ERA

The Resident Command for Erasing Files

The ERA resident command frees space once occupied by disk files, thereby permitting the storage of new files. ERA will erase a single file (1) or a group of files (2). ERA also displays a message when it cannot erase a file (3). Caution should be exercised when using ERA because erased files cannot be recovered under most circumstances.

1 ERASING A DISK FILE

To erase a file from a disk, you should enter a command line in the following form:

A>**ERA {filename.ext}** ***RETURN***

Where {**filename.ext**} is the complete name of the file you want to delete.

Files that do not reside on a disk in the default drive can only be erased when their file name is preceded in the command line by the appropriate drive specification.

2 ERASING GROUPS OF FILES

A group of files with similar names can be deleted by a single ERA command line when ambiguous file names (names with the "*" or "?" characters) are used, as in the following example:

A>**ERA B:PROGRAM?.PRN *RETURN***

which would delete files with names such as PROGRAM1.PRN, PROGRAM2.PRN, PROGRAM7.PRN, and PROGRAM/.PRN from the disk in drive B.

The following command example shows how to delete even more files at once:

A>**ERA B:*.* *RETURN***

This entry would erase every file from a disk. Because of the destructive potential of this form of the command, erase will ask you for confirmation with this message:

```
ALL FILES (Y/N)?
```

The command will not be executed until you confirms it by pressing the "**Y**" key. If the "**N**" key is pressed, the system prompt will be displayed.

3 CONSOLE RESPONSE TO ERA COMMANDS

When ERA finds the specified file and erases it, the system prompt returns.

If the specified file does not reside on a disk in the specified drive, then the console will display the following message:

```
NO FILE
```

If the file you desire to erase is write protected by the "R/O" (read only) status, or if you switched disks between drives without performing a warm boot, a message like the following will be displayed:

```
Bdos Err On x: File R/O
```

Where "x" is the letter of the drive containing the write-protected file.

If the disk containing the file to be erased is mechanically write protected (with adhesive tabs for 5.25-inch, without adhesive tabs for 8-inch, or with disk drive cabinet switches for 8-inch and Winchester Disks), then a message in the following form will be displayed:

```
Bdos Err On x: Bad Sector
```

Where "x" is the drive from which you tried to erase a file.

If you specified a drive in an ERA command, and that drive contains no disk, then a message in the following form will be displayed:

```
Bdos Err On x: Select
```

Where "x" is the drive from which you tried to erase a file.

FORMAT

The Utility that Prepares the Disk Surface

FORMAT prepares a floppy disk or a Winchester Disk partition for the storage of data by establishing storage areas on the disk surface. At the same time, FORMAT erases any data that remains on the disk from prior use, and sometimes inspects the recording surface for imperfections that could impair data storage or transmission. FORMAT also enables you to determine how much data you will be able to store on the disk.

CAUTION: Because FORMAT erases all existing data on a disk, make certain that you only format blank disks or disks containing expendable data. You can use the DIR (refer to the text titled "DIR") or STAT (refer to the text titled "STAT") commands to check a disk for valuable data files before formatting it. Notice, however, that the DIR and STAT commands cannot always find all of the data on a disk.

You can use the FORMAT utility through either of two methods: the FORMAT Prompt Method or the System Prompt Method.

1 FORMAT PROMPT METHOD

With this FORMAT method, you load the FORMAT utility into memory by entering a command at the system prompt. Then you answer a series of FORMAT prompts to define the formatting operation.

1.1 FORMAT Invocation

Answer the system prompt with a command in the following form:

A>**FORMAT *RETURN***

When invoked through the FORMAT prompt method, FORMAT first identifies itself with name, version number, and a caution about its capabilities. Then it asks you if you wish to continue the operation, as shown:

```
Format Version 2.04
This program is used to initialize a disk.
All information currently on the disk will be destroyed.
Is that what you want? (y/n):
```

If you wish to prepare a disk, enter a Y at this prompt and continue reading.

If you do **not** wish to prepare a disk, enter any character other than a Y. Control will return to the operating system, which will display a system prompt.

1.2 Specifying The Disk to be Formatted

After you have confirmed your intention to format a disk, FORMAT asks:

```
Which drive do you wish to use for this operation?:
```

Answer this prompt by entering the letter of the drive containing the disk you wish to prepare.

1.3 Disk Format Specification

Heath/Zenith disk drive models accommodate four different kinds of disk:

- 5.25-inch hard-sectored floppy disks;
- 5.25-inch soft-sectored floppy disks;
- 8-inch floppy disks; and
- Winchester Disk

Each of these disks types are formatted differently. Once you have specified your drive, the messages and prompts that FORMAT displays depend on the type of the disk you are formatting.

FORMATTING A 5.25-INCH HARD-SECTORED FLOPPY DISK

After you have selected a drive for this FORMAT operation, FORMAT prompts you as follows:

```
Put the disk you wish to be formatted in drive x.
Press RETURN to begin, anything else to abort.
```

Entering a carriage return at this prompt will begin the actual formatting operation, while entering any other keyboard character will cause all FORMAT activities to end as CP/M displays the system prompt.

When the operation is finished, FORMAT will display:

```
Do you have any other disks to be formatted?  (y/n):
```

After formatting, this disk has a file capacity of 90 kilobytes.

FORMATTING A 5.25-INCH SOFT-SECTORED FLOPPY DISK

After you select a drive for this FORMAT operation, FORMAT displays the prompt:

```
Which density? (S=single, D=double):
```

The "density" of a disk refers to the concentration of data on its surface. You can decide whether data will be stored on the disk at "single" density or "double" density by entering "S" or "D" at this prompt.

If you specify "D" for "double" density, data will be recorded on the disk at approximately twice the concentration as it would at "single" density. Higher density levels might also decrease data access reliability.

After the density is selected, FORMAT will display the prompt:

```
Number of Sides? (1=single, 2=double):
```

Since 5.25-inch soft-sectored disks have two usable sides, you have the option of selecting how many of these sides will be prepared to store data.

After you enter a valid response to each of these prompts, FORMAT will display either of the following two messages:

```
48 TPI drive -- 40 tracks will be formatted

Put the disk you wish to be formatted in drive x.
Press RETURN to begin, anything else to abort.
```

or

```
96 TPI drive -- 80 tracks will be formatted

Put the disk you wish to be formatted in drive x.
Press RETURN to begin, anything else to abort.
```

Entering a carriage return at this prompt will begin the actual formatting operation, while entering any other keyboard character will cause all FORMAT activities to end as CP/M displays the system prompt.

When the operation is finished, FORMAT will display the message:

```
Do you have any more disks to format? (y/n):
```

NOTE: Soft-sectored 5.25-inch disks supported by the Heath/Zenith CP/M are available in two varieties: those with 48 tracks per inch (48 tpi) and those with 96 tracks per inch (96 tpi). Each variety has different storage capacities, depending on the density and number of sides you select when formatting.

The following table shows the file capacities of 5.25-inch, soft-sectored, 48 tpi disks:

	Single-sided	Double-sided
Single density	90 kilobytes	190 kilobytes
Double density	148 kilobytes	308 kilobytes

The following table shows the file capacities of 5.25-inch, soft-sectored, 96 tpi disks:

	Single-sided	Double-sided
Single density	190 kilobytes	388 kilobytes
Double density	308 kilobytes	624 kilobytes

NOTE: The CP/M Operating System is set to accommodate either 48 tpi disks or 96 tpi disks in a Z-37 drive model. This setting can be adjusted through the CONFIGUR utility. (See "Submenu B: Set Disk Parameters" in the text on CONFIGUR.)

FORMATTING AN 8-INCH DISK

After you select the drive to be used to FORMAT this disk, the FORMAT utility will display one of the following prompts:

```
Which density? (S=single, D=double, E=extended double):
```

or

```
Which density? (S=single, D=double):
```

The "density" of a disk refers to the concentration of data on its surface. You can decide whether data will be stored on the disk at "single" density or "double" density or "extended double" density by entering "S" or "D" or "E" at this prompt. If you specify "D" for "double" density, data will be recorded on the disk at about twice the concentration as it would at "single" density. Extended double density is the highest level of data concentration possible on that disk. Higher density levels might also decrease data access reliability.

After you answer the density prompt, FORMAT displays:

```
Put the disk you wish to be formatted in drive x.
Press RETURN to begin, anything else to abort.
```

Entering a carriage return here will begin the formatting operation, while entering any other keyboard character will cause all FORMAT activities to end as CP/M displays the system prompt.

When the operation is finished, FORMAT will display the message:

```
Do you have any more disks to format? (y/n):
```

Heath/Zenith CP/M supports two varieties of 8-inch disk: single-sided and double-sided. Each variety has different storage capacities, depending on the density and number of sides you select during formatting.

The following table shows the file capacities of single-sided 8-inch disks:

Single density	Double density	Extended double density
241 kilobytes	482 kilobytes	596 kilobytes

The following table shows the file capacities of double-sided 8-inch disks:

Single density	Double density	Extended double density
490 kilobytes	980 kilobytes	1208 kilobytes

NOTE: The 8-inch floppy disk drive slot of the H/Z-67 drive model cannot be used to FORMAT a disk to extended double density. Furthermore, an 8-inch disk that is formatted to extended double density cannot be used in the H/Z-67 model.

FORMATTING A WINCHESTER DISK PARTITION

After you specify the drive name assigned to the partition, FORMAT displays the prompt:

```
Press RETURN to begin, anything else to abort.
```

Entering a carriage return here will begin the formatting operation, while entering any other keyboard character will cause all FORMAT activities to end as CP/M displays the system prompt.

When the formatting operation begins, the following message is displayed:

```
Formatting partition
```

When the operation is finished, FORMAT will display the message:

```
Do you have any more disks to format? (y/n):
```

1.6 Ending a FORMAT Operation

When FORMAT finishes preparing the surface of a disk or partition, it will display the following prompt:

```
Do you have more disks to format? (y/n):
```

The specified disk or partition is now ready for data storage.

If you wish to FORMAT another disk or partition immediately, press **Y** at this prompt.

If you have no other disks to FORMAT, press **N** at this prompt and FORMAT will display the following message:

```
Place a bootable disk in drive x and press any character:
```

Where "x" is the letter for the default drive.

If the disk that was used to perform bootstrap has been removed from drive A since bootstrap, then it should be inserted back into drive A now. The entry of any keyboard character will transfer control back to the operating system, which will display a system prompt.

2 SYSTEM PROMPT METHOD

The System Prompt Method enables you to include all of the specifications necessary for the FORMAT operation in a single command line entered at the CP/M system prompt.

2.1 Command Line Entry

FORMAT commands that are entered by the System Prompt Method should be entered in the following form:

A>**FORMAT {drive}:{[option,option]}** ***RETURN***

Where **FORMAT** is the command line function

where **{drive}** is the letter of the drive that contains the disk you wish to format (this letter must represent a valid drive in your hardware environment, such as **A, B, C, D, E,** or **F**); and

where **{[option,option]}** represents letters and/or numbers enclosed in square brackets [] and separated by commas , to specify how the formatting operation should be conducted.

2.2 FORMAT Options

FORMAT command lines entered by the System Prompt Method can include the following options:

SD Disk formatted to Single Density.

DD Disk formatted to Double Density.

1S One Side of disk formatted.

2S Two Sides of disk formatted.

F Fast formatting, because the routine test of disk surface media is not performed.

N No prompt displayed between FORMAT command entry and FORMAT execution.

T1 Type of disk to be formatted must be a 5.25-inch hard-sectored disk.

T3 Type of disk to be formatted must be a 5.25-inch soft-sectored disk.

T4 Type of disk to be formatted must be an 8-inch disk, within a Z47 or H47 drive model.

TH6 Type of "disk" to be formatted must be a partition on the Winchester Disk within a Z67 drive model. This partition must have been assigned a drive name by the ASSIGN utility.

TF6 Type of disk to be formatted must be an 8-inch disk, within the floppy disk slot of a Z67 drive model.

NOTE: It is not necessary to enter an option for disk type (T1, T3, T4, TH6, TF6) in a System Prompt Method FORMAT command, because FORMAT automatically checks the type of the disk within the specified drive.

However, disk type should be specified when you desire to format a disk **only if** it is of a particular type. If a disk type **is** specified as an option, and the drive specified for this operation contains a disk of a different type, then the following error message is displayed:

```
DISK IS NOT OF TYPE SPECIFIED
```

2.3 FORMAT Defaults

When you enter a FORMAT command line with a drive specification, and neglect to specify some or all of the possible options, FORMAT will prepare the disk according to the following default criteria:

- Disk formatted to Single Density (as if you specified the SD option);
- One side of disk formatted (as if you specified the 1S option);
- Disk surface will be tested for data retention (as if you did **not** specify the F option); and
- Prompts will be displayed between FORMAT command entry and FORMAT execution (as if you did **not** specify the N option)

Unless the N option is entered in the command line, the following prompt will be displayed after you enter the FORMAT command:

```
Format Version 2.04
This program is used to initialize a disk.
All information currently on the disk will be destroyed.
Is that what you want? (y/n):
```

To confirm your intention to run a FORMAT operation, enter a **Y** at this prompt. FORMAT will display a prompt in the form of the example below.

To abort the FORMAT utility at this point, enter any other keyboard character. The operating system will take control and display a system prompt.

```
Put the disk you wish to be formatted in drive x.
Press RETURN to begin, anything else to abort.
```

To begin execution of the FORMAT operation, enter a carriage return at this prompt.

To abort the FORMAT utility, enter any other keyboard character. The operating system will take control and display a system prompt.

2.4 System Prompt Method Examples

A>**FORMAT B:** ***RETURN***

FORMAT will prepare the surface of the disk in drive B. By default, this disk will be formatted to single density on only one side. Also by default, FORMAT will display prompts before formatting, and test the disk surface while formatting.

A>**FORMAT B:[DD,2S,T4]** ***RETURN***

FORMAT will prepare the surface of the disk in drive B to double density and on both sides, as specified by options. FORMAT will display prompts before formatting and test the disk surface while formatting, by default. However, if drive B is not a Z47 or H47 drive containing an 8-inch disk, an error message will be displayed and no formatting operation will occur.

A>**FORMAT B:[SD,1S,DD,2S]** ***RETURN***

If your command line contains contradictory options, FORMAT will acknowledge the last one. Hence, in this case, FORMAT will prepare the surface of the disk in drive B to double density, as specified by the last density option (DD), and on both sides, as specified by the last side quantity option (2S). FORMAT will display prompts before formatting and test disk surface while formatting, by default.

A>**C:format B:[Dd,f,N]** ***RETURN***

The FORMAT utility, in this case, is stored on the disk in non-default drive C. It will prepare the surface of the disk in drive B to double density, as specified by the "Dd" option. But since no option is specified for the number of sides, only one side will be formatted (by default). The "f" option specifies that this formatting operation will be performed without a disk media test. The "N" option specifies that FORMAT will not prompt you to confirm your intentions before the formatting operation begins.

NOTE: FORMAT command lines can be edited (by pressing the ***DELETE*** key) or erased entirely (by holding down the ***CTRL*** key and pressing the ***X*** key).

3 FORMAT ERROR MESSAGES

```
Drive not available in current configuration.
```

EXPLANATION: If you entered a drive name that does not exist in the hardware environment, then a different drive name should be entered. If you entered a drive name for which your copy of the CP/M Operating System has not yet been customized, then MAKEBIOS or CONFIGUR must be run before FORMAT will accept the drive name entry. If you are using a Z67 drive model and entered the drive name for a partition that has not yet been established by the ASSIGN utility, then ASSIGN must be run before FORMAT.

```
Unable to format this disk. It is write protected.
Do you have any more disks to format? (y/n):
```

EXPLANATION: If the disk is a 5.25-inch disk you should remove the adhesive tab from the notch on the disk cover. If the disk is an 8-inch disk the you should cover the notch on the disk cover using the tabs provided, and push down the write protect switch on the front of the H/Z47 or H/Z67 cabinet.

```
Unable to format this disk. Place a different disk in the drive
and press any key to begin...
```

EXPLANATION: The disk to be formatted is damaged or improperly inserted in the drive. You should try the operation again and replace the disk if the message appears again.

```
Media error
```

EXPLANATION: The disk to be formatted is damaged or improperly inserted in the drive. You should try the operation again and replace the disk if the message appears again.

```
Wrong type of media, or media inserted improperly,
or media damaged.
```

EXPLANATION: You may have tried to FORMAT a hard-sectored disk in a soft-sectored drive, or vice versa. You should check that the proper type of disk is being used. If the proper disk type is being used, then the disk is probably damaged and a different disk should be used for the operation.

```
ILLEGAL FORMAT OPTION
```

EXPLANATION: System Prompt Method command line was entered with undefined characters in place of options.

```
ILLEGAL COMMAND SYNTAX
```

EXPLANATION: System Prompt Method command line was entered with undefined characters in place of options.

```
DISK IS NOT OF TYPE SPECIFIED
```

EXPLANATION: A System Prompt Method command line specified a drive that contained a disk which did not match the specified disk type.

```
OPTION NOT AVAILABLE
```

EXPLANATION: A System Prompt Method command line included option characters which were not possible under the circumstances. Re-enter command with pertinent options.

```
Disk is not partitioned. The complete disk surface will be formatted.
Is that what you want? (y/n):
```

EXPLANATION: The Winchester Disk has not yet been partitioned. Therefore, any FORMAT operation will erase and prepare the surface of the entire Winchester Disk. To continue FORMAT operation, press Y. If you wish to partition the Winchester Disk before formatting, run the PART utility as instructed in the PART manual.

```
PARTITION IS SMALLER THAN MINIMUM ALLOWABLE SIZE
```

EXPLANATION: Winchester Disk partitions must contain a minimum number of sectors before they can be prepared by FORMAT. Either ASSIGN a partition containing this minimum quantity of sectors, or use the PART utility to repartition the Winchester Disk with few enough partitions so that at least one is of minimum size. Refer to the Z67 manual and the PART manual to determine the minimum allowable size for a partition.

```
PARTITION IS LARGER THAN CP/M MAXIMUM SIZE -- ONLY 8 MEGABYTES USABLE
```

EXPLANATION: Winchester Disk partitions should contain no more than eight megabytes of storage capacity if used with the CP/M Operating System. If a partition contains more, it cannot be formatted. Either ASSIGN a partition of suitable size for formatting purposes, or repartition the Winchester Disk with the PART utility and form a partition within the eight megabyte limit. Then format the properly sized partition.

