```
//1commands

   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 or
   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.

   //2conventions
                        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.


   //1cntrlchars

   Control Character                          Function

   CTRL-A     moves cursor one character to the left.  Banked  system
              only.

   CTRL-B     moves cursor from beginning to end of command line  and
              back without affecting command.  Banked system only.

   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-F     moves cursor one character to the right. Banked  system
              only.

   CTRL-G     deletes character at current cursor position if in  the
              middle of a line.  Banked system only.

   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 sends
          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 carriage return.

CTRL-P    echoes console output to the list 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-W    recalls previous command line if current line is empty;
          otherwise moves cursor to end of line. CTRL-J,-M,-R,-U
          and RETURN update the command line buffer for recall
          with CTRL-W. Banked system only.

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

///1COPYSYS

Syntax:

     COPYSYS

Explanation:

COPYSYS copies the CP/M 3 system from a CP/M 3 system diskette to
another diskette. The new diskette must have the same format as
the original system diskette.

Example:

     A>COPYSYS

///1DATE

Syntax:

     DATE {CONTINUOUS}
     DATE {time-specification}
     DATE SET

Explanation:

The DATE command lets you display and set the date and time of
day.
```

///2Examples

A>DATE

Displays the current date and time.

A>DATE C

Displays the date and time continuously.

A>DATE 08/14/82 10:30:0

Sets the date and time.

A>DATE SET

Prompts for date and time entries.

///1DEVICE

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.

///2Options

[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	

///2Examples

A>DEVICE

Displays the physical devices and current assignments of

A > DEVICE AUX: = SERIAL [19200]

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.

.// 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.

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.

Examples

A>DIR

   Displays all files in user 0 on drive A that have the Directory attribute.

A>DIR B:

   Displays all DIR files in user 0 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.

3B>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 0 on drive A that have the system (SYS) attribute.

A>DIRS *.COM

   Displays all SYS files with filetype COM on drive A in user 0. A command (.COM) file in user 0 with the system attribute can be accessed from any user number on that drive, and from any drive in the search chain (see SETDEF).

. 2WithOptions

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.

///3Options

Option	Function
ATT	display 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 the password protection mode and the time stamps. If there is no directory label, DIR displays two file entries on a line, omitting the password and time stamp columns. The display is alphabetically sorted. (See SET for a description of file attributes, directory labels, passwords 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 (1024
              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=(0,1,...,15)
              displays files under the user numbers specified.

///3Examples

A>DIR C: [FULL]

     Displays full set of characteristics for all files in user 0
     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 0 on drive D with Read-Write
     and System attributes.

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

     Displays all the files in all user numbers (0-15) 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:*.*.

///1DUMP

Syntax:

        DUMP filespec

Explanation:

DUMP displays the contents of a  file  in  hexadecimal  and  ASCII
format.

Example:

      A>DUMP ABC.TEX

/./ied

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 filespec.

//2commands

                         ED Command Summary


Command                  Action

nA
      append n lines from original file to memory buffer

0A
      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^Zdel_to_str
          juxtapose strings

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

    nL, -nL, 0L
          move CP n 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 23 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, 0T
          type n lines

    U, -U
          upper-case translation
    V, -V
          line numbering on/off
    0V
          display free buffer space
    nW
          write n lines to new file
    0W
          write until buffer is half empty
    nX
          write or append n lines to X$$$$$$$.LIB
```

```
nX(filespec{^Z})
        write n lines to filespec;
        append if previous X command applied to same file

0X{^Z}
        delete file X$$$$$$$.LIB

0X(filespec{^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.

  ///2Examples

      A>ED TEST.DAT
      A>ED TEST.DAT B:
      A>ED TEST.DAT TEST2.DAT
      A>ED TEST.DAT B:TEST2.DAT

///1erase

Syntax:

      ERASE {filespec} [CONFIRM]

Explanation:

The ERASE command removes  one  or  more  files  from  the
directory  of  a  disk.  Wildcard  characters are accepted in the
filespec.  Directory and data space are  automatically  reclaimed
for  later  use  by  another  file.  The  ERASE  command  can  be
abbreviated to ERA.

