SUP'R'TERMINAL

M & R Enterprises
Sunnyvale, California

PRELIMINARY

DISCLAIMER OF ALL WARRANTIES AND LIABILITY

M&R ENTERPRISES makes no warranties, either express or implied, with respect to this manual or with respect to the software described in this manual; its quality, performance, merchantability, or fitness for any particular purpose. M&R ENTERPRISES' software is sold "as is". the entire risk as to its quality and performance is with the buyer. Should the software or hardware prove defective following their purchase, the buyer assumes the cost of all necessary servicing, repair or correction, and any incidental or consequential damages. In no event will M&R ENTERPRISES be liable for direct, indirect, INCIDENTAL or consequential damages resulting from any defect in the software or hardware, even if M&R ENTERPRISES has been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you.

The information in this manual is believed to be correct at the time of publication, but M&R ENTERPRISES assumes no liability arising from the use of this manual.

NINETY-DAY WARRANTY

M&R ENTERPRISES warrants the products it manufactures against defects in material and workmanship for a period of ninety days from the date of purchase.

During said warranty period, M&R ENTERPRISES will repair (or at its option replace) at NO-CHARGE, components that prove to be defective, provided the product is returned, shipping prepaid, to:

M&R ENTERPRISES
285 Sobrante, Suite E
Sunnyvale, California 94086

PROOF OF PURCHASE

Your sales receipt is your warranty validation. Dated proof of purchase (such as bill of sale or cancelled check) must be provided when requesting warranty work to be performed.

"NO OTHER WARRANTIES ARE EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. M&R ENTERPRISES IS NOT RESPONSIBLE FOR CONSEQUENTIAL DAMAGES." Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

NOTICE

M&R ENTERPRISES reserves the right to make improvements in the product described in this manual at any time and without notice.

Documentation By

THE INNER LOOP

P.O. Box 31324
San Francisco, California 94131

Documentation Written By
Morgan P. Caffrey

Illustrations & Layout By
Douglas J. Platz

SUP'R'TERMINAL Design By
John R. Wilbur Jr.

Firmware By
Andy Hertzfeld

INTRODUCTION

Welcome to the world of 80-column video display on the APPLE II computer!

What SUP'R'TERMINAL Is

The SUP'R'TERMINAL is an 80-character per line, 24-line per screen peripheral interface "card" (circuit board) used with a separate adaptor board. It requires a video "monitor" instead of a standard or color television set.

What SUP'R'TERMINAL Does

SUP'R'TERMINAL provides an independent video display storage area within the APPLE II. From this display storage area, a video signal is produced and routed to your video display monitor via a separate video cable. When SUP'R'TERMINAL is turned on, normal character output to the standard APPLE II screen is inhibited. The APPLE II graphics are not displayed on the SUP'R'TERMINAL monitor. Graphics features can still be used, but only when two monitors are connected simultaneously. The color signal associated with the graphics feature is considerably reduced.

What Is Expected Of The Reader

This manual is written with the expectation that the user is familiar with the APPLE II power-on sequence, normal keyboard functions, etc. Users who are exploring the APPLE II for the first time should become familiar with the available user's guides:

"APPLE BASIC PROGRAMMING MANUAL"
"APPLESOFT BASIC PROGRAMMING REFERENCE MANUAL"
"DOS 3.2 INSTRUCTIONAL AND REFERENCE MANUAL"
"APPLE REFERENCE MANUAL"
"APPLE PASCAL REFERENCE MANUAL"

NOTE: It is not necessary to read ALL of these manuals. They are available for different system configurations and all contain useful general information.

This manual consists of the following sections:

1. **Installation Guide**
2. **Operator's Guide**
3. **Programmer's Guide**
4. **Sample Program ListingS**
5. **Appendices**

TABLE OF CONTENTS

Contents Continued

SECTION 4 - SAMPLE PROGRAMS

SECTION 1 - INSTALLATION GUIDE

What Happens During Instalation

One integrated circuit (IC) is removed from its socket in the APPLE II main circuit board ("motherboard"). The IC chip is transferred (inserted) to a SUP'R'TERMINAL adaptor board.

The adaptor board is inserted in the socket from which the IC chip was removed.

The cable leading forward from the adaptor board is plugged into the main SUP'R'TERMINAL circuit card, completing the computer-to- SUP'R'TERMINAL connections.

