QL-Screen Editor

QED SCREEN EDITOR

Introduction

The QED Screen Editor is a powerful yet easy-to-use text editor that runs on a standard QL under QDOS. It was developed on the QL in C (using GST's QC compiler) and its size is around 42Kb. QED can be used to edit program source files for all GST products (such as QC or the QL Macro Assembler) and can also be used to edit(SuperBASIC program sources and the source code for languages from other QL software houses such as Metacomco, Prospero or Computer One.

In style, QED is a reasonably standard text editor which uses the screen as a window into the file being edited. Text can be scrolled (both vertically and horizontally) through the window by using the cursor keys or by command. Facilities are provided to insert, delete and overwrite text, together with extensive search and replace and text block operations.

QED has been designed to be very simple to learn and operate: comprehensive help information and function key menus are provided, and you should be fully able to operate the program without reference to the manual.

How to Run the QED Screen Editor

Before running QED we recommend that you take a backup copy of the program.

To run the QED Screen Editor use either EXEC or EXEC_W, or, if you possess the QL Toolkit, EX or EW. Thus EXEC MDV1_QED will load QED from MDV1 and execute it. Note that if you use EXEC or EX you will have to press CTRL+C to switch control to QED, but you will retain the option of switching back to SuperBASIC to take appropriate action if you encounter an error such as disk (or Microdrive) full.

QED will then prompt you for the name of the file to be edited which may be either a new or existing file: respond with <device> <file> as usual. If you are creating a file you can now commence text input, otherwise QED will prompt for the name of a second file. If you are using Microdrives you must respond with a second <device> <file> but if you are using floppy disks you can choose either a second file name or retention of the existing file name by pressing ENTER.

If two different file names are used, QED will edit from the original file into the new file leaving the original unchanged. If a single file name is used, QED will edit from <file> into <file>_TOP these being renamed <file>_BAK and <file> respectively at the end of the edit. Note that QED also uses the temporary work files named <file>_BOTTOM and <file> _EMP.

The benefit of this method of file handling is that an original version of the file will always be preserved no matter what happens. The penalty is the requirement for sufficient room on the device for a second copy of the file. Other than this there is no restriction on file size, and files much larger than the available RAM space can be edited without problem.

Use of Windows

QED makes full use of QDOS windows. The default screen area used by QED is identical to the default used by QDOS devices CON_ and SCR_, but this size and position can be modified using the GST program WINDOW_MGR (though note that QED will ignore any specification of paper and ink colours and tag data). QED can only be used sensibly in monitor mode (or mode 4 text on a TV - sized screen).

Up to four windows are displayed as follows:

- **Status Window.** This is a black one-line window at the top of the QED screen. This line displays the name of the file being edited, the current tab value, insertion mode and the line and column numbers.

- **Help Window.** If help menus are switched on, a green window will be positioned immediately below the status line. This will contain details of the current command options. If this menu is switched off, the space it occupies is allocated to the text window.

- **Text Window.** This is a white window that holds the text to be edited that will scroll both vertically and horizontally (up to a maximum line length of 158 characters) if necessary.

- **Menu Window.** Even if help information is switched off, a green, single-line menu is always displayed at the bottom of the screen area. This usually displays the currently valid command keys and their actions, but is occasionally used to hold one line prompts or error messages (in the latter case the background colour is changed to red).

Once you are familiar with QED operation, the one-line menu and status line together provide sufficient information for text editing, the help window being useful for the initial learning phase and for occasional reference.

QED Command Structure

All QED commands fall into one of two categories:

- **Function Key Commands.** The more complex commands are invoked via a menu system whose choices are displayed in the single-line menu at the bottom of the screen area. Commands are actioned by depressing a function key in the range F1 - F4 followed by zero or more alphanumeric keys, normally a single alphabetic key. If a sequence of keystrokes is required, menu and/or help information is displayed in the appropriate windows for each stage of the operation.

- **Cursor and Edit Commands.** These commands are concerned with cursor movement (and hence text pan and scroll functions) and character, word and line deletion. They are invoked by a cursor key together with zero or more of the shift keys (ALT, CTRL, SHIFT) depressed simultaneously. A table summarising these commands is given in the CURSOR COMMANDS help window.

