

DIGITAL
RESEARCH TM

CP/M
Operating System

Manual

THIS IS THE TRUNCATED VERSION -- SHORTENED FOR BREVITY, SUITABLE FOR PRINTING. The full User's Manual is over 300 pages.

This truncated version covers the CCP intrinsic commands and the basic utilities and serves as a handy reference. Refer to the original DRI manuals for the complete story.

The complete manual is available on the web or from www.bleeve.me.

COPYRIGHT

Copyright C 1976, 1977, 1978, 1979, 1982, and 1983 by Digital Research. All rights reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of Digital Research, Post Office Box 579, Pacific Grove, California 93950.

This manual is, however, tutorial in nature. Thus, the reader is granted permission to include the example programs, either in whole or in part, in his own programs.

DISCLAIMER

Digital Research makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Digital Research reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Digital Research to notify any person of such revision or changes.

TRADEMARKS

CP/M and CP/NET are registered trademarks of Digital Research. ASM, DESPOOL, DDT, LINK-80, MAC, MP/M, PL/1-80 and SID are trademarks of Digital Research. Intel is a registered trademark of Intel Corporation. TI Silent 700 is a trademark of Texas Instruments Incorporated. Zilog and Z80 are registered trademarks of Zilog, Inc.

The CP/M Operating System Manual was printed in the United States of America.

First Edition: 1976
Second Edition: July 1982
Third Edition: September 1983

Table of Contents

1 CP/M Features and Facilities

1.1	Introduction	1-1
1.2	Functional .. Description	1-3
1.2.1 General Command Structure	1-3
1.2.2 File References	1-4
1.3	Switching . Disks	1-7
1.4	Built-in ... Commands	1-7
1.4.1 ERA	1-8
1.4.2 DIR	1-9
1.4.3 REN	1-10
1.4.4 SAVE	1-11
1.4.5 TYPE	1-11
1.4.6 USER	1-12
1-12	1.5 Line Editing and Output Control	
1-14	1.6 Transient Commands	
	1.6.1 STAT	I-15
	1.6.2 ASM	1-22
	1.6.3 LOAD	1-24
	1.6.4 PIP	1-25
	1.6.5 ED	1-35
	1.6.6 SYSGEN	1-37
	1.6.7 SUBMIT	1-39
	1.6.8 DUMP	1-41
	1.6.9 MOVCPM	1-42
	1.7 BDOS Error Messages	1-44
	1.8 Operation of CP/M on the MDS	1-46

2 ED

2.1	Introduction to ED	2-1
2.1.1	ED Operation	2-1
2.1.2	Text Transfer Functions	2-3
2.1.3	Mei-norv Buffer Organization	2-4
2.1.4	Line numbers and ED Start Up	2-5
2.1.5	Metnorv Buffer Operation	2-7
2.1.6	Command Strings	2-8

Table of Contents (continued)

6.9	Reserved Locations in Page Zero	6-26
6.10	Disk Parameter Tables	6-28
6.11	The DISKDEF Macro Library	6-34
6.12	Sector Blocking and Deblocking	6-39

Appendixes

A	The MDS Basic I/O System (BIOS)	A-1
B	A Skeletal CBIOS	B-1
C	A Skeletal GETSYS/PUTSYS Program	C-1
D	The MDS-800 Cold Start Loader for CP/M 2	D-1
E	A Skeletal Cold Start Loader	E-1
F	CP/M Disk Definition Library	F-1
G	Blocking and Deblocking Algorithms	G-1
H	Glossary	H-1
I	CP/M Messages	I-1

Tables

1-1	Line-editing Control Characters	1-12
1-2	CP/M Transient Commands	1-14
1-3	Physical Devices	1-17
1-4	PIP Parameters	1-31
2-1	ED Text Transfer Commands	2-3
2-2	Editing Commands	2-8
2-3	Line-editing Controls	2-9
2-4	Error Message Symbols	2-18
2-5	ED Control Characters	2-19
2-6	ED Commands	2-20
3-1	Reserved Characters	3-6
3-2	Arithmetic and Logical Operations	3-7
3-3	Assembler Directives	3-10
3-4	Jumps, Calls, and Returns	3-17
3-5	Immediate Operand Instructions	3-19
3-6	Increment and Decrement Instructions	3-20

Section 1

CP/M Features and Facilities

1.1 Introduction

CP/M is a monitor control program for microcomputer system development that uses floppy disks or Winchester hard disks for backup storage. Using a computer system based on the Intel 8080 microcomputer, CP/M provides an environment for program construction, storage, and editing, along with assembly and program checkout facilities. CP/M can be easily altered to execute with any computer configuration that uses a Zilog Z80 or an Intel 8080 Central Processing Unit (CPU) and has at least 20K bytes of main memory with up to 16 disk drives. A detailed discussion of the modifications required for any particular hardware environment is given in Section 6. Although the standard Digital Research version operates on a single-density Intel MDS 800, several different hardware manufacturers support their own input-output (I/O) drivers for CP/M.

The CP/M monitor provides rapid access to programs through a comprehensive file management package. The file subsystem supports a named file structure, allowing dynamic allocation of file space as well as sequential and random file access. Using this file system, a large number of programs can be stored in both source and machine executable form.

CP/M 2 is a high-performance, single console operating system that uses table-driven techniques to allow field reconfiguration to match a wide variety of disk capacities. All fundamental file restrictions are removed, maintaining upward compatibility from previous versions of release 1.

Features of CP/M 2 include field specification of one to sixteen logical drives, each containing up to eight megabytes. Any particular file can reach the full drive size with the capability of expanding to thirty-two megabytes in future releases. The directory size can be field-configured to contain any reasonable number of entries, and each file is optionally tagged with Read-Only and system attributes. Users of CP/M 2 are physically separated by user numbers, with facilities for file copy operations from one user area to another. Powerful relative-record random access functions are present in CP/M 2 that provide direct access to any of the 65536 records of an eight-megabyte file.

CP/M also supports ED, a powerful context editor, ASM, an Intel-compatible assembler, and DDT, debugger subsystems. Optional software includes a powerful Intel-compatible macro assembler, symbolic debugger, along with various high-level languages. When coupled with CP/M's Console Command Processor (CCP), the resulting facilities equal or exceed similar large computer facilities.

CP/M is logically divided into several distinct parts:

- BIOS (Basic I/O System), hardware-dependent
- BDOS (Basic Disk Operating System)
- CCP (Console Command Processor)
- TPA (Transient Program Area)

The BIOS provides the primitive operations necessary to access the disk drives and to interface standard peripherals: teletype, CRT, paper tape reader/punch, and user-defined peripherals. You can tailor peripherals for any particular hardware environment by patching this portion of CP/M. The BDOS provides disk management by controlling one or more disk drives containing independent file directories. The BDOS implements disk allocation strategies that provide fully dynamic file construction while minimizing head movement across the disk during access. The BDOS has entry points that include the following primitive operations, which the program accesses:

- SEARCH looks for a particular disk file by name.
- OPEN opens a file for further operations.
- CLOSE closes a file after processing.
- RENAME changes the name of a particular file.
- READ reads a record from a particular file.
- WRITE writes a record to a particular file.
- SELECT selects a particular disk drive for further operations.

The CCP provides a symbolic interface between your console and the remainder of the CP/M system. The CCP reads the console device and processes commands, which include listing the file directory, printing the contents of files, and controlling the operation of transient programs, such as assemblers, editors, and debuggers. The standard commands that are available in the CCP are listed in Section 1.2.1.

The last segment of CP/M is the area called the Transient Program Area (TPA). The TPA holds programs that are loaded from the disk under command of the CCP. During program editing, for example, the TPA holds the CP/M text editor machine code and data areas. Similarly, programs created under CP/M can be checked out by loading and executing these programs in the TPA.

Any or all of the CP/M component subsystems can be overlaid by an executing program. That is, once a user's program is loaded into the TPA, the CCP, BDOS, and BIOS areas can be used as the program's data area. A bootstrap loader is programmatically accessible whenever the BIOS portion is not overlaid; thus, the user program need only branch to the bootstrap loader at the end of execution and the complete CP/M monitor is reloaded from disk.

The CP/M operating system is partitioned into distinct modules, including the BIOS portion that defines the hardware environment in which CP/M is executing. Thus, the standard system is easily modified to any nonstandard environment by changing the peripheral drivers to handle the custom system.

1.2 Functional Description

You interact with CP/M primarily through the CCP, which reads and interprets commands entered through the console. In general, the CCP addresses one of several disks that are on-line. The standard system addresses up to sixteen different disk drives. These disk drives are labeled A through P. A disk is logged-in if the CCP is currently addressing the disk. To clearly indicate which disk is the currently logged disk, the CCP always prompts the operator with the disk name followed by the symbol >, indicating that the CCP is ready for another command. Upon initial start-up, the CP/M system is loaded from disk A, and the CCP displays the following message:

```
CP/M VER x.x
```

where x.x is the CP/M version number. All CP/M systems are initially set to operate in a 20K memory space, but can be easily reconfigured to fit any memory size on the host system (see Section 1.6.9). Following system sign-on, CP/M automatically logs in disk A, prompts you with the symbol A>, indicating that CP/M is currently addressing disk A, and waits for a command. The commands are implemented at two levels: built-in commands and transient commands.

1.2.1 General Command Structure

Built-in commands are a part of the CCP program, while transient commands are loaded into the TPA from disk and executed. The following are built-in commands:

- ERA erases specified files.
- DIR lists filenames in the directory.
- REN renames the specified file.
- SAVE saves memory contents in a file.
- TYPE types the contents of a file on the logged disk.

Most of the commands reference a particular file or group of files. The form of a file reference is specified in Section 1.2.2.

1.2.2 File References

A file reference identifies a particular file or group of files on a particular disk attached to CP/M. These file references are either unambiguous (ufn) or ambiguous (afn). An unambiguous file reference uniquely identifies a single file, while an ambiguous file reference is satisfied by a number of different files.

File references consist of two parts: the primary filename and the filetype. Although the filetype is optional, it usually is generic. For example, the filetype ASM is used to denote that the file is an assembly language source file, while the primary filename distinguishes each particular source file. The two names are separated by a period, as shown in the following example:

```
filename.typ
```

In this example, filename is the primary filename of eight characters or less, and typ is the filetype of no more than three characters. As mentioned above, the name

```
filename
```

is also allowed and is equivalent to a filetype consisting of three blanks. The characters used in specifying an unambiguous file reference cannot contain any of the following special characters:

```
< > . , ; : = ? * [ ] % | ( ) / \
```

while all alphanumerics and remaining special characters are allowed.

An ambiguous file reference is used for directory search and pattern matching. The form of an ambiguous file reference is similar to an unambiguous reference, except the symbol ? can be interspersed throughout the primary and secondary names. In various commands throughout CP/M, the ? symbol matches any character of a filename in the ? position. Thus, the ambiguous reference

```
X?Z.C?M
```

matches the following unambiguous filenames

XYZ.COM

and

X3Z.CAM

The wildcard character can also be used in an ambiguous file reference. The * character replaces all or part of a filename or filetype. Note that

.

equals the ambiguous file reference

??????????

while

filename.*

and

*.typ

are abbreviations for

filename.???

and

?????????.typ

respectively. As an example,

A>DIR *.*

is interpreted by the CCP as a command to list the names of all disk files in the directory. The following example searches only for a file by the name X.Y:

A>DIR X.Y

Similarly, the command

```
A>DIR X?Y.C?M
```

causes a search for all unambiguous filenames on the disk that satisfy this ambiguous reference.

The following file references are valid unambiguous file references:

```
X  
X.Y  
XYZ  
XYZ.COM  
GAMMA  
GAMMA.1
```

As an added convenience, the programmer can generally specify the disk drive name along with the filename. In this case, the drive name is given as a letter A through P followed by a colon (:). The specified drive is then logged-in before the file operation occurs. Thus, the following are valid file references with disk name prefixes:

```
A:X.Y  
P:XYZ.COM  
B:XYZ  
B:X.A?M  
C:GAMMA  
C:*.ASM
```

All alphabetic lower-case letters in file and drive names are translated to upper-case when they are processed by the CCP.

1.3 Switching Disks

The operator can switch the currently logged disk by typing the disk drive name, A through P, followed by a colon when the CCP is waiting for console input. The following sequence of prompts and commands can occur after the CP/M system is loaded from disk

A:

CP/M VER 2.2

A>DIR List all files on disk A.

A:SAMPLE ASM SAMPLE PRN

A>B: Switch to disk B.

B>DIR *.ASM List all ASM files on B.

B:DUMP ASM FILES ASM

B>A: Switch back to A.

1.4 Built-in Commands

The file and device reference forms described can now be used to fully specify the structure of the built-in commands. Assume the following abbreviations in the description below:

ufn unambiguous file reference

afn ambiguous file reference

Recall that the CCP always translates lower-case characters to upper-case characters internally. Thus, lower-case alphabets are treated as if they are upper-case in command names and file references.

1.4.1 ERA Command

Syntax:

ERA afn

The ERA (erase) command removes files from the currently logged-in disk, for example, the disk name currently prompted by CP/M preceding the >. The files that are erased are those that satisfy the ambiguous file reference afn. The following examples illustrate the use of ERA:

ERA X.Y The file named X.Y on the currently logged disk is removed from the disk directory and the space is returned.

ERA X.* All files with primary name X are removed from the current disk.

ERA *.ASM All files with secondary name ASM are removed from the current disk.

ERA X?Y.C?M All files on the current disk that satisfy the ambiguous reference X?Y.C?M are deleted.

ERA *.* Erase all files on the current disk. In this case, the CCP prompts the console with the message

ALL FILES (Y/N)?

which requires a Y response before files are actually removed.

ERA B:*.PRN All files on drive B that satisfy the ambiguous reference ???????.PRN are deleted, independently of the currently logged disk.

1.4.2 DIR Command

Syntax:

DIR afn

The DIR (directory) command causes the names of all files that satisfy the ambiguous filename afn to be listed at the console device. As a special case, the command

DIR

lists the files on the currently logged disk (the command DIR is equivalent to the command DIR *.*). The following are valid DIR commands:

DIR X.Y
DIR X?Y.C?M
DIR ??Y

Similar to other CCP commands, the afn can be preceded by a drive name. The following DIR commands cause the selected drive to be addressed before the directory search takes place:

DIR B:
DIR B:X.Y
DIR B:*.A?M

If no files on the selected disk satisfy the directory request, the message NO FILE appears at the console.

1.4.3 REN Command

Syntax:

REN ufn1=ufn2

The REN (rename) command allows you to change the names of files on disk. The file satisfying ufn2 is changed to ufn1. The currently logged disk is assumed to contain the file to rename (ufn2). You can also type a left-directed arrow instead of the equal sign if the console supports this graphic character. The following are examples of the REN command:

REN X.Y=Q.R The file Q.R is changed to X.Y.

REN XYZ.COM=XYZ.XXX The file XYZ.COM is changed to XYZ.XXX.

The operator precedes either ufn1 or ufn2 (or both) by an optional drive address. If ufn1 is preceded by a drive name, then ufn2 is assumed to exist on the same drive. Similarly, if ufn2 is preceded by a drive name, then ufn1 is assumed to exist on the drive as well. The same drive must be specified in both cases if both ufn1 and ufn2 are preceded by drive names. The following REN commands illustrate this format:

REN A:X.ASM=Y.ASM The file Y.ASM is changed to X.ASM on drive A.

REN B:ZAP.BAS=ZOT.BAS The file ZOT.BAS is changed to ZAP.BAS on drive B.

REN B:A.ASM=B:A.BAK The file A.BAK is renamed to A.ASM on drive B.

If ufn1 is already present, the REN command responds with the error FILE EXISTS and not perform the change. If ufn2 does not exist on the specified disk, the message NO FILE is printed at the console.

1.4.4 SAVE Command

Syntax:

SAVE n ufn

The SAVE command places n pages (256-byte blocks) onto disk from the TPA and names this file ufn. In the CP/M distribution system, the TPA starts at 100H (hexadecimal) which is the second page of memory. The SAVE command must specify 2 pages of memory if the user's program occupies the area from 100H through 2FFH. The machine code file can be subsequently loaded and executed. The following are examples of the SAVE command:

SAVE 3 X.COM Copies 100H through 3FFH to X.COM.

