MTU-130

FULL SCREEN TEXT EDITOR

PRELIMINARY

~~September, 1981~~
January, 1982

REV. ~~A~~ C

MTU-130

FULL SCREEN TEXT EDITOR

JANUARY, 1982

REV. C

TABLE OF CONTENTS

1. INTRODUCTION 1

2. BEGINNING THE EDIT SESSION 2

3. THE EDITOR DISPLAY 2
3.1 The Function Key Menus 3

4. TEXT MODE EDITING 4
4.1 Deleting and Inserting Characters 5
4.2 Deleting and Adding Lines 5

5. COMMAND MODE 5
5.1 FIND 6
5.2 MARK 8
5.3 CHANGE 8
5.4 ADDFILE 9
5.5 INSFILE 9
5.6 TABSTOPS 10
5.7 UPDATE 10
5.8 QUIT 10
5.9 MACRO 10
5.10 JOINLINE 11
5.11 SPLIT 11

6. SPECIAL CHARACTERS 12

7. APPENDICES 13
A. Control key Summary 13
B. Command/Function Summary 14

1. INTRODUCTION

The MTU Editor is a general-purpose, full-screen text editor for generating and modifying files of ASCII characters. It is well suited for editing programs and data. It is not a word processor and does not perform operations such as centering and justification. It is very easy to use since the text displayed on the screen is always an exact image of the final resulting output. Corrections are made by "steering" the cursor to the desired area and striking over existing text. The "INSERT" and "DELETE" keys are used to quickly insert or delete both characters and lines. A "FIND" command is available to rapidly search for specified lines or character strings in the file. The Editor can perform global or limited search-and-replace operations. It can copy, move, save and delete multi-line blocks of text and can merge other files.

The MTU Editor operates using a unique proprietary algorithm which allows files larger than memory to be edited "in place". Unlike all other editors, you can "move about" freely in the file, either forward or backward without any limitations. No special file formats are required. The Editor operates on an ordinary ASCII text file. When you QUIT the Editor, the file will be updated.

2. BEGINNING THE EDIT SESSION

The form of the CODOS command needed to enter the Editor is:

```
EDIT <filename> <linesize>
```

Filename is the name of a new or existing text file to be edited. Linesize is an optional argument which specifies the maximum line length in characters. The linesize default value is 80 characters, the width of a display line. You may specify a value between 1 and 160 decimal characters. Since the display is 80 characters wide, if you specify more than 80 characters, the Editor will operate in "Landscape Mode". In Landscape Mode, each text line becomes two display lines. If a given text line is less than 80 characters in Landscape Mode, two display lines will still be used to show it but the second one will be blank. If you specify a linesize, make sure it is large enough to accommodate all the lines in the file. If the Editor encounters lines longer than the maximum linesize specified, it will truncate them. If you wish to start from scratch with a new file, then type in a file name not already in use. In either case, the file must not be on a write-protected disk. Existing files must not be locked.

For example, suppose you wish to start from scratch to create a text file called PROGRAM1.E. The command to do this is:

```
EDIT PROGRAM1.E
```

The filename PROGRAM1.E does not exist so the Editor will start with a "clean slate". The line size defaulted to 80 characters.

If you wish to edit the existing file STARTUP.J on drive 1, the command is:

```
EDIT STARTUP.J:1
```

If you want to edit a file in Landscape Mode with a 136 character maximum line length then the command specifies this:

```
EDIT PROGRAM1.E 136
```

If the file to be edited exists, the Editor prompts:

```
SAVE BACKUP FILENAME? (CR = NONE) =
```

Remember that the Editor edits the file in place so if you need to refer back to the file as it was before the edit session, you will want to make a copy. It is good practice to maintain a backup copy just in case.

3. THE EDITOR DISPLAY

The Editor display consists of three areas on the screen:

1. Command/Status Area

2. Text Window

3. Function Key Legends

The Command/Status Area (CSA) occupies the top three lines of the display. You type commands on the left hand part of the top line to tell the editor to perform certain operations. The right hand portion of the top line displays the file line number of the first line displayed in the text window and the file line number the cursor is on in the text window. Initially both of these line numbers are 1.

The second and third lines of the display are normally blank and are used to display error messages and prompts.

