MICROPERIPHERALS
A Division of PERTEC Computer Corporation
6741 VARIEL AVENUE - CANOGA PARK - CALIFORNIA 91303 - PHONE: (213) 348-1391
TELEX/TWX 910-494-2788

TEXT EDITOR

Revision A
January, 1977

NOTICE

TABLE OF CONTENTS

INTRODUCTION

The iCOM Text Editor allows the user to create and modify alphanumeric text with capabilities including inserting, deleting and changing characters or lines, resulting in output of text to the user designated floppy disk file. Also included in the Text Editor are line edit, search, automatic tabs the user can set, and command iteration.

SECTION I

LOADING AND OPERATION

1-1 LOADING

When loading the iCOM Text Editor, two assumptions are made:

a. The text to be edited resides in a file on a floppy disk in the drive

b. There is sufficient room for the output file to be added to the output diskette.

From the FDOS-II command mode, type:

EDIT,INPUT-FILENAME,OUTPUT-FILENAME(Cr)

The Text Editor will print:

"ICOM 8080 TEXT EDITOR VER X.X"

and the prompt (@) character will appear. The Text Editor is now ready to accept command input.

1-2 OPERATION

The Text Editor operates on input from one of two sources:

a. The system console device

b. The specified floppy disk input file.

The program stores the input into a memory buffer, called the workspace. The buffer pointer locates the point at which operation will be performed in the workspace. The edit process would take these steps:

- Read text from the input file into a buffer
- Write edit commands to modify text
- Output modified text to output file

1-3 BUFFER POINTER

Following the Append command, the buffer pointer is positioned at the beginning of a workspace prior to execution of any commands.

EXAMPLE:

BUFFER POINTER

When deleting or inserting a character, the buffer pointer is moved to the location where the edit or change will take place.

The buffer pointer always resides between two characters.

EXAMPLE:

BUFFER POINTR

This buffer pointer is located between the "T" and "R". Insertion of the character "E" would give this string:

BUFFER POINTER

Note that the buffer pointer now resides after the inserted character.

1-4 CARRIAGE RETURN

The Text Editor divides contents of the workspace into two classifications: CHARACTERS and LINES. A LINE is the space between two line feed characters. A CHARACTER is a single ASCII character. The Text Editor treats CARRIAGE RETURN and LINE FEED as characters which can be manipulated as all other characters are. The Text Editor divides a line, in this case the alphabet, as follows:

EXAMPLE:

```
Line 1:  ABCDEFGHI(Cr)(Lf)
Line 2:  IJKLMNOP(Cr)(Lf)
Line 3:  QRSTUVWXYZ(Cr)(Lf)
```

If the first line feed (Lf) character were removed, the Text Editor would handle the data in 2 lines:

```
Line 1:  ABCDEFGH(Cr)IJKLMNOP(Cr)(Lf)
Line 2:  QRSTUVWXYZ(Cr)(Lf)
```

1-5 COMMAND EXECUTION AND TERMINATION

Text Editor commands are single letters typed in response to the prompt (@) printed by the program. Arguments may be associated with the commands. Commands are terminated and executed with two ESCAPE or ALT MODE characters, which are echoed with the dollar sign ($).

Note: Do not mistake use of the CARRIAGE RETURN key for the ESCAPE character, since CARRIAGE RETURN is regarded as data by the Text Editor.

ESCAPE key is used as the command terminator

LINE FEED is used as the internal line terminator.

The Text Editor automatically supplies a LINE FEED whenever a CARRIAGE RETURN (Cr) is typed, so it is not necessary to insert the LINE FEED.

BREAK character is used to terminate execution of a command which automatically returns the operation to an input mode.

```
EXAMPLE:    @(printed by program)35T$$

Line 1:     ICOM MANUFACTURES(Cr)
Line 2:     MICROPERIPHERALS(Cr)
Line 3:     FOR ALL(break)
            @(break typed by operator)
            (Now in command input mode)
```

1-6 n COMMAND

The range values of n permitted are -254 to +255. If not in this range, n is evaluated modulo 256. If not present, n is assumed to be positive 1.

