BACKUP

Syntax:

```
BACKUP
```

Explanation:

This command copies the entire contents of a diskette, track by track, to another diskette. Once the command is entered, the user is prompted for the source and destination drive. The total command is confirmed by a question:

```
BACKUP DR X TO DR Y (YES/NO-GO):
```

Entering YES-GO will cause the process to begin. After the backup is finished, you are prompted for the option of requesting another backup.

BACKUP
EXAMPLE

```
A>BACKUP
Source Drive <SRC>: B<cr>
Destination Drive <DST>: C<cr>
Backup B to C (YES/NO-GO): Y<cr>
Another Backup (YES/NO-GO): N<cr>
Place system disk in A and press GO:A>
```

This example backs up diskette in
drive B onto diskette in drive C
and then returns to CP/M.

DEVICE

Syntax:

```
DEVICE < NAMES " VALUES " physical-dev " logical-dev>

DEVICE  logical-dev=physical-dev <option> <,physical-dev
<option>,...>

DEVICE logical-dev = NULL

DEVICE physical-dev <option>

DEVICE  CONSOLE [ PAGE " COLUMNS = columns " LINES = lines]
```

Explanation:

DEVICE displays current logical device assignments and physical device names. DEVICE assigns logical devices to peripheral devices attached to the computer. DEVICE also sets the communications protocol and speed of a peripheral device, and displays or sets the current console screen size.

DEVICE
OPTIONS

[XON " NOXON " baud-rate]

XON
refers to the XON/XOFF communications protocol.

NOXON
indicates no protocol and the computer sends data to the device whether or not the device is ready to receive it.

baud-rate
is the speed of the device. The system accepts the following baud rates:

50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200

DEVICE
EXAMPLES

```
A>DEVICE
```

Displays the physical devices and current assignments of the logical devices in the system.

```
A>DEVICE NAMES
```

Lists the physical devices with a summary of the device characteristics.

```
A>DEVICE VALUES
```

Displays the current logical device assignments.

```
A>DEVICE CRT
```

Displays the attributes of the physical device CRT.

```
A>DEVICE CON
```

Displays the assignment of the logical device CON:

```
A>DEVICE CONOUT:=LPT,CRT
```

Assigns the system console output (CONOUT:) to the printer (LPT) and the screen (CRT).

```
A>DEVICE AUXIN:=CRT2 [XON,9600]
```

Assigns the auxiliary logical input device (AUXIN:) to the physical device CRT using protocol XON/XOFF and sets the transmission rate for the device at 9600.

```
A>DEVICE LST:=NULL
```

Disconnects the list output logical device (LST:).

```
A>DEVICE LPT [XON,9600]
```

Sets the XON/XOFF protocol for the physical device LPT and sets the transmission speed at 9600.

```
A>DEVICE CONSOLE [PAGE]
```

Displays the current console page width in columns and length in lines.

```
A>DEVICE CONSOLE
              [COLUMNS=40 LINES=16]
```

Sets the screen size to 40 columns and 16 lines.

FORMAT

Syntax:

```
FORMAT
```

Explanation:

This command formats a diskette to the NABU standard, either single or double sided, with 1024-byte sectors. It also gives the option of formatting at 96 tracks per inch instead of the standard 48. When the prompt:Format A single sided (YES/NO-GO): is answered with a Y<cr> (ie YES), formatting begins. You are kept informed of the progress by:

```
Track = nnn
```

When the formatting is finished, the option is given to format another diskette.

FORMAT
EXAMPLE

```
A>FORMAT
    DISK FORMATTING UTILITY
```

```
Which drive is to be formatted: A
96 Track per inch (YES/NO-GO):  N
Single or Double Sided (S/D):  S
Format A single sided (YES/NO-GO):
Another FORMAT (YES/NO-GO):  N
Place system disk in A and press GO:
A>
```

This example formatted a disk in drive A at 48 TPI, single sided, and double density.

GET

Syntax:

GET <CONSOLE INPUT FROM> FILE filespec<[<ECHO"NO ECHO> " SYSTEM]>

GET <CONSOLE INPUT FROM> CONSOLE

Explanation:

GET directs the system to take console input from a file for the next system command or user program entered at the console.

