1 The FULL SCREEN source text EDITOR for & 3 BASIC on the TRS-80* MODELS 1 and 3

by David Willen

COMPUTER APPLICATIONS UNLIMITED

Cat. #1210-20

TABLE OF CONTENTS

TABLE OF CONTENTS

I. INTRODUCTION

Congratulations on your purchase of CAU's FULL SCREEN EDITOR for BASIC. This unique product will greatly enhance your ability to create, modify, and manipulate BASIC programs. It is designed for use on the TRS-80 Model 1 & 3 and is compatible with both Model 1 Level 2, and Model 3 BASIC, as well as disk BASIC on both machines.

Once loaded and activated, a single command in BASIC "READY" mode will transfer control to the editor. Another simple command allows you to "exit" from the editor and return to BASIC "READY" mode. While the editor is in control you will enjoy the many advantages of full screen editing.

This manual describes all of the features and facilities of the editor. The editor tape includes a sample BASIC program. This program will be used in the examples that appear throughout the manual.

II. ENVIRONMENT

The largest BASIC program that can be edited is determined by the amount of memory that is in your system. The largest BASIC statement allowed is 240 characters long, excluding the statement's line number.

The editor will occupy the highest 5k addresses in your system. A relocating loader provided with the editor will allow you to load the editor into this area regardless of whether you have a 16k, 32k, or 48k system. If you require a keyord debounce or some othe machine language routine (i.e. specialized printer driver) then you must locate it below the editor (be sure to adjust your MEMORY SIZE accordingly).

In a 16k system the editor uses hex addresses 6C00 - 7FFF.
In a 32k system the editor uses hex addresses AC00 - BFFF.
In a 48k system the editor uses hex addresses EC00 - FFFF.

III. INVOKING THE EDITOR

1) Loading the editor from tape

On one side of the tape is the program for Model 1 and on the other side is the Model 3 version. Make sure the right side of the tape for your machine is fully rewound and facing up in the tape player. Then set the tape recorder to "play."

Turn on the computer (if you have disk drives installed you must hold down the BREAK key as you do this).

If you have a Model 3 then you must reply "L" to the "Cass?" question.

Reply to the MEMORY SIZE? question as follows:
If you have a 16k system then reply 27647.
If you have a 32k system then reply 44031.
If you have a 48k system then reply 60415.

Type in SYSTEM and hit the ENTER key.
Type in LOADER and hit the ENTER key.

As the relocating loader is read in an asterisk will flash slowly in the upper right hand corner of the screen. When the tape stops the computer will display *?
Leave the tape recorder set to "play."

Type in / and hit the ENTER key.
The relocating loader will display CONFIGURATION?

If you have a 16k system (small) hit the S key.
If you have a 32k system (medium) hit the M key.
If you have a 48k system (large) hit the L key.

The relocating loader will load in the editor from the tape. As it does so an asterisk will flash slowly in the upper right hand corner of the screen. If a C appears in this corner then the tape has not been read properly and the procedure must be repeated.

When the tape stops the messages

```
     LOAD AND RELOCATE COMPLETE.
     HIT ENTER TO ACTIVATE THE EDITOR.
```
will be displayed.

Hit ENTER and the editor's initialization frame will appear. The editor has been activated and can now be invoked at any time from BASIC "READY" mode. At this point it is necessary to hit the SPACE BAR. We also recommend that you type in NEW to make sure that BASIC is properly initialized.

2) Loading the Sample BASIC Program

Now that you are back in BASIC "READY" mode you can enter or load (from tape or disk) a program that you wish to work on. The editor tape itself is now positioned to the sample BASIC program. This program will be used in the examples that follow so if this is your first reading of this manual we suggest you load the program in:

Make sure the tape recoarder is still set to "play." Type in CLOAD and hit the ENTER key. When the sample program is loaded you can LIST and RUN it. It is a simple program which draws a border around the screen and then flashes the number "1" once, "2" twice, and so on up to "6". When the sample program completes you must hit the BREAK key because it goes into an "infinite loop" at statement 400.

Appendix D contains a listing of this program which you may wish to refer to from time to time as you follow the examples that appear throughout this manual.

IV. INTRODUCTION TO FULL SCREEN EDITING

You are now in what is probably a very common position. There is a program in your computer which has just been RUN. You would like to look at it and possibly also change it. Up to now you have had to rely on the BASIC commands LIST and EDIT to accomplish these tasks. However, now that the full screen editor is active you have a new alternative. Type in XEDIT and hit the ENTER key.

The full screen editor is now in control and is displaying the beginning of your resident BASIC program. Notice that the display format is different from that produced by the LIST command. The five leftmost columns of each screen line are reserved for statement line numbers. Statement line numbers are always displayed as five digits. When a statement is longer than a screen line then it is continued on the next screen line starting in column 7. The first statement in the sample program is an example of this.

Notice also that no statement is allowed to start below the 12th line on the screen. This insures that the last statement displayed always has room to expand to its maximum length. The maximum length of any statement is 240 characters (excluding the line number). When a statement is this long it will occupy five screen lines.

You have probably also observed that there is a blinking cursor at the beginning of the first statement on the screen. The editor is in Blinking Cursor Mode.

V. BLINKING CURSOR MODE

1) Changing a Character or string of characters

In order to change a character you must move the cursor to the character that you wish to change and type in the new character desired. The display will immediately reflect the change and the cursor will advance to the next character in the statement. Therefore a string of characters can be easily modified and verified for any editing errors. Examples will be given after you are shown how to move the cursor.

2) Moving the Cursor

The following commands are auto-repeating. This means that as long as you hold down the command key the function of the key will be repeated. The cursor cannot be moved to the statement line number on the display. It is also not possible to move the cursor beyond the last character in any statement while in the Blinking Cursor Mode.

a) → (right arrow)	- moves the cursor one column to the right.
b) ← (left arrow)	- moves the cursor one column to the left.
c) SHIFT →	- moves the cursor to the next tab position on the statement line. Tab positions are at every eighth character. This is a "high speed" forward space.
d) SHIFT ←	- moves the cursor back to the beginning of the statement line. This is a "high speed" backspace.
e) ENTER	- moves the cursor to the beginning of the next statement in the program. This function is ignored if the cursor is already on the last statement of the program.
f) SHIFT ↑	- moves the cursor to the beginning of the previous statement in the program. This function is ignored if the cursor is already on the first statement of the program.

Examples

In the first statement of the sample program the word PROGRAM is mispelled as PROGRAN. To correct this error depress the → key until the cursor has moved under the N and type an M. You have corrected the mistake.

Let us assume that we wish to chage the word BORDER in statement 40 to FRINGE. Depress the ENTER key until the cursor has moved down to statement 40. Then depress the → key until the cursor has moved under the B and type in FRINGE. You have made the change.