LIST

The Utility that Prints File Contents on Paper

The LIST utility enables you to obtain paper copies of files by entering a command (1) for one or more files (2) to be printed. Special printout characteristics can be set when you enters the command with LIST parameters (3). You can stop a LIST printout in progress (4). LIST is used to print out only certain types of files (5).

1 METHODS OF ENTERING LIST COMMANDS

Two different methods can be used to enter LIST commands: the LIST Prompt Entry Method and the System Prompt Entry Method.

1.1 LIST Prompt Entry Method

With the LIST Prompt Entry Method, you enter LIST in response to the system prompt. This entry is sufficient to invoke LIST, which displays its own prompt—the asterisk (*). You can now enter the argument portion of the command line in response to the "*" prompt supplied by LIST. LIST prompt entries are made in the following form:

A>**LIST** ***RETURN***

{argument}** ***RETURN

Where **{argument}** is the name of the file(s) to be listed.

After the LIST operation is finished, LIST again displays the "*" prompt. You can enter another argument or return to the operating system by entering a carriage return.

1.2 System Prompt Entry Method

To invoke a similar operation without splitting up the command line, you must include the argument in the response to the system prompt, as in the following example:

A>**LIST {argument}** ***RETURN***

Where **{argument}** is the name of the file(s) to be listed.

After LIST finishes the latter printing operation, it automatically returns you to the operating system, and the system prompt is displayed.

EXAMPLE OF LIST USAGE

The following command, entered using the LIST Prompt Method:

A>**LIST** ***RETURN***

REPORT.DOC** ***RETURN

and this command, entered using the System Prompt Method:

A>**LIST REPORT.DOC** ***RETURN***

will produce the same results. Both commands will produce a paper copy of the file named "REPORT.DOC".

2 PRINTING CONTENTS OF MORE THAN ONE FILE

Any number of files can be specified in a single LIST command. To initiate a listing of several files, the names of the files are entered in the argument separated by single spaces.

If the files to be listed reside on different disks, their names should be preceded by a drive specification (drive letter and colon).

Both of the following examples demonstrate how to LIST the contents of the specified files:

A>**LIST** ***RETURN***

PRINTOUT.DOC B:PROGRAM.PRN B:REPORT.DOC C:SYSTEMX.PRN** ***RETURN

or

A>**LIST PRINTOUT.DOC B:PROGRAM.PRN B:REPORT.DOC C:SYSTEMX.PRN** ***RETURN***

3 LIST PARAMETERS

You can specify parameters in a LIST argument to alter the characteristics of a standard printout. These parameters allow you to select printout characteristics such as the date, the number of copies desired, the width of tabs, etc. If no parameters are specified, LIST will print a document with default value characteristics.

The parameters used to specify printout characteristics are as follows:

PARAMETER NAME	KEYBOARD ENTRY	DESCRIPTION OF PRINTOUT CHARACTERISTIC	DEFAULT VALUE
Date	**[D xxx...x]**	First 10 characters of specified date printed on upper left corner of each document page.	none
Heading	**[H xxx...x]**	First 60 characters of specified heading printed on top line of each document page.	file name
No Heading	**[N]**	No heading or date printed on any document page.	file name
Lines per Page	**[L nn]**	Each document page has the specified number of lines. Specification of zero/page lines causes continuous printing.	60 lines per page
Tab Stop Width	**[T nn]**	Each tab stop within text is expanded or contracted to specified number of spaces.	8 spaces

PARAMETER NAME	KEYBOARD ENTRY	DESCRIPTION OF PRINTOUT CHARACTERISTIC	DEFAULT VALUE
Page Number	**[P nn]**	Page numbering sequence begins with specified number on first file page.	page 1
Upper Case	**[U]**	All letters in document printed in upper case.	upper and lower case
Copies Desired	**[C nn]**	Specified number of copies are printed.	1 copy
Erase	**[E]**	File is erased from disk after LIST operation is completed.	file retained on disk

USE OF LIST COMMAND LINE PARAMETERS

Parameters are entered, enclosed in square brackets, after the last file name in the LIST command. If more than one parameter is entered, each should be enclosed in a set of brackets.

A LIST command entered with parameters using the LIST Prompt Method might appear as follows:

A>**LIST *RETURN***
PRINTOUT.DOC PROGRAM.PRN [H Today's Work] [D 31-Feb-81] [P 09] *RETURN

This command will cause the contents of the files PRINTOUT.DOC and PROGRAM.PRN to be printed, with the following heading across the top of the first page of each file:

```
Today's Work                                    31-Feb-81  Page  9
```

The parameters will take effect on all of the files specified in that command line.

When the command is entered using the LIST Prompt Entry Method, any parameters entered (except for the starting Page Number parameter and Erase parameter) will remain in effect with any files specified at successive asterisk (*) prompts, until new values are entered for the parameters, or until control is returned to the operating system.

When a LIST command line is entered using the System Prompt Method, letters in the heading, date, and page number line might be automatically translated into upper case. The following System Prompt command demonstrates this character translation:

A>**LIST PRINTOUT.DOC PROGRAM.PRN [H Today's Work] [D 31-Feb-81] [P 09]** ***RETURN***

This command will cause the contents of the files PRINTOUT.DOC and PROGRAM.PRN to be printed with the following heading across the top of the first page of each file:

```
TODAY'S WORK                                    31-FEB-81  PAGE  9
```

4 ABORTING A LIST OPERATION

After a LIST command has been entered, the printout can be aborted by pressing any keyboard character. When you abort a command that was entered by the LIST Prompt Method, the next printout might also be automatically aborted soon after it begins. Therefore, it is advisable to exit from, and reinvoke, the LIST utility after aborting a printout.

5 FILES THAT SHOULD BE LISTED

Only files containing ASCII characters should be listed. ASCII character files include files composed using a text editor (such as ED), or file with the "PRN" extension created by an assembler (such as ASM).

Files such as those bearing the extension "COM" or "HEX" will produce meaningless printouts if LISTed because they are not composed of ASCII characters.

Files composed in a word processor will LIST, but they might possess features (such as bold face characters or page breaks) that LIST will not print in the same way as the word processor's printout command.

6 LIST ERROR MESSAGES

```
File not found
```

EXPLANATION: A file specified in the LIST command argument does not exist on the disk in the logged drive. If the file does exist on a disk, that disk should be inserted in a drive and the appropriate drive should be specified before the file name in the argument.

```
Syntax error in command line
```

EXPLANATION: The LIST command was improperly entered. Often occurs if a space is not entered between file names, or when an invalid parameter is specified.

LOAD

The Utility that Loads a "HEX" File for Execution

The LOAD utility puts an assembled hexadecimal file in the Transient Program Area (1), and translates the file into a "COM" file, which is executable under the CP/M operating system (2). Only certain types of files can be loaded (3).

1 LOAD INVOCATION

To LOAD a hexadecimal file into the Transient Program Area (TPA), you should respond to the system prompt with an entry in the form:

A>**LOAD {hex file}** ***RETURN***

Where **{hex file}** is the primary file name of an Intel hexadecimal file. The file extension is omitted from this entry because LOAD always assumed the extension to be "HEX".

This command translates the hex file into machine code that can be executed under CP/M, gives it a "COM" extension, and writes it back to the disk.

2 EXECUTING A LOADED PROGRAM

You can start execution of the program by entering the primary name of the file (and a carriage return) in response to the system prompt. Thus the file is executed as if it were a regular CP/M utility.

It is only necessary to LOAD a hex file once. The "COM" version of the file will be stored on the disk from which the "HEX" version came. Thereafter, CP/M treats it like another "COM" file.

The operation can take place on a non-default drive if the file name is prefixed by a drive name. Thus the entry of the following command:

A>**LOAD B:BETA** ***RETURN***

brings the LOAD program into the TPA from the default disk and then operates on the file "BETA.HEX", which resides in drive B. The file named "BETA.COM" is written to the disk in drive B. (The file "BETA.HEX" will also remain on the disk.) Now you can execute BETA.COM be responding to the system prompt as shown:

A>**BETA** ***RETURN***

Thus, you can "invent" new CP/M commands by using LOAD to translate a "HEX" file into a CP/M-executable "COM" file.

3 LOAD REQUIREMENTS FOR HEX FILES

To be loaded, a file must contain valid Intel hexadecimal format records. The ASM utility can be used to produce such a file from a file with the "ASM" extension. The hex file must begin at address 100H, which is the memory location at which the Transient Program Area (TPA) begins. In addition, the addresses in the hex records must be in ascending order. Gaps in memory regions are filled with zeroes by the LOAD command, as the hex records are read. Thus, LOAD must be used only for creating CP/M command files, which operate in the TPA. Program files that occupy memory locations other than the beginning of the TPA (address 100H) can be loaded using the DDT utility. After a program is loaded, LOAD displays a message in the following form:

```
FIRST ADDRESS nnnn
LAST ADDRESS nnnn
BYTES READ nnnn
RECORDS WRITTEN nn
```

4 LOAD ERROR MESSAGES

```
INVALID HEX DIGIT
```

EXPLANATION: The hexadecimal file you tried to LOAD contains upper bits set in ASCII words.

```
ERROR ADDRESS nnnn
```

EXPLANATION: An error occurred at address "nnnn".

```
CHECK SUM ERROR
```

EXPLANATION: The hexadecimal file did not produce the correct check sum during LOAD execution.

```
ERROR: CANNOT OPEN SOURCE, LOAD ADDRESS nnnn
```

EXPLANATION: The hexadecimal source file specified in the LOAD command line is not present on the specified disk.

```
NO MORE DIRECTORY SPACE
```

EXPLANATION: The directory on the disk is full.

```
CANNOT CLOSE FILES
```

EXPLANATION: The loaded file is not present when LOAD tries to close the "COM" file.

```
INVERTED LOAD ADDRESS, LOAD ADDRESS nnnn
```

EXPLANATION: The loaded hexadecimal file did not start at beginning of the TPA (0100H) as it should have.

MAKEBIOS

The Utility that Customizes the Operating System for Disk Drives

The MAKEBIOS utility helps you to create the part of CP/M known as the Basic Input/Output System (BIOS). The kind of BIOS that MAKEBIOS creates is selected by you to accommodate any combination of disk drives offered by Heath/Zenith. You can implement this utility by running MAKEBIOS (1) and selecting a drive combination (2). MAKEBIOS execution is automatic (3).

NOTE: The BIOS is a component of the CP/M Operating System that makes the system work on Heath/Zenith hardware. The file "BIOS.SYS" (which is included with your CP/M software) contains the BIOS.

1 RUNNING MAKEBIOS

After making a backup copy of the CP/M Distribution Disk, you can run MAKEBIOS by entering a command line in the form:

A>**SUBMIT MAKEBIOS {destination drive}:{file name} {source drive}** ***RETURN***

Where {**destination drive**} is the name of the drive/partition device to which you wish to send the customized BIOS;

where {**file name**} is the optional name of the customized BIOS. If no name is entered, the file containing the tailored BIOS will be called "BIOS.SYS". Any file on the destination disk previously named "BIOS.SYS" will be overwritten.;

where {**source drive**} is the optional name of the drive/partition upon which the following files reside: MAKEBIOS.COM, MAKEBIOS.SUB, PREL.COM, BIOS.ASM; and

where the following files must reside on the default drive: ASM.COM and SUBMIT.COM.

An example of such a command line would be as follows:

A>**SUBMIT B:MAKEBIOS C: B:** ***RETURN***

NOTE: Neither the default disk nor the disk containing the MAKEBIOS files can be write protected during a MAKEBIOS run.

2 SELECTING THE APPROPRIATE DRIVE COMBINATION

The entry of the preceding example command line will first produce the display of a menu listing assortments of Heath/Zenith drive controllers:

```
A>MAKEBIOS A:1  C:

BIOS SELECTION MENU

A -- H17 ONLY
B -- H37 ONLY
C -- H47 ONLY
D -- H67 ONLY
E -- H17 AND H37
F -- H17 AND H47
G -- H17 AND H67
H -- H37 AND H47
I -- H37 AND H67
J -- H47 AND H67

ENTER SELECTION:
```

You should enter the selection letter corresponding to the disk drive combination being used.

NOTE: The above selections refer to disk drive controllers. Most disk drive models bear the same number as that of the controller that controls them, except that the "H-17" controller can control the H-77 drive unit, the H/Z-87 drive unit, or the H/Z-17 drive within the H/Z-89 microcomputer model.

3 MAKEBIOS EXECUTION

From this point on, MAKEBIOS execution is automatic. No further user participation is required.

After you enter a letter at the "ENTER SELECTION:" prompt, a series of command lines and execution messages will appear on the screen. The following example shows the form of this display. (Numeric values will vary.)

```
A>ASM BIOS.AAZ
CP/M ASSEMBLER - VER 2.0
1537
02DH USE FACTOR
END OF ASSEMBLY

A>REN A:BIOS.HX0=BIOS.HEX
A>MAKEBIOS A:2

A>ASM BIOS.AAZ
CP/M ASSEMBLER - VER 2.0
1637
02DH USE FACTOR
END OF ASSEMBLY

A>REN A:BIOS.HX1=BIOS.HEX
A>PREL A:BIOS C:BIOS

A>REN C:BIOS.SYS=BIOS.HX1
A>MAKEBIOS C:3

MAKEBIOS FUNCTION COMPLETE
```

When the display is complete, the destination drive will contain a BIOS suitable for your disk drive combination. This BIOS will bear the file name "BIOS.SYS", or whatever name you might have specified.

This new file will now help control input/output operations for your hardware. It may also be transferred to other disks or partitions, using the PIP utility with the "[R]" parameter.

4 MAKEBIOS FUNCTION

The CP/M Operating System requires a Basic Input/Output System (BIOS) to control input/output operations for any peripheral devices that are connected to the Heath/Zenith computer. Among these devices may be the disk drives, a keyboard, a video monitor and a printer.

Your CP/M Distribution Disk (or disk set) comes with a BIOS that will adequately control one of your disk drive types. However, Heath/Zenith offers four different kinds of disk drive, and each hardware environment can include any combination of one or two different drive types. Because a BIOS capable of controlling all combinations would take up too much memory space, the MAKEBIOS files are provided to enable to create a BIOS that will control your own disk drive combination.

Therefore, users with two different kinds of disk drive should run MAKEBIOS.

5 A PERSPECTIVE ON MAKEBIOS INVOCATION

You do not invoke the MAKEBIOS.COM utility directly. You enter a SUBMIT command, which triggers the automatic entry of several command lines. The command lines that invoke MAKEBIOS.COM and other CP/M commands are stored in the file named "MAKEBIOS.SUB".

The utilities and resident commands specified within the MAKEBIOS.SUB file automatically modify, assemble, and relocate the source file BIOS.ASM. After these activities are performed, the assembled BIOS file is written to the specified file name (or default name BIOS.SYS) on the specified drive.

6 FURTHER BIOS MODIFICATIONS

You can perform further modifications to the BIOS before running MAKEBIOS by carefully altering the source file BIOS.ASM, which is included with your CP/M distribution media.

However, BIOS.ASM should not be altered until after the the distribution disk has been copied, and the MAKEBIOS.SUB file has been SUBMITted.

As noted within the BIOS.ASM program, no alterations should be made to the first five lines of the program under any circumstances. The first five lines of the program affect subsequent locations in the program, and user alterations to them could easily render the entire program useless.

7 MAKEBIOS ERROR MESSAGE

```
ERROR IN EXECUTION OF MAKEBIOS
```

EXPLANATION: Check the parameters in the SUBMIT MAKEBIOS command line, check the arrangement of the necessary disk files, and run MAKEBIOS again.

MOVCPMxx (MOVCPM17, MOVCPM37, MOVCPM47, and MOVCPM67)

The Utilities that Customize a CP/M Operating System Kernel to Fit Memory Size and Disk Type

After selecting the proper MOVCPM utility to use (1), you can implement it to adjust the system kernel so that it has the proper memory limit and will work on disks of a particular type (3). This utility should be followed immediately with another utility or command (5).

1 SELECTING THE PROPER MOVCPM UTILITY

MOVCPM17 is run when you wish to move a system kernel to a 5.25-inch, hard-sectored disk.

MOVCPM37 is run when you wish to move a system kernel to a 5.25-inch, soft-sectored disk.

MOVCPM47 is run when you wish to move a system kernel to an 8-inch disk that will be used to perform bootstrap in an H/Z47 drive model.

MOVCPM67 is run when the you wish to move a system kernel to a Winchester Disk partition, or to an 8-inch disk that will be used to perform bootstrap in an H/Z67 drive model.

In this text section, MOVCPMxx will be used to indicate any of these four utilities designed for use with CP/M Version 2.2.04., where "xx" is the model number of the drive unit receiving the system kernel.

2 FUNCTION OF MOVCPMxx

MOVCPMxx loads the kernel of a CP/M Operating System (the part exclusive of the file BIOS.SYS) into a special location in computer memory. At this location, it adjusts the system kernel to either a specified memory size or the total available memory size of the computer. MOVCPMxx must also observe and measure the BIOS.SYS file that will eventually be used with the system kernel, in order to allow sufficient space for the BIOS.SYS file. In addition, MOVCPMxx must make adjustments for the type of disk to which the system kernel will eventually be transferred. MOVCPMxx, however, will rely upon the SYSGEN utility to copy the system kernel that MOVCPMxx loaded into memory.

