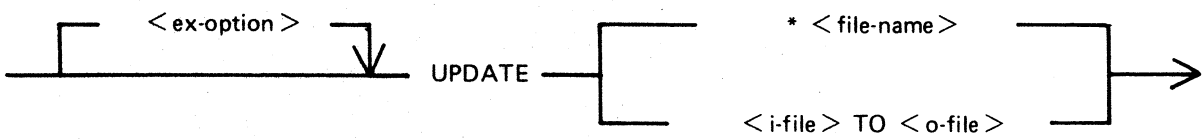
UNLOAD utility



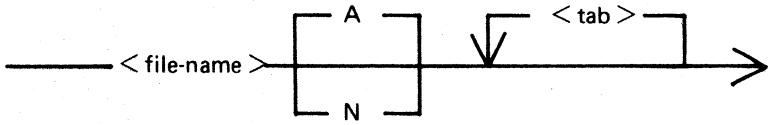
<dump-spec> is defined as :



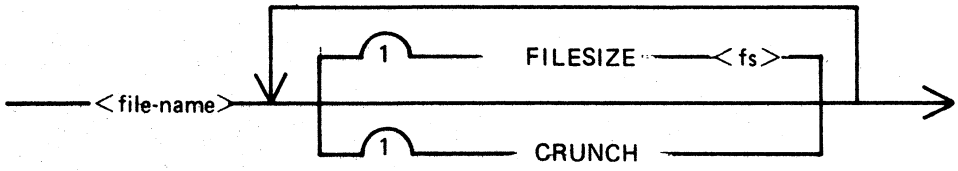
UPDATE utility



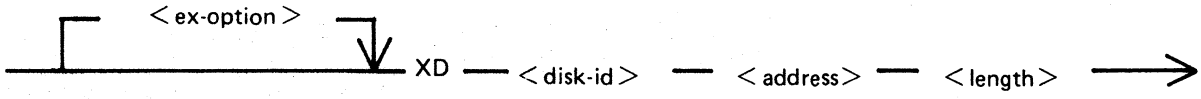
<i-file> is defined as :



<o-file> is defined as :



XD utility



APPENDIX B
EXAMPLES OF PRINTED UTILITY OUTPUT

This appendix provides sample output from some of the CMS–common utilities described in section 4.

SPO and console input messages are underlined. Some utilities use SPO display messages for output. Output print listings use a printer if one is available, or (for a B 80 or B 800) a console file. Print files can be either labelled or unlabelled : if a file is labelled, the name is printed following

?DATA

at the beginning of the listing, and

?END

at the end of the listing (for example, refer to the PL output listing). In this appendix, print files are shown boxed in : other output is on the SPO.

The meaning of the input messages are given in section 4. The utilities are given here in alphabetical order.

CHECKADUMP ARTAPE

```
      NO DISCREPANCIES BETWEEN DUMP TAPE ART
...APE  AND DISK SYSTEM
```

CHECK.DISK MYDISK

```
   10/CHECK.DISK <  2>MYDISK/SYSMEM  DM READ
... PARITY ERROR WHILE IN READ 227F
   ERROR NOTIFIED ON READING SECTOR     5489
...
   10/CHECK.DISK <  2>MYDISK/SYSMEM  DM READ
... PARITY ERROR WHILE IN READ 227F
   ERROR NOTIFIED ON READING SECTOR     5552
...
   10/CHECK.DISK <  2>MYDISK/SYSMEM  DM READ
... PARITY ERROR WHILE IN READ 227F
   ERROR NOTIFIED ON READING SECTOR     5553
...
   CHECK.DISK ON MYDISK  COMPLETED - ERRORS
...NOTIFIED
```

COMPARE FRED WITH MYDISK/FRED

FRED WITH MYDISK/FRED COMPARED - 1 ERRORS

```
DIFFERENCE DETECTED AT BYTE @002F@

      FRED RECORD 2 IS:
@0000@ 5448 4953 2049 5320 4120 4455 4D4D 5920 4649 4C45 2043 5245 4154 4544 2054 4F20"THIS IS A DUMMY FILE CREATED TO "
@0020@ 4445 4D4F 4E53 5452 4154 4520 484F 5778 2020 2020 2020 2020 2020 2020 2020 2020"DEMONSTRATE HOWx               "
@0040@ 2020 2020 2020 2020 2020 2020 2020 2020                                          "                "

      MYDISK/FRED RECORD 2 IS:
@0000@ 5448 4953 2049 5320 4120 4455 4D4D 5920 4649 4C45 2043 5245 4154 4544 2054 4F20"THIS IS A DUMMY FILE CREATED TO "
@0020@ 4445 4D4F 4E53 5452 4154 4520 484F 5720 2020 2020 2020 2020 2020 2020 2020 2020"DEMONSTRATE HOW                "
@0040@ 2020 2020 2020 2020 2020 2020 2020 2020                                          "                "
```

DA MYDISK
BOJ DA VERSION [3.01.01]