You now know how to move the cursor to any statement on the display and to any character within a statement. Remember that SHIFT → and SHIFT ← allow you to rapidly position the cursor anywhere on a long statement line. Any character that the cursor is on can be changed by simply typing in the new character desired.

3) Deleting Characters - SHIFT↓D (Model 3)
SHIFT D (Model 1)

Move the cursor to the character that you wish to delete. Hit SHIFT.D and the character will be deleted. The rest of the characters in the statement will be shifted one character to the left to fill the gap that was left when the character was deleted.

This function auto-repeats so that a string of characters can be deleted by keeping SHIFT.D depressed. This function can only be invoked when the editor is in Blinking Cursor Mode.

Example

Let us assume that we wish to shorten the comment in statement 70 from NOW FLASH NUMBERS INSIDE THE BORDER
to NOW FLASH NUMBERS INSIDE BORDER.
To accomplish this move the cursor until it is under the T in the word THE and depress SHIFT↓D until all three characters in the word are deleted.

VI. INSERT MODE

1) Insert Characters within a statement - SHIFT↓B (Model 3)
SHIFT I (Model 1)

Move the cursor to the character that you wish to insert characters before and hit SHIFT↓B. The editor will now be in the Insert Mode and the cursor will no longer be blinking. A blank will appear where you will be inserting characters. You can now type in all of the characters that you wish to insert. As they are typed in the rest of the characters in the statement will be shifted to the right.

When you have finished inserting characters at the present position hit the ENTER key. The extra blank will be deleted and the editor will return to Blinking Cursor Mode.

Example

Suppose you want to modify statement 30 so that it sets variable Q equal to 5 after the CLS but before the A=1. Move the cursor until it is under the A in statement 30 and hit SHIFT↓B. Notice that the editor is now in Insert Mode as indicated by the non-blinking cursor. Type in Q=5: and hit ENTER. Notice that the editor is back in Blinking Cursor Mode.

2) Backspace in Insert Mode

It is important to realice that while in Insert Mode the cursor control keys described in Sec. V.2 do not operate. Two of these keys do have alternate functions, however. One of these is the ENTER key, which we have already discussed. It is used to "exit" from Insert Mode back into Blinking Cursor Mode.

The other control key which has an alternate function in Insert Mode is the ← (left arrow) key. When used in Insert Mode it causes the cursor to be backspaced to the previous character position. The character that occupied this position is deleted and the editor remains in Insert Mode. This allows you to easily correct any mistakes you might make while typing in characters in Insert Mode.

3) Extending a Statament - SHIFT↓E (Model 3)
SHIFT X (Model 1)

Hit SHIFT↓E and the editor will go into Insert Mode with the cursor at the end of the statement it was on. This function must be invoked in Blinking Cursor Mode. Remember that <u>regardless</u> of where the cursor is on the statement when this function is invoked, the cursor will go to the end of the statement.

This function is useful when you wish to extend a BASIC statement with additional characters.

Example

Suppose you wish to extend statement 80 with a comment explaining what the GOSUB 500 does. Move the cursor so that it is anywhere on statement 80 and hit SHIFT↓E. Notice that the editor is now in Insert Mode (the cursor has stopped blinking) and that the cursor is at the end of statement 80. Now type in :REM CALL DELAY SUBROUTINE and hit the ENTER key. Notice that the editor is back in Blinking Cursor Mode and that the statement has been extended.

4) Macro Key Facility

The macro key facility is available whenever the editor is in Insert Mode. Macro keys are a kind of "shorthand" that allow you to insert often used BASIC keywords or other character strings with a single keystroke. There are 26 macro keys available and they are the alphabetic keys (A - Z) when struck in conjunction with the SHIFT and @ keys (i.e.: SHIFT@A, SHIFT@B, SHIFT@C, etc.). On the Model 1, the macro keys may be invoked with just the SHIFT key, the @ key is not needed.

Example

Let us suppose we want to modify statement 30 so that it prints out the value of B immedately after setting B equal to 2. Move the cursor up to statement 30 and then move it over until it is under the C in "...C=3...". Now hit SHIFT↓B. The editor is now in Insert Mode. Instead of typing in PRINT hit SHIFT@P. Notice that the word PRINT has been inserted! Now complete the statement by typing in B: and then hit ENTER. You have accomplished the change and the macro key replaced five keystorkes with one keystroke.

When the editor is initially loaded the 26 macro keys are set to provide the "default" keywords listed in Appendix C. You can, however, redefine any or all of these macro keys by using an extended command which will be described later.

VII. SCROLLING THE DISPLAY

1) Scrolling by Lines

The editor can display 12 BASIC statements at most at any time. Since programs are often much larger than this we must be able to position the display to any portion of the program being edited. This capabilty is called "scrolling." When the editor is in Blinking Cursor Mode you can control scrolling with the cursor control keys already described.

Example

The editor is currently displaying statements 10 thru 100 of the resident BASIC program. Let us assume that we wish to view and possibly modify the next statement after statement 100. Depress the ENTER key until the cursor is on statement 100. Now hit the ENTER key once more. Notice that the function of the ENTER key has been fulfilled: the cursor is now at the begining of the next statement, statement 110.

In order to bring statement 110 into view on the screen, however, the display was "scrolled upward" so that the top statement (statement 10) is no longer on the screen.
The ENTER key is auto-repeating even when it causes scrolling to occur. To observe this, depress the ENTER key for a few monments. You will be able to view successive segments of the program as it scrolls up the display. Release the ENTER key and the scrolling process stops.

You can also scroll the display in the other direction, to bring into view statements that precede the top statement on the screen. Depress SHIFT ↑ until the cursor reaches the top statement on the screen. Now hit SHIFT ↑ once more. Notice that the display has "scrolled downward" to bring into view the preceding statement. This function is also auto-repeating. To demonstrate this, depress SHIFT ↑ until the display has scrolled all the way back up to the beginning of the program (statement 10).

2) Scrolling by Pages

The functions just described allow scrolling up or down by single statements at a time. It is also possible to scroll downward by an entire screen of statements (called a "page") at a time. The control key which performs this function is SHIFT↓Z. On some Model 1 machines, just use SHIFT↓.

Example

If you have been following the examples the editor should now be displaying the beginning of the program, statements 10 thru 100. Let us assume that we have made all the editing changes we desired on this "page" and we now want to go on to the next part of the program. Hit SHIFT↓Z. Notice that the editor is now displaying statements 110 thru 220.

this function is also auto-repeating so you can "page thru" a long program by holding down SHIFT↓Z.

You now know how to scroll the display so that you may view and edit any portion of the resident BASIC program. This scrolling capability is sequential which means that even though you can go forward and backward, you must step through each program statement in the order in which they appear. Later on in this manual you will be shown how to use extened commands to scroll the display immediately to any statement in the program that you select.

