Southern Software

EDIT

Full-Screen Editor

for TRS-80* BASIC

Southern Software

PO Box 39, Eastleigh, Hants, England, SO5 5WQ

Southern Software Full-Screen BASIC Editor for TRS-80 (Models I or III) or Video Genie

Installing the Editor (Size 3328 bytes, Hex 0D00)

Refer to the separate sheet, RELOC, for instructions on loading and relocating the tape supplied. You need to allocate 3328 bytes of memory in protected storage for the Editor. If you have no other machine code programs in protected memory then answer the "MEMORY SIZE?" question as follows:

Machine Size	MEMORY SIZE answer
16K	29440
32K	45824 (TRSDOS: 45760)
48K	62208 (TRSDOS: 62144)

Use LEVEL II (not Disk BASIC) to load the tape initially, and to relocate the Editor to the appropriate target address for your machine. Then you can dump a 3328-byte core-image file either on tape (using TBUG or Southern Software's TSAVE) or on disk using the TRSDOS or NEWDOS DUMP command.

If you load the Editor directly from this dumped copy in the future, remember to leave enough protected memory for it under LEVEL II or DOS.

Under TRSDOS, the top 64 bytes of memory are corrupted by the loader, and so cannot be used for the Editor.

Invoking the Editor

Activate the Editor by branching to its first location:

```
SYSTEM (enter)
*? /aaaaa (enter)        where aaaaa is your chosen starting address of the Editor
READY
```

From now on the Editor can be invoked by the /EDIT command:

/EDIT or /EDIT nnnn or /EDIT.

where nnnn is a statement number. "." stands for "current line number").

To deactivate the full screen Editor, use the SYSTEM command again to branch to the starting location of the Editor; this second invocation reverses the effect of the first. You can activate and deactivate the Editor as many times as you like. You must NOT destroy the Editor in storage while it is activated - else BASIC will fail. If you leave BASIC (using CMD"S") while the Editor is activated (or deactivated), you can reenter BASIC and reactivate it, without reloading it.

Early versions of ACCEL2 treat /EDIT as a syntax error, when ACCEL2 is activated. Circumvent this by activating the editor after ACCEL2. Under NEWDOS80 precede the /EDIT by a blank to avoid NEWDOS80 preempting /.

The Display Format

After you use the /EDIT command you will see 15 lines of your program, starting from the one you specified on the /EDIT command, or at the beginning of the program if you did not specify a line number. The left hand column and right hand column normally contain a vertical bar. The bar is broken to show you where program lines spill over into more than one line on the screen. The bar at the end of the line kinks to show you when you have changed a line. The bar at the left hand end of a line will be changed by the Editor to a different - flashing - character to remind you about things you have done to the line.

The bottom line of the screen is used for messages and commands.

The Cursor

At one place on the screen there will be a flashing blob. This is called the cursor. You can see the character that is under the cursor as it flashes. Normally when you press a key the corresponding character is placed where the cursor is (replacing the character that was there). The cursor then moves onto the next position. This is the normal way to change characters.

The four arrow keys on the TRS-80 keyboard enable you to move the cursor without changing the characters that the cursor moves over. The cursor moves one step in the direction of the arrow each time you press the key. Like all other keys, if you hold an arrow key down for a second or so the function of the arrow will be repeated quickly until you release the key. This means you can move the cursor all over the screen easily. So you use the arrow keys to move the cursor to the characters that you want to change.

You can do this even if you cannot see the line you want to change - to get to earlier lines of your program, you move the cursor up towards the top of the screen and keep going! The Editor will automatically bring the earlier lines of your program into view. Similarly, to see later program lines you try to move the cursor off the bottom of the screen and the lines automatically come into view.

This is called "automatic scrolling".

Special Keys - "BREAK", "ENTER", "CLEAR", Arrows and "@"

The "BREAK" key puts you into a special command mode. The commands are described below. Pressing "BREAK" twice (that is pressing BREAK when in special command mode) puts you back into BASIC.

The "ENTER" key is used to complete some operations. These are operations that give you a chance to change your mind. Pressing "ENTER" commits any changes. You press "CLEAR" if you have changed your mind - this causes the operation to be cancelled. Most commands also have the effect of "ENTER" on operations that have not yet been completed, as does automatic scrolling.

The four arrow keys move the cursor. If you press an arrow key while holding down the SHIFT key, you can type the arrow as a plain character. The Editor does NOT support the special meanings that BASIC applies to the arrow keys (backspace, tab, newline) or to SHIFT/arrow (delete line, 32-char display). The newline key is accepted and placed in the program, but it is shown as a down arrow.

The "@" key has a special meaning to the Editor. If you wish to type a plain "@" then press the "0" (zero) key while holding down the SHIFT key.

The Control Key

The Southern Software BASIC Editor uses the "@" key like a shift key to give a special meaning to the key you press at the same time. When used in this way the "@" key is called the "control" key. In the rest of this document "@/X" means hold down the "@" key and then press the "X" (for example).