A detailed command summary is given below, though it should not be necessary to consult this during normal operation.

Function Key Command Summary

In the table below commands are invoked by a function key in the range F1-F4 followed by the capitalised letter(s) of the words shown in the second and third columns (if any).

Certain commands invoke a further dialogue by prompting for a filename, option selection or Yes/No confirmation. In these cases the action required should be obvious.

F Key	Alpha 1	Alpha 2	Action
F1			help window on/off
F2	Again		repeat last find or replace command
F2	Find		search for a text string
F2	Goto	1	goto predefined mark 1
F2	Goto	2	goto predefined mark 2
F2	Goto	3	goto predefined mark 3
F2	Goto	4	goto predefined mark 4
F2	Goto	Bottom	goto bottom of file
F2	Goto	End	goto end of block
F2	Goto	Previous	goto cursor position before previous command
F2	Goto	Start	goto start of block
F2	Goto	Top	goto top of file
F2	Insert		insert mode on/off
F2	Mark	1	define mark position 1
F2	Mark	2	define mark position 2
F2	Mark	3	define mark position 3
F2	Mark	4	define mark position 4
F2	Quit		abandon edit and retain original file
F2	Replace		find and replace a text string
F2	Save	Continue	save and continue editing current file
F2	Save	New	save current file and reprompt for new edit
F2	Save	eXit	save current file and exit to QDOS
F3	Copy		copy marked block
F3	Delete		delete marked block
F3	End		mark end of block
F3	Hide		delete block markers
F3	Move		move marked block to cursor position
F3	Read		read named file into text at cursor position
F3	Start		mark start of block
F3	Write		write marked block to named file
F4			redisplay the entire QED screen

Note that the ESCape key can be used to abort commands during the command entry sequence.

Cursor and Edit Command Summary

In the table below the character '+' indicates that the key combination must be depressed simultaneously:

Keys	Action
←	move cursor left one character
→	move cursor right one character
↑	move cursor up one line
↓	move cursor down one line
← + SHIFT	move cursor left one word
→ + SHIFT	move cursor right one word
← + ALT	move cursor to start of line
→ + ALT	move cursor to end of line
↑ + ALT	move to previous page
↓ + ALT	move to next page
← + CTRL	delete character left
→ + CTRL	delete character right
← + CTRL + SHIFT	delete word left
→ + CTRL + SHIFT	delete word right
← + CTRL + ALT	delete line

Note that these commands will automatically invoke horizontal and/or vertical scrolling as necessary to ensure that the cursor is always visible.

Error Handling

In general, QED is helpful and easy-to-use by having a well-defined.command interface with few special cases and no obscure features. Because of this, QED assumes that when you invoke a command you mean it, and consequently there are no facilities to undo the action of a command once the command entry sequence has been completed.

When QED detects an error that must be brought to your attention, it does one of two things: either an alarm bleep is sounded (if you type an invalid keystroke) or an error message with a red background is displayed in the menu window. The error messages will indicate what has occurred and what action must be taken.

In the case of an unrecoverable I/0 error, the original version of the file will remain unchanged.

If the disk or Microdrive becomes full during editing, this is reported to you, and the command is usually aborted when you type ESCape. However, there are three commands that are retried when you type ESCape, these are: F2 Save, F3 Read and F3 Write. If you have started QED using the EXEC or EW commands, then you have the option of switching back to SuperBASIC with CTRL+C (in order to make sufficient space available for the file in question) prior to typing ESCape in QED.

To avoid these problems altogether it is advisable to ensure that you have enough room on your disk or Microdrive before starting the edit.

Contents

1. Introduction

The screen editor ED may be used to create a new file or to alter an existing one. The text is displayed on the screen, and can be scrolled vertically or horizontally as required. The size of the program is about 20K bytes and it requires a minimum workspace of 8K bytes.