///2Option

    [CONFIRM]  option  informs  the  system  to  prompt  for
               verification  before  erasing  each  file  that
               matches  the  filespec.  CONFIRM  can  be
               abbreviated to C.

///2Examples

A>ERASE X.PAS

      Removes the file X.PAS from the disk in  drive A.

A>ERA *.PRN
Confirm (Y/N)?Y

      All files with the filetype PRN are removed from  the  disk
      in drive A.

B>ERA A:MY*.* [CONFIRM]

      Each file on drive A with a filename that begins with MY  is
      displayed  with a question mark for confirmation.  Type Y to
```

erase the file displayed, N to keep the file.

```
A>ERA B:*.*
Confirm (Y/N)?Y
```

All files on drive B are removed from the disk.

//1filespec

FILESPEC FORMAT

CP/M 3 identifies every file by its unique file specification, which can consist of four parts: the drive specification, the filename, the filetype and the password. 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. A semicolon must precede a password.

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	0-3 letters and/or numbers
password	password	optional	0-8 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
filename;password
d:filename;password
filename.typ;password
d:filename.typ;password
```

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

//1GENCOM

Syntax:

```
GENCOM [COM-filespec] [RSX-filespec] ...
                [[LOADER | NULL | SCB=(offset,value)]]
```

Explanation:

The GENCOM command creates a special COM file with attached RSX files. The GENCOM command can also restore a previously GENCOMed file to the original COM file without the header and

RSX's. GENCOM can also attach header records to COM files.

///2Options

LOADER sets a flag to keep the program loader active.

NULL indicates that only RSX files are specified. GENCOM creates a dummy COM file for the RSX files. The output COM filename is taken from the filename of the first RSX-filespec.

SCB=(offset,value)
sets the System Control Block from the program by using the hex values specified by (offset,value).

///2Examples

```
A>GENCOM MYPROG PROG1 PROG2
```

Generates a new COM file MYPROG.COM with attached RSX's PROG1 and PROG2.

```
A>GENCOM PROG1 PROG2 [NULL]
```

Creates a COM file PROG1.COM with RSX's PROG1 and PROG2.

```
A>GENCOM MYPROG
```

GENCOM takes MYPROG.COM, strips off the header and deletes all attached RSX's to restore it to its original COM format.

```
A>GENCOM MYPROG PROG1 PROG2
```

GENCOM looks at the already-GENCOMed file MYPROG.COM to see if PROG1.RSX and PROG2.RSX are already attached RSX files in the module. If either one is already attached, GENCOM replaces it with the new RSX module. Otherwise, GENCOM appends the specified RSX files to the COM file.

///1GET

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.

//.2Options

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.

//.2Examples

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 console for console input.

A>GET FILE XINC [SYSTEM]

Immediately directs the system to get subsequent console input from file XINC because it includes the SYSTEM option. The system reverts back to the console for console input when it reaches the end of file in XINC. Or XINC 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.

//.1HELP

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>SET PASSWORD
    HELP>.PASSWORD
    HELP>.
    HELP><cr>
```

///1HEXCOM

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.

///1INITDIR

Syntax:

```
   INITDIR {d:}
```

Explanation:

The INITDIR Command initializes a disk directory to allow date and time stamping of files on that disk. INITDIR can also recover time/date directory space.

Example:

```
A>INITDIR D:

INITDIR WILL ACTIVATE TIME-STAMPS FOR SPECIFIED DRIVE.
Do you want to re-format the directory on D: (Y/N)?Y
```

///1LIB

Syntax:

```
        LIB filespec{[I|M|P|D]}
        LIB filespec{[I|M|P]}=filespec{modifier}
                                {,filespec{modifier} ... }
```

Explanation:

A library is a file that contains a collection of object modules. Use the LIB utility to create libraries, and to append, replace, select or delete modules from an existing library. Use LIB to obtain information about the contents of library files.

LIB creates and maintains library files that contain object modules in Microsoft REL file format. These modules are produced by Digital Research's relocatable macro-assembler program, RMAC, or any other language translator that produces modules in Microsoft REL file format.

You can use LINK-80 to link the object modules contained in a library to other object files. LINK-80 automatically selects from the library only those modules needed by the program being linked, and then forms an executable file with a filetype of COM.

///2Options

I	The INDEX option creates an indexed library file of type .IRL. LINK-80 searches faster on indexed libraries than on non-indexed libraries.
M	The MODULE option displays module names.
P	The PUBLICS option displays module names and the public variables for the new library file.
D	The DUMP option displays the contents of object modules in ASCII form.

///2Modifiers

Use modifiers in the command line to instruct LIB to delete, replace, or select modules in a library file. Angle brackets enclose the modules to be deleted or replaced. Parentheses enclose the modules to be selected.

LIB Modifiers

Delete	<module=>
Replace	<module=filename.REL>

```
                              If module name and filename are the
                              same this shorthand can be used:

                              <filename>

                  Select      (modFIRST-modLAST,mod1,mod2,...,modn)

../ZExamples

A>LIB TEST4[P]

    Displays  all  modules and publics in TEST4.REL.

A>LIB TEST5[P]=FILE1,FILE2

    Creates TEST5.REL from FILE1.REL and FILE2.REL and  displays
    all modules and publics in  TEST5.REL.

A>LIB TEST=TEST1(MOD1,MOD4),TEST2(C1-C4,C6)

    Creates a library file TEST.REL from modules in  two  source
    files.  TEST1.REL  contributes  MOD1 and MOD4.  LIB extracts
    modules C1, C4, and all the  modules  located between  them,
    as well as module C6 from TEST2.REL.

A>LIB FILE3=FILE3<MODA>

    Creates  FILE3.REL  from  FILE3.REL, omitting MODA which  is
    a module in FILE3.REL.

A>LIB FILE6=FILE5<MODA=FILEB.REL>

    Creates  FILE6.REL  from FILE5.REL, FILEB.REL replaces MODA.

A>LIB FILE6=FILE5<THISNAME>

    Module   THISNAME  is  in  FILE5.REL.   When   LIB   creates
    FILE6.REL  from FILE5.REL the file THISNAME.REL replaces the
    similarly named module THISNAME.

A>LIB FILE1[I]=B:FILE2(PLOTS,FIND,SEARCH-DISPLAY)

    Creates FILE1.IRL on drive A  from  the  selected  modules
    PLOTS,   FIND,   and   modules  SEARCH  through  the  module
    DISPLAY, in FILE2.REL on drive B.

.. 1LINK

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.

///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 data 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 0100H.
Q	Lists symbols with leading question mark
S	Search preceding file as a library
$Cd	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 of object file;
                       d can be Z or disk drive A-P.
                       Default is to same drive as
                       first file in the LINK-80 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-80 command.
```

///2Examples

A>LINK b:MYFILE[NR]

LINK-80 on drive A uses as input MYFILE.REL on drive B and produces the executable machine code file MYFILE.COM on drive B. The [NR] option specifies no symbol table file.

A>LINK m1,m2,m3

LINK-80 combines the separately compiled files m1, m2, and m3, resolves their external references, and produces the executable machine code file m1.COM.

A>LINK m=m1,m2,m3

LINK-80 combines the separately compiled files m1, m2, and m3 and produces the executable machine code file m.COM.

A>LINK MYFILE,FILE5[s]

The [s] option tells LINK-80 to search FILE5 as a library. LINK-80 combines MYFILE.REL with the referenced subroutines contained in FILE5.REL on the default drive A and produces MYFILE.COM on drive A.

///1mac

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.

^/^2Examples

```
    A>MAC SAMPLE

    A>MAC SAMPLE $PB AA HB SX
```

^/^2Options

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 thru D. X, P and Z specify console, printer, and zero output, respectively.

Assembly Options That Direct Input/Output

A	source drive for .ASM file (A-D)
H	destination drive for .HEX file (A-D, Z)
L	source drive for macrolibrary .LIB files called by the MACLIB statement.
P	destination drive for .PRN file (A-D, X, P, Z)
S	destination drive for .SYM file

Assembly Options That Modify Contents Of Output File

+L	lists input lines read from macrolibrary .LIB files
-L	suppresses listing (default)
+M	lists all macro lines as they are processed during assembl
-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 symbol file 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)

^/^1PATCH

Syntax:

```
    PATCH filename{.typ} {n}
```

Explanation:

The PATCH command displays or installs patch number n to the CP/M 3 system or command files. The patch number n must be between 1 and 32 inclusive.

Example:

```
A>PATCH SHOW 2
```

Patches the SHOW.COM system file with patch number 2.

///1PIP (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.
///2Examples

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 SET FROM USER :

```
        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
```

///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 :00 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=60)
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.

///PUT

Syntax:

```
     PUT CONSOLE {OUTPUT TO} FILE filespec {option} | CONSOLE
     PUT PRINTER {OUTPUT TO} FILE filespec {option} | PRINTER
     PUT CONSOLE {OUTPUT TO} CONSOLE
     PUT PRINTER {OUTPUT TO} PRINTER
```

Explanation:

PUT puts console or printer output to a file for the next command entered at the console, until the program terminates. Then console output reverts to the console. Printer output is directed to a file until the program terminates. Then printer output is put back to the printer.

PUT with the SYSTEM option directs all subsequent console/printer output to the specified file. This option terminates when you enter the PUT CONSOLE or PUT PRINTER command.

//2Options

[{ECHO | NO ECHO} {FILTER | NO FILTER} | {SYSTEM}]

ECHO	specifies that output is echoed to the console. This is the default option when you direct console output to a file.
NO ECHO	specifies that file output is not echoed to the console. NO ECHO is the default for the PUT PRINTER command.
FILTER	specifies filtering of control characters, which means that control characters are translated to printable characters. For example, an ESCape character is translated to ^[.
NO FILTER	means that PUT does not translate control characters. This is the default option.
SYSTEM	specifies that system output as well as program output is written to the file specified by filespec. Output is written to the file until a subsequent PUT CONSOLE command redirects console output back to the console.

//2Examples

```
A>PUT CONSOLE OUTPUT TO FILE XOUT [ECHO]
```

Directs console output to file XOUT with the output echoed to the console.

```
A>PUT PRINTER OUTPUT TO FILE XOUT
A>MYPROG
```

Directs the printer output of program MYPROG to file XOUT. The output is not echoed to the printer.

```
A>PUT PRINTER OUTPUT TO FILE XOUT2 [ECHO,SYSTEM]
```

Directs all printer output to file XOUT2 as well as to the printer (with ECHO option), and the PUT is in effect until you enter a PUT PRINTER OUTPUT TO PRINTER command.

```
A>PUT CONSOLE OUTPUT TO CONSOLE
```

Directs console output back to the console.

```
A>PUT PRINTER OUTPUT TO PRINTER
```

Directs printer output back to the printer.

//1RENAME

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.

//.2Examples

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
```

//.1RMAC

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.

//.2Options

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- drive for 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.

///2Example

```
A>RMAC TEST $PX SB RB
```

Assembles the file TEST.ASM from drive A, sends the listing file (TEST.PRN) to the console, puts the symbol file (TEST.SYM) on drive B and puts the relocatable object file (TEST.REL) on drive B.

///1SAVE

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.

///2Example

```
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.

///1SET

Syntax:

```
SET [options]
SET d: [options]
SET filespec [options]
```

Explanation:

SET initiates password protection and time stamping of files. It also sets the file and drive attributes Read-Write, Read-Only, DIR and SYS. It lets you label a disk and password protect the label. To enable time stamping of files, you must first run INITDIR to format the disk directory.

///2Label

Syntax:

```
SET {d:} [NAME=labelname.typ]
SET [PASSWORD=password]
SET [PASSWORD=<cr>
```

///3Examples

A>SET [NAME=DISK100]

Labels the disk in the default drive as DISK100.

A>SET [PASSWORD=SECRET]

Assigns SECRET to the disk label.

A>SET [PASSWORD=<cr>

Nullifies the existing password.

///2Passwords