SAVE 40 Q Copies 100H through 28FFH to Q. Note that 28 is the page count in 28FFH, and that $28H = 2 * 16 + 8 = 40$ decimal.

SAVE 4 X.Y Copies 100H through 4FFH to X.Y.

The SAVE command can also specify a disk drive in the ufn portion of the command, as shown in the following example:

SAVE 10 B:ZOT.COM Copies 10 pages, 100H through 0AFFH, to the file ZOT.COM on drive B.

1.4.5 TYPE Command

Syntax:

TYPE ufn

The TYPE command displays the content of the ASCII source file ufn on the currently logged disk at the console device. The following are valid TYPE commands:

TYPE X.Y
TYPE X.PLM
TYPE XXX

The TYPE command expands tabs, CTRL-I characters, assuming tab positions are set at every eighth column. The ufn can also reference a drive name.

TYPE B:X.PRN The file X.PRN from drive B is displayed.

1.4.6 USER Command

Syntax:

USER n

The USER command allows maintenance of separate files in the same directory. In the syntax line, n is an Integer value in the range 0 to 15. On cold start, the operator is automatically logged into user area number 0, which is compatible with standard CP/M 1 directories. You can issue the USER command at any time to move to another logical area within the same directory. Drives that are logged-in while addressing one user number are automatically active when the operator moves to another. A user number is simply a prefix that accesses particular directory entries on the active disks.

The active user number is maintained until changed by a subsequent USER command, or until a cold start when user 0 is again assumed.

1.5 Line Editing and Output Control

The CCP allows certain line-editing functions while typing command lines. The CTRL-key sequences are obtained by pressing the control and letter keys simultaneously. Further, CCP command lines are generally up to 255 characters in length; they are not acted upon until the carriage return key is pressed.

Table 1-1. Line-editing Control Characters

Character Meaning

CTRL-C	Reboots CP/M system when pressed at start of line.
CTRL-E	Physical end of line; carriage is returned, but line is not sent until the carriage return key is pressed.
CTRL-H	Backspaces one character position.

CTRL-I	Terminates current input (line-feed).
CTRL-M	Terminates current input (carriage return).
CTRL-P	Copies all subsequent console output to the currently assigned list device (see Section 1.6.1). Output is sent to the list device and the console device until the next CTRL-P is pressed.
CTRL-R	Retypes current command line; types a clean line following character deletion with rubouts.
CTRL-S	Stops the console output temporarily. Program execution and output continue when you press any character at the console, for example another CTRL-S. This feature stops output on high speed consoles, such as CRTs, in order to view a segment of output before continuing.
CTRL-U	Deletes the entire line typed at the console.
CTRL-X	Same as CTRL-U.
CTRL-Z	Ends input from the console (used in PIP and ED).
rub/del	Deletes and echoes the last character typed at the console.

1.6 Transient Commands

Transient commands are loaded from the currently logged disk and executed in the TPA. The transient commands for execution under the CCP are below. Additional functions are easily defined by the user (see Section 1.6.3).

Table 1-2. CP/M Transient Commands

<u>Command</u>	<u>Function</u>
STAT	Lists the number of bytes of storage remaining on the currently logged disk, provides statistical information about particular files, and displays or alters device assignment.
ASM	Loads the CP/M assembler and assembles the specified program from disk.
LOAD	Loads the file in Intel HEX machine code format and produces a file in machine executable form which can be loaded into the TPA. This loaded program becomes a new command under the CCP.
DDT	Loads the CP/M debugger into TPA and starts execution.
PIP	Loads the Peripheral Interchange Program for subsequent disk file and peripheral transfer operations.
ED	Loads and executes the CP/M text editor program.
SYSGEN	Creates a new CP/M system disk.
SUBMIT	Submits a file of commands for batch processing.
DUMP	Dumps the contents of a file in hex.
MOVCPM	Regenerates the CP/M system for a particular memory size.

Transient commands are specified in the same manner as built-in commands, and additional commands are easily defined by the user. For convenience, the transient command can be preceded by a drive name which causes the transient to be loaded from the specified drive into the TPA for execution. Thus, the command

B:STAT

causes CP/M to temporarily log in drive B for the source of the STAT transient, and then return to the original logged disk for subsequent processing.

1.6.1 STAT Command

Syntax:

```
STAT
STAT "command line"
```

The STAT command provides general statistical information about file storage and device assignment. Special forms of the command line allow the current device assignment to be examined and altered. The various command lines that can be specified are shown with an explanation of each form to the right.

STAT If you type an empty command line, the STAT transient calculates the storage remaining on all active drives, and prints one of the following messages:

```
d: R/W, SPACE: nnnK
```

```
d: R/O, SPACE: nnnK
```

for each active drive d:, where R/W indicates the drive can be read or written, and R/O indicates the drive is Read-Only (a drive becomes R/O by explicitly setting it to Read-Only, as shown below, or by inadvertently changing disks without performing a warm start). The space remaining on the disk in drive d: is given in kilobytes by nnn.

STAT d: If a drive name is given, then the drive is selected before the storage is computed. Thus, the command STAT B: could be issued while logged into drive A, resulting in the message

```
BYTES REMAINING ON B: nnnK
```

STAT afn The command line can also specify a set of files to be scanned by STAT. The files that satisfy afn are listed in alphabetical order, with storage requirements for each file under the heading:

```
RECS BYTES EXT D:FILENAME.TYP
rrrr bbbk ee d:filename.typ
```

where rrrr is the number of 128-byte records allocated to the file, bbb is the number of kilobytes allocated to the file ($bbb = rrrr * 128 / 1024$), ee is the number of 16K extensions ($ee = bbb / 16$), d is the drive name containing the file (A ... P), filename is the eight-character primary filename, and typ is the three-character filetype. After listing the individual files, the storage usage is summarized.

STAT d:afn The drive name can be given ahead of the afn. The specified drive is first selected, and the form STAT afn is executed.

STAT d:=R/O This form sets the drive given by d to Read-Only, remaining in effect until the next warm or cold start takes place. When a disk is Read-Only, the message

```
BDOS ERR ON d: Read-Only
```

appears if there is an attempt to write to the Read-Only disk. CP/M waits until a key is pressed before performing an automatic warm start, at which time the disk becomes R/W.

The STAT command allows you to control the physical-to-logical device assignment. See the IOBYTE function described in Sections 5 and 6. There are four logical peripheral devices that are, at any particular instant, each assigned one of several physical peripheral devices. The following is a list of the four logical devices:

- CON: is the system console device, used by CCP for communication with the operator.
- RDR: is the paper tape reader device.
- PUN: is the paper tape punch device.
- LST: is the output list device.

The actual devices attached to any particular computer system are driven by subroutines in the BIOS portion of CP/M. Thus, the logical RDR: device, for example, could actually be a high speed reader, teletype reader, or cassette tape. To allow some flexibility in device naming and assignment, several physical devices are defined in Table 1-3.

Table 1-3. Physical Devices

<u>Device</u>	<u>Meaning</u>
---------------	----------------

TTY:	Teletype device (slow speed console)
CRT:	Cathode ray tube device (high speed console)
BAT:	Batch processing (console is current RDR:, output goes to current LST: device)
UC1:	User-defined console
PTR:	Paper tape reader (high speed reader)
UR1:	User-defined reader #1
UR2:	User-defined reader #2
PTP:	Paper tape punch (high speed punch)
UP1:	User-defined punch #1
UP2:	User-defined punch #2
LPT:	Line printer
UL1:	User-defined list device #1

It is emphasized that the physical device names might not actually correspond to devices that the names imply. That is, you can implement the PTP: device as a cassette write operation. The exact correspondence and driving subroutine is defined in the BIOS portion of CP/M. In the standard distribution version of CP/M, these devices correspond to their names on the MDS 800 development system.

The command,

STAT VAL:

produces a summary of the available status commands, resulting in the output:

```
Temp R/O Disk d:$R/O
Set Indicator: filename.typ $R/O $R/W $SYS $DIR
Disk Status: DSK: d:DSK
Iobyte Assign:
```

which gives an instant summary of the possible STAT commands and shows the permissible logical-to-physical device assignments:

```
CON:=TTY:CRT:BAT:UCI:
RDR:=TTY:PTR:URI:UR2:
PUN:=TTY:PTP:UP1:UP2:
LST:=TTY:CRT:LPT:ULI:
```

The logical device to the left takes any of the four physical assignments shown to the right. The current logical-to-physical mapping is displayed by typing the command:

STAT DEV:

This command produces a list of each logical device to the left and the current corresponding physical device to the right. For example, the list might appear as follows:

```
CON:=CRT:
RDR:=URI:
PUN:=PTP:
LST:=TTY:
```

The current logical-to-physical device assignment is changed by typing a STAT command of the form:

```
STAT ld1=pd1,ld2=pd2,...,ldn=pdn
```

where ld1 through ldn are logical device names and pd1 through pdn are compatible physical device names. For example, ld1 and pd1 appear on the same line in the VAL: command shown above. The following example shows valid STAT commands that change the current logical-to-physical device assignments:

```
STAT CON:=CRT:
STAT PUN:=TTY:;LST:=LPT:;RDR:=TTY
```

The command form,

```
STAT d:filename.typ $$
```

where d: is an optional drive name and filename.typ is an unambiguous or ambiguous filename, produces the following output display format:

Size	Recs	Bytes	Ext	Acc
48	48	6K	1	R/O A:ED.COM
55	55	12K	1	R/O (A:PIP.COM)
65536	128	16K	2	R/W A:X.DAT

where the \$\$ parameter causes the Size field to be displayed. Without the \$\$, the Size field is skipped, but the remaining fields are displayed. The Size field lists the virtual file size in records, while the Recs field sums the number of virtual records in each extent. For files constructed sequentially, the Size and Recs fields are identical. The Bytes field lists the actual number of bytes allocated to the corresponding file. The minimum allocation unit is determined at configuration time; thus, the number of bytes corresponds to the record count plus the remaining unused space in the last allocated block for sequential files. Random access files are given data areas only when written, so the Bytes field contains the only accurate allocation figure. In the case of random access, the Size field gives the logical end-of-file record position and the Recs field counts the logical records of each extent. Each of these extents, however, can contain unallocated holes even though they are added into the record count.

The Ext field counts the number of physical extents allocated to the file. The Ext count corresponds to the number of directory entries given to the file. Depending on allocation size, there can be up to 128K bytes (8 logical extents) directly addressed by a single directory entry. In a special case, there are actually 256K bytes that can be directly addressed by a physical extent.

The Acc field gives the R/O or R/W file indicator, which you can change using the commands shown. The four command forms,

```
STAT d:filename.typ $R/O
STAT d:filename.typ $R/W
STAT d:filename.typ $SYS
STAT d:filename.typ $DIR
```

set or reset various permanent file indicators. The R/O indicator places the file, or set of files, in a Read-Only status until changed by a subsequent STAT command. The R/O status is recorded in the directory with the file so that it remains R/O through intervening cold start operations. The R/W indicator places the file in a permanent Read-Write status. The SYS indicator attaches the system indicator to the file, while the DIR command removes the system indicator. The filename.typ may be ambiguous or unambiguous, but files whose attributes are changed are listed at the console when the change occurs. The drive name denoted by d: is optional.

When a file is marked R/O, subsequent attempts to erase or write into the file produce the following BDOS message at your screen:

BDOS Err on d: File R/O

lists the drive characteristics of the disk named by d: that is in the range A:, B:,...,P:. The drive characteristics are listed in the following format:

```
d: Drive Characteristics
65536: 128 Byte Record Capacity
8192: Kilobyte Drive Capacity
128: 32 Byte Directory Entries
0: Checked Directory Entries
1024: Records/Extent
128: Records/Block
58: Sectors/Track
2: Reserved Tracks
```

where d: is the selected drive, followed by the total record capacity (65536 is an eight-megabyte drive), followed by the total capacity listed in kilobytes. The directory size is listed next, followed by the checked entries. The number of checked entries is usually identical to the directory size for removable media, because this mechanism is used to detect changed media during CP/M operation without an intervening warm start. For fixed media, the number is usually zero, because the media are not changed without at least a cold or warm start.

The number of records per extent determines the addressing capacity of each directory entry (1024 times 128 bytes, or 128K in the previous example). The number of records per block shows the basic allocation size (in the example, 128 records/block times 128 bytes per record, or 16K bytes per block). The listing is then followed by the number of physical sectors per track and the number of reserved tracks.

For logical drives that share the same physical disk, the number of reserved tracks can be quite large because this mechanism is used to skip lower-numbered disk areas allocated to other logical disks. The command form

STAT DSK:

produces a drive characteristics table for all currently active drives. The final STAT command form is

STATUSR:

which produces a list of the user numbers that have files on the currently addressed disk. The display format is

```
Active User: 0
Active Files: 0 1 3
```

where the first line lists the currently addressed user number, as set by the last CCP USER command, followed by a list of user numbers scanned from the current directory. In this case, the active user number is 0 (default at cold start) with three user numbers that have active files on the current disk. The operator can subsequently examine the directories of the other user numbers by logging in with USER 1 or USER 3 commands, followed by a DIR command at the CCP level.

1.6.2 ASM Command

Syntax:

```
ASM ufn
```

The ASM command loads and executes the CP/M 8080 assembler. The ufn specifies a source file containing assembly language statements, where the filetype is assumed to be ASM and is not specified. The following ASM commands are valid:

```
ASM
ASM GAMMA
```

The two-pass assembler is automatically executed. Assembly errors that occur during the second pass are printed at the console.

The assembler produces a file:

X.PRN

where X is the primary name specified in the ASM command. The PRN file contains a listing of the source program with embedded tab characters if present in the source program, along with the machine code generated for each statement and diagnostic error messages, if any. The PRN file is listed at the console using the TYPE command, or sent to a peripheral device using PIP (see Section 1.6.4). Note that the PRN file contains the original source program, augmented by miscellaneous assembly information in the leftmost 16 columns; for example, program addresses and hexadecimal machine code. The PRN file serves as a backup for the original source file. If the source file is accidentally removed or destroyed, the PRN file can be edited by removing the leftmost 16 characters of each line (see Section 2). This is done by issuing a single editor macro command. The resulting file is identical to the original source file and can be renamed from PRN to ASM for subsequent editing and assembly. The file

A.HEX

is also produced, which contains 8080 machine language in Intel HEX format suitable for subsequent loading and execution (see Section 1.6.3). For complete details of CP/M's assembly language program, see Section 3.

The source file for assembly is taken from an alternate disk by prefixing the assembly language filename by a disk drive name. The command

```
ASM B:ALPHA
```

loads the assembler from the currently logged drive and processes the source program ALPHA.ASM on drive B. The HEX and PRN files are also placed on drive B in this case.

1.6.3 LOAD Command

Syntax:

LOAD ufn

The LOAD command reads the file ufn, which is assumed to contain HEX format machine code, and produces a memory image file that can subsequently be executed. The filename ufn is assumed to be of the form:

X.HEX

and only the filename X need be specified in the command. The LOAD command creates a file named

X.COM

that marks it as containing machine executable code. The file is actually loaded into memory and executed when the user types the filename X immediately after the prompting character > printed by the CCP.

Generally, the CCP reads the filename X following the prompting character and looks for a built-in function name. If no function name is found, the CCP searches the system disk directory for a file by the name

X.COM

If found, the machine code is loaded into the TPA, and the program executes. Thus, the user need only LOAD a hex file once; it can be subsequently executed any number of times by typing the primary name. This way, you can invent new commands in the CCP. Initialized disks contain the transient commands as COM files, which are optionally deleted. The operation takes place on an alternate drive if the filename is prefixed by a drive name. Thus,

LOAD B:BETA

brings the LOAD program into the TPA from the currently logged disk and operates on drive B after execution begins.

Note: the BETA.HEX file must contain valid Intel format hexadecimal machine code records (as produced by the ASM program, for example) that begin at 100H of the TPA. The addresses in the hex records must be in ascending order; gaps in unfilled memory regions are filled with zeroes by the LOAD command as the hex records are read. Thus, LOAD must be used only for creating CP/M standard COM files that operate in the TPA. Programs that occupy regions of memory other than the TPA are loaded under DDT.

