DynaStar 2.2

A High Speed Screen Editor
for the
OS9 and FLEX 9 Operating Systems

User's Manual

by

Allan G. Jost

DYNASOFT SYSTEMS LIMITED
P.O. Box 51
Windsor Junction, Nova Scotia
Canada B0N 2V0

1.0 INTRODUCTION

DynaStar was written for all those microcomputer users who, like myself, have high speed video terminals and are tired of editors designed for 110 baud teletypes.

DynaStar is dedicated to the idea that text editors should be easy to use, easy to learn, and easy to adapt to a wide variety of terminals. We have tried to produce a program which meets all of these criteria, and we hope the menu system provided will help you to make effective use of its features without constantly thumbing through the manual. If we have failed, we would appreciate your comments.

DynaStar was designed to meet the needs of both the word-processing user and the programmer. For use as a word-processor, it contains several features which are intended to support facilities in the companion DynaForm print formatter, available separately. These features are only mentioned briefly in this manual and are explained in detail in the DynaForm User's Manual.

1.1 A Guided Tour

The best way to learn about DynaStar is to use it. Refer to Chapter 5 to install it on your system, then just type "Ds" and use the menus to guide your experimentation and familiarization with the many features it provides. If you have not yet installed a "gotoxy" module, it will configure itself automatically, asking you only to tell it how to get to "home" position (by typing the appropriate keys) during its initialization sequence. It will also ask for a key sequence to clear to end-of-line. If your terminal can't do this, just type a carriage return. This procedure will be eliminated later when you have installed your "gotoxy" module.

If all has gone well, you should be presented with a copyright message and a menu (the "files" menu) offering several options. For now just type the letter "N" to start a New edit session. On the top line of the screen you will be asked for a file name. Pick one that is not already on your disk, and type it in, followed by a carriage return. After a brief pause, you should be presented with a new menu (the main editing menu).

Before doing anything else, sit back and get familiar with the organization of the information at the top of the screen. The very top line should contain the name of the file you have just established, and a word at the right hand end telling you that you are in "INSERT" mode. This line is called the "status" line, and its main purpose is to constantly remind you what file you are working on, and what "modes" you are in. It is also where most operator prompts will appear when required by certain commands, and where you should look for error messages when your terminal "beeps" at you.

The next five lines contain a menu of commands available to you for editing text, and below this will be a line full of minus signs and exclamation marks. This is the "ruler" line which serves the dual purpose of separating the text editing area from the menu area, and telling you where your margins and tabs are.

To learn how to edit text, you must have text to edit. Type in a few lines of anything you like, just pretending the keyboard is a standard typewriter. Fill the whole screen if you like.

After you have enough on the screen to make it interesting, sit back and have a look at the menu. It will tell you very briefly about a lot of single letter commands you can give to manipulate the text you have just typed. Most of the command letters have a "^" in front of them. What this tells you is that you must hold the "ctrl" key down while typing that letter in order to convert it from a "typable" letter to an editor command. The first two lines of the menu tell you how to move the cursor around. To get the feel of it, try typing ^E ("control" E) and watch the cursor. It should move straight up. Then try typing ^S, ^D, and ^X (any order) and watch which way the cursor goes. Note that these four keys are arranged on the keyboard in a diamond pattern which reflects the cursor movement they cause. Learn to know and love this diamond. You will use it a lot.

You will notice that the cursor will not move to where there is no text. This is one of the fundamental rules of this editor. The cursor can only be placed over characters which are actually in your text buffer. To get it past the last character in the buffer, you must type in more characters. To get it past the last character in a line, you must type more characters into the line. When it is apparently sitting on an empty space at the extreme right end of a line, it is actually sitting on an invisible carriage return character. If you type more characters while it is sitting on a carriage return, it will insert the characters in the line, pushing the carriage return ahead of it.

Now look on the fourth line of the menu (look for the word "DELETE"). Place the cursor over any character on your screen and type ^G. The character should disappear and the line should close in from the right. Now press the key marked "DEL" (or "RUBOUT" or "BKSP" if you can't find a "DEL"). The character immediately to the left of the cursor should disappear, and the cursor should back up one position. Place the cursor over the first character of any word and type ^T. You should get the idea. Now retype the word that just disappeared and watch it being put back where it came from. For your last trick, get the cursor over the very last character of a line (it should be an invisible carriage return) and delete it with a ^G. Now, to split those two lines apart again, type a carriage return.

That should be enough to get you started. Play with the other commands on the menu to see what they do.

You will observe on the last line of the menu that you can call up additional menus by typing ^B, ^K, ^P or ^Q. Try typing ^Q and watch the menu area. You will be presented with a whole new list of commands. To execute one of these commands you will have to type another letter. (For the second character it doesn't matter if you hold down the "CTRL" key.) ^B, ^K, ^P and ^Q are "prefix" commands, meaning they are actually the first letter of a two-keystroke command. If you don't want to try any of these commands just now, press the space bar.

To wrap up your first lesson, make sure you are on the main editing menu and type ^N (the menu calls this one "doNe"). You should find yourself back on the "files" menu, which is where you started in the first place. Type the letter "D". After a few seconds, you should see a listing of the file names in your current data directory. One of them should start with "SCRATCH". Now type "X" (for "eXit"). You should get the your standard system prompt. Type "dir" (OS9) or "cat" (FLEX). Notice that "SCRATCH" has disappeared, but that there is now a new file there with the name you picked way back at the beginning of this exercise.

DynaStar is written entirely in Dynasoft Pascal, which is a p-code implementation of an integer subset of Pascal. Many people believe that p-code is too slow for applications like screen editors. Judge for yourself.

2.0 DESCRIPTION OF EDITING COMMANDS

All of the commands available in the DynaStar Editor are described below. They are covered a menu at a time, in approximately the order they should be learned by the beginning user.

We will start with the main editing menu. Most basic editing can be done from this menu.

It is first necessary to explain the two basic operating modes of DynaStar. When you first start an editing session, the editor is in "INSERT" mode, which you can verify from the right end of the status line. In this mode, any printable character you type is inserted in your text buffer at the current cursor location, "pushing" any text to the right of the cursor ahead as it goes. It is possible to operate all of the time in this mode, since text can be replaced by deleting old text and then typing in new text to replace it.