3 MOVCPMxx COMMAND LINE ENTRY

The MOVCPMxx command line is entered in the following form, with one mandatory specification, two optional specifications and a space separating each, as shown:

MOVCPMxx nn z *RETURN*

Where **xx** is the number that identifies a particular MOVCPM utility being used. It also stands for the type of disk drive that will be the destination of the system kernel (17, 37, 47, or 67). One of these numbers is a mandatory component of any MOVCPM command;

where the **nn** variable represents the memory size the transferred system kernel will occupy, in multiples of 1024 bytes (kilobytes). This is an optional value. If the "*" character or no value is entered, the system kernel will be set to occupy the entire memory capacity of the computer being used, by default. (H/Z89 computers have RAM capacities of 32K, 48K, or 64K.) This value can be less than or equal to the actual memory capacity of the computer. It should not be greater than the memory capacity of the computer.

where the **z** variable represents the location of the BIOS that is to be matched up with the system kernel being moved. You should enter the name of the drive that contains the appropriate BIOS. If the you has renamed the BIOS to be used (by using the MAKEBIOS or REN commands), then this variable must include the name of the drive containing this BIOS and the name of the file containing the BIOS. This variable is optional. If omitted, the MOVCPM utility will assume that the created system kernel will be matched with a copy of the BIOS that was loaded into memory during bootstrap.

The "*" character must be entered when you specify no value for the memory ("nn") variable, and **do** specify a value for the BIOS source ("z"). In this sort of command line, the "*" character acts as a space filler so that the computer does not interpret the source value entered as a memory value because it was entered in the memory value space.

During execution, the MOVCPMxx utility will respond with a message in the following form:

```
MOVCPMxx VERSION 2.v

CONSTRUCTING nnk CP/M vers 2.v
READY FOR "SYSGEN" OR
"SAVE 38 CPMnn.COM"
```

4 MOVCPMxx EXAMPLES

The following command lines and accompanying explanations are specific examples:

A>**MOVCPM67 64** ***RETURN***

The system kernel created by this command will be bootable from an H/Z67 drive. It will operate with 64K of RAM. The kernel will be adjusted using the BIOS in computer memory for reference.

A>**MOVCPM37* C:** ***RETURN***

The system kernel created by this command will be bootable from an H/Z37 drive. It will probe computer memory and operate at computer's memory capacity. This kernel will be adjusted using the BIOS.SYS file stored in drive C: for reference.

A>**MOVCPM1732 B:MYBIOS.SYS** ***RETURN***

This command will adjust a system kernel that will eventually help to make a 5.25-inch hard-sectored disk bootable. It will operate with 32K of RAM. This kernel will be adjusted by referring to the BIOS file "MYBIOS.SYS", which resides in drive B:.

5 AFTER RUNNING MOVCPMxx. . .

MOVCPMxx should be immediately followed by the SYSGEN utility, which will transfer the adjusted CP/M system kernel from the TPA to the system tracks of a specified blank disk; or by the SAVE utility, which will transfer the adjusted CP/M system kernel from the TPA to a disk file.

If you perform any other activity immediately after running MOVCPMxx, the work of MOVCPMxx will probably be destroyed.

NOTE: The BSYSGEN utility, although similar to the SYSGEN utility, cannot be used to transfer a system kernel moved by a MOVCPMxx command.

6 MOVCPMxx ERROR MESSAGES

```
INVALID MEMORY SIZE
```

EXPLANATION: Valid memory sizes are between 32K and 64K.

```
SYNCHRONIZATION ERROR
```

EXPLANATION: The serial number of the MOVCPMxx utility used must match that of the operating system used.

```
READ ERROR
```

EXPLANATION: MOVCPMxx cannot read data from a file you specified because the file is flawed.

```
NO FILE
```

EXPLANATION: MOVCPMxx cannot read data from a file you specified because it cannot find the file on the specified drive.

```
NO SPACE
```

EXPLANATION: The BIOS file you specified will not fit in the Transient Program Area.

```
BAD LOAD
```

EXPLANATION: A file specified by you did not load properly. You should try the MOVCPMxx command again or specify a different file.

```
CAN'T OPEN BIOS.SYS
```

EXPLANATION: MOVCPMxx is unable to use the specified BIOS because it is not stored in a file named "BIOS.SYS", or because it is a defective file.

```
FATAL ERR F25: NOT ENUF MEMORY
```

EXPLANATION: MOVCPMxx execution was aborted; you should perform bootstrap and try again.

```
File not found.
```

EXPLANATION: MOVCPMxx could not find the file you specified.

PIP

The Utility that Copies Data between Files, Disks, and/or Devices

PIP stands for Peripheral Interchange Program, the CP/M utility that can be invoked (1, 2, 3) to copy (4) and link (5) files or parts of files (6), and to direct intput and output between logical devices (8). PIP also allows files to be transferred between different disks when using a one-drive hardware environment (9). PIP operations can be regulated by parameters (11).

1 METHODS OF ENTERING PIP COMMANDS

Two different methods can be deployed to enter PIP commands: the PIP prompt method and the system prompt method.

1.1 PIP Prompt Method

With the PIP prompt mode, you enter "PIP" in response to the system prompt. This entry is sufficient to invoke PIP, which displays its own prompt—the asterisk (*). You can now enter the argument half of the command line in response to the "*" prompt supplied by PIP, as shown in the following example:

```
A>PIP RETURN
*{argument} RETURN
*
```

After this PIP operation is finished, PIP again displays the "*" prompt. In reply, you can enter another argument to command PIP to perform another operation, or enter a carriage return only to return to the operating system and obtain a system prompt.

1.2 System Prompt Method

To induce a similar operation without splitting up the command line, the system prompt mode of command entry can be used. With this mode, you must include the argument in the response to the system prompt, as shown in the following example:

```
A>PIP {argument} RETURN
A>
```

After PIP finishes this operation, it automatically returns you to the operating system, and the system prompt is displayed.

All characters specified in such a command line will be automatically translated into UPPER CASE.

NOTE: PIP sends data to a destination from a source, but it does not remove the data from the source. It merely copies an image of the data, and then sends it. Therefore, any data that is copied will remain intact at the source after the PIP operation.

2 THE ARGUMENT IN A PIP COMMAND LINE

PIP commands require an argument regardless of which invocation method is used. PIP arguments must specify a "data source" (which can be either a file or a logical input device), and a "data destination" (which can be either a file, a disk, or a logical output device). The "data destination" is entered first and followed by an " = " sign, which is followed by the "data source", as shown in these examples:

A>**PIP** ***RETURN***
{data destination} = {data source} ***RETURN***

or

A>**PIP {data destination} = {data source}** ***RETURN***

In all cases, copied data is transferred in a right-to-left direction, with respect to the command line components. Hence, the file, disk, or logical output device to receive the data is always specified on the left hand side of the " = " sign; and the file or logical input device to submit the data is always specified on the right hand side of the " = " sign.

If entered in the system prompt mode, characteristics entered in a PIP command line will be in upper case in any future displays. Characters entered in a PIP prompt mode command line will reappear in the case in which they were entered.

3 CHARACTERISTICS OF DATA DESTINATION AND SOURCE SPECIFICATIONS

A data destination can be a file, a disk, or a logical device. A data source can be a file or a logical device. Each must be specified in the command line as shown below:

DATA SPECIFICATION CHARACTERISTICS	EXAMPLES
Data can be transferred to a disk by specifying the letter of the drive in which the disk resides, and a colon.	B:=SENDOVER.DOC B:=CON:
Files on non-default disks are identified by specifying the drive in which they reside immediately before the file name.	B:NEW.DOC=C:SEND.DOC B:CREATED.DOC=CON: CON:=B:SENDOVER.DOC
Logical devices are specified by entering the three-letter code for that particular device and a colon.	CON:=LST: LST:=SENDOVER.DOC CREATED.DOC=CON:

4 FILE COPYING EXAMPLES

The following argument will cause PIP to transfer a copy of the file SENDOVER.DOC from drive B to the same disk, and give it the file name CREATED.DOC. In effect, this operation creates a file backup with a different name on the same disk.

B:CREATED.DOC = B:SENDOVER.DOC *RETURN*

The following argument will cause PIP to transfer a copy of the file SENDOVER.DOC from drive C to the disk in drive B, and give it the file name CREATED.DOC.

B:CREATED.DOC = C:SENDOVER.DOC *RETURN*

The following argument will cause PIP to transfer a copy of the file SENDOVER.DOC from the currently logged drive to the disk in drive B, and give it the same file name.

B: = SENDOVER.DOC *RETURN*

The following argument will cause PIP to transfer a copy of the file SENDOVER.DOC from drive C to the disk in drive B, and give it the same file name.

B: = C:SENDOVER.DOC *RETURN*

Each of the following arguments will cause PIP to transfer a copy of the file SENDOVER.DOC from the disk in drive B to the disk in the currently logged drive, and give it the file name SENDOVER.DOC. It should be noted that the data source in this argument is **not** the disk in drive B, but the file SENDOVER.DOC which resides on the disk in drive B. Because the source file is not specified in the typical data source location, PIP assumes that the source file is the same as the destination file, which is specified in its typical location.

SENDOVER.DOC = B: *RETURN*

or

SENDOVER.DOC = B:SENDOVER.DOC *RETURN*

5 DATA SOURCE CONCATENATION

While only one data destination can be specified in a single PIP command line, several data sources can be specified. Thus you can merge data from several locations to one location.

When more than one data source is specified, each data source specification is separated by a comma.

The length of a PIP command line cannot exceed 255 characters. (If the user tries to enter a 256th character into the command line, PIP will begin execution based on the first 255 characters entered.)

PIP will concatenate data sources in the order in which their specifications are entered in the command line.

The following PIP argument will cause PIP to get a copy of the files THISFILE.DOC and THATFILE.DOC from the disk in drive C and the file YOURFILE.DOC from the disk in the default drive. PIP will then combine all three files into one file (in the order they are specified), place this file on the disk in drive B, and assign the name COMBINED.DOC to the transferred combination of the three source files.

B:COMBINED.DOC = C:THISFILE.DOC,C:THATFILE.DOC,YOURFILE.DOC *RETURN

Both files and devices can be specified as data sources in the same command line.

NOTE: If you wish to concatenate object files with a PIP operation, you must specify the O parameter after the file name of each object file that you wish to concatenate.

6 COPYING A BLOCK OF DATA

Blocks of files, as well as whole files, can be copied to another file. To do this, you enter a PIP command that specifies the source file name and the beginning and end of the block being transferred. Such an entry requires the use of PIP parameters as in the following example:

BLOCKFIL.TXT = WHOLEFIL.TXT[Sbeginning ^ ZQend ^ Z]** ***RETURN

In this example, "BLOCKFIL.TXT" is the name of the destination file to receive a block of text from the source file named "WHOLEFIL.TXT". The block is specified by entering a bracket, the parameter "S", a unique string of text from the "beginning" of the block, a CTRL-Z character, the parameter "Q", a unique string of text from the "end" of the block, a CTRL-Z character, and another bracket.

The following example demonstrates how the user can concatenate blocks from several files, by using the "S" and "Q" parameters to specify the beginning and end of each block:

USBLOCK.TXT = GETADD.TXT[SFour ^ ZQearth ^ Z],CONSTIT.TXT[SWhen ^ ZQAmerica ^ Z]** ***RETURN

With the entry of this command line, PIP would combine the specified block from the file "GETADD.TXT" with the specified block from "CONSTIT.TXT". With this data, PIP would create the file "USBLOCK.TXT".

NOTE: The strings following the S and Q parameters will be translated to UPPERCASE if you are using PIP by the System Prompt Method. They will **not** be translated to upper case if you are using the Utility Prompt Method.

7 COPYING DATA TO AN INTEL HEXADECIMAL FILE

PIP performs a special function if the data destination is a file with the "HEX" extension (an Intel hex formatted machine code file) and the data source (for one of several data sources) is an input/output device, such as a paper tape reader.

In such a case, PIP checks to ensure that the source is a properly formed Intel hexadecimal file, with legal hexadecimal values and checksum records. If an invalid input record is found, PIP reports an error message at the console and waits for corrective action.

It is usually sufficient to open the reader and rerun a section of the tape (by pulling the tape back about 20 inches). When the tape is ready for the second reading, you should enter a carriage return at the console, and PIP will attempt another read. If the tape position cannot be properly read, then you should simply continue the read (by entering a carriage return after the error message), and enter the record manually, using the ED utility after the disk file is constructed from the other data source components.

For convenience, PIP allows a CTRL-Z ("end-of-file" character) to be entered from the console if the source file is a RDR: device. In such a case, PIP reads the specified device and monitors the keyboard. If a CTRL-Z is entered at the keyboard, then the read operation is terminated normally.

For instance, the following PIP argument could be entered to create a "HEX" file:

CREATED.HEX = CON:,B:SENDOVER.HEX,PTR: *RETURN

The preceding argument will cause PIP to create the data destination file CREATED.HEX by reading the data from source device CON: (as you enter hexadecimal values at the keyboard and eventually enter a CTRL-Z "end-of-file" character), the data from source file SENDOVER.HEX (which includes a CTRL-Z), and finally the data from source device PTR: (until a CTRL-Z is read from the paper tape).

NOTE: Hexadecimal data can be checked for valid format by specifying the "H" or "I" format in the PIP argument.

8 COPYING DATA TO OR FROM LOGICAL DEVICES

The following examples, and their ensuing explanations, demonstrate PIP commands that transfer data to and/or from computer-recognized logical devices:

LST:=SENDOVER.PRN *RETURN

The preceding argument will cause PIP to copy the source file SENDOVER.PRN to the LST: device.

CON:=B:THISFILE.ASM,C:THATFILE.ASM,YOURFILE.ASM *RETURN

The preceding argument will cause PIP to concatenate three ASM source files from disk B, disk C, and the logged disk, respectively; and to copy the combination to the CON: destination device.

PUN:=NUL:,SENDOVER.ASMmr,EOF:,NUL: *RETURN

The preceding argument will cause PIP to send 40 nulls from the NUL: source device to the PUN: destination device, and then to copy the SENDOVER.ASM source file to the PUN:, then to send a CTRL-Z "end-of-file" character from the EOF: source device to the PUN:, and finally to send 40 more nulls from the NUL: source device to the PUN: data destination device.

8.1 Input/Output Devices Accessible through PIP

The PIP utility enables you to transfer data directly to or from the logical and physical devices being used. PIP supports data transfer with respect to the devices indicated in the following table:

LOGICAL DEVICE NAME	PHYSICAL DEVICE NAME	DEVICE DEVICE DESCRIPTION AND/ OR CATALOG NAME OF RECOMMENDED INPUT/OUTPUT MACHINE
CON:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	CRT:	Any non-handshaking RS-232 video terminal at port 0E8H
	BAT:	A batch pseudo device using RDR: for input and LST: for output
	UC1:	Any handshaking RS-232 terminal with ETX/ACK protocol, eg. Diablo KSR 1640 printing terminal
RDR:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	PTR:	Null source, not implemented, returns an "end-of-file" character when accessed
	UR1:	An input modem at port 0D8H
	UR2:	System terminal

PUN:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	PTP:	Null sink, not implemented
	UP1:	An output modem at port 0D8H
	UP2:	System terminal
LST:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H (eg. WH-34)
	CRT:	Any non-handshaking RS-232 video terminal at port 0E8H
	LPT:	line printer (eg. H-14, WH-14, WH-24, H/Z25, WH-36, Epson MX80)
	UL1:	Any Diablo printer

Table 2-3:
Logical/Physical Devices

You must be certain that the destination device that is specified is capable of receiving data, and that the specified source device is capable of sending data.

ADDITIONAL LOGICAL DEVICES USED WITH PIP

All of the devices in the preceding table can also be referenced using the STAT utility, which assigns physical devices to logical devices on a temporary basis.

The PIP utility can also gain access to five additional devices, which are defined below:

NUL: Sends 40 "nulls" (ASCII zeroes) to the device (This device is usually accessed at the beginning and/or end of the output when a paper tape punch device is used.)

EOF: Sends an "end-of-file" character (ASCII CTRL-Z) to the destination device. (Such a transfer is performed automatically after PIP operation involving a file composed with ASCII characters.)

INP: Special input source which can be "patched" into the PIP program itself. Through this source, PIP accepts data input character by character using a system call to memory location 103H. The data returns from location 109H. The parity bit must be preset at zero.

OUT: Special output destination which can be "patched" into the PIP program. PIP transmits data from register C to this destination using a system call to memory location 106H. It should be noted that locations 109H through 1FFH of the PIP memory image are not used, and can be replaced by special purpose drivers using DDT.

PRN: This device is accessed for the same purposes as the LST: device. In its operation, however, tabs are expanded at every eighth character position, lines are numbered, and page breaks occur every 60 lines. Output directed to the LST: device will be treated identically if the [T8NP] or [T8NP60] parameter is entered with the LST: argument.

8.2 Suspending PIP Operations

When sending material with PIP to the CON: logical device, the copy operation can be suspended by entering the ***CTRL-S*** character at the keyboard, and resumed by entering any character other than CTRL-C. The operation can be aborted by entering any keyboard character other than CTRL-S while data is being transferred. PIP will respond to such an entry with the message:

```
ABORTED
```

9 USING PIP WITH A ONE-DRIVE HARDWARE ENVIRONMENT

When performing a PIP operation in a one-drive hardware environment, the "PIP" command should be entered alone at the system prompt (using invocation method 1.1) to produce the "*" "*" prompt.

When using the PIP utility to copy a file to a file on another disk, the disks involved must be inserted in the drive alternately to allow PIP to read from one file and then write to another. The command line argument should be entered as follows:

B:{destination file name} = A:{source file name}** ***RETURN

Where "**B:**" specifies logical drive device B;

where "**A:**" specifies logical drive device A; and

where both A and B are referred to as physical drive device 0 (the only physical drive device in the hardware environment).

This entry causes PIP to display the message:

```
Put disk B in drive A: and press RETURN
```

The disk desired to receive the file copy should be placed in the drive, and a carriage return entered. PIP will then display the following message:

```
Put disk A in drive A: and press RETURN
```

The disk containing the data source file should be placed in the drive, and a carriage return entered. You should repeat these two steps alternately, as the PIP prompts indicate, until the entire source file has been copied to the destination file. This process will vary in length, depending on the size of the file being copied. You should be careful to keep track of which disk is A and which is B.