The Text Window is separated from the CSA by a full-width horizontal line which, when there are tabstops set, has "tic" marks running along it. These tic marks show the tabstops that were initialized by CODOS prior to entering the Editor. The default tabstops occur every eight columns starting with column 9. The Text Window occupies the 20 text display lines immediately below this tabstop display line. You will enter all your text in this area and make all your editing changes here. The Text Window will display the first 20 lines of the file, if 20 lines exist. If you elect to edit a file in landscape mode, then only 10 lines will be displayed, each occupying 2 display lines.

3.1 THE FUNCTION KEY MENUS

The bottommost line of the display holds eight function key "boxes". Each box contains a legend labeling a corresponding function key. The order of the legends on the display matches the function keys on the keyboard. The leftmost box on the display represents the leftmost function key and so forth.

If you specify a new file when you enter the Editor, the initial function key legends will be:

```
COMMAND  FIND  MARK  CHANGE    TABSTOP  UPDATE  QUIT  OTHER
```

The cursor will be blinking in the home position in the Text Window. The Editor is ready for you to type in some text. You are in the text mode.

If you are editing an existing file, the initial function key legends will be:

```
TEXT  FIND  MARK  CHANGE    TABSTOP  UPDATE  QUIT  OTHER
```

The cursor will be blinking in the CSA. The Editor is ready for you to enter a command. You are in the command mode.

These legends make up the function key primary menu. The primary menus for editing an old file and a new file are identical with the exception of function key F1. Whenever you press the function key F1, you change modes. For instance, if you start out in the text mode and press F1, the cursor jumps from the text window to the CSA. The legend for function key F1 changes from COMMAND to TEXT. If you want to move out of the Command mode and into the text mode, press F1 again. The cursor will now be in the text window and the legend for F1 is COMMAND.

Depress the rightmost function key, OTHER, to access the secondary menu:

```
TEXT  FIND  JOINLINE  SPLIT  ADDFILE  INSFILE  MACRO  OTHER
```

Depress OTHER to return to the primary menu. With the exception of F1 and the FIND function, all of the functions in the secondary menu will return you to the primary menu when they complete their operations.

MARK has a submenu of its own which displays when you depress MARK:

TEXT FIND MARK DELETE COPY MOVE WRITE CANCEL

Depress CANCEL to return to the primary menu. With the exception of F1 and FIND, all the functions in the MARK submenu will return you to the primary menu when they complete their operations.

4. TEXT MODE EDITING

In Text Mode the cursor is in the text window . If you entered the Editor with a new file name, you are starting with a "blank sheet" and you simply type to create new text. You terminate the line when you press RETURN. If you release CAPS LOCK on the keyboard, you can enter both upper and lower case letters. If you fill up the entire 20 line display area, the text will automatically scroll up one line. Line 1 disappears off the top of the text window and a new blank line appears at the bottom for you to continue typing on. If you make a mistake and want to erase the previous character, you can depress RUBOUT. You may depress BACKSPACE to backup the cursor and type over the existing character to replace it.

The cursor control keys move the cursor in the text window. The left arrow, Cursor-Left, performs the same function as backspace. Notice that if you hold a cursor key down it will automatically repeat after a brief delay. This lets you move larger distances quickly. You can move the cursor to the right without wiping out characters by using the Cursor-Right key. All the cursor control keys will repeat if held down. (Space, backspace, rubout and delete will also repeat if held down.) If you attempt to move beyond the margins of the text window using Cursor-Left or Cursor-Right, the cursor "wraps around: to the opposite margin on the same line. If you hold down the SHIFT key and depress Cursor-Left or Cursor-Right, the cursor will move to the left or right margin, respectively.

You can move down by repeatedly depressing RETURN, or by using the Cursor-Down key. Cursor-Down will not let you move down beyond the last line of the file. Depressing RETURN at the end of the file actually adds new empty ("null") lines to the file. After you reach the last line of visible text, if you can Cursor-Down there are additional null lines in the file. If you Cursor-Down beyond the last displayed line, the Text Window will scroll up to show the next line.

The Cursor-Up key operates in a similar manner. If you Cursor-Up to beyond the top line of the Text Window, the display will scroll down one line to display the previous line. When the display no longer scrolls down you have reached the beginning of the file.