Console input is taken from a file until the program terminates. If the file is exhausted before program input is terminated, the program looks for subsequent input from the console. If the program terminates before exhausting all its input, the system reverts back to the console for console input.

With the SYSTEM option, the system immediately goes to the specified file for console input. The system reverts to the console for input when it reaches the end of file. Re-direct the system to the console for console input with the GET CONSOLE INPUT FROM CONSOLE command as a command line in the input file.

GET
OPTIONS

ECHO
specifies that input is echoed to the console. This is the default option.

NO ECHO
specifies that file input is not echoed to the console. The program output and the system prompts are not affected by this option and are still echoed to the console.

SYSTEM
specifies that all system input is immediately taken from the

disk file specified in the command line. GET takes system and program input from the file until the file is exhausted or until GET reads a GET console command from the file.

GET
EXAMPLES

```
A>GET FILE XINPUT
A>MYPROG
```

Tells the system to activate the GET utility. Since SYSTEM is not specified, the system reads the next input line from the console and executes MYPROG. If MYPROG program requires console input, it is taken from the file XINPUT. When MYPROG terminates, the system reverts back to the file for console input.

```
A>GET FILE XIN2 [SYSTEM]
```

Immediately directs the system to get subsequent console input from file XIN2 because it includes the SYSTEM option. The system reverts back to the console for console input when it reaches the end of file in XIN2. Or XIN2 may redirect the system back to the console if it contains a GET CONSOLE command.

```
A>GET CONSOLE
```

Tells the system to get console input from the console. This command may be used in a file (previously specified in a GET FILE command), which is already being read by the system for console input. It is used to re-direct the console input back to the console before the end-of-file is reached.

RENAME

Syntax:

```
RENAME <new-filespec=old-filespec>
```

Explanation:

RENAME lets you change the name of a file in the directory of a disk. To change several filenames in one command use the * or ? wildcards in the file specifications. The RENAME command can be abbreviated REN. REN prompts you for input.

RENAME
EXAMPLES

```
A>RENAME NEWFILE.BAS=OLDFILE.BAS
```

The file OLDFILE.BAS changes to NEWFILE.BAS on drive A.

```
A>RENAME
```

The system prompts for the filespecs:

```
Enter New Name:X.PRN
Enter Old Name:Y.PRN
Y       .PRN=X          .PRN
A>
```

File X.PRN is renamed to Y.PRN on drive A.

```
B>REN A:PRINTS.NEW = PRINCE.NEW
```

The file PRINCE.NEW on drive A changes to PRINTS.NEW on drive A.

```
A>RENAME S*.TEX=A*.TEX
```

The above command renames all the files matching A*.TEX to files with filenames S*.TEX.

```
A>REN B:NEWLIST=B:OLDLIST
```

The file OLDLIST changes to NEWLIST on drive B. Since the second drive specifier, B: is implied by the first one, it is unnecessary in this example. The command line above has the same effect as the following:

```
A>REN B:NEWLIST=OLDLIST
           or
A>REN NEWLIST=B:OLDLIST
```

SUBMIT

Syntax:

```
SUBMIT <filespec> <argument> ... <argument>
```

Explanation:

The SUBMIT command lets you execute a group (batch) of commands from a SUBmit file (a file with filetype of SUB).

SUBMIT

SUBFILE

The SUB file can contain the following types of lines:

- Any valid CP/M 3 command
- Any valid CP/M 3 command with SUBMIT parameters ($Ø-$9)
- Any data input line
- Any program input line with parameters ($Ø to $9)

The command line cannot exceed 135 characters.

The following lines illustrate the variety of lines which may be entered in a SUB file:

```
DIR
DIR *.BAK
MAC $1 $$$4
PIP LST:=$1.PRN[T$2 $3 $5]
DIR *.ASM
PIP
<B:=*.ASM
<CON:=DUMP.ASM
<
DIR B:
```

SUBMIT
EXECUTE

Syntax:

```
SUBMIT
SUBMIT filespec
SUBMIT filespec argument ... argument
```

Examples:

```
A>SUBMIT
A>SUBMIT SUBA
A>SUBMIT AA ZZ SZ
A>SUBMIT B:START DIR E:
```

SUBMIT
PROFILE.SUB

Everytime you power up or reset your computer, CP/M 3 looks for a special SUBmit file named PROFILE.SUB to execute. If it does not exist, CP/M 3 resumes normal operation. If the PROFILE.SUB file exists, the system executes the commands in the file. This file is convenient to use if you regularly execute a set of commands before you do your regular session on the computer.