10 PIP's METHOD OF OPERATION

File names and device names can both be used in a single data source argument, with PIP reading each name entered (starting with the name entered on the left-hand side of the source argument) until it reaches "end-of-file". "End-of-file" is indicated by a CTRL-Z character in ASCII files, and by the actual end of the file in non-ASCII files. Each of the data source names entered is read and then concatenated to the preceding name (to its left) until the last name has been read.

The data destination device or file receives a copy of the data from the source files and/or devices. When data from an ASCII file is written, the CTRL-Z "end-of-file" character is appended to the result.

As the PIP operation begins, a temporary file is established in the directory of the destination disk. This file has a name that consists of the primary name specified in the data destination, and a "$$$" extension. This temporary file is not changed to the actual file (with the extension specified in the command line argument) until successful completion of the PIP operation.

Files with a "COM" extension are always assumed to be non-ASCII files.

11 PIP PARAMETERS

The PIP utility performs enhanced data transfer operations when command line parameters are used. Most PIP parameters should be entered after the data source and enclosed between a single set of square brackets. A carriage return must follow the closing bracket. Most parameters are entered in the following form:

{data destination}={data source}{[parameters]}** ***RETURN

Where {**data destination**} is a specification of the file, disk, or device to receive the transferred data;

where {**data source**} is one or more specifications of the file(s) or device(s) from which the transferred data will be copied; and

where {**[parameters]**} is the specification of PIP parameters used to regulate the transfer operation.

Several parameters can be enclosed between brackets in a single PIP command line.

NOTE: If you wish to concatenate object files with a PIP operation, you must specify the O parameter after **each** file name that is specified as part of the data source.

The parameters used to regulate transfer operations are as follows:

B Block mode transfer: PIP transfers data from a source device to a buffer (data reservoir in computer memory), which retains bits of a continuous flow of data before writing it on a disk. The buffer is emptied to a disk when the CTRL-S character is transmitted to the buffer from the source device. PIP then resumes transmitting the continuous flow of data from the source device. If the buffer overflows with data, PIP will display an error message.

Dn Delete characters: Causes PIP to transfer all of the characters within the source file or source device—except for the characters to the right of the nth column, which are truncated so that the file can be sent to a narrow printer or console.

E Echo transfers: Causes display indicating what units of data are being transferred, as they are being transferred.

F Filter Form Feeds From File: Removes form feeds that are embedded in a file. The "P" parameter can be entered in the same command line to insert new form feeds.

Gn Get file from user area "n": Enables user to access files through a different user area number "n" (in the range 0-15). Can only be implemented if the file "PIP.COM" is in the current user number.

H Hex data transfer: Transferred data is checked for proper Intel hexadecimal file format, and unnecessary characters between hex records are removed during the transfer operation. If errors occur, PIP will display prompts for corrective action.

I Ignore records: Transferred Intel hex files will not include ":00" records. The I parameter automatically sets the H parameter.

L Lower case letters: Translates all copied alphabetic characters to lower case.

N Number lines: Applies line numbers to the beginning of each transferred line of data. The numbers begin at one and increase sequentially by ones. The numbers are separated from data lines by a colon. If "N2" is specified, then each number is preceded by a zero and followed by a tab space. This tab space is expanded if the "T" parameter is also set.

O Object file transfer: The normal CP/M end of file will be ignored in any data transferred with this parameter. This parameter is useful in transferring files with non-ASCII characters.

NOTE: If you wish to concatenate object files with a PIP operation, you must specify the O parameter after the file name of each object file that you wish to concatenate. Such an operation can be caused by a command line in the following form:

A>PIP B:ALL.OBJ=FILE0.OBJ[O],FILE1.OBJ[O],FILE2.OBJ[O] ***RETURN***

Pn Page ejects: Transferred data is interrupted at the beginning of the file and every "n" lines thereafter. If "n" is omitted from the parameter, the value "1" is assumed. If the value "1" is entered or assumed with the parameter, page ejects will occur every 60 lines. If the "F" parameter is entered in the same command line, then form feed suppression takes place before the new page ejects are inserted.

Qstring ^ Z Quit copying block: The text "string" between the "Q" and the " ^ Z" in this parameter marks the end of a block of text to be transferred. (Block begins with text "string" in the "Sstring ^ Z" parameter.

R Read system files: Enables transferral of files with the "SYS" status

Sstring ^ Z Start copying block: The text "string" between the "S" and the " ^ Z" in this parameter marks the start of a block of text to be transferred. (Block ends with text "string" in the "Qstring ^ Z" parameter.)

NOTE: the strings in the S and Q parameters are translated to upper case by the Console Command Processor if PIP is invoked using the system prompt method (entire command on one line):

A>PIP B:THIS.DOC=C:THAT.DOC[STHE BEGINNING ^ ZQTHE END. ^ Z]

Therefore, the file's actual text must be in upper case to match the string. However, if PIP is invoked using the PIP prompt method (command line argument entered in response to PIP "*" prompt):

A>PIP
***B:THIS.DOC=C:THAT.DOC[SThe beginning ^ ZQthe end. ^ Z]**

then the string text will not be automatically translated into upper case.

Tn Tab expansion: Expands each occurrence of tab character to every "n"th column in transferred text.

U Upper case: Translates lower case alphabetic characters to upper case during text transferral.

V Verify operation: Compares copy of transferred data with original to ensure exact correspondence. Might increase command execution time by as much as two times. This parameter only works if the data destination is a disk file.

W Write over files: Writes newly transferred data file over old data file, even if old file has been given R/O (read only) status. PIP will not prompt you before deleting the old file.

Z Zero parity bit: Sets parity bit on input to 0 for each ASCII character in transferred data.

12 PIP ERROR MESSAGES

DISK READ ERROR

EXPLANATION: Source of data is flawed. You should repeat PIP command or replace data source.

DISK WRITE ERROR

EXPLANATION: Destination disk did not have enough space for the copied data. You should clear out some space on the destination and repeat PIP command.

VERIFY ERROR

EXPLANATION: Destination disk has flawed media. You should replace destination disk and repeat PIP command.

ABORTED

EXPLANATION: A keyboard entry during a PIP data transfer caused this message and an end to PIP execution.

BAD PARAMETER

EXPLANATION: You entered an invalid parameter or an invalid bracket in a PIP command line.

INVALID USER NUMBER

EXPLANATION: You specified a user number out of the 0-15 range.

INVALID DIGIT

EXPLANATION: You entered an invalid digit for a PIP parameter, and should re-enter the entire command line.

CHECKSUM ERROR

EXPLANATION: PIP verification found a discrepancy between a data source and data destination file. The PIP operation should be repeated.

INVALID FORMAT

EXPLANATION: You entered PIP command line in improper form, and should repeat entry of command.

NO DIRECTORY SPACE

EXPLANATION: The directory of the destination disk is full.

```
NO FILE: =x:filename.ext
```

EXPLANATION: PIP could find no source file by the name "filename.ext" on the disk in drive "x".

```
START NOT FOUND
```

EXPLANATION: PIP could not find the beginning of a string that was specified with the "S" and "Q" parameters. You should re-enter command specifying a string beginning that exists in the data source.

```
QUIT NOT FOUND
```

EXPLANATION: PIP could not find the end of a string that was specified with the "S" and "Q" parameters. You should re-enter command specifying a string end that exists in the data source.

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

EXPLANATION: This message prompts you that the destination already has a file by that name. Use can have the old file destroyed by answering Y to this prompt.

```
NOT FOUND
```

EXPLANATION: You should re-enter the PIP command with the name of an accessible data source.

```
REQUIRES CP/M 2.0.0 OR NEWER FOR OPERATION
```

EXPLANATION: PIP must be used with CP/M version 2.2.02 or 2.2.03 or 2.2.04.

```
UNRECOGNIZED DESTINATION
CANNOT WRITE
```

EXPLANATION: You should re-enter a PIP command with a valid data destination.

```
INVALID PIP FORMAT
CANNOT READ
```

EXPLANATION: You made syntax error in command line, and PIP could not write to the data destination.

```
INVALID SEPARATORS
```

EXPLANATION: You should re-enter PIP command with the proper square brackets around a parameter.

PREL

The Utility that Creates a Relocatable File from Two Hexadecimal Output Files

The PREL utility is invoked (1) to take the hex output files of two assemblies and generates a relocatable file from them (2). The first of the assemblies should have as an initial origin of 000H and the second assembly should be one page higher, or 0100H. The hex output of the first assembly should be given a file name extension of "HX0", and the hex output of the second should have "HX1" for its extension. PREL can be effectively implemented as one of the batched commands in a SUB file (3).

1 PREL INVOCATION

A PREL command line is entered in the following form:

A>**PREL {hex file name} {relocatable file name}** ***RETURN***

Where {**hex file name**} is the primary name of the two assembled hex files (These files are given the extensions HX0 and HX1); and

where {**relocatable file name**} is the primary name of the output page relocatable file. (The output file will have the extension PRE.)

For example, the command line:

A>**PREL MYPROG TESTBIOS** ***RETURN***

would create the file TESTBIOS.PRE from the two hex files MYPROG.HX0 and MYPROG.HX1.

2 PREL FUNCTION

PREL will compare the two hex files and perform two separate functions:

- PREL converts the hex file to a binary file (similar to the function of the LOAD utility.)

- PREL appends a relocation table to the end of the file which includes a bit for each byte in the module. If a particular bit has a value of one, then the associated byte must be offset when the program is loaded into memory. An error message will result if the hex values for a given byte differ by more than 0 or 1. An error message will also result if the addresses in the hex files do not increase sequentially.

3 PREL OPERATION

All addresses that need to be relocated should be expressed relative to the base of the module, and not defined absolutely. A convenient way to do this is to omit "ORG" statements from the program and to use the SUBMIT utility to enter batched commands from a SUB file. The commands in the SUB file should perform the required assemblies and the conversion to a relocatable format. An example of this type of batched command might be:

```
PIP TEMP0.ASM=ORG0.ASM,$1.ASM
ASM TEMP0.AAZ
REN $1.HX0=TEMP0.HEX
ERA TEMP0.ASM
PIP TEMP1.ASM=ORG1.ASM,$1.ASM
ASM TEMP1.AAZ
ERA TEMP1.ASM
REN $1.HX1=TEMP1.HEX
PREL $1 $1
ERA $1.HX0
ERA $1.HX1
```

where the files ORG0.ASM and ORG1.ASM contain nothing but "org 0000H" and "org 0100H" statements respectively. This command script could be invoked with the entry:

A>**SUBMIT MAKEPRE MYPROG *RETURN***

Where the command sequence is contained in a file called MAKEPRE.SUB, the original assembly file is called MYPROG.ASM and the final result is a page relocatable file called MYPROG.PRE.

4 PREL ERROR MESSAGES

```
Phase error
```

EXPLANATION: Hexadecimal files do not contain correct starting addresses.

```
Cannot open input file
```

EXPLANATION: The file with the extension "HX0" or "HX1" was not found.

```
Cannot create output file
```

EXPLANATION: The directory on the destination disk is full.

```
Hex files not monotonic
```

EXPLANATION: The hexadecimal addresses are not increasing sequentially.

```
Write error
```

EXPLANATION: The destination disk is full.

REN

The Resident Command that Renames Files

The REN command allows you to assign a new name to any disk file (1). If it cannot rename a file, it also displays the reason (2).

1 RENAMING A FILE

To rename a file, you should respond to the system prompt with a command line in the following form:

A>**REN {new name}={old name}** ***RETURN***

Where {**new name**} is the name you wishes to assign to the file; and

where {**old name**} is the name of the file before it is renamed.

For instance, the name of the file "BIOS.SYS" can be assigned to a file currently named "1747.SYS" by entering the following command:

A>**REN BIOS.SYS=1747.SYS** ***RETURN***

If file receiving the new name does not reside on a disk in the default drive, then the REN command line must include a drive specification which precedes the new file name, as shown:

A>**REN C:BIOS.SYS=1747.SYS** ***RETURN***

2 CONSOLE RESPONSE TO RENAMING COMMANDS

If a file is successfully renamed, the system prompt will appear.

```
A>
```

If the file being renamed does not exist on the default drive and no drive is specified (or if an incorrect drive is specified), then REN cannot find the file you want to rename and will display the error message:

```
NO FILE
```

If you try to assign a file a name that is actually the current name of an existing file, REN will not perform the change and displays the error message:

```
FILE EXISTS
```

If the file being renamed has Read/Only access status (as described in the text on the STAT utility), REN does not rename the file. However CP/M will display an error message in the following form:

```
Bdos Err on x: File R/O
```

SAVE

The Resident Command that Copies Data from Memory to a Disk File

The SAVE command is used to copy the hexadecimal contents of a span of memory locations to a disk file (1). The amount of data that SAVE will copy is measured in pages (2). The data that SAVE copies from the computer's memory space can come from a variety of sources (3).

1 SAVE COMMAND LINE

To SAVE a program onto a disk and create a file for it, you should respond to the system prompt with a command in the form:

A>**SAVE {pages} {file name}** ***RETURN***

Where **{pages}** is the decimal number of pages of memory contents that SAVE should copy to the disk, and

where **{file name}** is the name of the file created to store the data.

NOTE: A page of memory in CP/M is 256 (decimal) bytes, or a span of 0100H (hexadecimal) memory locations.

For example, to copy 3 pages (the memory contents between address 100H and address 3FFH) of data from main memory to the disk in non-default drive B and give this block of data the file name "PROGRAM.ASM", you would enter the following command line:

A>**SAVE 3 B:PROGRAM.ASM** ***RETURN***

2 SAVE DESCRIPTION

Data in the computer's Transient Program Area (TPA) memory space can be blocked off by specifying the first and last memory locations that it occupies. Programs loaded into the TPA always begin at memory location 0100H (hexadecimal). This is also the location from which SAVE starts copying blocks of data. Therefore, if you specified one page (100H bytes) in the SAVE command line, SAVE will start copying data from memory location 100H and continue through memory location 1FFH.

Only whole pages can be transferred; and SAVE recognizes only decimal integers for pages.

3 COMMON APPLICATIONS FOR SAVE

SAVE will copy the hexadecimal values for any data that resides in memory locations 100H through 32768H (in a 32K computer). This area of memory is known as the Transient Program Area, and it is used to accommodate data manipulated by utilities and application programs. Data often remains in these locations after being manipulated by CP/M utilities.

One useful application is to SAVE a program that has been placed in memory by the DDT utility.

Another common application is to SAVE a copy of the operating system onto a file. The CP/M operating system always occupies the area of memory between locations 0H and 100H. However, if this operating system is loaded into the Transient Program Area by a MOVCPM utility, it can be saved onto a file. To do this, you must enter a command in the following form:

A>**SAVE 38 CPMkk.COM** ***RETURN***

Where "**kk**" is the number of kilobytes of memory to which the system has been adjusted by the MOVCPM utility.

NOTE: To produce the system prompt (A>) and allow entry of the SAVE command, you must exit from the utility with a warm boot (performed by entering ***CTRL-C***).

4 COMPUTING PAGES TO BE SAVED

To SAVE a program from memory, you must determine how many pages this block occupies. This requires a conversion of the program's hexadecimal size to decimal pages.

The hexadecimal size of a program is displayed when the program is loaded into the TPA under DDT. This display indicates that the program begins at address 0100H and ends at an address identified as "NEXT". You should take the two left-hand digits from the "NEXT" value, and convert them into a decimal number. This decimal number is the number of pages to be saved.

For example, if you are debugging a program that occupies memory locations 0100H through 28FFH, the results of this debugging activity can be recorded by taking the hexadecimal digits "28", converting them into the decimal value "40", and entering the following command:

```
A>SAVE 40 PROGRAMX.HEX    RETURN
```

If the disk to which you try to SAVE a data block does not have enough space to record a file of the specified size, then the following error message will be displayed:

```
NO SPACE
```

SETLP

The Utility that Temporarily Sets Printer Characteristics

The SETLP utility enables you to change the operating system's baud rate setting for a serial printer and to determine whether the LPT device will be used to run a serial printer or a parallel printer. The SETLP utility can be run with a single command line.

Baud rate is the speed with which data is transferred between the computer and a serial printer. Baud rate is approximately equal to the number of bits transferred per second. A baud rate setting is not necessary when you are using a parallel printer.

1 REASONS TO USE SETLP

The system's baud rate setting for the LST: device (usually used for a printer) can be changed by running the CONFIGUR utility. However, SETLP was designed as a convenience so that you could change the baud rate setting without encountering all of the menus and options of the more extensive CONFIGUR program. SETLP can temporarily alter the baud rate values kept in memory, but it will not affect the baud rate values stored on the disk by CONFIGUR.

Hence a baud rate set by SETLP will remain in effect only until the computer is reset or until another SETLP command is executed to change the baud rate.

SETLP is useful if you have more that one kind of printer and use each of these printers alternately. However, if the printers used are linked to the input/output ports by different physical device names, then CONFIGUR or STAT must be used to change the operating system for these physical device names.

NOTE: SETLP does not change the baud rate of the printer itself. A printer's baud rate is usually changed by manual printer adjustments that are explained in the manual(s) supplied with your printer(s).

SETLP can also be used to quickly set the system to run either a serial printer or a parallel printer through the LPT: device.

2 SETLP COMMAND METHODS

The SETLP utility is invoked and executed by entering a command line in the H-89-3 Command Method, the H-89-11/LPT: Command Method, or the H-89-11/UL1: Command Method.

2.1 H-89-3 Command Form

This SETLP method is helpful to users with an H-89-3 interface card:

A>**SETLP {baud}** ***RETURN***

Where **SETLP** is the command line function; and

where **{baud}** is the number for the baud rate to which the user wishes to temporarily adjust the operating system. The baud rate must be specified one space to the right of the SETLP function. No further command line argument is necessary.

The printer for which you are setting this baud rate must use the physical device assigned to logical device LST:. This match is made through the CONFIGUR utility or the STAT utility.

For example, the command

A>**SETLP 1200** ***RETURN***

could be entered to temporarily change the system baud rate setting to 1200.

2.2 H-89-11/UL1: Command Method

This SETLP method is helpful to users with H-89-11 interface cards and printers that are accessed through the UL1: physical device. (Diablo daisy wheel printers are accessed through this device.) It enables these users to temporarily change the baud rate that the operating system uses to send data to these printers.

