

Elektronik

MIKRODATORTEKNIK

Häfte 19: SKÄRMEDITOR (6809)

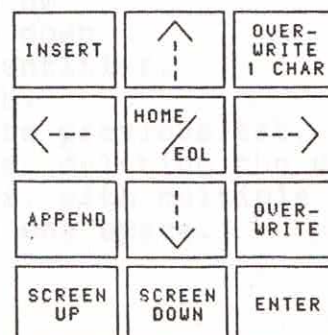
OmegaSoft Screen Editor

INNEHÅLL

	sid
Screen Editor Commands	1
Introduction	3
File Handling Commands	5
EDIT, NEW	5
GET, SAVE, SAVE AND QUIT, SAVE PARTIAL, QUIT	6
Status Line	7
Text Append and Insert	7
Position to Line	9
Cursor Movement	9
Tabs	10
Character and Line Delete	11
Overwrite	11
Move and Copy	12
Text Searching	12

OBS! Denna skärmeditor är modifierad för Heathkits bildskärm H-19

H-19



Elektronik

MIKRODATENTEKNIK

Heft 19: SKRIBENTOR (8809)

INNEHALT

1	Texteingabe
2	Textbearbeitung
3	Textformatierung
4	Textsuche
5	Textsortierung
6	Textkopieren
7	Textdrucken
8	Textspeichern
9	Textöffnen
10	Textschließen
11	Textverwaltung
12	Textverwaltung

Die Skribentor ist ein Textverarbeitungsprogramm, das auf dem IBM PC/XT/AT lauft. Es ist in der deutschen Sprache programmiert und kann in mehreren Sprachen ausgeführt werden.

Text	↑	Text
Text	→	Text
Text	↓	Text
Text	←	Text

Text	Text
Text	Text
Text	Text
Text	Text

FLEX
... FLEX

FLEX

Häfte 10

SR. CMD

Skriver i Pascal

Skärmtyper ~~6800~~

H-19A

(Finns en version f UT100)

Fullskärmseditor

+++ SR <CR>

+++ SR ADD.TXT <CR>

Kommandon s 19-1

Ställ om Tab

Q-buffrade kommandon exekveras
där på resp knapps nedtrycker.

SR fixar g .BAK - fil utan
skr ö ga fil c sa namn.

Knep:

- * Anv back-up - disk
- * Ändra filnamn (lägg + en siffra)

OmegaSoft Screen Editor

SCREEN EDITOR COMMANDS

Non-buffered commands continued (see page 19-9)

original version

	9 _____	Overwrite one character.	1 (digit one)
	3 _____	overwrite characters until escape.	V
H-19	1 _____	enter append mode.	A
	7 _____	enter insert mode.	;

Text search commands

Find _____ will position cursor, response is
 CR = stop command
 Space = continue
 M = multiple occurrences on line

Replace _____ will position cursor, response is
 CR = stop command
 Space = continue
 R = Replace and continue
 M = multiple occurrences on line
 E = entire buffer

Note : When the 'F' prompt appears enter the find pattern - if you enter just a carriage return then this will terminate the command. When the 'R' prompt appears enter the Replace pattern. A '~' is used as a don't care in the find and Replace patterns. The entire buffer option will enable the replacement throughout the rest of the buffer. The multiple occurrences option will allow a match on each occurrence of the find pattern on any one line, else only the first match on a line is recognized. The search will start on the current line, continue to the end of the buffer, start back at the top, and stop on the current line. The cursor will be positioned at the last match when the command terminates.

Insert or append mode

Character entry until an escape is entered. Insert mode will insert characters before the current cursor position. If a carriage return is entered during insert mode then append mode will be entered with the last part of the line carried over until append mode is exited. Append mode will start new lines past the current line with no partial line carried over. In addition the following special characters are recognized :

BS or DEL	_____	backspaces over character.
HT or cntrl T	_____	forward tab.
cntrl R	_____	reverse tab.
cntrl A	_____	copies remainder of last line entered.
cntrl X	_____	removes all characters entered so far.

OmegaSoft Screen Editor

INTRODUCTION

The OmegaSoft Screen Editor is designed to provide a facility for editing Pascal and assembly language programs. The software is designed for maximum efficiency by using a minimum of file accesses and by using some of the features of "smart" terminals.

Minimum Terminal Requirements

Between 80 and 128 characters across on a terminal or memory mapped CRT can be supported. There is no practical limit on the number of lines that can be supported. The terminal must accept a cursor positioning command consisting of a lead-in sequence, line row, and character column. Facilities are provided for reversing row and column in the sequence and for adding an offset of \$32 (space) to the row and column parameters.