?DATA CONS

£

DCL

CARTRIDGE IDENTIFIER	MYDISK	SERIAL NUMBER	000009
OWNERS IDENTIFICATION	TIO		
INITIALIZATION DATE	79150	INITIALIZATION SYSTEM	BDS
PACK CODE	0	ACCESS CODE	
RESTRICTED CARTRIDGE	NO	INTEGRITY FLAG	0
BAD SECTOR COUNT	000000	ACTUAL ERROR COUNT	000000
NUMBER OF CYLINDERS	88/@0058@	UNIT OF ALLOCATION (SECTORS)	1/@01@
NUMBER OF TRACKS/CYLINDER	2/@02@	NUMBER OF SECTORS/TRACK	32/@20@
NAME LIST ADDRESS	38/@0026@	NAME LIST LENGTH	5/@05@
AVAILABLE TABLE ADDRESS	32/@0020@	AVAILABLE TABLE LENGTH	6/@06@
ADDRESS OF FIRST DFH	43/@002B@	MAXIMUM NUMBER OF FILES	51/@0033@

£

DA (continued) :

DFH OF FRED

```
SECTOR:    44/@002C@   FILE IDENTIFIER     FRED            FILE TYPE         @01@ = SOURCE LANGUAGE

                       CREATION DATE       79150           LAST ACCESS DATE            79150
                       GENERATION NUMBER     0/@0000@      IMPLEMENTATION LEVEL NUMBER     0/@00@
                       FLAGS               BIT 0 = 0                 BIT 2 = 0

                       RECORD SIZE          80/@0050@
                       RECORDS/BLOCK         1/@0001@     SECTORS/BLOCK                   1/@0001@
                       MAX FILE SIZE        54/@0036@     SAVE FACTOR  -      (???)

                       MAX AREAS IN USE      1/@01@       OVERFLOW PACK-ID          ???????
                       REC IN LAST AREA      4/@0004@     SPARE BYTES IN LAST RECORD      0/@0000@

                       USER COUNTS: TOTAL USERS  -      (0)  BITS 0-2
                                    OUTPUT USERS -       (0)  BIT  4
                                    LOCK ACCESS USERS -  (0)  BITS 5-7

                       AREA   BIT MAP                  AREA START      AREA SIZE      NUMBER OF RECORDS
                               1 2                       ADDRESS                          IN AREA

                        1      1 0 = ALLOCATED HERE     94/@005E@       54/@0036@          4/@0004@
                       2-16    0 0 = NOT ALLOCATED
```

£

DA (continued) :

```
    DFH NEXT
SECTOR:    45/@002D@   FILE IDENTIFIER     AMEND            FILE TYPE         @10@ = S-CODE

                  CREATION DATE       78107                LAST ACCESS DATE            79150
                  GENERATION NUMBER       0/@0000@         IMPLEMENTATION LEVEL NUMBER     0/@00@
                  FLAGS               BIT 0 = 0                    BIT 2 = 0

                  RECORD SIZE           180/@00B4@
                  RECORDS/BLOCK           1/@0001@         SECTORS/BLOCK                   1/@0001@
                  MAX FILE SIZE          36/@0024@         SAVE FACTOR  -      (???)

                  MAX AREAS IN USE        1/@01@           OVERFLOW PACK-ID          ???????
                  REC IN LAST AREA       36/@0024@         SPARE BYTES IN LAST RECORD      0/@0000@

                  USER COUNTS: TOTAL USERS  -       (0)  BITS 0-2
                               OUTPUT USERS -       (0)  BIT  4
                               LOCK ACCESS USERS - (0)  BITS 5-7

                  AREA   BIT MAP                    AREA START      AREA SIZE      NUMBER OF RECORDS
                           1 2                        ADDRESS                          IN AREA

                    1      1 0 = ALLOCATED HERE      148/@0094@       36/@0024@          36/@0024@
                  2-16     0 0 = NOT ALLOCATED

£
    AVAIL.TABLE AVAILABLE
       AVAILABLE TABLE ADDRESS:    32/@0020@      LENGTH:     6/@0006@

            SECTOR          STATUS      LENGTH        START        END (+1)
             32/@0020@   1  AVAIL     1038/@040E@    198/@00C6@   1236/@04D4@
                         2  AVAIL      213/@00D5@   1913/@0779@   2126/@084E@
                         3  AVAIL      259/@0103@   1336/@0538@   1595/@063B@
                         4  AVAIL      354/@0162@   2216/@08A8@   2570/@0A0A@
             37/@0025@  29  AVAIL     3016/@0BC8@   2616/@0A38@   5632/@1600@
£
```