The new video signal is routed to the video monitor from the video-out jack at the front of the main SUP'R'TERMINAL circuit card.

NOTE: This process requres a tool to remove and insert the IC chip. It is recommended that the novice NOT attempt this installation procedure. We suggest that dealer assistance be obtained for installation.

Parts Required:

1. SUP'R'TERMINAL main circuit card.
2. Adaptor plug-in board.
3. Video cable with connectors (The same kind as used with the APPLE II standard VIDEO OUT.
4. Video monitor with 8+ megahertz BANDWIDTH

Installation Steps

The installation steps described below void.the APPLE II warranty.

1. TURN OFF THE APPLE II. REMOVE THE POWER CABLE.

2. Use an IC puller to remove IC "C2" from its socket in the "motherboard". See Illustration 1. Be careful not to bend the pins. This IC chip will be used in step 3.

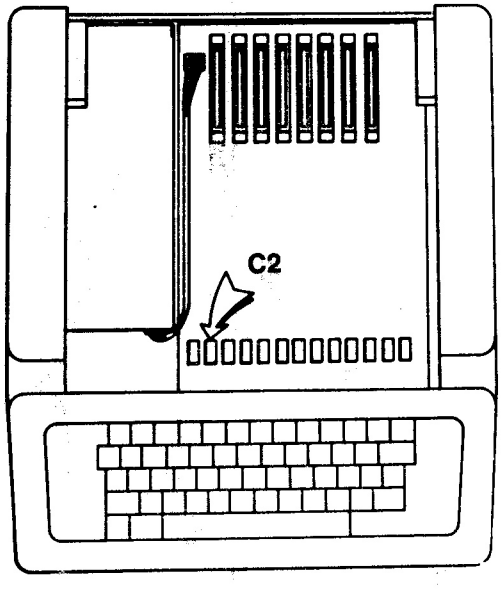


Illustration 1

3. Insert the IC chip into the empty socket of the adaptor board as shown in Illustration 2. Be sure to orient the front of the IC chip (with the half-moon or single dot) to the front of the adaptor board (with the wires). Use special care when inserting the IC chip not to bend the pins. Don't rush this. Most of us don't keep extra IC chips around.

WARNING: If you insert the IC chip backwards in the socket and apply power, THE CHIP WILL BE DESTROYED.

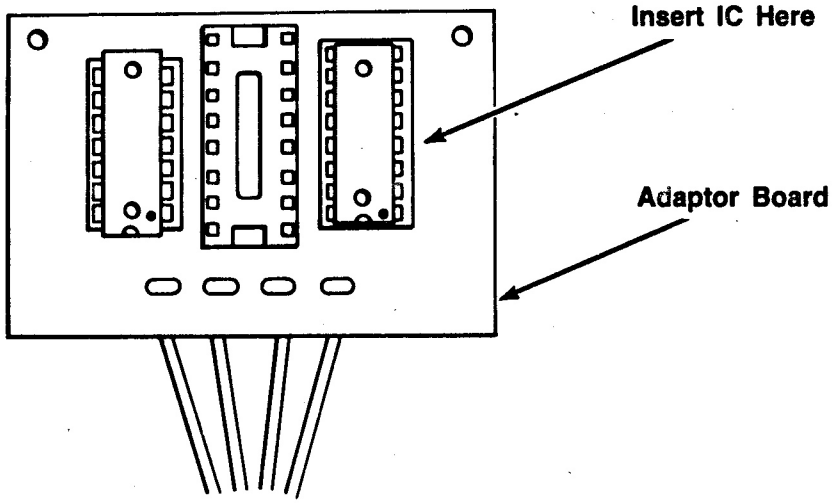


Illustration 2

4. Insert the adaptor board pins (wires to the front of the computer) into the empty "C2" socket on the "motherboard". Take care to hold the motherboard firmly to avoid excessive bending. Firm continuous pressure works best. See Illustration 3.