The two numbers in the upper right-hand corner of the CSA change as you move the cursor up and down. The first number is the line number of the file which is displayed on the first line of the Text Window. This number changes when the display scrolls. The second number in the upper right-hand corner of the CSA is called the current line number. The current line number tells you which line the cursor is on in the file. It changes whenever the cursor changes lines. Remember that in Landscape Mode that two display lines make one text line and the current line number is a count of text lines.

You can page throught the text file by holding down the SHIFT key along with Cursor-Up or Cursor-Down. SHIFT-Cursor-Down advances the display to the next 20 lines (10 lines in Landscape Mode) of the file. Both of these functions reposition the cursor to the top of the Text Window.

Use the HOME key to position the cursor on the first character of the top line of the Text Window. SHIFT-HOME returns the beginning of the file to the text window and positions the cursor on the first character in the file.

4.1 DELETING AND INSERTING CHARACTERS

The DELETE key removes the character the cursor is on and closes up the gap if there are any characters to the right on the current line. Holding DELETE down so that it repeats is useful when removing entire words or more. Delete differs from

RUBOUT in that RUBOUT erases the character but leaves a blank space to type on, whereas DELETE removes the character totally and pulls the next character in under the cursor.

Depressing the INSERT key causes the Editor to enter the "Insert Mode". Thereafter any characters typed will be inserted in front of the character the cursor is presently on and any remaining characters on the line will shift right one space. To get out of insert mode, depress RETURN, BACKSPACE, or any cursor control key. RUBOUT leaves you in Insert Mode so you can correct errors while you are inserting. If you attempt to insert more characters than will fit on the line, the Editor will sound a tone and issue a line overflow error message.

4.2 DELETING AND ADDING LINES

If you hold down the SHIFT key and depress DELETE, the line the cursor is on will disappear and the remaining text on the page will scroll up to close up the space where the line was. The cursor will be positioned where it was but on the next line of text at the completion of the operation. If you continue to hold down SHIFT-DELETE, you will continue to delete lines . When the last line of the file is deleted, the cursor moves up one line and the next SHIFT-DELETE acts on that line.

If you want to start over the line you are typing by clearing it to blanks without deleting it, hold down CONTROL and press X. This erases all characters and positions the cursor to column 1.

Hold down SHIFT and press INSERT to insert a null line above the line the cursor is currently on. Since the null line is inserted above the current line, you can insert lines before the first line of the file. Newly created empty lines are called null lines rather than blank lines because initially they do not contain any characters. A blank line may contain one or more blanks. Blanks are characters to the Editor.

5. COMMAND MODE

You can enter Command Mode from Text Mode at any time by pressing function key F1. The Editor will display the blinking cursor on the command line and wait for you to type in a command. Commands may be in either upper case or lower case letters. Typing in TEXT followed by a carriage return is the equivalent to depressing the TEXT function key. The editing functions you use in the Text Window can also be used to edit your commands, keeping in mind that you have only one line and a maximum line length of 68 characters to work with. You can also recall previous commands by holding down CONTROL and pressing B while in the Command Mode. Depressing any function key other than "text" will leave you in command mode. You may press function keys from text mode to enter a command. A null command (that is, just a carriage return in command mode) is equivalent to depressing the TEXT function key. You may use SHIFT-Cursor-Down and SHIFT-Cursor-Up while in command mode to move the Text Window forward and backward by 20 lines.

5.1 FIND

FIND is used to tell the Editor to search for a specified line number or character string. Line numbers for a file start with 1 and each RETURN terminates one line. The line numbers are not part of the file itself. If you edit a BASIC program which has a line number for the first few characters of each line, these are considered to be part of the text by the Editor and have no relation to the line numbers the Editor uses. The Editor keeps track of its position automatically and the line number associated with any line changes automatically as lines are inserted or deleted above it.