SECTION II

USE OF TEXT EDITOR COMMANDS

2-1 INTRODUCTION

Text Editor commands are divided into four groups of function. They are:

INPUT

BUFFER POINTER MANIPULATION

DATA MANIPULATION

OUTPUT

Also included in this section is an explanation of:

COMMAND STRINGS

COMMAND ITERATIONS

USE OF TABS

To illustrate the use of these commands, the following phrase will be used to exemplify data in the workspace:

ICOM MANUFACTURES MICROPERIPHERALS FOR ALL MICROPROCESSOR SYSTEMS

2-2 INPUT COMMANDS

2-3 "A" COMMAND (Append)

FORMAT: A$$

The Append command causes text to be read from the specified disk file and appended to the workspace until one of the following is achieved:

End of file

End of file character (CTL-Z) read

Workspace full

50 lines read

"A" may be repeated until a complete input file is read. However, when lengthy files are being edited, it is recommended that only 3 or 4 Appends be issued at any one time to avoid the possibility of exceeding memory capacity. When more text is required, the 255P$$ will write the text in the workspace to the output file, clearing the workspace for additional Appends.

To execute "A", type A$$ (escape character) in response to the prompt (@). The Text Editor will type @ at the left margin when loading is completed. The Append will supply its own end-of-file when the end of the file is reached, and only actual text characters will be put in the workspace.

2-4 "I" COMMAND (Insert)

FORMAT: Istring$$

The Insert command causes text to be input into the workspace by inserting text at the location of the buffer pointer. After insertion, the buffer pointer is positioned to the right of the last character inserted. The Insert command is followed by an argument consisting of the text to be inserted, as follows:

EXAMPLE: @IICOM MANUFACTURES MICROPERIPHERALS$$

The "I" command will now cause the text to be inserted into the workspace at the location of the buffer pointer; in this case, the buffer pointer is positioned immediately after the "S". To see the relationship of the "I" command to the buffer pointer, the following example is given:

Original workspace text: ICOM MANUFACTURES MICROPERIPHERALS

"I" command issued at position of buffer pointer: @I EXCELLENT(sp)(Cr)(Lf)$$

Note the inclusion of CARRIAGE RETURN and LINE FEED following the inserted text, and a space preceding the word EXCELLENT.

Workspace now contains: ICOM MANUFACTURES EXCELLENT MICROPERIPHERALS

The argument to an I command may be of any length that does not exceed the number of characters remaining in the workspace, and may be made up of any characters except ESCAPE, ALT MODE, or BREAK.

If the inserted text exceeds space available in the workspace, the Text Editor echoes the BELL character. The command should then be terminated and the text stored in the output file using the PUNCH command. (See Part IV., OUTPUT COMMANDS, in this section.)

2-5 BUFFER POINTER MANIPULATION COMMANDS

B - Beginning of Workspace

L - Line

M - Move

Z - End of Workspace

2-6 B COMMAND (Beginning of Workspace)

FORMAT: B$$

The B command positions the buffer pointer to the beginning of the workspace.

2-7 L COMMAND (Line)

FORMAT: nL$$

The L command moves the buffer pointer the number of lines specified by n, either backward or forward, depending upon whether the + or - is designated. Sign default is positive (+). An argument of Ø causes the buffer pointer to move to the beginning of the current line. If n is greater than the number of lines between the buffer pointer and the end, or beginning of the workspace, the buffer pointer is moved only to the end, or beginning of the workspace.

NOTE: The LINE FEED character is not used here. The Text Editor automatically provides LINE FEED when CARRIAGE RETURN is used.

2-8 M COMMAND (Move)

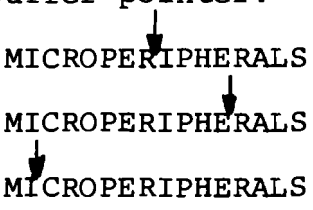
FORMAT: nM$$

The M command moves the buffer pointer forward or backward the number of characters specified by n, depending upon whether a positive or negative argument is used.

If n is greater than the number of characters from the buffer pointer to either the beginning or the end, the buffer pointer will only move to the beginning or the end. If n is Ø, the buffer pointer will remain in place.

EXAMPLE:

Present location of buffer pointer:



2-9 Z COMMAND (End of Workspace)

FORMAT: Z$$

The Z command is used to position the buffer pointer to the end of the workspace and is used prior to appending text to the end of the workspace.