Desired Terminal Requirements

The next most important attribute is either a clear line or clear to end of line control sequence. Before issuing this command the cursor will always be positioned to the first character in the line, therefore either sequence will have the same effect. If this sequence is not available then the editor will have to clear a line by writing a space into every position on the line - this is very slow.

The next attribute is either a clear page or clear to end of page control sequence. Before issuing this command the cursor will be positioned to the first character on the screen, therefore either sequence will have the same effect. If this sequence is not available then the editor will have to clear the screen by clearing each line one at a time, this is not desirable.

This editor has the capability of using the built in terminal commands for insert line, delete line, delete character, and reverse scroll. If these are not available then editing will be slowed, but should still be more efficient and easier to use than a line editor.

The insert line command must cause the line on which the cursor is positioned and all lines below it to move down one line. The bottom line is lost and the line where the cursor is positioned to is cleared to spaces.

The delete line command deletes the line on which the cursor is positioned and moves all lines below it up one line, forcing the last line of the screen to a blank line.

The delete character command deletes the character at the cursor location and moves all characters to the right one column to the left.

INTRODUCTION

The Computer Screen Editor is designed to provide a facility for editing textual and assembly language programs. The software is designed for maximum efficiency by using a minimum of line accesses and by using some of the features of "smart" terminals.

Minimum Terminal Requirements

Between 80 and 128 characters across on a terminal or memory mapped CRT can be supported. There is no practical limit on the number of lines that can be supported. The terminal must accept a control positioning command consisting of a lead-in sequence, line, and character column. Facilities are provided for reversing row and column in the sequence and for adding an offset of 12 (spaces) to the row and column parameters.

Desired Terminal Capabilities

The next most important attribute is either a clear line or clear to end of line control sequence. Before issuing this command the cursor will always be positioned to the first character in the line, therefore either sequence will have the same effect. If this sequence is not available then the editor will have to clear a line by writing a space into every position on the line - this is very slow.

The next attribute is either a clear page or clear to end of page control sequence. Before issuing this command the cursor will be positioned to the first character on the screen. Therefore either sequence will have the same effect. If this sequence is not available then the editor will have to clear the screen by clearing each line one at a time; this is not desirable.

This editor has the capability of using the built in command commands for insert line, delete line, delete character, and reverse cursor. If these are not available then editing will be slow, but should still be more efficient and easier to use than a line editor.

The insert line command must cause the line on which the cursor is positioned and all lines below it to move down one line. The bottom line is lost and the line where the cursor is positioned is cleared to space.

The delete line command deletes the line on which the cursor is positioned and moves all lines below it up one line, forcing the last line of the screen to a blank line.

The delete character command deletes the character at the cursor location and moves all characters to the right one column to the left.

OmegaSoft Screen Editor INTRODUCTION

The reverse scroll command will cause the screen to scroll one down line. The cursor will always be positioned to the first character on the screen when this command is issued. Note that this command may be simulated by issuing an insert line command.

It is desirable to display the relative line numbers in half brightness if possible. Note that to use this feature your terminal must accept a sequence to place all further characters into half-brightness, accept a string of characters, and then accept a sequence to place all further characters into full-brightness. No character positions may be used up to store the half and full brightness commands in the terminal.

General Information

A line number is provided that is relative to the start of the text. The first line of the text is line one. Normally the cursor is always to the right of the line numbers (you cannot edit them). The only exception is when the cursor is positioned at line zero in which case the cursor is at the upper left hand corner of the screen. Line zero is only valid in certain circumstances which will be pointed out. If there are no text lines in the edit buffer then the cursor will be on line zero.

The cursor can only be positioned within the valid characters of a line, if you position the cursor past the end of the line you are on then it will move to the far right of the line (representing the carriage return if you like).

This editor does not use scratch files and does not allow you to edit a file that will not fit into memory all at once. When you start to edit a file it is read completely into the edit buffer. When you are done editing the file it is written back under the same file name as you started with (unless told to do otherwise).

It is possible to modify this editor to read in parts of files, edit them, write them to a scratch file, read in the next block, etc. and then rename the scratch file to the original file name. This however, would defeat the purpose of writing an efficient editor. Either you don't have enough memory or your files are quite large. It is usually better to have smaller, more manageable files rather than using paging in an editor.

The first thing that must be done is to install a custom version of the editor on your system for your terminal. To do this refer to the operating system dependant INSTALLATION section at the rear of this manual. After doing that you should read through the rest of the manual. For those of us who prefer to jump right into things you can use the Quick Reference section after doing the editor installation.

OMEGA-8000 Editor INTRODUCTION

The reverse search command will return the screen to the first line. The cursor will always be positioned to the first character on the screen when this command is issued. Note that this command may be simulated by issuing an insert line command.