The other basic mode is "OVERTYPE". You can put the editor in this mode by typing the sequence ^QO (this notation means "hold the CTRL key down while pressing the letter Q, then type the letter O"). The word "OVERTYPE" will appear on the end of the status line when you do this. In this mode, printable characters that you type are typed "over" and replace the characters that were there before. However, OVERTYPE mode will not replace a carriage return character: if you "overtype" to the end of a line so that the cursor reaches an existing carriage return, the carriage return will be pushed ahead, permitting the line to be extended as if the editor was in INSERT mode.

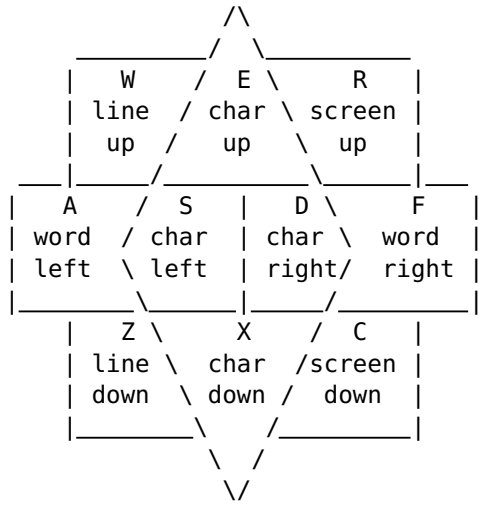
OVERTYPE mode is most useful to correct simple spelling errors in text. The biggest problem is that you tend to forget you are in OVERTYPE mode, which sometimes leads to confusion. It is just as easy to correct isolated spelling errors from INSERT mode by first deleting the offending character and then typing in the correction, and this method usually involves fewer keystrokes than entering OVERTYPE mode, typing a one character correction, and then re-entering INSERT mode. At Dynasoft we rarely use OVERTYPE for this reason. ^QO is a "toggle" command: it switches back and forth between OVERTYPE and INSERT mode each time you type it.

It is possible in either "INSERT" or "OVERTYPE" mode to type past the right edge of your screen. When this happens, the cursor will move down to the beginning of the line below, and you can continue typing this "line extension" on the new screen line. No carriage return is entered into the text buffer, and the line will print correctly if listed on a printer capable of printing lines wider than the screen. Inserting or deleting characters in a line occupying more than one screen line will shift characters in any extension lines as required. In this sense, "logical" lines are distinct from "screen" lines. However, for the purposes of the cursor movements described below, it is screen

lines which are significant, and vertical movements of the cursor will not distinguish between logical and screen lines. In a similar manner, the line delete functions ^Y and ^QY operate on screen lines, not logical lines.

2.1 Basic Cursor Motion Commands

Most of the control keys on the main editing menu are arranged in a geometric pattern at the left of the keyboard which reflects the function performed by each command. This pattern is centered around the basic cursor motions of up, down, left and right, which are arranged in a simple diamond pattern.



The keys adjacent to the four basic cursor motion keys perform functions related to the basic motions. For instance, while ^S moves the cursor left one character, ^A moves it left one word. Similarly, ^D moves right one character and ^F moves right a word. On the top line, ^E moves the cursor up one character (to the line above), and ^R moves up one screenful. On the other side of E, ^W does not move the cursor, but it scrolls upward one line in the file (the screen actually scrolls down). In the same way, on the bottom line, ^X moves the cursor down one character, ^C moves down a screenful, and ^Z scrolls one line downwards.

In addition to the cursor motion keys there are several keys for deleting text. ^G deletes the character under the cursor. DEL (or BKSP or ^H) deletes the character to the left of the cursor. ^T deletes the word to the right of the cursor, and ^V deletes the word to the left (useful when entering text to back up and retype a mispelled word). The sequence ^QY deletes everything on the line to the right of the cursor, and ^Y deletes the entire line.

Prefixing any of the four basic motion keys (^E,^S,^D,^X) with a ^Q moves the cursor as far as it will go on the screen in

the direction corresponding to the basic key. Prefixing ^R and ^C with a ^Q moves the cursor to the very top and bottom of the edit buffer respectively.

Some commands prompt for additional input if necessary on the status line of the display. Commands requiring a single-letter response continue as soon as the letter is typed, and do not care if the reply is typed in upper or lower case. Some commands require a multiple-character response such as a file name or search string. The response to these prompts can usually be up to 40 characters in length, and is terminated by pressing the RETURN key (which is not considered part of the response string). During the typing of such responses, typing errors can be corrected by using either the DEL, ^S, ^H, or BACK SPACE keys to back up to the offending character. Typing an "empty" response (just a RETURN) to one of these prompts will generally cancel the command.

Error conditions detected during operation of any of the editing commands will usually result in an error message on the status line, in conjunction with an audible beep from the terminal. The ESC key must be pressed to exit from an error condition. Pressing any other key will result in another beep. (On the Radio Shack Color Computer the BRK key is used instead of the ESC key.)

Descriptions of the editing commands on the main editing menu follow:

^S	cursor left	Moves the cursor one character to the left if possible, moving it to the end of the previous line and scrolling the screen down if either action is necessary.
^D	cursor right	Moves the cursor right one character, moving to the beginning of the next line if necessary, and scrolling the screen up if the cursor is on the last line of the screen.
^A	left word	Moves the cursor left to the first letter of the preceding word. For purposes of this command and others involving "word boundaries", the cursor also stops on all punctuation marks and at the beginning and end of logical lines.
^F	right word	Moves the cursor to the beginning of the next word to the right, as described above.
^I or TAB	tab right	Move the cursor right to the next tab stop. In OVERTYPE mode, this is a simple cursor movement. In INSERT mode this command inserts blanks to get to the next tab stop,

		pushing any text ahead of the cursor over in the process. (On the Color Computer the "right arrow" (->) is used as a TAB key.)
^E	cursor up	Moves the cursor up to the line above its current location. The screen is scrolled down if necessary to allow this. If the line above ends to the left of the cursor location, the cursor is moved left to the end of that line.
^X	cursor down	Moves the cursor down to the same location, if possible, on the line below its current location, as described above.
^R	up screen	Scrolls upward in the text buffer one "screenful", leaving two lines of overlap with the old display. If the cursor is in the overlap region, it remains over the same character, otherwise it will land on the last character displayed on the new screen.
^C	down screen	Scrolls downward in the text buffer one screenful, as described above.
^Z	scroll up	Scrolls the screen up one line, leaving the cursor over the same character if possible.
^W	scroll down	Scrolls the screen down one line, leaving the cursor over the same character if possible.
^G	delete char	Deletes the character under the cursor, closing in the line from the right.
BKSP or DEL	delete left	Deletes the character immediately to the left of the cursor, backing up the cursor and closing in the line in the process. (On the Color Computer, this is the "left arrow" (<-) key.)
^V	delete word left	Deletes the characters to the left of the cursor, up to and including the first character of the current word. This command is most useful for correcting a word that has just been typed, if it would be quicker to retype the word than to back up to the character(s) in error and then space back to where you were.
^T	delete word right	Deletes the characters in the current word to the right of the cursor.