The FIND command can position the Text Window rapidly to any position in the file because it does not display the file as it searches. You can move the Text Window either forward or backward to any specified line. The simplest form of the FIND command is:

```
FIND <linenumber>
```

which will display the specified line number as the first line of the Text Window. For example:

```
FIND 85
```

will display lines 85 through 104 of the file in the text window (lines 85 through 94 in Landscape Mode). If a number is specified which is larger than the last line of the file, the last line of the file will be displayed instead. Alternatively, you may specify a displacement from the line number the cursor is presently on by using a signed numeric argument. For example:

```
FIND +11
```

will advance the display by eleven lines from the second line number displayed at the right of the CSA:

```
FIND -100
```

will backup from the present display position. If you attempt to backup beyond line 1, line 1 will be displayed instead.

In addition to searching for line numbers, the FIND command can also search for a specified character string. This form of the FIND command is:

```
FIND "<string>" [<linelimit>]
```

where <string> is the desired character string. The string must be enclosed in either a set of single or double quotes. The string can include blank characters and can be as long as you can type on the command line. The search for the string is always begun from the current cursor position in the text window rather than from the first line of the display. The search will be continued until the specified string is found or the end of the file is reached. If the end of the file is reached, the message "NOT FOUND" will be displayed. For example:

```
FIND "HIS"
```

will search forward from the current line until the three characters "his" are found in the file. The text window will show this line at the top of the display. If you then enter Text Mode, the cursor will redisplay on the first character of the matching string. If the string is not found, the Text Window will not change and the message "NOT FOUND" appears.

You can also limit the search. This is done by specifying the optional line-limit argument in parenthesis. You specify the last line number to be searched in the file. For example:

```
FIND "SOLAR DEVICE" (150)
```

will only search from the present cursor position up to line 150 for the string. The line limit specified must be beyond the present position or you will get an "ILLEGAL ARGUMENT" error message. You may also specify a positive displacement from the present position in a limited search. For example:

```
FIND "SOLAR DEVICE" (+200)
```

will limit the search to the next 200 lines starting at the current line.

Multiple string or line arguments may be "strung together" on a FIND command. For example, if you are presently at line 200 and wish to search for a string starting with line 1, you can do it with a single FIND command:

```
FIND  1  "100"
```

This moves the Text Window to line 1, without redisplaying it, and then searches forward for the string '100'. The Editor knows that you are looking for the string "100" and not the line number 100 because you enclosed it in quotes. This would be a way to search for a BASIC line number that was actually part of the text.

As a more complex example, suppose you wanted to find the string "JMP" which must occur within 10 lines after the string "JSR", and the "JMP" must occur between lines 100 and 200 of the file. The command to do this would be:

```
FIND 100 "JMP" (200) "JSR" (+10)
```

In the event that a string search fails in a multiple-argument FIND command, the display will be updated to the last "successful" position before the "NOT FOUND" message is given. For example, if the "JMP" was found at line 157 in the example above, but no "JSR" was found within the next 10 lines, then the Text Window would be displayed starting at line 157.

Often the instance of the string which the Editor finds is not the one you had in mind and you wish to repeat the command starting with the next line. Typing a +1 displacement in front of the string will advance the cursor to the next line before beginning the search, for example:

```
FIND +1 "JMP"
```

Alternatively, you can enter the Text Window, advance the cursor by a character or a line and then use the COMMAND function key and CNTRL-B to repeat the FIND command. Since searches always start from the current cursor position, if you don't advance the cursor the FIND command will simply find the same string again.

5.2 MARK

The Mark command allows you to designate a single line or a block of text which you can then delete, copy, move or write to a device. The designated lines are shown in reverse video. When you depress the MARK function key, its secondary menu displays. At the same time, the line the cursor was on while in Text Mode is designated. To designate a block of text you simply mark the beginning and the end of the block. You may mark either end first. When you have designated both ends, the entire block will display in reverse video. SHIFT-Cursor-Up, SHIFT-Cursor-Down and the Find command may be used to get to text not currently displayed in the text window. You can also designate a block of text larger than what is displayable in the text window.