2-10 DATA MANIPULATION COMMANDS

S - Search

C - Change

D - Delete

K - Kill

2-11 S COMMAND (Search)

FORMAT: Stext$$

The S command causes the Text Editor to search for a specified character string in the workspace. The string to be searched out appears as an alphanumeric argument after the S command is input, and the Text Editor begins its search at the current location of the buffer pointer. If a match is found, the buffer pointer is positioned immediately after the last character of the matched character string. If the end of the workspace is reached before locating the character string, the Text Editor prints "CANNOT FIND". The search function argument is limited to 16 characters.

EXAMPLE: The phrase, "FOR ALL MICROPROCESSOR SYSTEMS" will be added to "ICOM MANUFACTURES MICROPERIPHERALS".

LINE	INSTRUCTION	COMMENT
1	@SMICROPERIPHERALS$$	Text Editor searches for string
2	CANNOT FINE "MICROPERIPHERALS" @	String not found. Either it does not exist in the workspace or the buffer pointer resides past occurrence of the string
3	@B$$ @	Buffer pointer is moved to the beginning of the workspace.
4	@SMICROPERIPHERALS$$ @	String searched from beginning of workspace.
5	@IFOR ALL MICROPROCESSOR(Cr) SYSTEMS$$	String is found and new text is inserted using the I command.
6	@-2L$$ @	Buffer pointer is moved back two lines.
7	@3T$$ ICOM MANUFACTURES MICROPERIPHERALS(Cr)(Lf) FOR ALL MICROPROCESSOR(Cr)(Lf) SYSTEMS	Three lines are printed.
8	@	Text Editor is ready to accept more commands.

2-12 C COMMAND (Change)

FORMAT: Carg$text$$

The C command causes a specified character string of up to 16 characters to be searched out and changed by substituting it with another string. If the search string is greater than 16 characters, only the first 16 are recognized and affected. If the text string is greater than 16 characters, the entire text string will replace the search (argument) string.

EXAMPLE: Change the word "MANUFACTURES" to "MAKES"

After the search requirement is satisfied, the buffer pointer will reside after the "S" of MANUFACTURES. The search argument is then deleted and the replacement text is inserted ahead of the buffer pointer.

LINE	INSTRUCTION	COMMENT
1	@CMANUFACTURES$MAKES$$	Workspace is searched for the string MANUFACTURES, and when found, changes it for the string MAKES.
2	@ØL3T$$	Buffer pointer is moved to the beginning of line (ØL) and 3 lines are typed.
3	ICOM MAKES MICROPERIPHERALS(Cr)(Lf) FOR ALL MICROPROCESSOR(Cr)(Lf) SYSTEMS	MAKES has been successfully changed from MANUFACTURES.

2-13 D COMMAND (Delete)

FORMAT: nD$$

The D command causes the number of characters specified by n to be deleted from the workspace. The positive or negative sign determines which direction from the buffer pointer characters will be deleted. A Ø argument will effect no movement of the buffer pointer.

EXAMPLE:

Current location of buffer pointer: MICROPERIPHERALS ARE

If command is 4D: MICROPERIPHERALS

If command is -7D MICROPERI ARE

SAMPLE PHRASE, when deletion is complete, will read:

"ICOM MANUFACTURES MICROPERIPHERALS FOR MICROPROCESSOR SYSTEMS"

The word, "ALL" will be deleted.

LINE	INSTRUCTION	COMMENT
1	@2L$$	Buffer pointer moves 2 lines down
2	@4D$$	4 characters after buffer pointer are deleted, including
3	@-2L4T$$	space. Buffer pointer is moved back 2 lines and then prints 4 lines.
4	ICOM MANUFACTURES(Cr)(Lf) MICROPERIPHERALS FOR(Cr)(Lf) MICROPROCESSOR(Cr)(Lf) SYSTEMS(Cr)(Lf)	The edit is performed successfully.
	@	Editor is ready to accept new commands.

2-14 K COMMAND (Kill)

FORMAT: nK$$

The K command performs the same operation to a line that the D command performs to a character. In this case, however, if n is Ø, the characters from the buffer pointer back to the first previous line feed will be deleted. If n is greater than the number of lines between the buffer pointer and the end, or beginning of the workspace, the lines will be deleted and the buffer pointer will reside at the end, or beginning, of the workspace. + is assumed when the minus sign does not qualify n. If the buffer pointer resides in the middle of a line and the command is 2K, the reaminder of the line is deleted, and the following line is deleted.

