CPM3.TXT rev 96-11-07

Herne Data Systems Ltd.,
PO Box 250, Tiverton, ON N0G 2T0 CANADA.
Voice/fax 519-366-2732,
e-mail herne@herne.com,
internet: http://www.herne.com

* *

External Utilities

External CP/M commands, commonly referred to as 'transient programs' are used to perform functions which are too complex or specialized for the resident command interpreter. The external commands include extensions to the CCP internal commands as well as utilities and user applications. The programs reside on disk in the form of a file with the file type .COM. When the program is called from the CCP command line, it is loaded from disk into the Transient Program Area (TPA), beginning at address 0100 hex in Bank 1 of RAM.

This document outlines the standard CP/M external utilities as supplied with your system disk. It does NOT include specific application programs such as word processors or spreadsheets. The following notation is used in the syntax descriptions, according to standard CP/M conventions:

{ } Curly braces are used to indicate optional input, generally in the form of a file specification. The user does NOT type in the braces.

[ ] Square brackets are used to indicate 'switches' to enable or disable various options. The user MUST type in the left hand (leading) square bracket, while the right hand (trailing) bracket is optional. In most cases, the options can be specified with only one or two characters, instead of spelling out the entire word and several switches may be included in the same set of brackets.

| The ASCII pipe (or vertical bar) is used to separate various optional formats in the command line specification. The pipe is NOT typed in on the command line. For example, if the command line states: {NAMES | VALUES}, you may use either the option NAMES or the VALUES on the command line, but normally not both simultaneously.

unambiguous filespec The unambiguous filespec, or ufn, is used to uniquely specify a single file. It consists of an optional logical drive specifier (such as D:) followed by a primary filename of from one to eight ASCII characters, a period and a filetype extension of up to three characters, such as A:TESTFILE.DOC. The ufn must NOT contain any of the following special characters:

ambiguous filespec The ambiguous filespec, or afn, is similar to the ufn, except it MAY contain the '*' and/or '?' wild card characters. It is used to specify a 'pattern' for performing an operation on a group of files with similar names. The '*' character will match all remaining characters in a filename or extension, regardless of the number of characters specified. (For example, the afn A:TE*.C?M will match all files on drive A: which have (two or more characters in the filename AND begin with TE) AND (a file extension that starts with a C and ends with an M) such as: TEST.COM, TEXT.CDM, TENT.CM, but NOT examples such as CTEXT.COM or TEST.OCM.)

DATE

The DATE command lets you display or set the current system date and time of day. The date is displayed in the MM/DD/YY format, where MM is the month (1 = January, 12 = December), DD is the day; and YY is the year. The time is displayed in a 24 hour military format, and includes hours, minutes and seconds. The general syntax is:

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

With the {time-specification}, all fields must be included. For example:

Command	What it does
DATE	Displays the current date and time, then returns to the CCP command prompt..
DATE C	Displays the date and time continuously, until you press a key, then returns to the CCP command prompt.
DATE 06/12/89 11:30:0	Sets the date and time as specified (June 12, 1989, 11:30:00 am, in this case).
DATE SET	Prompts for new date and time entries.

DEVICE

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

The general syntax is:

```
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]
```

The device {options} are:

```
[ 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
```

The following baud rates are 'legal' but are not physically reliable with the C-128 CP/M mode due to processor speed, software overhead, etc.:

```
1800   2400     3600     4800
7200   9600     19200
```

The following are legal physical devices under the C128 CP/M implementation:

Name	Function
KEYS	keyboard
40COL	40 column monitor
80COL	80 column monitor
PTR1	Commodore Serial Printer (Serial Bus Device 4)
PTR2	Commodore Serial Printer (Serial Bus Device 5)
RS232	RS-232 User port (DEC and later versions only)

For example:

Command	What it does
DEVICE	Displays the physical devices and current assignments of the logical devices in the system.
DEVICE NAMES	Lists the supported physical devices with a summary of the device characteristics.
DEVICE VALUES	Displays the current logical device assignments.
DEVICE 80COL	Displays the attributes of the physical

device 80COL.

DEVICE CON — Displays the assignment of the logical device CON:

DEVICE LST:=NULL — Disconnects the list output logical device (LST:).

DEVICE CONSOLE [PAGE] — Displays the current console page width in columns and length in lines.

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 CCP resident commands and are outlined in the document "Internal Commands". 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 {d:} [options]
DIR {filespec} {filespec} ... [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,)	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...)	displays files under the user numbers specified.