Once your text is marked, you are ready to proceed with one of the secondary menu functions. Depress CANCEL if you do not want to proceed. CANCEL returns you to the primary menu and unmarks the designated text. DELETE will erase the designated lines of text. The remaining text scrolls up to fill in the space. Before you COPY or MOVE, relocate the cursor to where you are going to copy or move using SHIFT-Cursor Up, SHIFT-Cursor-Down or the FIND command. Then depress COPY or MOVE. COPY and MOVE will insert the designated text directly above the line the cursor is on. When you select WRITE you must also specify the name of the file or the device you want the designated text output to. For example:

```
    WRITE  SUBROUTINE1.A
```

will create a new file containing only the lines you marked. If you write to a device such as a printer then your command would be:

```
    WRITE P
```

As with CANCEL, when DELETE, COPY, MOVE OR WRITE operations are completed, the marked lines are restored to normal video and the primary menu redisplays. If you want to copy or move a block larger than 100 lines, you can do this very quickly if you write it to a temporary file and then use the INSFILE command described in section 5.5.

5.3 CHANGE

CHANGE will search out and replace a string throughout all or just part of a file. The search is always forward from the current cursor postion. After you depress CHANGE you must specify the string to be changed and the string you want in its place. Each string must be enclosed in either single or double quotation marks. For example:

```
    CHANGE  "REM"  "READ"
```

The replacement string does not have to be the same size although the replacement will not be made if it causes the line to overflow . If this is the case you will get a line overflow error message instead.

If you wish to search out a string to delete it without replacing it with another string then omit the second string:

```
CHANGE "REM" ""
```

There are also three optional arguments for the CHANGE command which if used must be entered in the following order:

```
CHANGE [<repeat factor>] <string 1> [(<qualifier>)] <string 2> [Y]
```

The repeat factor is the number of instances of string 1 to be replaced, defaulting to 1. The qualifier, defines the limits of the search as with the FIND command. When "Y" is specified, the Editor will replace each matching string as it is encountered. If "Y" is not specified, the Editor will highlight each matching string found in reverse video. You are then prompted to enter "Y" to replace the string, "N" to veto the replacement, or "C" to cancel the CHANGE operation altogether. (See Figure 1.) If you enter "N", the next occurrence of the string will be shown. If you want to replace all instances of string 1, specify a repeat factor that exceeds the number of times string 1 appears in your file as in:

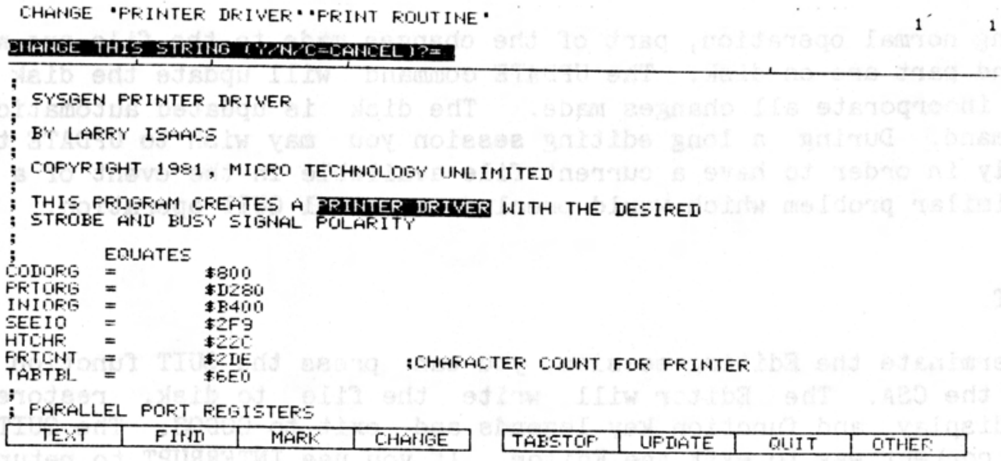
```
CHANGE 1000 " " ":" Y
```

which immediately changes all semicolons to colons (assuming that less than 1000 semicolons exist) without waiting for confirmation. The following example:

```
CHANGE "HIS" (+100) "HIS/HER"
```

changes one occurrence of "HIS" to "HIS/HER", provided "HIS" is found within 100 lines of the present cursor location.