```
SET [PROTECT=ON]
SET [PROTECT=OFF]
SET filespec [PASSWORD=password]
SET filespec [PROTECT=READ]
SET filespec [PROTECT=WRITE]
SET filespec [PROTECT=DELETE]
SET filespec [PROTECT=NONE]
SET filespec [attribute-options]
```

///3Modes

Password Protection Modes

Mode	Protection
READ	The password is required for reading, copying writing, deleting or renaming the file.
WRITE	The password is required for writing, deleting or renaming the file. You do not need a password to read the file.
DELETE	The password is only required for deleting or renaming the file. You do not need a password to read or modify the file.
NONE	No password exists for the file. If a password password exists, this modifier can be used to delete the password.

///2Attributes

RO	sets the file attribute to Read-Only.
RW	sets the file attribute to Read-Write.
SYS	sets the file attribute to SYS.
DIR	sets the file attribute to DIR.
ARCHIVE=OFF	means that the file has not been backed up (archived).
ARCHIVE=ON	means that the file has been backed up (archived). The Archive attribute can be turned on by SET or by PIP when copying a group of files with the PIP [A] option. SHOW and DIR display the Archive option.
F1=ON|OFF	turns on or off the user-definable file attribute F1.
F2=ON|OFF	turns on or off the user-definable file attribute F2.
F3=ON|OFF	turns on or off the user-definable file attribute F3.
F4=ON|OFF	turns on or off the user-definable file attribute F4.

///3Examples

SET [PROTECT=ON]

Turns on password protection for all the files on the disk. You must turn on password protection before you can assign passwords to files.

SET [PROTECT=OFF]

Disables password protection for the files on your disk.

```
A>SET MYFILE.TEX [PASSWORD=MYFIL]
```

MYFIL is the password assigned to file MYFILE.TEX.

```
B>SET *.TEX [PASSWORD=SECRET, PROTECT=WRITE]
```

Assigns the password SECRET to all the TEX files on drive B. Each TEX file is given a WRITE protect mode to prevent unauthorized editing.

```
A>SET MYFILE.TEX [RO SYS]
```

Sets MYFILE.TEX to Read-Only and SYStem.

///2Default

```
A>SET [DEFAULT=dd]
```

Instructs the system to use dd as a password if you do not enter a password for a password-protected file.

///2Time-Stamps

Syntax:

```
SET [CREATE=ON]
SET [ACCESS=ON]
SET [UPDATE=ON]
```

Explanation:

The above SET commands allow you to keep a record of the time and date of file creation and update, or of the last access and update of your files.

///3Options

[CREATE=ON]	turns on CREATE time stamps on the disk in the default or specified drive. To record the creation time of a file, the CREATE option must be turned on before the file is created.
[ACCESS=ON]	turns on ACCESS time stamps on the disk in the default or specified drive. ACCESS and CREATE options are mutually exclusive; only one can be in effect at a time. If you turn on the ACCESS time stamp on a disk that previously had CREATE time stamp, the CREATE time stamp is automatically turned off.
[UPDATE=ON]	turns on UPDATE time stamps on the disk in the default or specified drive. UPDATE time stamps record the time the file was last modified.

///3Examples

```
A>SET [ACCESS=ON]
```

```
          A>SET [CREATE=ON,UPDATE=ON]

///2Drives

Syntax:

          SET [d:] [RO]
          SET [d:] [RW]


Example:

A>SET B: [RO]

      Sets drive B to Read-Only.

///1SETDEF

Syntax:

          SETDEF [ d: [,d: [,d: [,d:]]]] [[ TEMPORARY = d: ] '
                                          [ ORDER = (typ [,typ]) ]]
          SETDEF [DISPLAY ' NO DISPLAY]

          SETDEF [PAGE | NOPAGE]

Explanation:

SETDEF allows the user to display or define  up  to  four  drives
for  the program search order, the drive for temporary files, and
the file type search order.   The   SETDEF   definitions   affect
only   the   loading   of programs  and/or  execution  of  SUBMIT
(SUB) files.  SETDEF turns on/off the system Display and  Console
Page  modes. When   on, the system displays the location and name
of programs loaded or SUBmit  files  executed,  and  stops  after
displaying one full console screen of information.