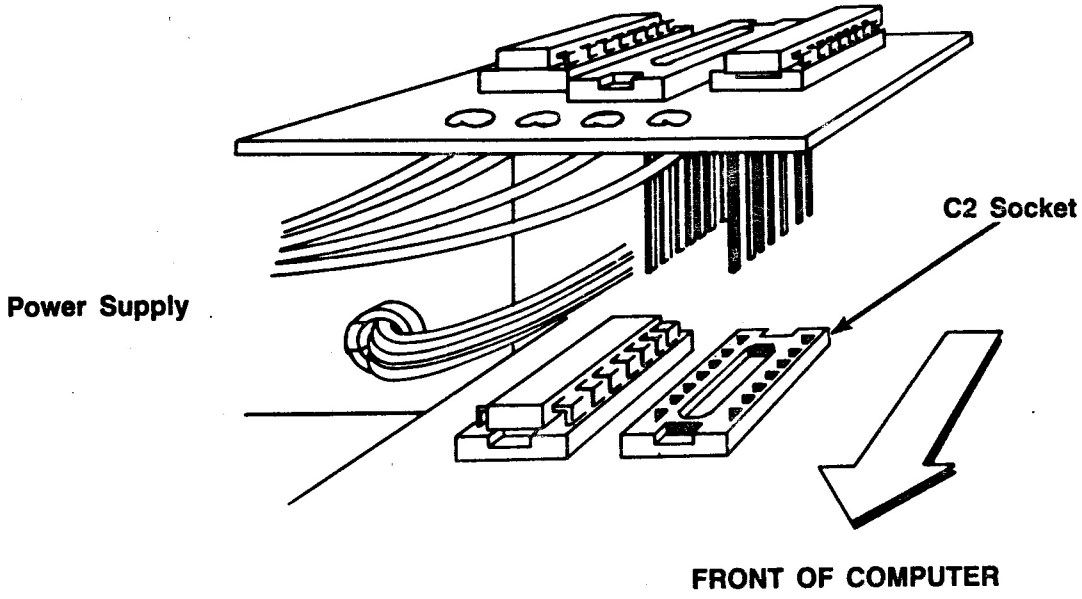


Illustration 3

5. Connect the cable leading from the adaptor board to the four-pin connector on the SUP'R'TERMINAL main circuit card (see Illustration 4). The black wire must be on top.

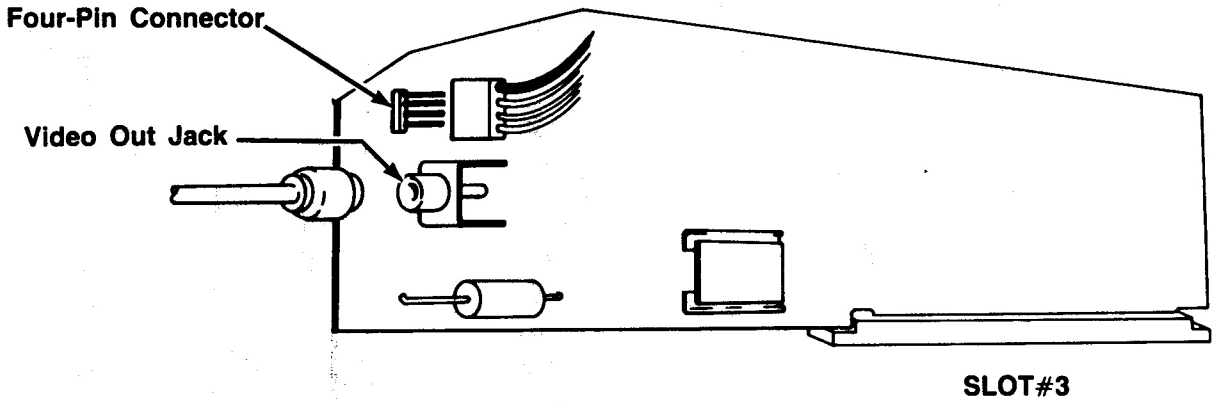


Illustration 4

6. Insert the video cable into the SUP'R'TERMINAL's VIDEO-OUT jack. See Illustration 5. Route the cable forward, around the side and out the small vent in the back. Connect the cable to the video monitor.