The editor is invoked using **EXEC** or **EXEC_W** as follows

```
EXEC_W mdv1_ed
```

The difference between invoking a program with **EXEC** or **EXEC_W** is as follows. Using **EXEC_W** means that the editor is loaded and SuperBASIC waits until the editing is complete. Anything typed while the editor is running is directed to the editor. When the editor finishes, keyboard input is directed at SuperBASIC once more.

Using **EXEC** is slightly more complicated but is more flexible. In this case the editor is loaded into memory and is started, but SuperBASIC carries on running. Anything typed at the keyboard is directed to SuperBASIC unless the current window is changed. This is performed by typing **CTRL – C** (pressing the CONTROL key and C together), which switches to another window. If just one copy of ED is running then **CTRL – C** will switch to the editor window, and characters typed at the keyboard will be directed to the editor. A subsequent **CTRL – C** switches back to SuperBASIC. When the editor is terminated a **CTRL – C** will be needed to switch back to SuperBASIC once more. More than one version of the editor can be run concurrently (subject to available memory) if **EXEC** is used. In this case **CTRL – C** switches between SuperBASIC and the two versions of the editor in turn.

Once the program is loaded it will ask for a file name which should conform to the standard Qdos file name syntax. No check is made on the name used, but if it is invalid a message will be issued when an attempt is made to write the file out, and a different file name may be specified then if required. All subsequent questions have defaults which are obtained by just pressing **ENTER**.

The next question asks for the workspace required. ED works by loading the file to be edited into memory and sufficient workspace is needed to hold all the file plus a small overhead. The default is 12K bytes which is sufficient for small files. The amount can be specified as a number or in units of 1024 bytes if the number is terminated by the character K. If you ask for more memory than is available then the question is asked again: The minimum is 8K bytes.

You are next asked if you wish to alter the window used by ED. The default window is normally the same as the window used in the initialisation of ED although this may be altered if required. See Appendix A for details of how to do this. If you type **N** or just press **ENTER** then the default window is used. If you type **Y** then you are given a chance to alter the window. The current window is displayed on the screen and the cursor keys can be used to move the window around. The combination **ALT** and the cursor keys will alter the size of the window although there is a minimum size which may be used. Within this constraint you can specify a window anywhere on the screen, so that you can edit a file and do something else such as run a SuperBASIC program concurrently. When you are satisfied with the position of the window press **ENTER**.

Next, an attempt is made to open the file specified, and if this succeeds then the file is read into storage and the first few lines displayed on the screen. Otherwise a blank screen is provided, ready for the addition of new data. The message "File too big" indicates that more workspace should be specified.

When the editor is running the bottom line of the screen is used as a message area and command line. Any error messages are displayed there, and remain displayed until another editor command is given.

Editor commands fall into two categories – immediate commands and extended commands. Immediate commands are those which are executed immediately, and are specified by a single key or control key combination. Extended commands are typed in onto the command line, and are not executed until the command line is finished. A number of extended commands may be typed on a single command line, and any commands may be grouped together and groups repeated automatically. Most immediate commands have a matching extended version.

Immediate commands use the function keys and cursor keys on the QL in conjunction with the special keys **SHIFT**, **CTRL** and **ALT**. For example, delete line is requested by holding down the **CTRL** and **ALT** keys and then pressing the left arrow key. This is described in this document as **CTRL – ALT – LEFT.** Function keys are described as **F1**, **F2** etc.

The editor attempts to keep the screen up to date, but if a further command is entered while it is attempting to redraw the display, the command is executed at once and the display will be updated later, when there is time. The current line is always displayed first, and is always up to date.

2. Immediate commands

2.1 Cursor control

The cursor is moved one position in either direction by the cursor control keys **LEFT**, **RIGHT**, **UP** and **DOWN**. If the cursor is on the edge of the screen the text is scrolled to make the rest of the text visible. Vertical scroll is carried out a line at a time, while horizontal scroll is carried out ten characters at a time. The cursor cannot be moved off the top or bottom of the file, or off the left hand edge of the text.

The **ALT – RIGHT** combination will take the cursor to the right hand edge of the current line, while **ALT – LEFT** moves it to the left hand edge of the line. The text will be scrolled horizontally if required. In a similar fashion **SHIFT – UP** places the cursor at the start of the first line on the screen, and **SHIFT – DOWN** places it at the end of the last line on the screen.