DA (continued) :

```
NAME.LIST NAMES
    NAME LIST ADDRESS:    38/@0026@    LENGTH:     5/@0005@

            SECTOR          STATUS      INDEX    DFH ADDRESS
            38/@0026@   1   SYSMEM       00       43/@002B@
                        2   FRED        01       44/@002C@
                        3   AMEND       02       45/@002D@
                        4   XD          03       46/@002E@
            39/@0027@   4   COBOL4      0E       57/@0039@
                        8   COBOL1      12       61/@003D@
                        9   COBOL5      13       62/@003E@
                       10   COBOL3      14       63/@003F@
            40/@0028@   1   COBOL7      16       65/@0041@
                        6   COBOL       1B       70/@0046@
£
READ @040E
                   CHARACTER: 0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0 1  2 3  4 5  6 7  8 9  A B  C D  E F
SECTOR:1038/@040E@   0/@000@: 0251 008C 7150 4B51 008D 8A4B 3738 5100 8D51 008C 4B37 2F84 7189 5F84 718B 6F0C
                               Q         P  K Q        K  7 8  Q      Q       K 7  /                 _
                  32/@020@: 2E40 0CB4 0D2E 400C B40D 9188 0C0D 0C51 008D 6102 D04F 0C42 0251 008C 994C 0C77
                            . @        .  @                    Q             0    B    Q        L
                  64/@040@: 0551 008D 8A4B 84F2 5F84 F36F 0C84 F65F 84F7 6F0C 0D84 FA5F 84FB 6F0C 0D77 7484
                               Q         K
                  96/@060@: F65F 84F7 6F0C 771D 5100 8C71 501F 7763 AC77 0671 56E2 0802 0251 008C 5100 8D71
                                                Q         P ?                V           Q       Q
                 128/@080@: 505B 4B37 4E84 FA5F 84FB 6F0C 771D 5100 8C71 601F 773D AC77 0671 57E2 0802 0251
                            P [  K 7  N                      Q          ?    =            W           Q
                 160/@0A0@: 008C 5100 8D71 605B 4B37 2851 008C 7150 1F77 2051
                                 Q          [  K 7  ( Q        P  ?     Q
£
```

DA (continued) :

```
READ NEXT
                    CHARACTER: 0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0 1  2 3  4 5  6 7  8 9  A B  C D  E F
SECTOR:1039/@040F@   0/@000@: 008C 7160 6F77 0871 62E2 0802 0237 1071 3CE2 0802 0251 008C 5100 8D71 605B 4818
                                                          7       <           Q       Q           [  K
                    32/@020@: 2E40 0CB4 0D2E 400C 840D 9188 0C0D 0C51 008D 6102 D04F 0C42 0251 008C 994C 0C77
                              . @        .  @                    Q             O    B    Q        L
                    64/@040@: 0551 008D 8A4B 84F2 5F84 F36F 0C84 F65F 84F7 6F0C 0D84 FA5F 84FB 6F0C 0D77 7484
                               Q        K
                    96/@060@: F65F 84F7 6F0C 771D 5100 8C71 501F 7763 AC77 0671 56E2 0802 0251 008C 5100 8D71
                                                 Q        P ?                V          Q       Q
                   128/@080@: 505B 4837 4E84 FA5F 84FB 6F0C 771D 5100 8C71 601F 773D AC77 0671 57E2 0802 0251
                              P [  K 7  N                         Q        ` ?   =           W          Q
                   160/@0A0@: 008C 5100 8D71 605B 4837 2851 008C 7150 1F77 2051
                                   Q          [  K 7  ( Q        P  ?     Q
£
END

EOJ DA
```

KA MYDISK/ DSKAUL

```
WED 30 MAY 79                    USAGE MAP OF MINI DISK MYDISK   SERIAL NO. 000009  OWNER TIO              PAGE    1.
 INITIALISED 79150 ON    BDS FOR MAXIMUM OF  51 FILES   ALLOCATION UNIT   1  ERROR COUNT 000000  BAD SECTOR COUNT 000000
    88 CYLINDERS    2 TRACKS PER CYLINDER   32 SECTORS PER TRACK
     AREA ADDRESS       AREA LENGTH    STATUS        FILE NAME
     ------------       -----------    ------        ---------

       94 @00005E@     1142 @000476@  AVAILABLE
     1336 @000538@       56 @000038@  AVAILABLE
     1522 @0005F2@       73 @000049@  AVAILABLE
     2216 @0008A8@      354 @000162@  AVAILABLE
     2620 @000A3C@     3012 @000BC4@  AVAILABLE
```