VIII. CLEAR COMMAND MODE

"Clear" commands apply to a specific statement or block of statements within the entire program instead of to specific characters within a statement. They are called "clear" commands because the CLEAR key is always used to invoke them. The following procedure is used to enter Clear Command Mode:

- Scroll the display and/or move the cursor so that the cursor is on the statement that you wish to deal with.
- Hit the CLEAR key. The statement will be highlighted to the right of the line number field. The cursor will no longer be visible. This indicates that the editor is in Clear Command Mode.
- To exit from Clear Command Mode at this point (for example, if you changed you mind or selected the wrong statement) simply hit the CLEAR key again. The statement will be un-highlighted, the cursor will return to its original position, and the editor wil be back in Blinking Cursor Mode.
- If the CLEAR key is not hit a second time as just described then the next key that is hit will determine the function to be performed.

The choices are:

1) Delete Statement - D

Hit D in Clear Command Mode and the highlighted statement will be deleted from the program. The editor will then return to Blinking Cursor Mode.

Example

Asssume that we wish to delete statement 150. Move the cursor until it is on statement 150 (it can be positioned anywere on the statement). Now hit the CLEAR key. Notice the highlighting to the right of the line number 150. This indicates which statement you have selected. Now hit the D key. Notice that statement 150 has been deleted from the program and that the editor is back in Blinking Cursor Mode.

2) Insert Statement - I

Hit I in Clear Command Mode and a new statement will be inserted into the program at the point immediately following the selected statement. The cursor will be placed at the beginning of this new statement and will not be blinking. This indicates, of course, that the editor is now in Insert Mode. You can, therefore, type in the desired contents of the new statement, using the ← key to correct mistakes (Insert Mode "backspace"). The macro key facility is also availale.

When the new statement has been completely typed in, you can hit the ENTER key and the process will be repeated: another new statement will be inserted after the one just completed. The cursor will be placed at the beginning of this new statement and the editor will again be in Insert Mode.

In this fashion as many statements as are needed can be inserted into any point in the program. When all desired statements have been inserted you can hit the CLEAR key and the editor will go back into Blinking Cursor Mode.

Example

Let us assume that we wish to insert three statements that print the values of A, B, and C respectively after statement 400. Scroll the display and move the cursor until it is on statement 400 Now hit the CLEAR key. Notice that line number 400 has been highlighted to indicate that you have selected that statement. Now hit the I key. You will observe that a new statement, numbered 405, has appeared. You will also observe that the cursor is not blinking and therefore the editor is in Insert Mode. Now type in the first new statement PRINT A and then hit the ENTER key. The next new statement, numbered 410, has appeared. Let us be a bit more sophisticated this time and enter the second new statement as follows: hit SHIFT@P and then B. Now hit the ENTER key. Similarily, we can now enter the third new statement, numbered 415, as follows: hit SHIFT@P and then hit C. Since at this point you do not wish to insert any additional statements, hit the CLEAR key. The editor will go back into Blinking Cursor Mode.

You have probably noticed that, in the above example, each new statement was assigned a line number that was 5 greater than the preceding line number. This is because the editor's <u>Increment Value</u> is set at 5 when it is initially loaded. Later on in this manual you will be shown how to set the editor's <u>Increment Value</u> to any value you desire.

3) Copy Statement - C

Hit C in Clear Command Mode and the highlighted statement will be marked for copying. The editor will return to Blinking Cursor Mode. See the H "clear" command for how to complete this function. The statement will remain highlighted while the copy is pending.

4) Move Statement - M

Hit M in Clear Command Mode and the highlighted statement will be marked for moving. The editor will return to Blinking Cursor Mode. See the H "clear" command for how to complete this function.

The statement will remain highlighted while the move is pending.

5) Move/Copy Statement to Here - H

After you have marked a statement for move or copy as described above, the editor returns to Blinking Cursor Mode. You can therefore move the cursor anywhere you wish, make character changes, scroll the display, and in general do anything except mark another statement (or block of statements, see section on blocks) for a move, copy, delete, or insert.

To complete the pending move or copy, move the cursor to another statement in the program and hit the CLEAR key to enter Clear Command Mode. Then hit the H key and the statement marked for move or copy will be moved/copied to immediately after the statement you have just selected. The editor will then return to Blinking Cursor Mode.

<u>Example</u>

Suppose you want to copy REM statement 270 to after statement 290 so that the comment appears in both places in the program.

Move the cursor to statement 270. Hit the CLEAR key and then the C key. Notice that the editor has returned to Blinking Cursor Mode, but statement 270 remains highlighted. The copy is now "pending." Now move the cursor to statement 290, hit the CLEAR key and then hit the H key. Notice that a new statement, numbered 295, has been inserted into the program and contains the same comment as statement 270. Statement 270 has therefore been copied to after statement 290.

In the above example the new statement was given a line number of 295 which is 5 greater than the number of the statement it was inserted after. This was determined, as is for the Insert Statement function, by the editor's <u>Increment Value</u>. If, for example, you had set the editor's <u>Increment Value</u> to 1 prior to requesting the copy, then the new statement would have received a line number of 291. You will be shown how to set the editor's <u>Increment Value</u> later.

The difference between a move and a copy is that when a statement is copied it is actually reproduced at another point in the program but when a statement is moved it is deleted from its original location and therefore only appears at the new location selected.

The move function can therefore be thought of as a "renumbering" of the statement that was moved.

Whenever the editor performs such "renumbering" it automatically searches the entire program for all references to the statement that is being moved. Such references can appear after the BASIC keywords GOTO, GOSUB, THEN, ELSE, and RESUME. If the editor finds such a reference then it is changed to reflect the moved statement's new location.

<u>Example</u>

Let us assume that we wish to move statement 500 to after REM statement 510 which describes it. Notice that statement 500 is a subroutine which is "called" from twelve other statements in the program. Move the cursor to statement 500 and hit the CLEAR key and then the M key. The statement has been marked to be moved. Now move the cursor to statement 510 and hit the CLEAR key and then hit the H key. Statement 500 has been moved to after statement 510 and therefore now has the new number 515. Notice that all of the "GOSUB 500"s in the program have been changed and now read "GOSUB 515"!

6) Blocks of statements - B

A block of statements can be deleted, copied, or moved just as easily as a single statement. The procedure is exactly the same as for a single statement except that a block of statements is selected.

To select the beginning of a block, move the cursor to the statement that starts the block and enter Clear Command Mode by hitting the CLEAR key. Then hit the B key to mark the statement as a block terminator. The statement will remain highlighted and the editor will return to Blinking Cursor Mode. The next "clear" command (delete, copy, or move) that you use will apply to the entire block as opposed to just the statement that you entered it on. This next "clear" command can be entered at any other statement in the program so you can scroll the display (either up or down) to select the "other end" of the block. The block is defined as all of the statements between and including the statement that you marked as a block terminator and the statement that you select for delete, copy, or move.

