

Section 3

HEATH TEXT EDITOR

TED-8



TABLE OF CONTENTS

INTRODUCTION	3-3
EDITOR MODES OF OPERATION	3-4
The Command Mode	3-4
The Text Mode	3-4
THE COMMAND STRUCTURE	3-5
Range Expressions	3-5
The Verb	3-10
The Option Field	3-11
The Qualifier String	3-12
THE PARAMETER FIELD	3-12
THE COMMANDS	3-12
TERMINAL CONTROL CHARACTERS	3-25
Input Control Characters	3-25
Output Control Characters	3-26
Tape Errors	3-26
COMMAND COMPLETION	3-27
APPENDIX A	3-28
Loading From the Software Distribution Tape	3-28
Loading From a Configured Tape	3-29
APPENDIX B	3-30
Command Structure	3-30
Range Expression Forms	3-30
Line Expression Forms	3-30
Verb (Command) Forms	3-31
Option	3-33
Qualifier String	3-34
Parameter Field	3-34
Terminal Control Characters	3-34
APPENDIX C	3-35
INDEX	3-37

INTRODUCTION

The Heath H8 Text Editor (TED-8) converts your H8 computer and terminal into a very sophisticated typewriter. This typewriter is not only capable of generating text, but also has powerful editing capabilities. With these capabilities, even a poor typist can create error-free text, organized as desired.

The principle purpose of TED-8 is the preparation of a special form of text for the Heath H8 Assembly Language program called source code. The format for assembly language source code is discussed in Section 4, the Heath Assembly Language Manual (HASL-8).

TED-8 is not restricted to producing source code for assembly language. It can also be used to prepare reports, write letters, and edit manuscripts.

Text is stored in a section of memory called the buffer. All memory not used by the text editor program is available as buffer. Editing can be done by command, referencing the desired line. When the buffer is full, text can be placed onto magnetic or punched paper tape files. Additional text can be read in from previously created magnetic or punched paper tape files, or can be placed in the buffer from the terminal keyboard. TED-8's tape handling capabilities allow it to edit large files on a piecemeal basis.

Slightly more than 4096 bytes of H8 memory are occupied by TED-8 and in addition to its program size, 200 additional bytes of working space are also required. The balance of H8 memory is available for use as buffer. An 8K memory, therefore, has approximately 4K of buffer space, which provides sufficient room for a well documented 300-line program. With less program documentation, more lines of code can be accommodated in the buffer. The same buffer accommodates approximately 175 lines of solid text, such as found in a report or letter.

A compression technique is used in the Text Editor so large quantities of blanks will use very little buffer. This allows you to use sufficient blanks between the source code columns to make the source code easy to read, without using proportionate amounts of memory.


TED-8 has many unique features that are discussed in detail on the following pages. Some of these features are:

- 16 commands for text editing versatility.
- Terminal control of output and input operations.
- Command completion and command error analysis.

EDITOR MODES OF OPERATION

Two modes of operation called the “Command Mode” and the “Text Mode,” are available in TED-8. These two modes distinguish between editing commands and text being entered into the buffer.

The Command Mode

The command mode can be subdivided into three areas: input commands, output commands, and editing commands. You execute all commands by typing the appropriate command on the terminal, and follow this with a carriage return; this will be indicated throughout this reference Manual by the symbol . In actual use, no symbol is printed when a carriage return is typed on the terminal, as the carriage or cursor simply moves to the first column of the next line. In the command mode, the prompt character - - (a double dash) appears in the first two columns.

The Text Mode

In this mode you can add text to the buffer from the terminal keyboard, the normal source for most text. But once a source file has been created, it can be stored on a tape file. Later, you can use this tape file as a source of text to be added to the buffer.

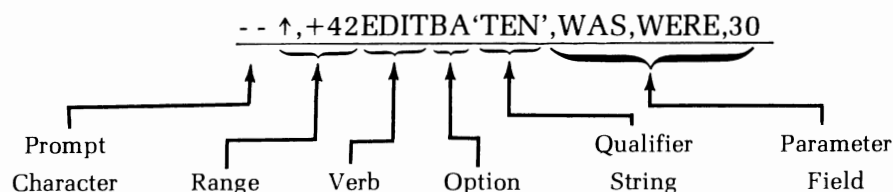
Type the ESCAPE (ESC) key when you wish to return from the text mode to the command mode. This performs two functions. First, it deletes the line of text which it was on when the key was struck. Second, it returns the Text Editor to the command mode. To preserve the last line of text, type a carriage return to generate a new line before striking the ESCAPE key. If the console terminal does not have an ESCAPE key, use CONTROL-C.

THE COMMAND STRUCTURE

The basic commands for TED-8 have the following form:

```
[<range>] [<verb>] [<option>] [<qualifier string>] [<parameters>]
```

The **range** indicates what lines in the buffer the command affects; the **verb** is the basic command. The **option** permits you to view the line before or after (or both) the command is executed. The **qualifier string** limits the command to those lines containing a given string; and the **parameters** (parameter field) contains specific instructions for some commands. For example, the command



is read as follows:

The lines starting with the first line (↑) and ending with the 43rd line (+42) are to be edited (EDIT). The EDIT command replaces one string with another. The option (BA) indicates that you wish to view the lines before and after editing. The affected lines are limited to those containing the string "TEN" (the optional qualifier string). And in this particular case, the parameter field specifies the old and new strings and the count. The word WAS is to be replaced by the word WERE, a maximum of 30 times. NOTE: TED-8 strings used in the range expression or qualifier strings must be enclosed in a single quote (') (apostrophe).

Range Expressions

The range expressions define the buffer lines on which the command is to operate. They may define a single line, or two expressions may be used to define the range over which the command works.

A range expression may consist of:

- A line expression.
- A two-line expression.
- A null.
- A blank.
- An equal sign.



These different range expressions are explained below.

NOTE: Text must be in the buffer before a range expression will be accepted. A bell code will be sound as you try to complete the range expression. To use the following examples, use the INSERT command (see Page 3-12).

SINGLE-LINE EXPRESSIONS

Either one of the following two line expressions may be used to define a single line:

<u>THIS SYMBOL</u>	<u>DEFINES</u>
↑	The first line.
\$	The last line.