Figure 1



```
CHANGE 'PRINTER DRIVER''PRINT ROUTINE'
CHANGE THIS STRING (Y/N/C=CANCEL)?=
;
; SYSGEN PRINTER DRIVER
;
; BY LARRY ISAACS
;
; COPYRIGHT 1981, MICRO TECHNOLOGY UNLIMITED
;
; THIS PROGRAM CREATES A PRINTER DRIVER WITH THE DESIRED
; STROBE AND BUSY SIGNAL POLARITY
;
;        EQUATES
CODORG   =          $800
PRTORG   =          $D280
INIORG   =          $B400
SEEIO    =          $2F9
HTCHR    =          $22C
PRTCNT   =          $2DE          ;CHARACTER COUNT FOR PRINTER
TABTBL   =          $6E0
;
; PARALLEL PORT REGISTERS
```

5.4 ADDFILE

ADDFILE appends any file that already exist to the end of the file you are editing. After you depress ADDFILE, give the name of the file you wish to append and press RETURN. The file will be added on to the end of the file you are editing regardless of the cursor's present location. For example:

```
ADDFILE MYTEXT.T
```

5.5 INSFILE

INSFILE allows you to insert a file that already exists into the file you are editing. Move the cursor to the line below where you want the insertion made. The entire file will be inserted directly above the line the cursor is on. After you depress INSFILE, give the name of the file you wish to insert and press RETURN. For instance:

```
INSFILE SUBS.A:1
```

5.6 TABSTOPS

TABSTOPS makes it possible for you to replace the currently set tabstops with new ones. Depress TABSTOPS, specify the column numbers you want tabstops placed in and press RETURN. The tabstops must be specified in ascending order. You can set up to 31 tabstops. For example:

```
TABSTOPS 10 15 20
```

The "tic" marks on the topside of the horizontal line splitting the CSA from the text window indicate the current tabstop positions. In Landscape Mode, the tic marks for tabstops set in columns 1 through 80 appear on the topside of the horizontal line. The tic marks for tabstops set in the second display line are on the underside of the horizontal line. As tabstops are updated, the tic marks are also updated. The TAB key moves the cursor to each tabstop.

When the Editor encounters a tab character in a file, it automatically replaces it with the number of blank spaces required to reach the next tabstop.

The tabstops you see upon entering the Editor are determined by the values which exist in the global tabstop table in memory. CODOS initializes the tabstop table to place a tabstop every 8 columns during power-up. The global tabstop table resides permanently in memory at addresses $06E0 through $06FF. The tabstops may be changed prior to entering the Editor by using the CODOS SET command. See the CODOS section of this manual for an explanation of the CODOS SET command.

5.7 UPDATE

During normal operation, part of the changes made to the file are maintained in memory and part are on disk. The UPDATE command will update the disk copy of the file to incorporate all changes made. The disk is updated automatically by the QUIT command. During a long editing session you may wish to UPDATE the file occasionally in order to have a current file available in the event of a power failure or similar problem which would preclude a normal QUIT operation.

5.8 QUIT

To terminate the Editing session, you can press the QUIT function key or type QUIT in the CSA. The Editor will write the file to disk, restore the normal 24-line display and function key legends and exit to CODOS. The QUIT command is the only correct way to exit the Editor. If you use INTERRUPT to return to CODOS, the resulting file status is undefined and the display window will not be restored to it usual format.

5.9 MACRO

MACRO enables you to redefine a command or a string as a number so that each time you depress the ENTER key followed by the number, the Editor executes the command or enters the corresponding string. For example:

```
MACRO 2 'PROCEDURE'
```

means that if you depress the "ENTER" key, followed by a "2", you will see the word "PROCEDURE" at the cursor's present location.

You may have up to eight macros so the number argument assigned to a string must be between 1 and 8. If you assign a macro number already in use to another string, the second string replaces the first in defining the number. The string may be up to 31 characters long and must be enclosed in quotes.