7. Insert the SUP'R'TERMINAL main circuit board into SLOT 3.

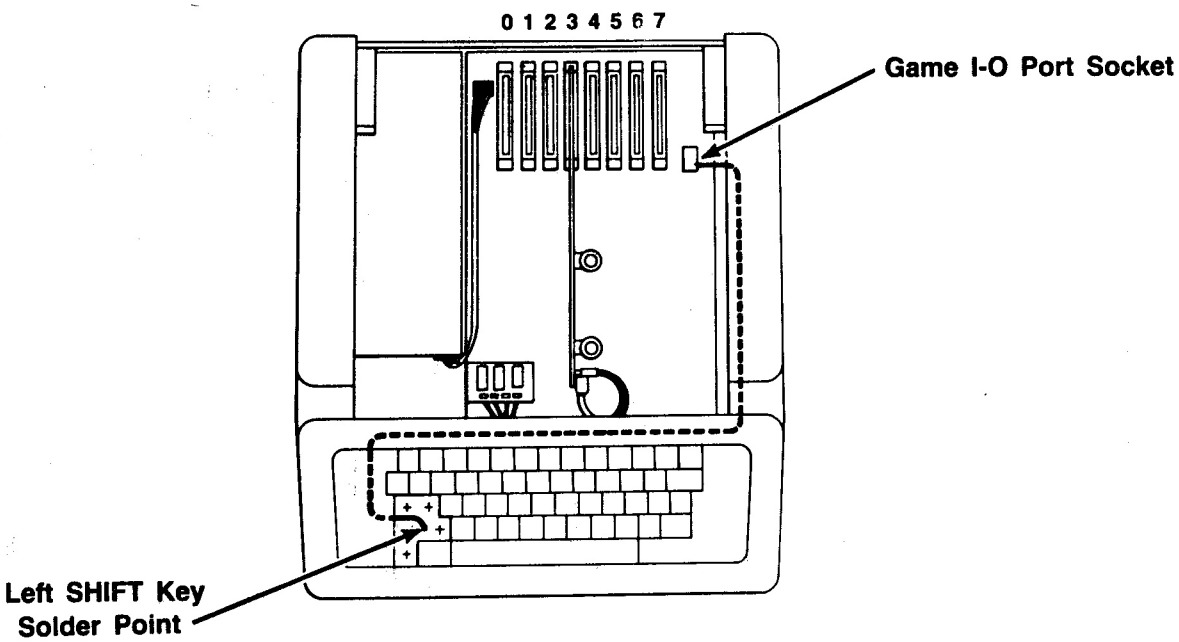


Illustration 5

Adjustments

You may adjust the strength of the video signal (brightness) and the sharpness of the CHARACTER IMAGE.

Read the Operation section of initialization and the, if required, return here to make any adjustments.

Video Signal Strength

Print a set of characters on the screen. Include both normal and inverse characters. Adjust the Video Strength Wheel until both kinds of character display are legible and comfortable for your eyes.

Video Signal Balance (Image Sharpness)

The horizontal bars on each character provide the image sharpness. Use the balance adjustment wheel to adjust for the sharpest possible character on your monitor (See Illustration 6).

This completes the installation process. The installed system will have no effect on APPLE II operation until intialized (this is automatic on PASCAL systems).

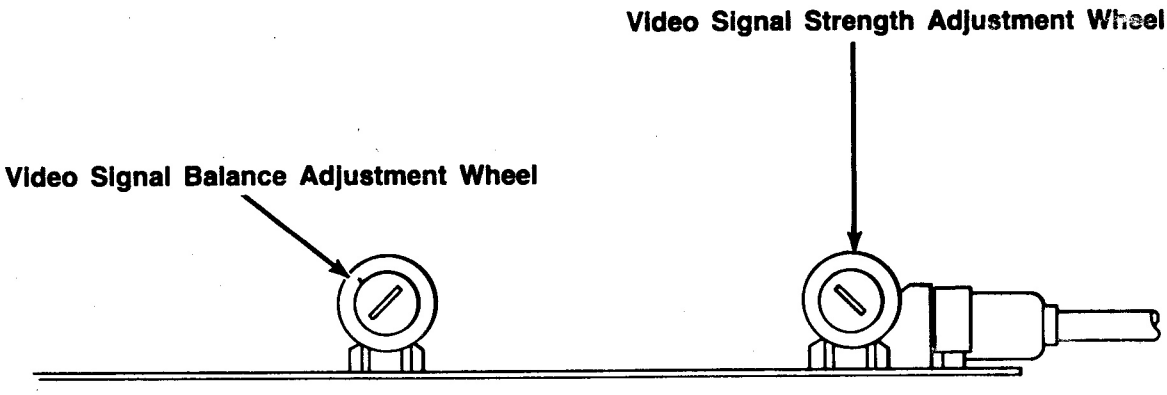


Illustration 6

Optional SHIFT-Key Modification

NOTE: The hardware modification described below permanently voids the APPLE II warranty. It is HIGHLY recommended that only a qualified technician perform this procedure.