TYPE

Syntax:

```
TYPE <filespec<[ PAGE " NOPAGE ]>>
```

Explanation:

The TYPE command displays the contents of an ASCII character file on your screen.

[PAGE]
Causes the console listing to be displayed in paged mode; i.e., stop automatically after listing n lines of text, where n normally defaults to 24 lines per page.

[NOPAGE]
Turns off Console Page Mode and continuously displays a typed file on the screen.

TYPE
EXAMPLES

```
A>TYPE MYPROG.PLI
```

Displays the contents of the file MYPROG.PLI on your screen.

```
A>TYPE B:THISFILE [PAGE]
```

Displays the contents of the file THISFILE from drive B on your screen twenty four lines at a time.

CHECK

Syntax:

```
CHECK
```

Explanation:

This command checks the diskette, sector by sector, for any bad sectors. This command does not check the validity of the information on the diskette, but only verifies the integrity of each sector. As soon as checking begins, you will be kept informed of the progress by:

```
  VERIFING TRACK:  nnn
```

If any bad sectors are found, you will be informed by:

```
  ERROR ON TRACK = nnn,
          SECTOR = yyy
```

As checking progresses, a running total is kept of the number of bad sectors as:

```
  TOTAL BAD SECTORS: xxx
```

When the check is finished, the option is given to do another check by:

```
  ANOTHER CHECK (YES/NO-GO):
```

CHECK
EXAMPLE

```
A>CHECK
Enter drive to check (A-P): A
Verifying track:  39
     Total bad sectors:      Ø
Another CHECK (YES/NO-GO) N
Put system disk in A and press GO:
A>
```

This example checked the diskette in drive A and found no errors.

DIR

The DIR command displays the names of files and the characteristics associated with the files.

The DIR command has three distinct references:

```
        DIR
        DIRS
        DIR with Options
```

DIR and DIRS are built-in utilities. DIR with Options is a transient utility and must be loaded into memory from the disk.

DIR
BUILT-IN

Syntax:

```
        DIR  <d:>
        DIR  <filespec>
        DIRS <d:>
        DIRS <filespec>
```

Explanation:

The DIR and DIRS Built-in commands display the names of files cataloged in the directory of an on-line disk. DIR lists the names of files in the current user number that have the Directory (DIR) attribute. DIR accepts the * and ? wildcards in the file specification.

DIR
  BUILT-IN
    EXAMPLES

```
A>DIR
```

Displays all files in user Ø on drive A that have the Directory attribute.

```
A>DIR B:
```

Displays all DIR files in user Ø on drive B.

```
2A>DIR C:ZIPPY.DAT
```

Displays the name ZIPPY.DAT if the file is in user 2 on drive C.

```
4A>DIR *.BAS
```

Displays all DIR files with filetype BAS in user 4 on drive A.

```
B3>DIR X*.C?D
```

Displays all DIR files in user 3 on drive B whose filename begins with the letter X, and whose three character filetype contains the first character C and last character D.

```
A>DIRS
```

Displays all files for user Ø on drive A that have the system (SYS) attribute.

```
A>DIRS *.COM
```

Displays all SYS files with filetype COM on drive A in user Ø. A command (.COM) file in user Ø with the system attribute can be accessed from any user number on that drive, and from any drive in the search chain (see SETDEF).

DIR
  WITHOPTIONS
Syntax:

```
        DIR <d:> [options]

        DIR <filespec>  <filespec> ... [options]
```

Explanation:

The DIR command with options is an enhanced version of the DIR built-in command and displays your files in a variety of ways. DIR can search for files on any or all drives, for any or all user numbers. One or two letters is sufficient to identify an option. You need not type the right hand square bracket.

DIR
  WITHOPTIONS
    OPTIONS

Option   Function

ATT
    displays the file attributes.

DATE
    displays date and time stamps of files.

DIR
    displays only files that have the DIR attribute.

DRIVE=ALL
    displays files on all on-line drives.

DRIVE=(A,B,C,....,P)
    displays files on the drives specified.

DRIVE=d
    displays files on the drive specified by d.