It is desirable to display the relative line numbers in half brightness if possible. Note that to use this feature you must first accept a sequence to place all further characters into half-brightness, accept a series of characters, and then accept a sequence to place all further characters into full brightness. No character positions may be used up to store the half and full brightness commands in the terminal.

General Information

A line number is provided that is relative to the start of the text. The first line of the text is line one. Normally the cursor is always to the right of the line number (you cannot edit them). The only exception is when the cursor is positioned at line zero in which case the cursor is at the upper left hand corner of the screen. Line zero is only valid in certain circumstances which will be pointed out. If there are no lines in the edit buffer then the cursor will be on line zero.

The cursor can only be positioned within the valid characters of a line. If you position the cursor past the end of the line you are on then it will move to the far right of the line (representing the carriage return if you like).

This editor does not use separate files and does not allow you to edit a file that will not fit into memory all at once. When you start to edit a file it is read completely into the edit buffer. When you are done editing the file it is written back under the same file name as you started with (unless told to do otherwise).

It is possible to modify this editor to read in pairs of files, edit them, write them to a separate file, read in the next pair, etc. and then rename the source file to the original file name. This however, would defeat the purpose of writing an efficient editor. Either you don't have enough memory or your files are quite large. It is usually easier to have multiple, non-manageable files rather than using paging in an editor.

The first thing that must be done is to install a custom version of the editor on your system for your terminal. To do this refer to the operating system dependent INSTALLATION section at the rear of this manual. After doing that you should read through the rest of the manual. For those of us who prefer to jump right into things you can use the Quick Reference section after doing the editor installation.

OmegaSoft Screen Editor

FILE HANDLING COMMANDS

EDIT

Syntax : E <filename>

Will clear the edit buffer and load in the entire contents of the named file. This file name will be remembered for use with a Save command without a file name parameter.

If there is already a file in the edit buffer that has been modified since the last save then you will be prompted to save the old file. If you answer with an "N" or "n" then the old file will not be saved. If you answer with an "C" or "c" then it will save the old file into a new file name that you specify. Any other answer will cause the old file to be written to the file name remembered from the last Edit command. Note that some commands will destroy this remembered file name in which case you will be prompted for the file name as in when answering with a "C".

While the file is loading you may get a "Memory full" error. This means that the file will not fit in the edit buffer and you cannot modify the file. You can however save what part will fit into another file. At the end of the file loading you can get a "n lines truncated" error indicating that "n" lines were too long to fit, the characters at the end are simply discarded. This should only happen with files that may have been prepared with a different editor.

NEW

Syntax : N <filename>

The new command will clear the edit buffer, make sure the file does not exist, remember the file name for use with a future save command, and enter the append mode at line one.

Note that the file is not actually opened at this time. If there is an old file in the edit buffer that has been modified then you will be prompted to save it as in the EDIT command.

This command is the normal method of creating new files.

FILE HANDLING COMMANDS

EDIT

Syntax: E file-name

Will edit the edit buffer and load in the entire contents of the named file. This file name will be remembered for use with a later command without a file name parameter.

If there is already a file in the edit buffer that has been modified since the last save, you will be prompted to save the old file. If you answer with an "n", then the old file will not be saved. If you answer with a "y", then it will save the old file into a new file name that you specify. An error answer will cause the old file to be written to the file name remembered from the last EDIT command. Note that some commands will destroy this remembered file name in which case you will be prompted for the file name as in when answering with a "y".

While the file is loading you may get a "Memory full" error. This means that the file will not fit in the edit buffer and you cannot modify the file. You can however save what part will fit into another file. At the end of the file loading you can get a "line truncated" error indicating that a line was too long to fit. The characters at the end are simply discarded. This should only happen with files that may have been prepared with different editors.

END

Syntax: W file-name

The new command will clear the edit buffer, save the file does not exist, remember the file name for use with a later command, and start the append mode at line one.

Note that the file is not actually opened at this time. It is an old file in the edit buffer that has been modified then you will be prompted to save it as in the EDIT command.

This command is the most method of creating new files.

OmegaSoft Screen Editor
FILE HANDLING COMMANDS

GET

Syntax : G <filename>

This command will open the specified filename and read its entire contents into the edit buffer past the line where the cursor is positioned. This is useful for merging files together, or with the use of the save partial command can be used to "cut and paste" parts of files together.

This command destroys the remembered filename from an EDIT or NEW command.

SAVE

Syntax : S [<filename>]

This command will write out the entire contents of the edit buffer to the specified file. No changes are made to the edit buffer. If there is a valid remembered filename then the filename parameter can be omitted.

SAVE and QUIT

Syntax : SQ

This command will act the same as both a SAVE command with no filename and a QUIT command. This is a short-cut that is commonly used since in general you will be editing one file and saving it back and then exiting the editor.