This modification permits the shift key to perform the actual upper case function normally associated with that key.

To make the modification:

1. Remove the left SHIFT, CTRL and ESC key caps.
2. Insert a long (3 feet) wire through the opening to the left and route the wire to the 16-pin game I-O port area.
3. Strip the end of the wire near the SHIFT key and "tin" the end.
4. Uncover the P. C. board feed-through pad directly to the right of the SHIFT key. Tin this pad.
5. Remove extra keys as necessary.
6. Solder and neatly route the wire to the GAME I-O port.
7. Replace the key caps.
8. Remove excess wire, strip the end and insert the end in ANNUNCIATOR pin 4 of the 16-pin GAME I-O port.

While this procedure is not directly involved with the operation of SUP'R'TERMINAL, the software to handle this modification has been included in the program ROM. See Section 2, under SHIFT KEY for directions on how to use the feature.

SECTION 2 - OPERATOR'S GUIDE

The operation of the APPLE II with the SUP'R'TERMINAL is, in most ways, unchanged. Few general program alterations are required.

Initialization

The SUP'R'TERMINAL card needs to be initialized in order for the hardware to produce the video image. There are two kinds of initializations, "cold start" and "warm start". Each consists of the identical procedure but produces different results.

Cold Start

A cold start is performed only the first time SUP'R'TERMINAL is initialized after power up (or when forced by a CTRL-TR). A character set is transferred from the EPROM to the Character storage RAM. The screen is blanked and the cursor displayed at top left.

Warm Start

A warm start turns SUP'R'TERMINAL on, leaving the cursor at position 0 on the existing line position. The existing character set is not disturbed. The screen is not cleared.

Initialization From Integer & Applesoft BASICS

To initialize SUP'R'TERMINAL from either BASIC, enter:

PR #3 RETURN

This activates the board and screen RAM and, if a cold start, "downloads" a character font from the program EPROM to the character RAM area.

Disk System HELLO Program

To simplify the process, enter the following statements for either BASIC and make it part of each diskette's HELLO program.

```
10 D$=CHR$(4):REM CTRL-D
20 PRINT D$;"PR #3"
30 END
```

These statements initialize SUP'R'TERMINAL. Any other HELLO program statements may be carried out in their normal sequence.

Cassette System Initialization

Systems operating without the DISK II may also use SUP'R'TERMINAL. To initialize a cassette-based system, enter the "PR #3" at initialization. That is all that is necessary.

Initialization From PASCAL Systems

Place any bootable PASCAL diskette in the drive and type:

PR #6 (The APPLE II standard)

SUP'R'TERMINAL comes up initialized and running.

PASCAL Notes:

Everything in the PASCAL system works exactly as it does in BASIC except, as noted above, the ESCAPE commands are not valid in the PASCAL system.

All default PASCAL control characters work. The shift key mod, toggled by CTRL-V, also works.

80 Columns

One minor change can be made to take full advantage of SUP'R'TERMINAL capabilities. The "SETUP" program on the "APPLE 3" diskette can be used to change "screen width" from 79 to 80 characters. This will cause a full prompt line to be displayed.

Horizontal scrolling and screen switching are no longer needed so they are not implemented.

PASCAL Commands

CTRL-T commands must be printed; just typing them has no effect. This can be achieved from the FILER by transferring keyboard: to screen:

CTRL-A and CTRL-V work all the time.

Reset Recovery Procedure

If you happen to press RESET, SUP'R'TERMINAL will be temporarily disconnected. This leaves the screen exactly as it was but keystrokes APPEAR to have no effect (unless you have a separate TV displaying the standard APPLE II VIDEO OUT). The video display may waver somewhat in the event of a reset. This is normal.

To recover from the accidental RESET:

Standard ROM - You are actually in the monitor mode.

Type: **3D0G RETURN** - goes to BASIC
Type: **PR#3 RETURN** - performs a warm start
Cassette System Initialization
SUP'R'TERMINAL comes up initialized and running.

Autostart ROM - you are in BASIC:

Type **PR 3 RETURN** - performs a warm start.

Initialized Character Mode

Since all BASIC and Disk Operating System commands must be in upper case, initialization sets the character mode to upper case only. Note that this affects alphabetic characters only. Number keys always print numbers unless the SHIFT key is pressed.

SUP'R'TERMINAL Functions

Once your SUP'R'TERMINAL board is operating, APPLE BASIC and PASCAL control keys perform standard functions as described in the respective manuals.