A>**SETLP {baud}** ***RETURN***

Where **SETLP** is the command line function; and

where **{baud}** is the number for the baud rate to which the user wishes to temporarily adjust the operating system. The UL1: physical device must currently be matched with the LST: logical device. (The CONFIGUR or STAT utilities can help you to match these devices.)

For example, the command

A>**SETLP 300** ***RETURN***

could be entered to temporarily change the baud rate to 1200 of a serial printer accessed through the UL1: device.

NOTE: Using the H-89-11/UL1: Command Method, the baud rate for the UL1: physical device can be changed without affecting the baud rate for the LPT: physical device.

2.3 H-89-11/LPT: Command Method

This SETLP method is helpful to users with an H-89-11 interface card and a printer that is accessed through the LPT: physical device. (Most dot-matrix printers are accessed through this device.) It enables these users to temporarily change the system's baud rate setting for serial printers, and/or to quickly set the system to run either a serial printer or a parallel printer through the LPT: device.

A>**SETLP {parameter} {baud}** ***RETURN***

Where **SETLP** is the command line function;

where **{parameter}** is the letter that stands for a type of printer. The letter **S** is used for serial printers, and the letter **P** is used for parallel printers; and

where **{baud}** is a decimal number for the baud rate to which the user wishes to temporarily adjust the operating system. A baud rate can be specified for serial printers, but not parallel printers. The LPT: physical device must currently be matched with the LST: logical device. (The CONFIGUR or STAT utilities can help you to match these devices.)

For example, the command

A>**SETLP S 4800** ***RETURN***

could be entered to temporarily change the system baud rate to 4800, for a serial printer accessed through the LPT: device.

You can also temporarily change the system baud rate to 4800, for a serial printer accessed through the LPT: device. To make this change, enter the following command:

A>**SETLP 4800** ***RETURN***

The result of entering the preceding two example commands is identical—even though the S parameter is not used in the second command—because the system assumes that you wish to set up for a serial printer whenever you specify a baud rate in a SETLP command.

The following example command

A>**SETLP P *RETURN***

could be entered to temporarily change the system so that it can access a parallel printer through the LPT: device.

NOTE: The "Default I/O Configuration" menu (submenu C) in the CONFIGUR utility and the STAT utility both show whether the LST: logical device (for printers) has been matched with the LPT: or UL1: or an other physical device.

3 POSSIBLE BAUD RATES

SETLP can be used to adjust the operating system for any of the following baud rates:

75	150	1200	9600
110	300	2400	19200
134	600	4800	38400

Make certain that any baud rate entered in a SETLP command line matches the baud rate that has been set on the serial printer being used. Instructions for setting (or resetting) the baud rate of a printer can be found in your printer's manual.

4 SETLP ERROR MESSAGES

```
ERROR - Invalid argument
```

EXPLANATION: The command entered did not contain syntax for the parameter and/or baud rate; or SETLP was used with the wrong interface card. Enter a command that uses the syntax described in "2 SETLP Command Methods", and sets up the system for hardware that is connected in your hardware environment.

```
ERROR - Improper configuration
```

EXPLANATION: You tried to run SETLP with the wrong version of CP/M. Use a CP/M Operating System and a SETLP utility that has been copied from the same set of CP/M distribution media. Then enter the command again.

STAT

The Utility that Reports Disk Statistics and Assigns Types of Status

The STAT utility has two functions: to provide statistical information about disk space or file size (1), and to allow you to change the assignment of various types of status (2).

1 REPORTING STATISTICS CONCERNING DISKS AND FILES

1.1 Available Disk Space

The amount of unoccupied space on the disk in the default drive that is available for use can be determined by entering the command "STAT" and a carriage return at the default drive prompt, as in the following example:

A>**STAT** ***RETURN***

If drive A is the only drive that has been logged since the system was booted, then STAT will respond with a display like the following:

```
A: R/W, Space: 3k
```

Where "A:" indicates that the disk in drive A was investigated by STAT;

where "R/W," stands for Read/Write, meaning that data and files can be written to or read from the disk. If STAT encounters a disk that has been write protected, it will report that the disk is "R/O" or read/only. Data and files cannot be recorded to or erased from a read/only disk; and

where "3k" indicates the amount of space that is unoccupied on the disk, and thus available to record files. Memory space is expressed in units of 1024 bytes, or kilobytes (k). One byte holds one character. The disk that was investigated for the preceding example has enough remaining memory space to store 3072 characters.

If you have logged other drives since the system was last booted, the preceding response to the basic "STAT" command line will also produce a listing for the disks in each of the drives that have been logged, as shown:

```
A: R/W, Space: 3k
B: R/W, Space: 18k
C: R/W, Space: 65k
```

Even if a non-default drive has not yet been logged, you can still ascertain the space remaining on the disk within it by specifying the drive in the STAT command line:

A>**STAT B:** ***RETURN***

The entry of this command line might produce a response in the following form:

```
Bytes Remaining ON B: 24k
```

1.2 Disk Space Consumed by a File

To find out how much space one particular file occupies, requires specification of the file name and the drive in which the file resides. An appropriate entry for such a command would be:

A>**STAT C:PRINTOUT.DOC *RETURN***

Such an entry might produce a display in the following form:

```
Recs  Bytes  Ext Acc
 140    18k    2 R/W C:PRINTOUT.DOC
Bytes Remaining ON C:  72k
```

Where "Recs" stands for the number of records within the file. The file PRINTOUT.DOC contains 140 data records. A record is the amount of data it takes to fill up 128 bytes (one-eighth of a kilobyte) of memory space. Thus, one record consists of 128 data characters. However, STAT cannot report fractions of records; and if a file contains anywhere from 1 to 128 characters, STAT will still report that the file contains one record;

where "Bytes" stands for the amount of memory space the file occupies, expressed in kilobytes (1024 bytes). In this display, STAT reports that the file PRINTOUT.DOC occupies 18 kilobytes of memory space. One kilobyte of space will hold 8 records. However, STAT cannot report fractions of kilobytes. If a file contains anywhere from 1 to 8 records, STAT will still report that it occupies one kilobyte of space;

where "Ext" stands for the number of extents that the file occupies. In this display, STAT reports that the file PRINTOUT.DOC occupies two extents. One extent will process 16 kilobytes of data. However, STAT cannot report fractions of extents. If a file occupies anywhere from 0 to 16 kilobytes, STAT will still report that it requires one extent to be processed; and

where "Acc" indicates the file's access status. The "R/W" access status (read/write) enables you to run commands to freely erase data from the file, or add data to the file. The "R/O" access status imposes limitations on the commands that you can run to change the data in a file. The file PRINTOUT.DOC has R/W status.

1.3 Disk Space Consumed by Groups of Files

To obtain the status of all of the files on a disk without entering a command line specifying each file on the disk, the drive letter and the ambiguous filename "*.*" can be entered, as in the following command line:

A>**STAT B:*.*** ***RETURN***

Such an entry might produce a display like this:

```
 Recs  Bytes  Ext Acc
  175    22k    2 R/W    B:DATE0927.DOC
  160    20K    2 R/W    B:DATE1003.DOC
   15     2k    1 R/W    B:REPORT1.TXT
    9     1K    1 R/W    B:REPORT2.FIL
  119    15k    1 R/O    B:THATFILE.COM
  201    25k    2 R/O    B:THISFILE.COM
Bytes Remaining On B:  26k
```

(Such an operation will only reveal the files within the currently logged user area.) STAT can also convey the status of groups of files on a disk when ambiguous file names are entered with the asterisk and/or question mark representing only part of the file names, as in the following sample command lines:

A>**STAT B:*.COM** ***RETURN***

A>**STAT B:DATE????.DOC** ***RETURN***

A>**STAT B:REPORT?.*** ***RETURN***

1.4 Measuring the Size of a File

In addition to ascertaining the actual amount of disk space occupied by a file, STAT can also report on the outermost boundaries of the disk space occupied by a file, or the difference between the first and last sectors occupied by the file.

In the case of sequential files, the size of the file is the same as the number of records because records are recorded on adjacent sectors. The records of Random files, however, are often scattered over a wider disk area than they actually occupy. To determine the size of a file, the normal command line for status of a particular file is entered, followed by a space and the characters "$" and "S", as shown:

A>**STAT C:PRINTOUT.DOC $S *RETURN***

The preceding entry might produce the response:

```
 Size  Recs  Bytes  Ext Acc
  140   140    18k    1 R/W C:PRINTOUT.DOC
Bytes Remaining On C: 72k
```

NOTE: If disks are switched from one drive to another between STAT runs, the individual file statistics will be accurate, but the "Bytes Remaining on X: nnk" message will display a kilobyte count that reflects the remaining disk space on the disk that was first logged within that particular drive since bootup. Therefore, to ensure a correct listing of remaining disk capacity, you should perform a warm boot (enter ***CTRL-C***) if disks are switched between STAT runs.

2 ASSIGNING STATUS

The STAT utility also performs functions which allow disk protection, manipulation of file indicators, disk parameter statistics, user number display, and input/output device assignment. A list of these special STAT functions can be invoked by entering the following command:

A>**STAT VAL:** ***RETURN***

In response, STAT will display the following list:

```
Temp R/O Disks: d:=R/O                                        (2.1)
Set Indicator:  d:filename.typ $R/O $R/W $SYS $DIR  (2.2, 2.3)
Disk Status   : DSK: d:DSK:                                   (2.4)
User Status   : USR:                                          (2.5)
Iobyte Assign:                                                (2.6)
CON: = TTY: CRT: BAT: UC1:
RDR: = TTY: PTR: UR1: UR2:
PUN: = TTY: PTP: UP1: UP2:
LST: = TTY: CRT: LPT: UL1:
```

2.1 Temporarily Changing a Disk's Access Mode

The category title "Acc" in many STAT displays stands for the access mode of a file. A file's access mode could be either R/W or R/O. The R/W (Read/Write) mode allows you to write data to the file or erase data from it. The R/O (Read/Only) mode prevents insertion or deletion of data within the file.

To temporarily change the access mode of all files on a disk from R/W to R/O, requires the entry of "STAT", a space, the name of the drive containing the disk to be protected, a colon, an "=" sign, the new access mode "R/O", and a carriage return. The following example illustrates such a command line:

A>**STAT B:=R/O** ***RETURN***

After the entry of this command line, no data or software on disk B can be erased or overwritten. The protection furnished by the R/O access mode will remain in effect until a warm or cold boot is performed.

If you protect files in this manner, invoke the ED utility, and try to write to or delete from a file on the disk, the following error message is displayed:

```
Bdos Err On B: R/O
```

CP/M will not allow alteration to file contents, and will perform a warm boot with the next key pressed after this message is displayed. After this warm boot is performed, the disk's files are reset to the R/W access mode.

2.2 Changing a File's Access Mode

The R/W (Read/Write) mode allows you to write data to the file or erase data from it. The R/O (Read/Only) mode prevents insertion or deletion of file data, and it can be applied to files by the set indicator commands "$R/O" and "$R/W".

Individual files, or groups of them, can be given the R/O status with the entry of "STAT", a space, the drive specification, the file name, a space, the characters "$R/O", and a carriage return. For instance, the file named "ARCHIVAL.DOC" on the disk in drive B can be protected from alteration by the following entry:

A>**STAT B:ARCHIVAL.DOC $R/O *RETURN***

To give R/O status to a group of files, enter the "*" or "?" wildcard characters in place of the variable part of the names of the files in the group. After this command has been given, the following message is displayed:

```
ARCHIVAL.DOC set to R/O
```

Attempts to write to or erase from this file using any text editor will produce the message:

```
Bdos Err On B: File R/O
```

You must enter a keyboard character to warm boot the system. The R/O attribute is recorded in the directory and will not change if the system is rebooted. The file can be made Read/Write again by entering a similar command with the characters "$R/W" in place of "$R/O", as in the following example:

A>**STAT B:ARCHIVAL.DOC $R/W *RETURN***

2.3 Changing a File's System Status

Individual files, or groups of them, can be given the system status, which prevents their mention in directory listings, prevents access by the editor, and prevents copying by PIP (except with the "[R]" parameter). This status is applied to a file when the entry of "STAT", the drive specification, and the file name is followed by the characters "$SYS". For instance, the file "SYSTEM.COM" on drive B becomes ineligible for directory mention, editing, and copying (except with [R]) when the following command is entered:

A>**STAT B:HIDEFILE.COM $SYS *RETURN***

After this command has been given, the message:

```
HIDEFILE.COM set to SYS
```

is displayed.

To illustrate how this status protects a file, efforts to edit the file using the ED utility will produce the message:

```
"SYSTEM" FILE NOT ACCESSIBLE
```

In addition, a check of this file using a different form of the STAT command will also produce noteworthy results, as shown. If you now enter the command:

A>**STAT B:HIDEFILE.COM *RETURN***

the following display will appear:

```
 Recs  Bytes  Ext  Acc
   38     5k    1  R/W B: (HIDEFILE.COM)
Bytes Remaining On B: 19k
```

When a file bears the SYS status, it will have parenthesis around its listing in a STAT file report.

The system attribute can be removed from this file by the following entry:

A>**STAT B:HIDEFILE.COM $DIR *RETURN***

The **$DIR** argument in a STAT command line lifts all of the restrictions imposed by the $SYS argument in a previous STAT command.

2.4 Listing Disk Characteristics

Characteristics of all disks being used can be obtained by entering "STAT", a drive specification, "DSK:", and a carriage return, as shown:

A>**STAT B:DSK: *RETURN***

STAT will list a display similar to the following:

```
  B: Drive Characteristics
736: 128 Byte Record Capacity
 92: Kilobyte Drive Capacity
 64: 32 Byte Directory Entries
 64: Checked Directory Entries
128: Records/ Extent
  8: Records/ Block
 20: Sectors/ Track
  3: Reserved Tracks
```

Where the letter "B" to the left of "Drive Characteristics" indicates the selected drive, which contains the disk being analyzed (a 5.25-inch hard-sectored disk in the example above);

where the second and third lines indicate the disk's total "Record Capacity" and "Kilobyte Drive Capacity", respectively (these totals are sometimes dependent upon the selections made during the FORMAT utility);

where the directory can hold a maximum of "64" entries;

where the number of "Checked Directory Entries" is usually identical to the directory size for floppy disks, since this mechanism is used to detect changed disk media during CP/M operation without an intervening warm boot;

where the number of "Records/ Extent" indicates the addressing capacity of each directory entry;

where the number of "Records/ Block" indicates the basic size of a block. By knowing that a record contains 128 bytes, you can calculate the number of bytes in a block;

where the number of "Sectors/ Track" can help you to determine the number of sectors on the disk if you multiply this number by the number of tracks on the disk; and

where the "Reserved Tracks" are reserved for the operating system and the disk file directory.

Any drive from A through P can be specified in this command, as long as it is a valid drive in the hardware environment and operating system being used. If no drive is specified in the command line, then disk characteristics will be displayed for the disk in each drive that has been logged since the last cold boot.

2.5 Examining User Areas

User access to certain files can be controlled by assigning user areas to the files. (See text on USER resident command.) The following form of the STAT utility will display a list of the numbered user areas which have files on the currently logged disk.

A>**STAT USR:** ***RETURN***

This entry might produce the response:

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

The first line of the display lists the currently logged user area, or "Active User", which was set at zero either by the last USER command, or by a cold boot.

The second line lists the other user area numbers that have been established for files on the disk. This list is displayed as "Active Files", and indicates that additional files can be accessed by logging user area one or three.

To determine the "Active User" and "Active Files" on a non-default disk, a similar STAT command can be entered with a drive specification at the end of the command line, as in the following example:

A>**STAT USR:C:** ***RETURN***

You can examine the directories of the other user areas by first issuing the USER resident command with the appropriate area number. The DIR (directory) resident command will then help to determine what files are in a particular user area.

2.6 Temporarily Matching Logical and Physical Devices

The STAT utility also enables you to monitor and temporarily control the assignment of physical input/output devices to the appropriate logical device names. In general, there are four logical peripheral devices which, at any given time, are each assigned to one of several physical devices. The four logical devices, and the physical devices assigned to each of them, are shown in the display invoked by the **STAT VAL:** command, and in the following table:

LOGICAL DEVICE NAME	PHYSICAL DEVICE NAME	DESCRIPTION AND/OR CATALOG NAME OF RECOMMENDED NAME NAME INPUT/OUTPUT MACHINE
CON:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	CRT:	Any non-handshaking RS-232 video terminal at port 0E8H
	BAT:	A batch pseudo device using RDR: for input and LST: for output
	UC1:	Any handshaking RS-232 terminal with ETX/ACK protocol, eg. Diablo KSR 1640 printing terminal
RDR:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	PTR:	Null source, not implemented, returns an "end-of-file" character when accessed
	UR1:	An input modem at port 0D8H
	UR2:	System terminal
PUN:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H
	PTP:	Null sink, not implemented
	UP1:	An output modem at port 0D8H
	UP2:	System terminal
LST:	TTY:	Any non-handshaking RS-232 ASCII terminal at port 0D0H (eg. WH-34)
	CRT:	Any non-handshaking RS-232 video terminal at port 0E8H
	LPT:	line printer (eg. H-14, WH-14, WH-24, H/Z25, WH-36, Epson MX80)
	UL1:	Any Diablo printer

Table 2-4
Possible Device Assignments

EXAMINATION OF CURRENT DEVICE ASSIGNMENTS

The current assignments of the physical devices to logical devices are shown in the display invoked by the following command entry:

A>**STAT DEV:** ***RETURN***

This command produces a display similar in form to the following:

```
CON: is CRT:
RDR: is UR1:
PUN: is UP1:
LST: is UL1:
```

CHANGING DEVICE ASSIGNMENTS

The current logical to physical device assignment can be changed by entering a STAT command in the following form:

A>**STAT {logical device}:={physical device}:** ***RETURN***

Where both the "**logical device**" and the "**physical device**" are specified by their four-character names.