SAVE PARTIAL

Syntax : SP <first_line>,<last_line> <filename>

This command will save only a portion of the edit buffer to the file. First_line through last_line will be written, the edit buffer will not be modified, and the remembered filename will be destroyed.

QUIT

Syntax : Q

This command will exit the editor and is not buffered (a carriage return is not required). If the contents of the edit buffer have been modified since the last save command then you will be prompted to save it as in the EDIT command.

END

Syntax: <fillname>

This command will open the specified filename and read the entire contents into the edit buffer. The line where the cursor is positioned. This is useful for saving files together, or saving the use of the save partial command can be used to "put" and "get" pairs of files together.

This command also saves the contents of the edit buffer into the specified filename from an EDIT or END command.

SAVE

Syntax: <fillname>

This command will save the entire contents of the edit buffer to the specified file. No changes are made to the edit buffer. It saves a valid remembered filename when the filename parameter can be omitted.

SAVE and QUIT

Syntax: <fillname>

This command will save the entire contents of the edit buffer to the specified file. This is a direct-copy that is commonly used when saving a file. It saves a valid remembered filename when the filename parameter can be omitted.

SAVE PARTIAL

Syntax: <fillname> <start_line> <end_line>

This command will save only a portion of the edit buffer to the specified file. The portion specified will be written. The edit buffer will not be modified, and the remembered filename will be unchanged.

QUIT

Syntax: <fillname>

This command will save the entire contents of the edit buffer to the specified file. If the contents of the edit buffer have been modified since the last save command then you will be prompted to save as in the EDIT command.

OmegaSoft Screen Editor

STATUS LINE

To display the status line the ? key must be hit while in command mode. To exit this display hit any key. The status line will display the editor version number, the copyright notice, the current file name, if any, the last line number in the edit buffer, and the amount of memory available (hex) in the edit buffer. Note that there is some memory that cannot be used and you will not be able to change the edit buffer if this value falls below 200 hex bytes.

TEXT APPEND AND INSERT

If an 'I' is entered in command mode you will enter the text append mode. You exit this mode by entering an escape (cntrl []). The editor will start creating new lines immediately following the line where the cursor was when this mode was entered. To start a new file, or insert lines before the first line of the file you can move the cursor to line zero and enter this mode.

Any text typed in will be entered into the edit buffer, with a carriage return starting a new line. You cannot type past the end of the line and this editor will not do "word-wrap". A number of special control characters are recognized in text append and text insert modes, they are :

BS (cntrl H) or DEL (\$7F) : Backspaces one character and removes that character from the buffer (destructive backspace). These backspace codes can be changed by the user during installation.

HT (cntrl I) or cntrl T : Forward Tab - move to next tab stop. If there are no further tab stops to the right then the cursor will move to the far right of the line.

cntrl R : Reverse Tab - move to previous tab stop. If there are no further tab stops to the left then cursor will move to the first position on the line (tab 0).

cntrl A : Copies previous line to current line starting at the current cursor position. Useful for duplicating equivalent or similar text from the previous line.

cntrl X : Removes all characters on the line and positions the cursor at the first position on the line.

STATUS LINE

The display and status line the 7 key must be hit while in command mode. To edit status display hit any key. The status line will display the current status of the system. The copyright notice will be displayed. If any, the line number in the status buffer, and the number of memory available (bytes) in the status buffer. Note that there is some memory that cannot be used as you will not be able to change the edit buffer if this value falls below the low byte.

TEXT BUFFER AND INSERT

If an ' is entered in command mode you will enter the text append mode. You exit this mode by entering an escape key. The editor will start creating new lines immediately following the line where the cursor was when this mode was entered. To insert a new line or insert lines before and after the line where you can move the cursor to line zero and enter this mode.

Any text typed in will be entered into the edit buffer. When carriage return is typed a new line will be entered after the end of the line and this will be the "word-wrap". A number of special control characters are recognized in text editors and text insert modes, they are:

As insert or delete a character or one character and remove that character from the buffer. (insert or delete). When backspace codes can be changed by the user during installation.

By default, if of shift T, control T - move to next line up. If there are no further lines up then the cursor will move to the first line of the file.

Control B: Reverse Tab - move to previous tab stop. If there are no further tab stops, the cursor will move to the first position of the line.

Control A: Cursor position line to correct line starting at the current cursor position. Control Z: Application termination. Similar text lines are possible.

Control X: Remove all characters on the line to the position of the cursor at the first position of the line.

OmegaSoft Screen Editor TEXT APPEND AND INSERT

If a "7" is entered when in the command mode then text insert mode will be entered. This mode is exited by typing an escape. The text to the right of the cursor is moved to the far right of the line and you can enter text until you run into the first character of the text to the right (the last part of the original line). If you enter a carriage return while in this mode then you will enter the text append mode. The only difference is that if you typed the carriage return with text to the right of the cursor then the text to the right will be carried down to the next line, and will continue to do this until the escape is entered at which point it will join with the contents of the last line entered.