ESCAPE Commands

The ESCAPE Key functions (which do NOT work with PASCAL systems) are briefly listed here.

ESCAPE Commands

ESC @	CLEAR SCREEN (HOME CURSOR)
ESC A	NON-COPY FORWARD SPACE
ESC B	CURSOR BACK
ESC D	CURSOR UP
ESC E	Clear to end of line
ESC F	Clear to end of screen
ESC I	Move cursor up - locks cursor-move mode.
ESC J	Move cursor left - locks cursor-move mode.
ESC M	Move cursor down - locks cursor-move mode.
ESC K	Move cursor right - locks cursor move mode

The four keys (I, J, K, M) form a diamond which points in the direction of the cursor movement. Once any of the four ESC key combinations have been used, all four of the keys in the diamond assume cursor movement control functions. No further use of the ESC key is required until the cursor-move mode is released. To release the cursor-move mode, press any other key. It will not print. This feature is exactly the same as described in the manuals for the Autostart ROM and APPLE II PLUS systems as well as the new "APPLE II REFERENCE MANUAL".

ESC Q -- Clears the screen, Cancels SUP'R'TERMINAL operation and enables standard APPLE 40 column video and color output.

Control Key Commands

Changing Display Case Mode

Remember that the starting case display is upper case so that all commands will work normally.

Lower Case

To display lower case letters:

Press **CTRL-A** once

This sets the character mode to lower case. To get single upper case letters from the lower case mode, press CTRL-A once, followed by the character to be capitalized. The next character you type will be capitalized. The character mode immediately returns to lower case.

Upper Case

To get into the upper case mode again:

Type: **CTRL-A CTRL-A**

(This can be done by holding down the CTRL key while pressing the "A" key twice.)

Special Features

Those who have been working with the Standard ROM system have some pleasant additional editing features to work with.

STOPLIST: CTRL-S
CTRL-S halts temporarily, anything being printed out. Any key starts the printing again. This feature differs from the autostart ROM in that it takes effect at any time rather than just after carriage returns.

SHIFT-KEY MODIFICATION: CTRL-V

CTRL-V is a "switch" command for use by those who have had the optional SHIFT key modification done (described in Section 1). The first use switches the feature on, the next use switches it off, etc.

This modification allows the SHIFT key to perform its normal function of upper and lower case. The SHIFT key will function to provide UPPER CASE. TO DISABLE THE FEATURE, OR HAVE ACCESS TO THE "<>@]^" characters, just repeat the command.

FT BRACKET "[" - CTRL-K
CTRL-K prints the left bracket character. This means that no other function can be assigned to the CTRL-K key.
Cursor Positioning

CURSOR COLUMN-TAB: CTRL-I
CTRL-I advances the cursor to the next multiple-of-8 column position, which is very useful for creating tables.

--HOME: CTRL-L (ASCII Form Feed): CTRL-L clears the screen and positions the cursor to the top left position. It replaces the Applesoft "HOME" and Integer CALL -936 commands. The ESC SHIFT-P" also provides the same function but only from the the keyboard (on input).

NOTE: If CTRL-L is directly followed by a RETURN, a SYNTAX ERROR message will occur. To prevent this, always type CTRL-X after an immediate-mode CTRL-L.

Also an accidental CTRL-L in the middle of a program will clear the screen of any display. This function cannot be disabled.

CURSOR ABSOLUTE POSITION: CTRL-^ (CTRL SHIFT-N)
CTRL-(SHIFT-N) Prepares SUP'R'TERMINAL to interpret the next two characters keyed as the absolute horizontal and vertical position of the cursor.

Altering Screen Display "Window"

The APPLE standard video display can be manipulated so that you control the amount of screen area being used for output. This is ordinarily done with "POKEs" to special locations. This feature is provided with SUP'R'TERMINAL using a special CTRL-KEY sequence.

(If you have trouble with this concept, enter and run the sample program "WINDOW MAKER" provided in the Section 4.)

The video display window is altered with the commands shown below:

CTRL-TT n - WINDOW TOP (n = 0 - 23)
CTRL-TB n - WINDOW BOTTOM (n = 1 - 24)
CTRL-TL n - WINDOW LEFT (n = 0 - 79)
CTRL-TW n - WIDOW WIDTH (n = 1 - 80)

NOTE: WINDOW LEFT + WIDTH MUST BE less than or equal to 80

The method of entering the "n" consists of pressing the appropriate character key. The ASCII value of that key is interpreted and used for the number.