1.6.4 PIP

Syntax:

PIP

PIP destination=source#1,source#2,...,source#n

PIP is the CP/M Peripheral Interchange Program that implements the basic media conversion operations necessary to load, print, punch, copy, and combine disk files. The PIP program is initiated by typing one of the following forms:

PIP

PIP command line

In both cases PIP is loaded into the TPA and executed. In the first form, PIP reads command lines directly from the console, prompted with the * character, until an empty command line is typed (for example, a single carriage return is issued by the operator). Each successive command line causes some media conversion to take place according to the rules shown below.

In the second form, the PIP command is equivalent to the first, except that the single command line given with the PIP command is automatically executed, and PIP terminates immediately with no further prompting of the console for input command lines. The form of each command line is

destination=source#1,source#2,...,source#n

where destination is the file or peripheral device to receive the data, and source#1,...,source#n is a series of one or more files or devices that are copied from left to right to the destination.

When multiple files are given in the command line (for example, $n > 1$), the individual files are assumed to contain ASCII characters, with an assumed CP/M end-of-file character (CTRL-Z) at the end of each file (see the O parameter to override this assumption). Lower-case ASCII alphabets are internally translated to upper-case to be consistent with CP/M file and device name conventions. Finally, the total command line length cannot exceed 255 characters. CTRL-E can be used to force a physical carriage return for lines that exceed the console width.

The destination and source elements are unambiguous references to CP/M source files with or without a preceding disk drive name. That is, any file can be referenced with a preceding drive name (A: through P:) that defines the particular drive where the file can be obtained or stored. When the drive name is not included, the currently logged disk is assumed. The destination file can also appear as one or more of the source files; in which case the source file is not altered until the entire concatenation is complete. If it already exists, the destination file is removed if the command line is properly formed. It is not removed if an error condition arises. The following command lines, with explanations to the right, are valid as input to PIP:

X=Y	Copies to file X from file Y, where X and Y are unambiguous filenames; Y remains unchanged.
X=Y,Z	Concatenates files Y and Z and copies to file X, with Y and Z unchanged.
X.ASM=Y.ASM,Z.ASM	Creates the file X.ASM from the concatenation of the Y and Z.ASM files.
NEW.ZOT=B:OLD.ZAP	Moves a copy of OLD.ZAP from drive B to the currently logged disk; names the file NEW.ZOT.
B:A.U=B:B.V,A:C.W,D.X	Concatenates file B.Y from drive B with C.W from drive A and D.X from the logged disk; creates the file A.U on drive B.

For convenience, PIP allows abbreviated commands for transferring files between disk drives. The abbreviated PIP forms are

```
PIP d:=afn
PIP d1:=d2:afn
PIP ufn=d2:
PIP d1:ufn=d2:
```

The first form copies all files from the currently logged disk that satisfy the afn to the same files on drive d, where d = A...P. The second form is equivalent to the first, where the source for the copy is drive d2 where d2 = A ... P. The third form is equivalent to the command PIP d1:ufn=d2:ufn which copies the file given by ufn from drive d2 to the file ufn on drive d1. The fourth form is equivalent to the third, where the source disk is explicitly given by d2.

The source and destination disks must be different in all of these cases. If an afn is specified, PIP lists each ufn that satisfies the afn as it is being copied. If a file exists by the same name as the destination file, it is removed after successful completion of the copy and replaced by the copied file.

The following PIP commands give examples of valid disk-to-disk copy operations:

B=*.COM	Copies all files that have the secondary name COM to drive B from the current drive.
A:=B:ZAP.*	Copies all files that have the primary name ZAP to drive A from drive B.
ZAP.ASM=B:	Same as ZAP.ASM=B:ZAP.ASM
B:ZOT.COM=A:	Same as B:ZOT.COM=A:ZOT.COM
B:=GAMMA.BAS	Same as B:GAMMA.BAS=GAMMA.BAS
B:=A:GAMMA.BAS	Same as B:GAMMA.BAS=A:GAMMA.BAS

PIP allows reference to physical and logical devices that are attached to the CP/M system. The device names are the same as given under the STAT command, along with a number of specially named devices. The following is a list of logical devices given in the STAT command

CON: (console)
RDR: (reader)
PUN: (punch)
LST: (list)

while the physical devices are

TTY: (console , reader, punch, or list)
CRT: (console, or list), UC1: (console)
PTR: (reader), URI: (reader), UR2: (reader)
PTP: (punch), UPI: (punch), UP2: (punch)
LPT: (list), ULI: (list)

The BAT: physical device is not included, because this assignment is used only to indicate that the RDR: and LST: devices are used for console input/output.

The RDR, LST, PUN, and CON devices are all defined within the BIOS portion of CP/M, and are easily altered for any particular I/O system. The current physical device mapping is defined by IOBYTE; see Section 6 for a discussion of this function. The destination device must be capable of receiving data, for example, data cannot be sent to the punch, and the source devices must be capable of generating data, for example, the LST: device cannot be read.

The following list describes additional device names that can be used in PIP commands.

-NUL: sends 40 nulls (ASCII 0s) to the device. This can be issued at the end of punched output.

-EOF: sends a CP/M end-of-file (ASCII CTRL-Z) to the destination device (sent automatically at the end of all ASCII data transfers through PIP).

-INP: is a special PIP input source that can be patched into the PIP program. PIP gets the input data character-by-character, by CALLing location 103H, with data returned in location 109H (parity bit must be zero).

-OUT: is a special PIP output destination that can be patched into the PIP program. PIP CALLs location 106H with data in register C for each character to transmit. Note that locations 109H through 1FFH of the PIP memory image are not used and can be replaced by special purpose drivers using DDT (see Section 4).

-PRN: is the same as LST:, except that tabs are expanded at every eighth character position, lines are numbered, and page ejects are inserted every 60 lines with an initial eject (same as using PIP options [t8np]).

File and device names can be interspersed in the PIP commands. In each case, the specific device is read until end-of-file (CTRL-Z for ASCII files, and end-of-data for non-ASCII disk files). Data from each device or file are concatenated from left to right until the last data source has been read.

The destination device or file is written using the data from the source files, and an end-of-file character, CTRL-Z, is appended to the result for ASCII files. If the destination is a disk file, a temporary file is created (\$\$\$ secondary name) that is changed to the actual filename only on successful completion of the copy. Files with the extension COM are always assumed to be non-ASCII.

The copy operation can be aborted at any time by pressing any key on the keyboard. PIP responds with the message ABORTED to indicate that the operation has not been completed. If any operation is aborted, or if an error occurs during processing, PIP removes any pending commands that were set up while using the SUBMIT command.

PIP performs a special function if the destination is a disk file with type HEX (an Intel hex-formatted machine code file), and the source is an external peripheral device, such as a paper tape reader. In this case, the PIP program checks to ensure that the source file contains a properly formed hex file, with legal hexadecimal values and checksum records.

When an invalid input record is found, PIP reports an error message at the console and waits for corrective action. Usually, you can open the reader and rerun a section of the tape (pull the tape back about 20 inches). When the tape is ready for the reread, a single carriage return is typed at the console, and PIP attempts another read. If the tape position cannot be properly read, continue the read by typing a return following the error message, and enter the record manually with the ED program after the disk file is constructed.

PIP allows the end-of-file to be entered from the console if the source file is an RDR: device. In this case, the PIP program reads the device and monitors the keyboard. If CTRL-Z is typed at the keyboard, the read operation is terminated normally.

The following are valid PIP commands:

PIP LST:=X.PRN

Copies X.PRN to the LST device and terminates the PIP program.

PIP

Starts PIP for a sequence of commands. PIP prompts with *.

*CON:=X.ASM,Y.ASM,Z.ASM

Concatenates three ASM files and copies to the CON device.

*X.HEX=CON:,Y.HEX,PTR:

Creates a HEX file by reading the CON until a CTRL-Z is typed, followed by data from Y.HEX and PTR until a CTRL-Z is encountered.

PIP PUN:=NUL:,X.ASM,EOF:,NUL:

Sends 40 nulls to the punch device; copies the X.ASM file to the punch, followed by an end-of-file, CTRL-Z, and 40 more null characters.

(carriage return)

A single carriage return stops PIP.

You can also specify one or more PIP parameters, enclosed in left and right square brackets, separated by zero or more blanks. Each parameter affects the copy operation, and the enclosed list of parameters must immediately follow the affected file or device. Generally, each parameter can be followed by an optional decimal integer value (the S and Q parameters are exceptions). Table 1-4 describes valid PIP parameters.

Table 1-4. PIP Parameters

<u>Parameter</u>	<u>Meaning</u>
B	Blocks mode transfer. Data are buffered by PIP until an ASCII x-off character, CTRL-S, is received from the source device. This allows transfer of data to a disk file from a continuous reading device, such as a cassette reader. Upon receipt of the x-off, PIP clears the disk buffers and returns for more input data. The amount of data that can be buffered depends on the memory size of the host system. PIP issues an error message if the buffers overflow.
Dn	Deletes characters that extend past column n in the transfer of data to the destination from the character source. This parameter is generally used to truncate long lines that are sent to a narrow printer or console device.
E	Echoes all transfer operations to the console as they are being performed.
F	Filters form-feeds from the file. All embedded form-feeds are removed. The P parameter can be used simultaneously to insert new form-feeds.
Gn	Gets file from user number n (n in the range 0-15).
H	Transfers HEX data. All data are checked for proper Intel hex file format. Nonessential characters between hex records are removed during the copy operation. The console is prompted for corrective action in case errors occur.
I	Ignores :00 records in the transfer of Intel hex format file. The I parameter automatically sets the H parameter.
L	Translates upper-case alphabets to lower-case.
N	Adds line numbers to each line transferred to the destination, starting at one and incrementing by 1. Leading zeroes are suppressed, and the number is followed by colon. If N2 is specified, leading zeroes are included and a tab is inserted
a following	the number. The tab is expanded if T is set.

- O Transfers non-ASCII object files. The normal CP/M end-of-file is ignored.
- Pn Includes page ejects at every n lines with an initial page eject. If n = 1 or is excluded altogether, page ejects occur every 60 lines. If the F parameter is used, form-feed suppression takes place before the new page ejects are inserted.
- QS^Z Quits copying from the source device or file when the string S, terminated by CTRL-Z, is encountered.
- R Reads system files.
- Ss^Z Start copying from the source device when the string s, terminated by CTRL-Z, is encountered. The S and Q parameters can be used to abstract a particular section of a file, such as a subroutine. The start and quit strings are always included in the copy operation.
- If you specify a command line after the PIP command keyword, the CCP strings following the S and Q parameters to uppercase. If you do not specify a command line, PIP does not perform the automatic upper-case translation.
- Tn Expands tabs, CTRL-I characters, to every nth column during the transfer of characters to the destination from the source.
- U Translates lower-case alphabets to upper-case during the copy operation.
- V Verifies that data have been copied correctly by rereading after the write operation (the destination must be a disk file).
- W Writes over R/O files without console interrogation.
- Z Zeros the parity bit on input for each ASCII character.

The following examples show valid PIP commands that specify parameters in the file transfer.

PIP X.ASM=B:[V]

Copies X.ASM from drive B to the current drive and verifies that the data were properly copied.

PIP LPT:=X.ASM[NT8U]

Copies X.ASM to the LPT: device; numbers each line, expands tabs to every eighth column, and translates lower-case alphabetic to upper-case.

PIP PUN:=X.HEX[I],Y.ZOT[H]

First copies X.HEX to the PUN: device and ignores the trailing :00 record in X.HEX; continues the transfer of data by reading Y.ZOT, which contains HEX records, including any :00 records it contains.

PIP X.LIB=Y.ASM[sSUBR1:^zqJMP L3^z]

Copies from the file Y.ASM into the file X.LIB. The command starts the copy when the string SUBR1: has been found, and quits copying after the string JMP L3 is encountered.

PIP PRN:=X.ASM[p50]

Sends X.ASM to the LST: device with line numbers, expands tabs to every eighth column, and elects pages at every 50th line. The assumed parameter list for a PRN file is nt8p60; p50 overrides the default value.

Under normal operation, PIP does not overwrite a file that is set to a permanent R/O status. If an attempt is made to overwrite an R/O file, the following prompt appears:

DESTINATION FILE IS R/O, DELETE (Y/N)?

If you type Y, the file is overwritten. Otherwise, the following response appears:

** NOT DELETED **

The file transfer is skipped, and PIP continues with the next operation in sequence. To avoid the prompt and response in the case of R/O file overwrite, the command line can include the W parameter, as shown in this example:

```
PIP A:=B:*.COM[W]
```

The W parameter copies all nonsystem files to the A drive from the B drive and overwrites any R/O files in the process. If the operation involves several concatenated files, the W parameter need only be included with the last file in the list, as in this example:

```
PIP A.DAT=B.DAT,F:NEW.DAT,G:OLD.DAT[W]
```

Files with the system attribute can be included in PIP transfers if the R parameter is included; otherwise, system files are not recognized. For example, the command line:

```
PIP ED.COM=B:ED.COM[R]
```

reads the ED.COM file from the B drive, even if it has been marked as an R/O and system file. The system file attributes are copied, if present.

Downward compatibility with previous versions of CP/M is only maintained if the file does not exceed one megabyte, no file attributes are set, and the file is created by user 0. If compatibility is required with nonstandard, for example, double-density versions of 1.4, it might be necessary to select 1.4 compatibility mode when constructing the internal disk parameter block. See Section 6 and refer to Section 6.10, which describes BIOS differences.

Note: to copy files into another user area, PIP.COM must be located in that user area. Use the following procedure to make a copy of PIP.COM in another user area.

```
USER 0          Log in user 0.
DDT PIP.COM (note PIP size s) Load PIP to memory.
GO             Return to CCP.
USER 3          Log in user 3.
SAVE s PIP.COM
```

In this procedure, s is the integral number of memory pages, 256- byte segments, occupied by PIP. The number s can be determined when PIP.COM is loaded under DDT, by referring to the value under the NEXT display. If, for example, the next available address is 1D00, then PIP.COM requires 1C hexadecimal pages, or 1 times 16 + 12 = 28 pages, and the value of s is 28 in the subsequent save. Once PIP is copied in this manner, it can be copied to another disk belonging to the same user number through normal PIP transfers.

1.6.5 ED Command

Syntax:

ED ufn

The ED program is the CP/M system context editor that allows creation and alteration of ASCII files in the CP/M environment. Complete details of operation are given in Section 2. ED allows the operator to create and operate upon source files that are organized as a sequence of ASCII characters, separated by end-of-line characters (a carriage return/line-feed sequence). There is no practical restriction on line length (no single line can exceed the size of the working memory) that is defined by the number of characters typed between carriage returns.

The ED program has a number of commands for character string searching, replacement, and insertion that are useful for creating and correcting programs or text files under CP/M. Although the CP/M has a limited memory work space area (approximately 5000 characters in a 20K CP/M system), the file size that can be edited is not limited, since data are easily paged through this work area.

If it does not exist, ED creates the specified source file and opens the file for access. If the source file does exist, the programmer appends data for editing (see the A command). The appended data can then be displayed, altered, and written from the work area back to the disk (see the W command). Particular points in the program can be automatically paged and located by context, allowing easy access to particular portions of a large file (see the N command).

If you type the following command line:

ED X.ASM

the ED program creates an intermediate work file with the name

X.\$\$\$

to hold the edited data during the ED run. Upon completion of ED, the X.ASM file (original file) is renamed to X.BAK, and the edited work file is renamed to X.ASM. Thus, the X.BAK file contains the original unedited file, and the X.ASM file contains the newly edited file. The operator can always return to the previous version of a file by removing the most recent version and renaming the previous version. If the current X.ASM file has been improperly edited, the following sequence of commands reclaim the backup file.

DIR X.*Checks to see that BAK file is available.

ERA X.ASMErases most recent version.

REN X.ASM=X.BAKRenames the BAK file to ASM.

