

A.P.P.L.E.

PRESENTS:

Higher Text Plus

Apple PugetSound Program Library Exchange

Higher Text Plus
by
Darrell and Ron Aldrich

Higher Text program copyright©
1980 by Darrell and Ron Aldrich
All Rights Reserved

Higher Text documentation
Entire contents copyright© 1981
by Val J. Golding

Apple][and Applesoft are
registered trademarks of
Apple Computer, Inc.

I Quick Reference Command Summary

A. Higher Text Generator Commands

Command	CHR\$	Description	Page
CALL 3072	-	Cold start if HT generator in memory	7
CALL 3075	-	Warm start if HT generator in memory	7
Ctrl Q	17	Select Large Font	8
Ctrl E	5	Select Expanded Small Font display mode	10
Ctrl R	18	Select Regular Small Font display mode	10
Ctrl T	20	Select Tall Small Font display mode	10
Ctrl W	23	Select Wide Small Font display mode	10
Ctrl Y	25	Select Bold Face mode	9
Ctrl Z	26	Deselect Bold Face mode	9
Ctrl C+	3+	Color assignment prefix	11
Ctrl B0	2+0	Assign current color to background	12
Ctrl A*	-	Upper case shift lock	13
Ctrl S*	-	Lower case shift lock	13
Ctrl X*	-	Cancel current line being typed	13
Ctrl H	-	Move cursor one position left	13
Ctrl I	-	Move cursor one position right	13
Ctrl J	-	Move cursor one line down	13
Ctrl K	-	Move cursor one line up	13
Ctrl L	12	Clear screen, home cursor	14
Ctrl N	14	Clear from cursor to end of current line	14
Ctrl O	15	Clear from cursor to end of page	14
Ctrl F	6	Scroll text up one line	14
Ctrl P	16	Select smooth scroll mode	14
Ctrl @	0	Deselect smooth scroll mode	14
Ctrl B+	2+	Background mode assignment prefix	26
Ctrl V*	-	Select or deselect key click function	28
Ctrl G	7	Bell	28

* May be used in immediate mode (from the keyboard) only

+ Indicates suffix required

Quick Reference Command Summary

B. Higher Text Character Editor Commands

Command	Description	Page
SYSTEM COMMANDS		
Ctrl L[ower]	Select/deselect Large Font lower case	15
1	Select Small Font 1	16
2	Select Small Font 2	16
3	Select Small Font 3	16
Ctrl C[ancel]	Exit editor, return to BASIC	17
RUN	Return to editor from BASIC	16
E[ditor]	Run the editor not currently in use	17
D[isk]	Display disk catalog	18
L[oad]	Load a font from disk	18
O[utput]	Output (save) a font to disk	18
I	Move system cursor up one line	18
J	Move system cursor left one position	18
K	Move system cursor right one position	18
L	Move system cursor down one line	18
P[lace]	Place a character in the edit window	18
[return]	Return a character from the edit window to the font	18
R[estart]	Clear edit window, restart editing	19
X	Set a dot in the matrix (X marks the spot)	19
C[lear]	Clear a dot in the matrix	19

EDIT WINDOW COMMANDS

W	Move edit cursor up one line	19
A	Move edit cursor left one position	19
S	Move edit cursor right one position	19
Z	Move edit cursor down one line	19
Ctrl W	Scroll character in edit window up one line	19
Ctrl A	Scroll character in edit window left one position	19
Ctrl S	Scroll character in edit window right one position	19
Ctrl Z	Scroll character in edit window down one line	19
Y	Delete horizontal line at edit cursor position	20
G	Delete vertical line at edit cursor position	20
H	Insert vertical line at edit cursor position	20
B	Insert horizontal line at edit cursor position	20

HIGHER TEXT TABLE OF CONTENTS

I	QUICK REFERENCE COMMAND SUMMARIES	2
A.	Higher Text Generator Commands	3
B.	Higher Text Character Editor Commands	4
II	OVERVIEW	4
	So now that I've Got it, What Do I do?	4
	What's This About More Than Six Hi-Res Colors?	5
	Pretty Printing	5
	Identical Editors are Like Identical Twins	5
	Up, Up and Away (Higher Text in Action)	6
III	THE TEXT GENERATOR	7
	Introduction	7
A.	CHARACTER PARAMETERS	8
1.	Font sizes	8
a.	Large Font	8
b.	Small Font Display Modes	9
2.	Character Colors	9
3.	Background Colors	12
4.	Upper and Lower Case	13
B.	CURSOR CONTROLS	13
C.	SCREEN FUNCTIONS	13
1.	Clearing Functions	14
2.	Scrolling Functions	14
3.	TEXT Command	14
IV	THE CHARACTER EDITORS	14
	Introduction	14
A.	EDIT SYSTEM FEATURES	16
1.	Entering and Exiting the Editors	16
2.	Loading and Saving Fonts and Generators	17
3.	Moving Characters To and From the Edit Window	18
4.	Swapping Large Font characters	18
B.	CHARACTER EDITING FEATURES	19
1.	Clearing the Edit Window	19
2.	Setting and Clearing Dots in the Matrix	19
3.	Using the Edit Cursor	19
4.	Scrolling the Character in the Edit Window	19
5.	Inserting and Deleting Lines	20
V	TECHNICAL DATA	20
A.	MEMORY ALLOCATION	20
1.	Memory Map	20
2.	Memory Usage by Higher Text and Friends	20
3.	High Resolution Screen Pointers	22
B.	FONT CONFIGURATIONS	22
1.	Normal Configurations	22
2.	Non-Standard Configurations	23
3.	Calculating Font Indexes	23
4.	Garbage Fonts	24
C.	INTEGER BASIC HI-RES ROUTINES	24
D.	CTRL B[ackground] SCREEN FUNCTIONS	25

VI GENERAL INFORMATION	28
A. CREATING YOUR OWN PROGRAM DISKETTE	28
B. SUBSIDIARY FUNCTIONS	28
1. Key Click	28
2. Control G (bell)	28
C. SUPPORTING PROGRAMS	29
1. Program Line Editor	29
2. LOMEM: (The &LOMEM: Utility)	29
3. Billboard	29
a. Slide Maker	30
b. Climber	31
4. Higher Fonts	31
5. Demo Programs	32
VII GLOSSARY	35
VIII USING A GENERATOR IN COMMERCIAL PROGRAMS	37

I Quick Reference Command Summary

A. Higher Text Generator Commands

Command	CHR\$	Description	Page
CALL 3072	-	Cold start if HT generator in memory	7
CALL 3075	-	Warm start if HT generator in memory	7
Ctrl Q	17	Select Large Font	8
Ctrl E	5	Select Expanded Small Font display mode	10
Ctrl R	18	Select Regular Small Font display mode	10
Ctrl T	20	Select Tall Small Font display mode	10
Ctrl W	23	Select Wide Small Font display mode	10
Ctrl Y	25	Select Bold Face mode	9
Ctrl Z	26	Deselect Bold Face mode	9
Ctrl C+	3+	Color assignment prefix	11
Ctrl B0	2+0	Assign current color to background	12
Ctrl A*	-	Upper case shift lock	13
Ctrl S*	-	Lower case shift lock	13
Ctrl X*	-	Cancel current line being typed	13
Ctrl H	-	Move cursor one position left	13
Ctrl I	-	Move cursor one position right	13
Ctrl J	-	Move cursor one line down	13
Ctrl K	-	Move cursor one line up	13
Ctrl L	12	Clear screen, home cursor	14
Ctrl N	14	Clear from cursor to end of current line	14
Ctrl O	15	Clear from cursor to end of page	14
Ctrl F	6	Scroll text up one line	14
Ctrl P	16	Select smooth scroll mode	14
Ctrl @	0	Deselect smooth scroll mode	14
Ctrl B+	2+	Background mode assignment prefix	26
Ctrl V*	-	Select or deselect key click function	28
Ctrl G	7	Bell	28

* May be used in immediate mode (from the keyboard) only

+ Indicates suffix required

Quick Reference Command Summary

B. Higher Text Character Editor Commands

Command	Description	Page
SYSTEM COMMANDS		
Ctrl L[ower]	Select/deselect Large Font lower case	15
1	Select Small Font 1	16
2	Select Small Font 2	16
3	Select Small Font 3	16
Ctrl C[ancel]	Exit editor, return to BASIC	17
RUN	Return to editor from BASIC	16
E[ditor]	Run the editor not currently in use	17
D[isk]	Display disk catalog	18
L[oad]	Load a font from disk	18
O[utput]	Output (save) a font to disk	18
I	Move system cursor up one line	18
J	Move system cursor left one position	18
K	Move system cursor right one position	18
L	Move system cursor down one line	18
P[lace]	Place a character in the edit window	18
[return]	Return a character from the edit window to the font	18
R[estart]	Clear edit window, restart editing	19
X	Set a dot in the matrix (X marks the spot)	19
C[lear]	Clear a dot in the matrix	19

EDIT WINDOW COMMANDS

W	Move edit cursor up one line	19
A	Move edit cursor left one position	19
S	Move edit cursor right one position	19
Z	Move edit cursor down one line	19
Ctrl W	Scroll character in edit window up one line	19
Ctrl A	Scroll character in edit window left one position	19
Ctrl S	Scroll character in edit window right one position	19
Ctrl Z	Scroll character in edit window down one line	19
Y	Delete horizontal line at edit cursor position	20
G	Delete vertical line at edit cursor position	20
H	Insert vertical line at edit cursor position	20
B	Insert horizontal line at edit cursor position	20

II OVERVIEW

Higher Text is a complete package for producing text on either high-resolution graphic screen. The Higher Text diskette includes 13 complete upper and lower case 96 character ASCII character sets. Examples of the included fonts may be found in Section VI. Characters may be printed in several sizes, shapes and colors, with selectable background colors and modes. Two editors are provided for modifying any individual character, or designing your own entire character set. Higher Text allows you to print on the hi-res screens with PRINT, VTAB and HTAB [TAB] statements, exactly as if you were printing to the regular text screen.

So now that I've got it, what can it do?

It would almost be simpler to list what Higher Text can *not* do. Higher Text is without question, the finest program of its kind written for the Apple] [. Higher Text comes with 13 character sets, also known as "fonts", identified on the catalog by the suffix ".F". In addition, separate diskettes are available that contain many more fonts, thus it is possible to utilize any one or more of these fonts in your Applesoft or Integer Basic hi-res program, without needing to create your own.

In this introductory section, we will describe briefly some of the characteristics of Higher Text and their respective applications. Higher Text is a complex *system* which effectively converts your hi-res screen to function as a regular text screen, but with the added facilities of up to ten colors and multiple character sizes. It is composed of two main bodies, the character *generator* and the character *editor*.

The character *generator* is that part of the Higher Text System which interprets the BASIC program lines, converts them to instructions understandable by the Apple] [hi-res graphics routines, places the specified characters at the desired locations on the hi-res screen, and contains within it one complete small font and an upper case large font.