Extended Cursor Control Keys

"@" with numeric keys help you do more things with the cursor:

@/1 - makes the cursor jump back to the beginning of the current line.

@/2 - makes the cursor jump to the beginning of the next line

The following are alternatives to the arrow keys for those keyboards without them. You may also find them convenient if you have a numeric keypad.

@/3 - is the same as the down-arrow; that is it moves the cursor down one line.

@/4 - is the same as "<-"; that is it moves the cursor one character to the left.

@/5 - is the same as "->"; that is it moves the cursor one character to the right.

@/6 - is the same as the up-arrow; that is it moves the cursor up one line.

@/7 - lets you insert characters.
The first time you press @/7 an "I" appears in the bottom right hand corner of the screen - this means that you are in "insert mode". The next time you press @/7 the "I" disappears and you return to normal mode. ENTER also ends "insert mode".

When in "insert" mode, the character that you type does not replace the character at the cursor position, but rather the existing character and all the rest of the line are moved up to make room for the new character - so the new character is inserted into the line.

@/8 deletes the character that is at the cursor position and moves the rest of the line back to fill the space.

@/9 deletes the rest of the line starting from the cursor position.

Line Commands

"@" with a letter is used mainly for commands that operate on whole lines. The line that is affected is the one containing the cursor.

@/L means lock, and is a Shift Lock which switches you from upper case only to upper/lower case, or back again. To ensure a valid BASIC program, lower case characters persist only inside quoted strings or remarks.

Deleting Lines.

@/D deletes the line containing the cursor. Actually it just causes a "D" to flash at the left hand end of the line. When you next press "ENTER" the line will be deleted. This is a special feature to enable you to change your mind. If you decide not to delete the line (for example, because you selected the wrong line) press "CLEAR" instead of ENTER.

You can also delete a line just by deleting all the characters of the line. If a line is blank when you press "ENTER" (or the line only contains a line number) the line is automatically deleted.

Splitting and Joining Lines.

@/S will split the current line into two lines, at the cursor position, deleting the character under the cursor, which will normally be a ":" statement divider.

@/J will join the line identified by the cursor with the line below, inserting a ":" between them.

Creating New Lines.

The Editor lets you create lines in several different ways - to save typing. But you should be careful how you create line numbers. When the Editor creates a line, it often has the same number as an existing line and/or its number is out of sequence with the lines before and after it. Normal BASIC does not allow this but the Southern Software BASIC Editor does. This is essential to allow you to get the benefit of copying and moving lines. When you have created a line with a line number that normal BASIC would not allow the Editor flashes a "$" character at the left hand end of the line. You should change the line number to a valid one by overtyping the existing one.

You should not return to BASIC while there is an invalid line number in your program or else you will get strange effects. If you do return to BASIC with invalid line numbers you will not harm your program - just return to the Southern Software BASIC Editor and correct the error. You can even save and reload such a program in the normal format but NOT with the ASCII option.

The Editor does not duplicate the function of AUTO, and where you have many new lines to add or insert in your program, you may find it more convenient to exit from the Editor and use AUTO.

@/I inserts a new line after the line containing the cursor. If you dont type anything into the new line before the next time you press "ENTER" the new line will disappear again. You cannot use @/I to insert a line at the top of your program - use @/R to repeat the top line line and update the first version of it.

@/R replicates the current line. That is it creates an exact copy of the line immediately after it. You use this function to save typing when you need to create a similar line.

@/C copies a line to a specified position. After you press @/C a "C" will flash at the left hand end of the line that contained the cursor. You then move the cursor to the program line where you want the copy of the line placed. Then press @/A if you want the copy placed "after" the line with the cursor, or @/B if you want it placed "before" the line with the cursor.

Note that there can be automatic scrolling between pressing the @/C and the @/A or @/B. You can even update lines on the way, or you can use the "find" command (see later) to get you where you want the copy of the line.

The result of the copy command is an exact copy of the original line. You will always have to change the line number of the new line or the original.

@/M moves a line to a new position. You use @/M with @/A or @/B to show where the line is to be moved to - just like @/C.

@/P positions the line according to its line number. @/P is an alternative way to move or copy a line. Instead of copying or moving a line and then changing it and/or its line number, you change the line number, or the line number of a copy created by @/R and then use @/P. The line is then moved to its correct sequential place according to the line number you have given it.

Block Commands

@/Q starts a block command. The line containing the cursor when you press @/Q is the beginning of a block of lines. You then move the cursor to the end of a block of lines (which may cause autoscrolling) and press a key for a line command. The line command then applies to the whole block of lines.

In this way you can copy, move, replicate, delete, and position whole blocks of lines.

Also note

- If you change your mind about the beginning of the block, you can repeat the @/Q command. The original beginning of the block is forgotten and the second one is used.
- You can specify the end of the block as a line that is before the beginning. The effect is the same.
- You can use the "find" command to get you to where you want the end of the block to be.
- Block delete happens immediately you press @/D (unlike line delete).
- Block position keeps the block of lines together. It repositions the whole block according to the number on the first line of the block.