You can abort the edit at any point (reboot, power failure, CTRL-C, or CTRL-Q command) without destroying the original file. In this case, the BAK file is not created and the original file is always intact.

The ED program allows the user to edit the source on one disk and create the back-up file on another disk. This form of the ED command is

ED ufn d:

where ufn is the name of the file to edit on the currently logged disk and d is the name of an alternate drive. The ED program reads and processes the source file and writes the new file to drive d using the name ufn. After processing, the original file becomes the back-up file. If the operator is addressing disk A, the following command is valid.

ED X.ASM B:

This edits the file X.ASM on drive A, creating the new file X.\$\$\$ on drive B. After a successful edit, A:X.ASM is renamed to A:X.BAK, and B:X.\$\$\$ is renamed to B:X.ASM. For convenience, the currently logged disk becomes drive B at the end of the edit. Note that if a file named B:X.ASM exists before the editing begins, the following message appears on the screen:

FILE EXISTS

This message is a precaution against accidentally destroying a source file. You should first erase the existing file and then restart the edit operation.

Similar to other transient commands, editing can take place on a drive different from the currently logged disk by preceding the source filename by a drive name. The following are examples of valid edit requests:

ED A:X.ASM Edits the file X ASM on drive A, with new file and back-up on drive A.

ED B:X.ASM A: Edits the file X.ASM on drive B to the temporary file X.\$\$\$ on drive A. After editing, this command changes X.ASM on drive B to X.BAK and changes X.\$\$\$ on drive A to X.ASM

1.6.6 SYSGEN Command

Syntax:

SYSGEN

The SYSGEN transient command allows generation of an initialized disk containing the CP/M operating system. The SYSGEN program prompts the console for commands by interacting as shown.

SYSGEN<cr>

Initiates the SYSGEN program.

SYSGEN VERSION x.x

SYSGEN sign-on message.

SOURCE DRIVE NAME
(OR RETURN TO SKIP)

Respond with the drive name (one of the letters A, B, C, or D) of the disk containing a CP/M system, usually A. If a copy of CP/M already exists in memory due to a MOVCPM command, press only a carriage return. Typing a drive name d causes the response:

SOURCE ON d THEN TYPE RETURN

Place a disk containing the CP/M operating system on drive d (d is one of A, B, C, or D). Answer by pressing a carriage return when ready.

FUNCTION COMPLETE

System is copied to memory. SYSGEN then prompts with the following:

DESTINATION DRIVE NAME
(OR RETURN TO REBOOT)

If a disk is being initialized, place the new disk into a drive and answer with the drive name. Otherwise, press a carriage return and the system reboots from drive A. Typing drive name d causes SYSGEN to prompt with the following message:

DESTINATION ON d
THEN TYPE RETURN

Place new disk into drive d; press return when ready.

FUNCTION COMPLETE

New disk is initialized in drive d.

The DESTINATION prompt is repeated until a single carriage return is pressed at the console, so that more than one disk can be initialized.

Upon completion of a successful system generation, the new disk contains the operating system, and only the built-in commands are available. An IBM-compatible disk appears to CP/M as a disk with an empty directory; therefore, the operator must copy the appropriate COM files from an existing CP/M disk to the newly constructed disk using the PIP transient.

You can copy all files from an existing disk by typing the following PIP command:

```
PIP B:=A:*. *[v]
```

This command copies all files from disk drive A to disk drive B and verifies that each file has been copied correctly. The name of each file is displayed at the console as the copy operation proceeds.

Note that a SYSGEN does not destroy the files that already exist on a disk; it only constructs a new operating system. If a disk is being used only on drives B through P and will never be the source of a bootstrap operation on drive A, the SYSGEN need not take place.

1.6.7 SUBMIT Command

Syntax:

SUBMIT ufn parm#1 ... parm#n

The SUBMIT command allows CP/M commands to be batched for automatic processing. The ufn given in the SUBMIT command must be the filename of a file that exists on the currently logged disk, with an assumed file type of SUB. The SUB file contains CP/M prototype commands with possible parameter substitution. The actual parameters parm#1 ... parm#n are substituted into the prototype commands, and, if no errors occur, the file of substituted commands are processed sequentially by CP/M.

The prototype command file is created using the ED program, with interspersed \$ parameters of the form:

\$1 \$2 \$3 ... \$n

corresponding to the number of actual parameters that will be included when the file is submitted for execution. When the SUBMIT transient is executed, the actual parameters parm#1 ... parm#n are paired with the formal parameters \$1 ... \$n in the prototype commands. If the numbers of formal and actual parameters do not correspond, the SUBMIT function is aborted with an error message at the console. The SUBMIT function creates a file of substituted commands with the name

\$\$\$SUB

on the logged disk. When the system reboots, at the termination of the SUBMIT, this command file is read by the CCP as a source of input rather than the console. If the SUBMIT function is performed on any disk other than drive A, the commands are not processed until the disk is inserted into drive A and the system reboots. You can abort command processing at any time by pressing the rubout key when the command is read and echoed. In this case, the \$\$\$SUB file is removed and the subsequent commands come from the console. Command processing is also aborted if the CCP detects an error in any of the commands. Programs that execute under CP/M can abort processing of command files when error conditions occur by erasing any existing \$\$\$SUB file.

To introduce dollar signs into a SUBMIT file, you can type a \$\$ which reduces to a single \$ within the command file. An up arrow, ^, precedes an alphabetic character s, which produces a single CTRL-X character within the file.

The last command in a SUB file can initiate another SUB file, allowing chained batch commands.

Suppose the file ASMBL.SUB exists on disk and contains the prototype commands:

```
ASM $1
DIR $1.*
ERA *.BAK
PIP $2:=$1.PRN
ERA $1.PRN
```

then, you issue the following command:

```
SUBMIT ASMBL X PRN
```

The SUBMIT program reads the ASMBL.SUB file, substituting X for all occurrences of \$1 and PRN for all occurrences of \$2. This results in a \$\$\$SUB file containing the commands:

```
ASM X
DIR X.*
ERA *.BAK
PIP PRN:=X.PRN
ERA X.PRN
```

which are executed in sequence by the CCP.

The SUBMIT function can access a SUB file on an alternate drive by preceding the filename by a drive name. Submitted files are only acted upon when they appear on drive A. Thus, it is possible to create a submitted file on drive B that is executed at a later time when inserted in drive A.

An additional utility program called XSUB extends the power of the SUBMIT facility to include line input to programs as well as the CCP. The XSUB command is included as the first line of the SUBMIT file. When it is executed, XSUB self-relocates directly below the CCP. All subsequent SUBMIT command lines are processed by XSUB so that programs that read buffered console input, BDOS Function 10, receive their input directly from the SUBMIT file. For example, the file SAVER.SUB can contain the following SUBMIT lines:

```
XSUB
DDT
I $1.COM
R
G0
SAVE 1 $2.COM
```

a subsequent SUBMIT command, such as

```
A:SUBMIT SAVER PIP Y
```

substitutes PIP for \$1 and Y for \$2 in the command stream. The XSUB program loads, followed by DDT, which is sent to the command lines PIP.COM, R, and G0, thus returning to the CCP. The final command SAVE 1 Y.COM is processed by the CCP.

The XSUB program remains in memory and prints the message

```
(xsub active)
```

on each warm start operation to indicate its presence. Subsequent SUBMIT command streams do not require the XSUB, unless an intervening cold start occurs. Note that XSUB must be loaded after the optional CP/M DESPOOL utility, if both are to run simultaneously.

1.6.8 DUMP Command

Syntax:

```
DUMP ufn
```

The DUMP program types the contents of the disk file (ufn) at the console in hexadecimal form. The file contents are listed sixteen bytes at a time, with the absolute byte address listed to the left of each line in hexadecimal. Long typeouts can be aborted by pressing the rubout key during printout. The source listing of the DUMP program is given in Section 5 as an example of a program written for the CP/M environment.

1.6.9 MOVCPM CommandSyntax:

MOVCPM

The MOVCPM program allows you to reconfigure the CP/M system for any particular memory size. Two optional parameters can be used to indicate the desired size of the new system and the disposition of the new system at program termination. If the first parameter is omitted or an * is given, the MOVCPM program reconfigures the system to its maximum size, based upon the kilobytes of contiguous RAM in the host system (starting at 0000H). If the second parameter is omitted, the system is executed, but not permanently recorded; if * is given, the system is left in memory, ready for a SYSGEN operation. The MOVCPM program relocates a memory image of CP/M and places this image in memory in preparation for a system generation operation. The following is a list of MOVCPM command forms:

MOYCPM	Relocates and executes CP/M for management of the current memory configuration (memory is examined for contiguous RAM, starting at 100H). On completion of the relocation, the new system is executed but not permanently recorded on the disk. The system that is constructed contains a BIOS for the Intel MDS 800.
MOVCPM n	Creates a relocated CP/M system for management of an n kilobyte system (n must be in the range of 20 to 64), and executes the system as described.
MOYCPM * *	Constructs a relocated memory image for the current memory configuration, but leaves the memory image in memory in preparation for SYSGEN operation.
a	
MOYCPM n *	Constructs a relocated memory image for an n kilobyte memory system, and leaves the memory image in preparation for a SYSGEN operation.

For example, the command,

```
MOVCPM * *
```

constructs a new version of the CP/M system and leaves it in memory, ready for a SYSGEN operation. The message

```
READY FOR 'SYSGEN' OR  
'SAYE 34 CPMxx.COM'
```

appears at the console upon completion, where xx is the current memory size in kilobytes. You can then type the following sequence:

SYSGEN	This starts the system generation.
SOURCE DRIVE NAME RETURN TO SKIP)	Respond with a carriage return to skip the CP/M read (OR operation, because the system is already in memory as a result of the previous MOVCPM operation.
DESTINATION DRIVE NAME OR RETURN TO REBOOT)	Respond with B to write new system to the disk in drive B. SYSGEN prompts with the following message:
DESTINATION ON B, THEN TYPE RETURN	Place the new disk on drive B and press the RETURN key when ready.

If you respond with A rather than B above, the system is written to drive A rather than B. SYSGEN continues to print this prompt:

```
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)
```

until you respond with a single carriage return, which stops the SYSGEN program with a system reboot.

You can then go through the reboot process with the old or new disk. Instead of performing the SYSGEN operation, you can type a command of the form:

```
SAVE 34 CPMxx.COM
```

at the completion of the MOVCPM function, where xx is the value indicated in the SYSGEN message. The CP/M memory image on the currently logged disk is in a form that can be patched. This is necessary when operating in a nonstandard environment where the BIOS must be altered for a particular peripheral device configuration, as described in Section 6.

The following are valid MOVCPM commands:

MOVCPM 48	Constructs a 48K version of CP/M and starts execution.
MOVCPM 48 *	Constructs a 48K version of CP/M in preparation for permanent recording; the response is READY FOR 'SYSGEN' OR 'SAVE 34 CPM48.COM'
MOVCPM	Constructs a maximum memory version of CP/M and starts execution.

The newly created system is serialized with the number attached to the original disk and is subject to the conditions of the Digital Research Software Licensing Agreement.

1.7 BDOS Error Messages

There are three error situations that the Basic Disk Operating System intercepts during file processing. When one of these conditions is detected, the BDOS prints the message:

```
BDOS ERR ON d: error
```

where d is the drive name and error is one of the three error messages:

```
BAD SECTOR  
SELECT  
READ ONLY
```

The BAD SECTOR message indicates that the disk controller electronics has detected an error condition in reading or writing the disk. This condition is generally caused by a malfunctioning disk controller or an extremely worn disk. If you find that CP/M reports this error more than once a month, the state of the controller electronics and the condition of the media should be checked.

You can also encounter this condition in reading files generated by a controller produced by a different manufacturer. Even though controllers claim to be IBM compatible, one often finds small differences in recording formats. The MDS-800 controller, for example, requires two bytes of ones following the data CRC byte, which is not required in the IBM format. As a result, disks generated by the Intel MDS can be read by almost all other IBM-compatible systems, while disk files generated on other manufacturers' equipment produce the BAD SECTOR message when read by the MDS. To recover from this condition, press a CTRL-C to reboot (the safest course), or a return, which ignores the bad sector in the file operation.

Note: pressing a return might destroy disk integrity if the operation is a directory write. Be sure you have adequate back-ups in this case.

The SELECT error occurs when there is an attempt to address a drive beyond the range supported by the BIOS. In this case, the value of d in the error message gives the selected drive. The system reboots following any input from the console.

The READ ONLY message occurs when there is an attempt to write to a disk or file that has been designated as Read-Only in a STAT command or has been set to Read-Only by the BDOS. Reboot CP/M by using the warm start procedure, CTRL-C, or by performing a cold start whenever the disks are changed. If a changed disk is to be read but not written, BDOS allows the disk to be changed without the warm or cold start, but internally marks the drive as Read-Only. The status of the drive is subsequently changed to Read-Write if a warm or cold start occurs. On issuing this message, CP/M waits for input from the console. An automatic warm start takes place following any input.

6.9 Reserved Locations in Page Zero

Main memory page zero, between locations 0H and 0FFH, contains several segments of code and data that are used during CP/M processing. The code and data areas are given in the following table.

Table 6-6. Reserved Locations in Page Zero

<u>Locations</u>	<u>Contents</u>
0000H-0002H	Contains a jump instruction to the warm start entry location 4A03H+b. This allows a simple programmed restart (JMP 0000H) or manual restart from the front panel.
0003H-0003H	Contains the Intel standard IOBYTE is optionally included in the user's CBIOS (refer to Section 6.6).
0004H-0004H	Current default drive number (0=A,...,15=P).
0005H-0007H	Contains a jump instruction to the BDOS and serves two purposes: JMP 0005H provides the primary entry point to the BDOS, as described in Section 5, and LHLD 0006H brings the address field of the instruction to the HL register pair. This value is the lowest address in memory used by CP/M, assuming the CCP is being overlaid. The DDT program changes the address field to reflect the reduced memory size in debug mode.
0008H-0027H	Interrupt locations I through 5 not used.
0030H-0037H	Interrupt location 6 (not currently used) is reserved.
0038H-003AH	Restart 7; contains a jump instruction into the DDT or SID program when running in debug mode for programmed breakpoints, but is not otherwise used by CP/M.
003BH-003FH	Not currently used; reserved.

Table 6-6. (continued)

<u>Locations</u>	<u>Contents</u>
0040H-004FH	A 16-byte area reserved for scratch by CBIOS, but is not used for any purpose in the distribution version of CP/M.
0050H-005BH	Not currently used; reserved.
005CH-007CH	Default File Control Block produced for a transient program by the CCP.
007DH-007FH	Optional default random record position.
0080H-00FFH	Default 128-byte disk buffer, also filled with the command line when a transient is loaded under the CCP.

This information is set up for normal operation under the CP/M system, but can be overwritten by a transient program if the BDOS facilities are not required by the transient.

If, for example, a particular program performs only simple I/O and must begin execution at location 0, it can first be loaded into the TPA, using normal CP/M facilities, with a small memory move program that gets control when loaded. The memory move program must get control from location 0100H, which is the assumed beginning of all transient programs. The move program can then proceed to the entire memory image down to location 0 and pass control to the starting address of the memory load.

If the BIOS is overwritten or if location 0, containing the warm start entry point, is overwritten, the operator must bring the CP/M system back into memory with a cold start sequence.

Appendix H Glossary

address: Number representing the location of a byte in memory. Within CP/M there are two kinds of addresses: logical and physical. A physical address refers to an absolute and unique location within the computer's memory space. A logical address refers to the offset or displacement of a byte in relation to a base location. A standard CP/M program is loaded at address 0100H, the base value; the first instruction of a program has a physical address of 0100H and a relative address or offset of 0H.

allocation vector (ALV): An allocation vector is maintained in the BIOS for each logged-in disk drive. A vector consists of a string of bits, one for each block on the drive. The bit corresponding to a particular block is set to one when the block has been allocated and to zero otherwise. The first two bytes of this vector are initialized with the bytes AL0 and AL1 on, thus allocating the directory blocks. CP/M Function 27 returns the allocation vector address.

AL0, AL1: Two bytes in the disk parameter block that reserve data blocks for the directory. These two bytes are copied into the first two bytes of the allocation vector when a drive is logged in. See allocation vector.