TED-8 is always pointing to some line of text. Once you have explicitly pointed to a line by referencing the first line (↑) or the last line (\$), you may make future line references by referencing to the current line pointer. NOTE: Once a DELETE has been executed, the current line pointer is reset and you must explicitly reference a line to reestablish the line pointer.

- A. + count. A plus (+) followed by a decimal number (n) refers to the nth line past the current line pointer.
- B. - count. A minus - (n) refers to the nth line preceding the current line pointer.
- C. + 'string'. The + 'string' refers to the first line in the text buffer which contains the designated 'string' after the current line pointer.
- D. - 'string'. The - 'string' refers to the first line in the buffer, preceding the current line pointer, containing the indicated 'string'.

For example, assume the buffer contains:

```
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
--
```

TED-8 responds as follows to the range commands.

```
--↑PRINT
THIS IS THE FIRST LINE
--$PRINT
THIS IS THE LAST LINE
--↑+2PRINT
THIS IS THE THIRD LINE
--↑+'FLEECE'PRINT
ITS FLEECE WAS WHITE AS SNOW
--$-1PRINT
THE LAMB WAS SURE TO GO
--$-'EVERYWHERE'PRINT
AND EVERYWHERE THAT MARY WENT
--
```

MULTIPLE LINE EXPRESSIONS

Use a two-line expression when you want to define a group of lines to be operated on by the command. Use the comma as a delimiter to separate the start line from the stop line. The symbols ↑, \$, + count, - count, + 'string', and - 'string' have the same meaning as they do with a single-line command. NOTE: A wide range of combinations may be used to identify the first and last lines of a two-line expression.

For example, you could print the contents of the previous buffer using these commands:

```
--↑,↑+3PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
```

```
--↑+2,+3PRINT
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
```

```
--↑+'FLEECE',+1PRINT
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
```

```
--$-'THAT',+'SURE'PRINT
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
```

```
--$-'THAT',+2PRINT
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
```

--

THE BLANK

When the verb is preceded by a single blank (space), the range is the entire buffer. For example, printing the entire buffer is accomplished by:

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
```

--

Note the blank (space) between the prompt character (--) and the word PRINT. The blank is created by typing the space bar on the console terminal.

THE NULL

The NULL expression (the absence of any character) sets a single-line range at the first line of the previous command.

For example:

```
--↑+2,+'FIFTH'PRINT
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
--PRINT
THIS IS THE THIRD LINE
--
```

Note that there is no blank (a NULL) between the prompt (--) and the word PRINT.

```
--↑+2,+1PRINT
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
--, +2PRINT
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
--
```

NOTE: In this example, the NULL starts the range at the first line of the previous range, and the ,+2 directs TED-8 to print two additional lines.



THE EQUAL (=)

The = expression sets the range to the range of the previous command. For example:

```
--$-'MARY',+'GO'PRINT  
  
AND EVERYWHERE THAT MARY WENT  
THIS IS THE SEVENTH LINE  
THE LAMB WAS SURE TO GO  
---=PRINT  
AND EVERYWHERE THAT MARY WENT  
THIS IS THE SEVENTH LINE  
THE LAMB WAS SURE TO GO  
--
```

Note that the command =PRINT causes the same lines to be printed in the first and second halves of the example.

The Verb

The verb specifies the action to be taken by the Editor. For example, the command PRINT or the command EDIT are verbs within the text editor's vocabulary.

All verbs are command completed. As soon as the Text Editor receives enough characters to know that only one command is possible, it prints the balance of the command without any additional keys being struck.

The verb will be refused if it is not valid for the current TED-8 condition. For example, an attempt to PRINT an empty buffer is meaningless and the PRINT verb will be rejected.

For example, when you type the P key, the Text Editor knows that no other command begins with P. Therefore, the P is immediately followed by RINT. However, if you type the N, it does not know if the command NEWIN, NEWOUT, or NEXT is to be used. So the Editor prints NE and waits for a W or X. If it receives a W, it then waits for an I or an O. It completes these by filling in the N or UT. If it receives an X following the E, it then prints the T. A complete list of all of the command verbs, and their specific actions and limitations follows in "The Commands" (Page 3-12).

The Option Field

The option field contains characters which let you view the line to be worked on, and/or the line after it has been worked on.

As its name implies, it is completely optional. You may insert any one of the following three option forms, or none at all.

1. B The BEFORE option displays the line **before** the command is executed.
2. A The AFTER option displays the line **after** the command execution.
3. BA This is a combination of the previous two commands. The line is displayed **before** and **after** command execution.

For example, presume that the buffer erroneously contained

```
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS RED AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
--
```

It may be edited to read correctly by

```
--↑+'RED'EDITBA,RED, WHITE, 1
ITS FLEECE WAS RED AS SNOW
ITS FLEECE WAS WHITE AS SNOW
--
```

Note that after the command EDIT, the options B & A are used to check the work.

NOTE: There are certain times when the command renders the option meaningless. DELETE BA, for example. The AFTER portion of the command is meaningless, as the line (or lines) cannot be displayed after deletion.

The Qualifier String

The qualifier string is a further restriction upon the range expression. It is completely optional. If it is not indicated, it is not used.

The range expression may indicate work over a certain number of sequential lines, starting with a given line. The qualifier string further limits these lines to those within the range containing the string specified in the qualifier field. This string is enclosed in single quotes (') and contains all normal ASCII characters with the exception of the single quote (').

For example, to print the entire buffer (of the previous example), but only those lines with the string 'line':

```
-- PRINT 'LINE'  
THIS IS THE FIRST LINE  
THIS IS THE THIRD LINE  
THIS IS THE FIFTH LINE  
THIS IS THE SEVENTH LINE  
THIS IS THE LAST LINE
```

THE PARAMETER FIELD

This is a special field used with the EDIT, TAB, NEWIN, and NEWOUT commands. These commands require additional operating information, which is placed in the parameter field. The nature of this information depends upon the command used. The exact format is discussed under those commands.

THE COMMANDS

The following paragraphs give a complete description of each of the command verbs. Examples of many commands are also given to demonstrate some of the combinations of range expressions, options, qualifier strings, and parameter fields (if required) that may be used with this command. NOTE: All possible combinations of range expressions, options, qualifier strings, and parameter fields are not given. You must review the section for each of these expressions to determine all possible command structures.

INSERT

The INSERT command places TED-8 in the text mode, and is used to add text to the buffer from the console keyboard. This command adds text to the buffer on

the next line following the first line of the range expression. The second line of the range expression is meaningless. Also, if the range expression is given as a line minus a count, the appending point is still the next line following the indicated line. The option, qualifier string, and parameter fields are ignored for the INSERT command. The range command blank INSERT is a special case that is used to insert a line before the first line in the buffer.

For example:

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
--

--$INSERT
THIS IS AN ADDITIONAL LAST LINE
<CNTRL-C>
--

--↑+'FIFTH'INSERT
THIS IS A NEW LINE INSERTED AFTER THE FIFTH LINE
--

-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
THIS IS A NEW LINE INSERTED AFTER THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
THIS IS AN ADDITIONAL LAST LINE
--
```