^Y	delete line	Deletes the entire screen line on which the cursor is sitting.
^J or LF	line insert	Inserts a carriage return at the cursor location, and leaves the cursor just before the CR. Any text to the right of or below the cursor is pushed down one line. If this is issued when the cursor is on the left margin it has the effect of inserting a blank line into which new text can be typed. If it is issued in the middle of a line, it splits the line, leaving the cursor on the end of the top piece.
^Q	supplementary prefix	Prefix command bringing up a supplementary menu containing additional cursor movement and find/replace commands.
^B	block prefix	Prefix command bringing up the block operations menu.
^K	formatting prefix	Prefix command bringing up the formatting and tab setting menu.
^P	Print control prefix	Prefix command for inserting print control characters for the DynaForm Print Formatter. The sequence "^P." is used to display additional menus summarizing DynaForm's "dot" commands.
^N	doNe	Return to the "files" menu.
ESC	macro learn or error escape	This key is used to enter or leave "macro learn mode". Keyboard macros are fully explained in the chapter "KEYBOARD MACROS". The ESC key is also used to exit from error conditions. (On the Color Computer, this is the BRK key.)

2.2 The ^Q Supplementary Prefix

^Q is a prefix command which may be thought of as the first character of a two-character command sequence. To execute the command a second character must be typed after the ^Q. The second command letter can be typed with or without the CTRL key and in upper or lower case. If the prefix was typed accidentally, it may be cancelled by pressing the space bar. Most of the ^Q commands are supplementary cursor moves.

^QE	top of screen	Moves the cursor to the first character location on the screen: (i.e. to the upper left corner).
^QX	bottom screen	Moves the cursor to the last character on the screen.
^QR	top of buffer	Moves the cursor to the first character in the text buffer, re-displaying the screen if necessary.
^QC	bottom buffer	Moves the cursor to the last character in the text buffer, re-displaying the screen if necessary.
^QS	left margin	Moves the cursor to the left margin on the current line.
^QD	right end	Moves the cursor to the last character on the current line.
^QP	next Paragraph	Moves the cursor to the beginning of the first word of the next paragraph. A new paragraph is recognized by any of the following conditions: 1) one or more blank lines. 2) a line indented past the "wrap margin". 3) a line beginning with a period (a DynaForm "dot" command).
^QT	top of block	Moves the cursor to the first character of the currently marked block, if a block is still active. Block marking is discussed under the ^B block prefix menu.
^QB	bottom block	Moves the cursor to the last character of the currently marked block, if a block is active.
^QF	find string	Prompts for a search string on the status line (up to 40 characters, terminated by a carriage return) and searches the text buffer, from the cursor location forward, for a string matching the search string. If such a string is found, the cursor is moved to the first character of the found string, and the

	screen is re-displayed if necessary. If the string is not found, the message "NOT FOUND" is displayed on the status line, and pressing the ESC key will return the cursor to its original position.
^QL repLace string	Prompts for a search string as in the "find" command, and then prompts for a second string which is to replace the next occurrence in the text buffer of a string matching the search string. It first performs a search in the same manner as the "find" command, producing the same "NOT FOUND" message if the string is not found. If a match is found, the message "Replace? (y/n)" is displayed on the status line, and the cursor is placed on the first character of the found string. (If the editor is operating without a "gotoxy" module, the cursor will be on the status line.) At this point, if "y" or "Y" or "^Y" is typed, the found string will be deleted and replaced with the replacement string, otherwise the replace is cancelled and the cursor is left on the found string. This command can be used to delete a string by providing an "empty" (RETURN only) replacement string.
^QA Again	Repeats the last "find" or "replace" operation without re-prompting for the search/replacement strings. The same error messages may result, and confirmation is requested before any replacement is performed.
^QY delete line right	Deletes all characters in the current line to the right of the cursor.
^QO Overtype on/off	Turns OVERTYPE mode on if currently in INSERT mode; turns INSERT mode on if currently in OVERTYPE mode.

2.3 ^B Block Commands

The ^B prefix activates a set of commands for manipulating blocks of text. To perform a block operation it is first necessary to "mark" a block of characters by placing the cursor over the first and last characters and "marking" these characters as either "top" or "bottom". When either the top or bottom of a block has been marked, but not the other end, the message "BLOCK PARTLY MARKED" temporarily replaces the mode reminders on the right end of the status line, and when both ends of a block are marked, the message "BLOCK MARKED" appears in the same area.

It is not possible to perform normal editing (i.e. inserting or deleting) when block markers are set, and any attempt to do so will result in an error message. This is because the markers are kept as offsets into the text buffer, and these offsets would be altered by inserting or deleting characters in the buffer. Normal operation can be restored by carrying through the intended block operation, if it is a type which results in the "unmarking" of the block, or by explicitly issuing an "Unmark" command.

It is possible to place the "top" and "bottom" markers in the wrong sequence in the text buffer ("bottom" above "top"). If this is done, the "BLOCK PARTLY MARKED" message will appear, as if only one of the markers was set, and any attempt to execute a block operation will result in an error message. The condition can be corrected by re-setting either one of the markers. The ^QT and ^QB commands can be used to verify the location of block markers.

Although the "Insert from file" command is included in the ^B menu because of its similarity to other block operations, it does not require that a block be marked to operate.

^BT mark Top — "Marks" the current cursor location as the beginning of a block. The status line will show either "BLOCK MARKED" or "BLOCK PARTLY MARKED" depending on whether the other end of the block has been correctly marked. The editor will not insert or delete text while block markers are active.

^BB mark Bottom — "Marks" the current cursor location as the end of a block, as described above.

^BU Unmark block — Removes the current block markers and restores normal operation of insert and delete functions.