ALV: See allocation vector.

ambiguous filename: Filename that contains either of the CP/M wildcard characters, ? or *, in the primary filename, filetype, or both. When you replace characters in a filename with these wildcard characters, you create an ambiguous filename and can easily reference more than one CP/M file in a single command line.

American Standard Code for Information Interchange: See ASCII.

applications program: Program designed to solve a specific problem. Typical applications programs are business accounting packages, word processing (editing) programs and mailing list programs.

archive attribute: File attribute controlled by the high-order bit of the t3 byte (FCB + 11) in a directory element. This attribute is set if the file has been archived.

argument: Symbol, usually a letter, indicating a place into which you can substitute a number, letter, or name to give an appropriate meaning to the formula in question.

ASCII: American Standard Code for Information Interchange. ASCII is a standard set of seven-bit numeric character codes used to represent characters in memory. Each character requires one byte of memory with the high-order bit usually set to zero. Characters can be numbers, letters, and symbols. An ASCII file can be intelligibly displayed on the video screen or printed on paper.

assembler: Program that translates assembly language into the binary machine code. Assembly language is simply a set of mnemonics used to designate the instruction set of the CPU. See ASM in Section 3 of this manual.

back-up: Copy of a disk or file made for safekeeping, or the creation of the duplicate disk or file.

Basic Disk Operating System: See BDOS.

BDOS: Basic Disk Operating System. The BDOS module of the CP/M operating system provides an interface for a user program to the operating. This interface is in the form of a set of function calls which may be made to the BDOS through calls to location 0005H in page zero. The user program specifies the number of the desired function in register C. User programs running under CP/M should use BDOS functions for all I/O operations to remain compatible with other CP/M systems and future releases. The BDOS normally resides in high memory directly below the BIOS.

bias: Address value which when added to the origin address of your BIOS module produces 1F80H, the address of the BIOS module in the MOVCPM image. There is also a bias value that when added to the BOOT module origin produces 0900H, the address of the BOOT module in the MOVCPM image. You must use these bias values with the R command under DDT or SID" when you patch a CP/M system. If you do not, the patched system may fall to function.

binary: Base 2 numbering system. A binary digit can have one of two values: 0 or 1. Binary numbers are used in computers because the hardware can most easily exhibit two states: off and on. Generally, a bit in memory represents one binary digit.

Basic Input/Output System: See BIOS.

BIOS: Basic Input/Output System. The BIOS is the only hardware-dependent module of the CP/M system. It provides the BDOS with a set of primitive I/O operations. The BIOS is an assembly language module usually written by the user, hardware manufacturer, or independent software vendor, and is the key to CP/M's portability. The BIOS interfaces the CP/M system to its hardware environment through a standardized jump table at the front of the BIOS routine and through a set of disk parameter tables which define the disk environment. Thus, the BIOS provides CP/M with a completely table-driven I/O system.

BIOS base: Lowest address of the BIOS module in memory, that by definition must be the first entry point in the BIOS jump table.

bit: Switch in memory that can be set to on (1) or off (0). Bits are grouped into bytes, eight bits to a byte, which is the smallest directly addressable unit in an Intel 8080 or Zilog Z80. By common convention, the bits in a byte are numbered from right, 0 for the low-order bit, to left, 7

for the high-order bit. Bit values are often represented in hexadecimal notation by grouping the bits from the low-order bit in groups of four. Each group of four bits can have a value from 0 to 15 and thus can easily be represented by one hexadecimal digit.

BLM: See block mask.

block: Basic unit of disk space allocation. Each disk drive has a fixed block size (BLS) defined in its disk parameter block in the BIOS. A block can consist of 1K, 2K, 4K, 8K, or 16K consecutive bytes. Blocks are numbered relative to zero so that each block is unique and has a byte displacement in a file equal to the block number times the block size.

block mask (BLM): Byte value in the disk parameter block at $DPB + 3$. The block mask is always one less than the number of 128 byte sectors that are in one block. Note that $BLM = (2^{**} BSH) - 1$.

block shift (BSH): Byte parameter in the disk parameter block at $DPB + 2$. Block shift and block mask (BLM) values are determined by the block size (BLS). Note that $BLM = (2^{**} BSH) - 1$.

blocking & deblocking algorithm: In some disk subsystems the disk sector size is larger than 128 bytes, usually 256, 512, 1024, or 2048 bytes. When the host sector size is larger than 128 bytes, host sectors must be buffered in memory and the 128-byte CP/M sectors must be blocked and deblocked by adding an additional module, the blocking and deblocking algorithm, between the BIOS disk I/O routines and the actual disk I/O. The host sector size must be an even multiple of 128 bytes for the algorithm to work correctly. The blocking and deblocking algorithm allows the BDOS and BIOS to function exactly as if the entire disk consisted only of 128-byte sectors, as in the standard CP/M installation.

BLS: Block size in bytes. See block.

boot: Process of loading an operating system into memory. A boot program is a small piece of code that is automatically executed when you power-up or reset your computer. The boot program loads the rest of the operating system into memory in a manner similar to a person pulling himself up by his own bootstraps. This process is sometimes called a cold boot or cold start. Bootstrap procedures vary from system to system. The boot program must be customized for the memory size and hardware environment that the operating system manages. Typically, the boot resides on the first sector of the system tracks on your system disk. When executed, the boot loads the remaining sectors of the system tracks into high memory at the location for which the CP/M system has been configured. Finally, the boot transfers execution to the boot entry point in the BIOS jump table so that the system can initialize itself. In this case, the boot program should be placed at 900H in the SYSGEN image. Alternatively, the boot program may be located in ROM.

bootstrap: See boot.

BSH: See block shift.

BTREE: General purpose file access method that has become the standard organization for indexes in large data base systems. BTREE provides near optimum performance over the full range of file operations, such as insertion, deletion, search, and search next.

buffer: Area of memory that temporarily stores data during the transfer of information.

built-in commands: Commands that permanently reside in memory. They respond quickly because they are not accessed from a disk.

byte: Unit of memory or disk storage containing eight bits. A byte can represent a binary number between 0 and 255, and is the smallest unit of memory that can be addressed directly in 8-bit CPUs such as the Intel 8080 or Zilog Z80.

CCP: Console Command Processor. The CCP is a module of the CP/M operating system. It is loaded directly below the BDOS module and interprets and executes commands typed by the console user. Usually these commands are programs that the CCP loads and calls. Upon completion, a command program may return control to the CCP if it has not overwritten it. If it has, the program can reload the CCP into memory by a warm boot operation initiated by either a jump to zero, BDOS system reset (Function 0), or a cold boot. Except for its location in high memory, the CCP works like any other standard CP/M program; that is, it makes only BDOS function calls for its I/O operations.

CCP base: Lowest address of the CCP module in memory. This term sometimes refers to the base of the CP/M system in memory, as the CCP is normally the lowest CP/M module in high memory.

checksum vector (CSV): Contiguous data area in the BIOS, with one byte for each directory sector to be checked, that is, CKS bytes. See CKS. A checksum vector is initialized and maintained for each logged-in drive. Each directory access by the system results in a checksum calculation that is compared with the one in the checksum vector. If there is a discrepancy, the drive is set to Read-Only status. This feature prevents the user from inadvertently switching disks without logging in the new disk. If the new disk is not logged-in, it is treated the same as the old one, and data on it might be destroyed if writing is done.

CKS: Number of directory records to be checked summed on directory accesses. This is a parameter in the disk parameter block located in the BIOS. If the value of CKS is zero, then no directory records are checked. CKS is also a parameter in the diskdef macro library, where it is the actual number of directory elements to be checked rather than the number of directory records.

cold boot: See boot. Cold boot also refers to a jump to the boot entry. point in the BIOS jump table.

COM: Filetype for a CP/M command file. See command file.

command: CP/M command line. In general, a CP/M command line has three parts: the command keyword, command tail, and a carriage return. To execute a command, enter a CP/M command line directly after the CP/M prompt at the console and press the carriage return or enter key.

command file: Executable program file of filetype COM. A command file is a machine language object module ready to be loaded and executed at the absolute address of 0100H. To execute a command file, enter its primary filename as the command keyword in a CP/M command line.

command keyword: Name that identifies a CP/M command, usually the primary filename of a file of type COM, or a built-in command. The command keyword precedes the command tail and the carriage return in the command line.

command syntax: Statement that defines the correct way to enter a command. The correct structure generally includes the command keyword, the command tail, and a carriage return. A syntax line usually contains symbols that you should replace with actual values when you enter the command.

command tail: Part of a command that follows the command keyword in the command line. The command tail can include a drive specification, a filename and filetype, and options or parameters. Some commands do not require a command tail.

CON: Mnemonic that represents the CP/M console device. For example, the CP/M command PIP CON:=TEST.SUB displays the file TEST.SUB on the console device. The explanation of the STAT command tells how to assign the logical device CON: to various physical devices. See console.

concatenate: Name of the PIP operation that copies two or more separate files into one new file in the specified sequence.

concurrency: Execution of two processes or operations simultaneously.

CONIN: BIOS entry point to a routine that reads a character from the console device.

CONOUT: BIOS entry point to a routine that sends a character to the console device.

console: Primary input/output device. The console consists of a listing device, such as a screen or teletype, and a keyboard through which the user communicates with the operating system or applications program.

Console Command Processor: See CCP.

CONST: BIOS entry point to a routine that returns the status of the console device.

control character: Nonprinting character combination. CP/M interprets some control characters as simple commands such as line editing functions. To enter a control character, hold down the CONTROL key and strike the specified character key.

Control Program for Microcomputers: See CP/M.

CP/M: Control Program for Microcomputers. An operating system that manages computer resources and provides a standard systems interface to software written for a large variety of microprocessor-based computer systems.

CP/M 1.4 compatibility: For a CP/M 2 system to be able to read correctly single-density disks produced under a CP/M 1.4 system, the extent mask must be zero and the block size 1K. This is because under CP/M 2 an FCB may contain more than one extent. The number of extents that may be contained by an FCB is EXM + 1. The issue of CP/M 1.4 compatibility also concerns random file I/O. To perform random file I/O under CP/M 1.4, you must maintain an FCB for each extent of the file. This scheme is upward compatible with CP/M 2 for files not exceeding 512K bytes, the largest file size supported under CP/M 1.4. If you wish to implement random I/O for files larger than 512K bytes under CP/M 2, you must use the random read and random write functions, BDOS functions 33, 34, and 36. In this case, only one FCB is used, and if CP/M 1.4 compatibility is required, the program must use the return version number function, BDOS Function 12, to determine which method to employ.

CP/M prompt: Characters that indicate that CP/M is ready to execute your next command. The CP/M prompt consists of an upper-case letter, A-P, followed by a > character; for example, A>. The letter designates which drive is currently logged in as the default drive. CP/M will search this drive for the command file specified, unless the command is a built-in command or prefaced by a select drive command: for example, B:STAT.

CP/NET: Digital Research network operating system enabling microcomputers to obtain access to common resources via a network. CP/NET consists of MP/M masters and CP/M slaves with a network interface between them.

CSV: See checksum vector.

cursor: One-character symbol that can appear anywhere on the console screen. The cursor indicates the position where the next keystroke at the console will have an effect.

data file: File containing information that will be processed by a program.

deblocking: See blocking & deblocking algorithm.

default: Currently selected disk drive and user number. Any command that does not specify a disk drive or a user number references the default disk drive and user number. When CP/M is first invoked, the default disk drive is drive A, and the default user number is 0.

default buffer: Default 128-byte buffer maintained at 0080H in page zero. When the CCP loads a COM file, this buffer is initialized to the command tail; that is, any characters typed after the COM file name are loaded into the buffer. The first byte at 0080H contains the length of the command tail, while the command tail itself begins at 0081H. The command tail is terminated by a byte containing a binary zero value. The I command under DDT and SID initializes this buffer in the same way as the CCP.

default FCB: Two default FCBs are maintained by the CCP at 005CH and 006CH in page zero. The first default FCB is initialized from the first delimited field in the command tail. The second default FCB is initialized from the next field in the command tail.

delimiter: Special characters that separate different items in a command line; for example, a colon separates the drive specification from the filename. The CCP recognizes the following characters as delimiters: . : = ; < > - , blank, and carriage return. Several CP/M commands also treat the following as delimiter characters: , [] () \$. It is advisable to avoid the use of delimiter characters and lower-case characters in CP/M filenames.

DIR: Parameter in the diskdef macro library that specifies the number of directory elements on the drive.

DIR attribute: File attribute. A file with the DIR attribute can be displayed by a DIR command. The file can be accessed from the default user number and drive only.

DIRBUF: 128-byte scratchpad area for directory operations, usually located at the end of the BIOS. DIRBUF is used by the BDOS during its directory operations. DIRBUF also refers to the two-byte address of this scratchpad buffer in the disk parameter header at DPbase + 8 bytes.

directory: Portion of a disk that contains entries for each file on the disk. In response to the DIR command, CP/M displays the filenames stored in the directory. The directory also contains the locations of the blocks allocated to the files. Each file directory element is in the form of a 32-byte FCB, although one file can have several elements, depending on its size. The maximum number of directory elements supported is specified by the drive's disk parameter block value for DRM.

directory element: Data structure. Each file on a disk has one or more 32-byte directory elements associated with it. There are four directory elements per directory sector. Directory elements can also be referred to as directory FCBs.

directory entry: File entry displayed by the DIR command. Sometimes this term refers to a physical directory element.

disk, diskette: Magnetic media used for mass storage in a computer system. Programs and data are recorded on the disk in the same way music can be recorded on cassette tape. The CP/M operating system must be initially loaded from disk when the computer is turned on. Diskette refers to smaller capacity removable floppy diskettes, while disk may refer to either a diskette,

removable cartridge disk, or fixed hard disk. Hard disk capacities range from five to several hundred megabytes of storage.

diskdef macro library: Library of code that when used with MAC, the Digital Research macro assembler, creates disk definition tables such as the DPB and DPH automatically.

disk drive: Peripheral device that reads and writes information on disk. CP/M assigns a letter to each drive under its control. For example, CP/M may refer to the drives in a four-drive system as A, B, C, and D.

disk parameter block (DPB): Data structure referenced by one or more disk parameter headers. The disk parameter block defines disk characteristics in the fields listed below:

- SPT is the total number of sectors per track.
- BSH is the data allocation block shift factor.
- BLM is the data allocation block mask.
- EXM is the extent mask determined by BLS and DSM.
- DSM is the maximum data block number.
- DRM is the maximum number of directory entries-1.
- AL0 reserves directory blocks.
- AL1 reserves directory blocks.
- CKS is the number of directory sectors check summed.
- OFF is the number of reserved system tracks.

The address of the disk parameter block is located in the disk parameter header at DPbase + 0AH. CP/M Function 31 returns the DPB address. Drives with the same characteristics can use the same disk parameter header, and thus the same DPB. However, drives with different characteristics must each have their own disk parameter header and disk parameter blocks. When the BDOS calls the SELDSK entry point in the BIOS, SELDSK must return the address of the drive's disk parameter header in register HL.

disk parameter header (DPH): Data structure that contains information about the disk drive and provides a scratchpad area for certain BDOS operations. The disk parameter header contains six bytes of scratchpad area for the BDOS, and the following five 2-byte parameters:

- XLT is the sector translation table address.
- DIRBUF is the directory buffer address.
- DPB is the disk parameter block address.
- CSV is the checksum vector address.
- ALV is the allocation vector address.

Given n disk drives, the disk parameter headers are arranged in a table whose first row of 16 bytes corresponds to drive 0, with the last row corresponding to drive n - 1.

DKS: Parameter in the diskdef macro library specifying the number of data blocks on the drive.

DMA: Direct Memory Access. DMA is a method of transferring data from the disk into memory directly. In a CP/M system, the BDOS calls the BIOS entry point READ to read a sector from the disk into the currently selected DMA address. The DMA address must be the address of a 128-byte buffer in memory, either the default buffer at 0080H in page zero, or a user-assigned buffer in the TPA. Similarly, the BDOS calls the BIOS entry point WRITE to write the record at the current DMA address to the disk.