Example

Suppose we want to delete the block of three statements that we inserted erlier, statements 405, 410, and 415. Move the cursor until it is on statement 405. Hit the CLEAR key and then the B key. Statement 405 is now marked as a block terminator. Notice that it remains highlighted. Now move the cursor to statement 415. Hit the CLEAR key and then the D key. The block of statements from 405 through 415 has been deleted.

Note that you could have selected statement 415 first as the block terminator and then moved the cursor up to statement 405 to select Delete. Either approach would have yielded the same result.

You may wish to refer to Appendix D while following the next example.

Now suppose we want to move the block of statements numbered 270 through 300 to the end of the program. Move the cursor until it is on statement 270. Hit the CLEAR key and then hit the B key. Statement 270 has been marked as a block terminator. Now move the cursor to statement 300. Hit the CLEAR key and then the M key. The block of statements numbered 270 through 300 has been selected to be moved. Notice that the statements at each end of the block are highlighted. Now move the cursor to statement 600. Hit the CLEAR key and then the H key.

The block of statements numbered 270 through 300 will be moved to after statement 600 and will therefore be renumbered as statements 605 through 620. Notice that the display has been scrolled so that the first moved statement (statement 605) is at the top of the screen. This scrolling is automatic whenever a block of lines is moved. Now use SHIFT . to scroll back to statement 260. Notice that the moved statements no longer exist at their old location between statements 260 and 400. The editor has also changed all references in the program to statements 270 through 300 to reflect their new locations, numbered 605 through 620. The only reference of this kind in the entire program was within the moved block itself! Statement 300 used to read:

```
300  F=F-1 : IF F>0 THEN 280 ' FLASH IT "F" TIMES
     AND THEN GO ON
```

Notice that in its new location it now reads:

```
620  F=F-1 : IF F>0 THEN 610 ' FLASH IT "F" TIMES
     AND THEN GO ON
```

The above example illustrates a feature of the editor which is called SELECTIVE RENUMBER because it allows you to do just that. You can selectively renumber any group of statements in your program without destroying the program logic. You accomplish this by performing a block move. The maximum number of statements that can be block moved at a time is 50.

7) Cancel Pending Command - ENTER

While there is a pending block, copy, or move command a graphics block will appear in the lower right hand corner of the display once the display is scrolled. This will remind you that you have such a command pending in case you scroll the highlighted statement (s) off the screen.

If you have such a command pending and change your mind about completing it or discover that you have selected the wrong statement, then you can cancel the pending command. This is done as follows: enter Clear Command Mode by hitting the CLEAR key (the cursor can be on any statement). Then hit the ENTER key. This CLEAR key - ENTER key sequence will cancel all pending commands.

IX. MORE CLEAR COMMANDS

The following commands are initiated exactly like "clear" commands. However, since these commands do not apply to any specific statement, you can initiate them by hitting the CLEAR key without regard to where the cursor is.

1) Scroll to Top - T

   Hit the CLEAR key to enter Clear Command Mode and then hit the T key. The display will be scrolled to the top of the program. Getting to the beginning of a very long program is faster this way than it would be if you scrolled up a statement at a time using SHIFT ↑. You can try out this function now with the sample program.

2) Exit from Editor - E

   Hit the CLEAR key to enter Clear Command Mode and then hit the E key. This will cause an "exit" from the editor. Your computer will return to BASIC "READY" mode. BASIC will print a syntax error message which you can ignore. This is simply BASIC's way of responding to the XEDIT command that you used to invoke the editor.

   You should use this exit function whenever you have finished editing your program and wish to RUN, LIST, LLIST, SAVE, or CSAVE it. Remember you can always return to the editor by simply entering the command XEDIT while in BASIC "READY" mode.

3) Repeat Find - F

   Hit the CLEAR key to enter Clear Command Mode and then hit the F key. This will cause the previously entered FIND command to be repeated. The FIND command will be discussed in the next chapter.

4) Enter Extended Command Mode - SPACE BAR

   Hit the CLEAR key to enter the Clear Command Mode and then hit the SPACE BAR. The editor will now be in Extended Command Mode and a command prompt line will appear at the bottom of the display. An extended command can now be entered on this line (which reads "ENTER COMMAND ======>").

X. EXTENDED COMMAND MODE

As discussed above, when the CLEAR key - SPACE BAR sequence is used the bottom line of the screen is cleared so that you can enter an extended command. If you change your mind and do not wish to enter an extended command then hit the ENTER key while the command line is empty. The editor will return to Blinking Cursor Mode.

The commands discussed in this chapter assume that you have already placed the editor in Extended Command Mode. The editor will execute the extended command that you have typed in when you hit the ENTER key. After it has been executed the editor will always return to Blinking Cursor Mode. The ← key will backspace to allow you to correct any mistakes made while entering the extended command.

The first character of all extended commands is a verb which determines the function to be performed. It is always followed by one or more operands. No blanks are allowed between the verb and its operands except in the cases of FIND and CHANGE where any blanks will be considered part of the operand itself.

1) Set Increment Value - Inumber

This is the command that allows you to change the editor's <u>Increment Value</u> to a number other than the default of 5. Type in I followed immediately (no spaces allowed) by the number that you want the editor to use as its <u>Increment Value</u>. This new value will remain in effect until you change it again.

<u>Example</u>

Suppose we wish to insert another comment after REM statement 75. Because a statement numbered 80 already exists, we cannot perform an insert with the <u>Increment Value</u> set to 5. Let us therefore change the <u>Increment Value</u> to 2. Hit the CLEAR key and then hit the SPACE BAR. Notice that the editor is now in Extended Command Mode as indicated by the message "ENTER COMMAND ====>" which appears at the bottom of the screen. Now type in I2 and hit the ENTER key. Notice that the editor has returned to the Blinking Cursor Mode. The <u>Increment Value</u> has been set to 2. Now move the cursor until it is on statement 75. Hit the CLEAR key and then hit the I key. Notice that a new statement has been inserted between statements 75 and 80. The new statement is numbered 77. Now type in the contents of the new statement as follows: REM A NEW COMMENT and hit the CLEAR key. The Insert has been accomplished.

2) Search for Statement - Snumber

This useful command lets you scroll the display at once to any point in your program. Type in S followed immediately by the number of the statement that you wish to scroll to. The editor will scroll the display so that the statement with the number you selected is at the top of the screen. if your program does not contain a statement with the number you selected then the message "NOT FOUND" will appear for a moment and the display will not be scrolled.