NOTE: Use the ESCAPE key (or the CONTROL-C) to return to the command mode. This will cause the current line to be erased. If you strike the ESCAPE key after inserting a partial line, that partial line will be lost. Therefore, to preserve the last line of inserted text, type a carriage return (this will create a new blank line in the text buffer) prior to typing the ESCAPE key.

REPLACE

This command causes the eligible line(s) in the command range to be replaced. Once the executing carriage return is typed, the terminal cursor moves to the start of the first line to be replaced. Type the replacement lines one at a time. Tabs, backspaces, and rub-outs may be used as part of the replacement text. However, typing a carriage return signifies replacement of another line within the command range.

After the lines indicated by the range expression have been replaced, TED-8 reverts to the command mode. Typing ESCAPE (or CONTROL-C) returns the editor to the command mode, erasing the current line.

The option and qualifier strings may be used. There is no parameter field for the REPLACE command.

For example:

```
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
THIS IS A NEW LINE INSERTED AFTER THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
THIS IS AN ADDITIONAL LAST LINE
--

--↑+5REPLACEB
THIS IS A NEW LINE INSERTED AFTER THE FIFTH LINE
THIS LINE IS REPLACED
--

--↑+6,+'LAST'REPLACE'LINE'
THIS REPLACES THE OLD SEVENTH LINE
THIS REPLACES THE OLD LAST LINE
--
```

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
THIS LINE IS REPLACED
AND EVERYWHERE THAT MARY WENT
THIS REPLACES THE OLD SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS REPLACES THE OLD LAST LINE
THIS IS AN ADDITIONAL LAST LINE
--
```

Note: Only lines containing the string 'line' are replaced in the second example, as the string 'line' is used as a qualifier.

DELETE

The DELETE command causes the eligible line(s) in the command range to be deleted. You may use the option B; however, the option A is meaningless. You may also use the qualifier string. There is no parameter field accompanying the DELETE command. You may not delete the entire buffer. To do this, use the BLITZ command.

For example:

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
THIS LINE IS REPLACED
AND EVERYWHERE THAT MARY WENT
THIS REPLACES THE OLD SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS REPLACES THE OLD LAST LINE
THIS IS AN ADDITIONAL LAST LINE
--

--$DELETE
--$PRINT
THIS REPLACES THE OLD LAST LINE
--
```



```
--↑,$-1DELETED
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
THIS LINE IS REPLACED
AND EVERYWHERE THAT MARY WENT
THIS REPLACES THE OLD SEVENTH LINE
THE LAMB WAS SURE TO GO
--PRINT
THIS REPLACES THE OLD LAST LINE
--
```

EDIT

Use the EDIT command to replace one string with another. The string to be replaced and the new string are given in the parameter field of the EDIT command. The parameter field of the EDIT command takes the form:

```
<delimiter> old string <delimiter> new string <delimiter> <count>
```

The **delimiter** is an arbitrary delimiting character. (It may not be a carriage return). The **count** is a decimal count showing the number of replacements to be made. NOTE: Only ONE replacement is made PER LINE. If the count is left null, it defaults to one. If an asterisk (*) is substituted for the count, the value 65,536 is assumed.

The EDIT command makes full use of all range expressions, options, and qualifier strings; and as noted, a specific parameter field.

The following is an example of a text buffer prior to using an EDIT command, and text buffer after the EDIT command is executed.

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
--

--EDIT'LINE',LINE,OF MANY LINES,5
--
```



```
-- PRINT
THIS IS THE FIRST OF MANY LINES
MARY HAD A LITTLE LAMB
THIS IS THE THIRD OF MANY LINES
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH OF MANY LINES
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH OF MANY LINES
THE LAMB WAS SURE TO GO
THIS IS THE LAST OF MANY LINES
--
```

NOTE: The use of an arbitrary delimiting character permits you to choose a delimiting character not contained in the old or new strings. For example, if a comma (,) is used as a delimiter, it may not appear in either string. If either string contains a comma, you can use a slash (/) as a delimiter.

PRINT

The PRINT command causes the eligible line(s) in the command range to be printed. This is the most common method for viewing portions of the text buffer or the entire text buffer. All forms of the range expressions are utilized. The option commands have no effect. You may use the qualifier string to limit the range expression. There is no parameter field for the PRINT command.

NOTE: While the PRINT command is executing, you may type control characters to aid in viewing the text buffer. See "Terminal Control Characters," Page 3-25.

TAB

The TAB command sets tab stops for entering text. This is especially useful when you are entering source code for assembly programs so the label field, op code field, operand field, and comments field can be clearly separated. TED-8 converts a TAB into an appropriate number of blanks and inserts these into the text. The TAB character itself is not inserted into the text. NOTE: Blanks inserted by a TAB do not use proportional quantities of buffer space. The TAB command uses the parameter field to set the tab stop up to a maximum of six tabs. The format of the parameter field is:

TAB, 1, , n

The numbers 1 through n specify the columns, by number, which the TAB is to stop. Previously-set tab stops are discarded once the TAB command is called.

Once the tabs are set, you may space out to the next tab by typing control 1. (This is sometimes labeled as TAB on a keyboard.) If no more tabs are available, a bell code is echoed. The TAB command does not use the range field, option field, or qualifier field.

The following example shows setting the tabs and inserting text using these tabs. NOTE: The circled I (Ⓢ) indicates that the user typed a tab. The cursor automatically moves over to the beginning of the new columns. The user did not type in the blanks between columns. This was performed by the TAB command.