As an example of this usage the following Pascal segment is going to be broken down into 3 lines instead of one, with the cursor located at the underlined character in the text and your entries in bold print.

```
if count = 3 then count := 4 ;
```

position the cursor to the "then", enter a "7", and enter a carriage return :

```
if count = 3 then count := 4 ;
```

will result in

```
if count = 3
```

```
-
```

```
then count := 4 ;
```

type in a tab character and then escape :

```
if count = 3
```

```
- then count := 4 ;
```

By positioning the cursor to "count" and re-entering the insert text mode this sequence can be repeated to produce :

```
if count = 3
```

```
then
```

```
count := 4 ;
```

OMEGAPOINT REPORT EDITOR TEXT APPEND AND INSERT

If a "V" is entered when in the command mode then the insert mode will be entered. This mode is entered by typing an escape. The text to the right of the cursor is moved to the left of the line and you can enter text only. You can also enter the last character of the text to the right (the last part of the original line). If you enter a carriage return while in this mode then you will enter the text append mode. The only difference is that if you typed the carriage return with text to the right of the cursor then the text to the right will be carried down to the next line, and will continue to do this until the escape is entered at which point it will join with the contents of the last line entered.

An example of this usage the following Pascal segment is doing to be broken down into 3 lines instead of one, with the cursor located at the underlined character in the text and your entries in bold print.

```
if count = 3 then count := 4 ;
```

position the cursor to the "then", enter a "V", and enter a carriage return :

```
if count = 3 then count := 4 ;
```

will result in

```
if count = 3
```

```
then count := 4 ;
```

type in a tab character and then escape :

```
if count = 3
```

```
then count := 4 ;
```

By positioning the cursor to "count" and re-entering the insert text mode this sequence can be repeated to produce :

```
if count = 3
```

```
then
```

```
count := 4 ;
```

OmegaSoft Screen Editor

POSITION TO LINE

Syntax : P <expression>

Where expression consists of line numbers with any valid combination of the operators plus "+" and minus "-". No spaces may exist in the expression. A line number can be a decimal number, the period "." which represents the current line, or a dollar sign "\$" which represents the last line of the file. You cannot position the cursor to a line that exceeds the last line number, you can however, position to line zero.

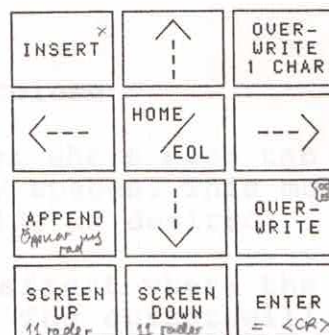
CURSOR MOVEMENT

The basic cursor movement is controlled by a group of keys controlled by the right hand :

H-19



? HELP
 <CTRL> S DEL CHAR
 <CTRL> X DEL LINE



* U+ 19a c <ESC>

The 4, 5 and 6 keys control cursor left/right movement, 4 will move the cursor left one character (but not to the left of the first character of the line). 6 will move the cursor right one character, if past the end of the line the cursor will move to the far right of the screen to indicate that it positioned on the end of line marker. The 5 key will move the cursor alternately to the far left and far right of the line.

The 8 and 2 keys will move the cursor up (8) or down (2) one line on the screen. Moving up past line one will position you at line zero, you cannot move down past the last line.

The 0 and . (period) keys will move the cursor up (0) or down (.) one half page (number-of-lines-on-screen >> 2). You cannot move to line zero using the 0 key.

The space bar will move you to the next identifier on the current line. An identifier is defined to be any sequence of characters containing upper or lower case letters, digits 0 through 9, and the characters "_", "\$", and ".". After you are positioned to the last identifier using the space bar will position you back to the first identifier on the line.

HT (cntrl I) or cntrl T will move you to the next tab stop if still within the current line, or the far right of the screen. Cntrl R will position you to the previous tab stop.

POSITION TO LINE

Symbol : P (expression)

Where expression consists of line numbers with any valid combination of the operators plus "+", and minus "-", no spaces may exist in the expression. A line number can be a decimal number, the period ".", which represents the current line, or a dollar sign "\$" which represents the last line of the file. You cannot position the cursor on a line that exceeds the last line number, you can however, position to line zero.

CURSOR MOVEMENT

The basic cursor movement is controlled by a group of keys controlled by the right hand:

RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT

RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT
RIGHT	DOWN	LEFT

H-10

The 4 and 5 keys control cursor left/right movement. 4 will move the cursor left one character (but not to the left of the first character of the line). 5 will move the cursor right one character, it past the end of the line the cursor will move to the far right of the screen to indicate that it is positioned on the end of line marker. The 6 key will move the cursor alternately to the far left and far right of the line.