Try this command now on the sample program. Notice that it allows you to quickly position the program to any for viewing and/or editing. Try it again as S999 and observe the "NOT FOUND" message. There is, of course, no statement number 999 in the sample program.

3) Find a Character String - Fstring

This command lets you search through your program for a specific string of characters. Type in F followed immediately by the character string desired. Evey character typed in before you hit ENTER is significant (this includes BLANKS). The search begins at the statement currently at the top of the display and continues until the end of the program is reached. If the string is found then the display is scrolled so that the statement containing the string is at the top of the display. If the string is not found then the message "NOT FOUND" will appear for a moment and the display will not be scrolled.

Remember that a FIND command can be easily repeated by using the F "clear" command. This means that after you have executed a FIND command you can search for subsequent occurrences of the same string by simply hitting the CLEAR key and then the F key.

Example

Suppose we wish to identify every point in the sample program where the variable C is modified. We can do this by searching for the character string "C=". Hit the CLEAR key and then the T key. This scrolls the display to the top of the program and insures that our search will include the entire program. Now hit the CLEAR key and then the SPACE BAR. The editor is now in the Extended Command Mode. Type in the command FC= and hit the ENTER key.

The display will scroll so that statement 30 is at the top of the screen. Statement 30 was the first statement found which contains the string "C=". Now hit the CLEAR key and then the F key. The display will scroll to statement 180 which contains the next occurrence of the same string. Again hit the CLEAR key and then the F key. The message "NOT FOUND" will appear briefly because the string does not appear anywhere else in the program.

4) Change Character String - Cstring

When this command is entered the program is searched (starting with the statement presently at the top of the screen) for the string specified in the previously entered FIND command. Each occurrence of the string is changed to the character string specified in this CHANGE command. When entering the CHANGE command remember that each character you type in is significant (including blanks) up until you hit the ENTER key. When this command is completed the editor returns, as always, to Blinking Cursor Mode. The display is not scrolled.

Note that because the FIND and CHANGE character strings do not have to be of equal length it is possible for statements to grow or shrink in lenght when they are affected by the CHANGE command.

Example

Suppose we wish to change all PRINT statements in the program to LPRINT statements. Hit the CLEAR key and then the T key. This scrolls the display to the top of the program and insures that we will make the change throughout the entire program. Hit the CLEAR key and then the SPACE BAR. The editor is now in Extended Command Mode. Type in the command FPRINT and hit the ENTER key. The display will scroll to the first occurrence of "PRINT" in the program. Now hit the CLEAR key and SPACE BAR to enter Extended Command Mode again. Type in the command CLPRINT and hit ENTER. Notice that each occurrence of the string PRINT has been changed to LPRINT in the program!

Remember that all blanks entered after the command verbs F or C are considered part of the FIND or CHANGE string. Blanks embedded within the FIND or CHANGE string are also permited. The maximum length of FIND/CHANGE strings is 20 characters.

5) Define Macro Key - Dkeystring

This command lets you specify the character string to be inserted when you use a macro key. If you are working on a program which uses a certain keyword or string of characters often then you may wish to assign that string to a macro key. Once this is done you can use that macro key whenever the editor is in Insert Mode to rapidly insert the keyword or string into your program (see Sec. VI.4).

The DEFINE MACRO KEY command is entered as follows: type in D followed immediately by the letter (A - Z) of the macro key you wish to define, followed immediately by a string of characters that you wish the macro key to represent.

Example

Assume that we want to modify the sample program and intend to make extensive use of the READ keyword. We may wish to define the macro key SHIFT@R to supply this keyword. Hit the CLEAR key then the SPACE BAR to enter Extended Command Mode. Type in DRREAD and hit ENTER. The macro key SHIFT@R has been redefined. To illustrate this, move the cursor until it is on statement 10. Hit the CLEAR key and then the I key. The editor is now in Insert Mode and we are inserting a new statement. Now hit SHIFT@R and then type in A,B,C and then hit the CLEAR key. We have add a READA,B,C statement. Remember if you are using a Model 1 then just use the SHIFT key and the character to invoke the macro.

Note that once a macro key is defined using this command it will retain its new definition until you define it again. The default definitions for each macro key are specified in Appendix C.

The maximum number of characters that any macro key can represent is 6. While this is enough to represent most BASIC keywords you may have an often used string in your program which is longer. For example, suppose you are entering a program in which the word DIFFERENTIAL occurs often. You could pick two keys which are adjacent on the keybord and use each of them to represent part of this word. If you picked D and F keys you would enter the extended commands DDDIFFER and DFENTIAL. Now when the editor is in Insert Mode you can hold down the SHIFT@ keys and hit the D and F keys in succession to quickly insert the word DIFFERENTIAL.

6) Renumber - Nnumber,number

This powerful command allows you to renumber your entire program. To use it type in N followed immediately by the number to be assigned to the first program statement, followed by a comma and then the number to be used as an increment in calculating all subsequent statement numbers. For example, the extended command N10,10 will renumber the entire program with new statement numbers 10,20,30, 40,... etc. The extended command N100,5 will renumber the entire program with new statement numbers 100, 105, 110, 115,... etc.

When the editor performs this renumber function it also changes all statement number references in the program to reflect the new numbers. Statement number references can appear after the BASIC keywords GOTO, GOSUB, THEN, ELSE, and RESUME.

If the changing of such references would cause any statement to exceed 240 characters in length, or would cause the entire program to exceed the available memory then the editor will display the message "RENUMBER ERROR" for a moment (Model 1 will display the message "NO ROOM TO PERFORM RENUMBER"). The renumber function will not be performed in this case. This restriction also applies to the SELECTIVE RENUMBER function described earlier.

If the operands specified in the renumber command would cause a statement to be numbered greater than 65535 then the editor will display the message "LINE# ERROR" for a moment and the renumber function will not be performed. The Model 1 message will be "NO ROOM BETWEEN LINE NUMBERS".

XI. ERROR MESSAGES AND SPECIAL CONSIDERATIONS

1) Conflict with BASIC Abbreviations

BASIC normally allows you to abbreviate the keyword PRINT with the ? symbol. This will not work while using the full screen editor. Don't forget that you can always use the macro keys to quickly insert PRINT or any other keyword while in Insert Mode.

The BASIC exponentiation operator is represented by the ↑ key and this is not changed when using the full screen editor. BASIC also allows the use of the ↓ key to represent a "line feed."

This is also possible while using the editor although the representation is slightly different. When entering a statement in "regular" BASIC you can hit the ↓ key and the "line feed" function code will be inserted into the statement at that point. BASIC will also cause the "line feed" to occur so the next character you type in will appear at the beginning of the next line on the screen. When using the editor you can still use the ↓ key to represent the "line feed" but it will appear on the screen as a "\" (on a Model 1 it will appear as a "↓") character instead of actually causing a "line feed" to occur.