The same technique works for defining a command as a macro:

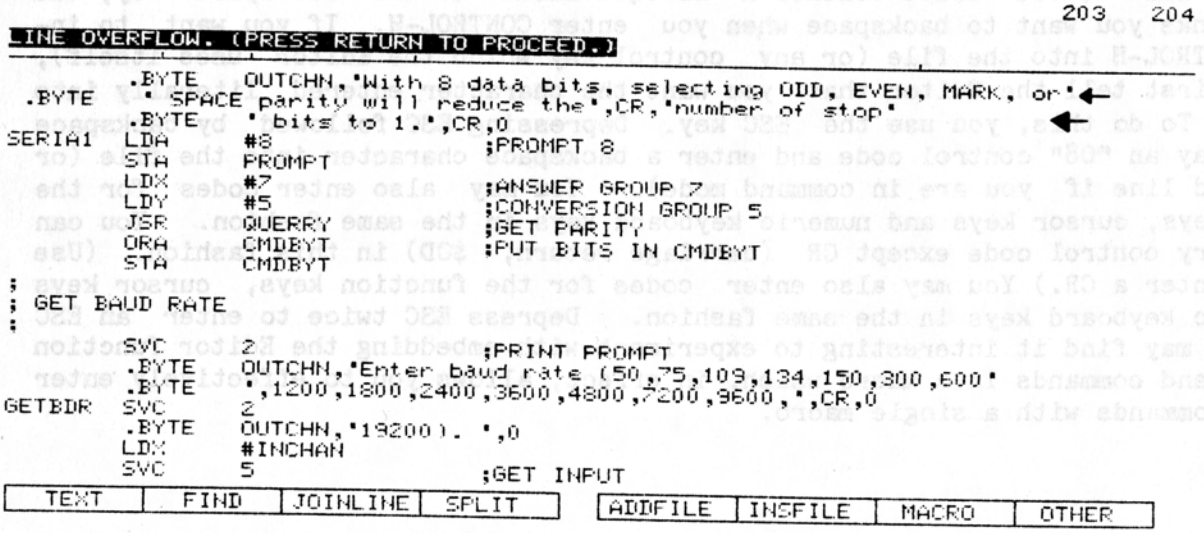
MACRO 6 "TABSTOP 4 8 12 0"

"ENTER" followed by a "6" resets the tabstops to 4, 8 and 12 respectively at the cursor's present location in the text window. In case you forget what one of your macros stands for, "ENTER" followed by the macro number on the command line will display the string or command that macro number defines. You may then cancel it with CNTRL-X.

The technique described in section 6 may be used in conjunction with the MACRO command to save both time and space.

5.10 JOINLINE

Joinline appends the line the cursor is on to the end of the previous line. If this causes a line overflow, the overflowing part of the line will not be appended but will remain on the next line and a line overflow error message will display. Figure 2 below shows an overflow condition that occurred while attempting to join lines 203 and 204.



```
LINE OVERFLOW. (PRESS RETURN TO PROCEED.)
         .BYTE    OUTCHN,'With 8 data bits, selecting ODD, EVEN, MARK, or'
.BYTE    ' SPACE parity will reduce the',CR,'number of stop'
         .BYTE    'bits to 1.',CR,0
SERIA1   LDA      #8                 ;PROMPT 8
         STA      PROMPT
         LDX      #7                 ;ANSWER GROUP 7
         LDY      #5                 ;CONVERSION GROUP 5
         JSR      QUERRY             ;GET PARITY
         ORA      CMDBYT             ;PUT BITS IN CMDBYT
         STA      CMDBYT
;
; GET BAUD RATE
;
         SVC      2                  ;PRINT PROMPT
         .BYTE    OUTCHN,'Enter baud rate (50,75,109,134,150,300,600'
         .BYTE    ',1200,1800,2400,3600,4800,7200,9600,',CR,0
GETBDR   SVC      2
         .BYTE    OUTCHN,'19200). ',0
         LDX      #INCHAN
         SVC      5                  ;GET INPUT
```

Figure 2

5.11 SPLIT

SPLIT divides a single line into two lines. The character the cursor is on begins the second line. After you have positioned the cursor on the character you want to start the second line with, depress SPLIT and then RETURN. The new line will appear directly below the original single line.

6. SPECIAL CHARACTERS

If you have other used other text editors, you may have encountered a common and frustrating problem involving embedded control characters. The root of this problem is the fact that the ASCII character set contains 32 non-displayable control characters which frequently become included in text files. Although you can't see them, the computer and the text editor treat them just like other characters. For example, you might see the word "the" on the screen, but a text editor refuses to acknowledge that "the" exists. This may be because an invisible control character is embedded between, for example, the "t" and the "h".