The 8 and 7 keys will move the cursor up (8) or down (7) one line on the screen, moving up past line one will position you at line zero, you cannot move down past the last line.

The 0 and 1 (period) keys will move the cursor up (0) or down (1) one half page (number of lines on screen is 3). You cannot move to line zero using the 0 key.

The space bar will move you to the next identifier on the current line. An identifier is defined to be any sequence of characters containing upper or lower case letters, digits 0 through 9, and the characters "_", ".", and "\$". After you are positioned to the last identifier using the space bar will position you back to the first identifier on the line.

BT (ctrl I) or ctrl T will move you to the next tab stop. If still within the current line, or the far right of the screen, ctrl R will position you to the previous tab stop.

OmegaSoft Screen Editor

TABS

The default tab settings are defined by a string in the setup (.ps) file, see the installation section of this manual. The tab stops can be reset when you are in the command mode. There are two types of tabbing, repetitive and non-repetitive.

REPETITIVE

Syntax : **TR <increment>**

Where increment is a decimal number that defines the multiplier to be used when setting tabs. For instance if you type in : TR 2 then you will get tabs at 0,2,4,6,8...., TR 5 will get you tabs at 0,5,10,15..... This mode is useful for Pascal and other structured languages.

NON-REPETITIVE

Syntax : **TN column column column**

Where column is a decimal number where each tab is to be placed. The columns must be separated by spaces. This mode can be used to space out assembly language fields if desired.

Note that the tabs will also affect where the cursor is placed when appending lines of text. The cursor will be positioned to the closest tab stop to the left that is no further right than the first character of the previous line.

TAB MARKERS

Syntax : **T+ or T-**

If T+ is entered then lines will be displayed with tab markers at those tab positions up until the first non-blank character in a line. This is useful in structured languages to make sure matching symbols (such as begin ... end) are on the same tab position. T- will turn off this mode (default).

OmegaSoft Screen Editor

CHARACTER AND LINE DELETE

Individual characters on a line can be deleted by typing in a cntrl S. If the cursor is on the far right of the screen then by entering a cntrl S you will be effectively deleting the carriage return at the end of the line. This has the effect of merging the current line you are on with the following line. If the following line contains leading spaces then only the first space is kept. As an example, in the following pascal segment we wish to merge the "if k = 3" and "then s := false ;" into one line.

```
if k = 3
  then s := false ;
```

position the cursor to the far right of the screen :

```
if k = 3
  then s := false ;
```

and hit the cntrl S, the result will be :

```
if k = 3 then s := false ;
```

Whole lines can be deleted by positioning the cursor to the line to be deleted and hitting a cntrl X.

A range of lines can be deleted by using the delete command.

Syntax : DE <expression> or DE <expression>,<expression>

The first form will delete the line number given. The second form will delete a range of lines. There can be no spaces between the first expression, the comma, and the second expression.

OVERWRITE

Overwriting allows you to change one or more characters without the necessity of deleting the character and then inserting the new character. There are two forms of overwrite, the first is single character overwrite. This is accomplished by positioning the cursor to the character to be changed, hitting the 9 key in command mode and then typing in the new character. The second method is for multiple characters. Position the cursor to the first character to be changed, type "3" in command mode, and then start typing the new characters. This mode is exited by typing an escape or by running off the end of the current line.

CHARACTER AND LINE MODES

Individual characters on a line can be deleted by typing a backspace. If the cursor is on the far right of the screen then by entering a control X you will be effectively deleting the carriage return at the end of the line. This has the effect of moving the current line you are on with the following line. If the following line contains leading spaces then only the first space is kept. As an example, in the following Pascal segment we wish to merge the "if x = 3" and "then x := false" into one line.

```
if x = 3
```

```
then x := false ;
```

position the cursor to the far right of the screen ;

```
if x = 3
```

```
then x := false ;
```

and hit the control X, the result will be :

```
if x = 3 then x := false ;
```

Whole lines can be deleted by positioning the cursor to the line to be deleted and hitting a control X.

A range of lines can be deleted by using the delete command.

Syntax : DE <expression> or DE <expression> , <expression>

The first form will delete the line number given. The second form will delete a range of lines. There can be no spaces between the first expression, the comma, and the second expression.

OVERWRITE

Overwriting allows you to change one or more characters without the necessity of deleting the character and then inserting the new character. There are two forms of overwrite, the first is single character overwrite. This is accomplished by positioning the cursor to the character to be changed, hitting the key in command mode and then typing in the new character. The second method is for multiple characters. Position the cursor on the first character to be changed, type "3" in command mode, and then start typing the new characters. This mode is exited by typing an escape or by running off the end of the current line.