DN: Parameter in the diskdef macro library specifying the logical drive number.

DPB: See disk parameter block.

DPH: See disk parameter header.

DRM: 2-byte parameter in the disk parameter block at DPB + 7. DRM is one less than the total number of directory entries allowed for the drive. This value is related to DPB bytes AL0 and AL1, which allocates up to 16 blocks for directory entries.

DSM: 2-byte parameter of the disk parameter block at DPB + 5. DSM is the maximum data block number supported by the drive. The product BLS times (DSM + 1) is the total number of bytes held by the drive. This must not exceed the capacity of the physical disk less the reserved system tracks.

editor: Utility program that creates and modifies text files. An editor can be used for creation of documents or creation of code for computer programs. The CP/M editor is invoked by typing the command ED next to the system prompt on the console.

EX: Extent number field in an FCB. See extent.

executable: Ready to be run by the computer. Executable code is a series of instructions that can be carried out by the computer. For example, the computer cannot execute names and addresses, but it can execute a program that prints all those names and addresses on mailing labels.

execute a program: Start the processing of executable code.

EXM: See extent mask.

extent: 16K consecutive bytes in a file. Extents are numbered from 0 to 31. One extent can contain 1, 2, 4, 8, or 16 blocks. EX is the extent number field of an FCB and is a one-byte field at FCB + 12, where FCB labels the first byte in the FCB. Depending on the block size (BLS) and the maximum data block number (DSM), an FCB can contain 1, 2, 4, 8, or 16 extents. The EX field is normally set to 0 by the user but contains the current extent number during file I/O. The term FCB folding describes FCBs containing more than one extent. In CP/M version 1.4, each FCB contained only one extent. Users attempting to perform random record I/O and maintain

CP/M 1.4 compatibility should be aware of the implications of this difference. See CP/M 1.4 compatibility.

extent mask (EXM): A byte parameter in the disk parameter block located at DPB + 3. The value of EXM is determined by the block size (BLS) and whether the maximum data block number (DSM) exceeds 255. There are EXM + 1 extents per directory FCB.

FCB: See File Control Block.

file: Collection of characters, instructions, or data that can be referenced by a unique identifier. Files are usually stored on various types of media, such as disk, or magnetic tape. A CP/M file is identified by a file specification and resides on disk as a collection of from zero to 65,536 records. Each record is 128 bytes and can contain either binary or ASCII data. Binary files contain bytes of data that can vary in value from 0H to 0FFH. ASCII files contain sequences of character codes delineated by a carriage return and line-feed combination; normally byte values range from 0H to 7FH. The directory maps the file as a series of physical blocks. Although files are defined as a sequence of consecutive logical records, these records can not reside in consecutive sectors on the disk. See also block, directory, extent, record, and sector.

File Control Block (FCB): Structure used for accessing files on disk. Contains the drive, filename, filetype, and other information describing a file to be accessed or created on the disk. A file control block consists of 36 consecutive bytes specified by the user for file I/O functions. FCB can also refer to a directory element in the directory portion of the allocated disk space. These contain the same first 32 bytes of the FCB, but lack the current record and random record number bytes.

filename: Name assigned to a file. A filename can include a primary filename of one to eight characters; a filetype of zero to three characters. A period separates the primary filename from the filetype.

file specification: Unique file identifier. A complete CP/M file specification includes a disk drive specification followed by a colon, d:, a primary filename of one to eight characters, a period, and a filetype of zero to three characters. For example, b:example.tex is a complete CP/M file specification.

filetype: Extension to a filename. A filetype can be from zero to three characters and must be separated from the primary filename by a period. A filetype can tell something about the file. Some programs require that files to be processed have specific filetypes.

floppy disk: Flexible magnetic disk used to store information. Floppy disks come in 5 1/4- and 8-inch diameters.

FSC: Parameter in the diskdef macro library specifying the first physical sector number. This parameter is used to determine SPT and build XLT.

hard disk: Rigid, platter-like, magnetic disk sealed in a container. A hard disk stores more information than a floppy disk.

hardware: Physical components of a computer.

hexadecimal notation: Notation for base 16 values using the decimal digits and letters A, B, C, D, E, and F to represent the 16 digits. Hexadecimal notation is often used to refer to binary numbers. A binary number can be easily expressed as a hexadecimal value by taking the bits in groups of 4, starting with the least significant bit, and expressing each group as a hexadecimal digit, 0-F. Thus the bit value 1011 becomes 0BH and 10110101 becomes 0B5H.

hex file: ASCII-printable representation of a command, machine language, file.

hex file format: Absolute output of ASM and MAC for the Intel 8080 is a hex format file, containing a sequence of absolute records that give a load address and byte values to be stored, starting at the load address.

HOME: BIOS entry point which sets the disk head of the currently selected drive to the track zero position.

host: Physical characteristics of a hard disk drive in a system using the blocking and deblocking algorithm. The term, host, helps distinguish physical hardware characteristics from CP/M's logical characteristics. For example, CP/M sectors are always 128 bytes, although the host sector size can be a multiple of 128 bytes.

input: Data going into the computer, usually from an operator typing at the terminal or by a program reading from the disk.

input/output: See I/O.

interface: Object that allows two independent systems to communicate with each other, as an interface between hardware and software in a microcomputer.

I/O: Abbreviation for input/output. Usually refers to input/output operations or routines handling the input and output of data in the computer system.

IOBYTE: A one-byte field in page zero, currently at location 0003H, that can support a logical-to-physical device mapping for I/O. However, its implementation in your BIOS is purely optional and might or might not be supported in a given CP/M system. The IOBYTE is easily set using the command:

```
STAT <logical device> = <physical device>
```

The CP/M logical devices are CON:, RDR:, PUN:, and LST::; each of these can be assigned to one of four physical devices. The IOBYTE can be initialized by the BOOT entry point of the

BIOS and interpreted by the BIOS I/O entry points CONST, CONIN, CONOUT, LIST, PUNCH, and READER. Depending on the setting of the IOBYTE, different I/O drivers can be selected by the BIOS. For example, setting LST:=TTY: might cause LIST output to be directed to a serial port, while setting LST:=LPT: causes LIST output to be directed to a parallel port.

K: Abbreviation for kilobyte. See kilobyte.

keyword: See command keyword.

kilobyte (K): 1024 bytes or 0400H bytes of memory. This is a standard unit of memory. For example, the Intel 8080 supports up to 64K of memory address space or 65,536 bytes. 1024 kilobytes equal one megabyte, or over one million bytes.

linker: Utility program used to combine relocatable object modules into an absolute file ready for execution. For example, LINK-80(TM) creates either a COM or PRL file from relocatable REL files, such as those produced by PL/1-80(TM).

LIST: A BIOS entry point to a routine that sends a character to the list device, usually a printer.

list device: Device such as a printer onto which data can be listed or printed.

LISTST: BIOS entry point to a routine that returns the ready status of the list device.

loader: Utility program that brings an absolute program image into memory ready for execution under the operating system, or a utility used to make such an image. For example, LOAD prepares an absolute COM file from the assembler hex file output that is ready to be executed under CP/M.

logged in: Made known to the operating system, in reference to drives. A drive is logged in when it is selected by the user or an executing process. It remains selected or logged in until you change disks in a floppy disk drive or enter CTRL-C at the command level, or until a BDOS Function 0 is executed.

logical: Representation of something that might or might not be the same in its actual physical form. For example, a hard disk can occupy one physical drive, yet you can divide the available storage on it to appear to the user as if it were in several different drives. These apparent drives are the logical drives.

logical sector: See sector.

logical-to-physical sector translation table: See XLT.

LSC: Diskdef macro library parameter specifying the last physical sector number.

LST: Logical CP/M list device, usually a printer. The CP/M list device is an output-only device referenced through the LIST and LISTST entry points of the BIOS. The STAT command allows assignment of LST: to one of the physical devices: TTY:, CRT:, LPT:, or UL1:, provided these devices and the IOBYTE are implemented in the LIST and LISTST entry points of your CP/M BIOS module. The CP/NET command NETWORK allows assignment of LST: to a list device on a network master. For example, PIP LST:=TEST.SUB prints the file TEST.SUB on the list device.

macro assembler: Assembler code translator providing macro processing facilities. Macro definitions allow groups of instructions to be stored and substituted in the source program as the macro names are encountered. Definitions and invocations can be nested and macro parameters can be formed to pass arbitrary strings of text to a specific macro for substitution during expansion.

megabyte: Over one million bytes; 1024 kilobytes. See byte, and kilobyte.

microprocessor: Silicon chip that is the central processing unit (CPU) of the microcomputer. The Intel 8080 and the Zilog Z80 are microprocessors commonly used in CP/M systems.

MOVCPM image: Memory image of the CP/M system created by MOVCPM. This image can be saved as a disk file using the SAVE command or placed on the system tracks using the SYSGEN command without specifying a source drive. This image varies, depending on the presence of a one-sector or two-sector boot. If the boot is less than 128 bytes (one sector), the boot begins at 0900H, the CP/M system at 0980H, and the BIOS at 1F80H. Otherwise, the boot is at 0900H, the CP/M system at 1000H, and the BIOS at 2000H. In a CP/M 1.4 system with a one-sector boot, the addresses are the same as for the CP/M 2 system-except that the BIOS begins at 1E80H instead of 1F80H.

MP/M: Multi-Programming Monitor control program. A microcomputer operating system supporting multi-terminal access with multi-programming at each terminal.

multi-programming: The capability of initiating and executing more than one program at a time. These programs, usually called processes, are time-shared, each receiving a slice of CPU time on a round-robin basis. See concurrency.

nibble: One half of a byte, usually the high-order or low-order 4 bits in a byte.

OFF: Two-byte parameter in the disk parameter block at DPB + 13 bytes. This value specifies the number of reserved system tracks. The disk directory begins in the first sector of track OFF.

OFS: Diskdef macro library parameter specifying the number of reserved system tracks. See OFF.

operating system: Collection of programs that supervises the execution of other programs and the management of computer resources. An operating system provides an orderly input/output

environment between the computer and its peripheral devices. It enables user-written programs to execute safely. An operating system standardizes the use of computer resources for the programs running under it.

option: One of many parameters that can be part of a command tail. Use options to specify additional conditions for a command's execution.

output: Data that is sent to the console, disk, or printer.

page: 256 consecutive bytes in memory beginning on a page boundary, whose base address is a multiple of 256 (100H) bytes. In hex notation, pages always begin at an address with a least significant byte of zero.

page relocatable program: See PRL.

page zero: Memory region between 0000H and 0100H used to hold critical system parameters. Page zero functions primarily as an interface region between user programs and the CP/M BDOS module. Note that in non-standard systems this region is the base page of the system and represents the first 256 bytes of memory used by the CP/M system and user programs running under it.

parameter: Value in the command tail that provides additional information for the command. Technically, a parameter is a required element of a command.

peripheral devices: Devices external to the CPU. For example, terminals, printers, and disk drives are common peripheral devices that are not part of the processor but are used in conjunction with it.

physical: Characteristic of computer components, generally hardware, that actually exist. In programs, physical components can be represented by logical components.

primary filename: First 8 characters of a filename. The primary filename is a unique name that helps the user identify the file contents. A primary filename contains one to eight characters and can include any letter or number and some special characters. The primary filename follows the optional drive specification and precedes the optional filetype.

PRL: Page relocatable program. A page relocatable program is stored on disk with a PRL filetype. Page relocatable programs are easily relocated to any page boundary and thus are suitable for execution in a nonbanked MP/M system.

program: Series of coded Instructions that performs specific tasks when executed by a computer. A program can be written in a processor-specific language or a high-level language that can be implemented on a number of different processors.

prompt: Any characters displayed on the video screen to help the user decide what the next appropriate action is. A system prompt is a special prompt displayed by the operating system. The alphabetic character indicates the default drive. Some applications programs have their own special prompts. See CP/M prompt.

PUN: Logical CP/M punch device. The punch device is an output-only device accessed through the PUNCH entry point of the BIOS. In certain implementations, PUN: can be a serial device such as a modem.

PUNCH: BIOS entry point to a routine that sends a character to the punch device.

RDR: Logical CP/M reader device. The reader device is an input-only device accessed through the READER entry point in the BIOS. See PUN:.

READ: Entry point in the BIOS to a routine that reads 128 bytes from the currently selected drive, track, and sector into the current DMA address.

READER: Entry point to a routine in the BIOS that reads the next character from the currently assigned reader device.

Read-Only (R/O): Attribute that can be assigned to a disk file or a disk drive. When assigned to a file, the Read-Only attribute allows you to read from that file but not write to it. When assigned to a drive, the Read-Only attribute allows you to read any file on the disk, but prevents you from adding a new file, erasing or changing a file, renaming a file, or writing on the disk. The STAT command can set a file or a drive to Read-Only. Every file and drive is either Read-Only or Read-Write. The default setting for drives and files is Read-Write, but an error in resetting the disk or changing media automatically sets the drive to Read-Only until the error is corrected. See also ROM.

Read-Write (R/W): Attribute that can be assigned to a disk file or a disk drive. The Read-Write attribute allows you to read from and write to a specific Read-Write file or to any file on a disk that is in a drive set to Read-Write. A file or drive can be set to either Read-Only or Read-Write.

record: Group of bytes in a file. A physical record consists of 128 bytes and is the basic unit of data transfer between the operating system and the application program. A logical record might vary in length and is used to represent a unit of information. Two 64-byte employee records can be stored in one 128-byte physical record. Records are grouped together to form a file.

recursive procedure: Code that can call itself during execution.

reentrant procedure: Code that can be called by one process while another is already executing it. Thus, reentrant code can be shared between different users. Reentrant procedures must not be self-modifying; that is, they must be pure code and not contain data. The data for reentrant procedures can be kept in a separate data area or placed on the stack.

restart (RST): One-byte call instruction usually used during interrupt sequences and for debugger break pointing. There are eight restart locations, RST 0 through RST 7, whose addresses are given by the product of 8 times the restart number.

R/O: See Read-Only.

ROM: Read-Only memory. This memory can be read but not written and so is suitable for code and preinitialized data areas only.

RST: See restart.

R/W: See Read-Write.

sector: In a CP/M system, a sector is always 128 consecutive bytes. A sector is the basic unit of data read and written on the disk by the BIOS. A sector can be one 128-byte record in a file or a sector of the directory. The BDOS always requests a logical sector number between 0 and (SPT-1). This is typically translated into a physical sector by the BIOS entry point SECTRAN. In some disk subsystems, the disk sector size is larger than 128 bytes, usually a power of two, such as 256, 512, 1024, or 2048 bytes. These disk sectors are always referred to as host sectors in CP/M documentation and should not be confused with other references to sectors, in which cases the CP/M 128-byte sectors should be assumed. When the host sector size is larger than 128 bytes, host sectors must be buffered in memory and the 128-byte CP/M sectors must be blocked and deblocked from them. This can be done by adding an additional module, the blocking and deblocking algorithm, between the BIOS disk I/O routines and the actual disk I/O.

sectors per track (SPT): A two-byte parameter in the disk parameter block at DPB + 0. The BDOS makes calls to the BIOS entry point SECTRAN with logical sector numbers ranging between 0 and (SPT - 1) in register BC.

SECTRAN: Entry point to a routine in the BIOS that performs logical-to-physical sector translation for the BDOS.

SELDSK: Entry point to a routine in the BIOS that sets the currently selected drive.

SETDMA: Entry point to a routine in the BIOS that sets the currently selected DMA address. The DMA address is the address of a 128-byte buffer region in memory that is used to transfer data to and from the disk in subsequent reads and writes.

SETSEC: Entry point to a routine in the BIOS that sets the currently selected sector.