For example:

Command	What it does
DIR C: [FULL]	Displays full set of characteristics for all files in the current user area on drive C:.
DIR C: [DATE]	Lists the files on drive C: and their dates, if time and date stamping has been initialized on the disk.
DIR D: [RW,SYS]	Displays all files drive D: with Read-Write and System attributes.
DIR [US=ALL, DR=ALL]	Displays all the files in all user numbers (0-15) on all on-line drives.
DIR [exclude] *.DAT	Lists all the files on current drive and user that do not have a filetype of .DAT.
DIR [SIZE] *.PLI *.COM	Displays all the files of type PLI, and

	COM in size display format.
DIR [DR=AL US=AL] TEST	Displays the filename TEST. if it is found on any drive in any user number. (This is a handy way to search for files if you do not remember where thay are).
DIR [size,rw] D:	Lists each Read-Write file that resides on drive D:, with its size in kilobytes. Note that D: is equivalent to D:*.*.

DUMP

DUMP displays the contents of a file in hexadecimal and ASCII format. The general syntax is:

```
DUMP unambiguous filespec
```

Example:

Command	What it does
DUMP ABC.TEX	Displays the contents of the file ABC.TEX

ED

ED is the CP/M standard character file editor. To redirect or rename the new version of the file specify the destination drive or destination filespec.

Format:

```
ED input-filespec {d:|output-filespec}
```

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
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 n pages (23 lines) forward (n) or backward (-n) and display 23 lines at console
Q	abandon new file, return to CP/M
R{^Z}	read X$$$$$$$.LIB file into buffer
Rfilespec{^Z}	read filespec into buffer
Sold_string^Znew_string	substitute string
nT, -nT, 0T	type n lines
U, -U	upper-case translation
V, -V	line numbering on/off
0V	display free buffer space
nW	write n lines to new file
0W	write until buffer is half empty
nX	write or append n lines to X$$$$$$$.LIB
nXfilespec{^Z}	write n lines to filespec; append if previous xcommand applied to same file

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

    0xfilespec{^Z}  delete filespec

    nZ              wait n seconds

Notes:    (a)  CP points to the current character being
               referenced in the edit buffer.
          (b)  Use {^Z} to separate multiple commands on the same
               line.

Examples:

Command             What it does

ED TEST.DAT         edits the file TEST.DAT on the current drive

ED TEST.DAT B:      edits the file TEST.DAT on the current drive
                    and puts the modified file on drive B: with
                    the same name.

ED TEST.DAT TEST2.DAT    edits the file TEST.DAT on the current
                         drive and renames the modified file to
                         TEST2.DAT on the current drive.


ERASE
=====

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.

ERASE with confirm or if the files are password protected, is a
transient utility ERASE.COM

Syntax:

     ERASE {filespec} [CONFIRM]


Options:

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

Examples:

Command             What it does

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

ERASE *.PRN
     Confirm (Y/N)?Y
                    All files with the filetype PRN are removed
                    from the disk in drive A.

ERASE 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.

```
ERASE B:*.*
     Confirm (Y/N)?Y
```

All files on drive B are removed from the disk.

GENCOM

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.

Syntax:

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

Options

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).

Examples:

Command	What it does
GENCOM MYPROG PROG1 PROG2	Generates a new COM file MYPROG.COM with attached RSX's PROG1 and PROG2.
GENCOM PROG1 PROG2 [NULL]	Creates a COM file PROG1.COM with RSX's PROG1 and PROG2.
GENCOM MYPROG	GENCOM takes MYPROG.COM, strips off the header and deletes all attached RSX's to restore it to its original COM format.
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.

GET

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.

Syntax:

GET {CONSOLE INPUT FROM} FILE filespec{[{ECHO|NO ECHO} | SYSTEM]}

To re-direct the system to the console for console input use the:

```
 GET CONSOLE INPUT FROM CONSOLE
```

command as a command line in the input file.

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. 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.

Examples:

Command	What it does
GET FILE XINPUT 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.
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.
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.

HELP

HELP displays information about CP/M topics and commands.

Syntax:

```
HELP {topic} {subtopic1 ... subtopic8} {[NOPAGE|LIST]}
```

HELP	displays a list of topics and provides summarized information for CP/M 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:

Command	What it does
HELP DATE	Displays help on the DATE command
HELP DIR OPTIONS	Displays help on the subtopic options of the DIR command.
HELP>.OPTIONS	
HELP>SET	
HELP>SET PASSWORD	
HELP>.PASSWORD	
HELP>.	
HELP><cr>	Enters the HELP system and displays help on the listed topics.