The combinations **SHIFT – RIGHT** and **SHIFT – LEFT** take the cursor to the start of the next word or to the space following the previous word respectively. The text will be scrolled vertically or horizontally as required. The **TAB** key can also be used. If the cursor position is beyond the end of the current line then **TAB** moves the cursor to the next tab position, which is a multiple of the tab setting (initially 3). If the cursor is over some text then sufficient spaces are inserted to align the cursor with the next tab position, with any characters to the right of the cursor being shuffled to the right.

2.2 Inserting text

Any letter typed will be added to the text in the position indicated by the cursor, unless the line is too long (there is a maximum of 255 characters in a line). Any characters to the right of the text will be shuffled up to make room. If the line exceeds the size of the screen the end of the line will disappear and will be redisplayed when the text is scrolled horizontally. If the cursor has been placed beyond the end of the line, for example by means of the **TAB** or cursor control keys, then spaces are inserted between the end of the line and any inserted character. Although the QL keyboard generates a different code for **SHIFT – SPACE** and **SHIFT – ENTER** these are mapped to normal **SPACE** and **ENTER** characters for convenience.

An **ENTER** key causes the current line to be split at the position indicated by the cursor, and a new line generated. If the cursor is at the end of a line the effect is simply to create a new, empty blank line after the current one. Alternatively **CTRL – DOWN** may be used to generate a blank line after the current, with no split of the current line taking place. In either case the cursor is placed on the new line at the position indicated by the left margin (initially column one).

A right margin may be set up so that **ENTER**s are automatically inserted before the preceding word when the length of the line being typed exceeds that margin. In detail, if a character is typed and the cursor is at the end of the line and at the right margin position then an automatic new line is generated. Unless the character typed was a space, the half completed word at the end of the line is moved down to the newly generated line. Initially there is a right margin set up at the right hand edge of the window used by ED. The right margin may be disabled by means of the EX command (see later).

2.3 Deleting text

The **CTRL – LEFT** key combination deletes the character to the left of the cursor and moves the cursor left one position. If the cursor is at the start of a line then the new line between the current line and the previous is deleted (unless you are on the very first line). The text will be scrolled if required. **CTRL – RIGHT** deletes the character at the current cursor position without moving the cursor. As with all deletes, characters remaining on the line are shuffled down, and text which was invisible beyond the right hand edge of the screen may now become visible.

The combination **SHIFT – CTRL – RIGHT** may be used to delete a word or a number of spaces. The action of this depends on the character at the cursor. If this character is a space then all spaces up to the next non-space character on the line are deleted. Otherwise characters are deleted from the cursor, and text shuffled left, until a space is found. The **CTRL – ALT – RIGHT** command deletes all characters from the cursor to the end of the line. The **CTRL – ALT – LEFT** command deletes the entire current line.

2.4 Scrolling

Besides-the vertical scroll of one line obtained by moving the cursor to the edge of the screen, the text may be scrolled 12 lines vertically by means of the commands **ALT – UP** and **ALT – DOWN**. **ALT – UP** moves to the previous lines, moving the text window up; **ALT – DOWN** moves the text window down moving to lines further on in the file. The **F4** key rewrites the entire screen, which is useful if the screen is altered by another program besides the editor. Remember that you can switch out of the editor window and into some other job by typing, **CTRL – C** at any point, assuming that there is another job with an outstanding input request. SuperBASIC will be available only if you entered the editor using **EXEC** rather than **EXEC_W**. If there is enough room in memory you can run two versions of ED at the same time if you wish.

2.5 Repeating commands

The editor remembers any extended command line typed, and this set of extended commands may be executed again at any time by simply pressing **F2**. Thus a search command could be set up as the extended command, and executed in the normal way. If the first occurrence found is not the one required, typing **F2** will cause the search to be executed again. As most immediate commands have an extended version, complex sets of editing commands can be set up and executed many times. Note that if the extended command line contains repetition counts then the relevant commands in the group will be executed many times each time the **F2** key is pressed.