For example, the following entry will change the physical device assigned to the console logical device (CON:) to a teletype printer terminal (TTY:):

A>**STAT CON:=TTY:** ***RETURN***

Several device assignments can be affected in one command by entering equations in series, separated by commas, as shown:

A>**STAT RDR:=PTR:,PUN:=PTP:,LST:=LPT:** ***RETURN***

The physical device names may or may not actually correspond to the devices which the names imply. For instance, the PTR: device could be implemented as a cassette/write operation if you wish. The exact correspondence and driving subroutine is defined in the BIOS portion of CP/M.

Physical to logical device pairings assigned by the STAT utility will survive a warm boot, but are replaced by default assignments when a cold boot is performed. Default assignments, that will survive a cold boot, may be changed through the CONFIGUR utility.

3 STAT ERROR MESSAGES

```
Bad Delimiter
```

EXPLANATION: You entered a command line with improper syntax while assigning some type of status (2). Command should be re-entered in the proper form.

```
Invalid Assignment
```

EXPLANATION: You tried to make a device assignment with a device name that is not listed in Table 2-4. You should re-enter assignment using valid device names.

```
Invalid File Indicator
```

EXPLANATION: You tried to set a file for a particular status using an indicator or syntax that is not acceptable. You should re-enter command with a proper indicator.

```
File Not Found
```

EXPLANATION: STAT found none of the specified files on the logged disk. You should re-enter command specifying different file names, or perform a warm boot and insert a disk that has the specified file(s).

```
Wrong CP/M Version (Requires n.n)
```

EXPLANATION: The STAT.COM file in use must be used with the operating system version corresponding to the number ("n.n") in the error message.

```
Invalid Disk Assignment
```

You entered a disk assignment command that included a status other than "R/O" or "R/W". Re-enter the command with correct syntax.

```
** Aborted **
```

You tried to enter a command while a STAT status command was still executing. When you assign status (such as R/O, R/W, SYS, or DIR) to a file, you must wait for the status command to execute completely and for the system prompt to appear before you type the next command.

SUBMIT

The Utility that Triggers Automatic Execution of CP/M Commands

The SUBMIT utility enables you to initiate automatic execution of several sequential commands (3) by issuing only one SUBMIT command (2). The SUBMIT command accesses a file containing a precomposed sequence of commands (1). You can also vary the execution of this command sequence with each invocation by substituting command line parameters such as file names, drive names, and device names for variables inserted into the command file.

1 SUB FILES

A SUB file is a file with the "SUB" extension, which is comprised of a sequence of command lines composed using a text editor (eg ED). The SUB file is the source of the command sequence referenced by the SUBMIT command. The command lines within a SUB file can contain prototype parameters to represent command line fields. These prototype parameters appear in the SUB file in the form:

```
$n
```

Where each numeral ("n") preceded by a dollar sign ("$") takes the place of a file name, drive name, or device name which you will substitute into the SUB file with an actual parameter entered in the SUBMIT command line.

The following example shows the text of the SUB file with the name "MULTIJOB.SUB", which contains two prototype parameters:

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

2 SUBMIT COMMAND LINES

The SUBMIT utility is invoked and operated by user entry of a command line in the form of the following example:

SUBMIT MULTIJOB PROGRAMX LST *RETURN*

Where **MULTIJOB** is the name of the SUB file filled with batch commands; and

where **PROGRAMX** and **LST** are parameters which will be substituted into one or more locations in the batch file.

An actual SUBMIT parameter is the explicit specification of a command, file, drive, or device which you want to substitute for a prototype parameter within the SUB file. The number of parameters specified in a SUBMIT command line must correspond to the number of different prototype parameters that appear within the SUB file in the form "$n".

3 SUBMIT EXECUTION

After the entry of a SUBMIT command line, SUBMIT will create a file named $$$.SUB, and send it to the Console Command Processor (CCP) portion of the operating system. The CCP recognizes the commands in the SUB file and executes them in sequence. The $$$.SUB file contains the actual parameters expressed in the SUBMIT command substituted for the corresponding prototype parameters which are imbedded in the original SUB file. If created in response to the preceding example command line, the $$$.SUB file would contain the following commands:

```
ASM PROGRAMX
DIR PROGRAMX.*
ERA *.BAK
PIP LST:=PROGRAMX.PRN
ERA PROGRAMX.PRN
```

4 SUBMITTING A SUB FILE FROM A DRIVE OTHER THAN A

The $$$.SUB file must exist on a disk in drive A for the commands to be executed. However, the SUB file used by SUBMIT to create the $$$.SUB file can reside on a disk in any existing drive; and the $$$.SUB file can be created on a disk in any existing drive.

If $$$.SUB is created on a disk in a drive other than drive A, you must adhere to one of the following sets of stipulations to initiate command execution:

- The drive location of the disk containing the SUB file must be specified in the SUBMIT command line. $$$.SUB is created on a bootable disk, the disk is placed in drive A, and the system must be rebooted with drive A as the default drive.

- The drive location of the disk containing the SUB file must be specified in the SUBMIT command line. The $$$.SUB file can be created on any disk and then copied to a disk in drive A using the PIP utility. The disk in drive A must be write enabled and have sufficient space to accommodate the $$$.SUB file.

5 ABORTING SUBMIT COMMAND EXECUTION

Execution of submitted commands can be aborted at any time by entering a ***DELETE, RETURN,*** or ***CTRL-C*** when the command is read and echoed. In this case, the $$$.SUB file is removed, and the subsequent commands come from the console. Command processing is also aborted if the CCP detects an error in any of the commands. Programs which execute under CP/M can abort processing of command files when error conditions occur by simply erasing any existing $$$.SUB file.

6 UNCONVENTIONAL TEXT IN A SUB FILE

You can insert comments into a SUB file so that they will be displayed with the other messages produced by the processing of SUB file commands. To ensure that the CCP ignores the comments and does not interpret them as commands, they must be entered on a line which begins with a semicolon (;).

To introduce literal dollar signs into a SUB file, you may enter "$$", which translates to a single "$" within the SUB file. Furthermore, a caret symbol (^) can precede the alphabetic character X, which produces a single CTRL-X character within the file.

7 CHAINED BATCH COMMANDS

The last command in a SUB file can be a SUBMIT command, which could either initiate execution of the commands within another SUB file, or repeat execution of the commands within the same file. You can include different parameters to be substituted for SUB file prototypes to cause varied repetition of the same set of commands.

8 SUBMIT ERROR MESSAGES

```
Error on Line nnn      No 'SUB' File Present
```

EXPLANATION: SUBMIT command can only be issued if a file with the "SUB" extension exists on a logged disk, and if that file is specified in the SUBMIT command line. You should re-enter the command with a different argument, or log the disk with the appropriate SUB file.

```
Disk Write Error
```

EXPLANATION: SUBMIT found insufficient disk space while trying to create $$$.SUB file on the disk. You should clear off some space on the disk or use another disk.

```
Parameter Error
```

EXPLANATION: You entered an invalid parameter in the SUB file. You should edit SUB file, with only decimal integers following the "$" sign for a parameter.

```
Directory Full
```

EXPLANATION: SUBMIT tried to create SUB file on a disk that didn't have enough directory space to accommodate the SUB file name. You should clear off some space on the disk or use another disk.

```
Cannot Close, Read/Only?
```

EXPLANATION: SUBMIT is trying to write data to a disk (as one of the activities prescribed in a SUB file command line), but the file being written to has R/O (read/only) status. You must use the STAT utility to change the file's status to "R/W" (read/write); or change the SUB file command line that is trying to write to the file; or enter different SUBMIT parameters.

SYSGEN

The Utility that Puts the Operating System on a Disk

The SYSGEN utility is used to transfer the operating system to a disk. Under some circumstances, SYSGEN does this task by itself. Sometimes SYSGEN needs the help of other utilities before an entire, usable system can be put on the disk.

1 SYSGEN INVOCATION

No matter which SYSGEN method is to be performed, the SYSGEN utility is invoked by entering the following command at the system prompt:

A>**SYSGEN *RETURN***

A display in the following form will appear:

```
SYSGEN VER 2.0.04

SOURCE DRIVE NAME (OR RETURN TO SKIP):
```

The next entry you makes depends on the method used.

2 SYSGEN METHODS

You must consider the situation before running SYSGEN, so that the appropriate SYSGEN method is used.

An operating system must be compatible with the disk it is copied to, and Heath/Zenith supports four types of disk: (5.25-inch hard-sectored, 5.25-inch soft-sectored, 8-inch, and any disk used in the H/Z67 Winchester Disk drive model). Furthermore, Heath/Zenith computers can accommodate systems of different memory capacities. The MOVCPM utilities are used to customize the system for both disk type and memory capacity.

- If a MOVCPM utility was not necessary to customize the operating system for disk type or memory capacity, then the "Disk to Disk System Copying" method (2.1) should be used. (The BSYSGEN utility could also be used in this case.)

- If a MOVCPM utility is used to customize an operating system before transfer, then this system will reside in the computer when SYSGEN is invoked, and the "Computer to Disk System Copying" method (2.2) should be used.

NOTE: When the CP/M Operating System is copied to a disk, it is moved in two parts: the system kernel and the BIOS.SYS file. The "Computer to Disk System Copying" method does not copy the file BIOS.SYS. If this method is used, then the PIP or MAKEBIOS utility must also be used to copy the BIOS.SYS file.

2.1 Disk to Disk System Copying

If the operating system is being copied between two disks of the same type, you can copy both the system kernel and the BIOS.SYS file to the destination disk using this SYSGEN method. (This method is used when a MOVCPM activity does **not** precede the copy activity.)

Under these circumstances, you should answer the "SOURCE DRIVE NAME" prompt by typing the letter for a drive name, as shown:

```
SOURCE DRIVE NAME (OR RETURN TO SKIP): x
```

Where **x** is the letter of the source drive.

SYSGEN will respond with a prompt in the following form:

```
SOURCE ON x, THEN TYPE RETURN
```

You should answer this prompt with a carriage return. SYSGEN will read the system kernel from the source disk, and signal that it has done so with the following message:

```
FUNCTION COMPLETE
```

Then SYSGEN will offer the option of copying BIOS.SYS from the source disk with the prompt:

```
COPY BIOS.SYS (Y/N):
```

If the you wish to copy the file BIOS.SYS from the source disk, then **Y** should be pressed, and SYSGEN will again display the message:

```
FUNCTION COMPLETE
```

If you wish to copy a BIOS.SYS file that is not stored on the source disk, (or a file by another name that contains a BIOS) then **N** should be pressed.

After either entry, SYSGEN will prompt for the drive that contains the destination disk. You should answer the prompt as shown:

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

Where **y** is the letter of a valid, working drive.

Then SYSGEN will display the prompt:

```
DESTINATION ON y, THEN TYPE RETURN
```

You should enter a carriage return at this prompt, to confirm the destination drive choice. SYSGEN will put the system kernel, and in some cases the BIOS.SYS file, onto the destination disk. (The disk in drive "y").

Then SYSGEN will again display the prompt:

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

If you wish to copy the same system components to a different destination disk, then the letter of the drive containing this disk should be entered.

If you do not wish to SYSGEN other disks, then a carriage return should be pressed. The SYSGEN activity will end, and CP/M will display the system prompt.

2.2 Computer to Disk System Copying

If you have just run a MOVCPM utility to customize the operating system for disk type or memory capacity, then this system still resides in the memory of the computer. You can copy this system from the computer to the disk by using this method.

When SYSGEN prompts for "SOURCE DRIVE NAME", you must enter a carriage return—**not** a drive name.

```
SOURCE DRIVE NAME (OR RETURN TO SKIP) RETURN
```

SYSGEN will now prompt you for the drive that contains the destination disk. You should answer the prompt as shown:

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

Where **y** is the letter of the drive containing the disk that is to receive the system kernel. (The BIOS.SYS file can not be copied through this SYSGEN method.)

SYSGEN will request confirmation with the prompt:

```
DESTINATION ON y, THEN TYPE RETURN
```

You should enter a carriage return at this prompt. SYSGEN will put the system kernel onto the destination disk (the disk in drive y).

Then SYSGEN will again display the prompt:

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

If you wish to put the system kernel on another disk, then type the letter of the drive containing the other disk.

If you do not wish to put the system kernel on another disk, and if you have not specified drive A at a previous "DESTINATION DRIVE" prompt, then enter a carriage return at this "DESTINATION DRIVE" prompt. The SYSGEN activity will end and CP/M will display the system prompt.

Under this method, a carriage return should **not** be entered at this prompt.

If you do not wish to SYSGEN other disks, and you did specify drive A at a previous "DESTINATION DRIVE" prompt, then you must reset the computer at this "DESTINATION DRIVE" prompt (rather than entering a carriage return). Resetting the computer at this prompt is necessary because entry of a carriage return at this prompt causes a warm boot, which would cause the new system kernel (recently recorded on the disk in drive A) to be loaded into computer memory. It is undesirable to load any part of this new system kernel into memory on a warm boot because it might have just been changed by a MOVCPM utility. Therefore, it might be of a different size than the system kernel that was loaded into memory at bootup.

When you use SYSGEN by the "Computer to Disk" method, SYSGEN will not copy the BIOS.SYS file. Therefore, in order to make the destination disk bootable, you must copy the BIOS.SYS file to it using the PIP utility or the SUBMIT and MAKEBIOS utilities.

The PIP utility can be used to transfer the BIOS.SYS file, as shown:

A>**PIP y:=x:BIOS.SYS[RV]** ***RETURN***

Where **y** is the destination disk receiving BIOS.SYS;

where **x** is the source disk from which BIOS.SYS is copied; and

where **[RV]** are PIP parameters used to help you copy a file that has system status, and to help you verify the accuracy of the copy operation.

The MAKEBIOS utility can be used to transfer the BIOS.SYS file, as shown:

A>**SUBMIT MAKEBIOS y x** ***RETURN***

Where **y** is the destination disk receiving BIOS.SYS; and

where **x** is the disk containing the files BIOS.ASM, MAKEBIOS.COM, MAKEBIOS.SUB, and PREL.COM.

NOTE: The MAKEBIOS utility customizes the system before transferring it to the destination disk. See the text on MAKEBIOS for details on its use.

3 SYSGEN ERROR MESSAGES

```
INVALID DRIVE NAME
```

EXPLANATION: You must specify a drive letter corresponding to the names of drives that exist in the hardware environment, and are recognized by the operating system that was loaded at bootstrap.

```
NO SOURCE FILE ON DISK
```

EXPLANATION: The drive specified as "SOURCE DRIVE" did not contain the file BIOS.SYS. You should use a different disk in the source drive, or rename a BIOS file that has been given a different name to "BIOS.SYS".

```
SOURCE FILE INCOMPLETE
```

EXPLANATION: SYSGEN failed in an attempt to copy the file BIOS.SYS from the disk in the source drive. This file might have been damaged by disk media flaws or partially overwritten. You should reset, perform bootstrap, and re-enter the SYSGEN command using a different disk in the source drive.

```
WRITE ERROR DURING BIOS.SYS
```

EXPLANATION: You should try SYSGEN again with a destination disk that is write enabled, formatted, and has at least 6 kilobytes of free space.

```
ERROR READING BIOS.SYS
```

EXPLANATION: SYSGEN failed in an attempt to copy the file BIOS.SYS from the disk in the source drive. This file might have been damaged by disk media flaws or partially overwritten. You should reset, perform bootstrap, and re-enter the SYSGEN command using a different disk in the source drive or using a different disk to perform bootstrap.

```
PERMANENT ERROR, TYPE RETURN TO IGNORE
```

EXPLANATION: The system kernel or BIOS.SYS file are either incompatible with the destination disk type or otherwise flawed. You should reset, perform bootstrap, and re-enter the SYSGEN command using a different disk in the source drive or using a different disk to perform bootstrap. Under some circumstances, you must use a MOVCPM utility before SYSGEN.

TYPE

The Resident Command that Displays File Contents on the Console

The TYPE command displays to the console the contents of specified files (1). Fast scrolling displays can be controlled to make them readable (2). TYPE is best used on certain kinds of files (3).

1 DISPLAYING A FILE

To produce a console display of the contents of a disk file, you should respond to the system prompt with a command line in the following form:

A>**TYPE {file name}** ***RETURN***

Where **{file name}** is the complete, explicit name of a file.

If you must examine a file that resides on disk in a non-default drive, this drive must be specified immediately before the file name specification. For example, if the file "LOGDRIVE.TXT" resides on a disk in drive B, its contents will be displayed on the screen when the following command is entered:

A>**TYPE B:LOGDRIVE.TXT** ***RETURN***

2 SCREEN DISPLAY CHARACTERISTICS

The display produced by TYPE will scroll by on the console until the entire contents of the file have been displayed. You can terminate the display during its execution and return to the operating system by pressing any character other than CTRL-S. The display scroll can be halted temporarily by entering ***CTRL-S*** character, and restarted by pressing CTRL-S (or any other character) again.

Some of the special features (boldface type, underlining, etc.) inserted into text files by some word processing systems might not be indicated in screen displays produced by the TYPE command.

3 PRACTICAL USAGE OF TYPE

The most desirable usage of the TYPE command is to display files composed with ASCII characters (printable letters and numbers). If the TYPE command is issued for a file with a ".COM" extension (or for any file stored on the disk in binary form) the resulting screen display will contain a meaningless series of characters.

To obtain a paper printout of the file, enter a ***CTRL-P*** before entering the TYPE command line. Printouts will not exhibit special features (boldface type, underlining, etc.) that are inserted into some text files by word processing programs.

USER

The Resident Command that Controls User Access to File Areas

The USER command enables a user to access only files in a specified area of the disk directory (1). You can log into USER areas by invoking a single USER command line (2). You must implement a sequence of commands to put files into a non-zero USER area (3).

1 USER INVOCATION

File directories supported by the CP/M system are divided into 16 user areas. When a user is logged into a particular USER area, then only the files within that area are accessible (except by implementing the PIP utility with the "G" parameter).

The USER command is useful when several users have files stored on the same disk. File directory (DIR) and status checks (STAT) invoked by a particular user will list only the files in the user's specified area.

Whenever a cold boot is performed, you are automatically logged in to USER area number 0. When in this area, only files in USER area 0 of the directory are accessible.