^BC Copy block — A copy of the text block currently marked is inserted in the text buffer at the current cursor location, leaving the original text block and its markers intact. This command is used to replicate a block of text at other

		locations in the buffer. The markers are left intact so that the text may be copied to additional locations if desired. If the block markers are no longer needed, they must be explicitly Unmarked.
^BM	Move block	The text block currently marked is deleted from its original location and inserted at the current cursor position. The block markers are discarded.
^BK	Kill block	The text block currently marked is deleted from the text buffer, and the markers are cancelled.
^BW	Write block to file	Prompts for a file name, attempts to create a new output file with that name, and copies the text in the currently marked block to the output file. The file is then closed. The original text is left unaltered, and remains marked.
^BI	Insert from file	Prompts for a file name, attempts to open the file for input, and reads the contents of the file into the text buffer, inserting at the current cursor location. If the contents of the file will not fit in the space available, an error message is printed and the operation aborted.

2.4 ^K Formatting and Tab Commands

The ^K menu contains commands relating to the automatic formatting of text and the setting and clearing of tabs.

^KC Center line — Centers the printable characters of the line containing the cursor between the left and right margins, leaving the cursor at the end of the centered string. Note that the presence of "print control" characters in a line will effect the centering of the line: if you wish to place control characters in a centered line, center it first and then insert the control characters.

^KW Word Wrap on/off — Turns word wrap ON if it is OFF, or OFF if it is ON. If word wrap is ON, the word "WRAP" appears on the status line. When word wrap is active, an attempt to type past the current right margin (with the cursor "pushing" a single carriage return ahead of it) will result in an automatic movement to a new line. If the word being typed does not fit on the original line, the entire word is moved down to the next line to ensure that the original line will fit within the margins. If the wrap margin has been set (see below) to the right of the left margin, the new line is automatically indented to the wrap margin, otherwise the new line starts at the left margin. Word Wrap permits the typing of entire paragraphs without the use of the RETURN key, while ensuring that the text will fit within the currently set margins. Turning Wrap OFF also turns Justify OFF (see below).

^KJ Justify on/off — Turns Justify mode ON if it is OFF, and OFF if it is ON. Turning Justify ON also turns Word Wrap ON, and shows the words "JUSTIFY" and "WRAP" on the status line. When Justify mode is active, automatic word wrap operations will also pad lines with extra spaces between the words so that the lines will always end exactly at the current right margin, producing a "clean" rather than a "ragged" right edge. In performing the expansion of lines, spaces are never inserted to the left of the wrap margin, so that the edge under the wrap margin will also be uniform. This feature was used to produce the paragraph you are now reading. The presence of "print control" characters in a justified line will make the line appear too long on the screen, but the line will print

correctly, since DynaStar correctly compensates for their presence.

^KD inDent on/off — Turns auto-indent mode ON or OFF. Auto-indent mode is designed as a convenience for programmers writing code in structured languages such as Pascal. When indent mode is ON, typing a carriage return in INSERT mode will automatically indent the next line to the position of the first non-blank character on the line above, preserving the indentation of structured code. At this point, the indentation can be increased using the space or tab keys, decreased using the DEL key, or maintained by simply typing away. Turning indent mode ON turns WRAP and JUSTIFY OFF.

^KA Adjust paragraph — Readjusts the text from the cursor location to the end of the next paragraph to conform to the currently set margins. This process involves the stripping of existing carriage returns and excess blanks which may have been inserted by a previous word wrap or margin justification operation. When stripping blanks, an attempt is made to leave at least one blank between words and two blanks after periods, question marks, and exclamation marks. A word wrap operation is then performed to make the text conform to the current margins, including automatic indentation to the wrap margin. If JUSTIFY mode is ON, the text is also justified to produce a clean edge at the right margin. If there is text to the left of the wrap margin on the first line of the section being readjusted, it is not altered. However, all text on lines below the first is reformed by the operation, regardless of its position on a line. The operation is stopped at the end of the current paragraph, as defined for the ^QP command, and the cursor is left on the first character of the first non-empty line past this point. This makes it very convenient to repeat the Adjust operation on the next paragraph in sequence.

^K TAB
or
^KI set tab — Sets a tab in the column corresponding to the current cursor position. This is visually indicated by placing an exclamation mark in the ruler line at the position above the cursor. If the wrap margin is at this location, it is converted to a tab.

^K- or ^KM	reMove tab	Clears the tab directly above the cursor in the ruler line by replacing it with a "-" sign. This command can also be used to remove the wrap margin.
^KV	set wrap margin	Sets the wrap margin at the cursor column by displaying the letter "W" in the ruler line directly above the cursor. Word wrap operations will automatically indent new lines to this margin. Tab operations consider the wrap margin to be equivalent to a standard tab.
^KR	set right margin	Defines the current right margin to be at the column corresponding to the cursor location, and truncates or extends the ruler line as required. If the ruler line is extended by this operation, standard tabs are generated every five columns (at columns 1, 6, 11, 16, etc.). It is not possible to define a right margin beyond the width of the screen. The right margin setting is used only by the word wrap and margin justification operations.
^KK	Kill all tabs	Removes ALL tabs fom the ruler line. If this is done accidentally, the fastest way to regenerate standard tabs is to move the right margin to column one, and then extend the ruler line by moving the margin back to its original column.

2.5 ^P Print Control Commands

The ^P prefix is used to insert print control characters in text, and to bring up a series of menus summarizing the "dot" commands for the DynaForm print formatter.

Typing a letter after the ^P prefix results in the insertion of a special character pair at the cursor location. The character pair consists of a special form of the character "^", followed by the capital of the letter typed. The "^" is not a standard "^" but is in fact a "^" with the high order (sign) bit set to distinguish it from the ordinary form. (It will generally display as a standard "^".) The DynaForm print formatter takes special action when this sequence is found in a file. The pairs "^D", "^B", and "^U" have special meaning described below; any other pair results in the sending of a standard control character to your printer in the place of the pair. This feature can be used to activate special features of certain printers such as the Epson.

As with the other prefix menus, typing a space after a ^P will abort the supplementary command and return you to the main editing menu. In addition to this, however, typing a period (".") after a ^P will display special menus containing a summary of the dot commands for the DynaForm print formatter. There are approximately 5 of these special menus, and you can see them in turn by typing additional periods, or return to the main editing menu by typing any other character.

The following ^P sequences are interpreted by DynaForm to provide special effects at print time:

^PB	Boldface on/off	Turns Boldface ON or OFF. Any text between a pair of these sequences will be printed in boldface. The method of producing the boldface effect on a printer will vary, but in general it results in multistriking, sometimes with a small shift between passes to produce a heavy effect, depending on the printer used.
^PD	Double-strike on/off	Turns Double-strike ON or OFF. Any text between a pair of these sequences will be printed twice (with no shift), generally resulting in crisper type without the bold effect caused by ^PB.
^PU	Underline on/off	Turns Underline ON or OFF. Text between a pair of these sequences will be underlined. Only non-blank characters will be underlined.