3. Extended commands

Extended command mode is entered by pressing the **F3** key. Subsequent input will appear on the command line at the bottom of the screen. Mistakes may be corrected by means of **CTRL – LEFT** and **CTRL – RIGHT** in the normal way, while **LEFT** and **RIGHT** move the cursor over the command line. The command line is terminated by pressing **ENTER**. After the extended command has been executed the editor reverts to immediate mode. Note that many extended commands can be given on a single command line, but the maximum length of the command line is 255 characters. An empty command line is allowed, so just typing **ENTER** after typing **F3** will return to immediate mode.

Extended commands consist of one or two letters, with upper and lower case regarded as the same. Multiple commands on the same command line are separated from each other by a semicolon. Commands are sometimes followed by an argument, such as a number or a string. A string is a sequence of letters introduced and terminated by a delimiter, which is any character besides letters, numbers, space, semicolon or brackets. Thus valid strings might be

```
/happy/     !23 feet!       :Hello!:     "1/2"
```

Most immediate commands have a corresponding extended version. See the table of commands for full details (section 4).

3.1 Program control

The command **X** causes the editor to exit. The text held in storage is written out to file, and the editor then terminates. The editor may fail to write the file out either because the file name specified when editing started was invalid, or because the Microdrive becomes full. In either case the editor remains running, and a new destination should be specified by means of the **SA** command described below. Alternatively the **Q** command terminates immediately without writing the buffer; confirmation is requested in this case if any changes have been made to the file. A further command allows a 'snapshot' copy of the file to be made without coming out of ED. This is the **SA** command. **SA** saves the text to a named file or, in the absence of a named file, to the current file. For example:

***SA**/mdv2_savedtext/
or
***SA**

This command is particularly useful in areas subject to power failure or surge. It should be noted that **SA** followed by **Q** is equivalent to the **X** command. Any alterations made between the **SA** and the **Q** will cause ED to request confirmation again; if no alterations have been made the program will be quitted immediately with the file saved in that state. **SA** is also useful because it allows the user to specify a file name other than the current one. It is therefore possible to make copies at different stages and place them in different files.

The **SA** command is also useful in conjunction with the **R** command. Typing **R** followed by a file name causes the editor to be re-entered editing the new file. The old file will be lost when this happens, so confirmation is requested (as with the **Q** command) if any changes to the current file have been made. The normal action is therefore to save the current file with **SA**, and then start editing a new file with **R**. This saves having to load the editor into memory again, and means that once the editor is loaded the Microdrive containing it can be replaced by another.

The **U** command 'undoes' any alterations made to the current line if possible. When the cursor is moved from one line to another, the editor takes a copy of the new line before making any changes to it. The **U** command causes the copy to be restored. However the old copy is discarded and a new one made in a number of circumstances. These are when the cursor is moved off the current line, or when scrolling in a horizontal or vertical direction is performed, or when any extended command which alters the current line is used. Thus **U** will not 'undo' a delete line or insert line command, because the cursor has been moved off the current line.

The **SH** command shows the current state of the editor. Information such as the value of tab stops, current margins, block marks and the name of the file being edited is displayed. Tabs are initially set at every three columns; this can be changed by the command **ST**, followed by a number n, which sets tabs at every n columns. The left margin and right margin can be set by **SL** and **SR** commands, again followed by a number indicating the column position. The left margin should not be set beyond the width of the screen. The **EX** command may be used to extend margins; once this command is given no account will be taken of the right margin on the current line. Once the cursor is moved off the current line, margins are enabled once more.

3.2 Block control

A block of text can be identified by means of the **BS** (block start) and **BE** (block end) commands. The cursor should be moved to the first line required in a block, and the **BS** command given. The cursor can then be moved to the last line wanted in the block, by cursor control commands or in any other way, such as searching. The **BE** command is then used to mark the end of the block. Note, however, that if any change is made to the text the block start and block end become undefined once more. The start of the block must be on the same line, or a line previous to, the line which marks the end of the block. A block always contains all of the line(s) within it.