The character *editor* is that part of Higher Text which you control to modify or redefine characters. Two cursor oriented editors make up this one module. The capability to create your own fonts, or to modify existing ones, is handled by these editors, one for small fonts, and one for large fonts.

Character sets you have developed or modified may be saved in two different ways:

- as a font which can be loaded back into Higher Text at any time, or as a generator, which can be saved on a separate diskette, along with your BASIC program.

What's this about more than six Hi-Res colors?

There is a technique known as "dithering" which is very much like mixing two colors of paint together to produce another color. As in paint, when blue and yellow are mixed to produce green, the original pigments remain, and it is the *illusion of seeing these bits of pigments so close together that we accept as a new color*. So it is in hi-res graphics when adjacent dots on the screen are "mixed"; it appears to our eyes that we have created a new color. Just as when you mix a gallon of blue paint and a gallon of yellow, you end up with two gallons of green; when you "mix" two vertical dot rows in hi-res, you end up with twice the number of dots, thus the four additional colors of yellow, lavender, pink and aqua are available in the "expanded" character mode only.

Pretty Printing

Large font characters are composed from a 14X16 dot matrix and have but one display mode. They may be displayed in any of the six basic hi-res colors.

The small fonts are developed from a 7X8 dot matrix, and may be displayed in any of the available display modes, Regular, Wide, Tall, and Expanded, and in up to six hi-res colors, plus four additional colors that may be used with the Expanded mode only.

The Wide and Expanded modes may be displayed on a background of any of the six standard hi-res colors, and a table has been provided to explain how text/background mixtures are obtained. As a consequence of limitations inherent in the Apple][hi-res display, results may *not* be as you anticipate, thus selection of text and background colors must be undertaken with care. For example, to print orange text on a white background, blue must be selected as the text color. Except when a black background has been chosen, the text color *displayed* will be the complement of the text color *selected*.

Identical Editors are Like Identical Twins

As indicated previously, two editors are supplied, one for the small font group and one for a large font. Each performs its chores in an identical manner. Two cursors are provided. One is used to *select* the character you wish to edit, and the second cursor is the *edit* cursor which is used in the edit window to set or clear points in the matrix.

Several options and combinations are available for saving an entire font or generator to disk. When using the small font editor, three complete fonts, numbered 1, 2, and 3, are always displayed on the screen. Any one font may be saved individually, or all three may be saved at the same time. In addition, you may save a large font without a lower case and one small font together. All possible combinations of saving various fonts and generators together are described in detail in the section on the character editor.

Up, up and Away (Higher Text in Action)

Throughout this manual, wherever there are procedural differences between Applesoft Basic and Integer Basic, the Integer Basic function will be shown second, enclosed in square brackets. In program listings, those pertaining *only* to Applesoft will be preceded by a right square bracket (]); those pertaining only to Integer will be preceded by a right angle bracket (>). If a program line is bilingual (so to speak), no special notation will be used.

Similarly, references to hexadecimal addresses throughout will be preceded by a dollar sign (\$), and followed by a decimal address enclosed in square brackets.

When you boot on the Higher Text diskette, it will automatically configure itself to your system. If you have an Apple][Plus, the BASIC programs will be in Applesoft. If for any reason you wish to reconfigure your Higher Text diskette, enter either FP or INT from the keyboard, whichever is appropriate, and BRUN BOOT. Note that the disk must not be write protected when BRUNing BOOT.

After BRUNing Higher Text, or following a boot, a four choice menu will appear. Selection one is entitled DEMO, which we suggest you try before reading further in this manual. DEMO will show you in a moment or two, the very sophisticated uses to which Higher Text may be put. In addition, if you have Integer Basic available, you may run from the keyboard the program DEMO 48, which is an expanded version of the DEMO program on the menu.

If you exit the menu with choice four, EXIT, you will return to command mode, but with a difference. Higher Text will still be up and running, and you will still be viewing the hi-res screen. Each of the character generator command codes will perform their expected functions, and you may enjoy the novelty of looking at your catalog displayed in any font mode. Just enter a Ctrl Q for the large font, carriage return and then type catalog as you normally would. Note that if you are in lower case mode, neither DOS nor BASIC will recognize any commands you may type.

Should you list a program while Higher Text is up, any control characters imbedded in the listing will be implemented by Higher Text, thus you may see your list change fonts and/or colors from line to line, etc. It is probably a good idea to exit Higher Text with RESET if you are seriously interested in examining the listing. Higher Text may be reentered with a CALL 3075 which will preserve all variables and data, or a CALL 3072 for a "cold" start. If you inadvertently exit either editor with a Ctrl C, you may return to the editor and save all data with a RUN.

To actually use one of the Higher Text fonts in your own program, the font must first be saved as a *generator*, not a font, using a file name of your choice. Then you must enter the following two lines in an Applesoft program; line 20 only for an Integer program.

```
10 PRINT CHR$(4) "BRUN LOMEM:" & LOMEM: 16384
20 PRINT CHR$(4) "BRUN FILENAME"
> 20 PRINT D$; "BRUN FILENAME": REM D$ = Ctrl D
```

Applesoft requires that the start of program be moved upward in memory, which is accomplished by the &LOMEM: program. If you wish to use hi-res screen two, the 16384 should be replaced with 24576. Filename is, of course, the name under which you have saved your generator. Line 10 is not required by Integer Basic. Lines 10 and 20 should be entered prior to declaring any variables.

Preceding this section are two summaries of Higher Text commands, one each for the generator and the editor. Each summary is indexed with a page number where special information on the referenced command may be found. Special note should be taken that the commands for the generator are all control characters, while those for the editor are *regular* characters, except where specifically shown as "ctrl [chr]". A control character is entered by holding the control key down, while at the same time depressing the character key specified.

Sections III and IV are devoted to functional descriptions and examples of each feature of the generator and editor, respectively. Section V is for the brave of heart who are interested in knowing details that have been omitted from Sections III and IV, including a memory map and important addresses.

Section VI describes use of the &LOMEM: utility and the Improved Integer Basic Hi-res [Wozpak] routines, Section VII is a glossary of terms used in this manual, and Section VIII explains how to obtain authorization for commercial use of a Higher Text Generator.

III THE TEXT GENERATOR

Introduction

The text generator is the program on the Higher Text diskette that is named "Higher Text". It contains within it \$C00 [3072] bytes of character fonts. You may, if desired, load a font from the disk by its file name, or use the font that Higher Text automatically loads in when it is run or booted. Additional fonts may be saved in various portions of free memory as indicated herein. Figure 2 on page 17, shows the three possible configurations of fonts within a generator. A generator is SAVED from either of the two editors, using a file name of your choice, and it is this generator that enables your BASIC program to make use of the Higher Text fonts.

As explained on page 17, the usual method of using Higher Text, or a Higher Text generator from your BASIC program, is with the BRUN command. However, there may be occasions when you do not wish the generator to be brought up immediately, in which case the generator may be BLOADED, and later activated with a CALL 3072. Note this does not eliminate the need in Applesoft to BRUN and use the &LOMEM: 16384 (or 24576) prior to declaring any variables. Further information on &LOMEM: may be found in Section VI.

If Higher Text or a generator has already been BRUN and you wish to disable it for any reason, this may be done with a PRINT CHR\$(4) [Ctrl D]; "PR#0": PRINT CHR\$(4) [Ctrl D]; "IN#0". The PRINT CHR\$(4) [Ctrl D] is not required from immediate mode. A CALL 3075 will restart Higher Text with all fonts and variables intact. This is called the "warm start" address because it preserves all data. The "cold start" address is CALL 3072, which would destroy all variables and start from scratch.

The following subjects are covered in this section:

A. Character Parameters

1. Font Sizes and Display Modes
2. Character Colors
3. Background Colors
4. Upper and Lower Case

B. Cursor Controls

C. Screen Functions

D. Brief Demo programs

All control functions in the generator, with the exception of upper and lower case shift locks, are fully programmable, meaning they may be imbedded in your BASIC program lines. You may find it convenient in Applesoft to substitute the equivalent CHR\$(n) for the control character you wish to use. Note also that if control functions are entered as separate PRINT statements, (and they may be, if you wish) then they should be followed with a semicolon (;) to avoid unintentional scrolling.

A. CHARACTER PARAMETERS

Text characters can be operated upon in five different ways.

- You may choose the font *size* (large or small)
- You may choose the display *mode* of a small font
- You may choose the font *color*
- You may choose the *background* color in some cases
- You may choose upper or lower *case*.

Any of the above characteristics may be intermixed within a single PRINT statement, provided that the display parameters and requirements for the specific selection are adhered to. For example, if you have selected a pink expanded character, you can not change to a tall character, because a tall character will not display in pink. Text is entered normally, exactly as you would do when writing a PRINT, INPUT or REM statement to the regular text screen. All the usual commands are recognized: HOME [CALL -936], TEXT*, HTAB [TAB], VTAB, etc.

Your best guide in learning to correctly select the character specifications you desire, is by first entering and observing the results of the short sample programs at the conclusion of this manual, and then, using those programs as examples, write your own test programs. These will determine if the control characters you have entered indeed produce the desired effect. In addition, the DEMO program in Section V. (D) may be of help.

* See TEXT command on page 14.

1. Font Sizes.

a. Large Font

The large font will display 12 lines of 20 characters, and is selected preceding a PRINT statement with CHR\$(17) [Ctrl Q]. It may be displayed in standard colors on colored backgrounds.

b. Small Font Display Modes.

There are four small font display modes: Regular, Wide, Tall and Expanded. They are all formed from a 7x8 dot matrix and are identified by their screen format characteristics, which are set forth in Table I. In addition, each of the four display modes has its own intrinsic features, also detailed in Table I, along with the appropriate commands to select each mode.

The Wide and Expanded characters are the only ones that may be displayed against a colored background. Table II lists the necessary text and background color selections to achieve the desired color mix for the Wide or Expanded display mode. Regular and Tall characters have a secondary display mode referred to as bold face. This is similar to the bold face or enhanced mode of some hard copy printers, where the method used is to overstrike the characters at a one dot offset.

Bold face mode is selected with CHR\$(25) [Ctrl Y] and deselected with CHR\$(26) [Ctrl Z].

2. Character Colors.

The assignment of character colors in Higher Text appears at first to be a rather complex proceeding, because of the variety of combinations of font sizes, classes, colors, and background colors. That each class may not be displayed in each available mode or color is again a result of the limitations of the Apple high resolution video display. After some experimentation, it will be found to be simpler than it appears.

The next paragraphs attempt to define the color choice selections available to each of the font characters, and how to enable them. You are additionally referred to the program examples given later in this document. Three font display modes will display characters in six colors: Expanded, Wide, and Large. In addition, the Expanded mode may be displayed in the four "dithered" colors: yellow, lavender, pink, and aqua.