2) Conflict with "ON ERROR" Statement

The editor gains control of the computer when you type XEDIT by intercepting BASIC's syntax error routine. If your resident BASIC program contains an "ON ERROR..." statement then under certain conditions BASIC will transfer control to your "ON ERROR" routine instead of to the editor when you type in XEDIT. This problem can be avoided by typing in the BASIC command CLEAR (do not confuse this with the CLEAR key) before typing in XEDIT whenever it is desired to use the editor.

3) Running out of Memory

If your program grows so large that it exceeds the available memory then the editor will abort the current function and display the message "MEM FULL" for a moment. The editor will then return to Blinking Cursor Mode. On a Model 1 the message "MEMORY FULL" will appear.

At this point it is possible that space is being wasted by old run time and/or string variables. To make all possable memory available you should exit from the editor and enter the BASIC command CLEAR 0. Then re-enter the editor (using the command XEDIT) and continue to edit your program. Under certain conditions it is possible for this technique to provide you with significantly more available memory to work with.

4) Invalid Requests

In the event that you request a conflicting or non-existent function the editor will display the message "INVALID RQST" for moment and will return to Blinking Cursor Mode. Message for Model 1 is "INVALID REQUEST".

An example of this would be if you hit the CLEAR key and then the C key to mark a statement for copying, scroll around a bit, and then hit the CLEAR key and then the M key.

Since you cannot have both a move and a copy pending at the same time you will get the "INVALID RQST" message. The move command will be ignored since it was invalid and the copy command will remain pending.

5) Inserting Statements

The statement insert, copy, and move functions all insert new statements in your program. They all use the editor's Increment Value to calculate the new statement's numbers. When the calculated statement number does not fit between the existing statement numbers (as in an earlier example where we wanted to insert a statement between statement 75 and 80 while the editor's Increment Value was set to 5) the editor aborts the function and displays the message "LINE# ERROR" for a moment. The editor then returns to Blinking Cursor Mode. On Model 1 the message will be "NO ROOM BETWEEN LINE NUMBERS".

6) Creating New Programs

When you have finished working with a program and would like to work on a different one you should always use the BASIC command NEW. Never use the editor's block delete function to empty the computer's program memory.

You will notice that when the computer's program memory is empty (i.e.: there is no BASIC program in the machine) the editor will not take control when you type in XEDIT. If you wish to use the editor to input a new program "from scratch" you must type in the first statement while in "regular" BASIC mode. Once this is done you will be able to invoke the editor with the XEDIT command. You can then proceed to enter the rest of your program by using the statement insert function. You will find that the macro keys are most useful when using the editor in this fashion.

7) Null Statments

It is possible when using the character delete function to delete all of the characters from a statement. Since BASIC does not allow such "null" statements, the editor will convert them to END statements.

XII. SUMMARY OF COMMANDS AND MODES

For Model 1:

BLINKING CURSOR MODE

- Change character - type new character over blinking cursor.

- Delete character - SHIFT D, auto-repeats.

- Insert character - SHIFT I, editor enters Insert Mode.

- Extend statement - SHIFT X, editor enters Insert Mode with cursor at end of current statement.

- Cursor Control Keys : (all of these functions are auto-repeating)

- → - right one column.

- ← - left one column

- SHIFT → - right one tab position.

- SHIFT ← - back to beginning of current statement.

- ENTER - down to beginning of next statement.

- SHIFT ↑ - up to beginning of previous statement.

- SHIFT ↓ - down to beginning of next page. Some Model 1 will require that you also hold down the Z key.

INSERT MODE

- Add new characters by typing them over non-blinking cursor.

- Correct mistakes by "backspacing " with the ← key.

- Insert strings quickly with macro keys SHIFT A through SHIFT Z.

- Exit back to Blinking Cursor Mode by hitting the ENTER key.

CLEAR COMMAND MODE

To enter Clear Command Mode hit the CLEAR key while cursor in on desired statement. Statement will be highlighted. To exit without entering any command hit the CLEAR key again. Otherwise the next key determines the function as follows:

- D	- delete statement or block of statements.
- C	- copy statement or block of statements.
- M	- move statement of block of statements.
- H	- move or copy statement or block of statements to here.
- I	- insert statement. Editor enters Insert Mode but ENTER key causes another insert to occur, CLEAR key is used to return to Blinking Cursor Mode.
- ENTER	- cancel pending move, copy or block command.
- B	- mark block terminator statement.
- T	- scroll display to top of program.
- E	- exit from editor. (Returns to BASIC "READY" condition).
- F	- repeat previously entered FIND command.
- SPACE BAR	- enter Extended Command Mode.

EXTENDED COMMAND MODE

Type in extended command on command prompt line at bottom of screen. To exit with no command just hit ENTER key.

- Inumber	- set editor's Increment Value to number.
- Snumber	- scroll display to statement number.
- Fstring	- find character string. Search starts at top of current display.
- Cstring	- change previously specified find string to this string. Applies to rest of program starting with top of current display.
- Dkeystring	- define macro key to represent string.
- Nnumber1,number2	- renumber entire program. Assign number1 to first statement and use number2 as an increment thereafter.

For Model 3:

BLINKING CURSOR MODE

- Change character - type new character over blinking cursor.

- Delete character - SHIFT↓D, auto-repeats.

- Insert character - SHIFT↓B, editor enters Insert Mode.

- Extend statement - SHIFT↓E, editor enters Insert Mode with cursor at end of current statement.

- Cursor Control Keys : (all of these functions are auto-repeating)

- → - right one column.

- ← - left one column

- SHIFT → - right one tab position.

- SHIFT ← - back to beginning of current statement.

- ENTER - down to beginning of next statement.

- SHIFT ↑ - up to beginning of previous statement.

- SHIFT↓Z- down to beginning of next page.

INSERT MODE

- Add new characters by typing them over non-blinking cursor.

- Correct mistakes by "backspacing " with the ← key.

- Insert strings quickly with macro keys SHIFT@A through SHIFT@Z.

- Exit back to Blinking Cursor Mode by hitting the ENTER key.

CLEAR COMMAND MODE

To enter Clear Command Mode hit the CLEAR key while cursor in on desired statement. Statement will be highlighted. To exit without entering any command hit the CLEAR key again. Otherwise the next key determines the function as follows:

- D	- delete statement or block of statements.
- C	- copy statement or block of statements.
- M	- move statement of block of statements.
- H	- move or copy statement or block of statements to here.
- I	- insert statement. Editor enters Insert Mode but ENTER key key causes another insert to occur, CLEAR key is used to return to Blinking Cursor Mode.
- ENTER	- cancel pending move, copy or block command.
- B	- mark block terminator statement.
- T	- scroll display to top of program.
- E	- exit from editor. (Returns to BASIC "READY" condition).
- F	- repeat previously entered FIND command.
- SPACE BAR	- enter Extended Command Mode.