EXCLUDE
    displays files that DO NOT MATCH the files specified in the command line.

FF sends an initial form feed to the printer device if the printer has been activated by CTRL-P.

FULL
    shows the name, size, number of 128-byte records, and attributes of the files. If there is a directory label on the drive, DIR shows time stamps. If there is no directory label, DIR displays two file

entries on a line, omitting the time stamp columns. The display is alphabetically sorted. (See SET for a description of file attributes, directory labels, and protection modes.)

LENGTH=n
displays n lines of printer output before inserting a table heading. n is a number between 5 and 65536.

MESSAGE
displays the names of drives and user numbers DIR is searching.

NOSORT
displays files in the order it finds them on the disk.

RO displays only the files that have the Read-Only attribute.

RW displays only the files that are set to Read-Write.

SIZE
displays the filename and size in kilobytes (1Ø24 bytes).

SYS
displays only the files that have the SYS attribute.

USER=ALL
displays all files in all user numbers for the default or specified drive.

USER=n
displays the files in the user number specified by n.

USER=(Ø,1,...,15)
displays files under the user numbers specified.

DIR
WITHOPTIONS
EXAMPLES

```
A>DIR C: [FULL]
```

Displays full set of characteristics for all files in user Ø on drive C.

```
A>DIR C: [DATE]
```

Lists the files on drive C and their dates.

```
A>DIR D: [RW,SYS]
```

Displays all files in user Ø on drive D with Read-Write and

System attributes.

```
3A>DIR [USER=ALL, DRIVE=ALL]
```

Displays all the files in all user numbers (Ø-15) in all on-line drives.

```
B6>DIR [exclude] *.DAT
```

Lists all the files on drive B in user 6 that do not have a filetype of .DAT.

```
3B>DIR [SIzE] *.PLI *.COM *.ASM
```

Displays all the files of type PLI, COM, and ASM in user 3 on drive B in size display format.

```
A>DIR [drive=all user=all] TESTFILE.BOB
```

DIR displays the filename TESTFILE.BOB if it is found on any drive in any user number.

```
A>DIR [size,rw] D:
```

DIR lists each Read-Write file that resides on Drive D, with its size in kilobytes. Note that D: is equivalent to D:*.*.

HELP

Syntax:

```
HELP <topic> <subtopic1 ... subtopic8><[NOPAGE"LIST]>
```

Explanation:

HELP displays a list of topics and provides summarized information for CP/M 3 commands.

HELP topic displays information about that topic. HELP topic subtopic displays information about that subtopic.

One or two letters is enough to identify the topics. After HELP displays information for your topic, it displays the special prompt HELP> on your screen, followed by a list of subtopics.

- Enter ? to display list of main topics.
- Enter a period and subtopic name to access subtopics.
- Enter a period to redisplay what you just read.
- Press the RETURN key to return to the CP/M 3 system prompt.
- [NOPAGE] option disables the 24 lines per page console display.
- Press any key to exit a display and return to the HELP> prompt.

Examples:

```
A>HELP

A>HELP DATE

A>HELP DIR OPTIONS

A>HELP>.OPTIONS
HELP>SET
HELP>.
HELP><cr>
```

```
Topics available:

BACKUP      CHECK       CNTRLCHARS  COMMANDS    COPYSYS     DATE

DEVICE      DIR         DUMP        ERASE       FILESPEC
FORMAT
GET         HELP        NABUSHOW    PATCH       PIP (COPY)  PUT

RENAME      SAVE        SET         SETDEF      SHOW
SUBMIT
TYPE        USER

HELP> SAVE
```

SAVE

Syntax:

```
SAVE
```

Explanation:

SAVE copies the contents of memory to a file. To use SAVE, first issue the SAVE command, then run your program which reads a file into memory. Your program exits to the SAVE utility which prompts you for a filespec to which it copies the contents of memory, and the beginning and ending address of the memory to be SAVEd.

SAVE
EXAMPLE

```
A>SAVE
```

Activates the SAVE utility. Now enter the name of the program which loads a file into memory.

```
A>SID dump.com
```

Next, execute the program.

```
#g0
```

When the program exits, SAVE intercepts the return to the system and prompts the user for the filespec and the bounds of memory to be SAVEd.

```
SAVE Ver 3.0
Enter  file  (type RETURN  to exit):dump2.com
```

If file DUMP2.COM exists already, the system asks:

```
Delete dump2.com? Y
```

Then the system asks for the bounds of memory to be saved:

```
Beginning hex address: 100
Ending hex address: 400
```

The contents of memory from 100H (Hexadecimal) to 400H is copied to file DUMP2.COM.