2 INVOKING USER

To make a user area other than number 0 accessible, you should respond to the system prompt by entering a command line in the form:

A>**USER {area number}** ***RETURN***

Where**{area number}** indicates the number, from 0 to 15, of an area that contains files that you wish to access.

The USER command can be issued whenever the system prompt appears. The currently-logged USER area will be in effect for all of the disks in the hardware environment, until a different USER area is logged or a cold boot is performed.

3 PULLING FILES INTO USER AREAS

Files can be pulled into a USER area that contains the file PIP.COM, but to put the file PIP.COM into that USER area in the first place requires a special process.

To put the file PIP.COM into a USER area (other than area zero), you should follow the following steps, in sequence:

3.1 Insert disks that have the PIP.COM utility and at least 8 kilobytes of space available.

3.2 Log USER area zero (or whatever USER area contains PIP.COM) by entering the command **USER 0** ***RETURN***.

3.3 Invoke the PIP utility by entering the command **PIP** ***RETURN***. (Specify the name of the drive that contains PIP.COM if it is not the default drive.)

3.4 Press ***RETURN*** at the asterisk (*) prompt.

3.5 Log the desired USER area by entering the command **USER n** ***RETURN***, where **n** is the number (0-15) of the USER area you wish to establish.

3.6 Save the portion of the computer memory that now contains the file PIP.COM by entering the command **SAVE 29 PIP.COM** ***RETURN***. (Specify the name of the drive on which you wish to establish the USER area if it is not the default drive.)

These six steps will place the file "PIP.COM" into the desired USER area (n). Once PIP.COM resides within a USER area, it can help you to pull other files into that USER area. The following example display demonstrates how this procedure might appear on the console:

(3.2) A>**USER 0** ***RETURN***
(3.3) A>**PIP** ***RETURN***
(3.4) A>***** ***RETURN***
(3.5) A>**USER 5** ***RETURN***
(3.6) A>**SAVE 29 PIP.COM** ***RETURN***

If you have followed the preceding example procedure, file copies could now be pulled into USER area 5 by the entry of commands in the form:

A>**PIP B:=**{**drive**}:{**file**}[G{**source area**}] ***RETURN***

Where {**drive**} is the name of the drive from which file copies are being pulled;

where {**file**} is the name of the file being pulled into the new USER area; and

where{**source area**} is the number of the USER area from which files are being pulled.

XSUB

The Utility that Batches Commands Within Utility Programs for Automatic Processing

The XSUB utility extends the power of the SUBMIT utility to enable you to invoke utility programs, and then execute commands within these programs, while making only one entry at the terminal.

1 XSUB OPERATION

The XSUB command line is entered as the first line in a SUB file. A SUB file is a text file composed of command lines that are executed in sequence when a SUBMIT command line is entered at the system prompt.

When the XSUB command is executed, it moves beneath the component of the operating system known as the Console Command Processor (CCP). All command lines after the XSUB command line are processed by XSUB, so that programs which usually prompt you to make entries on the keyboard will accept entries directly from the SUB file.

The XSUB program remains in memory after execution, and displays the message "(xsub active)" each time a warm boot is performed during the execution of the $$$.SUB file. Thus the initial XSUB command in a SUB file can remain in effect throughout the execution of every command in the SUB file.

XSUB will display the message "XSUB already present" as you enter the command **XSUB** at the system prompt (through the console). However, this remnant of the XSUB program will not go into effect when subsequent SUB files are submitted. Therefore, you must make "XSUB" the first line of any SUB file in which XSUB execution is desired.

NOTE: XSUB will not support the submission of input for programs which, when run, prompt for single character input. Hence, the only programs that can effectively be used in a SUB file under XSUB are programs that accept input lines ended with a carriage return, such as DDT, PIP, LIST, and ED.

2 XSUB EXAMPLE

The file "MEGAJOB.SUB" contains the following command lines:

```
XSUB
DDT
I$1.HEX
R
G0
SAVE 4 $2.COM
```

Processing of this SUB file could be initiated by entering the following command line:

A>**SUBMIT MEGAJOB PRGRMY PRGRMZ *RETURN***

The preceding entry creates a $$$.SUB file in which the primary file name "PRGRMY" is substituted for the prototype name "$1" of the SUB file, and "PRGRMZ" is substituted for the "$2" prototype parameter.

As commands are read from the $$$.SUB file, the XSUB utility enters computer memory. Then DDT enters and executes the DDT commands "IPRGRMY.HEX", "R", and "G0", which are also included in the $$$.SUB file. The DDT command "G0" has the same effect as a warm boot, which enables the operating system to process the final command in the $$$.SUB file, "SAVE 4 PRGRMZ.COM".

Appendix A:

Operating System Error Messages

This appendix explains error messages produced by the CP/M Operating System.

CONVENTIONAL ERROR MESSAGES

BAD LOAD

Causes: You tried to run 3) a program that requires 1) more memory than CP/M has available or 2) more memory than your computer has available.

Remedies: 1) Use a MOVCPM utility and the SYSGEN utility in sequence to expand CP/M to the memory limit of your computer. 2) Add more memory to your computer. 3) Make your program smaller.

```
Bdos Err On x: Bad Sector
```

Causes: (Where "x:" is the name of a drive which CP/M tried and failed to read data from, or write data to, a disk.) 1) Disk used for a data transfer operation is damaged or extremely worn. 2) Disk drive controller is malfunctioning. 3) Disk being accessed was formatted by a disk drive controller that is not IBM compatible. 4) Disk being accessed has not been formatted at all. 5) Drive being accessed contains no disk. 6) Accessed disk is write protected.

Remedies: Enter a ***CTRL-C*** to abort the activity and perform a warm boot, or enter a carriage return to skip over the location of the error condition, or press **R** to retry the activity at the location of the error condition. If the activity is impossible at this location, then 1) inspect or replace disk. 2) inspect or replace controller card. 3) & 4) FORMAT disks in your own hardware environment. 5) Access a drive with a disk. 6) Write enable disk and perform warm boot before trying to write data to disk.

```
Bdos Err On x: Select
```

Causes: (Where "x:" is the name of a drive which you referred to in a command, but which CP/M cannot access or does not recognize.) 1) You entered a drive name that does not correspond to a drive in your hardware. 2) You tried to access a Winchester partition that was not first assigned a drive name.

Remedies: In all cases, pressing any keyboard character after such an error message will cause a warm boot. 1) Enter drive names that correspond to drives in your hardware. 2) Use ASSIGN utility on a partition before trying to access the partition.

```
Bdos Err On x: R/O
```

Causes: (Where "x:" is the name of a drive at which CP/M refuses to read or write data. 1) You tried to write to a file that had been given "Read/Only" status by the STAT utility. 2) You tried to write to a disk that had been given temporary "Read/Only" status by the STAT utility. 3) You tried to switch disks in a drive and then write to the new disk without performing a warm boot.

Remedies: In all cases, pressing any keyboard character after such an error message will cause a warm boot. 1) Change file status to "Read/ Write" using the STAT utility. 2) Perform a warm boot in any fashion (such as the ***CTRL-C*** entry) before trying to write data to the disk. 3) Perform a warm boot (with a ***CTRL-C***) after switching disks, but before accessing a switched disk.

NO FILE

Causes: 1) You invoked DIR Resident Command for a directory of disk files, and the file(s) specified not on the disk in the logged user area. 2) You invoked TYPE Resident Command to display contents of a file that was not on the disk in the logged user area. 3) You invoked ERA Resident Command to erase a file that was not on the disk in the logged user area. 4) You invoked REN Resident Command to try to rename a file that was not on the disk in the logged user area.

Remedies: 1) Unless such a message gives you adequate information, log to a different user area with the USER Resident Command before trying DIR command. 2), 3), & 4) Check disk directory using DIR Resident Command or STAT utility to obtain correct file name.

FILE EXISTS

Cause: You tried to rename a file (with REN Resident Command) using a name that another file on the same disk already has.

Remedy: Give a different name to the file you are renaming, or move the file that already has the name to a different disk or user area.

NO SPACE

Cause: You invoked SAVE Resident Command to store memory contents on a disk that didn't have enough room for the data from memory.

Remedy: Re-enter SAVE command specifying the name of a drive containing a disk that has sufficient room for the file, or specifying fewer pages of data.

```
SYNCHRONIZATION ERROR
```

Causes: You tried to invoke a copy of the SYSGEN utility that did not bear the same serial number identification as the Operating System.

Remedies: Only use the MOVCPM or SYSGEN utilities when the utility and the system (or backup copies of both) originally came from the same Distribution Disk.

```
CAN'T OPEN BIOS.SYS
```

Cause: You tried to perform bootstrap with a disk that does not contain an acceptable BIOS.SYS file.

Remedy: You should copy a properly customized BIOS.SYS file to the disk, or rename an already resident BIOS file to the name "BIOS.SYS".

```
ERROR DURING WARM BOOT - PRESS ANY KEY
```

Cause: You tried to perform a warm boot or exit from a utility after changing the position of the disk that originally resided in drive A.

Remedy: You should replace the disk that was originally in drive A: or the current default drive when bootstrap was performed. Then you must perform a warm boot by entering a ***CTRL-C***.

DETAILED DISK ERROR MESSAGES

Detailed disk error messages are additional messages that are displayed with conventional system error messages to provide more specific information about the cause of the error condition. These detailed messages will occur if you turn them "on" through the "Set Disk Parameters" submenu of the CONFIGUR utility.

If the operating system has difficulty in reading from or writing to a disk, it tries the operation again and again, up to ten times. If the disk still can't be read or written at this location, then the system displays a "Bdos Err" error message. (When a 5.25-inch disk drive has such difficulty, it records a "soft error" for each of the ten attempts, incrementing the SECNTnn variable each time.)

Detailed Disk Error Messages are displayed in the following form:

```
Hdd READ ERROR nn

Bdos Err On x: Bad Sector
```

Where "dd" is the drive type that obtained the error;

where "READ" or "WRITE" is the activity that the operating system was trying to perform;

where "nn" is the detailed error code; and

where "x" is the drive name of the drive at which the error was obtained.

Most of the specific detailed error codes are listed in the hardware manuals for the H/Z-37, H/Z-47, and H/Z-67 drive models. This text contains explanations of the detailed codes that occur with the H/Z-17 drive model, and some of those that occur with the H/Z-67 drive model.

H/Z-67 DETAILED ERROR MESSAGES (PARTIAL LISTING)

80 Hardware interface error

81 Hardware interface error

82 Hardware interface error

83 Attempt made to exceed partition boundaries

H/Z-17 DETAILED ERROR MESSAGES

01 Bad Track Error: Probably a problem in the floppy disk mechanism, often caused by setting the disk step rate too fast during a run of CONFIGUR (submenu B).

02 Head Sync Error: An error has occurred in the format of the floppy disk.

04 Header CRC Error: An error in reading the sector identification header, generally caused by defective media.

08 Data CRC Error: An error in reading the data on the sector, can be caused by power failure, power surge, defective media, or defective hardware.

10 Record Not Found: The specified track and sector cannot be located, due to poor alignment, flawed disk media, incorrect disk rotation speed, or the specification of a nonexistent track or sector.

20 Missing Data Sync: An error has occurred in the format of the floppy disk.

40 Write Protect Error: An attempt was made to write to a mechanically write-protected disk.

80 Unit Not Ready: The drive unit was not ready to read or write data. This can be caused by not having a disk in the drive or by leaving the drive open.

Appendix B:

Recommended Hardware for Heath/Zenith CP/M

WITH H/Z-89 COMPUTER

The minimum supportable system that will run under this implementation of CP/M is:

- An H/Z-89 computer system with at least 32K of random access read/write memory, with the Z89-7 or H88-7 ROM modification kit installed (already done on Z89-FA and later models), and with the MTR90 ROM. (If only H/Z-17 and H/Z-47 drives are used, MTR89 ROM will work.)

- One connected, functional disk drive (the H/Z-89 comes with a single drive with a H/Z-17 controller).

This implementation of CP/M for the H/Z-89 will support the following pieces of hardware:

THE COMPUTER:

- An H/Z-89 computer system with 32K or more (64K max) of random access read/write memory, with the Z-89-7 or H-88-7 ROM modification kit installed (already done on Z89-FA and later models), and with the MTR90 ROM. (If only H/Z-17 and H/Z-47 drives are used, MTR89 ROM will work.)

DISK DRIVES:

- From one to three hard-sectored 5.25-inch floppy disk drives (H/Z-17 controller).
- From one to three soft-sectored 5.25-inch floppy disk drives (Z-37 controller).
- Two 8-inch floppy disk drives (H/Z-47 controller).
- One Winchester Disk System—Winchester Disk and 8-inch Floppy Disk (Z-67 controller).

With a maximum of six drives connected and functional. (Note that only two types of drives may be connected at any given time.)

PRINTERS:

- A diablo 630, 1610, 1620, 1640, or 1650 (compatible) printer set to 1200 baud, and an RS-232C cable connected to the socket labeled DCE 340-347. The UL1: logical device driver should be connected to this port. It uses an ETX/ACK handshake protocol to control the printer output.
- An H14, WH14, WH24 or H/Z-25 dot-matrix printer, utilizing the RTS modem control for handshaking, connected to the socket labeled DCE 340-347.

If your printer requires a handshaking convention that is not discussed in the text on the CONFIGUR utility, it will be your responsibility to modify the BIOS physical device driver. The Heath/Zenith Software Group can not assume the responsibility for any such modifications.

- A WH36, or any printer that does not require "busy" handshaking, may be connected to the TTY: logical device, again it is plugged into the socket labeled DCE 340-347. In the Configuration program, specify the port number for the TTY: as being 340Q and move the LST: device to 320Q. (H/Z-89 computers are not currently supplied with port 320Q populated, it is made up of the empty set of sockets on the serial interface card.)
- A WH13 modem connected to the plug labeled DTE 330-337.

WITH H8 COMPUTER

The minimum supportable system that will run under this implementation of CP/M is:

- An H8 computer system with at least 32K of random access read/write memory and with the HA-8-8 Extended Configuration kit installed.

- A terminal connected to the computer with a serial interface at port 350Q (0E8H) on an H8-4 card or port 372Q (0FAH) on an H8-5 card.

- One connected, functional floppy disk drive.

This implementation of CP/M for the H8 will support the following pieces of hardware:

- An H8 computer system with 32K or more (64K max) of random access read/write memory and with the HA-8-8 Extended Configuration kit installed.

- A terminal connected to the computer with a serial interface at port 350Q (0E8H) on an H8-4 card or port 372Q (0FAH) on an H8-5 card.

- From one to three hard-sectored 5.25-inch floppy disk drives (H17 controller).

- Two 8-inch floppy disk drives (H/Z-47 controller)

With a maximum of six drives connected and functional. (Note that only two types of drives may be connected at any given time.)

PRINTERS:

- A Diablo 630, 1610, 1620, 1640, or 1650 (compatible) printer set to 1200 baud, and the RS-232 cable connected to port 340-347 on an H8-4 card. The UL1: logical device driver should be connected to this port. It uses ETX/ACK handshake protocol to control printer output.

- An H14, WH14, WH24 or H/Z-25 dot-matrix printer, utilizing the RTS modem control for handshaking, connected to port 340-347 on an H8-4 card.

If your printer requires a handshaking convention that is not discussed in the text on the CONFIGUR utility (page 2-70), it will be your responsibility to modify the BIOS physical device driver. The Heath/Zenith Software Group can not assume the responsibility for any such modifications.

- A WH36, or any printer that does not require "busy" handshaking, can be connected to the TTY: logical device, again it is plugged into port 340-347 on an 8-4 card. In the Configuration program, specify the number for the TTY: as being 340Q and move the LST: to 320Q.

- A WH13 modem connected to port 330-337 on an H8-4 card.

Appendix C:

Bootstrap

The "Startup Procedures" in "Volume I: CP/M Introductory Guide" provide detailed instructions on performing bootstrap. However, some microcomputers allow you to vary startup procedures by entering alternative bootstrap commands, or by causing the microcomputer to perform bootstrap automatically. This text explains the startup variations that are possible with some Heath/Zenith microcomputers.

ALTERNATIVE BOOTSTRAP COMMANDS

This text section shows you how to use different bootstrap commands to boot up from different drives. But it is first necessary that you understand some of the technical terms used to define the relationship between the microcomputer and disk drives. This text will introduce some new terms used to refer to disk drives. But these terms will apply to your drives only until you've completed bootstrap. After a successful bootstrap, the drive containing your bootable disk automatically becomes drive "A:", and the other disks will assume drive names such as "B:", "C:", "D", "E", or "F".

Your microcomputer contains circuit boards that control your disk drives. Each of these circuit boards, called "controller cards", can control two or three disk drive slots of the same type. For instance, the H/Z-47 controller card controls two 8-inch drives in the H/Z-47 drive model.

All of the drive slots controlled by a single controller card are referred to collectively as one "device". If you have only one kind of disk drive, then your microcomputer contains only one controller card, and the drives controlled by this card are collectively called the "primary device". If you have two kinds of disk drive, then your microcomputer contains two controller cards, and the drives controlled by each card are called either the "primary device" or the "secondary device".

COMMANDS FOR THE H/Z-88, H/Z-89, AND H/Z-90

With an H/Z-88, H/Z-89, or H/Z-90 microcomputer you can boot up with a disk in any of your drive slots because this microcomputer refers to each drive slot by a unique and constant name.

Since each device consists of two or three drives, the individual drive slots within each device are given "unit" numbers 0, 1, or 2. Thus each physical drive slot is identified by a device type ("primary" or "secondary") and a unit number ("0", "1", or "2").

Table C-1 shows all of the bootstrap commands possible within your hardware environment, and the drive that will be accessed by each. (Drives are identified in this table by their "device" and "unit" designations.)

BOOTSTRAP COMMAND ENTRY	VIDEO CONSOLE DISPLAY	DEVICE TYPE ACCESSED	UNIT NUMBER ACCESSED
B (or **B0**)	H:Boot (or H:Boot0)	primary	0
B1	H:Boot1	primary	1
B2	H:Boot2	primary	2
BS (or **BS0**)	H:Boot SD (or H:Boot SD0)	secondary	0
BS1	H:Boot SD1	secondary	1
BS2	H:Boot SD2	secondary	2

Table C-1:
Alternative Bootstrap Commands for the H/Z-88, H/Z-89, and H/Z-90

To perform bootstrap with a bootable disk in a particular drive slot, you type the boostrap command entry and a carriage return. Then the microcomputer will access the drive slot identified by the bootstrap command to copy the CP/M Operating System from the bootable disk.