Table II is a chart which contains the various character and background color command codes, and the resulting combinations of text and background colors produced by each. In some cases, more than one set of command codes will produce the same combination of text and background color. These are marked as "alternate" and set in italic type. Other command code combinations not listed in the table produce undesirable results, such as a colored character on a horizontally striped two color background. In certain cases these combinations may be used for special effects, which can be found through experimentation, then recorded for future use.

TABLE 1 FONT DISPLAY MODES

Character Display Mode	Matrix Size	Screen Format Lines of Chrs.	Select Display Mode with: Ctrl CHRS	Select Bold Face with: Ctrl CHRS	Deselect Bold Face with: Ctrl CHRS	Bkgn'd Colors Avail:	Character Colors Avail:*
Large	14x16	12 20	Q 17	- -	- -	6	6
Regular	7x8	24 40	R 18	Y 25	Z 26	2	2
Tall	7x8	12 40	T 20	Y 25	Z 26	2	2
Wide	7x8	24 20	W 23	- -	- -	6	6
Expanded	7x8	12 20	E 5	- -	- -	6	10

* In some cases, additional character colors may be used, but some characters will be imperfect, depending on where they are placed on the screen, and the background color present.

REGULAR	1	2
TALL	1	2
EXPANDED	1	2
WIDE	1	2
LARGE	1	2

CHARACTER COLOR SELECTION										
CHR Color PRINT	White Ctrl C0	Green Ctrl C1	Violet Ctrl C2	Orange Ctrl C3	Blue Ctrl C4	Yellow Ctrl C5	Lavender Ctrl C6	Pink Ctrl C7	Aqua Ctrl C8	
DSP MODE	EWL RT	EWL	EWL	EWL	EWL	E	E	E	E	E
BACKGROUND COLOR SELECTION	WHITE Ctrl C0 Ctrl B0	T X B	Black	Violet	Green	EWL				
			White	White	White	White	Blue	Grn/Orng.	Pink/Blk.	
	GREEN Ctrl C1 Ctrl B0	T X B	—	Black	White	alt. White	—	—	—	—
			—	Green	Orange	Orange	—	—	—	—
	VIOLET Ctrl C2 Ctrl B0	T X B	—	White	Black	alt. Black	—	—	—	—
			—	Violet	Violet	Blue	—	—	—	—
	ORANGE Ctrl C3 Ctrl B0	T X B	—	alt. Black	White	White	—	—	—	—
			—	Green	Orange	Orange	—	—	—	—
	BLUE Ctrl C4 Ctrl B0	T X B	—	alt. White	alt. Black	Black	—	—	—	—
			—	Violet	Violet	Blue	—	—	—	—
	BLACK Ctrl C9 Ctrl B0	T X B	White	Green	Black	Black	—	—	—	—
			Black	Black	Black	Black	—	—	—	—

TABLE II BACKGROUND/CHARACTER COLOR SELECTION CHART

Note: Selection of BLACK [Ctrl C9] as a Character Color will result in a character the same color as the background useful only as XDRAW.

3. Background Colors

All characters may be displayed against black or white backgrounds. The Expanded, Wide or Large modes may in addition be displayed against the other four available colored backgrounds. Setting a colored background is a wee bit tricky, since two command functions are required, and except on black backgrounds, the text color selected will produce its complement. The available combinations are shown in Table II.

The background color subroutine uses the text color command code as the first of two steps, thus text color selection must be made after establishing the background color. After deciding upon the combination of text and background colors you wish to use, first choose a background color by PRINTing Ctrl Cn, where n=a color from 0 to 4, or 9, from the color command code list, then PRINT a Ctrl B0 to pass the chosen color on to Higher Text's background color subroutine. For example, to print Expanded orange characters on a white background:

```
] 100 PRINT CHR$(5);
> 100 PRINT " " ;: REM CTL E IN QUOTES
101 REM SELECT EXPANDED CHR MODE
110 PRINT "0": REM SELECT WHITE BG
111 REM CTL C PRECEDES 0 IN QUOTES
112 REM 0 WILL NOT LIST IF HIGHER TEXT IS UP
120 PRINT "0" ;: REM PASS INFO TO BG ROUTINE
121 REM CTL B PRECEDES 0 IN QUOTES
122 REM SEE 112
130 PRINT "4" ;: REM SELECT BLUE TO PRODUCE ORANGE
    CHR
131 REM CTL C PRECEDES 4 IN QUOTES
132 REM SEE 112
140 PRINT "HELLO": END
```

This procedure can be simplified considerably by concatenating the required characters into an Applesoft string or, in either language, entering the characters directly into a string, as shown in the following examples:

```
] 150 SETUP$=CHR$(4)+"0"+CHR$(3)+"0"+CHR$(4)+"4": REM
    SET EXP,WHITE BG, ORANGE CHRS
160 PRINT SETUP$; "HELLO": END
    or
170 DIM SETUP$(10): SETUP$="c0b0c4" REM small chrs represent ctrl chrs
180 PRINT SETUP$; "HELLO": END
```

There are a few rare situations — for example where you might wish to place Higher Text characters on an existing screen that shows a graph, etc. — in which the Higher Text background conflicts with that of the graph. In this event, Higher Text has provided several alternatives through the Ctrl. B(ackground) function. Details on these alternate background display modes may be found in the technical section, Part D.

4. Upper and Lower Case

Higher Text has provided a simple means for the direct entry of lower case characters through the use of two shift lock functions: Control A, upper case shift lock, and Control S, lower case shift lock. These two control characters will not be entered into the text, as they are used in the immediate mode only. Once lower case characters have been entered, they remain a part of the program, regardless of whether or not Higher Text is "up" or in use.

After typing a Ctrl S, all following characters will be in lower case until one of the following conditions is met:

- A Ctrl A is entered (the following characters will revert to upper case)

- A carriage return is entered (the program line as it exists will be entered into memory)

- A Ctrl X is entered (the current line will be cancelled as in the normal Ctrl X function)

Following any of the above actions, the next characters typed will be in upper case. You are reminded that neither DOS nor BASIC will recognize commands or variable names typed in lower case, thus a Ctrl A must always be typed in a program line just prior to a command, if your current mode is lower case. Lower case may be used in any PRINT, REM, or INPUT statements.

Note lower case characters may only be typed while Higher Text, Program Line Editor or similar entry methods are up.

B. CURSOR CONTROLS

Four limited use cursor commands are provided in the Higher Text generator. These have the effect of moving the cursor in a PRINT or REM statement in accordance with the control used. A control H and Control I duplicate the functions of the left and right arrow keys, respectively, except the Control H will not be entered into the line of text. A Control J duplicates the function of Control J in the Apple, i.e., a line feed. The Control K is the most useful of the cursor functions in that it provides a reverse scroll, performing the inverse of a Control J. Try this example to get the idea:

```
100 PRINT "HELLO kkk GOODBYE"
```

where the small "k" represents a Control K. Try both listing and running the program and observe the interesting results.

C. SCREEN FUNCTIONS

Two types of screen controls exist in Higher Text, those which clear the screen, and those which scroll the screen. There are three of each type, and each is enabled by the control characters shown in the following table. Reminder: type the control key and the alphabetic character key simultaneously, just as when using the shift key.

1. Clearing Functions

CHRS	CTRL	FUNCTION PERFORMED
12	L	clears screen, homes cursor
14	N	clears current line from cursor position to end of line
15	O	clears from cursor position to end of page

2. Scrolling Functions

CHRS	CTRL	FUNCTION PERFORMED
6	F	scrolls text up one line
16	P	turns on smooth scroll mode
0	@	turns off smooth scroll mode.

Each of the preceding commands may be entered in program lines or used from immediate mode. In Integer Basic, the control character must be placed in a PRINT statement or assigned to a string. Applesoft provides the additional capability of PRINTing the equivalent CHR\$ function.

While Higher Text is up, if a program line is listed, you may notice a rather shaky scrolling motion as each line scrolls up. This is characteristic of a "normal" scroll on the high resolution screen. As shown in the table above, Higher Text also provides a super smooth scrolling feature. It is, however, considerably slower than the standard scroll, thus it is up to you to determine which method best suits your needs.

3. TEXT Command

The Higher Text TEXT command varies somewhat from the normal Apple usage, in that while it resets the scroll window, it does not return to the standard text mode, but remains on the current hi-res page. TEXT functions only in direct mode; under program control, a CALL -1220 should be used for the same function.

IV THE CHARACTER EDITORS

Introduction

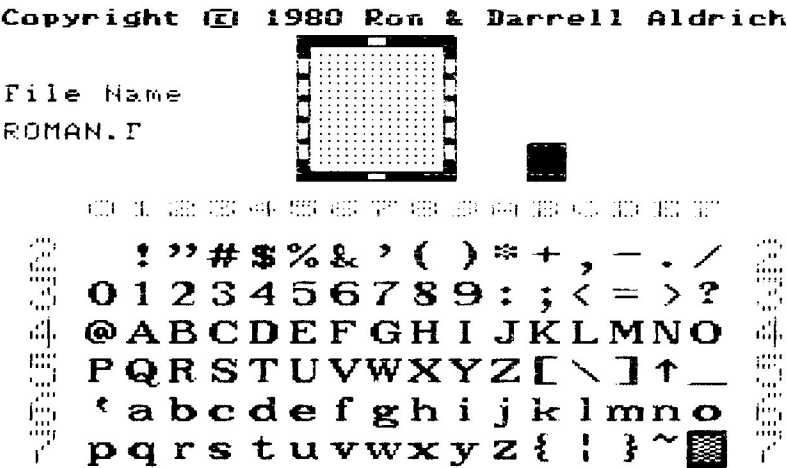
The Character Editor is the portion of Higher Text where you create or modify characters. Actually there are two complete editors, one for the 14x16 matrix (large font) and one for the 7x8 matrix (small font). Each functions in an identical manner, and will be treated here as a single editor, except where differences exist.

The editor itself is divided into two modules. One module handles the system chores, such as selecting the character to be edited, moving it to and from the edit window, loading and saving fonts and generators, etc. The second module performs the actual editing, moving the character within the matrix, inserting and deleting lines, etc. Each module is equipped with its own cursor, referred to hereafter as the *system* cursor and the *edit* cursor, respectively.

In rather general terms, you may wish to load a font, select a character with the system cursor, move it into the edit window where it can be manipulated, change its appearance by editing with the various available commands, return it to the character set, and either save the font with the modified character or select a new character and repeat the procedure.

Figure 1a shows the actual screen display provided by the Large Font Editor. The lower portion of the screen displays the current large font, which occupies four lines of 16 characters each. Control L is a *toggle*. If Control L is pressed, two additional lines, representing the lower case characters, will appear on the screen. Pressing Control L a second time will return the font to its previous state. Upon first entering the editor, the system cursor will be found in its home position, over the rightmost character of the last line. When Control L is toggled, the system cursor will home to the new last line.