PIP (COPY)

Syntax:

```
        DESTINATION          SOURCE

PIP d:<Gn> " filespec<[Gn]> = filespec<[o]>,... " d:<[o]>
```

Explanation:

The file copy program PIP copies files, combines files, and transfers files between disks, printers, consoles. or other devices attached to your computer. The first filespec is the destination. The second filespec is the source. Use two or more source filespecs separated by commas to combine two or more files into one file.

[o] is any combination of the available options. The [Gn] option in the destination filespec tells PIP to copy your file to that user number.

PIP with no command tail displays an * prompt and awaits your series of commands, entered and processed one line at a time. The source or destination can be any CP/M 3 logical device.

PIP (COPY)
  OPTIONS

PIP OPTIONS

A   Archive. Copy only files that have been changed since the last copy.
C   Confirm. PIP prompts for confirmation before each file copy.
Dn   Delete any characters past column n.
E   Echo transfer to console.
F   Filter form-feeds from source data.
Gn   Get from or go to user n.
H   Test for valid Hex format.
I   Ignore :ØØ Hex data records and test for valid Hex format.
K   Kill display of filespecs on console.
L   Translate upper case to lower case.
N   Number output lines
O   Object file transfer, ↑Z ignored.
Pn   Set page length to n. (default n=6Ø)
Qs↑z Quit copying from source at string s.
R   Read files that have been set to SYStem.
Ss↑z Start copying from the source at the string s.
Tn   Expand tabs to n spaces.
U   Translate lower case to upper case.
V   Verify that data has been written correctly.
W   Write over Read Only files without console query.
Z   zero the parity bit.

All options except C,G,K,O,R,V and W force an ASCII file transfer, character by character, terminated by a ↑Z.

PIP (COPY)
  EXAMPLES

COPY A FILE FROM ONE DISK TO ANOTHER

```
A>PIP b:=a:draft.txt
A>PIP b:draft.txt = a:
B3>PIP myfile.dat=A:[G9]
A9>PIP B:[G3]=myfile.dat
```

COPY A FILE AND RENAME IT

```
A5>PIP newdraft.txt=
             oldraft.txt
C8>PIP b:newdraft.txt=
             a:oldraft.txt
```

COPY MULTIPLE FILES

```
A>PIP b:=draft.*
```

```
A>PIP b:=*.*
B>PIP b:=c:.*.*
C>PIP b:=*.txt[g5]
C>PIP a:=*.com[wr ]
B>PIP a:[g3]=c:*.*
```

COMBINE MULTIPLE FILES

```
A>PIP b:new.dat= file1.dat,file2.dat
```

COPY, RENAME AND PLACE IN USER 1

```
A>pip newdraft.txt[g1]= oldraft.txt
```

COPY, RENAME AND GET FROM USER 1

```
A>PIP newdraft.txt= oldraft.txt[g1]
```

COPY TO/FROM LOGICAL DEVICES

```
A>PIP b:funfile.sue=con:
A>PIP lst:=con:
A>PIP lst:=b:draft.txt[t8]
A>PIP prn:=b:draft.txt
```

SID

Syntax:

```
     SID [pgm-flespec] [,sym-filespec]
```

Explanation:

The SID symbolic debugger allows you to monitor and test programs developed for the 8080 microprocessor. SID supports real-time breakpoints, fully monitored execution, symbolic disassembly, assembly, and memory display and fill functions. SID can dynamically load SID utility programs to provide traceback and histogram facilities.

SID COMMANDS

SID
COMMANDS

Command   Meaning

As   (Assemble)

Enter assembly language statements where s is the start address

Cs[b[,d]]   (Call)

Call to memory location from SID where s is the called address, b is the value of the BC register, pair d is the value of the DE register pair.

D[W[s]][,f]   (Display)

Display memory in hex and ASCII where W is a 16-bit word format, s is the start address, f is the finish address.

Epgm-filespec   (Load)

Load program and symbol table [,sym-filespec] for execution.