```
--TAB, 10, 20, 30, 40
--INSERT
*      DETERMINE MEMORY LIMIT

INIT1Ⓢ MOVⓈ      M, AⓈ      MOVE BYTE
Ⓢ      DADⓈ      DⓈ        INCREMENT TRIAL ADDRESS
Ⓢ      MOVⓈ      A, MⓈ
Ⓢ      DCRⓈ      MⓈ
Ⓢ      CMPⓈ      MⓈ
Ⓢ      JNEⓈ      INIT1Ⓢ    IF MEMORY CHANGED

INIT2Ⓢ DCXⓈ      H
Ⓢ      SPHLⓈ     Ⓢ        SET STACK POINTER=MEMORY LIMIT-1
Ⓢ      PUSHⓈ     HⓈ      SET *PC* VALUE ON STACK
Ⓢ      LXIⓈ      H, ERROR
Ⓢ      PUSHⓈ     HⓈ      SET: RETURN ADDRESS:
Ⓢ      MVIⓈ      A, UMI.1B+UMI.18+UMI.16X
Ⓢ      OUTⓈ      OP.TPCⓈ  SET 8 BIT, NO PARITY, 1STOP, X16

--
```

BLITZ

The BLITZ command discards all text in the working buffer. Because of the drastic action this command takes, BLITZ followed by a carriage return results in the question SURE? A Y in response to SURE? proceeds with BLITZ. If you inadvertently type a B, thus causing a BLITZ, you may abort the BLITZ by typing an N or any character except Y. The range option, and qualifier and parameter fields are ignored in a BLITZ command. The BLITZ command **does not** reset tab stops.

USE

The USE command displays the number of lines in the command range, the number of memory bytes currently used, and the number of free bytes. The USE command replies giving three values:

1. A line count. The number of lines within the command range. Type USE to display the total number of lines presently used within the buffer. A null USE results in a single line indication.
2. A byte count. The number of bytes used by the entire working buffer, not simply the lines within the command range.
3. A bytes free count. This is the number of remaining bytes in memory.

You can use the range expression and qualifier strings, but they have little meaning. There is NO parameter field. The following example demonstrates the USE command. The H8 employed in this example has 16 K of memory with the following text:

```
*      DETERMINE MEMORY LIMIT

INIT1  MOV   M,A      MOVE BYTE
        DAD   D      INCREMENT TRIAL ADDRESS
        MOV   A,M
        DCR   M
        CMP   M
        JNE   INIT1  IF MEMORY CHANGED

INIT2  DCX   H
        SPHL           SET STACK POINTER=MEMORY LIMIT-1
        PUSH  H      SET *PC* VALUE ON STACK
        LXI   H,ERROR
        PUSH  H      SET:RETURN ADDRESS:
        MVIA,UMI.1B+UMI,L8+UMI.16X
        OP.TPC SET 8 BIT,NO PARITY,1 STOP,X16

--
```

NOTE: TED-8 requires some room for work space. It refuses to allow more text when there are still 200 free bytes. These 200 bytes are its work space.



```
--USE  
LINES=00017  
USED=00338  
FREE=11537  
--
```

NEWIN

The NEWIN command opens a tape file for reading. It permits you to search a tape that contains a number of files for a particular named file to be used by the Editor. (It is not necessary to have an input tape file to do editing. You may create a new file in the buffer by using the INSERT command.) The name of the text file to be opened is contained in the NEWIN command parameter field. The parameter field for the NEWIN command is in the form:

```
<delimiter> <name> <delimiter>
```

Be sure the tape unit is made ready before you type the carriage return. TED-8 will then scan the tape until it finds the named text file.

Note: A file is acceptable if the leading characters in its name match all the characters supplied in the NEWIN parameter field. In other words, the full name need not be supplied. Thus, supplying null as a name (simply two delimiter characters together) allows a match on the first text file found.

After the label record is found, the tape stops before the first block of text. TED-8 responds by indicating the name of the file that has been found. Use a FILL or READ command to input text from this file. The NEWIN command does not use range expressions, options, or qualifier strings.

NOTE: If an input file is opened but not read to the end-of-file record, the NEWIN command asks for a go-ahead prior to searching for a new file. The reply Y, to SURE?, instructs NEWIN to proceed to find the new file.

For example:

```
--NEWIN"USR"  
  
OLD "IN" NOT FINISHED. SURE?YFOUND USR PROGRAM FOR BASIC #1  
--
```

FILL

This command fills the buffer with text from an input file opened by the NEWIN command. FILL causes successive records of text to be read from the input tape until:

1. An end-of-file is read; or
2. Less than 512 bytes (1 block) of free buffer space is left.

The FILL command ignores range, expressions, options, and qualifier strings. It does not use the parameter field. When the prompt returns, FILL is complete.

READ

The READ command is used to input one **record** of text from the tape. If insufficient buffer room exists to hold a record of text, an error message is given. In this situation, you must first empty the buffer before the READ command can be executed. The range option expression and qualifier fields are ignored. The READ command does not use the parameter field.

Text inputted by the READ command is appended to the working buffer. **NOTE:** The READ command differs from the FILL command, as the READ command only inputs one record from the tape. When the prompt returns, READ is complete.

NEWOUT

The NEWOUT command is used to open a new output file. The file name is supplied in the parameter field, in the same manner as in the NEWIN command. The form of the parameter field is:

```
<delimiter> <name> <delimiter>
```

The output tape should be readied before you type the carriage return. TED-8 will write a label record on the tape and then stop it, leaving it ready for text. Use the WRITE, FLUSH, or SAVE commands, as appropriate, to write text onto the tape.

The NEWOUT command does not use range expressions, options, or qualifier strings.

If you use NEWOUT without closing a previously opened file, NEWOUT responds with SURE? A reply of Y permits NEWOUT to write the label for the new file without closing the previous file. **WARNING:** This procedure results in a partially complete file being left on the output tape. The FLUSH or STOP OUTPUT commands are used to write an end-of-file.

WRITE

The WRITE command outputs text from the buffer onto tape. It starts at the top of the buffer and continues to the first line of the command range. Thus, the command