EXTENDED COMMAND MODE

Type in extended command on command prompt line at bottom of screen. To exit with no command just hit ENTER key.

- Inumber	- set editor's <u>Increment Value</u> to number.
- Snumber	- scroll display to statement number.
- Fstring	- find character string. Search starts at top of current display.
- Cstring	- change previously specified find string to this string. Applies to rest of program starting with top of current display.
- Dkeystring	- define macro key to represent string.
- Nnumber1,number2	- renumber entire program. Assign number1 to first statement and use number2 as an increment thereafter.

APPENDIX A. Custom Modification of the Editor

The editor has been designed so that it is possible to modify the speed at which the auto-repating functions will repeat. After you have used the editor for a while you may decide that you would like a faster or slower "action" on these repeating functions. This can be easily accomplished by changing the contents of two memory locations.

The recommended procedure is as follow:

- EXIT from the editor by using the E "clear" command.
- Use the BASIC commands

  POKE address1,value1 and
  POKE address2,value2 to modify the memory locations.

- Re-enter the editor by using the XEDIT command.

The addresses to change depend upon how much memory you have and are:

	16k	32k	48k
address1	28402	-20750	-4366
address2	28411	-20741	-4357

The editor is shipped with value1 = 48 and value2 = 32. Increasing these values will slow down the auto-repeat speed and decreasing them will speed up the auto-repeat speed.

Value1 controls the speed at which the →, ←, SHIFT→, and SHIFT↓D (on the Model 1 SHIFT D) keys repeat. Value2 controls the speed at which the ENTER, SHIFT↑, and SHIFT↓Z (some Model 1 just use SHIFT↓) keys repeat.

If, for example, you wish to slow down the speed with which the →, ←, SHIFT→, and SHIFT↓D keys repeat, and you have a 32k system, you might use the BASIC command POKE -20750,80.

APPENDIX B. Creating a Disk Resident Copy of the Editor

If you have a disk system you can create a disk "CMD" file containing the editor so that you don't have to load it in from tape each time you wish to use it. The following procedure should be used to create the disk "CMD" file:

For Model 1:

- Follow the instructions in Sec. III.1 for loading the editor from tape.
- When the messages LOAD AND RELOCATE COMPLETE. HIT ENTER TO ACTIVATE THE EDITOR. are displayed DO NOT hit ENTER.
- Instead, place a system diskette with at least 5 free granules into drive 0 and hit the RESET button at the back of the keyboard to bring up DOS.
- As soon as you get the DOS READY message type in the following command:

For a 32k system:

```
  DUMP XEDIT/CMD:0 (START=X'AC00',END=X'BFF0',TRA=X'ACF6')
```

For a 48k system:

```
  DUMP XEDIT/CMD:0 (START=X'EC00',END=X'FFF0',TRA=X'ECF6')
```

The editor is now resident on your diskette. To load the editor from DOS READY mode simply type in the command XEDIT and hit ENTER. The editor will be loaded and will display the message "BASIC EDITOR LOADED". You will then be returned to DOS READY mode. You now can load disk BASIC by typing in the command BASIC and then hitting the ENTER key. If you have a 32k system then reply 44031 to the "MEMORY SIZE" question. If you have a 48k system then reply 60415 to the "MEMORY SIZE" question. Once BASIC has initialized you must activate the editor prior to using it. This is done as follows:

- Type in SYSTEM and hit ENTER.
- If you have a 32k system type in /44032 and hit ENTER.
- If you have a 48k system type in /60416 and hit ENTER.

This will invoke the ediotr's initialization frame. You can now proceed in the manner described in chapter III.

For Model 3:

- Follow the instructions in Sec. III.1 for loading the editor from tape.
- When the messages LOAD AND RELOCATE COMPLETE. HIT ENTER TO ACTIVATE THE EDITOR. are displayed <u>DO NOT</u> hit ENTER.
- Instead place a system diskette with at least 7 free granules into drive 0 and hit the RESET button to bring up DOS.
- As soon as you get the DOS READY message type in the following command:

For a 32k system:

```
DUMP XEDIT (START=0AC00,END=0BFF0,TRA=0ACF7)
```

For a 48k system:

```
DUMP XEDIT (START=0EC00,END=0FFF0,TRA=0ECF7)
```

- Type in BASIC and reply to the "Files" and "MEMORY SIZE" questions by hitting ENTER. Then type in the following program:

  for 32k systems:

```
10 DEFUSR1=&HAC00
20 X=USR1(0)
30 NEW
```

  for 48k systems:

```
10 DEFUSR1=&HEC00
20 X=USR1(0)
30 NEW
```

- Type in the command SAVE "XEDIT/BAS:0" to save this program.

The editor is now resident on your diskette. To load the editor from DOS READY mode simply type in the command XEDIT and hit ENTER. The editor will be loaded and will display the message "BASIC EDITOR LOADED". You will then be returned to DOS READY mode. You now can load disk BASIC by typing in the command BASIC and then hitting the ENTER key. Reply as you wish to the "Files" question. If you have a 32k system then reply 44031 to the "MEMORY SIZE" question. If you have a 48k system then reply 60415 to the "MEMORY SIZE" question. Finally, type in the command RUN"XEDIT/BAS" to activate the editor.

Notice to Model 3 Users Only

If you encounter any problems with the disk copy of the editor which you made from tape, please try this alternate procedure for moving the editor to disk.

ALTERNATE PROCEDURE:

- Hold down the BREAK key and turn on the TRS-80. Reply "L" to the "Cass?" question.

- If you have 32k RAM then reply 44031 for "Memory Size?" If you have 48k RAM then reply 60415 for "Memory Size?"

- Prepare the editor tape for playback. Type in SYSTEM and hit ENTER. Then type in LOADER and hit ENTER.

- When the loader has been read in, type /6681 and hit ENTER. This will return you to BASIC READY mode.

- Type in the command POKE 18681,195 and hit ENTER. If you get an error message, then type in this command again.

- Type in the command POKE 18682,0 and hit ENTER.

- Type in the command POKE 18683,0 and hit ENTER.

- Type in the command SYSTEM and hit ENTER. Then type in the command /18432 and hit ENTER. This will activate the LOADER.

- The LOADER will ask for CONFIGURATION? Hit M key if you have 32k RAM. If you have 48k RAM then hit the L key. The LOADER will start reading in the editor program from tape.

- When the editor program has been read in, place the desired system diskette into drive 0 and hit the ENTER key. This will cause the system to boot up DOS.

- Once DOS has initialized, use the appropriate DUMP command as shown above to move the editor from memory to disk and then procede as directed.