SETTRK: Entry point to a routine in the BIOS that sets the currently selected track.

skew factor: Factor that defines the logical-to-physical sector number translation in XLT. Logical sector numbers are used by the BDOS and range between 0 and (SPT - 1). Data is written in consecutive logical 128-byte sectors grouped in data blocks. The number of sectors

per block is given by BLS/128. Physical sectors on the disk media are also numbered consecutively. If the physical sector size is also 128 bytes, a one-to-one relationship exists between logical and physical sectors. The logical-to-physical translation table (XLT) maps this relationship, and a skew factor is typically used in generating the table entries. For instance, if the skew factor is 6, XLT will be:

```
Logical: 0 1 2 3 4 5 6 ..... 25  
Physical: 1 7 13 19 25 5 11 ..... 22
```

The skew factor allows time for program processing without missing the next sector. Otherwise, the system must wait for an entire disk revolution before reading the next logical sector. The skew factor can be varied, depending on hardware speed and application processing overhead. Note that no sector translation is done when the physical sectors are larger than 128 bytes, as sector deblocking is done in this case. See also sector, SKF, and XLT.

SKF: A diskdef macro library parameter specifying the skew factor to be used in building XLT. If SKF is zero, no translation table is generated and the XLT byte in the DPH will be 0000H.

software: Programs that contain machine-readable instructions, as opposed to hard-ware, which is the actual physical components of a computer.

source file: ASCII text file usually created with an editor that is an input file to a system program, such as a language translator or text formatter.

SP: Stack pointer. See stack.

spooling: Process of accumulating printer output in a file while the printer is busy. The file is printed when the printer becomes free; a program does not have to wait for the slow printing process.

SPT: See sectors per track.

stack: Reserved area of memory where the processor saves the return address when a call instruction is received. When a return instruction is encountered, the processor restores the current address on the stack to the program counter. Data such as the contents of the registers can also be saved on the stack. The push instruction places data on the stack and the pop instruction removes it. An item is pushed onto the stack by decrementing the stack pointer (SP) by 2 and writing the item at the SP address. In other words, the stack grows downward in memory.

syntax: Format for entering a given command.

SYS: See system attribute.

SYSGEN image: Memory image of the CP/M system created by SYSGEN when a destination drive is not specified. This is the same as the MOVCPM image that can be read by SYSGEN if a source drive is not specified. See MOVCPM image.

system attribute (SYS): File attribute. You can give a file the system attribute by using the SYS option in the STAT command or by using the set file attributes function, BDOS Function 12. A file with the SYS attribute is not displayed in response to a DIR command. If you give a file with user number 0 the SYS attribute, you can read and execute that file from any user number on the same drive. Use this feature to make your commonly used programs available under any user number.

system prompt: Symbol displayed by the operating system indicating that the system is ready to receive input. See prompt and CP/M prompt.

system tracks: Tracks reserved on the disk for the CP/M system. The number of system tracks is specified by the parameter OFF in the disk parameter block (DPB). The system tracks for a drive always precede its data tracks. The command SYSGEN copies the CP/M system from the system tracks to memory, and vice versa. The standard SYSGEN utility copies 26 sectors from track 0 and 26 sectors from track 1. When the system tracks contain additional sectors or tracks to be copied, a customized SYSGEN must be used.

terminal: See console.

TPA: Transient Program Area. Area in memory where user programs run and store data. This area is a region of memory beginning at 0100H and extending to the base of the CP/M system in high memory. The first module of the CP/M system is the CCP, which can be overwritten by a user program. If so, the TPA is extended to the base of the CP/M BDOS module. If the CCP is overwritten, the user program must terminate with either a system reset (Function 0) call or a jump to location zero in page zero. The address of the base of the CP/M BDOS is stored in location 0006H in page zero least significant byte first.

track: Data on the disk media is accessed by combination of track and sector numbers. Tracks form concentric rings on the disk; the standard IBM single-density disks have 77 tracks. Each track consists of a fixed number of numbered sectors. Tracks are numbered from zero to one less than the number of tracks on the disk.

Transient Program Area: See TPA.

upward compatible: Term meaning that a program created for the previously released operating system, or compiler, runs under the newly released version of the same operating system.

USER: Term used in CP/M and MP/M systems to distinguish distinct regions of the directory.

user number: Number assigned to files in the disk directory so that different users need only deal with their own files and have their own directories, even though they are all working from the same disk. In CP/M, files can be divided into 16 user groups.

utility: Tool. Program that enables the user to perform certain operations, such as copying files, erasing files, and editing files. The utilities are created for the convenience of programmers and users.

vector: Location in memory. An entry point into the operating system used for making system calls or interrupt handling.

warm start: Program termination by a jump to the warm start vector at location 0000H, a system reset (BDOS Function 0), or a CTRL-C typed at the keyboard. A warm start reinitializes the disk subsystem and returns control to the CP/M operating system at the CCP level. The warm start vector is simply a jump to the WBOOT entry point in the BIOS.

WBOOT: Entry point to a routine in the BIOS used when a warm start occurs. A warm start is performed when a user program branches to location 0000H, when the CPU is reset from the front panel, or when the user types CTRL-C. The CCP and BDOS are reloaded from the system tracks of drive A.

wildcard characters: Special characters that match certain specified items. In CP/M there are two wildcard characters: ? and *. The ? can be substituted for any single character in a filename, and the * can be substituted for the primary filename, the filetype, or both. By placing wildcard characters in filenames, the user creates an ambiguous filename and can quickly reference one or more files.

word: 16-bit or two-byte value, such as an address value. Although the Intel 8080 is an 8-bit CPU, addresses occupy two bytes and are called word values.

WRITE: Entry point to a routine in the BIOS that writes the record at the currently selected DMA address to the currently selected drive, track, and sector.

XLT: Logical-to-physical sector translation table located in the BIOS. SECTTRAN uses XLT to perform logical-to-physical sector number translation. XLT also refers to the two-byte address in the disk parameter header at DPBASE + 0. If this parameter is zero, no sector translation takes place. Otherwise this parameter is the address of the translation table.

ZERO PAGE: See page zero.

End of Appendix H

Appendix I CP/M Error Messages

Messages come from several different sources. CP/M displays error messages when there are errors in calls to the Basic Disk Operating System (BDOS). CP/M also displays messages when there are errors in command lines. Each utility supplied with CP/M has its own set of messages. The following lists CP/M messages and utility messages. One might see messages other than those listed here if one is running an application program. Check the application program's documentation for explanations of those messages.

Table-1. CP/MErrorMessages

Message	Meaning
?	<p>DDT. This message has four possible meanings:</p> <ul style="list-style-type: none"> - DDT does not understand the assembly language instruction. - The file cannot be opened. - A checksum error occurred in a HEX file. - The assembler/disassembler was overlaid.
ABORTED	<p>PIP. You stopped a PIP operation by pressing a key.</p>
ASM Error Messages	
D	Data error: data statement element cannot be placed in specified data area.
E	Expression error: expression cannot be evaluated during assembly.
L	Label error: label cannot appear in this context (might be duplicate label).

Table 1-1. (continued)

Message	Meaning
N	Not implemented: unimplemented features, such as macros, are trapped.
O	Overflow: expression is too complex to evaluate.
P	Phase error: label value changes on two passes through assembly.
R	Register error: the value specified as a register is incompatible with the code.
S	Syntax error: improperly formed expression.
U	Undefined label: label used does not exist.
V	Value error: improperly formed operand encountered in an expression.

BAD DELIMITER

STAT. Check command line for typing errors.

Bad Load

CCP error message, or SAVE error message.

Bdos Err On d:

Basic Disk Operating System error on the designated drive: CP/M replaces d: with the drive specification of the drive where the error occurred. This message is followed by one of the four phrases in the situations described below.

Table 1-1. (continued)

Message	Meaning
---------	---------

Bdos Err On d: Bad Sector	
---------------------------	--

This message appears when CP/M finds no disk in the drive, when the disk is improperly formatted, when the drive latch is open, or when power to the drive is off. Check for one of these situations and try again. This could also indicate a hardware problem or a worn or improperly formatted disk. Press TC to terminate the program and return to CP/M, or press RETURN to ignore the error.

Bdos Err On d: File R/O	
-------------------------	--

You tried to erase, rename, or set file attributes on a Read-Only file. The file should first be set to Read-Write (R[W]) with the command: STAT filespec \$R/W.

Bdos Err On d: R/O	
--------------------	--

Drive has been assigned Read-Only status with a STAT command, or the disk in the drive has been changed without being initialized with a TC. CP/M terminates the current program as soon as you press any key.

Bdos Err on d: Select	
-----------------------	--

CP/M received a command line specifying a nonexistent drive. CP/M terminates the current program as soon as you press any key. Press RETURN or CTRL-C to recover.

Break "x" at c	
----------------	--

ED. "x" is one of the symbols described below and c is the command letter being executed when the error occurred.

Search failure. ED cannot find the string specified in an F, S, or N command.

? Unrecognized command letter c. ED does not recognize the indicated command letter, or an E, H, Q, or O command is not alone on its command line.

Table I-1. (continued)

Message	Meaning
-	The file specified in an R command cannot be found.
>	Buffer full. ED cannot put any more characters in the memory buffer, or the string specified in an F, N, or S command is too long.
E	Command aborted. A keystroke at the console aborted command execution.
F	Disk or directory full. This error is followed by either the disk or directory full message. Refer to the recovery procedures listed under these messages.

CANNOT CLOSE DESTINATION FILE--{filespec}

PIP. An output file cannot be closed. You should take appropriate action after checking to see if the correct disk is in the drive and that the disk is not write-protected.

Cannot close, R/O

CANNOT CLOSE FILES

CP/M cannot write to the file. This usually occurs because the disk is write-protected.

ASM. An output file cannot be closed. This is a fatal error that terminates ASM execution. Check to see that the disk is in the drive, and that the disk is not write-protected.

DDT. The disk file written by a W command cannot be closed. This is a fatal error that terminates DDT execution. Check if the correct disk is in the drive and that the disk is not write-protected.

SUBMIT. This error can occur during SUBMIT file processing. Check if the correct system disk is in the A drive and that the disk is not write-protected. The SUBMIT job can be restarted after rebooting CP/M.

Table 1-1. (continued)

Message	Meaning
---------	---------

CANNOT READ

PIP. PIP cannot read the specified source. Reader cannot be implemented.

CANNOT WRITE

PIP. The destination specified in the PIP command is illegal. You probably specified an input device as a destination.

Checksum error

PIP. A HEX record checksum error was encountered. The HEX record that produced the error must be corrected, probably by recreating the HEX file.

CHECKSUM ERROR

LOAD ADDRESS hhhh

ERROR ADDRESS hhhh

BYTES READ:

h h h h :

LOAD. File contains incorrect data. Regenerate HEX file from the source.

Command Buffer Overflow

SUBMIT. The SUBMIT buffer allows up to 2048 characters in the input file.

Command too long

SUBMIT. A command in the SUBMIT file cannot exceed 125 characters.

\CORRECT ERROR, TYPE RETURN OR CTRL-Z

PIP. A HEX record checksum was encountered during the transfer of a HEX file. The HEX file with the checksum error should be corrected, probably by recreating the HEX file.

Table 1-1. (continued)

Message	Meaning
---------	---------

DESTINATION IS R/O, DELETE (Y/N)?	
-----------------------------------	--

PIP. The destination file specified in a PIP command already exists and it is Read-Only. If you type Y, the destination file is deleted before the file copy is done.

Directory full	
----------------	--

ED. There is not enough directory space for the file being written to the destination disk. You can use the OXfilespec command to erase any unnecessary files on the disk without leaving the editor.

SUBMIT. There is not enough directory space to write the \$\$\$SUB file used for processing SUBMITS. Erase some files or select a new disk and retry.

Disk full	
-----------	--

ED. There is not enough disk space for the output file. This error can occur on the W, E, H, or X commands. If it occurs with X command, you can repeat the command prefixing the filename with a different drive.

DISK READ ERROR--{filespec}	
-----------------------------	--

PIP. The input disk file specified in a PIP command cannot be read properly. This is usually the result of an unexpected end-of-file. Correct the problem in your file.

Table 1-1. (continued)

Message	Meaning
---------	---------

DISK WRITE ERROR--{filespec }

DDT. A disk write operation cannot be successfully performed during a W command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space.

PIP. A disk write operation cannot be successfully performed during a PIP command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space and execute PIP again.

SUBMIT. The SUBMIT program cannot write the \$\$\$SUB file to the disk. Erase some files, or select a new disk and try again.

ERROR: BAD PARAMETER

PIP. You entered an illegal parameter in a PIP command. Retype the entry correctly.

ERROR: CANNOT OPEN SOURCE, LOAD ADDRESS hhhh

LOAD. Displayed if LOAD cannot find the specified file or if no filename is specified.

ERROR: CANNOT CLOSE FILE, LOAD ADDRESS hhhh

LOAD. Caused by an error code returned by a BDOS function call. Disk might be write-protected.

ERROR: CANNOT OPEN SOURCE, LOAD ADDRESS hhhh

LOAD. Cannot find source file. Check disk directory.

ERROR: DISK READ, LOAD ADDRESS hhhh

LOAD. Caused by an error code returned by a BDOS function call.

Table 1-1. (continued)

Message	Meaning
---------	---------

ERROR: DISK WRITE, LOAD ADDRESS hhhh	
--------------------------------------	--

LOAD. Destination disk is full.

ERROR: INVERTED LOAD ADDRESS, LOAD ADDRESS hhhh	
---	--

LOAD. The address of a record was too far from the address of the previously-processed record. This is an internal limitation of LOAD, but it can be circumvented. Use DDT to read the HEX file into memory, then use a SAVE command to store the memory image file on disk.

ERROR: NO MORE DIRECTORY SPACE, LOAD ADDRESS hhhh	
---	--

LOAD. Disk directory is full.

Error on line nnn message

SUBMIT. The SUBMIT program displays its messages in the format shown above, where nnn represents the line number of the SUBMIT file. Refer to the message following the line number.

FILE ERROR

ED. Disk or directory is full, and ED cannot write anything more on the disk. This is a fatal error, so make sure there is enough space on the disk to hold a second copy of the file before invoking ED.

FILE EXISTS

You have asked CP/M to create or rename a file using a file specification that is already assigned to another file. Either delete the existing file or use another file specification.

REN. The new name specified is the name of a file that already exists. You cannot rename a file with the name of an existing file. If you want to replace an existing file with a newer version of the same file, either rename or erase the existing file, or use the PIP utility.

Table 1-1. (continued)

Message	Meaning
---------	---------

File exists, erase it	ED. The destination filename already exists when you are placing the destination file on a different disk than the source. It should be erased or another disk selected to receive the output file.
-----------------------	---

** FILE IS READ/ONLY **	ED. The file specified in the command to invoke ED has the ReadOnly attribute. Ed can read the file so that the user can examine it, but ED cannot change a Read-Only file.
-------------------------	---

File Not Found	CP/M cannot find the specified file. Check that you have entered the correct drive specification or that you have the correct disk in the drive.
----------------	--

ED. ED cannot find the specified file. Check that you have entered the correct drive specification or that you have the correct disk in the drive.
--

STAT. STAT cannot find the specified file. The message might appear if you omit the drive specification. Check to see if the correct disk is in the drive.
--

FILE NOT FOUND--{filespec}	PIP. An input file that you have specified does not exist.
----------------------------	--

Filename required	ED. You typed the ED command without a filename. Reenter the ED command followed by the name of the file you want to edit or create.
-------------------	--

hhh??=dd	DDT. The ?? indicates DDT does not know how to represent the hexadecimal value dd encountered at address hhhh in 8080 assembly language. dd is not an 8080 machine instruction opcode.
----------	--

Table 1-1. (continued)

Message	Meaning
Insufficient memory	DDT. There is not enough memory to load the file specified in an R or E command.
Invalid Assignment	STAT. You specified an invalid drive or file assignment, or misspelled a device name. This error message might be followed by a list of the valid file assignments that can follow a filename. If an invalid drive assignment was attempted the message Use: d: = RO is displayed, showing the proper syntax for drive assignments.
Invalid control character	SUBMIT. The only valid control characters in the SUBMIT files of the type SUB are ^ A through ^ Z. Note that in a SUBMIT file the control character is represented by typing the circumflex, ', not by pressing the control key.
INVALID DIGIT--{filespec}	PIP. An invalid HEX digit has been encountered while reading a HEX file. The HEX file with the invalid HEX digit should be corrected, probably by recreating the HEX file.
Invalid Disk Assignment	STAT. Might appear if you follow the drive specification with anything except = R/O.
INV)ALID DISK SELECT	CP/M received a command line specifying a nonexistent drive, or the disk in the drive is improperly formatted. CP/M terminates the current program as soon as you press any key.