Special Commands

If you press "BREAK" when in normal mode, you can type a special command in the command area. When in special command mode the "BREAK", "ENTER" and "CLEAR" keys mean the following:

* BREAK - ignore any special command, leave the Southern Software BASIC Editor and return to BASIC. This gets you back to BASIC to run your program or to save or load programs etc.

* ENTER - execute the special command and return to normal mode.

* CLEAR - you have changed your mind. Return to normal mode ignoring any special command. (If your keyboard does not have a CLEAR key you can use @K as an alternative.)

Note that when entering a special command you can use the normal cursor functions (arrows and @/7, 8, 9) to help you correct mistakes in the special command. The Special Commands are:

* nnnnn

where nnnnn is a number between 0 and 65520. This positions the display so that at the middle of the screen is the statement with this number, or the first statement after the number if there is no exact match. In particular, the special command "0" will display the first lines of the program and "65520" will show the last lines.

* Fdxxx...d

where xxx... stands for any characters (up to eight) and d is any delimiting character you choose. This means find the first occurrence of the characters xxx... starting from where the cursor was when "BREAK" was pressed. When you press ENTER the cursor is moved to the first such occurrence. The trailing delimiter may be omitted unless you want to search for a string with trailing blanks. Examples of search strings are:

F RETURN F"RETURN" F"RETURN F READ A F/A=B / (search includes trailing blanks)

The characters xxx... are remembered by the Editor so that when in normal mode you can press @/F to repeat the find using the same string of characters.

* Gdyyy...d

where yyy... stands for any characters (up to eight) and d is any delimiter. This command is a "global change" command. When you press ENTER the first xxx... (set by the find command) starting from the current cursor position is replaced by yyy...

The characters yyy... are remembered by the Editor so that when in normal mode you can change the next xxx... to yyy... by pressing @/G. Thus if you wish to change all occurrences of xxx to yy then use the "F xxx" special command to set the "find" string xxx (and find the first one) and use the "G yy" special command to change the xxx to yy and to set the global change string to yy. Then hold down the @/G key - the Editor will move through the program changing each xxx to yy.

To change just some of the occurrences of xxx to yy you use the same special commands but then press @/F to find an occurrence of xxx, then press @/G only if you want to change this occurrence to yy, and then press @/F to go on to the next one.

To DELETE occurrences of a string, find it using the "F xxx..." command and then change it to nothing using the "G" command without any string. Hold @/G down to delete further occurences.

To INSERT a string xxx... in a number of places set the find string to null using the command "F" without a string, position the cursor to where you want the string inserted first, and use the command "G xxx...". This will insert the xxx... at the cursor position. Then position the cursor to the next place and press @/G, etc.

Notes:

1) The Editor uses a variable amount of stack space during editing, as a buffer to hold the currently modified line, and as workspace. This can be as much as 500-600 bytes. So a program that only just fits in unprotected memory may cause an OM (Out-of-Memory) when edited.

2) When you modify a program all BASIC variables are cleared and all files are closed. If you use EDIT to examine a program compiled under ACCEL or ACCEL2, clearing the variables in this way may mean the program will no longer run unless recompiled.

Summary of Functions.

Invocation

/EDIT	invoke Editor at start of program
/EDIT nnnnn	invoke Editor at line nnnnn (or next line)
/EDIT .	invoke Editor at "current" line

Special Keys

BREAK	c	enter special command mode,(twice = return to BASIC)
ENTER		commit operation and updates, reset modes
CLEAR		reset changes since last "ENTER" function, cancel operation
Arrow keys		move cursor one character in the direction of the arrow
SHIFT/0		types the normal "@" character
SHIFT/arrow		types the normal arrow character

Control Commands

@/D	bc	delete line or block of lines
@/I	e	insert new line
@/R	eb	replicate line or block of lines
@/C	bc	copy line or block of lines
@/M	bc	move line or block of lines
@/A	e	after here (use with @/C and @/M)
@/B	e	before here (use with @/C and @/M)
@/P	eb	position line or block of lines
@/Q	e c	start of block command
@/F	e	repeat find operation
@/G	e	global change operation
@/L	c	change shift lock
@/S	e	split line in two
@/J	e	join two lines to one

Notes: e = automatic "ENTER", b = block command (with @/Q), c = clear will cancel.

Special Commands

nnnnn	display statement nnnnn
F xxxxxxxx	find next occurrence of xxxxxxxx
G yyyyyyyy	change xxxxxxxx to yyyyyyyy

Cursor Commands

The following table is arranged like the keys on a TRS-80 numeric keypad:

7 insert mode	8 delete char	9 erase rest
4 <——	5 ——>	6 up
1 start of line	2 next line	3 down

Southern Software's full-screen BASIC Editor is distributed on an "as is" basis, without warranty. No liability or responsibility is accepted for any loss or damage caused or alleged to be caused by its use.