The effect of any of the three above commands can span more than one line, remaining in effect until another sequence turns the feature off again. The effects can also be combined: you can underline sections of boldface text, for instance, although only the text will be boldface in this case, but not the underlining.

You will notice if editing with JUSTIFY ON that print control characters will cause extension of lines containing them, appearing to disrupt the justified right margin. These lines will print correctly, however, because the control characters have "zero width" on a printer. DynaStar automatically adjusts its formatting to compensate for the extra width used up by these characters when displayed on a terminal.

3.0 KEYBOARD MACROS

DynaStar contains a facility which permits the user to re-define any control character to mean virtually any other character or sequence of characters. This permits the complete remapping of the control functions on the main editing menu, and it permits the building of special complex commands such as global search and replace. At a simpler level, macros can be used to perform "boiler plate" operations such as inserting a copyright notice with a single keystroke.

Once a macro has been assigned to a control character, typing that control character will always cause execution of the macro regardless of the context. This will sometimes require special care. As an example, suppose the control character ^Y has been redefined as a macro: you would then have to be careful not to answer a "y/n?" prompt with the CTRL key pressed, because inadvertent typing of a ^Y would trigger the macro instead of giving an affirmative response to the prompt.

In general, a macro is simply an expansion of the effect of a control key, sending a predefined string of characters to the editor in the place of the keystroke invoking the macro. If you define the ^L key to be a macro containing the character string "abc", then every time you subsequently type a ^L the editor will act as if you had typed "abc".

Macros are defined by entering "macro learn" mode, which can be done from both the "files" menu and the main editing menu by typing ESC (or BRK on the Color Computer). At this time the message "Macro learn" will appear on the status line. The next character you type should be the control character you wish to define as a macro (or else another ESC to cancel macro learn mode if you got there by accident). If you type a control character, it will be echoed on the status line in the usual form used on menus: as a capital letter preceded by a caret (^). At this point you can now type in the characters which are to be the body of the macro. You can type virtually any characters into a macro, including other control characters, which will again be displayed in the standard "caret" form. The only character which cannot be part of a macro is the ESC character. Typing an ESC terminates a macro definition, at which time it will disappear from the status line. The number of characters in a macro definition is arbitrary, but the total number of characters in all of your macros combined cannot exceed 400.

You can view your macros from the "files" menu with the "L" command. This command asks you whether you want to see a list of your currently defined macros (answer "?" for this option), or you want to see the contents of a particular macro (answer the prompt with the key corresponding to the desired macro). You can also redefine any macro by simply defining it again; in this case you should bear in mind that the previous definition space is not actually reclaimed, and if you do this too often in a single session you will overflow the macro definition area.

Macros are best explained with the aid of an example. Suppose you wanted to "move a command up" from one of the two-keystroke menus to the main menu: A good example would be the "find/replace Again" command (^QA), since there are many occasions when it would be convenient to invoke this command with a single keystroke. The following keystroke sequence would assign the ^L key to this function:

ESC ^L ^Q A ESC

After defining this macro, you can execute the ^QA command at any time from the main menu by typing ^L. Be careful, however, about typing ^L from other menus, because you might cause strange results. Macros are dangerous and should be treated with respect.

As a simpler example, let's just redefine one control key to be equivalent to another. We'll do something silly: swap the meaning of the ^S and ^D keys so they move the cursor in the opposite direction. The macro definition sequences required are:

ESC ^S ^D ESC
ESC ^D ^S ESC

The result of the above definitions will probably be total confusion, since these cursor movement commands will now work in an unnatural way, but the example serves to illustrate one more important point: macros cannot invoke macros. If a control character is contained in a macro, and that control character is itself a macro, this fact is ignored and the effect of the control character is defined by its original function as defined in this manual. If you think about it, this is the only way it would be possible to actually redefine the entire set of control commands.

There are two control characters which have special meaning within macros. The first is ^U. Whenever ^U is encountered during execution of a macro, the editor accepts one character from the keyboard in the place of the ^U. The second special character is ^\. When a macro execution encounters a ^\, the macro restarts at the first character in the macro definition. It is sort of like a GO TO statement, and in the same way as the infamous GO TO statement, it is very dangerous and must be used carefully.

We will finish the discussion of macros with a couple of useful examples: two forms of a global search and replace macro. The first is a global replace command of the type that searches the entire edit buffer for a particular match string, and replaces each instance of the match string with something else, but subject to confirmation from the operator at each occurrence. The macro definition is as follows:

ESC ^U ^Q A ^U ^\ ESC

To use this macro you first execute a standard ^QL replace command so that the ^QA ("Again") command will repeat it. Then, instead of using ^QA to repeat the replace, type ^U. This will cause execution of a ^QA command, prompt for confirmation, accept the user's response in place of the ^U imbedded in the macro, and then loop back and do it again automatically. This brings up another question: how do you stop a looping macro? The answer is that a macro aborts whenever an error message is generated, which will happen in the above example as soon as the search reaches the end of the edit buffer and causes a "NOT FOUND" message. The other way to terminate a macro is to type ESC at a point in the macro where a ^U causes the editor to read from the keyboard. In the above example, you would answer the "replace? (y/n)" prompt with "Y" to do the replace, "N" to reject the replace, or "ESC" to abort the macro.

Macros also terminate when a cursor move out of the edit buffer is attempted.

The last example is a global replace without operator confirmation. The macro definition is:

ESC ^O ^Q A Y ^\ ESC

This macro is executed the same way as the previous example: by first using a standard repLace command (^QL), and then repeating it automatically by typing ^O. In this case the macro executes the ^QA ("Again") command as before, but always gives a "Y" response to the confirmation prompt, and then repeats the process until the end of the buffer is reached. There is no way to terminate this macro from the keyboard: it must run its course.

Several control characters have deliberately been left unassigned in DynaStar to make keys available for use as macros: these are ^L, ^O, and ^U. There are also four non-alphabetic keys which can generate valid control characters suitable for macro use: ^\, ^], ^^, and ^_. The character ^[is not useable because it is equivalent to ESC, and ^M should not be used because it is really a CR. DEL is not actually a control character and it cannot be used as a macro.

As a final note, you will probably find macros tedious to use if "help" is enabled, because a lot of time will be wasted by the editor displaying menus for any prefix commands contained in your macros. The short answer to this complaint is that you should not attempt to use macros until you are thoroughly familiar with the operation of DynaStar, and by that time you will probably be operating with help OFF anyway.