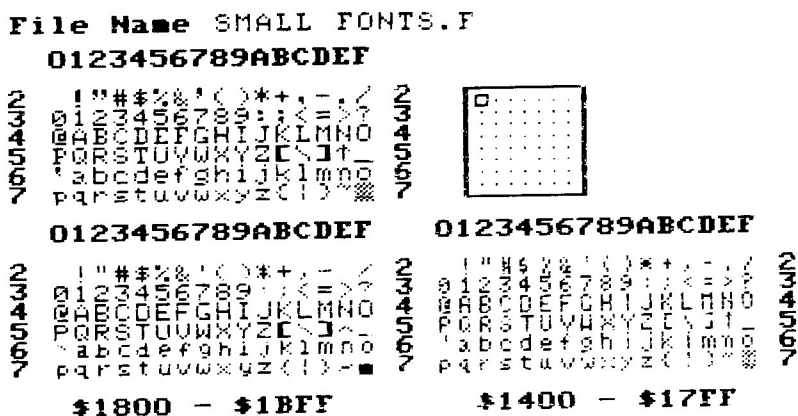
Figure 1A



Centered above the font is a 14x16 dot matrix. This is the *edit window*, and the *edit cursor* will be found in its home position, the upper left corner. Each editor is equipped with a *view window* to the right of the edit window. When a character has been placed in the edit window, the view window will display it, full size, in each of the six possible *colors*, just as it would appear in your program. It is important to check the view window while editing, as some combinations of adjacent dots may produce undesirable effects in certain colors.

Figure 1b shows the actual screen display provided by the Small Font Editor. Three small fonts are displayed together, in an L-shaped pattern, with the edit window to the right of the upper font (number 1), which is the default font. Small font number 2 is at the lower left, and number 3 is at the lower right. Entering a 1, 2, or 3 will select the corresponding small font, and the system cursor will be placed at its home position at the lower right corner of the selected font. Initially, if the system cursor has been moved from its home position, and a new font is selected with the 1, 2 or 3, then the system cursor will be placed over the same relative character as in the previous font. Note that each of the three small fonts are complete with both upper and lower case.

Figure 1B



The view window in the small font editor serves a purpose similar to that of the large font. It displays the current character being edited in each of the six possible *display modes*, i.e., regular, bold, tall, wide, narrow or expanded. Again, it is important to check each character being edited in the view window, as it is possible that some combinations of adjacent dots may produce undesirable effects in certain display modes.

A. EDIT SYSTEM FEATURES

1. Entering and Exiting the Editors

Three avenues of entry for the edit module exist. You may 1) boot the disk, or 2) RUN Hello and select either editor from choices 2 and 3 on the menu, or 3) type "RUN EDIT.SM (or) EDIT.LG" for the small font and large font editors, respectively, or if you have inadvertently left either editor and have not loaded another program, you may re-enter with a RUN, and all characters will be as you left them.

To exit either editor, you may type a Control C, which will will return you to immediate mode, with Higher Text still up, or you may type E to select the other editor. If you type an E, you will be asked "RUN LARGE [SMALL] FONT EDITOR (Y/N)". A 'N' response will give you the opportunity to back out, in the event you typed the E in error.

2. Loading and Saving Fonts and Generators

Let's get into this one by explaining again what a font is and what a generator is, and the various font/generator combinations. A Higher Text character font has three possible configurations, which are shown in figure 2A. generator is a font *plus* the Higher Text Operating system. In fact, the program named Higher Text is a generator, and is composed of nothing more than the Higher Text operating system and the default font that is supplied as an intrinsic part.

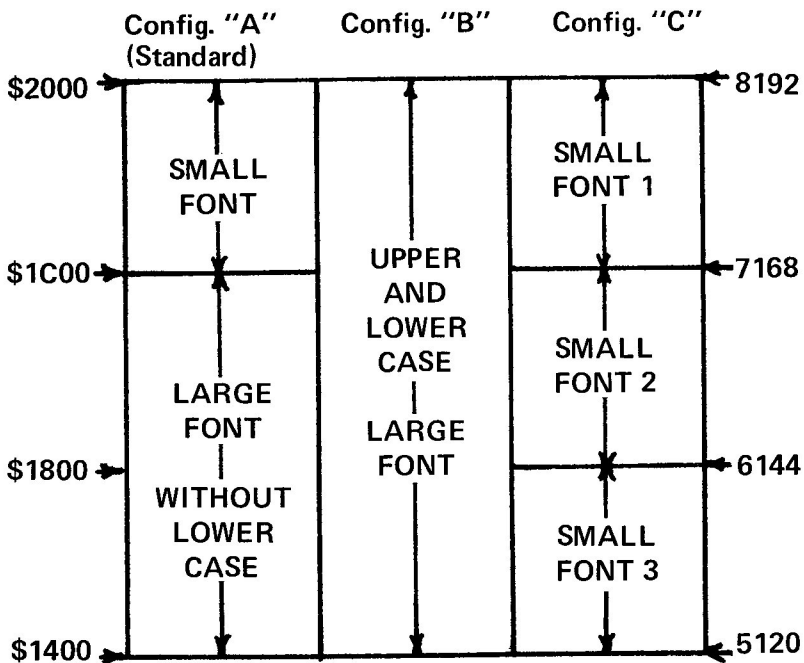


Figure 2
Higher Text Font Configurations

A font or a generator may be *saved from* or *loaded into* the editor in accordance with the configurations shown in figure 2. The criterion in determining whether you should save a font or a generator is its intended use. If it is one of a number of fonts you may wish to use at some time in the future, you probably would want to save it as a *font* only. If you are saving it to a separate diskette and have a specific application for it in a program, you should save it as a *generator*. To utilize a generator in your program, it must either be BRUN, or else BLOADED and enabled with a CALL 3072.

The O[utput] command is used to save either a font or generator to disk. When you type the O, the editor will ask whether you wish to save a font or a generator, and the appropriate answer should be made. Note that if you wish a suffix to be added to the file name of the font or generator, it must be specified at this time, as it is not done automatically.

From within the editor, the L[oad] command is used to load a font. Before you type the 'L', make certain you know the name of the font. The full name of the font should be specified, including any suffix. If you cannot remember the name of the font you want to load or save, type a 'D' (for Disk catalog) to refresh your memory. If you plunge madly ahead and attempt to load a non-existent file name, an error may result. However, entering a carriage return in response to the file name query will return you to the editor safe and sound, with no file loaded.

If you are using the small font editor when you go to load a file, you will be asked an additional question: "Normal Location (Y/1/2/3)". Normally, you would respond with a 'Y', which will load either a font or a generator. However, if you are loading a single small font, a numeric response may be used to "force" it into the specified location. The number selected would coincide with the fonts shown in figure 1b.

3. Moving Characters To and From the Edit Window

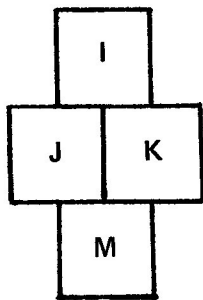


Figure 3

The system cursor is used to select the character from the displayed font that you wish to edit. Likewise, after editing a character, the position of the system cursor determines where that character will be returned to the displayed font. Four keys are used, 'I', 'J', 'K', and 'M', representing up, left, right, and down. You will note these keys are arranged in a cluster which simulates the points of a compass, as shown in figure 3.

When the system cursor has been placed over the character selected for editing, a 'P' will P[lace] the character in the edit window, where any operation may be performed on it. After editing, a [RETURN] will return the character to the position in the font shown by the system cursor. If you wish to return the character elsewhere in the font, the system cursor must be moved to that position before hitting return.

4. Swapping Large Font Characters

A situation may arise where you need a single character from a font not currently in memory. This may be accomplished by BLOADing the alternate font, BSAVEing the desired character in accordance with the below formula, reBLOADing the original large font, BLOADing the single character (again from the formula below), and finally outputting the modified font to disk.

Take the ASCII value of the character to be swapped, subtract \$20 [32], multiply the result by \$20 [32], and add this to \$1400 [5120]. This will be the A\$ [A] parameter for the BSAVE/BLOAD, and \$20 [32] will be the L\$ [L] parameter.

B. CHARACTER EDITING FEATURES

1. Clearing the Edit Window

There are two ways to clear the edit window. When a character has been completed and [RETURN]ed to the font, the window will automatically be cleared. If you are creating a character from scratch (i.e., one that has not been P[lace]d in the edit window) and have made such a botch of it that you wish to start over, the R[estart] command will also clear the edit window. However, if you are working on a character from the font and wish to restart, you should just type a 'P' and the same character will be placed in the edit window again, exactly as it was when you commenced editing.

2. Setting and Clearing Dots in the Matrix.

When the edit cursor is covering a dot that is to be added to the current character, enter 'X' to *set* the dot. (X marks the spot!) If you wish to remove a dot from the current character, typing a 'C' will *clear* it. The commands 'X' and 'C' may be typed at any time in the editing process.

3. Using the Edit Cursor

The position of the edit cursor determines where in the matrix a dot will be set or cleared. The edit cursor is configured exactly like the system cursor in that the function keys are arranged like the points of a compass. The edit cursor uses the commands 'W', 'A', 'S' and 'Z' to move the cursor up, left, right, and down, as shown in figure 4.

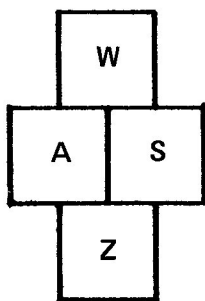


Figure 4

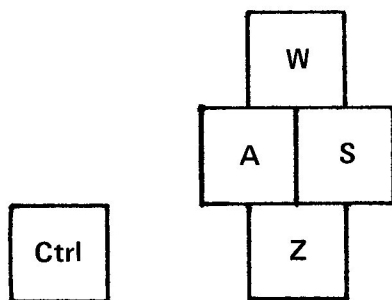


Figure 5

4. Scrolling the Character in the Edit Window

There are times when you may wish to *scroll* the character in the edit window, for example in modifying an "A" to an "Æ", you might wish to move the original "A" one or two dot positions to the left. The four control characters shown in figure 5, Ctrl W, Ctrl A, Ctrl S, and Ctrl Z will scroll the character up, left, right, or down. Note again the keypad/compass point arrangement. In our example above, if you wish to shift the "A" two dot positions left, you would hold the Ctrl key down and type AA.

5. Inserting and Deleting Lines

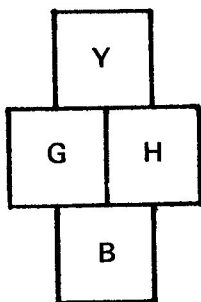


Figure 6

The insert/delete function has the effect of doing a partial scroll. If, for example, you wished to make a character taller, you no doubt would want to add dots at some appropriate place in the middle of the character. To do this, you would position the edit cursor at the location where you wanted to add a line, and type 'B', which would scroll that portion of the character at the cursor position downward, leaving a "blank" line where you could then set dots as needed. Conversely, a 'Y' would delete one horizontal line and shift the remainder of the character up one dot row.

The Insert/Delete key cluster is shown in figure 6, and the four compass commands are Y=delete horizontal line, G=delete vertical line, H=insert vertical line, and B=insert horizontal line.