```
$WRITE
```

writes the entire buffer.

This command uses range expressions, options, and qualifier strings. It has not parameter field. **NOTE: After lines are written, they are deleted from the buffer.**

FLUSH

The FLUSH command is used when editing is complete. It causes the working buffer to be written to the output tape, and then any remaining text on the input tape is copied over to the output tape without modification. The FLUSH command then writes an EOF (end-of-file) record on the output tape. This EOF record is needed for subsequent reading of the output file. The FLUSH command does not use range expression, options, qualifier strings, or parameter fields. **NOTE: After the lines are written, they are deleted from the buffer.**

NEXT

The NEXT command performs the following sequence:

```
$WRITE  
FILL
```

This command causes all the text in the buffer to be written to the output tape, and then refills the buffer from the input tape. This command is particularly useful when you are generating long tape files or when you perform minor editing on long tape files. The NEXT command does not use a range expression, options, or qualifier strings. It has no parameter field.

SAVE

The SAVE command outputs text from a working buffer onto tape. The SAVE command starts at the first line in the buffer and continues to the first line of the command range. Thus the command

```
$SAVE
```

saves the entire buffer.

The SAVE command uses range expressions, options, and qualifier strings. It has no parameter field. **NOTE:** After the buffer is saved, you can use a VERIFY to be sure the tape record contains the desired information before you erase the buffer. SAVE is identical to WRITE except the buffer is not deleted after SAVE outputs to the tape. **NOTE:** The command SAVE (no range field) will only save one line.

STOP

There are two forms of the STOP command. They are:

```
STOP INPUT
```

and

```
STOP OUTPUT
```

The STOP INPUT command sets the flag in TED-8, which indicates that an EOF has been read. Once TED-8 executes a STOP INPUT, the previously opened input file is presumed closed.

The STOP OUTPUT command instructs TED-8 to write an EOF record on the current output tape. STOP OUTPUT is used following a \$SAVE or \$WRITE command to close the output file.

SEARCH

The SEARCH command scans through a text file for a given character string. The desired character string is specified in the qualifier field. This command begins the scan at the first line in the command range and continues to the end of the buffer (regardless of the last line in the command range). If the given character string is still not found, the buffer is written onto the output tape and more text is added from the input file. The SEARCH stops when an EOF is read, or when the string is found.

When the string is found, the command range is set to that line. Subsequent commands can reference the line containing the desired string by using an equal (=) for the range expression. The SEARCH command uses the range expression, but does not use option or parameter fields.

For example:

```
-- PRINT
THIS IS THE FIRST LINE
MARY HAD A LITTLE LAMB
THIS IS THE THIRD LINE
ITS FLEECE WAS WHITE AS SNOW
THIS IS THE FIFTH LINE
AND EVERYWHERE THAT MARY WENT
THIS IS THE SEVENTH LINE
THE LAMB WAS SURE TO GO
THIS IS THE LAST LINE
--

--SEARCH"FIFTH"
--=EDITBA,FIFTH,FIFTH,
THIS IS THE FIFTH LINE
THIS IS THE FIFTH LINE
--
```

NOTE: The given character string is enclosed in double quotes (") not single quotes (').



VERIFY

The form of the VERIFY command is:

VERIFY n

where n is the number of records to be verified. If no value is specified, VERIFY verifies to the last record written and stops the tape (a record written by a SAVE for example). In either case, VERIFY stops when an EOF is read.

The VERIFY command is used to check a mass storage record without loading it in the buffer. The VERIFY command checks the sequence and CRC of each record in the file. If a valid ASCII data file is found preceded by a label record, VERIFY responds with