KA MYDISK/=

```
WED 30 MAY 79                    USAGE MAP OF MINI DISK MYDISK   SERIAL NO. 000009  OWNER TIO              PAGE    1.
 INITIALISED 79150 ON    BDS FOR MAXIMUM OF  51 FILES   ALLOCATION UNIT   1  ERROR COUNT 000000  BAD SECTOR COUNT 000000
    88 CYLINDERS    2 TRACKS PER CYLINDER   32 SECTORS PER TRACK
     AREA ADDRESS       AREA LENGTH    STATUS        FILE NAME
     ------------       -----------    ------        ---------

        0 @000000@       32 @000020@  ASSIGNED      *RESERVED
       32 @000020@        6 @000006@  ASSIGNED      *AVAIL.TABLE
       38 @000026@        5 @000005@  ASSIGNED      *FILE DIREC.
       43 @00002B@       51 @000033@  ASSIGNED      *FILE HEADRS
       94 @00005E@     1142 @000476@  AVAILABLE
     1236 @0004D4@      100 @000064@  ASSIGNED      COBOL4
     1336 @000538@       56 @000038@  AVAILABLE
     1392 @000570@      130 @000082@  ASSIGNED      COBOL2
     1522 @0005F2@       73 @000049@  AVAILABLE
     1595 @00063B@      128 @000080@  ASSIGNED      COBOL1
     1723 @0006BB@       91 @00005B@  ASSIGNED      COBOL5
     1814 @000716@       99 @000063@  ASSIGNED      COBOL3
     1913 @000779@      213 @0000D5@  ASSIGNED      COBOL6
     2126 @00084E@       90 @00005A@  ASSIGNED      COBOL7
     2216 @0008A8@      354 @000162@  AVAILABLE
     2570 @000A0A@       46 @00002E@  ASSIGNED      COBOL
     2616 @000A38@        4 @000004@  ASSIGNED      FRED
     2620 @000A3C@     3012 @000BC4@  AVAILABLE.
```

LIST MYDISK/FRED

```
REC.NO. FILE MYDISK/FRED ON DSK                REC/BLK SIZES    80/   80      WED 30 MAY 79                PAGE     1.

      1  HELLO READER                          48454C4C4F20524541444552202020202020202020202020202020202020202020
                                               20202020202020202020202020202020202020202020202020202020202020202020
                                               20202020202020202020202020202020
      2  THIS IS A DUMMY FILE CREATED TO       54484953204953204120445 54D4D592046494C4520435245415445442054 4F20
         DEMONSTRATE HOW                       44454D4F4E53545241544520484F572020202020202020202020202020202020202020
                                               20202020202020202020202020202020
      3  SOME OF THE CMS UTILITIES WORK.       534F4D45204F462054484520434D53205554494C495449455320574F524B2E20
                                               20202020202020202020202020202020202020202020202020202020202020202020
                                               20202020202020202020202020202020
      4              THANKYOU.                 20202020202020202020202020544841 4E4B594F552E202020202020202020202020
                                               20202020202020202020202020202020202020202020202020202020202020202020
                                               20202020202020202020202020202020
```

LIST MYDISK/FRED A

```
REC.NO. FILE MYDISK/FRED ON DSK                REC/BLK SIZES    80/   80      WED 30 MAY 79                PAGE     1.

      1  HELLO READER
      2  THIS IS A DUMMY FILE CREATED TO DEMONSTRATE HOW
      3  SOME OF THE CMS UTILITIES WORK.
      4              THANKYOU.
```

LR MYDISK/= <A>

```
WED 30 MAY 79                    DIRECTORY OF MINI DISK MYDISK   SERIAL NO. 000009  OWNER TIO              PAGE    1.
 INITIALISED 79150 ON    BDS FOR MAXIMUM OF  51 FILES   ALLOCATION UNIT   1  ERROR COUNT 000000  BAD SECTOR COUNT 000000
    88 CYLINDERS    2 TRACKS PER CYLINDER   32 SECTORS PER TRACK
 FILE NAME   ACTUAL  MAXIMUM  RECORD  RECS/  CREATED        FILE      NO.          AREA              AREA      OVERFLOW
             SIZE    SIZE     SIZE    BLOCK      ACCESSED   TYPE     AREAS       ADDRESSES          SIZES        DISK

*RESERVED      32      32      180      32  79150  79150   SYSTEM      1         0 @000000@       32 @000020@
*AVAIL.TABLE    6       6      180      32  79150  79150   SYSTEM      1        32 @000020@        6 @000006@
*FILE DIREC.    5       5      180      32  79150  79150   SYSTEM      1        38 @000026@        5 @000005@
*FILE HEADRS   51      51      180      32  79150  79150   SYSTEM      1        43 @00002B@       51 @000033@
AMEND          36      36      180       1  78107  79150  CODE 780226  1       148 @000094@       36 @000024@
COBOL          46      46      180       1  79139  79150  CODE 790519  1      2570 @000A0A@       46 @00002E@
COBOL1        128     128      180       1  78339  79150  CODE 780915  1      1595 @00063B@      128 @000080@
COBOL3         99      99      180       1  78339  79150  CODE 780525  1      1814 @000716@       99 @000063@
COBOL4        100     100      180       1  78339  79150  CODE 780822  1      1236 @0004D4@      100 @000064@
COBOL5         91      91      180       1  78339  79150  CODE 780830  1      1723 @0006BB@       91 @00005B@
COBOL7         90      90      180       1  78339  79150  CODE 780915  1      2126 @00084E@       90 @00005A@
FRED            4      54       80       1  79150  79150  SRCELANG     1        94 @00005E@       54 @000036@
XD             14      14      180       1  78107  79150  CODE 780226  1       184 @0000B8@       14 @00000E@
```