V TECHNICAL DATA

A. MEMORY ALLOCATION

1. Memory Map

Figure 7 on page 21 shows complete memory allocations for the Apple] [, including areas used by Higher Text and Program Line Editor.

2. Memory Usage by Higher Text and Friends

The Higher Text generator resides from \$C00 [3072] to \$13FF [5119]. In addition, it makes liberal use of the page three area from \$300 [768] to \$32F [815], for scratch storage and flags, etc., as detailed in Table III on page 22.

The &LOMEM: utility uses page three from \$330 [816] to \$3CF [975]. However, once &LOMEM: has been called, and the new start of an Apple-soft program set, then assuming there is no further requirement for this utility, the space may be used for user routines as needed. This area is available at any time from an Integer Basic program. From \$3D0 [976] to \$3FF [1023] is used by DOS and monitor for various vectors, etc., and hence is unavailable.

The Integer Basic High Resolution graphics routines use the area from \$7FD [2045] to \$BFF [3071], and are used from both Integer Basic and Applesoft programs. However, in Applesoft, if HGR or HGR2 is executed prior to BRUNning Higher Text or a generator, then this area becomes available for other use.

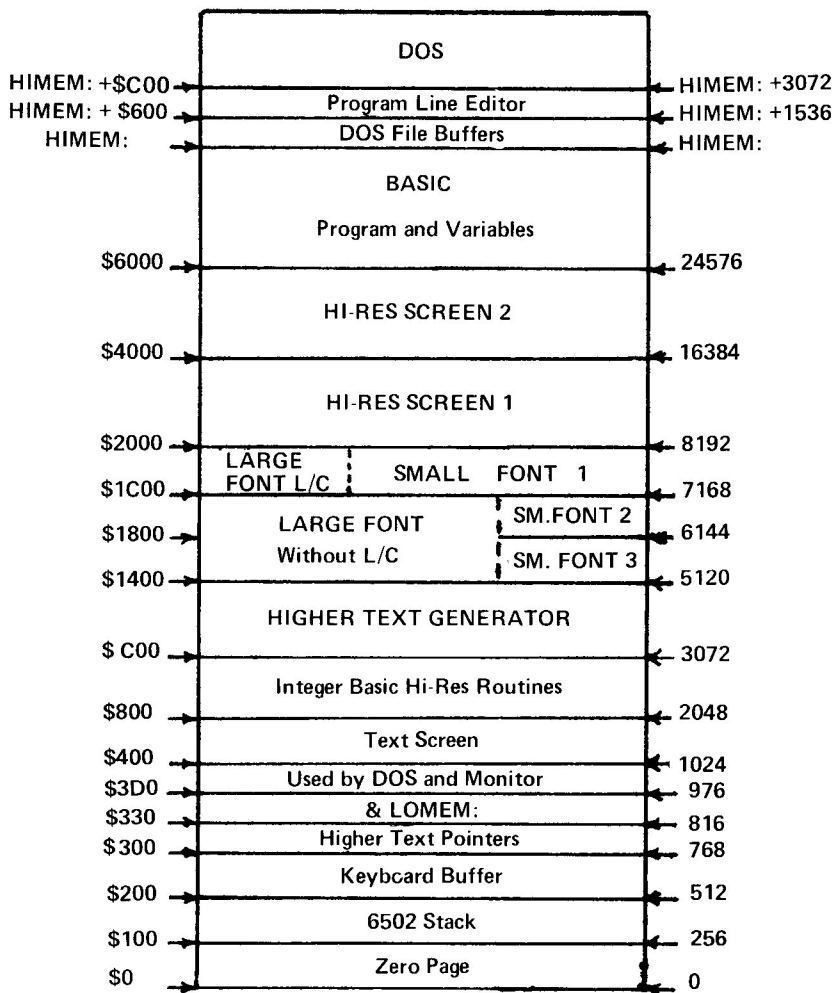


TABLE III

Address		Assembler Label	Comments
Hex	Decimal		
302	770	COLOR	Current color number
303	771	FONT2	Large font pointer
304	772	FONT1	Regular Font pointer
305	773	CTPTRL	
306	774	CTPTRH	
307	775	BKGNDL	
308	776	BKGNDH	
309	777	YSAV2	
30A	778	LWRFLG	
30B	779	LRFLG1	
30C	780	SIZFLG	
30D	781	INVFL2	
30E	782	SCLFLG	
30F	783	CLKFLG	Key click flag
310	784	MSKFLG	
311	785	BLDFLG	
312	786	BSTFLG	
313	787	CSTFLG	
313	788	EXPAN1	
314	789	EXPAN2	

3. High Resolution Screen Pointers

It is possible to draw on the hi-res screen page *not* currently in use, while continuing to view the present screen, by changing the value of the hi-res screen pointer. These pointers are located as follows: \$E6 [230] for Applesoft, \$326 [806] for Integer Basic. A \$20 [32] will permit drawing on page one; a \$40 [64] will permit drawing on page two, thus, from Applesoft, POKE 230,64 would allow you to draw on page two while continuing to view page one.

B FONT CONFIGURATIONS

1. Normal Configurations

The normal area for font storage is from \$1400 [5120] to \$1FFF [8191]. There are three possible font configurations, as shown in figure 2, page 17. The default font, the one loaded in with Higher Text, contains one large font without lower case, and one complete small font. It is referenced in figure 2 as "Configuration 'A' (standard)".

One complete small font will occupy the same amount of memory as the lower case portion of a large font; three complete small fonts will take up the same memory space as a *complete* large font, thus it may be seen that in "Configuration 'B'", we have three small fonts which store in RAM at \$1400 [5120], \$1800 [6144], and \$1C00 [7168], respectively. No specific action is required on your part to determine which configuration is saved to disk as a font or generator; this is established solely by the present data in memory.

Saving a font from either editor will result in saving all memory from \$1400 [5120] to \$1FFF [8191], unless small font 1, 2 or 3 is specified in the save parameters, in which case only the appropriate \$400 [1024] of memory will be saved. If a generator is specified, memory will be saved starting at \$7FD [2045].

2. Non-Standard Configurations

There may be times when you wish to have available more than one font configuration at a time, for example, a large font of standard characters, and a large font of Old English or another contrasting style. A Higher Text font may be loaded into any area of memory where space is available. The table below shows where the index registers are located, and the values to assign to them. In addition, in Applesoft, it will be necessary to use the &LOMEM: utility to protect the area used by the special font allocation. Use of &LOMEM: is explained on page 29.

3. Calculating Font Indexes

The Index to the starting address of the large font is held in decimal location 771; the small font starting address index is held in 772 decimal. The index values may be changed with a POKE, either in immediate mode or from within the program, thus it is possible to change fonts "on the fly". To calculate the index for a large font, divide the starting address by 256, and subtract 4. The result may be POKEd into 771. A small font starting address should be divided by 256 (no subtraction required) and POKEd into 772.

Suppose you wished to use a second large font starting at \$4000 [16384]. You would then add the following statements to the beginning of an Apple-soft program:

```
200 PRINT DS"BRUN LOMEM:"
210 &LOMEM:19456 : REM 16384 + 3072
.
.
.
500 POKE 771, 60: REM(16384/256) -4
```

To the extent that free memory allows, you may have as many fonts in memory at one time as you wish. To determine the starting address of the next font to be specially located, add the length of the last font loaded to the starting address, using the following lengths:

FONT TYPE	LENGTH
One small font	\$400 [1024]
One large font (complete) or one large upper case and one small font	\$C00 [3072]
One large font, no lower case	\$800 [2048]

TABLE IV. SELECTED FONT INDEXES

Font type	Address	POKE	Index
small	\$1400 [5120]	772	20
large	\$1400 [5120]	771	16
small	\$1800 [6144]	772	24
small	\$1C00 [7168]	772	28
small	\$4000 [16384]	772	64
large	\$4000 [16384]	771	60
large	\$6000 [24576]	771	92

4. "Garbage" Fonts

When the font configuration calls for a mixture of a large font without lower case and one small font, small fonts nos. 2 and 3 will appear on the edit screen as *garbage*, seemingly random collections of lines and hash marks, etc. In reality, what is being viewed is the upper case of the large font. Conversely, while in the large font editor under the same circumstances, the lower case will display as unintelligible garbage.

C. INTEGER BASIC HI-RES ROUTINES

Despite the fact that Hi-Res graphics commands are not built into the Apple II Integer Basic, a convenient scheme for referencing the subroutines and their parameters by name has been devised, as illustrated below.

TRADITIONAL METHOD	IMPROVED METHOD
--------------------	-----------------

> POKE 800, X MOD 256	> X0 = X
> POKE 801, X > 256	
> POKE 802, Y	> Y0 = Y
> POKE 812, C	> COLOR = C

The first statement (after BRUNing the Higher Text generator) of a program using the Hi-Res Subroutines should be as follows:

1 X0 = Y0 = COLOR = SHAPE = ROT = SCALE

The purpose of this statement is to enter the first six BASIC variable names in the symbol table in a fixed sequence. When executed, each of the six parameters will be assigned storage at fixed locations relative to the address contained in the BASIC "start of variables" pointer, LOMEM, making them readily accessible by the Hi-Res routines.

The following statement lists all of the new Hi-Res subroutine entry initializations available to BASIC programs. These routines operate similar to the ones described in the old Apple II Reference Manual.

5 INIT=2048 : CLEAR = 2062 : BKGND = 2865 : POSN = 2809 ;
 PLOT = 2830 : LINE = 2836 : DRAW = 2871 : DRAW1 = 2874 :
 XDRAW = 2884 : XDRAW1 = 2887 : FIND = 2556

The allowable color specification values may also be referenced by name, if the initialization statement below is included in your program. Note that "GREEN" is preceded by "LET" to avoid a syntax error due to confusion with the "GR" command.

```
8  BLACK = 0 : LET GREEN = 42 : VIOLET = 85 : WHITE = 127 :  
   ORANGE = 170 : BLUE = 213 : BLACK2 = 128 : WHITE2 = 255
```

For further information on these routines consult the Higher Graphics manual or the Programmer's Aid #1 manual.

D. CTRL B[ackground] SCREEN FUNCTIONS :

The common usage of Control B is with the suffix 0, which sets the background color, as explained in Background Colors on page 12. The Control B function is a highly technical achievement, complex and difficult for the first time user to comprehend. It extends the capabilities of Higher Text far beyond that of any other character generator currently available. A complete description of its various combinations and applications would require a manual unto itself, thus we will limit our coverage to conveying a general understanding of how it works, and its use for some applications.

Table V will give you an idea, in technical terms, of the operations performed. In 0-3 and 4-7 sequence, these are: EOR (exclusive OR) with the current color, OR the current page location, AND the page location, and EOR the page location. EOR, OR, and AND are Boolean operators which operate on binary data in a manner similar to the conventional add or subtract operators. A Boolean truth table is shown as Table VI, and in addition, you may be interested in getting a book from your library which more fully explains Boolean logic.