The text for the HELP screens are stored in the file HELP.HLP in the form of an indexed ASCII file. A poorly documented feature of

the HELP utility allows you to edit the text in the HELP.HLP and even add your own HELP topics. To convert the file into an editable ASCII file, use:

```
HELP [EXTRACT]
```

This creates the file HELP.DAT. Note that because of the large size of the HELP.HLP and HELP.DAT files, you will need a disk with a capacity of at least 180 k bytes, which is just slightly more than what is available with a C-128 Single Sided CP/M disk. Therefore, you must use either the 1750 RAM expander, or a double sided disk.

When you have finished with your editing, you can re-create the HELP.HLP file with:

```
HELP [CREATE]
```

HEXCOM

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.

Syntax:

```
HEXCOM filename
```

Example	What it Does
HEXCOM B:PROGRAM	Generates a command file PROGRAM.COM from the input hex file PROGRAM.HEX.

INITDIR

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. For a description of date and time stamps, see the document "The Disk System".

Syntax:

```
INITDIR {d:}
```

Example	What it Does
INITDIR B:	INITDIR WILL ACTIVATE TIME-STAMPS FOR SPECIFIED DRIVE. Do you want to re-format the directory on B: (Y/N)?Y

Sets up the disk in drive B: for date/time stamping.

KEYFIG

The purpose of the KEYFIG program is to allow you to alter the definition of almost ANY key on the keyboard. The only keys that you CANNOT modify are: the SHIFT keys, the SHIFT LOCK key, the CONTROL key, the 40/80 DISPLAY key and the COMMODORE key. At each step, options are presented in menu form. You can scroll through the options in the menus by using the up and down arrow keys at the top of the keyboard; pressing the return key selects the choice that is highlighted. At almost any point, you can exit the program by typing 'CTRL c'.

Setting Up Your Work File

The first thing you will be asked to do is set up your work file. You will be given a choice of 3 sources from which you can do this:

DEFAULT DEFINITIONS	which basically represent a standard set of key definitions.
CURRENT DEFINITIONS	which represent the most recently loaded set of definitions.
BOOT DISK DEFINITIONS	which represent the set of definitions stored on your boot disk, normally default definitons, unless you replace them via this program.

What To Do With Your Work File

Once your work file is set up, you will be given a choice of 3 things to do:

EDIT KEYS	which allows you to modify key definitions.
ASSIGN COLORS	which allows you to redefine the meaning associated with a particular color.
EXIT AND SAVE	This is provided here as a quick means of copying one set of definitions to another. For example, loading the default definitions into your work file and saving them as the current definitions, provides a means of restoring your current definitions after running an application which may have used a now undesired set of key definitions.

Key Values

Each key has 4 values associated with it:

normal value	which represents the unshifted value of the key as labelled.

SHIFTED value which represents the values of the keys, as labelled, obtained by typing the desired key and the shift key simultaneously.

CONTROL value which represents the value of the key obtained by typing the desired key and the CONTROL key simultaneously.

CAPS LOCK value which represents the value obtained while in CAPS LOCK mode. This mode is obtained by typing the COMMODORE key and stays in effect until you type it again.

The values for each key are listed in the document "The Console".

Selecting a Key to Edit

To select a key for editing, you must actually select the specific (1 of 4) value of the key that you want to modify. To do this, type the key so that the four values associated with it are displayed. Use the up and down arrow keys to scroll through the four values; type the return key to select the value that is highlighted. This is the specific key value that will be modified. The next time you view this key the new value you assigned will be displayed. You can modify as many keys as you want. When you are done, select the fifth choice - "exit and save work file". (NOTE: To view the up arrow, down arrow or return key, type the desired key and the control key simultaneously).

Editing Keys

Once you have selected a specific key value to edit, you will be given a choice of 5 ways of modifying the key:

ASSIGN a new character which allows you to do a 'one-for-one' replacement of the key.

ASSIGN a STRING which allows you to assign/edit a string (more than 1 character) to the key

ASSIGN a COLOR which allows you to assign an 80 or a 40 column color

ASSIGN a special function which allows you to assign a function from a list of currently available special functions.

ASSIGN a HEX value which allows you to replace the key value with a single hex value.