OmegaSoft Screen Editor

MOVE AND COPY

Move Syntax : **M** <expression> or **M** <expression>,<expression>

Copy Syntax : **C** <expression> or **C** <expression>,<expression>

The Move and Copy commands are identical except that in the copy command the source lines stay intact, in the move command the source lines are deleted. In either case a copy of the source lines are moved to the destination line range. The destination line range is defined to be immediately following the line where the cursor is positioned. In the first form of the syntax expression defines the line number of the source. In the second form the two expressions define a line number range for the source. There must be no spaces between the first expression, the comma, and the second expression.

TEXT SEARCHING

The find and replace commands allow you to search for a specific pattern of characters and to position the cursor there (find) or to change that pattern of characters to a new pattern (replace).

In either command you position the cursor to where you want to start searching, the search will continue to the end of the buffer and then start back at the top and continue until one line before where you started searching (circular buffer).

Find syntax : **F**

Replace syntax : **R**

When you enter these commands you will be prompted on the bottom line by a "F>", enter the pattern to search for and a carriage return. If you entered the replace command then you will receive a "R>" prompt, enter the new pattern to replace the search pattern. Searching will now commence and the cursor will be positioned to the first occurrence of the search pattern. If there is no occurrence then the bell will sound indicating the completion of this command. At this point the operation differs between the find and replace commands.

In the find command you have 3 possible responses. If you enter a carriage return then the cursor will stay at the found pattern and the command will be exited. If you enter a space then it will continue the search - sounding the bell and exiting the command if no further occurrences are found. If you enter a "M" then it will set a flag that will allow it to recognize multiple occurrences of the pattern on the same line - else it will only recognize the first occurrence. Hitting the "M" simply sets the flag and does not move the cursor.

OmegaSoft Screen Editor
TEXT SEARCHING

In the replace command you have 5 possible responses. Entering a carriage return will exit this command. Entering a space will replace the found pattern with the replacement pattern and continue on to the next occurrence. Entering a "M" will set the multiple occurrence flag as in the find command. Entering an "E" will signal the command to replace the found pattern with the replacement pattern in the rest of the buffer without further input from the user.

As shipped (except when run under the debugger) the character "~" may be used as a "don't care" character in the find and replace patterns. A "~" will match any character. As shipped upper and lower case letters are equivalent when searching, refer to the installation section for information on changing this.

OmniSearch 3.0 User Manual TEXT SEARCHING

In the replace command you have 2 possible responses. Entering a carriage return will exit this command. Entering a space will replace the found pattern with the replacement pattern and continue on to the next occurrence. Entering a "M" will not replace multiple occurrences flag as in the find command. Entering an "E" will signal the command to replace the found pattern with the replacement pattern in the rest of the buffer without further input from the user.

An escaped (except when run under the debugger) the character "" may be used as a "don't care" character in the find and replace patterns. A "" will match any character. An escaped upper and lower case letters are equivalent when searching, refer to the installation section for information on changing this.

DEBUG COMMAND SUMMARY

General system control

CALC	Calc	13-16
DELAY=<value>	DElay	13-17
DEPTH	DEPh	13-17
EXIT	Exit	13-17
FLAG[=<address>]	FLag	13-18
IND=ON or OFF	IND	13-18
MACH	MAch	13-18
MODE=1 or 0	MOde	13-18
REG	Reg	13-19
RESET	RESet	13-19
RET	RET	13-19
SET,<register list>	Set	13-20
STACK[,<value>]	STACK	13-20
STATES	STATes	13-20
TRAIL	TRAIL	13-20
X,<FLEX command>	X	13-21

Memory commands

ASM[,<address>]	Asm	13-21
DIS,<start address>,<stop address>	Dis	13-22
DUMP,<address>	DUmp	13-22
FILL,<start address>,<stop address>,<byte>]	FILL	13-22
FIND,<start address>,<stop address>,<string>	FIND	13-22
MEM,<address>	Mem	13-23

Simulation control

GO	Go	13-24
JUMP,<address>	Jump	13-24
PAST[,<value>]	PAst	13-24
SIM[,<count>]	SIM	13-24
START,<address>	Start	13-25
STEP[,<count>]	STEP	13-25
TRACE=<value>	Trace	13-25
TSM[,<count>]	TSm	13-26
'ctrl'C	operator halt	13-26
escape	halt on EOL	13-26
'ctrl'F	faster	13-26
'ctrl'S	slower	13-26