E*sym-filespec   (Load)

Load a symbol table file

F,f,d   (Fill)

Fill memory with constant value where s is the start address, f is the finish address, d is an eight-bit data item.

G[p][,a[,b]]   (Go)

Begin Execution where p is a start address, a is a temporary breakpoint.

H   (Hex)

Displays all symbols with addresses in Hex.

H.a

Displays hex, decimal, an ASCII values of a where a is a symbolic expression.

Ha,b

Computes hex sum and difference of a and b where a and b are symbolic expressions.

Icommand tail (Input)

Input CCP commad line

L[s][,f] (List)

List 8080 mnemonic instructions where s is the start address, f is the finish address.

Ms,h,d (Move)

Move Memory Block where s is the start address, h is the high address of the block, d is the destination start address.

P[p[,c]] (Pass)

Pass point set, reset, and display where p is a permanent breakpoint address,c is initial value of pass counter.

Rfilespec[,d] (Read)

Read Code/Symbols where d is an offset to each address.

S[W]s (Set)

Set Memory Values where s is address where value is sent, W is 16 bit word.

T[n[,c]] (Trace)

Trace Program Execution where n is the number of program steps, c is the utility entry address.

T[W][n[,c]] (Trace)

Trace Without Call where W instructs SID not to trace subroutines, n is the number of program steps, c is the utility entry address.

U[W][n[,c]] (Untrace)

Monitor Execution without Trace where n is the number of program steps, c is the utility entry address, W instructs SID not to trace subroutines.

V (Value)

Display the value of the next available location in memory (NEXT), the next location after the largest file read in (MSZE), the current value of the Program counter (PC), and the address of the end of available memory (END).

Wfilespec,s,f (Write)

Write the contents of a contiguous block of memory to filespec where f is finish address.

X[f][r]   (Examine)

Examine/alter CPU state where f is flag bit C,Z,M,E or I, r is register A,B,D,H,S or P.

SID UTILITIES

SID
UTILITIES

SID utilities, HIST.UTL and TRACE.UTL are special programs that operate with SID to provide additional debugging faclities.

The mechanisms for system initialization, data collection, and data display are described in the CP/M SID User's Guide.

The HIST utility creates a histogram (bar graph) showing the relative frequency of execution of code within selected program segments of the test program. The HIST utility allows you to monitor those sections of code that execute most frequently.

The TRACE utility obtains a backtrace of the instructions that led to a particular breakpoint address in a program under test. You can collect the addresses of up to 256 instructions between pass points in U or Tmodes.

SID EXAMPLES

SID
EXAMPLES

```
A>SID
```

CP/M 3 loads SID from drive A into memory. SID displays the # prompt when it is ready to accept commands.

```
A>B:SID SAMPLE.HEX
```

CP/M 3 loads SID and the program file SAMPLE.HEX into memory from drive B.

CNTRLCHARS

Control
Character   Function

CTRL-C stops executing program when entered at the system prompt or after CTRL-S.

CTRL-E forces a physical carriage return without sending command to CP/M 3.

CTRL-I same as the TAB key.

CTRL-H delete character to the left of cursor.

CTRL-J moves cursor to the left of the command line and seds command to CP/M 3. Line feed, has same effect as carriage return.

CTRL-K deletes character at cursor and all characters to the right.

CTRL-M same as cariage return.

CTRL-P echoes console output to the lit device.

CTRL-Q restarts screen scrolling after a CTRL-S.

CTRL-R retypes the characters to the left of the cursor on a new line; updates the command line buffer.

CTRL-S stops screen scrolling.

CTRL-U updates the command line buffer to contain the characters to the left of the cursor; deletes current line.

CTRL-X deletes all characters to the left of the cursor.

COMMANDS

CP/M 3 Command Format:

A>COMMAND [command tail] <cr>

A CP/M 3 command line is composed of a command, an optional command tail, and a carriage return. The command is the name of filename of a program to be executed. The optional command tail can consist of a drive specification, one or more file specifications, and some options or parameters.

COMMANDS

COMMAND CONVENTIONS

The following special symbols define command syntax.

[] surrounds an optional item.

" separates alternative items in a command line.

<cr> indicates a carriage return.

↑ indicates the Control Key.

n substitute a number for n.

s substitute a string (group) of characters for s.

o substitute an option or option list for o.