Assigning/Editing Strings

Once you have chosen to assign a string, you will be given a list of 32 available strings (some of which may already be defined.) Scroll through and select the one you want to assign to this key. At this point, the one you choose will be displayed near the top of the screen for editing. You can edit by typing keys and/or by

choosing one of the menu options presented - insert a color, insert a hex value, insert a special function or a second string (warning - the remainder of the string will be deleted if you insert a second string.) You can also use the left and right arrow keys and the insert and delete keys. When you are done, select the menu choice "exit string edit and save assigned string".

Assigning/Editing Color Values

Once you have chosen to assign a color, you will be given a choice of 5 color types to assign: an 80 column foreground color, an 80 column background color, a 40 column foreground color, a 40 column background color or a 40 column border color. Scroll through and select the one you want to assign. At this point, a color map consisting of 16 boxes labelled 'a' through 'p' will be displayed. Type the letter from the box representing the color that you want to assign. (NOTE that one letter (usually 'a') will appear to be missing, because it blends with the background color - assume ALL boxes are lettered sequentially!)

Assigning/Editing Special Functions

Once you have chosen to assign a special function you will be given a list of 16 currently available special functions. Scroll through the list and assign the function that you want this key to perform.

Assigning/Editing Hexadecimal Values

Once you have chosen to assign a hex value to a key, you will be prompted for the value to assign. Only charactersfrom 0-9 and a-f (upper or lower case) will be accepted. As you type characters, the current value is shifted left one nibble and the new value is ORed into the lst nibble. You can type as many characters as you want, but the last 2 you type will be the value assigned to the key. Type return when you are finished. Note: The value you type when the four key values are displayed will not be displayed as hex, but as what the hex value represents.

Finishing Up-Saving Your Work File

Once you have loaded your workfile and completed any editing you wanted to do, you will be given three choices as to what to do with your work file:

SAVE AS CURRENT	which makes the definitions in your work file effective immediately upon exiting this program
SAVE ON CP/M DISK	which will cause the definitions in your work file to be loaded the next time you boot from the disk to which it was saved
DON'T SAVE	a means of exiting if you made a mistake or changed your mind.

LIB

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 pruduces modules in Microsoft REL file format.

You can use LINK to link the object modules contained in a library to other object files. LINK 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.

Syntax:

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

Options

I	The INDEX option creates an indexed library file of type .IRL. LINK 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.

Modifiers

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)

Examples

Command	What it Does

```
LIB TEST4[P]   Displays all modules and publics in TEST4.REL.

LIB TEST5[P]=FILE1,FILE2
               Creates TEST5.REL from FILE1.REL and FILE2.REL and
               displays all modules and publics in TEST5.REL.

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.

LIB FILE2=FILE3<MODA=>
               Creates FILE2.REL from FILE3.REL, omitting MODA
               which is a module in FILE3.REL.

LIB FILE6=FILE5<MODA=FILEB.REL>
               Creates FILE6.REL from FILE5.REL, FILEB.REL
               replaces MODA.

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.

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.
```

LINK

LINK combines relocatable object modules such as those produced by RMAC, FORTRAN 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.

Syntax:

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

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.

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

B BIOS link in banked CP/M 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 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 command.

Examples	What it Does
LINK b:MYFILE[NR]	LINK 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.
LINK m1,m2,m3	

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

LINK m=m1,m2,m3

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

LINK MYFILE,FILE5[s]

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

MAC

MAC, the CP/M 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.

Syntax:

```
MAC filename {$options}
```

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 thru 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 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)

Examples	What it Does
MAC SAMPLE	Assembles the file SAMPLE.ASM using defaults
MAC SAMPLE $PB AA HC SX	Assembles the file SAMPLE.ASM on drive A:, putting .PRN file on drive B:, .HEX file on drive C:, and .SYM displayed on the console.

PATCH

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

Syntax:

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

Example	What it Does
PATCH SHOW 2	Patches the SHOW.COM system file with patch number 2.

PIP

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 target or 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.

Syntax:

```
PIP d:{Gn} | target_filespec{[Gn]} = source_filespec{[o]},...
           | d:{[o]}
```

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 logical device.

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.

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
```

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

PUT
===

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.

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
```

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.

Options

[{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.

Examples What it does

PUT CONSOLE OUTPUT TO FILE XOUT [ECHO]
Directs console output to file XOUT with the output echoed to the console.

PUT PRINTER OUTPUT TO FILE XOUT
MYPROG
Directs the printer output of program MYPROG to file XOUT. The output is not echoed to the printer.