EXAMPLE FOR PHRASE:

```
ICOM MANUFACTURES(Cr)(Lf)
MICROPERIPHERALS FOR(Cr)(Lf)
ALL MICROPROCESSOR(Cr)(Lf)
SYSTEMS(Cr)(Lf)
```

LINE	INSTRUCTION	COMMENT
1	@2L$$	Buffer pointer is moved forward 2 lines, to beginning of ALL...
2	@1K$$	One line following buffer pointer is deleted.
3	@B$$	Buffer pointer is moved to beginning of workspace.
4	@3T$$	Three lines are printed to check work.
5	ICOM MANUFACTURES(Cr)(Lf) MICROPERIPHERALS FOR(Cr)(Lf) SYSTEMS(Cr)(Lf) @	ALL MICROPROCESSOR has been deleted. The Text Editor now is ready to accept more commands.

2-15 OUTPUT COMMANDS

Output commands are:

P - Punch

T - Type

E - End

Output commands cause the text in the workspace to be output either to the system console device or the specified output disk file where it is either printed or recorded, depending upon the command.

2-16 P COMMAND (Punch)

FORMAT: nP$$

The P command causes the specified number of lines to be written to the output file.

EXAMPLE FOR: ICOM MANUFACTURES
MICROPERIPHERALS FOR
ALL MICROPROCESSOR
SYSTEMS

LINE	INSTRUCTION	COMMENT
1	@4P$$	Four lines are written.
2	ICOM MANUFACTURES(Cr)(Lf) MICROPERIPHERALS FOR(Cr)(Lf) ALL MICROPROCESSOR(Cr)(Lf) SYSTEMS(Cr)(Lf)	Four lines are written to the output file.
3	@4T$$ ABCDEFG(Cr)(Lf) HIJKLMN(CR)(Lf) OPQRSTU(Cr)(Lf) VWXYZ(Cr)(Lf)	Four lines are typed from new beginning of workspace.
4	@	Control is returned to Editor.

2-17 T COMMAND (Type)

FORMAT: nT$$

The T command causes the number of lines specified by n to be output to the system console device. Typing begins where the buffer pointer currently resides. n specifies the number of lines to be typed, if positive. If a -n is designated, T causes the number of previous lines to be typed. If Ø, the characters from the previous LINE FEED to the current buffer pointer location are typed.

If n is greater than the number of lines before or after the buffer pointer, only the existing lines are typed.

EXAMPLE for location of buffer pointer before fourth line in workspace:

LINE	INSTRUCTION	COMMENT
1	@T$$	T command types one line after buffer pointer
2	@-3T$$ ICOM MANUFACTURES(Cr)(Lf) MICROPERIPHERALS FOR(Cr)(Lf) ALL MICROPROCESSOR(Cr)(Lf)	Types 3 lines preceding buffer pointer.
3	@2T$$ SYSTEMS(Cr)(Lf)	Types 2 lines following buffer pointer. If existing number of lines is less than the argument, only existing lines will be typed.
4	@	Control is returned to Text Editor.

2-18 E COMMAND (End)

FORMAT: E$$

The E command causes the entire workspace to be written to the output file, any remaining text in the input file copied to the output file, FDOS-II loaded and executed, and the output file closed.

2-19 COMMAND STRINGS

Command strings may be made up of any number of commands, which are chained together, and will be executed as individual commands from left to right. Three commands, however, must NOT be included in the command string, but must be separated from other commands. They are the C, S, and I commands and are separated by using the escape (echoed as $) character. In this way the Text Editor can distinguish between text to be inserted or searched, and the next command.

COMMAND STRING EXAMPLE:	@B2L4K3DISYSTEMS$$
This would result in:	Buffer pointer positioned at beginning of workspace.
	Move buffer pointer down two lines.
	Delete 4 lines.
	Delete 3 characters.
	Insert string "SYSTEMS" into workspace.
INCORRECT INSERT COMMAND:	ISYSTEMS3L4D$$
This would result in:	SYSTEMS3L4D to be placed in workspace.
CORRECT FORMAT:	ISYSTEMS$3L4D$$
This would result in:	SYSTEMS to be inserted, buffer pointer to be moved the next 3 lines, and four characters to be deleted.

2-20 COMMAND ITERATIONS

The Text Editor allows a command string to be automatically executed the designated number of times specified by n, using command iteration. To cause a command string to automatically execute more than once, "less than" and "greater than" signs are used to surround the command.