Table 1-1. (continued)

Message	Meaning
INVALID	DRIVE NAME (Use A, B, C, or D)
	SYSGEN. SYSGEN recognizes only drives A, 5, C, and D as valid destinations for system generation.
Invalid File Indicator	
	STAT. Appears if you do not specify RO, RW, DIR, or SYS.
INVALID	FORMAT
	PIP. The format of your PIP command is illegal. See the description of the PIP command.
INVALID	HEX DIGIT
LOAD ADDRESS	hhhh
ERROR ADDRESS	hhhh
BYTES READ:	
h h h h	
	LOAD. File contains incorrect HEX digit.
INVALID MEMORY SIZE	
	MOVCPM. Specify a value less than 64K or your computer's actual memory size.
INVALID SEPARATOR	
	PIP. You have placed an invalid character for a separator between two input filenames.
INVALID USER NUMBER	
	PIP. You have specified a user number greater than 15. User numbers are in the range 0 to 15.

Table 1-1. (continued)

Message	Meaning
n ?	USER. You specified a number greater than fifteen for a user area number. For example, if you type USER 18<cr>, the screen displays 18?.
NO DIRECTORY SPACE	ASM. The disk directory is full. Erase some files to make room for PRN and HEX files. The directory can usually hold only 64 filenames.
NO DIRECTORY SPACE--{filespec}	PIP. There is not enough directory space for the output file. You should either erase some unnecessary files or get another disk with more directory space and execute PIP again.
NO FILE--{filespec}	DIR, ERA, REN, PIP. CP/M cannot find the specified file, or no files exist. ASM. The indicated source or include file cannot be found on the indicated drive. DDT. The file specified in an R or E command cannot be found on the disk.
NO INPUT FILE PRESENT ON DISK	DUMP. The file you requested does not exist.
No memory	There is not enough (buffer?) memory available for loading the program specified.

Table 1-1. (continued)

Message	Meaning
NO SOURCE FILE ON DISK	SYSGEN. SYSGEN cannot find CP/M either in C P / M x x . C 0 M form or on the system tracks of the source disk.
NO SOURCE FILE PRESENT	ASM. The assembler cannot find the file you specified. Either you mistyped the file specification in your command line, or the filetype is not ASM.
NO SPACE	SAVE. Too many files are already on the disk, or no room is left on the disk to save the information.
No SUB file Present	SUBMIT. For SUBMIT to operate properly, you must create a file with filetype of SUB. The SUB file contains usual CP/M commands. Use one command per line.
NOT A CHARACTER SOURCE	PIP. The source specified in your PIP command is illegal. You have probably specified an output device as a source.
** NOT DELETED **	PIP. PIP did not delete the file, which might have had the R/O attribute.
NOT FOUND	PIP. PIP cannot find the specified file.

Table I-1. (continued)

Message	Meaning
---------	---------

OUTPUT FILE WRITE ERROR

ASM. You specified a write-protected disk as the destination for the PRN and HEX files, or the disk has no space left. Correct the problem before assembling your program.

Parameter error

SUBMIT. Within the SUBMIT file of type sub, valid parameters are \$0 through \$9.

PARAMETER ERROR, TYPE RETURN TO IGNORE

SYSGEN. If you press RETURN, SYSGEN proceeds without processing the invalid parameter.

QUIT NOT FOUND

PIP. The string argument to a Q parameter was not found in your input file.

Read error

TYPE. An error occurred when reading the file specified in the type command. Check the disk and try again. The STAT filespec command can diagnose trouble.

READER STOPPING

PIP. Reader operation interrupted.

Record Too Long

PIP. PIP cannot process a record longer than 128 bytes.

Requires CP/M 2.0 or later

XSUB. XSUB requires the facilities of CP/M 2.0 or newer version.

Table 1-1. (continued)

Message	Meaning
---------	---------

Requires CP/M 2.0 or new for operation

PIP. This version of PIP requires the facilities of CP/M 2.0 or newer version.

START NOT FOUND

PIP. The string argument to an S parameter cannot be found in the source file.

SOURCE FILE INCOMPLETE

SYSGEN. SYSGEN cannot use your CP/M source file.

SOURCE FILE NAME ERROR

ASM. When you assemble a file, you cannot use the wildcard characters " and ? in the filename. Only one file can be assembled at a time.

SOURCE FILE READ ERROR

ASM. The assembler cannot understand the information in the file containing the assembly-language program. Portions of another file might have been written over your assembly-language file, or information was not properly saved on the disk. Use the TYPE command to locate the error. Assembly-language files contain the letters, symbols, and numbers that appear on your keyboard. If your screen displays unrecognizable output or behaves strangely, you have found where computer instructions have crept into your file.

SYNCHRONIZATION ERROR

MOVCPM. The MOVCPM utility is being used with the wrong CP/M system.

"SYSTEM" FILE NOT ACCESSIBLE

You tried to access a file set to SYS with the STAT command.

Table 1-1. (continued)

Message	Meaning
---------	---------

**** TOO MANY' FILES ****

STAT. There is not enough memory for STAT to sort the files specified, or more than 512 files were specified.

UNEXPECTED END OF HEX FILE--{filespec}

PIP. An end-of-file was encountered prior to a termination HEX record. The HEX file without a termination record should be corrected, probably by recreating the HEX file.

Unrecognized Destination

PIP. Check command line for valid destination.

Use: STAT d:=RO

STAT . An invalid STAT drive command was given. The only valid drive assignment in STAT is STAT d: = RO.

VERIFY ERROR:--{filespec}

PIP. When copying with the V option, PIP found a difference when rereading the data just written and comparing it to the data in its memory buffer. Usually this indicates a failure of either the destination disk or drive.

WRONG CP/M VERSION (REQUIRES 2.0)

XSUB ACTIVE

SUBMIT. XSUB has been invoked.

XSUB ALREADY PRESENT

SUBMIT. XSUB is already active in memory.

Table 1-1. (continued)

Message	Meaning
---------	---------

Your Input?	
-------------	--

If CP/M cannot find the command you specified, it returns the command name you entered followed by a question mark. Check that you have typed the command line correctly, or that the command you requested exists as a COM file on the default or specified disk.

End of Appendix I

A

Absolute line number, 2-5
 2-7, 2-10, 2-20, Access mode, 1-19
 afn (ambiguous file reference), 1-4, 1-7
 Allocation vector, 5-27
 Ambiguous file reference (afn), 1-4, 1-7
 ASM, 1-22, 3-1
 Assembler, 1-22, 3-1
 Assembler/disassembler module
 (DDT), 4-1 1
 Assembler errors, 3-24
 Assembly language mnemonics in
 DDT, 4-4, 4-7
 Assembly language program, 3-3
 Assembly language statement, 3-3
 Automatic command processing, 1-39

B

Base, 3-5
 Basic Disk Operating System (BDOS),
 1-2, 5-1, 6-1
 Basic I/O System (BIOS), 1-2, 5-1, 6-1
 BDOS (Basic Disk Operating System),
 1-2, 5-1, 6-1
 Binary constants, 3-5
 BIOS (Basic I/O System), 1-2, 5-1, 6-1
 BIOS disk definition, 6-34
 BIOS subroutines, 6-15
 Block move command, 4-8
 bls parameter, 6-35
 BOOT, 5-2, 6-13, 6-20
 BOOT entry point, 6-20
 Break point, 4-4, 4-6
 Built-in commands, 1-3

C

Case translation, 1-6, 1-7, 1-31, 1-32 1-33,
 2-21, 2-22, 3-7, 5-10, 5-11
 CCP (Console Command Processor),
 1-2, 4-1, 5-1, 6-1
 CCP Stack, 5-6
 Character pointer, 2-4
 cks parameter, 6-35
 Close File function, 5-20
 Code and data areas, 6-26
 Cold start loader, 6-13, 20, 25
 Command, 1-3
 Command line, 5-3
 Comment field, 3-4
 Compute File Size function, 5-33
 Condition flags, 3-17, 4-11
 Conditional assembly, 3-14
 CONIN, 6-21
 CONOUT, 6-21
 CONSOLE, 6-18
 Console Command Processor (CCP),
 1-2, 4-1, 5-1, 6-1
 Console Input function, 5-12
 Console Output function, 5-12
 CONST, 6-21
 Constant, 3-5
 Control characters, 2-19
 Control functions, 1-13
 CTRL-Z character, 5-7
 Copy files, 1-25
 CPU state, 4-3, 4-4
 cr (carriage return), 2-10
 Create files, 1-35
 Create system disk, 1-37
 Creating COM files, 1-24
 Currently logged disk, 1-3, 1-7, 1-15, 1-36

D

Data allocation size, 6-31
Data block number, 6-32
DB statement, 3-15
DDT commands, 4-4, 6-9
DDT nucleus, 4-11
DDT prompt, 4-2
DDT sign-on message, 4-1
Decimal constant, 3-5
Default FCB, 4-7
Delete File function, 5-22
DESPOOL, 6-17
Device assignment, 1-16
DIR, 1-9
DIR attribute, 1-20
dir parameter, 6-35
Direct console I/O function, 5-14
Direct Memory Address, 5-27
Directory, 1-9
Directory code, 5-19, 5-20, 5-21, 5-22,
5-23, 5-24, 5-25
Disassembler, 4-4, 1 1
Disk attributes, 1-15
Disk drive name, 1-6, 7
Disk I/O functions, 5-17-5-35
Disk parameter block, 6-30
Disk parameter header, 6-28
Disk parameter table, 6-28
Disk statistics, 1-15
Disk-to-disk copy, 1-27
DISKDEF macro, 6-34
Diskette format, 1-47
DISKS macro, 6-34
Display file contents, 1 -1 1
dks parameter, 6-35
DMA, 5-27
DMA address, 5-8
dn parameter, 6-35
DPBASE, 6-29

Drive characteristics, 1-21

Drive select code, 5-9

Drive specification, 1-7

DS statement, 3-16

DUMP, 1-41 5-40

DW statement, 3-15

E

ED, 1-35, 2-1-2-22, 6-6

ED commands, 2-8,19

ED errors, 2-18

Edit command line, 1-1 2

8080 CPU registers, 4-10

8080 registers, 3-6

end-of-file, 1-28, 5-7

END statement, 3-4, 3-11

EMDEF macro, 6-35

ENDIF statement, 3-13

EQU statement, 3-12

ERA, 1-8

Erase files, 1-8

Error messages, 1-44, 2-18, 3-24

Expression, 3-4

Extents, 1-19

F

FBASE, 5-2

FCB, 5-8,5-9

FCB format, 5-8, 5-9

FDOS (operations), 5-1, 5-4

File attributes, I - 20

File compatibility, 1-35

File control block (FCB), 5-8, 5-9

File expansion, 6-2

File extent, 5-8

File indicators, 1-20

File names, 1-4

File reference, 1-4

File statistics, 1-15, 1-19

Filetype, 5-6

Find command, 2-11

fsc parameter, 6-35

G

Get ADDR (Alloc) function, 5-27

Get ADDR (Disk Parms) function, 5-29
5-16

Get Console Status, 5-17

Get I/O Byte function, 5-15

Get Read/Oddly Vector function, 5-28

GETSYS, 6-3, 6-11

H

Hexadecimal constant, 3-5

HOME subroutine, 6-20, 22

I

6-35

Identifier, 3-3, 3-5

IF statement, 3-13

Initialized storage areas, 3-15

In-line assembly language, 4-4

Insert mode, 2-7

Insert String, 2-12

IOBYTE function, 6-17-6-19

J

jump vector, 6-15

juxtaposition command, 2-15

K

Key fields, 5-34

Label field, 3-3

Labels, 3-3, 3-4, 3-16

Library read command, 2-16

Line-editing control characters, 2-9, 4-2,

Line-editing functions, I-12

Line numbers, 2-5

LIST, 6-17, 6-21

List Output function, 5-14

LISTST, 6-24

LOAD, 1-24

Logged in, 1-3

Logical devices, 1-16, 1-28, 6-17

Logical extents, 5-8

Logical-physical assignments, 1-18, 6-19

Logical to physical device mapping, 6-18

Logical to physical sector translation 6-24,

Isc parameter, 6-35

M

Macro command, 2-17

Make File function, 5-25

Memory buffer, 2-1-2-7

Memory image, 4-3, 6-6, 6-7

Memory size, 1-42, 6-3, 6-8

MOVCPM, 1-42, 6-7

N

Negative bias, 6-7

O

[o] parameter, 6-35

Octal constant, 3-5

ofs parameter, 6-35

On-line status, 5-19

Open File function, 5-19

Operand field, 3-4, 3-6

Operation field, 3-4, 3-16

Operators, 3-9, 3-16

ORG directive, 3-11

P

Page zero, 6-26

Patching the CP/M system, 6-3

Peripheral devices, 6-17

Physical devices, 1-17, 6-17

Physical file size, 5-33

Physical to logical device assignment,
1-18, 6-19

PIP devices, 1-28

PIP parameters, 1-31

Print String function, 5-15

PRN file, 3-1

Program counter, 4-4, 4-6, 4-7, 4-11

Program tracing, 4-9

Prompt, 1-3

Pseudo-operation, 3-10

PUNCH, 6-17, 6-21

Punch Output function, 5-13

PUTSYS, 6-4, 6-11

Radix indicators, 3-5

Random access, 5-31, 5-32, 5-46

Random record number, 5-32

READ, 6-23

Read Console Buffer function, 5-16

Read only, 1-20

Read/only status, 1-20

Read random error codes, 5-31

Read Random function, 5-30

READ routine, 6-20

Read Sequential function, 5-23

Read/write, 1-20

READER, 6-18, 21

Reader Input function, 5-13

REN, 1-10

Rename file function, 5-25

Reset Disk function, 5-18

Reset Drive function, 5-35

Reset state, 5-18

Return Current Disk function, 5-26

Return Log-in Vector function, 5-26

Return Version Number function, 5-18

R/O, 1-20

R/O attribute, 5-29

R/O bit, 5-28

R/W, 1-20

S

SAVE, 1-11

SAVE command, 4-3

Search for First function, 5-21

Search for Next function, 5-22

Search strings, 2-11

Sector allocation, 6-13

SECTRAN, 6-2
SELDSK, 6-19, 6-22, 6-30
Select Disk function, 5-19
Sequential access, 5-8
Set DMA address function, 5-27
Set File Attributes function, 5-29
Set/Get User Code function, 5-30
Set I/O Byte function, 5-15
Set Random Record function, 5-34
SET statement, 3-13
SETDMA, 6-23
SETSEC, 6-23
SETTRK, 6-22
Simple character I/O, 6-17
Size in records, 1-19
skf parameter, 6-35, 6-37
Source files, 5-7
Stack pointer, 5-6
STAT, 1-15, 6-17, 6-38
Stop console output, 1-13
String substitutions, 2-14
SUBMIT, 1-39
SYS attribute, 1-20
SYSGEN, 1-37, 6-10
System attribute, 2-19, 5-29
System parameters, 6-20
System (re)initialization, 6-16
System Reset function, 5-11

T

Testing and debugging of programs, 41
Text transfer commands, 2-3
TPA (Transient Program Area), 1-2, 51
Trace mode, 4-10
Transient commands, 1-3, 1-14
Transient Program Area (TPA), 1-2, 5-1

Translate table, 6-37
Translation vectors, 6-30
TYPE, 1-11

U

ufn, 1-4, 1-7
Unambiguous file reference, 1-4, 1-7
Uninitialized memory, 3-16
Untrace mode, 4-10
USER, 1-12
USER numbers, 1-12, 1-22, 5-30

V

Verify line numbers command, 2-6, 21
Version independent programming, 5-18
Virtual file size, 5-33

W

Warm start, 5-2, 6-20
 WBOOT entry point, 6-20
Write routine, 6-24
Write Sequential function, 5-24
WRITE, 6-24
Write Protect Disk function, 5-28
Write random error codes, 5-32
Write Random function, 5-32
Write Random with Zero Fill function, 5-35

X

XSUB, 1-41