Once a block has been identified, a copy of it may be moved into another part of the file by means of the **IB** (insert block) command. The previously identified block is replicated immediately after the current line. Alternatively a block may be deleted by means of the **DB** command, after which the block start and end values are undefined. It is not possible to insert a block within itself.

Block marks may also be used to remember a place in a file. The **SB** (show block) command resets the screen window on the file so that the first line in the block is at the top of the screen.

A block may also be written to a file by means of the **WB** command. The command is followed by a string which represents a file name. The file is created, possibly destroying the previous contents, and the buffer written to it. A file may be inserted by the **IF** command. The file name given as the argument string is read into storage immediately following the current line.

3.3 Movement

The command **T** moves the screen to the top of the file, so that the first line in the file is the first line on the screen. The **B** command moves the screen to the bottom of the file, so that the last line in the file is the bottom line on the screen if possible.

The commands **N** and **P** move the cursor to the start of the next line and previous line respectively. The commands **CL** and **CR** move the cursor one place to the left or one place to the right, while **CE** places the cursor at the end of the current line, and **CS** places it at the start.

It is common for programs such as compilers and assemblers to give line numbers to indicate where an error has been detected. For this reason the command **M** is provided, which is followed by a number representing the line number which is to be located. The cursor will be placed on the line number in question. Thus **M1** is the same as the **T** command. If the line number specified is too large the cursor will be placed at the end of the file.

3.4 .Searching and Exchanging

Alternatively the screen window may be moved to a particular context. The command **F** is followed by a string which represents the text to be located. The search starts at one place beyond the current cursor position and continues forwards through the file. If found, the cursor is placed at the start of the located string. To search backwards through the text use the command **BF** (backwards find) in the same way as **F**. **BF** will find the last occurrence of the string before the current cursor position. To find the earliest occurrence use **T** followed by **F**; to find the last, use **B** followed by **BF** . The string after **F** and **BF** can be omitted; in this case the string specified in the last **F**, **BF** or **E** command is used. Thus

```
*F /wombat/
*BF
```

will search for wombat' in a forwards direction and then in a reverse direction.

The **E** (exchange) command takes a string followed by further text and a further delimiter character, and causes the first string to be exchanged to the last. So for example

E/wombat/zebra/

would cause the letters 'wombat' to be changed to 'zebra'. The editor will start searching for the first string at the current cursor position, and continues through the file. After the exchange is done the cursor is moved to after the exchanged text. An empty string is allowed as the search string, specified by two delimiters with nothing between them. In this case the second string is inserted at the current cursor position. No account is taken of margin settings while exchanging text.

A variant on the **E** command is the **EQ** command. This queries the user whether the exchange should take place before it happens. If the response is **N** then the cursor is moved past the search string. If the response is **Y** or **ENTER** then the change takes place; any other response (except **F2**) will cause the command to be abandoned. This command is normally only useful in repeated groups; a response such as **Q** can be used to exit from an infinite repetition.

All of these commands normally perform the search making a distinction between upper and lower case. The command **UC** may be given which causes all subsequent searches to be made with cases equated. Once this command has been given then the search string 'wombat' will match 'Wombat', 'WOMBAT', 'WoMbAt' and so on. The distinction can be enabled again by the command **LC**.

3.5 Altering text

The **E** command cannot be used to insert a new line into the text, but the **I** and **A** commands may be used instead. The **I** command is followed by a string which is inserted as a complete line before the current line. The **A** command is also followed by a string, which is inserted after the current line. It is possible to add control characters into a file in this way.

The **S** command splits the current line at the cursor position, and acts just as though an **ENTER** had been typed in immediate mode. The **J** command joins the next line on to the end of the current one.

The **D** command deletes the current line in the same way as the **CTRL – ALT – LEFT** command in immediate mode, while the **DC** command deletes the character at the cursor in the same way as **CTRL – RIGHT**.

3.6 Repeating commands

Any command may be repeated by preceding it with a number. For example,

4E/slithy/brillig/

will change the next four occurrences of 'slithy' to 'brillig'. The screen is verified after each command. The **RP** (repeat) command can be used to repeat a command until an error is reported, such as reaching the end of the file. For example,

RP E/slithy/brillig/

will change all occurrences of 'slithy' to 'brillig'.