4.0 THE "FILES" MENU

The "files" menu should perhaps have been discussed first instead of last since it is the first menu you see when you enter DynaStar. It is, however, also the last menu you use when you are finished editing a file, and you must return to it from the main editing menu by typing ^N ("doNe"). It is tempting to regard it as another "prefix" menu with the "prefix" ^N, but it is not really a true prefix menu, since you must deliberately execute commands to get OFF it again. It consists mainly of utility commands such as specifying a new file to edit, moving on to the next section of a file, and cleaning up at the end of a session. The "files" menu is the only part of DynaStar where there is a noticeable difference between the OS9 and FLEX versions.

N	edit New file	This command is used to establish the name of a file to be edited, and initiate editing of that file. It prompts for the name of a file on the status line, and attempts to open an existing file by that name from the current working directory. If the open is successful it reads text into the edit buffer until it is approximately 5/6 full or end-of-file is reached, and then enters the main editing menu with the cursor on the first character in the file. The file name is displayed on the status line as a reminder to the operator. If the open operation is unsuccessful, DynaStar assumes that you want to create a new file, and sends you to the main editing menu with an empty buffer. In both cases, it creates an output "SCRATCH" file to contain the fruits of your editing efforts until you decide whether you want to keep or discard what you have done.
O	edit Old file	This command is similar to the "N" command, except that the desired file must already exist for the command to be successful.
W	Write file	This command is used to tell DynaStar that you are finished with the current file you are working on and would like to Write it out to your directory but would like to stay in the editor so you can work on another file. If the current file is a new one, it simply writes out the contents of the text buffer to the SCRATCH file, closes it, and renames it to the name you specified on the last "N" command. If you were editing an existing file, it writes out the contents of the buffer to the SCRATCH file, copies any unprocessed part of the original file to "SCRATCH", deletes your old file, and renames

"SCRATCH" as above. It then clears the text buffer and waits for a new command.

If an "output file" error arises during operation of this or the "More" command, the SCRATCH file will be in an indeterminate state, and it would be wise to abandon it and return to the original file. The error may have resulted from a disk overflow, and it might be possible to remedy the situation by deleting unneeded files using the Shell command (OS9 version only). However, it will not be possible to repeat the "W" command, and the only way of recovering the text in the text buffer is to use the write command from the ^B block menu.

X write and eXit This command performs the same function as the Write command, except that when it is finished it exits to the operating system instead of remaining in the editor. It can also be used when no file is active to terminate the editor and exit to the operating system.

A Abandon file This command is used when you have really fouled up your edit and want to go back to where you were before you started. It first prompts you to make sure you are really sure you want to throw away all the work you have just done. If the answer is "Y" (yes) it deletes the SCRATCH file and clears the edit buffer. If you were editing an old file you will still have it the way it was before you started, and if you were creating a new file it will be gone.

E re-Enter This command is used to re-enter the main editing menu to continue an editing session if you have temporarily entered the "files" menu to do some utility operation such as change the "help" state or examine or clean up some files. It can also be used to enter the edit mode when no file has been established to experiment or to do a short item which you intend to print right away directly from the editor. In this situation, if you decide you want to save your text, you can use the write command from the ^B block menu.

D Directory Displays the names of the files in your current disk directory.

H Help on/off — Turns HELP mode on or off. When HELP is ON, menus are displayed at the top of the screen to tell you what commands are available to you at all times. When HELP is OFF, all menu displays are suppressed except the "files" menu, so that you will have most of your screen area available for editing. H is a toggle command that alternately turns HELP ON and OFF if you execute it repeatedly. You get no indication of the state of the help flag except the presence or absence of menus. HELP defaults to ON.

S Shell command (OS9 only) — This powerful (and dangerous) command prompts you for a one-line shell command and then executes it, returning to the editor when it is finished. Any edit currently in progress is undisturbed. It is possible to execute the shell command as a concurrent process by ending the command line with an ampersand ("&"). This is especially useful if you want to make a listing of a file you have just finished editing while you continue editing another. It is also possible to temporarily exit from DynaStar and enter an interactive session with shell by answering the prompt with an empty command line. This will result in the standard system prompt and you can then use any commands you like provided there is enough free memory available to execute them. When you want to return to your edit session, answer an "OS9:" prompt with an end-of-file character (normally ESC).

It is possible to use the Shell command to reinitiate DynaStar to edit a second file while you are still in the middle of an edit session. DynaStar will create a unique SCRATCH file for the new session because the new incarnation will have a new process id. When the second invocation of DynaStar terminates, you will be returned to your original session with everything , including the cursor position, the way you left it.

When executing shell commands from the editor, there are several things you should be extremely careful about. It is very important that you do not tamper with the SCRATCH file if one is active, and that you do not alter any file which you are in the middle of editing. It would also be very dangerous to remove or swap the disk containing your working directory, since there are probably files open on it, and

		indescribable damage could result to these files or to other files if you re-entered the editor with the wrong disk mounted. The DynaForm formatting program should not be run as a concurrent task if the document to be printed contains operator prompts. Such prompts would interfere with proper screen updating if an edit is also in progress.
F	FLEX command (FLEX only)	This very dangerous command prompts you for a one-line FLEX command and then executes it, returning to DynaStar when it is finished. This command is disabled when a file is open for editing. It should be used only to execute standard FLEX commands which execute completely from the "command" area at $C100. Executing any command which loads in or uses the "user" RAM area could have disastrous results. This command will be most useful for doing minor housekeeping tasks such as deleting or renaming a file.
C	Change directory (OS9 only)	This command prompts you for a new working directory name, and then re-assigns it as your current directory for the duration of your edit session. You should not do this in the middle of a file edit, because the editor would lose track of its files. Note that it is not possible to use the Shell command to change the directory.
M	More	This command is used when you have run out of space in the edit buffer or want to move on to a new section of the file you are working on which has not yet been read into the buffer. It writes out all of the text buffer up to but not including the line the cursor is on and then reads in as much more of your old file as is possible, setting aside approximately 1/6 of the buffer to leave room to insert text from the keyboard. This makes it possible to edit very large files in sections. Note that with both this command and the New command it is possible that the last line in the edit buffer might not have been completely read in, and it is wise in this situation to refrain from editing the incomplete line until it has been completely read in by the next More command.