LR MYDISK/= 100

```
WED 30 MAY 79                    DIRECTORY OF MINI DISK MYDISK   SERIAL NO. 000009  OWNER TIO              PAGE    1.
 INITIALISED 79150 ON    BDS FOR MAXIMUM OF  51 FILES   ALLOCATION UNIT   1  ERROR COUNT 000000  BAD SECTOR COUNT 000000
   88 CYLINDERS    2 TRACKS PER CYLINDER   32 SECTORS PER TRACK
 FILE NAME   ACTUAL  MAXIMUM  RECORD  RECS/  CREATED        FILE     NO.        AREA              AREA      OVERFLOW
              SIZE    SIZE    SIZE   BLOCK      ACCESSED   TYPE    AREAS      ADDRESSES          SIZES        DISK

CDBOL1          128     128     180       1  78339  79150  CODE 780915   1      1595 @00063B@      128 @000080@


        AREA            AREA                AREA            AREA                AREA            AREA
      ADDRESSES         SIZES             ADDRESSES         SIZES             ADDRESSES         SIZES

AVAILABLE AREAS
----------------
      198 @0000C6@     1038 @00040E@       1913 @000779@     213 @0000D5@       1336 @000538@     259 @0001030@
     2216 @0008A8@      354 @000162@       2616 @000A38@    3016 @000BC8@

             TOTAL AVAILABLE SPACE ON DISK            4880 @001310@


        AREA            AREA                AREA            AREA                AREA            AREA
      ADDRESSES         SIZES             ADDRESSES         SIZES             ADDRESSES         SIZES

TEMPORARY AREAS
----------------

       TOTAL SPACE ON DISK IN TEMPORARY USE               0 @000000@
```

```
MODIFY
?DATA CON
CMS UTILITY: MODIFY [VERSION 3.01.01]


USE PK1 FOR HELP
CODE.FILE? (PK1 depressed)
PK1=HELP   PK2=MODIFY PPB   PK3=MODIFY FPB   PK4=CODE.FILE   PK5=PRINT FPB/PPB   PK6=TERMINATE
CODE.FILE?          MYDISK/COBOL1
SELECT FUNCTION     (PK2 depressed)
PPB ATTRIBUTE       EOJ.SUPPRESS            0 (OFF)         NEW VALUE           ON
PPB ATTRIBUTE       (PK5 depressed)
  PPB OF CODE.FILE MYDISK /COBOL1

IMP.LEVEL.NO                          0
PROGRAM NAME             "COBOL1      "
S-LANGUAGE               "BIL.REV.10  "
INTERP.PACK                   "0000000"
INTERP.NAME              "BILINTERP   "
COMPILER NAME            "BIL 3.0.2   "
COMPILE DATE                  "780915"
EOJ.SUPPRESS                          1
CLASS                                 4 (A)
INIT.MESS                          @FF@
ENTRY POINT                           0
PST.LENGTH                           66
PST.LOCATION                          2
DST.LENGTH                          180
DST.LOCATION                          3
TCB.PA LENGTH                        88
TCB.PA LOCATION                      92
STACK LENGTH                        250
CCB.PA LENGTH                         0
CCB.PA LOCATION                       0
TCB.PE LENGTH                         0
IFNB LENGTH                         300
IFNB LOCATION                         4
PPB ATTRIBUTE    (PK6 depressed)
...CHEERIO
?END  CON
```

```
MODIFY CODE.FILE MYDISK/COBOL1,PPB,EOJ.SUPPRESS OFF,PRINT.PPB,END

   * MODIFICATIONS SUCCESSFUL *

?DATA LP

CMS UTILITY: MODIFY [VERSION 3.01.01]


CODE.FILE MYDISK/COBOL1,PPB,EOJ.SUPPRESS OFF,PRINT.PPB,END

   PPB OF CODE.FILE MYDISK /COBOL1

 IMP.LEVEL.NO                          0
 PROGRAM NAME              "COBOL1      "
 S-LANGUAGE                "BIL.REV.10  "
 INTERP.PACK                    "0000000"
 INTERP.NAME               "BILINTERP   "
 COMPILER NAME             "BIL 3.0.2   "
 COMPILE DATE                    "780915"
 EOJ.SUPPRESS                          0
 CLASS                                 4 (A)
 INIT.MESS                          @FF@
 ENTRY POINT                           0
 PST.LENGTH                           66
 PST.LOCATION                          2
 DST.LENGTH                          180
 DST.LOCATION                          3
 TCB.PA LENGTH                        88
 TCB.PA LOCATION                      92
 STACK LENGTH                        250
 CCB.PA LENGTH                         0
 CCB.PA LOCATION                       0
 TCB.PE LENGTH                         0
 IFNB LENGTH                         300
 IFNB LOCATION                         4

 ?END  LP
```