Table II on page 11 shows some 64 possible combinations of Font display mode/character color/background color using only Control B0. With some of the other Control B suffixes, and using various color combinations on the second hi-res screen, literally hundreds of combinations are possible.

The Applesoft program listed on page 27 is a demo that will take you through these myriad combinations, nine at a time, and allow you to select the ones that are most appropriate for a particular application. A smooth scroll command (Ctrl P) is imbedded in the program to give you time to see each variation before it scrolls away. Each sequence of nine possible character colors is interrupted with a request to "HIT ANY KEY" before proceeding. The current background mode and current character display mode are always in view at the top of the screen. A background of the six standard Apple hi-res colors in vertical bands is provided so that all possible combinations of font display modes and character colors may be viewed on each of the six backgrounds.

Note line 70: the DATA statement requires trailing spaces with each display mode name to fill out a total of nine characters.

The Ctrl B(ackground) command performs two types of logical operations with the current character color, using one of the following as a reference:

- the current character color
- the current page background color
- the alternate page background color

Operation type 1 is performed at the cursor position only. Operation type 2 is performed from the cursor to the end of line, and is executed only when a clear to end of line (Ctrl N) command is issued. A table of these functions appears below.

TABLE V CTRL B FUNCTIONS

Command Code	Logical Operation	Type 1 at cursor	Type 2 to end of line
Ctrl B0	EOR	a	a
Ctrl B1	OR	b	—
Ctrl B2	AND	b	—
Ctrl B3	EOR	b	—
Ctrl B4	EOR	a	c
Ctrl B5	OR	c	c
Ctrl B6	AND	c	c
Ctrl B7	EOR	c	c

TABLE VI TRUTH TABLE

AND

	0	0	1	1
and	0	and 1	and 0	and 1
	0	0	0	1

used as a mask. if one input is zero, the result will always be zero. used to clear any bit.

OR

	0	0	1	1
or	0	or 1	or 0	or 1
	0	1	1	1

if one operand is a one, the result will always be a one.

EOR

	0	0	1	1
eor	0	eor 1	eor 0	eor 1
	0	1	1	0

if any bit is different, the result will always be non-zero. the EOR of FF will always produce the complement. EOR is used for comparisons.

The most common application of Control B functions would be where you have an existing chart or graph you wish to label with text characters, and do not require a solid background. By BLOADing the chart to screen two and using the sub-routines described below, you may create a composite screen of the chart and labels on screen one. A further enhancement of this might be to have charts on disk — say for three different years — and use the labeling program to place the same labels on each chart as it is displayed.

Another example could be more closely related to the multi-color demo background, where your own background is composed of two or more colors, and you find your characters do not display properly on all of the backgrounds. The demo program can show you alternate combinations of Control C (character color) and Control B (background function) that would solve the problem.

Subroutine 80 sets up access to the second hi-res page, line 20 clears the screen, line 30 draws the vertical color bars, and subroutine 90-95 returns to screen one and transfers the contents of screen two to screen one, so that the same background is now on both screens. Note the GOSUB 80: GOSUB 90 at the end of line 45. This serves as a "refresh" for screen one, since some of the background will be destroyed by each cycle of nine color tests. You may wish to note this in particular, as it could prove handy in a future program.

LIST

5 REM

HIGHER TEXT CHR/BG COLOR DEMO

```

10 PRINT CHR$(4)"BRUN LOMEM:" &
   LOMEM: 24576
15 PRINT CHR$(4)"BRUN HIGHER TE
   XT"
20 GOSUB 80: HCOLOR= 0: FOR I = 0
   TO 20: HPLLOT 0,I TO 279,I: NEXT
30 FOR I = 0 TO 7: HCOLOR= I: FOR
   J = 20 TO 191: HPLLOT 35 * I,J
   TO 35 * I + 34,J: NEXT J,I: GOSUB
   90: PRINT CHR$(16);
35 FOR BG = 0 TO 7: PRINT CHR$(
   3);0; CHR$(2);0; CHR$(18); CHR$(
   25) VTAB 1: PRINT "BACKGROU
   ND MODE NBR: "BG;
40 FOR CHR = 1 TO 7: READ L,M,CH$
   : PRINT CHR$(L); CHR$(M);
45 VTAB 1: HTAB 24: PRINT CHR$(
   3);0; CHR$(2);0; CHR$(18); CHR$(
   25)" CHR= ";CH$; CHR$(L); CHR$(
   M); PRINT : POKE 34,2: GOSUB
   80: GOSUB 90
50 IF BG > 3 THEN PRINT CHR$(3
   );9; CHR$(2);0; CHR$(12): REM
   CLEAR PG 1 TO BLACK
55 VTAB 4: FOR CLR = 0 TO 9: PRINT
   CHR$(3);0; CHR$(2);0;CLR; CHR$(
   3);CLR; CHR$(2);BG;" THIS I
   S A TEST USING COLOR NBR: "
   CLR; CHR$(14): NEXT CLR: PRINT
   CHR$(2);0; CHR$(3);0
60 HTAB 4: PRINT "HIT ANY KEY": CALL
   - 756: NEXT CHR: RESTORE : NEXT
   BG: PRINT : PRINT "DONE": CALL
   - 1220: END
70 DATA 18,26,REG ,18,25,REG
   -BOLD ,20,26,TALL ,20,25,T
   ALL-BOLD,26,17,LARGE ,23,2
   3,WIDE ,5,5,EXPANDED.
80 POKE 230,64: POKE - 16304,0: POKE
   - 16302,0: POKE - 16297,0: POKE
   - 16299,0: RETURN
90 POKE 230,32: POKE - 16300,0
95 PRINT CHR$(2);7; CHR$(12): RETURN
   : REM
MOVE SCR N 2 TO SCR N 1

```

VI GENERAL INFORMATION

A. CREATING YOUR OWN PROGRAM DISKETTE

First you need to prepare a fresh diskette; using the DOS INIT routine. You must determine whether your program is in Integer Basic or Applesoft, then type INT or FP, whichever is appropriate to the program. Then, assuming you want the new diskette to boot and run your program, type INIT FILENAME, Vn, where FILENAME is the name of the program you will later save, and n is the volume number you wish DOS to assign to this diskette.

The disk drive will run for a minute or so, INITing the new diskette, and when it is finished, the flashing cursor will return. You have now created a "slave" diskette. We suggest for interchangeability with other systems that you convert it to a "master" diskette at this time, by BRUNning UPDATE 3.2.1 or MASTER CREATE, whichever is on your Apple master diskette, and following the prompts.

If you plan to save an Applesoft program, you will need the &LOMEM: utility on the new diskette. You can transfer this by using the FID program (if available) or by inserting the Higher Text diskette in the drive and typing BLOAD LOMEM:, then placing the new diskette in the drive and typing BSAVE LOMEM:, A816, L160.

You will also need a Higher Text generator on your new diskette. The easiest way to transfer this is to BRUN Higher Text, Load the generator, switch diskettes and save it to the new diskette, using the Higher Text L(oad) and O(utput) commands as described on page 18.

The last step is to save your Integer or Applesoft program on the new diskette, using the same file name with which the diskette was INITed. This transfer can be done with the FID utility, if you have it, or with conventional LOAD and SAVE commands, switching diskettes inbetween.

B. SUBSIDIARY FUNCTIONS

1. Key Click

For those Apple owners with a "silent" keyboard, Higher Text has a built in "key click" function which simulates the normal sound made when a key is depressed and touches the keystop. From direct mode it is selected and deselected with a Control V, which acts as a toggle. Under program control it may be selected with a POKE 783,128, and deselected with POKE 783,0.

2. Control G (bell)

As a minor embellishment, Higher Text has modified the "beep" sound of Apple's bell to a more pleasant and realistic sounding tone. The bell can be placed in a PRINT statement with a Control G, or in Applesoft, with a CHR\$(7). For those interested, the modified bell uses a pulse width modulated square wave, modulated by a ramp; for those not so interested, it still sounds pretty neat.

C. SUPPORTING PROGRAMS

1. Program Line Editor

Program Line Editor is a very useful utility program, not included on this diskette. It is available separately from your Higher Text distributor.

In addition to its editing capabilities, it is helpful to users of Higher Text by virtue of its capability to permit direct entry of lower case characters. PLE uses the same control commands for lower case as does Higher Text, i.e., Ctrl A and Ctrl S.

Higher Text will automatically disable PLE when called, i.e., with a CALL 3075 or BRUN HIGHER TEXT. However, PLE may be used for editing your current program if you exit HIGHER TEXT with a reset (or Ctrl reset). It is not necessary to remove PLE from memory.

With PLE in memory, it is then possible to enter program lines in lower case without having Higher Text up. At the same time, it may also be used for its editing capabilities.

2. LOMEM: (The &LOMEM: Utility)

Higher Text and its related routines use that area of memory where an Applesoft program is normally stored, thus a means must be provided to relocate the Applesoft program. LOMEM: is a separate binary program with that capability. In a Higher Text generator, this must be done before any variables are declared, since they would be overwritten by the memory move routine called by LOMEM:. Two steps are required: LOMEM: itself must be BRUN by your program, and &LOMEM: [locn] must be executed in the program (where [locn] equals in decimal, the new starting address for the Applesoft program). A program line to perform the above might look like this:

```
100 PRINT CHR$(4); "BRUN LOMEM:":$LOMEM: 16384
```

This would set the start of the Applesoft program past the first hi-res screen. &LOMEM: may be used more than once in a program, if needed, but only in an upward direction. If so, variables will be overwritten, but the program will be moved intact. Using &LOMEM: in a downward direction will destroy both variables *and* program. The proper way to reset the start of program to its normal [default] address is with the FP command. Once &LOMEM: has been executed, any subsequent Applesoft program will LOAD to the new address. The start of program address is unaffected by the NEW, CLEAR or regular LOMEM: commands.

3. Billboard

Billboard is a BASIC program containing two programs selectable from the Billboard menu, each of which is covered in the following sub-sections. Slidemaker is a utility which facilitates "labeling" a saved hi-res screen; Climber is a program to input, display and center lines of hi-res text.

a. Slidemaker

Slidemaker is a utility by Robert C. Clardy. It is included with the Higher Text diskette as an aid to labeling hi-res pictures, charts, etc. It may be selected from the Billboard menu.

After running Slidemaker, you will be asked if you wish to BLOAD a previously saved screen. This would normally be answered yes, as it is assumed you wish to label an existing screen. Be sure you have placed the diskette with the saved screen in the drive before proceeding.

Following your response to the preceding question, the Slidemaker command menu will be displayed. It is illustrated below in figure 8. In each case possible, the commands are designed to simulate those of Higher Text. All commands are control characters, since regular characters are used to write to the screen. The four cursor controls, K (up), H (left), I (right), and J (down) offer complete freedom in placing characters anywhere on the screen.

A Ctrl U, followed by a numeral in the range 0-9 will select a text color just as Higher Text's Ctrl C does. Colors and display modes may easily be intermixed, and Ctrl F may be used to load a different font, so that even fonts may be intermixed. The escape key will take you to and from the command menu at any time without disturbing the contents of the hi-res screen.