[] type square brackets to enclose an option list.

() type parens to enclose a range of options within an option list.

RW Read-Write attribute - opposite of RO

RO Read-Only attribute- opposite of RW

SYS System attribute - opposite of DIR

DIR Directory attribute - opposite of SYS

... preceding element can be repeated as many times as desired.

* wildcard: replaces all or part of a filename and/or filetype.

? wildcard: replaces any single character in the same position of a filename and/or filetype.

ED

Format:

```
     ED input-filespec [d:"output-filespec]
```

Explanation:

Character file editor. To redirect or rename the new version of the file specify the destination drive or destination filespc.

ED

COMMANDS

ED Command Summary

Command   Action

nA
   append n lines from original file to memory buffer

ØA
   append file until buffer is one half full

#A
   append file until buffer is full (or end of file)

B, -B
   move CP to the beginning (B) or bottom (-B) of buffer

nC, -nC
   move CP n characters forward (C) or back (-C) through buffer

nD, -nD
   delete n characters before (-D) or from (D) the CP

E save new file and return to CP/M-86

Fstring[↑Z]
   find character string

H save new file, reedit, use new file as original file

I<cr>
   enter insert mode

Istring[↑Z]
   insert string at CP

Jsearch_str↑Zins_str↑Zdl_to_str
   juxtapose string

```
nK, -nK
    delete (kill) n lines from the CP

nL, -nL, ØL
    move CPn lines

nMcommands
    execute commands n times

n, -n
    move CP n lines and display that line

n:
    move to line n

:ncommand
    execute command through line n

Nstring[↑Z]
    extended find string

O  return to original file

nP, -nP
    move CP 23 lines forward and
    display 2 lines at console

Q  abandon new file, return to
   CP/M-86

R[↑Z]
    read X$$$$$$$.LIB file into buffer

Rfilespec[↑Z]
    read filespec into buffer

Sdelete string↑Zinsert string substitute string

nT, -nT, ØT
    type n lines

U, -U
    upper-case tanslation

V, -V
    line numbering on/off

ØV
    display free buffer space

nW
    write n lines to new file
```

ØW

write until buffer half empty

nX

write or append n lines to X$$$$$$$.LIB

nXfilespec[↑Z]

write n lines to filespec; append if previous xcommand applied to same file

Øx[↑Z]

delete file X$$$$$$$.LIB

Øxfilespec[↑Z]

delete filespec

nZ

wait n seconds

Note: CP points to the current character being referenced in the edit buffer. Use [↑Z] to separate multiple commands on the same line.

USER

Syntax:

```
        USER [number]
```

Explanation:

The USER command sets the current user number. The disk directory can be divided into distinct groups according to a"User Number." User numbers range from Ø through 15.

USER
  EXAMPLES

```
A>USER
Enter User#:5
5A>
```

The current user number is now 5 on drive A.

```
A>USER 3
3A>
```

This command changes the current User Number to 3.

MAC

Syntax:

```
MAC filename [$options]
```

Explanation:

MAC, the CP/M 3 macro assembler, reads assembly language statements from a file of type .ASM, assembles the statements, and produces three output files with the input filename and filetypes of .HEX, .PRN, and .SYM. Filename.HEX contains INTEL hexadecimal format object code. Filename.PRN contains an annotated source listing that you can print or examine at the console. Filename.SYM contains a sorted list of symbols defined in the program.

MAC
OPTIONS

Use options to direct the input and output of MAC. Use a letter with the option to indicate the source and destination drives, and console, printer, or zero output. Valid drive names are A through O. X, P and Z specify console, printer, and zero output, respectively.

Assembly Options That Direct Input/Output

A source drive for .ASM file (A-O)

H destination drive for .HEX file (A-O, Z)

L source drive for macrolibrary .LIB files called by the MACLIB statement.