```
RECORD #000 OK
RECORD #001 OK
RECORD #002 OK      etc.
--
```

NOTE: Record #000 is the label record that contains the file name, etc. All other records are compressed ASCII records containing text. The number of records depends on the length of the file.

XPORT

The XPORT command allows you to specify an I/O port other than the Console Terminal for the INSERT (XINSERT) and PRINT (XPRINT) commands. The form of the XPORT command is:

XPORT, nnn

where nnn is the decimal number of the desired alternate I/O port. When this command is executed, TED-8 responds:

CONFIGURE PORT; GO'

and the H8 audio alert is sounded, indicating a return to monitor. You must then configure the alternate Port UART and press GO on the H8 front panel to restart TED-8. For example, to configure port 376₈ (254₁₀):

1. Type XPORT, 254 cr.
2. TED-8 Responds.
CONFIGURE PORT; GO'
3. Enter 377 116 using the MEM key (316 for a 110 baud device).
4. Press OUT.

5. Enter 377 005 using the MEM key.
6. Press OUT.
7. Press GO.

Port 376₈ (377₈ is the Mode/Command word for data port 376) is configured with a mode of 116₈ (eight bit word with a single stop bit) and with a mode of 005 (T×EN and R×E). TED-8 is returned to the operational mode. This port is now configured until an RST command is executed by PAM-8. Once configured, you may use XPRINT and XINSERT as desired.

XINSERT, nnn

The XINSERT command is identical to the INSERT command except that you insert text from port nnn, which you previously configured by XPORT. The XINSERT command is useful for loading text in full ASCII (not compressed), such as that found on paper tapes used with a teleprinter.

XPRINT

The XPRINT command is identical to the PRINT command except that the Output is directed to port nnn, which you previously configured by XPORT. The XPRINT command is used to divert output to a line printer or other similar device.

TERMINAL CONTROL CHARACTERS

TED-8 recognizes a number of control characters. Some of these are valid when you are typing into the TED-8, and others are only valid when TED-8 is printing information on the terminal or line printer. You can make a control character by simultaneously depressing the appropriate key plus the CONTROL (CNTRL) key. This is similar to depressing the SHIFT key along with a letter.

Input Control Characters

The following characters are only valid when you are inputting.

BACKSPACE (CONTROL-H).

The BACKSPACE character causes the **last character** on the line to be discarded. A BACKSPACE code is sent to the terminal so CRT terminals can perform a cursor BACKSPACE. If you attempt to BACKSPACE into column zero, a bell code is echoed noting this illegal operation. BACKSPACE is valid in the command and text modes.

BACKSPACE can be changed when the Text Editor is configured. See "Appendix A," or "Product Installation" on Page 0-19 of the "Introduction" to this "Software Reference Manual."



NOTE: The BACKSPACE is ignored by teleprinters and some other printing terminals.

RUBOUT

The RUBOUT key causes the **current line** to be discarded. A carriage return/line feed is sent to the terminal. Once the RUBOUT is typed, the entire line may be re-entered. RUBOUT is valid in the command and text modes. NOTE: RUBOUT can be changed when the text editor is configured. See "Appendix A," or "Product Installation" on Page 0-19 of the "Introduction to this "Software Reference Manual."

TAB (CONTROL I)

The TAB character is valid only when you are adding text to the buffer from the keyboard. It is used with the INSERT and REPLACE commands. Typing TAB (CONTROL I) causes the cursor to advance to the next tab column. If there is no next tab column, a bell code is echoed. TED-8 converts TAB's into the equivalent number of blanks and stores these in compressed form.

Output Control Characters

The following characters are only valid during execution of the PRINT command.

OUTPUT SUSPENSION AND RESTORATION CNTRL-S AND CNTRL-Q

If you type CONTROL S during an output, it suspends the output to the terminal by setting the suspend flag.

Typing CONTROL Q permits TED-8 to continue execution and outputting information to the terminal. The CONTROL Q clears the suspend flag.

THE DISCARD FLAG, CNTRL-O AND CNTRL-B

If you type a CONTROL O, it toggles the DISCARD FLAG. This stops output on the terminal but does not halt program execution until the program terminates. Typing CONTROL O again toggles the discard flag, resuming output. Type a CONTROL P to clear the discard flag. CONTROL O is often used to discard the remainder of long listings and other similar outputs.

Tape Errors

During a NEWIN, FILL, READ, or VERIFY, either one of the following two error messages may be generated:

SEQ ERR
CHKSUM ERR.

A sequence error (SEQ ERR) is generated if the file records are not in the proper sequence. For example, if two record numbers are not consecutive, an error is generated. The form of the sequence error is

SEQ ERR

Typing a blank after the SEQ ERR message generates a tape error message and the entire file must be reread.

A checksum error (CHKSUM ERR) is generated if the computed CRC for the record in question does not match the CRC recorded at the start of the record. The form of the checksum error message is

CHKSUM ERR IGNORE?

A Y in response to the question "ignore" aborts the error message and the next consecutive record is processed. NOTE: Ignoring the checksum error is not recommended unless there is no other way to recover the data. If a checksum error is flagged, the chances are very good that the data in the designated record is faulty, and may cause TED-8 to crash.

COMMAND COMPLETION

When TED-8 is in the command mode, each terminal keystroke is considered for validity. If the character belongs to no possible command, it is refused and the bell code is echoed to the terminal. If the character is accepted and the command syntax allows only one next character, TED-8 supplies and prints this character for you.

In addition to simple syntax checking, TED-8 processes command range expressions as they are being entered. If you should enter a range expression referring to nonexistent lines, TED-8 refuses further entry and echoes a bell code. Thus, should valid characters be rejected, it indicates that the command range expression is invalid. For example, if you should attempt to type

-↑+'TUESDAY'

and TED-8 refuses the "P" in PRINT, it means that it found no line containing Tuesday. NOTE: This is done before the command was executed, so you can use a backspace to eliminate the TUESDAY and replace this string with a string valid for the text. NOTE: If no information exists in the text buffer, TED-8 will not accept commands which need text to be valid. For example, the command PRINT is invalid when no text exists in the text buffer, as there is nothing to print.



APPENDIX A

Loading Procedures

Loading From the Software Distribution Tape

1. Load the tape in the reader.
2. Ready the tape transport.
3. Press LOAD on the H8 front panel.
4. A single beep indicates a successful load.
5. Press GO on the H8 front panel.
6. The console terminal will respond with:

```
HEATH/WINTEK H8 EDITOR
TED-8 #3.01.00.
COPYRIGHT WINTEK CORP., 01/77
```

7. Configure TED-8 as desired, answering the following questions. Prompt each question by typing its first character on the console terminal keyboard.

- AUTO NEW-LINE (Y/N)?
- BKSP=00008/
- CONSOLE LENGTH=00080/
- HIGH MEMORY=16383/
- LOWER CASE(Y/N)?
- PAD=4/
- RUBOUT=00127/
- SAVE?
-

REQUIRED PATCHES FOR

HEATH H8 TED - B

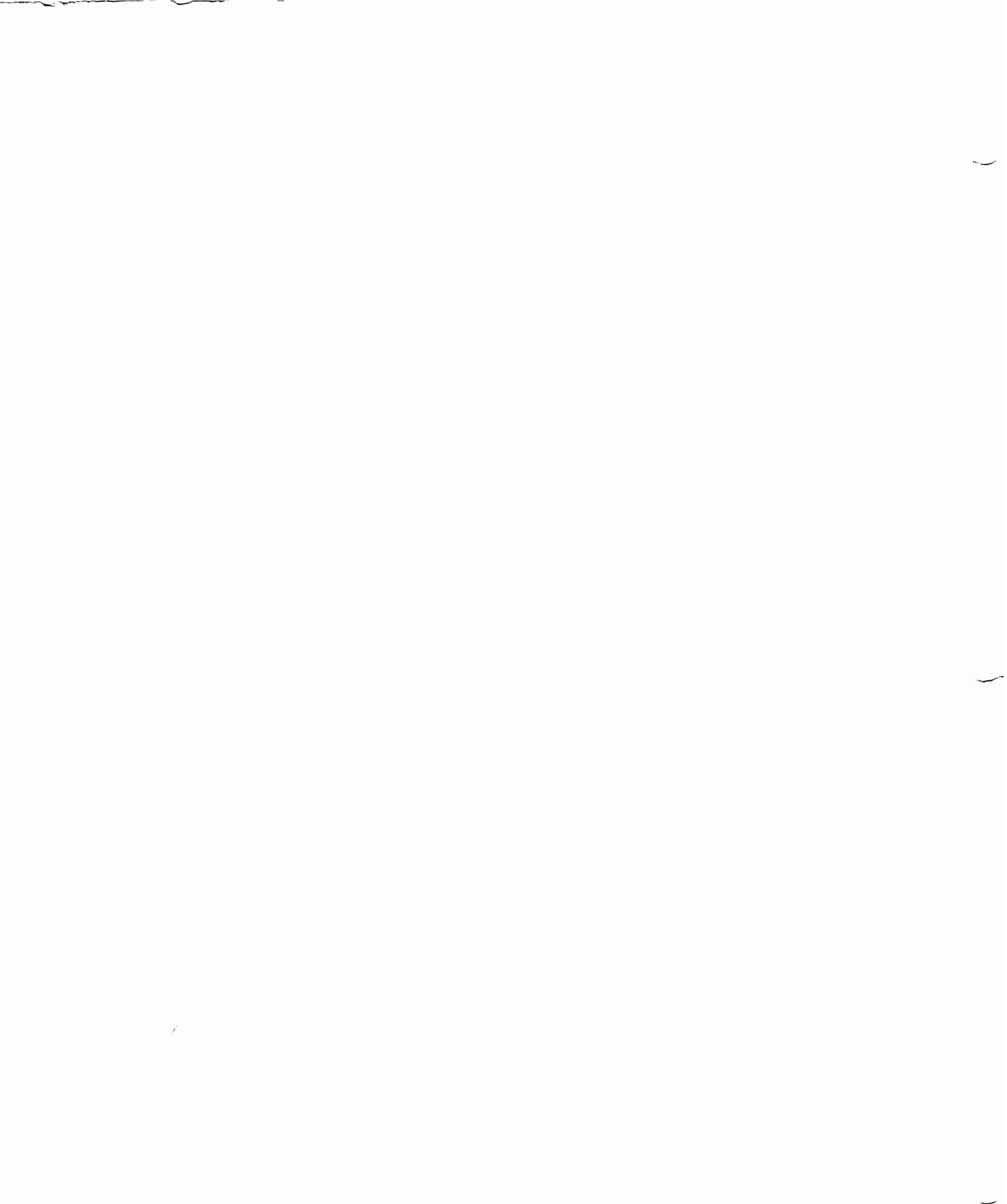
Upgrading 03.01.00.
To 03.01.01.

THESE PATCHES MUST BE INSTALLED IN THE
PRODUCT EVERY TIME A LOAD IS MADE FROM
THE CONFIGURATION TAP

NOTES: None.

USE: The DELETE command will not leave a 'previous command range'.
That is, a command immediately following a DELETE command
must specify a range via '\$', '^', or ' ' (blank).

043356 041
044131 315 240 053 312 175 044 345
044140 052 232 057 353 315 032 061 172
044150 234 341 365
046234 041
050067 041
061032 052 230 057 173 225 311 000
061314 061
064171 061



8. Before executing SAVE, have the tape transport ready at the DUMP port.
9. To use TED-8 directly from the distribution tape, type the RETURN key at any time rather than a question prompt key. TED-8 responds