///2Examples

A>SETDEF

      Displays current SETDEF parameters.

A>SETDEF [TEMPORARY=C:]

      Sets disk drive C as the drive  to  be  used  for  temporary
      files.

A>SETDEF C:,*

      Tells the system to search for a program on drive  C,  then,
      if not found, search for it on the default drive.

A>SETDEF [ORDER=(SUB,COM)]

      Instructs the system to search for a SUB  file  to  execute.
      If no SUB file is found, search for a COM file.

A>SETDEF [DISPLAY]
```

Turns on the system display mode. Henceforth, the system displays the name and location of programs loaded or submit files executed.

A>SETDEF [NO DISPLAY] Turns off the system Display mode.

///1SHOW

Syntax:

    SHOW {d:}[[SPACE |LABEL |USERS |DIR |DRIVE]]

Explanation:

The SHOW command displays the following disk drive information:

Access mode and the amount of free disk space
Disk label
Current user number and
Number of files for each user number on the disk
Number of free directory entries for the disk
Drive characteristics

///2Examples

A>SHOW

    A>SHOW [SPACE]

    Instructs the system to display access mode and amount of space left on logged-in drives.

A>SHOW B:

    Show access mode for drive B and amount of space left on drive B.

A>SHOW B:[LABEL]

    Displays label information for drive B.

A>SHOW [USERS]

    Displays the current user number and all the users on drive A and the corresponding number of files assigned to them.

A>SHOW C:[DIR]

    Displays the number of free directory entries on drive C.

A>SHOW [DRIVE]

    Displays the drive characteristics of drive A.

///1SID

Syntax:

    SID {pgm-filespec} {,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.

///2Commands

Command		Meaning
As	(Assemble)	Enter assembly language statements s is the start address
Cs{b{,d}}	(Call)	Call to memory location from SID 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 W is a 16-bit word format s is the start address f is the finish address
Epgm-filespec {,sym-filespec}	(Load)	Load program and symbol table for execution
E*sym-filespec	(Load)	Load a symbol table file
Fs,f,d	(Fill)	Fill memory with constant value s is the start address f is the finish address d is an eight-bit data item
G{p}{,a{,b}}	(Go)	Begin Execution p is a start address a is a temporary breakpoint
H	(Hex)	Displays all symbols with addresses in Hex
H.a		Displays hex, decimal, and 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 command line
L{s}{,f}	(List)	List 8080 mnemonic instructions s is the start address f is the finish address
Ms,h,d	(Move)	Move Memory Block 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
		p is a permanent breakpoint address
		c is initial value of pass counter
Rfilespec[,d]	(Read)	Read Code/Symbols
		d is an offset to each address
S{W}s	(Set)	Set Memory Values
		s is address where value is sent
		W is 16 bit word
T[n[,c]]	(Trace)	Trace Program Execution
		n is the number of program steps
		c is the utility entry address.
T{W}[n[,c]]	(Trace)	Trace Without Call
		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
		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.
		f is finish address
X{f}{r}	(Examine)	Examine/alter CPU state.
		f is flag bit C,Z,M,E or I.
		r is register A,B,D,H,S or P.

///2Examples

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.

///2Utilities

SID utilities. HIST.UTL and TRACE.UTL are special programs that operate with SID to provide additional debugging facilities. 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 T modes.

///1SUBMIT

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).

///2Subfile

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 ($0-$9)
     Any data input line
     Any program input line with parameters ($0 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:
```

///2Execute

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:
```

///2PROFILE.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.

///1TYPE

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.

///2Examples

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.

///1USER

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 0 through 15.

///2Examples

```
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.

///1XREF

Syntax:

          XREF (d:) filename ($P)

Explanation:

XREF provides  a  cross-referance  summary  of  variable  usage
in   a   program.  XREF requires the .PRN and .SYM files produced
by MAC or RMAC for input to the program.  The SYM and PRN  files
must  have  the same filename as the filename in the XREF command
tail.  XREF outputs a file of type .XRF.

Examples:

A>XREF b:MYPROG

A>XREF b:MYPROG $P
```