Breakpoints

B,<actions>,<address>[,<modifier><count>],IF<condition>]	B	13-27
B&<address>[,<modifier><count>],IF<condition>]	B&	13-27
R...Print register contents	R...	
Z...Zero the status counter	Z...	
T...Trace mode on	T...	
U...Trace mode off (untrace)	U...	
H...Histogram counter	H...	
M...Print message	M...	
J...Jump to new address	J...	
S...Stop simulation	S...	
BP[,<address>[-<address>]]	BP	13-29
CLB[,<address>[-<address>]]	CLB	13-29
CLH[,<address>[-<address>]]	CLH	13-29
CLM	CLM	13-29
HIST[,<address>[->address]]	Hist	13-30
RT[,<address>]	RT	13-30

Memory protection

BOUND\$[,<types>]	BOunds	13-31
CLP[,<type>]	CLP	13-31
PROT,<address>[-<address>],<type>	Prot	13-32

Execution traps

INST=<count>	INST	13-32
ITRAP=ON or OFF	ITrap	13-33
NEST=<value>	Nest	13-33
STOP=<address>	STOP	13-33
XFR=ON or OFF	XFR	13-33

Interrupt control

IRQ=<count>	IRQ	13-34
NMI=<count>	NMI	13-34
'Ctrl'I	'Ctrl'I	13-34
'Ctrl'N	'Ctrl'N	13-34

OmegaSoft Screen Editor

SCREEN EDITOR COMMANDS

Line syntax : "." = current, "*" = last, digits = offset or relative line number, + and - allowed.

Buffered Commands ('E' prompt)

Edit <filename> _____	Clear buffer and read file.
New <filename> _____	Clear buffer and set filename.
Get <filename> _____	Get file past cursor.
Save [<filename>] _____	Save entire file.
Save and Quit _____	Save entire file and quit.
Save Partial <line>,<line> _____	Save part of file.
Move <line> ,<line> _____	will move lines to after cursor position.
Copy <line> ,<line> _____	will copy lines to after cursor position.
DElete <line> ,<line> _____	will delete lines.
Position <line> _____	positions cursor to line.
Tabs Repetitive <incr> _____	Sets tabs at columns 0, incr, 2 * incr, 3 * incr, etc.
Tabs Non-repetitive [<column>] _____	Sets tab at entered columns. (Columns separated by spaces).
Tabs _____	Clear all tabs.
T+ _____	Enable display of tab markers.
T- _____	Disable display of tab markers.
? _____	Display version, filename, and line and memory status. Enter any key to exit.

When loading a new file or exiting the editor you will be prompted to save the current buffer if it has been modified since last saved. Any response other than an 'N' or 'n' will save the old buffer. A response of 'C' or 'c' will allow you to change the file name before saving.

Non-buffered commands (see page 19-9)

Quit _____	Exit editor.	original version
H-19 4 _____	move cursor left one character.	J
6 _____	move cursor right one character.	L
8 _____	move cursor up one line.	I
2 _____	move cursor down one line.	,
5 _____	alternate between left and right of line.	K
0 _____	move cursor 1/2 screen up	O (letter)
.	move cursor 1/3 screen down	
space	move cursor to next identifier.	
HT or cntrl T	move cursor to next tab.	
cntrl R	move cursor backwards to previous tab.	
cntrl S	single character delete. deleting the carriage return will merge lines, with multiple leading spaces compressed into one space.	
cntrl X	delete current line.	original version
9	Overwrite one character.	I (digit one)
H-19 3	overwrite characters until escape.	V
1	enter append mode.	A
7	enter insert mode.	i

Text search commands

Find _____ will position cursor, response is
CR = stop command
Space = continue
M = multiple occurrences on line

Replace _____ will position cursor, response is
CR = stop command
Space = continue
R = Replace and continue
M = multiple occurrences on line
E = entire buffer

Note : When the 'F' prompt appears enter the find pattern - if you enter just a carriage return then this will terminate the command. When the 'R' prompt appears enter the Replace pattern. A '-' is used as a don't care in the find and Replace patterns. The entire buffer option will enable the replacement throughout the rest of the buffer. The multiple occurrences option will allow a match on each occurrence of the find pattern on any one line, else only the first match on a line is recognized. The search will start on the current line, continue to the end of the buffer, start back at the top, and stop on the current line. The cursor will be positioned at the last match when the command terminates.

Insert or append mode

Character entry until an escape is entered. Insert mode will insert characters before the current cursor position. If a carriage return is entered during insert mode then append mode will be entered with the last part of the line carried over until append mode is exited. Append mode will start new lines past the current line with no partial line carried over. In addition the following special characters are recognized :

BS or DEL _____	backspaces over character.
HT or cntrl T _____	forward tab.
cntrl R _____	reverse tab.
cntrl A _____	copies remainder of last line entered.
cntrl X _____	removes all characters entered so far.