P	Print buffer (OS9 only)	This command invokes the DynaForm print formatter to print the contents of the edit buffer. It requires that you have DynaForm, and it is implemented using a "pipe", so it will only work on OS9 systems with a pipe processor. It is most useful to do a quick proof of a short document or for the production of letters without bothering to save them on disk first. The "P" command should not be used if the document being printed contains DynaForm "dot" commands requiring an operator response. Note that a document can also be printed by calling DynaForm via the Shell command, provided that the document has first been saved on disk, and the above restriction would not apply then.
ESC	macro learn	Typing ESC from the "files" menu enters "macro learn" mode. This is described fully in Chapter 3: Keyboard Macros.
L	List macros	The "L" command is used to display a list of currently defined keyboard macros, or to list the contents of a particular macro. When L is typed, a prompt is displayed on the status line. To display a particular macro, answer by typing the appropriate control character. To display a list of active macro definitions, type "?".

5.0 INSTALLATION AND TERMINAL REQUIREMENTS

5.1 OS9 Installation

The DynaStar distribution disk contains at least two program modules. There is a module "DS" which is the actual editor, and a module "PINTERP" which is a copy of the Dynasoft Pascal p-interpreter (the editor itself is in p-code). If the DynaForm print formatter has been purchased with the package, there will also be a module called "DF", which is also in p-code. Each of these files should be copied to your standard execution directory. If you already have a copy of Dynasoft Pascal on your system, you should replace your "PINTERP" module with the one from the DynaStar disk. The new interpreter is required to run Version II of DynaStar, and it will run any p-code programs generated by Dynasoft Pascal Release 1.5 or earlier. The editor automatically loads the interpreter when it is run. The best way to install a gotoxy module is to put it in the execution directory separately and include a command to load it in your startup file. It is also possible to include it in your copy of the file "DS" if you wish.

5.2 FLEX Installation

The FLEX distribution disk contains up to four files which must be copied to your "system" disk. Three of these must be installed; the fourth is optional but desirable. The three required files are DS.CMD, DYNASTAR.SYS, and INTERP15.SYS. These are the base command file, the editor itself (in p-code), and the Dynasoft Pascal p-code interpreter. DS.CMD is the program which you call from the command line, and it contains the input/output interface and code to load and run the p-code for the editor and the interpreter. The programs load in low memory starting at $0100, and all remaining user RAM from the end of the programs to MEMEND is used for a text buffer and work variables. DynaStar requires at least 24K of RAM to run.

The distribution disk also contains a series of files with names like ADM3.TXT and ADM3.GXY. These are the source and object code respectively for "gotoxy" modules for several popular terminals. DynaStar can operate without them, but will work much better if you install an appropriate "gotoxy" module for your terminal. If one of the supplied modules supports your terminal, all you have to do is rename the appropriate ".GXY" file to "GOTOXY.SYS" and copy it to your system disk. If one does not fit, you will have to write your own. The source code for all the supplied modules is provided in case one of them is close, and to provide examples for guidance. For further information on writing a "gotoxy" module, see section 5.4.

5.3 Terminal Requirements

DynaStar requires a video terminal which displays at least 64 characters on a line, and it should preferably be capable of

operating at 9600 baud. It must have at least rudimentary cursor addressing (the ability to "home" the cursor to the top left corner will do). To operate in the self- configuring mode, the OS9 "terminal descriptor" or FLEX "ttyset" parameters must correctly state the exact number of lines displayable on the screen, and the terminal must "scroll" the screen upwards when a line feed is issued from the bottom line. If the terminal has a command to erase from the cursor to end-of-line it will greatly speed up screen updating. DynaStar can handle screen formats as large as 132 characters by 48 lines.

5.4 The Gotoxy Module

DynaStar uses a module called 'gotoxy' find out how to do cursor addressing and establish other parameters of the terminal being used with it. This module must be a 6809 subroutine module, and the organization is as follows:

```
2 bytes (entry)   bra gtxy
1 byte            fcb lines-per-page
1 byte            fcb chars-per-line
1 byte            fcb scroll (1 if terminal scrolls, 0 if not)
1 byte            fcb length1 (of clear-to-end-of-line sequence)
length1 bytes     clear-to-end-of-line-sequence
1 byte            fcb length2 (of initialization sequence, or 0)
length2 bytes     terminal initialization sequence if needed
0 or more bytes   editor "startup" string
1 byte            fcb 0  (null to terminate startup string)
gtxy              (actual code for the gotoxy function)
```

The main entry is passed the desired x position in the x register, and the y position in register b. The upper left corner of the screen is (1,1).

The byte at entry+4 is a flag to indicate if the terminal scrolls when a line feed is issued on the bottom line.

The byte string at entry+6 is used by the editor to clear a line from the cursor position to end-of-line. This greatly speeds up the screen display routines. If your terminal lacks this function, the byte at entry+5 must be set to 0.

The byte just after the clear-to-end-of-line sequence is a byte count for an optional initialization sequence which the editor is to send to the terminal when it is first started up. If no initialization is required, it should be set to zero.

The "editor startup string" is an optional string of characters, terminated by a null ($00), which if included is passed to DynaStar as initial input as if it was typed from the keyboard. Use of this string makes it possible to automatically pre-define keyboard macros, re-map the main editing keyboard, or set up special modes such as margin justification. If a startup string is not included, at least one null must be placed here to indicate an empty string.

If no gotoxy module is found, DynaStar executes the self-configuration routine and prompts for the home-cursor and clear-to-end-of-line sequences. It checks the terminal descriptor (OS9) or ttyset parameters (FLEX) to find lines-per-page. If this is 16 it assumes a 64 character line, otherwise it assumes an 80 character line.

Gotoxy modules for several common terminals are included on the distribution disk, in both source and object format. If your terminal is not included in this set, you may find the source code for one of the others helpful in writing your own. If you write a gotoxy module for a terminal not included on the distribution disk, we would appreciate seeing it so we can include it on future disks to save other customers the trouble of writing their own.

When running under the OS9 operating system, the gotoxy module must be loaded into memory prior to running Ds. This can be accomplished at boot time by including an appropriate load command in the startup file, or it can be accomplished by making a new "Ds" file containing both the modules "Ds" and "gotoxy".