The MTU Text Editor provides a simple and effective solution to this problem. All control characters except CR (carriage return) and HT (tab) are actually displayed so you can see them. You can also edit these control characters just like any other character. The Editor permits you to enter all 128 possible ASCII character codes plus all special key codes (such as the cursor control key codes) directly into a text file. Experienced computer users will greatly appreciate this unusual feature.

To display ASCII control characters, a special character font is used which shows the ASCII code for the control character in hexadecimal. Since it takes two hex digits to represent a control code but each code is only one character, the hex value is displayed as a pair of small numbers turned "sideways" in the character cell. The ASCII codes for all characters are given in Appendix H of the CODOS manual.

The easiest way to see what these characters look like is to generate some. If you hold down the CONTROL key while typing "ABC", you will see the control codes 01, 02, and 03 displayed. You can generate most of the other control codes in the same fashion, except for those control codes which the Editor uses itself, such as backspace (BS = $08). Since CONTROL-H is equivalent to the backspace key, the Editor thinks you want to backspace when you enter CONTROL-H. If you want to insert a CONTROL-H into the file (or any control key which the Editor uses itself), you must first tell the Editor that you want the character entered literally into the line. To do this, you use the ESC key. Depressing ESC followed by backspace will display an "08" control code and enter a backspace character into the file (or the command line if you are in command mode). You may also enter codes for the function keys, cursor keys and numeric keyboard keys in the same fashion. You can enter every control code except CR (carriage return, $0D) in this fashion. (Use split to enter a CR.) You may also enter codes for the function keys, cursor keys and numeric keyboard keys in the same fashion. Depress ESC twice to enter an ESC code. You may find it interesting to experiment with embedding the Editor function key codes and commands in a macro which, in effect, allows you to effectively enter multiple commands with a single macro.

APPENDIX A

CONTROL KEY SUMMARY

BACKSPACE	Move cursor back one character.
CNTRL-B	Recall previous commands while in command mode.
CNTRL-X	Erase the current line to blanks. (Cursor repositions at column 1.)
Cursor-Down	Move cursor down one line in the same column. (Cursor stops at end of file.)
Cursor-Left	Move cursor back one character.
Cursor-Right	Move cursor forward one character.
Cursor-Up	Move cursor up one line in the same column.
DELETE	Delete the character the cursor is on.
ENTER	Implement a macro number (1-8) specified with the next key.
ESC	Enter next character literally into the line.
HOME	Move cursor to column 1 line 1 of the Text Window.
INSERT	Insert characters before the character the cursor is on.
RETURN	Advance the cursor to the next line or execute a command.
RUBOUT	Move the cursor back one character and replace it with a blank.
SHIFT-Cursor-Down	Advance Text Window 20 lines.
SHIFT-Cursor-Left	Move cursor to column 1 of current line.
SHIFT-Cursor-Right	Move cursor to right margin of current line.
SHIFT-Cursor-Up	Backup Text Window 20 lines.
SHIFT-DELETE	Delete the line the cursor is on.
SHIFT-HOME	Display the first 20 lines of the file with cursor at HOME.
SHIFT-INSERT	Insert an empty line above the line the cursor is on.
TAB	Advance the cursor to the next tabstop.

APPENDIX B

COMMAND/FUNCTION KEY SUMMARY

ADDFILE Append an existing file to the end of the file being edited.

CANCEL Unmark designated text and return to primary menu.

CHANGE Search out and replace specified string.

COMMAND Enter command mode and position cursor in CSA.

COPY Duplicate marked lines of text at present cursor position in text window.

DELETE Erases marked lines of text.

FIND Search out a specified line number or character string.

INSFILE Insert an existing file into the file being edited.

JOINLINE Append the line the cursor is on to the end of the previous line.

MACRO Redefine frequently used commands or strings as numbers for quick entry.

MARK Designate a line or block of text to be operated on.

MOVE Relocate marked text to present cursor position in text window.

OTHER Access or exit the secondary menu.

QUIT Update the file on disk and exit to CODOS

SPLIT Divide a single line into two lines.

TABSTOP Replace the currently set tabstops with new ones.

TEXT Enter text mode and position cursor in text window.

UPDATE Incorporate current edit into the file on disk.

WRITE Send marked text to a file or device.