PL SYS-LOG-HOLD

```
?DATA LINES

WED  6 JUN 79                    CMS LOG FILE PRINTOUT OF  SYS-LOG-HOLD                    PAGE   1

  TIME      DATE      MESSAGE   MIX      I/O      ENTRY    RECORD          MESSAGE     TEXT
HH:MM:SS   MM/DD/YY   TYPE    NUMBER   MESSAGE   NUMBER   NUMBER

           06/06/79   SYSTEM    15      INPUT       1        1      COMPARE FRED WITH MYDISK/FRED
           06/06/79   SYSTEM    10      OUTPUT      2        1      END OF FILE FRED BEFORE MYDISK/F
                                                                    RED - 0 ERRORS
           06/06/79   SYSTEM    15      INPUT       3        1      COPY MYDISK/FRED TO FRED EXTENDI
                                                                    NG
           06/06/79   SYSTEM    10      OUTPUT      4        1      MYDISK/FRED TO FRED BAD ATTRIBUT
                                                                    ES
           06/06/79   SYSTEM    15      INPUT       5        1      COPY FRED TO FRED FILESIZE 10
           06/06/79   SYSTEM    10      OUTPUT      6        1      FRED REMOVED
           06/06/79   SYSTEM    10      OUTPUT      7        1      FRED TO FRED COPIED
           06/06/79   SYSTEM    10      OUTPUT      8        1      NO RECORDS FOR COPYING FROM FRED
           06/06/79   SYSTEM    15      INPUT       9        1      COPY MYDISK/FRED TO FRED EXTENDI
                                                                    NG
           06/06/79   SYSTEM    11      OUTPUT     10        1      MYDISK/FRED TO FRED BAD ATTRIBUT
                                                                    ES
           06/06/79   SYSTEM    15      INPUT      11        1      PO DMB
           06/06/79   SYSTEM    15      OUTPUT     12        1       DMB  O.K.
           06/06/79   SYSTEM     9      OUTPUT     13        1       DMB E/    0 FILES OPEN
           06/06/79   SYSTEM    15      INPUT      14        1      PD E/=
           06/06/79   SYSTEM    12      OUTPUT     15        1      E CONTAINS -
           06/06/79   SYSTEM    12      OUTPUT     16        1      DELINP CMSCANDE DELINP.O
           06/06/79   SYSTEM    12      OUTPUT     17        1      ORD.OLD.S NORD.O FACOAT
           06/06/79   SYSTEM    12      OUTPUT     18        1      NORDO NORDS CUSMAS
           06/06/79   SYSTEM    12      OUTPUT     19        1      END PD

?END  LINES
```

```
SQ MYDISK FAST 2800 LIST
   LARGEST AVAILABLE SPACE IS 3012 SECTORS
   TOTAL AVAILABLE SPACE IS 4637 SECTORS IN
...5 AREA(S)
              *** SQ COMPLETED ***
```

USAGE MAP OF DISK MYDISK BEFORE SQUASH DATE : WED 30 MAY 79 PAGE 1.

FILE NAME	AREA NB	FILE OPEN	DISK ALLOCATION LENGTH	FROM	TO	AVAILABLE TABLE INFORMATION	REMARKS
			1142	@00005E@	@0004D3@	AVAILABLE SPACE	
CDBOL4	1		100	@0004D4@	@000537@		
			56	@000538@	@00056F@	AVAILABLE SPACE	
CDBOL2	1		130	@000570@	@0005F1@		
			73	@0005F2@	@00063A@	AVAILABLE SPACE	
CDBOL1	1		128	@00063B@	@0006BA@		
CDBOL5	1		91	@0006BB@	@000715@		
CDBOL3	1		99	@000716@	@000778@		
CDBOL6	1		213	@000779@	@00084D@		
CDBOL7	1		90	@00084E@	@0008A7@		
			354	@0008A8@	@000A09@	AVAILABLE SPACE	
CDBOL	1		46	@000A0A@	@000A37@		
			1041	@000A38@	@000E48@	AVAILABLE SPACE	
FRED	1		4	@000E49@	@000E4C@		
			1971	@000E4D@	@0015FF@	AVAILABLE SPACE	

SQ (continued) :

```
USAGE MAP OF DISK MYDISK  AFTER  SQUASH                    DATE : WED 30 MAY 79        PAGE  1.

 FILE NAME   AREA FILE        DISK ALLOCATION          AVAILABLE TABLE     REMARKS
             NB  OPEN    LENGTH     FROM       TO        INFORMATION
==================================================================================================

                         1142    @00005E@   @0004D3@   AVAILABLE SPACE
COBOL4        1           100    @0004D4@   @000537@
                           56    @000538@   @00056F@   AVAILABLE SPACE
COBOL2        1           130    @000570@   @0005F1@
                           73    @0005F2@   @00063A@   AVAILABLE SPACE
COBOL1        1           128    @00063B@   @0006BA@
COBOL5        1            91    @0006BB@   @000715@
COBOL3        1            99    @000716@   @000778@
COBOL6        1           213    @000779@   @00084D@
COBOL7        1            90    @00084E@   @0008A7@
                          354    @0008A8@   @000A09@   AVAILABLE SPACE
COBOL         1            46    @000A0A@   @000A37@
FRED          1             4    @000A38@   @000A3B@
                         3012    @000A3C@   @0015FF@   AVAILABLE SPACE
```