Under FLEX, the gotoxy module must be a file called "GOTOXY.SYS" on the system drive. If you have to write your own gotoxy, you must use the program "SAVESYS.CMD" (also on the distribution disk) to generate the executable copy, since it is in a special relocatable format. Note that you have no idea where DynaStar will load your gotoxy module in memory, so you must write it in position-independent code. Use the supplied source code examples for guidance. Once you have assembled your module (to a ".BIN" file), note the first and last assembled addresses, and use the sequence below to install it on your system disk. (The example assumes that the module assembled from $0100 to $0132):

```
GET 1.GOTOXY.BIN
SAVESYS 0.GOTOXY.SYS 0100 0132
```


6.0 RUNNING DYNASTAR

6.1 Running under OS9

DynaStar is called from OS9 by typing a command of the form:

Ds [initial-file] [:alternate-gotoxy] [#memsize]

The parameters on this command line are optional. The first is the name of an initial file to edit. If provided, this parameter will be processed as if it was entered as part of a New command from the "files" menu, and is simply a short cut to speed up initialization. The second parameter is the name of an optional alternate gotoxy module which is to be used instead of the standard one. This feature is provided for multi-user systems to permit operation of more than one terminal type on the same system. If, for example, you are running a multi-user system with mostly SWTPC terminals, but you wanted to occasionally run DynaStar with an ADM3 terminal, you could install a second gotoxy module called "adm3" and invoke DynaStar with the command:

Ds :adm3

The third parameter is the usual OS-9 memory size parameter. If it is omitted, DynaStar allocates a memory area of 8K bytes, which provides about 6K of space for the text buffer, adequate for small edits only. DynaStar can use up to 32K bytes of memory space, but on a typical 56K single-user system, 25K is the practical limit. A typical command line would probably be:

Ds #25k

If too much memory is requested, it is possible that the editor would work just fine until an attempt is made to save an edit file, at which time it might fail because there is not enough memory to rename the SCRATCH file. If this happens, it is possible to recover by leaving the editor and executing the rename operation from the keyboard. Ask for a little less memory the next time.

DynaStar is re-entrant and the same copy of the editor can be shared by several users on a multi-user system, subject to memory availability. On the typical 56K system, two users can operate with approximately 12K of memory each.

If you intend to use the DynaForm print formatter from DynaStar you must leave an additional 13k of memory to load and run the "DF" program.

6.2 Running under FLEX

DynaStar is called from FLEX with a command line of the form:

Ds [initial-file]

The parameter is optional. If supplied, it is processed by DynaStar as if it was entered as part of a standard New command from the "files" menu, and it simply specifies the name of the first file to be edited.

When editing large files, it is wise to be conscious of the amount of free space left on your working disk. There should be at least as much space as the size of the file you want to edit, since DynaStar creates a SCRATCH file of the same size during the course of an edit. This implies that it is impossible to edit a file larger than 1/2 of the capacity of a disk. When preparing a large document to be formatted with DynaForm, you will find it helpful to organize it as a series of smaller sub-files, and tie them together at print time using the DynaForm .FI (file insert) command.

APPENDIX I: CHANGES FROM DYNASTAR VERSION 2.0

The most important changes incorporated in Version 2.2 of DynaStar are as follows:

1) OVERTYPE mode has been changed so that "overtyping" past the end of a line will extend the line instead of replacing the carriage return and thus appending the current line to the next.

2) The Print command has been removed from the "files" menu. It was not practical to provide a version of this command which would work under both level 1 and 2 of OS9, and under FLEX. OS9 users can accomplish the same task by saving the text buffer to disk and invoking DynaForm via the Shell command, with the advantage that the printing can proceed concurrently. FLEX users will have to exit the editor to print a file.

3) The optional parameter "initial-file" has been added to the command line. In the OS9 version it is now necessary to signal an alternate gotoxy module by preceding its name with a colon.

4) Several minor bugs have been corrected. These relate to the processing of ^U in an "editor startup string", and the catching of illegal TAB and delete operations when block marks are active.

APPENDIX II: SUMMARY OF DYNAFORM DOT COMMANDS

```
.BP n               Begin Page #n
.CP n               Conditional Page
.PN n               set Page Number
.PL n               set Page Length [66]
.IG                 Ignore to next 'dot'
..                  Comment line
.HE text            Header
.FO text            Footer
.MT n               Margin at Top [3]
.HM n               Header Margin [2]
.MB n               Margin at Bottom [8]
.FM n               Footer Margin [2]
.PO n               Page Offset [8]
.SS                 set Single Space
.MS n               Multiple space [2]
.SP n               blank Space n lines
.OP n               OverPrint next n lines
.FI pathlist        File Insert
.MA xx              start Macro xx
.ME                 End Macro
.XX                 do macro xx
.DM text            Display Message
.IFE yy             If Even page do .yy
.IFO yy             If Odd page do .yy
.DXt text           inDeX entry (t=tag)
.XAt n              print indeX Alpha for tag t
.XNt n              print indeX Numeric for tag t
.SV name,text       Set Variable <name> to 'text'
.PV name,message    Prompt for Variable <name>
.DF pathlist        open Data File for mail-merge
.RV name1,name2,... Read Variables <name1>, <name2>, etc.
```

Notes:
Document auto-repeats when data file open, until eof.
[] is default value for n, # in header/footer = page#

APPENDIX III : SUMMARY OF DYNASTAR COMMANDS

"Files" menu:

		page
A	Abandon edit buffer without changing file	22
C	Change working directory	24
D	Directory	22
E	reEnter editing mode	22
ESC	Enter/Leave macro learn mode	25
F	execute FLEX command	24
H	Help on/off	23
L	List current macros	25
M	More: get next file section	24
N	edit New file	21
O	edit Old file	21
P	Print edit buffer	25
S	execute Shell command	23
W	Write edit buffer & update file	21
X	eXit editor (Writes if file active)	22

Main editing menu:

^	Hold "ctrl" key, type letter	
^A	Cursor left word	6
^B	Prefix: Block commands	8
^C	Down screenful	7
^D	Cursor right	6
^E	Cursor up	7
^F	Cursor right word	6
^G	Delete character under cursor	7
^H	Delete character left (same as BKSP)	7
^I	Tab right (same as TAB)	6
^J	(same as line feed)	8
^K	Prefix: Format and Tab commands	8
^N	doNe: Return to "files" menu	8
^P	Prefix: Print control	8
^Q	Prefix: supplementary commands	8
^R	Up screenful	7
^S	Cursor left	6
^T	Delete word right	7
^V	Delete word left	7
^W	Scroll down	7
^X	Cursor down	7
^Y	Delete line	8
^Z	Scroll up	7
BKSP	Delete character left	7
DEL	Delete character left	7
ESC	Enter/terminate macro learn mode	8
LF	Insert cr after cursor	8
TAB	Tab right (inserts spaces in INSERT mode)	6

INDEX

TABLE OF CONTENTS