Figure 8

```
SLIDE MAKER COMMANDS

CTRL A = UPPER CASE   CTRL N = CLEAR/EOL
  B = ERASE           O = CLEAR/EOP
  C = ---             P = ---
  D = LOAD/SAVE       Q = LARGE FONT
  E = EXPANDED FONT   R = REGULAR
  F = FONT SELECT     S = LOWER CASE
  G = ---             T = TALL TEXT
  H = BACKSPACE       U = COLOR
  I = SPACE RIGHT     V = KEY CLICK
  J = LINE FEED       W = WIDE TEXT
  K = MOVE UP         X = WIPE TEXT
  L = HOME CURSOR     Y = SET BOLD
CTRL M = RETURN       CTRL Z = CLEAR BOLD
```

PRESS ESC TO SEE INSTRUCTIONS.

```
      K
H + I    ==> CURSOR MOVEMENT.
      J
```

b. Climber

Climber, by David Kampschafer, is a simple but handy program which allows you to input up to 26 lines of text, and save them to a file for later reloading. Climber, as its name implies, is a scrolling mechanism, using Higher Text's smooth scroll mode, to do a continuous scroll of the saved text with optional centering. In addition to the command menu, shown below as figure 9, other menus to choose character colors, etc., may be selected from the main menu. For best results, characters should be limited to 20 or 40 characters per line, depending on the font selected.

Figure 9

THE CLIMBER COMMANDS

CONTROL

F-FONT SELECT
I-NEW TEXT
A-AUTOCENTER
ESC-MENU

CONTROL

U-COLOR
D-LOAD/SAVE
X-EXIT

PRESS RETURN TO BEGIN

4. Higher Fonts

Higher Fonts is a collective name applied to diskettes containing fonts that can be used with Higher Text. Higher Fonts I, the first of the series is separately available at the time this documentation was prepared. It is expected that subsequent volumes will be announced from time to time.

Higher Fonts diskettes can save you hours of valuable time by supplying ready made fonts for inclusion in your Higher Text programs. Each Higher Fonts diskette comes with a utility named Font Viewer, which allows you to preview fonts you may later wish to include in your program.

Following are samples of some of the wide variety of fonts currently available for use with Higher Text.

PUNIC.F
P O O R D Q P V H M K X A N O O X O P
DIGITAL.F
A B C D E F G H I J K L M N O P Q R S
BLOCK.LF
A B C D E F G H I J K L M N O P Q R S
APL.F
X I N L E _ W A I O B I T O K P P T
RUNES.F
F B L M M X X F I I N T P T M C H R G

BUILDING.F
 EGYPTIAN.F
 HEBREW.F
 RUSSIAN.F
 A B B Г Д Е Ж З И Й К А М H O П P C T
 BROADWAY.F
 A B C D E F G H I J K L M N O P Q R S
 QUOTE.F
 A B C D E F G H I J K L M N O P Q R S
 ORIENTAL.F
 A B C D E F G H I J K L M N O P Q R S

SHADED.F
 A B C D E F G H I J K L M N O P Q R S
 SHADOW.LF
 A B C D E F G H I J K L M N O P Q R S
 THREE-D.F
 A B C D E F G H I J K L M N O P Q R S
 ITALICS.F
 A B C D E F G H I J K L M N O P Q R S
 CHARLESTON.F
 A B C D E F G H I J K L M N O P Q R S
 NORMANDIA ITALIC.F
 A B C D E F G H I J K L M N O P Q R S
 SPACE FONTT 1.F
 A B C D E F G H I J K L M N O P Q R S

5. Demo Programs

Way back in the overview (remember that?) we made mention of the problems encountered when trying to LIST a Higher Text program with imbedded control characters. There is a fairly easy way to overcome this, and that is by setting up a standard set of subroutines to establish each usable control code as a subroutine, thus when the program is LISTed, the interpreter sees only the GOSUB, rather than a code that would cause it to change display modes in the middle of a LIST.

There are also other advantages to this method:

because mnemonic names can be attached to the subroutines, there is never any doubt as to their function. The same routines can be used in many programs, and an EXEC file can be written so they may be EXECed into any new or existing program.

Listings of Applesoft and Integer Basic subroutines follow; an EXEC file creator which may be used by either language also appears. In each listing, control characters are indicated by a "normal" character enclosed in square brackets. The square brackets should not be entered; just type the control character specified by typing the Ctrl key and the character key at the same time.

10 REM

APPLESOFT SUBROUTINES
FOR HIGHER TEXT

800 REG\$ = CHR\$(18):WID\$ = CHR\$(23):TALL\$ = CHR\$(20):EX\$ = CHR\$(5):LRG\$ = CHR\$(12):BOLD\$ = CHR\$(25):UNBOLD\$ = CHR\$(26)

810 BG\$ = CHR\$(2):B0\$ = BG\$ + "0"
"B1\$ = BG\$ + "1":B2\$ = BG\$ + "2":B3\$ = BG\$ + "3":B4\$ = BG\$ + "4":B5\$ = BG\$ + "5":B6\$ = BG\$ + "6":B7\$ = BG\$ + "7"

820 CLR\$ = CHR\$(3):WHITE\$ = CLR\$ + "0":GEEN\$ = CLR\$ + "1":VIO
LT\$ = CLR\$ + "2":OURANGE\$ = CLR\$ + "3":BLUE\$ = CLR\$ + "4":
YELLOW\$ = CLR\$ + "5":LAVNDER\$ = CLR\$ + "6":PINK\$ = CLR\$ + "7":AQUA\$ = CLR\$ + "8":BK\$ = CLR\$ + "9"

830 HOM\$ = CHR\$(12):ELN\$ = CHR\$(14):EPG\$ = CHR\$(15):SCROLL
\$ = CHR\$(6):SMOOTH\$ = CHR\$(16):QUITSMOOTH\$ = CHR\$(0):
RETURN : REM

900 PRINT REG\$; RETURN
901 PRINT WID\$; RETURN
902 PRINT TALL\$; RETURN
903 PRINT EX\$; RETURN
904 PRINT LRG\$; RETURN
905 PRINT BOLD\$; RETURN
906 PRINT UNBOLD\$; RETURN : REM
910 PRINT B0\$; RETURN
911 PRINT B1\$; RETURN
912 PRINT B2\$; RETURN
913 PRINT B3\$; RETURN
914 PRINT B4\$; RETURN
915 PRINT B5\$; RETURN
916 PRINT B6\$; RETURN
917 PRINT B7\$; RETURN : REM

920 PRINT WHITE\$; RETURN
921 PRINT GEEN\$; RETURN
922 PRINT VIOLT\$; RETURN
923 PRINT OURANGE\$; RETURN
924 PRINT BLUE\$; RETURN
925 PRINT YELLOW\$; RETURN

926 PRINT LAVNDER\$; RETURN
927 PRINT PINK\$; RETURN
928 PRINT AQUA\$; RETURN
929 PRINT BK\$; RETURN : REM

930 PRINT HOM\$; RETURN
931 PRINT ELN\$; RETURN
932 PRINT EPG\$; RETURN : REM

940 PRINT SCROLL\$; RETURN
941 PRINT SMOOTH\$; RETURN
942 PRINT QUITSMOOTH\$; RETURN

10 REM
APPLESOFT HIGHER TEXT DEMO
USE WITH FP SUBROUTINES

50 IF PEEK(2045) = 76 THEN 90
60 PRINT CHR\$(4)"BRUN LOMEM":
& LOMEM: 16384
70 PRINT CHR\$(4)"BRUN HIGHER T
EXT"
80 CALL 3072
90 GOSUB 800
300 FLAG = 1: PRINT "DSP MODE (R/
W/T/E/L) ": GET FT\$
310 PRINT "BG COL (W/G/V/O/B/K)
": GET COL\$
320 PRINT "CHR COL (W/G/V/O/B/Y/
L/P/A/K) ": GET CH\$
330 IF FT\$ = "R" OR FT\$ = "T" THEN
PRINT "REG OR BOLD ": GET B\$
400 ON FT\$ = "R" GOSUB 900: ON F
T\$ = "W" GOSUB 901: ON FT\$ =
"T" GOSUB 902: ON FT\$ = "E"
GOSUB 903: ON FT\$ = "L"
GOSUB 904
410 ON COL\$ = "W" GOSUB 920: ON
COL\$ = "G" GOSUB 921: ON COL
\$ = "V" GOSUB 922: ON COL\$ =
"O" GOSUB 923: ON COL\$ = "B"
GOSUB 924: ON COL\$ = "Y" GOSUB
925: ON COL\$ = "L" GOSUB 926
: ON COL\$ = "P" GOSUB 927: ON
COL\$ = "A" GOSUB 928: ON COL
\$ = "K" GOSUB 929
430 ON B\$ = "R" GOSUB 906: ON B\$
= "B" GOSUB 905
440 IF FLAG THEN FLAG = 0:COL\$ =
CH\$: GOSUB 910: GOTO 410
450 PRINT "HELLO": END

10 REM

EXEC FILE CREATE

```
20 D$ = "[D]": REM CTRL D
30 PRINT D$;"OPENSUBFILE"
40 PRINT D$;"WRITE SUBFILE"
50 POKE 33,33
60 LIST 100,942
70 PRINT D$;"CLOSE"
80 TEXT : END
```

>LIST

10 REM

INTEGER BASIC SUBROUTINES
FOR HIGHER TEXT

```
100 REG=900:WIDE=901:TALL=902:EXPND=
    903:LARGE=904:BOLD=905:UNBOLD=
    906
110 BKGND0=910:BKGND1=911:BKGND2=
    912:BKGND3=913:BKGND4=914:BKGND5
    =915:BKGND6=916:BKGND7=917
120 WHITE=920: LET GREEN=921:VIOLET=
    922:ORANGE=923:BLUE=924:YELLOW=
    925:LAVENDER=926:PINK=927:AQUA=
    928:BLACK=929
130 HOME=930:EOL=931:EOP=932:SCROLLU
    P=940:SMOOTH=941:UNSMOOTH=942
```