Commands may be grouped together with brackets and these command groups executed repeatedly. Command groups may contain further nested command groups. For example,

RP (/f/toves/;3(IB;N))

will insert three copies of the current block whenever the string 'toves' is located.

Note that some commands are possible, but silly. For example,

RP SR 60

will set the right margin to 60 ad infinitum. However, any sequence of extended commands, and particularly repeated ones, can be interrupted by typing any character while they are taking place. Command sequences are also abandoned if any error occurs.

4. Command list

4.1 Immediate commands

F2	Repeat last extended command
F3	Enter extended mode
F4	Redraw screen
LEFT	Move cursor left
SHIFT-LEFT	Move cursor to previous word
ALT-LEFT	Move cursor to start of line
CTRL-LEFT	Delete left one character
CTRL-ALT-LEFT	Delete line
RIGHT	Move cursor right
SHIFT-RIGHT	Move cursor to start of next word
ALT-RIGHT	Move cursor to end of line
CTRL-RIGHT	Delete right one character
CTRL-ALT-RIGHT	Delete to end of line
SHIFT-CTRL-RIGHT	Delete word to right
UP	Move cursor up
SHIFT-UP	Cursor to top of screen
ALT-UP	Scroll up
DOWN	Move cursor down
SHIFT-DOWN	Cursor to bottom of screen
ALT-DOWN	Scroll down
CTRL-DOWN	Insert blank line

4.2 Extended commands

/ s / indicates a string, **/ s / t /** indicates two exchange strings and **n** indicates a number.

A / s /	Insert line after current
B	Move to bottom of file
BE	Block end at cursor
BF	Backwards find
BS	Block start at cursor
CE	Move cursor to end of line
CL	Move cursor one position left
CR	Move cursor one position right

CS	Move cursor to start of line
D	Delete current line
DB	Delete block
DC	Delete character at cursor
E / s / t /	Exchange s into t
EQ / s / t /	Exchange but query first
EX	Extend right margin
F / s /	Find string s
I / s /	Insert line before current
IB	Insert copy of block
IF / s /	Insert file s
J	Join current line with next
LC	Distinguish between upper and lower case in searches
M n	Move to line n
N	Move cursor to start of next line
P	Move cursor to start of previous line
Q	Quit without saving text
R / s /	Re-enter editor with file s
RP	Repeat until error
S	Split line at cursor
SA / s /	Save text to file
SB	Show block on screen
SH	Show information
SL n	Set left margin
SR n	Set right margin
ST n	Set tab distance
T	Move to top of file
U	Undo changes on current line
UC	Equate U/C and l/c in searches
WB / s /	Write block to file s
X	Exit, writing text back.

Appendix A:
Changing the default window

The window to be used can be altered as part of the initialisation sequence. If this option is not required then the default window is used. This is initially the same as the window used during the start of the program, but if required the default window may be altered permanently by patching the programs. This is useful where a certain window size and position is always required and means that the window does not have to be positioned correctly each time the program is run.

The program INSTALL is supplied on the distribution Microdrive for this purpose. It is run by the command

```
LRUN mdv1_install
```

The program starts by asking whether the default window is to be set up for TV or monitor mode. The minimum window size is greater in TV mode because the characters used are larger. You should answer **T** if you are setting the default for use with TV mode and **M** if you are setting it for use with monitor mode. Note that the current mode in use is of no consequence.

The standard window will appear on the screen and can be moved by means of the cursor keys and altered in size by means of **ALT** cursor keys. This is similar to the mechanism used when altering the window during normal program initialisation. Once the window is in the right place and of the desired size, press **ENTER**.

The program now asks for the name of the file which is to be modified. If you wished to alter the editor then the file would probably be something like **'mdv1_ed'** . The next item requested is the name of the program. When a new job is running on the QL, it has a name associated with it. This can be inspected by suitable utilities. The name is six characters long, and whatever is typed here is used as the name and forced to the correct length. The name is of little importance except for job identification.

The INSTALL program will then modify the file specified. INSTALL can be run as many times as you like to alter the default window of the editor.