FORMAT: n<command string>$$

EXAMPLE:

LINE	INSTRUCTION	COMMENT
1	@4<ISYSTEMS.(Cr)(Lf)>B4T$$	Command is entered to insert string SYSTEMS.CRLF 4 times. The buffer pointer then moves to the beginning of the workspace, and types the first four lines of the workspace.
2	SYSTEMS. SYSTEMS. SYSTEMS. SYSTEMS.	The string SYSTEMS. CRLF had been inserted four times, the pointer moved to beginning of the workspace and 4 lines printed.
3	@	Control is returned to Text Editor.

Command iterations can be nested up to eight deep. If more are designated, the Text Editor will print the error message ITERATION STACK FAULT and the command with error condition removed, will have to be re-entered.

2-21 TABS

Horizontal tab character (CTRL I) is used wherever a space could be used. Tab stops are located every eight position.

EXAMPLE FOR EDITOR COMMAND:

```
@ILABEL(tab)MOV(tab)A,B(tab);(sp)COMMENTS$$
```

PRINTED RESULT:

```
COL     Ø       1       2
Ø123456789Ø123456789Ø123456789Ø

LABEL:  MOV     A,B     ; COMMENT
        JMP     HOME
```

APPENDIX A

TEXT EDITOR COMMANDS

INPUT--The Text Editor accepts input from either the system console device or the specified floppy disk input file. The program stores the input into a memory buffer, or "workspace". A buffer pointer specifies the location in the workspace where operations will be performed. The two input commands are:

A - APPEND

I - INSERT

POINTER CONTROL--Four buffer pointer manipulation commands move the buffer pointer to various locations in the workspace, permitting operations to occur at a designated point. The four commands are:

B - BEGINNING

L - LINE

M - MOVE

Z - END OF WORKSPACE

EDIT--Data Manipulation Commands do the actual editing of the workspace data. They permit characters, or lines to be deleted, changed or searched out. The four data manipulation commands are:

C - CHANGE

D - DELETE

K - KILL

S - SEARCH

OUTPUT--Output commands cause the text in the workspace to be output to the system console device or the specified output disk file, where it is either printed or recorded, depending upon the command. The three output commands are:

P - PUNCH

T - TYPE

E - END

COMMAND STRINGS--Command strings enable Text Editor commands to be chained together. Any number of commands may constitute strings and are executed as individual commands from left to right. Three commands, however, must be separated by the ESC ($) character. They are:

C - CHANGE

S - SEARCH

I - INSERT

COMMAND INTERATION--Command iteration allows a specified command string to execute automatically for a designated number of times. This is accomplished by bracketing the command with the "less than" and the "greater than" signs. The specified string may be designated to repeat up to eight times.

TAB STOPS--Tab stops in the Test Editor permit the user to produce a highly readable listing with the use of the horizontal tab character (CTRL I). Tab stops are located at every eighth position and are used wherever a space will appear.

APPENDIX B

iCOM TEXT EDITOR COMMAND SUMMARY

COMMAND	FORMAT	FUNCTION	PAGE
A - APPEND	A$$	Brings data in from file, 50 lines of text for each "A" command.	4
B - BEGINNING	B$$	Moves pointer to beginning of buffer.	6
C - CHANGE	Cstring1$string2$$	Changes string of up to 16 characters for another string.	9
D - DELETE	nD44	Removes specified number of characters from workspace.	9
E - END	E$$	Causes Editor to write contents of buffer to output file and any remaining data in the input file to be written to the output file and returned to FDOS-II.	13
I - INSERT	Itext$$	Inserts text from keyboard at the buffer pointer location.	5
K - KILL	nK$$	Deletes one line for each "K" command, or number of lines for "nK".	1.
L - LINE	nL$$	Moves buffer pointer specified number of lines either backward or forward.	6
M - MOVE	nM$$	Moves buffer pointer specified number of characters either backward or forward.	6
P - PUNCH	nP$$	Causes specified number of lines to be written to the output file.	12
S - SEARCH	Stext$$	Causes Editor to search for first instance of specified character string in workspace.	7

COMMAND	FORMAT	FUNCTION	PAGE
T - TYPE	nT$$	Causes number of lines specified by n to be output to console device beginning at current location of buffer pointer.	12
Z - END OF WORKSPACE	Z$$	Moves pointer to end of buffer.	7