TAPELR ARTAPE

```
THU 21 JUN 79                          TAPE SERIAL NUMBER 00000    PAGE    1.
DIRECTORY OF NRZI TAPE ARTAPE     DUMPED ON THU 21 JUN 79
FILE NAME    ACTUAL   MAXIMUM   RECORD   RECS/   CREATED            FILE
              SIZE      SIZE     SIZE    BLOCK        ACCESSED      TYPE

MYFILE            4         5      180        1   79172   79172     KEY
MYFILEQQ          6        10       50        1   79172   79172     DATA
A999             90        90      128        1   79158   79172     DATA
```

TAPEPD ARTAPE

```
   NRZI TAPE ARTAPE <00000> DUMPED ON THU 21
... JUN 79 CONTAINS -
   MYFILE MYFILEQQ A999
   END TAPEPD
```

APPENDIX C
GLOSSARY OF TECHNICAL TERMS

ADDRESS

A disk is divided physically into tracks and sectors, both numbered sequentially from zero upwards. These 'numbers' are referred to as 'addresses'. The MCP uses this address scheme to quickly locate data on disk.

ALPHANUMERIC

Consisting only of letters of the alphabet plus the ten numeric digits; that is, not containing any other special characters.

APPLICATION PROGRAM

User program that performs day-to-day functions such as invoicing, printing, inventory reports, etc.

ATTRIBUTE

Characteristic or quality.

BACK-UP

Term used to describe the method of insuring that copies of files exist to standby as alternatives.

BINARY-CODED DECIMAL (BCD)

A method of coding numeric information in 4-bit units representing 0 as bits 0000, 1 as bits 0001, 2 as bits 0010, up to 9 as bits 1001. For example, the number 1607 in BCD would take four 4-bit units (2 bytes), coded as 0001 0110 0000 0111.

BOJ

'Beginning of Job' The term used to notify the operator that a program has entered the 'mix' and has just started running.

BSMD

Abbreviation for 'Burroughs Super Mini Disk'.

BYTE

One alphanumeric character of data.

CHECKERBOARDED

Term applied to any disk having available spaces of varying sizes scattered about the disk amongst files. The term can also be applied to memory in a virtual memory system where 'locked' or 'save' areas are scattered through the memory in such a way as to impede getting overlayable memory areas of sufficient size for optimum throughput.

CMS

Computer Management System. A set of interrelated specifications for system software, including high-level language compilers, object-code formats, operator interface and data communications, which Burroughs has implemented on machines of different hardware characteristics.

COMPILATION DATE

The date on which a programmer's source code was compiled : that is, the creation date of the executable object program.

COMPILERS

Group of system programs that convert instructions written by a programmer in a language such as COBOL or RPG into a form which can be run or interpreted by the hardware or system software.

CONFIGURATION

Term used to describe the arrangement of various hardware devices in a particular system.

DATA FILE

A set of information usually or a disk, which is used as data to be input.

DEFAULT VALUE

Usually a meaning that a program will assume if not instructed otherwise.

DESTINATION

Disk to which information is being transferred.

DISK DIRECTORY

List, on Track 0, of file names, locations on disk, and sizes. Similar to a table of contents.

DISK FILE

Set of information residing on a disk medium, collectively referred to by its name, 'file-name' and the name of the disk on which it resides ('disk-name').

DISK NAME

Name by which a disk is known to MCP. Every disk medium has a 'label' of information written to it during disk initialization, and the disk name is part of the 'label'.

DUAL-PACK FILE (MULTI-VOLUME FILE)

A file that resides on two separate disks or logically defined disks (for example, DKA, DKB).

EOJ

'End of Job'. The term used to notify the operator that a program has terminated. 'Abnormal' end-of-job occurs when a program is terminated prematurely due to an error condition.

EXECUTION

The running of a program is termed 'program execution'. The operator can execute (or start) a program by entering the name of the program desired (or disk-name/program if program resides on user disk). When a program is 'executed', it enters the 'mix' and is assigned a 'mix number' by the MCP.

FAMILY (GROUP) OF FILES

Two or more disk files having at least the first letter of their names in common. For example, 'PR020', 'PRFILE', and 'PASM1' are members of a family of files that could be referred to as 'P—'.

HARDWARE

Term referring to all equipment on the system. Line printers and disk cabinets are examples.

HEXADECIMAL ('HEX')

A number system based on root 16, in contrast to common 'decimal' system based on root 10. To provide additional symbols, the letters A through F are used, so that counting procedes thus: 0, 1, 2, 3, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11, 12 ... for decimal numbers 0 through 18 ...