NOTE: The actual drive slot that is accessed by each of these bootstrap commands depends upon the assortment of disk drives you have and the settings of the switches inside your drives and microcomputer.

BOOTSTRAP COMMAND ENTRY	H8 LED DISPLAY	DEVICE TYPE ACCESSED	DRIVE SLOT ACCESSED
1	Pri H17	primary	slot in H/Z-89; or left-hand slot of H/Z-87, H-77, or H-17-3
	Pri H47	primary	left-hand slot of H/Z-47 (8-inch drives)
2	Sec H17	secondary	slot in H/Z-89; or left-hand slot of H/Z-87, H-77, or H-17-3
	Sec H47	secondary	left-hand slot of H/Z-47 (8-inch drives)

Table C-2
Alternative Bootstrap Commands for H-8

ENABLING THE AUTOMATIC BOOT

FOR THE H/Z-89

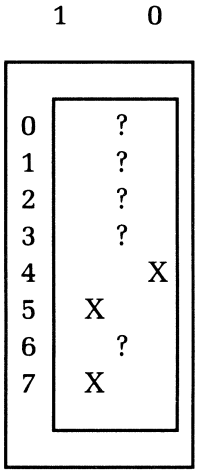
The H/Z-89 has the capability to automatically boot from the primary device. The feature makes it possible to simply turn on (or reset) the machine and have the CP/M system automatically come up.

CAUTION: Never have the disks inserted with the drive doors closed when the power is turned on or off. Current surges that occur when power is initially turned on (and when a unit is turned off) could magnetically damage the disk as well as any data that is stored there.

To enable the auto boot, you will have to make a minor change to the hardware. First, be certain that the computer and all of its peripheral devices are turned off and unplugged from the wall outlets. Carefully release the top of the computer cabinet by inserting a small screwdriver blade into the slot in each side of the unit. Slide the latch plates toward either the front or the back of the screen and carefully tilt the cabinet shell back.

The auto boot feature is enabled by changing one DIP switch setting on the CPU board. The DIP switch is light blue, and is located in the lower right quadrant of the CPU board. (The switch SW-501, is immediately to the right of the resistors RP508 and RP509.) Change only the position of the lowest switch, which is labeled "7", by sliding the small white switch to the left. You can do this by using the tip of a pen or a small screwdriver. Refer to Figure C-1 for a graphic representation of the proper position of the "7" switch. (Some of the other number switches are marked with a question mark because their positions are different depending on the user's combination of hardware.)

After setting the "7" switch, close the cover of the machine, by reversing the steps you used to open the cover. Plug the units in and turn on the power. Place a bootable disk into the drive, close the door, and the system goes directly into CP/M without the user performing the reset and bootstrap steps.



Device Type (port 174Q)
Device Type (port 174Q)
Device Type (port 174Q)
Device Type (port 170Q)
1 = Secondary (174Q) 0 = Primary(170Q)
1 = No Memory Test 0 = Memory Test
Baud Rate
1 = Auto Boot 0 = Not Auto Boot

Figure C-1:
The SW-501 Switch,
Set for Automatic Bootstrap
To the Primary Device

FOR THE H8

The Extended Configuration Option board (HA8-8) in the H8 has a single set of eight switches labeled 0-7. To enable the automatic boot function, change the position of switch 7.

Precautionary Notes:

Remember that if your hardware configuration includes a Z87, make sure that it is always powered up whenever you use the computer whether it is in immediate use or not.

Disks may get damaged if the disks are placed in the drives with the doors closed prior to powering up or down. Before using your system either wait for CP/M to display the system prompt before you insert your disks, or leave the drive doors open until power is applied. Conversely, prior to turning the power off, either remove all disks from your drives or open all the drive doors.

Appendix D

Basic Input/Output System (BIOS) Organization

The Heath/Zenith 2.2.04 release of CP/M includes several features to enhance the flexibility of the original CP/M system. As new hardware devices are introduced, they will be supported under this operating system. As a result, the BIOS (Basic Input Output System) occupies more than the amount of space provided for that purpose by Digital Research.

Therefore, Heath/Zenith has designed a program to fit where the BIOS is normally recorded on the system tracks. This program is capable of using the CP/M file system to open a disk file and read in the complete BIOS. It is referred to as the BIOS Loader, and contains just enough intelligence to read from the disk and do some simple terminal I/O in case of an error.

When the system is cold booted, the monitor ROM reads the first sectors of the disk in the selected drive into memory and then control is transferred to that program. The cold start loader contained in the first half of those sectors reads the contents of the first tracks into memory at the appropriate locations for the configured memory size. When that is complete, control is passed to the cold boot entry point of the newly loaded system.

The following BIOS features are explained in this appendix:

- The Heath/Zenith BIOS Loader (BLDR)
- Basic Addresses within the Heath/Zenith BIOS
- BIOS Options
- Algorithm for Disk Deblocking
- Disk Formats
- Page Zero and Clock Organization

THE HEATH/ZENITH BIOS LOADER (BLDR)

With the Heath/Zenith CP/M, instead of being the standard BIOS that is opened, it is the BIOS loader (BLDR) that gets opened, which in turn opens a file called BIOS.SYS and reads it into the appropriate locations in memory.

By using the BLDR concept, the Heath/Zenith implementation overcomes the size limitation imposed by the Digital Research design and at the same time simplifies the otherwise error-prone process of integrating one's own modified BIOS into the system. The penalties paid are slight—the disk from which you cold boot must contain the BIOS.SYS file, and it takes somewhat longer to cold boot because the second file must be opened, read, and relocated in memory.

Note that the BIOS.SYS file is only referenced during the cold boot process, so it only needs to be on the disk used to cold boot your system. The warm boot function of CP/M does not reread the BIOS, and therefore this file does not necessarily have to appear on disks that are used during a warm boot.

The BIOS.SYS file contains the BIOS in a relocatable file format that is compatible with the page relocatable format used in Digital Research's MP/M operating system. This relocatable format is reasonably compact, and storing the BIOS in relocatable form provides the advantage of not requiring relocation of the BIOS during MOVCPMnn.

BASIC ADDRESSES IN THE HEATH/ZENITH BIOS

The Heath/Zenith BIOS implementation is structured so that system utilities can readily access its device tables. This is primarily a requirement of CONFIGUR, which alters some of the values which are used at run-time. Many of these values would be assembly-time constants in the normal implementation of CP/M.

A user program can find the BIOS by using the address at location 0001H as a pointer (the operand of the jmp wboot instruction). That value will be 3 greater than the address of the start of the BIOS.

The first part of the BIOS is the standard Digital Research entry point "jump" table. This table includes jump instructions to each of the 17 entry points into the BIOS.

The "listst" entry point for checking the output status of the list device is implemented in this release. This entry point is used by some background printing spoolers.

Immediately following the standard entries is a one byte BIOS version number (in binary). It is important that programs which alter values in the BIOS in a direct way know exactly which version they are dealing with. For example, CONFIGUR must patch tables at the beginning of the BIOS and therefore needs to be reasonably certain that those tables are where it expects to find them.

Upon cold boot, CP/M will display a sign-on message like the following:

```
nnK HEATH/ZENITH CP/M 2.2.04 09/15/82
FOR Hdd DISKS WITH OPTION(S) cccc
```

Where "nnK" is the amount of Random Access Memory space that the currently loaded CP/M Operating System will occupy; and
where "2.2.04" is the version number.

The next byte is the mask used for the serial printer ready status check.

The next byte is used as the first mode byte for the BIOS. Six of the bits are assigned; the remainder are reserved. Bit 7 causes the CCP to attempt to execute the automatic command line on the default disk every time the disk is cold booted. Bit 6 performs the same function for each warm boot. Bit 3 contains the serial printer ready flag, 0 = low and 1 = high. Bit 2 supports the partitioning of the Z-67 Winchester Disk. Bit 1 enables the printing of disk error messages by the BIOS so the user will receive extended error-status messages. Bit 0 is set when the BIOS is configured for use with a H8-5 serial card for the console port.

The next byte is used as the second mode byte for the BIOS. Bit 2 contains the Parallel Printer Ready flag, 0 = low and 1 = high. Bit 1 indicates which printer to use as the LPT: device if a Z89-11 interface card is installed, 0 = serial and 1 = parallel. Bit 0 is set when at cold boot it is determined that a Z89-11 interface card is installed.

The next several bytes are used as character I/O device control structures. These specify the port number, baud rate, number of nulls required after a carriage return, and upper case mapping flag for each of the serial character I/O devices. The data is arranged as port number, followed by the baud rate divisor word.

The high order nibble of the baud rate divisor forces all output to upper case if the MSB is 1, and the least significant 3 bits of that nibble are the number of nulls sent after a carriage return to that device. There is one of these structures for each device.

Then two words (four bytes) are used as a soft error counter for the 5.25-inch disk drives. These words are incremented after each soft (recoverable) error. By using DDT to examine this value you can get an indication of the performance of your disk system.

The next byte is set for the maximum number of disks that this BIOS may be configured for. This allows programs to determine the length of the disk tables and to access only those tables that really exist.

This is followed by the disk parameter base and the disk parameter entries for each of the disk drives. In addition to the 16 bytes per disk assigned by Digital Research, the Heath/Zenith implementation has added 8 bytes to each entry.

The first of these 8 is a multi-purpose byte that is used to identify the drive type and its port address. The three most significant bits indicate the drive type. The second byte is the select code necessary for this drive. The next byte is the number of 128 byte records per physical sector. It is followed by a byte indicating the number of 128 byte records per allocation unit. The next four bytes are used for all drive controllers except the H/Z-47.

For the 5.25-inch disk drives the first of these indicates the current track position of the head for this drive. The next byte controls the step rate. When set, the most significant bit of that byte indicates that the drive will be up to speed in 250 mS instead of the normal 1000 mS. The next byte contains flags. The last byte is reserved for use during the physical to logical drive mapping process.

For the Z-67, these bytes are used a bit differently. The first word (two bytes) is the logical sector number of track 0 in a partition. The last word is the logical sector number used in upper bound limit checks.

In addition to these tables, it is necessary for some system utilities to have access to some of the values used for disk control. These values have been located in BIOS and in page 0 of memory. These values control the motor and select timeouts as well as the head settle and write-gate step delays.

At 000DH the current value of the H/Z-89 control latch at port 362Q is stored. This port controls the presence of RAM at zero and enables/resets the clock. In the H8, the clock enable and RAM at zero are split between two different ports.

The H/Z-89 contains special hardware that traps input and output instructions to some of the H8 control ports and generates a nonmaskable interrupt (NMI). Since the Z80 NMI vector falls within the RAM allocated for the default file control block (FCB), it is not possible to guard it with a return instruction or to install a jump to a user routine. Therefore, care must be taken that no inputs or outputs to the H8 control ports (360Q, 361Q, 372Q, 373Q) are attempted in the H/Z-89. Location 000EH is used for the value required to reset the clock on the H8, or 0 if on an H/Z-89. Location 000FH is reserved for the contents of the H17/H77/Z87 control port. The 16 bytes located at 0040-004FH are reserved.

An effort will be made to keep these values in their same relative positions. However, it is possible that between this release and future releases of CP/M from Heath/Zenith there will be some redefinition of the BIOS structure. There is a diminishing probability of change with each release thereafter. Software that accesses these tables should be written using an easily redefinable template to simplify adaptation to such changes.

BIOS OPTIONS

When CP/M is cold booted, it displays a message in the following form:

```
nnK HEATH/ZENITH CP/M 2.2.04 09/15/82
FOR {type} DISKS WITH OPTION(S) {code}
```

The "code" will show the options that are currently selected active within the BIOS.

The options are as follows:

P	The Z-67 is Partitioned
P2	BIOS supports two Z-67 Partitions
I	CRT Input is Interrupt Driven
B	CRT "BREAK" key's Interrupt is available (causes an immediate warm boot)
E3	BIOS supports Extended Double Density on Z-37 Disk Drive
E4	BIOS supports Extended Double Density on H/Z-47 Disk Drive
T	Time of Day Clock
E	Event Counter

ALGORITHM FOR DISK DEBLOCKING

The Heath/Zenith BIOS uses a deblocking algorithm which transparently converts the sectors stored on the 5.25-inch, the 8-inch disks, and on the Winchester Disk to the 128 byte logical records used by CP/M. The BIOS uses information provided by the BDOS to minimize the number of physical read and write operations.

When accessing the disk, the deblocker keeps track of the logical records that are currently buffered within the BIOS. If the BIOS is processing a read call, it is possible for it to retrieve the information from its buffers without any disk activity if the proper sector is already in memory as a by-product of a previous operation.

The buffering that occurs during the write operation means that records can be buffered in memory until the next disk operation. The buffer is only written to the disk if the calling program accesses a logical record that is not contained in the current physical sector, the buffer overflows, or the write is flagged as a "directory access."

One important ramification of this "write-behind" is that programs that do direct BIOS calls to perform track/sector disk I/O should make certain that the write buffer is emptied before the disk is removed or changed, or a cold or warm boot is performed. This can be done by specifying that the last write operation should be performed immediately without write-behind as the BDOS does when it accesses the disk directory.

DISK FORMATS

5.25-INCH HARD-SECTOR DISK FORMAT

The logical records are numbered from 1 to 20 on each track, and are stored in physical sectors 0 thru 9. The tracks of the disk are numbered 0 to 39. The CP/M system reserves the first three tracks for the Cold Start Loader, the CCP, the BDOS, and the BIOS Loader. The CP/M directory begins on track 3, and the usable space begins immediately following it.

8-INCH AND 5.25-INCH SOFT-SECTOR DISK FORMATS

The Heath/Zenith implementation of the BIOS transparently supports a mixture of single, double, and extended double density disks. In addition, each of the densities can be recorded on either single or double-sided media. The first time a disk is "logged on" after a disk reset function (such as a warm boot), the density and whether the disk is double or single sided is checked and used to dynamically modify the disk parameter tables for the particular drive. The most important ramification of this is that disks of different densities or number of recordable sides should be logged on after being interchanged.

The Heath/Zenith configuration offers the user three densities available on the 8-inch disk. Single density is comprised of 26 sectors per track with 128 bytes per sector. Double density contains 26 sectors per track with 256 bytes per sector. Extended density uses 8 sectors per track with 1024 bytes per sector. Deblocking is used on double and extended densities to make each track appear to have 52 and 64 sectors respectively.

The 5.25-inch soft-sectored disks have two possible densities. Single density offers 10 sectors per track with 256 bytes per sector. Double density has 16 sectors of 256 bytes each per track.

PAGE ZERO AND CLOCK ORGANIZATION

```
0000H           jmp     BIOS+3          ;wboot entry use value
                                        ; at 0001H as a pointer
                                        ; to the BIOS to access
                                        ; jump vectors and tables
0003H           db      IOBYTE          ;current IOBYTE
0004H           db      DEFAULT$DISK    ;current default disk
0005H           jmp     BDOS+6          ;BDOS entry point use value
                                        ; at 0006H to find the top
                                        ; of user memory (subtract
                                        ; CCP size if it is to be
                                        ; preserved)
0008H           jmp     clock           ;clock interrupt service
                                        ; vector
                dw      TICCNT          ;1 work counter incremented
                                        ; every 2 mS interrupts are
                                        ; enabled and clock is on
000DH           db      H88CTL          ;current contents of H88/Z89
                                        ; control latch (362Q)
000EH           db      H8CTL           ;current contents of H8
                                        ; control latch (360Q), or
                                        ; zero if on H/Z-89
000FH           db      DEVCTL          ;current contents of H17/H77
                                        ; Z87 control latch (177Q)
0018H           jmp     CRTINT          ;CRT input interrupt service
                                        ; vector
0020H           jmp     H37INT          ;H37 interrupt service vector
0023H   dlymo37 db      0               ;H37 motor turn-off delay
                                        ; counter
0024H   dlyh37  db      0               ;H37 deselect delay counter
0025H   h37ctl  db      0               ;H37 control register image
0026H   h37iret dw      0               ;where to go after H37
                                        ; interrupt
                                        ;
;using the value at location 0009H as a pointer to the clock routine,
; the disk (H17/H77/Z87) timers can be accessed, they occupy the three
;bytes immediately preceding the clock interrupt
;
        dlymo:  db      0               ;motor timer decremented
                                        ; every 512 mS until 0
        dlyh:   db      0               ;head load timer decremented
                                        ; every 512 mS until 0
        dlyw:   db      0               ;head settle timer
                                        ; decremented every 4 mS
                                        ; until 0
        clock:  ...                     ;clock interrupt service
                ...                     ; routine, located in BIOS
                ei
                ret
```

Index

TECHNICAL INFORMATION EXPLANATION

The technical information portion of your CP/M documentation package is intended for users who wish to modify the CP/M Operating System or examine the internal components of the system.

Knowledge of this information is not necessary for most CP/M users who only wish to use CP/M resident commands and to run utilities and application programs with the CP/M system.

This technical information consists of the following documentation items bound within three booklets:

- CP/M 2 System Interface: Chapter 5
- CP/M 2 Alteration: Chapter 6
- Appendix A: The MDS Basic I/O System (BIOS)
- Appendix B: A Skeletal CBIOS
- Appendix C: A Skeletal GETSYS/PUTSYS Program
- Appendix D: The MDS-800 Cold Start Loader for CP/M 2
- Appendix E: A Skeletal Cold Start Loader
- Appendix F: CP/M Disk Definition Library
- Appendix G: Blocking and Deblocking Algorithms.

This technical information was written by Digital Research Corporation, the original producers of CP/M version 2. The source listings in the appendices are for illustration purposes only and do not correspond to the modules actually supplied with your system.

The CP/M software products you have purchased are modifications of CP/M version 2. In these modifications, the Basic Input/Output System (BIOS) has been customized for your hardware by Heath and Zenith Data Systems.