APPENDIX C. Default Macro Key Definitions

When the editor is initially loaded the 26 macro keys are set to provide the following commonly used strings and BASIC keywords (remember on a Model 1 do not use the @ sign):

SHIFT@A -	ABS(
SHIFT@B -	******
SHIFT@C -	CHR$(
SHIFT@D -	DATA
SHIFT@E -	ELSE
SHIFT@F -	FOR
SHIFT@G -	GOTO
SHIFT@H -	GOSUB
SHIFT@I -	INPUT
SHIFT@J -	INKEY$
SHIFT@K -	CLEAR
SHIFT@L -	LPRINT
SHIFT@M -	MID$(
SHIFT@N -	NEXT
SHIFT@O -	OUT
SHIFT@P -	PRINT
SHIFT@Q -	PRINT"
SHIFT@R -	RETURN
SHIFT@S -	SIN(
SHIFT@T -	THEN
SHIFT@U -	USING
SHIFT@V -	VARPTR
SHIFT@W -	USR(
SHIFT@X -	FORX=
SHIFT@Y -	FORY=
SHIFT@Z -	STOP

APPENDIX D. Listing of Sample Program

```
10  REM  THIS PROGRAN DEMONSTRATES ALL OF THE FEATURES OF
    THE BASIC FULL SCREEN EDITOR.
20  REM (C) 1980 BY COMPUTER APPLACITIONS UNLIMITED.
30  CLS:A=1:B=2:C=3:D=4:E=5:F=6
40  REM DRAW A BORDER AROUND THE SCREEN:
50  FOR X=0 TO 127: SET(X,0) : SET(X,47) : NEXTX
60  FOR Y=0 TO 47 : SET(0,Y) : SET(127,Y): NEXTY
70  REM NOW FLASH NUMBERS INSIDE THE BORDER:
75  REM CODE TO FLASH"1":
80  PRINT@140," "; :GOSUB 500
90  PRINT@140,"1"; :GOUSB 500
100 A=A-1 : IF A>0 THEN 80 'FLASH IT "A" TIMES  AND  THEN
    GO ON
110 REM CODE TO FLASH "2":
120 PRINT@275," "; :GOSUB 500
130 PRINT@275,"2"; :GOUSB 500
140 B=B-1 : IF B>0 THEN 120 'FLASH IT "B" TIMES  AND  THEN
    GO ON
150 REM CODE TO FLASH "3":
160 PRINT@411," "; :GOSUB 500
170 PRINT@411,"3"; :GOUSB 500
180 C=C-1 : IF C>0 THEN 160 'FLASH IT "C" TIMES  AND  THEN
    GO ON
190 REM CODE TO FLASH "4":
200 PRINT@547," "; :GOSUB 500
210 PRINT@547,"4"; :GOUSB 500
220 D=D-1 : IF D>0 THEN 200 'FLASH IT "D" TIMES  AND  THEN
    GO ON
230 REM CODE TO FLASH "5":
240 PRINT@683," "; :GOSUB 500
250 PRINT@683,"5"; :GOUSB 500
260 E=E-1 : IF E>0 THEN 240 'FLASH IT "E" TIMES  AND  THEN
    GO ON
270 REM CODE TO FLASH "6":
280 PRINT@819," "; :GOSUB 500
290 PRINT@819,"6"; :GOUSB 500
300 F=F-1 : IF F>0 THEN 280 'FLASH IT "F" TIMES  AND  THEN
    GO ON
400 GOTO 400 'LOOP FOREVER HERE
500 FOR T=1 TO 1500 : NEXT T : RETURN
510 REM THE PRECEDING LINE IS THE DELAY SUBROUTINE
600 END
```

DISCLAIMER

All Computer Applications Unlimited programs are sold on an "AS IS" basis without warranty.

Computer Applicatons Unlimited shall have no liability or responsibility to the customer or any other person or entity with respect to any liablity, loss or damage caused or alleged to be caused directly or indirectly by programs sold by Computer Applications Unlimited, including but not limited to any interruption of service, loss of business or anticipatory profits or consequential damages resulting from the use or operation of such computer program.

***

If you encounter difficulty loading the editor tape then return the tape with proof of purchase. If the tape fails within the first 30 days from the date of purchase CAU will replace it at no charge. If the tape fails after 30 days please return the tape and enclose $6.00 for shipping and handling. Only original CAU tapes will be replaced.

***

If you would like to be on our mailing list please send your name and address to:

Computer Applications Unlimited
Post Office Box 214 Dept. 14B
Rye, New York 10580

***

FULL SCREEN TEXT EDITOR FOR BASIC

DISK DUMP ADDENDUM

If you requested a "disk dump" of the editor then a TRSDOS-compatible, single density disk has been included. This disk contains both the Model I and Model III versions of the editor, each of which can be used in either 32K or 48K RAM environments. The disk contains the following files:

Filename	Use With Model	Use With RAM	Description
XEDIT321/CMD	I	32K	Editor Program
XEDIT481/CMD	I	48K	Editor Program
XEDIT32/CMD	III	32K	Editor Program
XEDIT48/CMD	III	48K	Editor Program
XEDIT32/BAS	III	32K	Editor Startup Program
XEDIT48/BAS	III	48K	Editor Startup Program
SAMPLE/BAS	all	all	Demonstration Program

The disk DOES NOT contain an operating system. In a standard Model I system it can be used directly in drive 1. Model III users must CONVERT the files to double density before use (refer to the CONVERT utility in your DOS manual). We recommend that you COPY or CONVERT the files onto your own system disk and then put the original editor disk away to serve as a backup. Remember, you are only authorized to make copies of this program for your own use on your own computer. Giving copies to friends or associates is a violation of the copyright law!

Select the appropriate editor program file from the table above and rename it to XEDIT/CMD. Model III users should also select the appropriate editor startup program file and rename it to XEDIT/BAS. All of the other files (except SAMPLE/BAS) can now be deleted. To load the editor program into memory, type in the command XEDIT from DOS READY. The rest of the startup procedure depends on whether you are using a Model I or III.

For Model I, type in the command BASIC, then protect memory by specifiying a memory size of 44031 if you have a 32K system or 60415 if you have a 48K system. To activate the editor, type in the command SYSTEM and then the command /44032 for 32K systems or /60416 for 48K systems. Hit the SPACE BAR, then type in the command NEW. You can now load SAMPLE/BAS and proceed with chapter IV of the editor manual.

For Model III, you can start BASIC and activate the editor with a single command. This command is

```
     BASIC XEDIT/BAS -M:44031        for 32K systems, or
     BASIC XEDIT/BAS -M:60415        for 48K systems.
```

Then hit the SPACE BAR, load in SAMPLE/BAS, and proceed with chapter IV of the editor manual.

CAU, Inc., P.O. Box 214, Rye, New York 10580 Oct. 1982