```
HEATH TED-8 #3.01.00.
```

```
--
```

The Text Editor is ready to use.

Loading From Configured Tape

1. Load the tape in the tape transport.
2. Ready the tape transport.
3. Press LOAD on the H8 front panel.
4. A single beep indicates a successful load.
5. Press GO on the H8 front panel.
6. The console terminal responds with:

```
HEATH/WINTEK TEXT EDITOR #3.01.00.
```

```
--
```

The Text Editor is ready to use in the configured form.



APPENDIX B

Command Summary

Each of the commands for the Heath Text Editor are summarized in this Appendix. For a detailed explanation of each command, refer to “The Commands,” Page 3-12 to 3-24, and to “Terminal Control Characters,” Page 3-25.

Command Structure (Page 3-5)

The general form for an editor command is:

[<RANGE EXPRESSION>] [<VERB>] [<OPTION>] [<QUALIFIER STRING>] [<PARAMETERS>]

Range Expression Forms (Page 3-5)

- 1. NULL — First line in previous command range.
- 2. BLANK — The entire working buffer.
- 3. = — The previous command range.
- 4. A single line expression.
- 5. Two-line expressions.

Line Expression Forms (Pages 3-6 and 3-7)

	EXPRESSION	DEFINITION
1.	↑	The first line in the buffer.
2.	\$	The last line in the buffer.
3.	NULL	The first line in the previous command range.
4.	COMMA	Separates multiple line expressions.
5.	+COUNT	Move forward count decimal lines.
6.	−COUNT	Move backward count decimal lines.
7.	+‘STRING’	Move forward until ‘STRING’ is located.
8.	−‘STRING’	Move backward until ‘STRING’ is located.

Verb (Command) Forms

INSERT	Add to text buffer from keyboard. ESCAPE returns Editor to command mode (Page 3-12).
REPLACE	Replace eligible lines in command range from keyboard. ESCAPE returns Editor to command mode (Page 3-14).
DELETE	Deletes eligible lines in command range (*except entire buffer) (Page 3-15).
EDIT	Replaces old string with new string once per line. Parameter field: <arbitrary delimiter> old string <arbitrary delimiter> count. Count is decimal number of replacements (Page 3-16).
PRINT	Print eligible lines on console terminal (Page 3-17).
TAB	Sets TAB stops. Parameter field format is Tab 1, . . . , n. 1-n are up to 6 column numbers (Page 3-17).
BLITZ	Discards all text after a Y reply to SURE ? (Page 3-18).
USE	Displays number of lines in buffer, bytes used, and bytes free (Page 3-19).
NEWIN	Opens a new tape file to be read in. Parameter field specifies file name <delimiter> <name> <delimiter> (Page 3-20).



FILL	Fills the working buffer with text from input tape. Stops at end-of-file or less than 512 free bytes (Page 3-20).
READ	Reads one additional block (512 bytes) from input tape (Page 3-21).
NEWOUT	Opens a new tape file for output. Parameter field specifies file name <delimiter> <name> <delimiter> (Page 3-21).
WRITE	Writes text from the working buffer to output tape. Writes start of buffer to first line of common range. After writing, lines are deleted (Page 3-21).
FLUSH	Writes working buffer, balance of input file, and end-of-file character onto output tape. Use when editing is complete. Text is deleted when complete. (Page 3-22).
NEXT	Writes working buffer onto output tape. Then fills buffer from input tape (Page 3-22).
SAVE	Saves text from the working buffer on the output tape. Saves start of buffer to first line of command range. Buffer is not deleted (Page 3-22).
SEARCH	Searches text buffer and input tape after initial line for 'STRING'. Search stops when STRING is found or end-of-file is found. Command range set to line containing 'STRING' (Page 3-23).

VERIFY	Verifies the contents of file 'name.' Performs a CRC and a sequence test on each record and compares this computed CRC to the recorded CRC (Page 3-24).
XINSERT	Add to the Text buffer from the XPORT. Text at the load port is uncompressed ASCII. ESCAPE returns TED-8 to the command mode (Page 3-24).
STOP INPUT	Sets the EOF flag, indicating a closed file (Page 3-23).
STOP OUTPUT	Writes an EOF record to tape. Closes current output file (Page 3-23).
XPORT, nnn	Initializes port nnn (Page 3-24).
XPRINT, nnn	Diverts printing to line printer at port nnn (Page 3-25).

Option (Page 3-11)

The option field is:

B Print line before operating on it.

A Print line after operating on it.

BA Print line before and after operating on it.

NULL No option specified.



Qualifier String (Page 3-12)

Qualifier string, if present, takes the form 'string'. A qualifier string limits range expression to those lines containing the qualifier string.

Parameter Field (Page 3-12)

This field contains extra parameters needed by the EDIT, TAB, NEWIN, and NEWOUT commands. Format is command dependent.

Terminal Control Characters (Page 3-25)

BACKSPACE (CONTROL H)	Deletes one character in command and text modes.
RUBOUT	Deletes line in command and text modes.
TAB (CONTROL-I)	Advances cursor to next TAB column.
CONTROL S	Sets suspend output flag.
CONTROL Q	Resets suspend output flag.
CONTROL O	Toggles discard output flag.
CONTROL P	Resets discard output flag.

APPENDIX C

The following edited source file is typical of one you might create using TED-8. It is intended to show examples of some of TED-8's editing capabilities. The assembly of this example is shown on Page 4-53 and it is used as a special BASIC function in "Appendix E" of Section 5.