INTERPRETER

A system software item used as an intermediate step in the running of a program. Instead of using a compiler to transform programs directly to machine instructions which can be run by the processor, a compiler may transform the program to an intermediate form (called 'S-code'). The S-code can be 'interpreted' by the interpreter, that is, translated into machine instructions that can be run by the processor.

INTRINSIC

A 'command' used by the operator to direct the activities of the MCP. Intrinsics are actually a part of the MCP and therefore will never be seen on a disk file listing or in the 'mix'. Examples of intrinsics include 'DS' (discontinue the processing of a program) and 'DT' (retrieve or change system date).

KEYFILE

File used by system as an index to a master data file.

LABEL

A small space of disk on tape indicating the medium's contents, name, etc. A disk label may be created during the initialization process, and a tape label is created when the tape is purged.

MAIN MEMORY

Circuit boards inside processor where program code and data in immediate use are held.

MCP ("Master Control Program")

Program which is the central part of the CMS software system. It handles hardware devices, communicates with the operator, and controls processing of programs.

MIX

Term applied to the mixture of programs running in a multi-programming environment. A 'mix-number' is a number which is assigned by the MCP to a program when it enters the 'mix'. A 'null' mix is when no jobs are running.

The program's name and mix number can be used by the operator to refer to a particular program in the 'mix'.

MULTIPROGRAMMING

One processor working on more than one program at a time. Processing can be shared on a 'round-robbin' basis, and computation can be overlapped with input/output if there is more than one program 'in the mix'.

ON-LINE

Term used for equipment or media currently used as part of the system.

PACK

Synonym for 'disk'

PERIPHERAL

Hardware device used as input or output. Examples are line printer, disk drive unit, console keyboard.

PURGE

To erase when disks or tapes are 'purged', their contents are lost.

SECTOR

A disk is divided physically into data storage spaces called sectors, numbered sequentially from zero upwards. Each sector is 180 characters in length.

SOFTWARE

Term referring to programs and files, as distinct from the 'hardware' of the actual machine.

SOURCE DISK

Disk from which information is being transferred.

SOURCE FILE

A disk file containing statements (instructions) written by a programmer in a high-level language such as COBOL or RPG, before it has been transformed into a runnable program.

STAND-ALONE PROGRAMS

Programs that do not run under control of the MCP. In particular, functions of general use to all B80 users are held in a disk file called 'SAU'(Stand Alone Utilities). Examples include LS (list disk name and sizes), and RL (relabel a disk). Loading and execution of SAU is done with no need of the MCP. Refer to Section 8 for details.

STARFILE

A small disk file optionally used at the start of most CMS-common utilities. The information in the starfile is used to build up the initiating message for the utility, which could also be entered by the operator on the SPO. Starfiles are also called 'macro-files'.

SYSTEM DISK

The disk containing the copy of the MCP that is currently in use.

Note that a user disk may also contain MCP code files, but only the disk containing the MCP that is in use since the last warmstart is the system disk. There can be only one system disk at any time during operation. System disks cannot be used as system disks on more than one CMS product (see section 2 for details).

SYSTEM FILE

A disk file which is used by the system software. Special control is placed on these files to minimise the danger of accidental removal from the disk (see RM utility).

SYSTEM SOFTWARE UTILITY

A program of general use to all users, as opposed to an application program which performs a particular using day-to-day tasks, such as invoicing. Examples of utilities include COPY (copy files from one medium to another) and RM (remove files from a disk).

USER DISK

Any disk available to the system that is not a system disk.

VIRTUAL MEMORY

A software technique, implemented in the MCP, of allowing programs to execute (or several to execute together) when the total program memory requirements exceeds the amount of memory physically available. Some of the executing program's code and data, which is not in immediate use, is stored on disk media and not in main memory. When the code, or data, is required, space is made for it in main memory and the information read back from disk. To make space in memory, it may be necessary for the MCP to re-use some memory which has previously been used by the program and is not required at this moment. Before re-using memory containing data that could have been updated, the MCP writes this segment of memory to the program's 'virtual memory file' on disk.

This technique also applies to the code and data of the MCP and other system sofware.

VOLUME

Synonym for 'disk'.

WRITE INHIBIT

To prevent disk on tape media from being written to by a program. The manner in which this is accomplished depends upon the medium (see B80 or B800 System Operator manuals for details).

WRITE PERMIT

To allow any disk or tape medium to be written to by a program. The manner in which this is accomplished depends upon the medium (see B 80 or B 800 System Operator manuals for details).

APPENDIX D
RELATED DOCUMENTATION

The following manuals provide information concerning CMS System Software:

Manual	Form Number
CMS ARCS Reference Manual	2012713
CMS COBOL Reference Manual	2007266
CMS MCP Reference Manual	2007266
CMS RPG Reference Manual	2007274
CMS MPLII Reference Manual	2007563
CMS NDL Reference Manual	1090925
CMS Data Communications Subsystem Reference Manual	1090909