P destination drive for .PRN file (A-O, X, P, Z)

S destination drive for .SYM file

Assembly Options That Modify Contents Of Output File

+L lists input lines read from macrollibrary .LIB files
-L suppresses listing (default)

+M lists all macro lines as they are processed during assembly
-M suppresses all macro lines as they are read during assembly
*M lists only hex generated by macro expansions

+Q lists all LOCAL symbols in the symbol list
-Q suppresses all LOCAL symbols in the symbol list (default)

+S appends symbolfile to print file
-S suppresses creation of symbol file

+1 produces a pass 1 listing for macro debugging in .PRN file
-1 suppress listing on pass 1 (default)

HEXCOM

Syntax:

```
     HEXCOM filename
```

Explanation:

The HEXCOM Command generates a command file (filetype.COM) from a .HEX input file. It names the output file with the same filename as the input file but with filetype .COM. HEXCOM always looks for a file with filetype .HEX.

Example:

```
    A>HEXCOM B:PROGRAM
```

Generates a command file PROGRAM.COM from the input hex file PROGRAM.HEX.

FILESPEC

FILESPEC FORMAT

CP/M 3 identifies every file by its unique file specification, which can consist of four parts: the drive specification, the filename, and the filetype. The term "filespec" indicates any valid combination of the four parts of a file specification, all separated by their appropriate delimiters. A colon must follow a drive letter. A period must precede a filetype.

The symbols and rules for the parts of a file specification follow:

d: drivespec optional single alpha character (A-P) filename
filename 1-8 letters and/or numbers

typ   filetype optional Ø-3 letters and/or numbers

Valid combinations of the elements of a CP/M 3 file specification are:

```
filename
d:filename
filename.typ
d:filename.typ
```

If you do not include a drive specifier, CP/M 3 automatically uses he default drive.

Some CP/M 3 commands accept wildcard (* and ?) characters in the filename and/or filetype parts of the command tail. A wildcard in the command line can in one command reference many matching files on the default or specified user number and drive. (See Commands).

RMAC

Syntax:

```
  RMAC filespec [$Rd ¦ $Sd ¦ $Pd]
```

Explanation:

RMAC, a relocatable macro assembler, assembles .ASM files of into .REL files that you can link to create .COM files.

RMAC
OPTONS

RMAC options specify the destination of the output files. Replace d with the destination drive letter for the output files.

```
Option         d=output option

R- drive for REL file (A-O, Z)
S- drivefor SYM file  (A-O, X, P, Z)
P- drive for PRN file (A-O, X, P, Z)
```

A-O specifies drive A-O.
X means output to the console.
P means output to the printer.
Z means zero output.

LINK

Syntax:

```
    LINK d:[filespec,[[options]]=]filespec[[options]][,...]
```

Explanation:

LINK combines relocatable object modules such as those produced by RMAC and PL/I-80 into a .COM file ready for execution. Relocatable files can contain external references and publics. Relocatable files can reference modules in library files. LINK searches the library files and includes the referenced modules in the output file.

See the CP/M 3 Programmer's Utilities Guide for a complete description of LINK-80.

LINK
OPTIONS

Use LINK option switches to control execution parameters. Link options follow the file specifications and are enclosed within square brackets. Multiple switches are separated by commas.

LINK-80 Options

A   Additional memory; reduces buffer space and writes temporary workspace to disk

B   BIOS link in banked CP/M 3 system.
1.   Aligns data segment on page boundary.
2.   Puts length of code segment in header.
3.   Defaults to .SPR filetype.

Dhhhh
Data origin sets memory origin for common and data area

Gn   Go; set start address to label n

Lhhhh
Load; change default load address of module to hhhh. Default 0100H

Mhhhh
Memory size; Define free memory requirements for MP/M modules.

NL   No listing of symbol table at console

NR   No symbol table file

OC   Output .COM command file. Default

OP Output .PRL page relocatable file for execution under MP/M in relocatable segment

OR Output .RSP resident system process file for execution under MP/M.

OS Output .SPR system page relocatable file for execution under MP/M

Phhhh

Program origin; changes default program origin address to hhhh. Default is Ø1ØØH.

Q Lists symbols with leading question mark

S Search preceding file as a library

$d

Destination of console messages d can be X (console), Y (printer), or Z (zero output). Default is X.

$Id

Source of intermediate files; d is disk drive A-P. Default is current drive.

$Ld

Source of library files d is disk drive A-P. Default is current drive.

$Od

Destination of object file; d can be Z or disk drive A-P. Default is to same drive as first file in the LINK-8Ø command.

$Sd

Destination of symbol file; d can be Y or Z or disk drive A-P. Default is to same drive as first file in LINK-8Ø command.