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.

PUT CONSOLE OUTPUT TO CONSOLE
Directs console output back to the console.

PUT PRINTER OUTPUT TO PRINTER
Directs printer output back to the printer.

RENAME

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 with no filespec prompts you for input. RENAME is an internal command, but RENAME with password protected files requires the external utility RENAME.COM. You cannot rename files across drives or user areas.

Syntax:

```
    RENAME {new_filespec=old_filespec}
```

```
Example   What it does

RENAME FILE2.TXT=FILE1.TXT
          Renames the file FILE1.TXT to FILE2.TXT on the current
          drive.
```

RMAC

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

Syntax:

```
    RMAC filespec {$Rd | $Sd | $Pd}
```

Options

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.

```
Example   What it does

RMAC TEST $PX SB RB
          Assembles the file TEST.ASM from the current drive,
          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.
```

SAVE

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.

Syntax:

```
    SAVE
```

Example

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.

SET
===

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. For a discussion of passwords, date stamping and disk labels, see the document "The Disk System".

Syntax:

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

Label

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

Examples What it does

SET [NAME=DISK100]
Labels the disk on the default drive as DISK100.

SET [PASSWORD=SECRET]
Assigns SECRET to the disk label.

SET [PASSWORD=<cr>
Nullifies the existing password.

Passwords

```
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]
```

Password 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.

Attributes

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.

Examples What it does

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.

SET MYFILE.TEX [PASSWORD=MYFIL]
MYFIL is the password assigned to file MYFILE.TEX.

SET *.TEX [PASSWORD=SECRET, PROTECT=WRITE]
Assigns the password SECRET to all the TEX files on the current drive. Each TEX file is given a WRITE protect mode to prevent unauthorized editing.

SET MYFILE.TEX [RO SYS]
Sets MYFILE.TEX to Read-Only and SYStem.

Default

SET [DEFAULT=dd]

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

Time-Stamps

Syntax:

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

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.

Options

[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.

Examples What it does

SET [ACCESS=ON]
Turns on the access time stamp

SET [CREATE=ON,UPDATE=ON]
Turns on the create and update time stamps

Drives

Syntax:

```
SET {d:} [RO]
SET {d:} [RW]
```

Example	What it does
SET B: [RO]	Sets drive B to Read-Only.

SETDEF
======

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.

Syntax:

```
SETDEF { d: {,d: {,d: {,d:}}}} {[ TEMPORARY = d: ] |
         [ ORDER = (typ {,typ}) ]}
SETDEF [DISPLAY | NO DISPLAY]
SETDEF [PAGE | NOPAGE]
```

Examples What it does

SETDEF   Displays current SETDEF parameters.

SETDEF [TEMPORARY=C:]
          Sets disk drive C as the drive to be used for temporary
          files.

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

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.

SETDEF [DISPLAY]

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

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

SHOW
====
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

 Syntax:

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

Examples What it does

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

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

SHOW B:[LABEL]
          Displays label information for drive B.

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

SHOW C:[DIR]
Displays the number of free directory entries on drive C.

SHOW [DRIVE]
Displays the drive characteristics of drive A.

SID

The SID symbolic instruction 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.

Syntax:

SID {pgm-filespec} {,sym-filespec}

Commands

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 (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, SW uses 16 bit words
T{W}{n{,c}}	(Trace)	Trace Program Execution n is the number of program steps c is the utility entry address. W instructs SID not to trace subroutines
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. s is start address, 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.

Examples What it does

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

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

Utilities

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

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.

SUBMIT

The SUBMIT command lets you execute a group (batch) of commands from a SUBmit file (a file with filetype of SUB). This is similar to an MS-DOS .BAT file.

Syntax:

```
SUBMIT {filespec} {argument} ... {argument}
```

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 ($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:
```

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:
```

PROFILE.SUB

Everytime you power up or reset your computer, CP/M looks for a special SUBmit file named PROFILE.SUB to execute. If it does not exist, CP/M 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

The TYPE command displays the contents of an ASCII character file on your screen. TYPE with options or for password protected files is a transient utility requiring the program TYPE.COM.

Syntax:

```
TYPE {filespec [ PAGE | NOPAGE ]}
```

[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.

Example	What it does
TYPE B:THISFILE [PAGE]	Displays the contents of the file THISFILE from drive B on your screen twenty four lines at a time.

XREF

XREF provides a cross-reference 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.

Syntax:

```
XREF {d:} filename {$P}
```

→