>LIST

10 REM

INTEGER BASIC HIGHER TEXT DEMO

USE WITH INTEGER SUBROUTINES

```
70 IF PEEK (2045)<>76 THEN PRINT
    "BRUN HIGHER TEXT": REM
CTRL D IN QUOTES
80 CALL 3072
300 FLAG=1: PRINT "BSP MODE (R/W/T/E
    /L) ": GOSUB 600:FT=KEY
310 PRINT "BG COL (W/G/V/O/B/K) "
    : GOSUB 600:COL=KEY
320 PRINT "CHR COL (W/G/V/Q/B/Y/L/P/
    A/K) ": GOSUB 600:CH=KEY
330 IF FT= ASC("R") OR FT= ASC(
    "T") THEN GOSUB 500
400 FT=(FT=210)+2*(FT=215)+3*(
    FT=212)+4*(FT=197)+5*(FT=204
    )+899: GOSUB FT
410 XX=(COL=215)+2*(COL=199)+3*(
    COL=214)+4*(COL=207)+5*(COL=
    194)+6*(COL=217)+7*(COL=204
    )+8*(COL=208)
412 COL=XX+(9*(COL=193)+10*(COL=
    203)+919): GOSUB COL
430 IF NOT FLAG THEN 450
440 FLAG=0:COL=CH: GOSUB 910: GOTO
    410
450 PRINT "HELLO": END
500 PRINT "REG OR BOLD ": GOSUB
    600:B=KEY:B=(B=194)+2*(B=210
    )+904: GOSUB B: RETURN
600 KEY= PEEK (-16384): IF KEY<
    128 THEN 600: POKE -16368,0
    : RETURN
```

```
900 PRINT "[R]": RETURN
901 PRINT "[W]": RETURN
902 PRINT "[T]": RETURN
903 PRINT "[E]": RETURN
904 PRINT "[Q]": RETURN
905 PRINT "[Y]": RETURN
906 PRINT "[Z]": RETURN : REM

910 PRINT "[B10]": RETURN
911 PRINT "[B11]": RETURN
912 PRINT "[B12]": RETURN
913 PRINT "[B13]": RETURN
914 PRINT "[B14]": RETURN
915 PRINT "[B15]": RETURN
916 PRINT "[B16]": RETURN
917 PRINT "[B17]": RETURN : REM

920 PRINT "[C10]": RETURN
921 PRINT "[C11]": RETURN
922 PRINT "[C12]": RETURN
923 PRINT "[C13]": RETURN
924 PRINT "[C14]": RETURN
925 PRINT "[C15]": RETURN
926 PRINT "[C16]": RETURN
927 PRINT "[C17]": RETURN
928 PRINT "[C18]": RETURN
929 PRINT "[C19]": RETURN : REM

930 PRINT "[L]": RETURN
931 PRINT "[N]": RETURN
932 PRINT "[O]": RETURN : REM

940 PRINT "[F]": RETURN
941 PRINT "[P]": RETURN
942 PRINT "[@]": RETURN : REM
```

VII GLOSSARY

Note: Definitions used in this glossary may often refer to terminology within the context of, or pertaining only to the characteristics of, the Higher Text program, and not necessarily to general usage.

access	to locate and retrieve data	Ctrl	abbreviation for control or control character
address	a location in memory	cursor	usually a blinking character on the screen display which indicates where the next character typed will be placed.
allocate	to set aside or reserve space		
ASCII	an industry standard system of 128 computer codes assigned to specified alphanumeric and special characters.	data	facts or information used by or in a computer program
background	in Higher Text, the field upon which characters are printed, also used as a color reference by the Ctrl B function	default	the nominal value or condition assigned to a parameter when not specified by the user
base	a reference to number systems, i.e., base 10, the decimal number system	delete	to edit out a portion of text or a matrix
BASIC	a BASIC program, either Integer or Applesoft	display	the output of the Apple or program to a television set or monitor
binary	the base two number system, composed solely of the numbers 0 and 1	dithering	a process of dot mixing to produce additional hi-res colors
bit	one unit of binary data, either a zero or a one	DOS	the Apple Disk Operating System
BLOAD	DOS command to load a binary program	dot matrix	a grid or graph of specific dimensions used for drawing a character by placement of certain dots
boot	to bring the Disk Operating System up	editor	a program to automate the process of creating or changing characters
BRUN	DOS command to run a binary program	enter	a means to obtain access to a program or subroutine from keyboard or direct mode
byte	a hexadecimal representation of eight binary bits	EXEC file	a DOS text file which when called by the EXEC command reads data into Apple memory as if it were entered from the keyboard
character	one of the 128 available in the standard ASCII set or font	exit	a means to return to BASIC or direct mode from within a program
CHRS(x)	Applesoft function which prints the alphanumeric or special character specified by the ASCII value assigned to x	FID	a file transfer utility on the Apple DOS 3.3 master diskette
clear	in the editor, removes a dot from the matrix	file	data that has been saved to a diskette; may be a BASIC program or a Higher Text generator or font
command	an instruction to the program, usually input by the user	filename	name of a file that has been saved to diskette
complement	in colors, the opposite; the color 180° opposed in a color wheel. Blue and orange are complementary colors	font	a specific Higher Text character set
concatenate	to make one character string from two	generator	Higher Text operating system which includes at least one font
configuration	a specific group of software or hardware in a standard format		
control character	the first 32 characters of the ASCII set which do not print but instead perform particular computer functions		

hexadecimal	the Base 16 number system, composed of the numbers 0-9 and A-F, inclusive	program control	normally used to refer to instructions issued while a program is running
hi-res screen	a memory area in the Apple where high resolution graphics may be displayed; two such screens are available	protect	to prevent an area of memory from being overwritten
home	the normal or default position of a cursor	RAM	random access memory, memory that is available to the user
imbed	to implant or place within	register	a single memory location in which data may be stored
immediate mode	the normal condition of an Apple when a program is not running and commands may be entered from the keyboard	resides	literally, lives in; refers to a specific memory area in which a program or data may be found
input	to enter data or commands into a computer program	routine	a portion of a program in which a specific function is accomplished
insert	in editing, to place additional characters within a string or to add dots to a matrix	SAVE	DOS command to save a BASIC program
inverse line	the opposite of one line of text on the display screen. In text mode 24 lines are available	save	general term which references the saving to disk of any program or file
LOAD	DOS command to load a BASIC program	scroll	to move a line of text (usually upward) on the screen
load	general term which references the bringing in from disk of any program or file	set	in the editor, to place a dot in the dot matrix
master diskette	a diskette which has been updated to boot on any system	slave diskette	a diskette which will boot only on a system the same size as which it was created
menu	a screen display allowing the user to select from a number of choices	string	a group of ASCII characters, usually enclosed by quotes
mnemonic	a made-up word or abbreviation whose name conveys some indication of its function	subroutine	a portion of a program in which a specific function is accomplished, normally on more than one occasion
mode	a particular sub-type of operation	suffix	In Higher Text, a portion of a command which requires additional preceding data
module	a portion of a program devoted to a specific function	system	a collection of routines or programs operating as one
numeric	an ASCII character in the range 0-9	text	a line or string of ASCII characters
output	a computer response to commands or data which will print to a video display or hardcopy device (printer)	text screen	the normal Apple screen display, not used by Higher Text, which allows 24 lines of 40 characters each
page	a 256 byte portion of memory identified by its hex address	toggle	to switch from one mode to another. In Higher Text, usually by issuing the same command
parameter	a constant or value which a program requires to function, often specified by the user	variable	an alphanumeric representation to which may be assigned a number of real values
pointer	a memory location which contains the address of another memory location	window	a portion of the video display which is blocked off and assigned a specific duty
prefix	in Higher Text, a command which requires additional data		

I Quick Reference Command Summary

A. Higher Text Generator Commands

Command	CHR\$	Description	Page
CALL 3072	-	Cold start if HT generator in memory	7
CALL 3075	-	Warm start if HT generator in memory	7
Ctrl Q	17	Select Large Font	8
Ctrl E	5	Select Expanded Small Font display mode	10
Ctrl R	18	Select Regular Small Font display mode	10
Ctrl T	20	Select Tall Small Font display mode	10
Ctrl W	23	Select Wide Small Font display mode	10
Ctrl Y	25	Select Bold Face mode	9
Ctrl Z	26	Deselect Bold Face mode	9
Ctrlr C+	3+	Color assignment prefix	11
Ctrl B0	2+0	Assign current color to background	12
Ctrl A*	-	Upper case shift lock	13
Ctrl S*	-	Lower case shift lock	13
Ctrl X*	-	Cancel current line being typed	13
Ctrl H	-	Move cursor one position left	13
Ctrl I	-	Move cursor one position right	13
Ctrl J	-	Move cursor one line down	13
Ctrl K	-	Move cursor one line up	13
Ctrl L	12	Clear screen, home cursor	14
Ctrl N	14	Clear from cursor to end of current line	14
Ctrl O	15	Clear from cursor to end of page	14
Ctrl F	6	Scroll text up one line	14
Ctrl P	16	Select smooth scroll mode	14
Ctrl @	0	Deselect smooth scroll mode	14
Ctrl B+	2+	Background mode assignment prefix	26
Ctrl V*	-	Select or deselect key click function	28
Ctrl G	7	Bell	28

* May be used in immediate mode (from the keyboard) only

+ Indicates suffix required

Quick Reference Command Summary

B. Higher Text Character Editor Commands

Command	Description	Page
SYSTEM COMMANDS		
Ctrl L[ower]	Select/deselect Large Font lower case	15
1	Select Small Font 1	16
2	Select Small Font 2	16
3	Select Small Font 3	16
Ctrl C[ancel]	Exit editor, return to BASIC	17
RUN	Return to editor from BASIC	16
E[ditor]	Run the editor not currently in use	17
D[isk]	Display disk catalog	18
L[oad]	Load a font from disk	18
O[utput]	Output (save) a font to disk	18
I	Move system cursor up one line	18
J	Move system cursor left one position	18
K	Move system cursor right one position	18
L	Move system cursor down one line	18
P[lace]	Place a character in the edit window	18
[return]	Return a character from the edit window to the font	18
R[estart]	Clear edit window, restart editing	19
X	Set a dot in the matrix (X marks the spot)	19
C[lear]	Clear a dot in the matrix	19

Detach at perforation

EDIT WINDOW COMMANDS

W	Move edit cursor up one line	19
A	Move edit cursor left one position	19
S	Move edit cursor right one position	19
Z	Move edit cursor down one line	19
Ctrl W	Scroll character in edit window up one line	19
Ctrl A	Scroll character in edit window left one position	19
Ctrl S	Scroll character in edit window right one position	19
Ctrl Z	Scroll character in edit window down one line	19
Y	Delete horizontal line at edit cursor position	20
G	Delete vertical line at edit cursor position	20
H	Insert vertical line at edit cursor position	20
B	Insert horizontal line at edit cursor position	20

VIII USING A GENERATOR IN COMMERCIAL PROGRAMS

As you are aware from the copyright notice on the inside front cover of this manual, Higher Text is a copyrighted program. It is possible to make arrangements to use Higher Text Generator commercially by contacting:

Robert C. Clardy,
Agent for Darrell and Ron Aldrich
5221 120th Ave. SE
Bellevue, WA 98006

DISCLAIMER

This manual and the accompanying diskette are available only to members of Apple Pugetsound Program Library Exchange. Every effort has been made to provide error free programs and documentation. Inevitably some may remain. A.P.P.L.E. denies any responsibility for loss or damage of programs or equipment, direct or indirect, even if A.P.P.L.E. has been advised of the possibility of such damages.

Apple
PugetSound
Program
Library
Exchange

304 Main Ave. S.
Suite 300
Renton, Washington
98055
(206) 271-4514