```

-- PRINT
FPNRM      ORG      100000A-160Q
            EQU      073173A
            INX      B          INC
            INX      B          TO
            INX      B          EXPONENT
            LDAX     B
            ANA      B          SET CONDX CODE
            RZ
            DCR      A          /2
            JZ       USRI       IF UNDER FLOW
            DCR      A          /2 again (/4)
USRI        STAX     B
            CALL     FPNRM
            RET

            END      START

--↑↑'EQU' INSERTB
FPNRM      EQU      073173A
START      INX      B          INC
--↑↑'073173A' EDITBA, 073173A, 063207A, 1
FPNRM      EQU      073173A
FPNRM      EQU      063207A
-- PRINT
            ORG      100000A-160Q
FPNRM      EQU      063207A
START      INX      B          INC
            INX      B          TO
            INX      B          EXPONENT
            LDAX     B          (A) = ACCX EXP
            ANA      B          SET CONDX CODE
            RZ
            DCR      A          /2
            JZ       USRI       IF UNDER FLOW
            DCR      A          /2 again (/4)
USRI        STAX     B
            CALL     FPNRM
            RET

            END      START

```

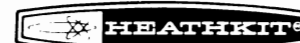
add start (arrow from PRINT to FPNRM)

change to 063207A (arrow from 073173A to 063207A)

Start (arrow from PRINT to FPNRM)

add (arrow from EXPONENT to (A) = ACCX EXP)

Ret to ACCX Normalize (arrow from RET to (A) = ACCX EXP)



```

--↑+'EXPONENT'INSERTB
          INX      B      EXPONENT
          LDAX     A      (A)=ACCX EXP
--'ACC'+1DELETEDB
          LDAX     A
--↑+'/'4'INSERTB
          DCR      A      /2AGAIN(/4)
          USRI     STAX   B      RET TO ACCX
--'ACCX'+1DELETEDB
          USRI     STAX   B
--'ACCX'INSERTB
          USRI     STAX   B      RET TO ACCX
          CALL     FPNRM   NORMALIZE
--'NORMALIZE'+1DELETEDB
          CALL     FPNRM
-- PRINT
          ORG      120000A-160Q
FPNRM     EQU      063207A
START     INX      B      INC UP
          INX      B      TO
          INX      B      EXPONENT
          LDAX     B      (A)=ACCX EXP
          ANA      A      SET CONDX CODE
          RZ
          DCR      A      /2
          JZ       USRI   IF UNDER FLOW
          DCR      A      /2AGAIN(/4)
USRI      STAX     B      RET TO ACCX
          CALL     FPNRM   NORMALIZE
          RET      IN CASE 0

          END      START

--NEWOUT"USR PROGRAM FOR BASIC #1.0"
--$SAVE
--STOP OUTPUT
SURE?Y
--VERIFY,5
RECORD #000 OK
RECORD #001 OK
RECORD #002 OK
*EOF*
--

```

INDEX

- After A, 3-11
- Arbitrary Delimiting Character, 3-17
- Backspace, 3-25
- Before (B), 3-11
- Blank, 3-8
- Blitz, 3-18
- Buffer, 3-3
- Checksum Error, 3-27
- Command Completion, 3-27
- Command Mode, 3-4
- Command Structure, 3-5 ff.
- Command Summary, 3-30
- Compression (Blanks), 3-4
- Control Characters, 3-25
- Control C, 3-4, 3-13, 3-14
- Control H, 3-25
- CNTRL-O, 3-26
- CNTRL-P, 3-26
- CNTRL-Q, 3-26
- CNTRL-S, 3-26
- Delete, 3-15
- Equal (=), 3-10
- Escape key, 3-4, 3-13, 3-14
- Fill, 3-20
- Flush, 3-22
- Insert before first line, 3-13
- Insert, 3-12
- Loading TED-8. 3-28 ff,
- Modes, 3-4
- Multiple Line Expression, 3-7
- Newin, 3-20
- Newout, 3-21
- Next, 3-22
- Null, 3-9
- Option Field, 3-11
- Parameter Field, 3-12
- Print, 3-17
- Qualifier String, 3-12
- Range Expressions, 3-6, 3-12
- Read, 3-21
- Replace, 3-14
- Rubout, 3-26
- Save, 3-22
- Search, 3-23
- Sequence Error, 3-27
- Single-Line Expressions, 3-6
- Stop, Input, 3-23
- Stop, Output, 3-23
- Sure, 3-18, 3-20
- Tab, 3-17, 3-26
- Tape Errors, 3-26
- Text Mode, 3-4
- Use, 3-19
- Verb, 3-10
- Verify, 3-24
- Write, 3-21
- XINSERT, 3-25
- XPORT, 3-24
- XPRINT, 3-25