To use this feature from the keyboard, search the table of ASCII characters (Appendix A) for the number associated with each key. The "space" character (160) is used for zero (since the actual zero point of the table consists of control characters which are being used in other ways). This feature is normally used under program control. It is explained in detail in Section 3.

NOTE: when the window TOP number is raised (moving the top of the window DOWN the screen) , the window BOTTOM lower limit raises accordingly (the bottom may not be higher than the top). Also, when the window LEFT number increases, the maximum window WIDTH goes down accordingly.

Character Display

The normal character display will be white characters on a black background. SUP'R'TERMINAL provides for inverse but not flashing characters.

To get inverse characters:

From Integer BASIC:
For INVERSE display: POKE 50,63 RETURN
For NORMAL display: POKE 50,255 RETURN

From Applesoft BASIC
For INVERSE display: INVERSE RETURN
FOR NORMAL DISPLAY: NORMAL RETURN

User-Defined Character Sets (Fonts)

You may define and store different character fonts of your own design. The character font format is described in Section 3. Once a font is defined and saved to diskette, you simply load the character font from the diskette to the proper location, described below.

The character set being displayed at any time is stored in RAM locations on the SUP'R'TERMINAL main circuit board. To change fonts, the contents of this RAM area must be replaced with the new font data.

Loading A Character Set

One character font resides on the SUP'R'TERMINAL program ROM. It is automatically loaded into the character RAM during cold-start initialization.

Once you have acquired additional fonts, and stored them on diskette(s), use one of the following procedures to load any font:

1. From Diskette to SUP'R'TERMINAL Character RAM

Type the following:

ESC-Q (Turns SUP'R'TERMINAL OFF)
POKE -12287,0 RETURN (Turns all SUP'R'TERMINAL memory banks off)
POKE -16206,0 RETURN (Turns on the Character RAM bank)
BLOAD anyfilename,A$C800 RETURN
PR #3 RETURN - performs warm start.

2. From Diskette to "Staging" Memory

The second method of loading character fonts is to pre-load them into alternate APPLE II RAM locations and load to the display area using special move commands described below. This area may be thought of as a "staging" area of memory. This method has the advantage of allowing much quicker font loading under software control.

Fonts loaded into the staging area must be of a special "compressed" format described in Section 3. Up to 10 fonts may be held in this staging area at one time. Each font may be loaded into the Character RAM area with only a few keystrokes.

To load the font into active use, type:

CTRL-TF n - Where n is a number key from 0 to 9.

The character keys in the ASCII table (Appendix A) from a "space" (160=0) to "*" (170=9) will also work.

The font is moved (from the addresses in the table below) into the active character set RAM. The font is translated during the move and the new font is immediately active.

These are the staging storage locations:

FONT#	HEX	STAGING LOC. DECIMAL	LOAD COMMAND
0 -	$1000	4096	CTRL-TF space
1 -	$1400	5120	CTRL-TF !
2 -	$1800	6144	CTRL-TF "
3 -	$1C00	7168	CTRL-TF #
4 -	$2000	8192	CTRL-TF $
5 -	$2400	9216	CTRL-TF %
6 -	$2800	10240	CTRL-TF &
7 -	$2C00	11264	CTRL-TF '
8 -	$3000	12288	CTRL-TF (
9 -	$3400	13312	CTRL-TF)

To load any compressed character set to any of the locations, type:

BLOAD anyfilename,Adecimal address
or
BLOAD anyfilename,A$hex address

NOTE: LOMEM (a BASIC command which "tells" BASIC where to store program statements or variables) should be set so as to avoid character set interference. If interference does occur, it will garble the character display on the monitor. To recover, just BLOAD the font again and type **"CTRL-TF n"** again. If the program has not accounted for this problem, it may reoccur.

This is done by interpreting the ASCII (numeric) value of the key pressed. Normally this is done only under program control and program use is covered in Section 3.

It can also be used to replace the "Immediate mode" VTAB function which does not work with SUP'R'TERMINAL. The sequence to get to any line at horizontal position zero is:

VTAB TABLE

COMMAND	KEY1 KEY2	CURSOR POSITION
CTRL-SHIFT-N	space space	- Line 0 (CTRL-Y also does this)
CTRL SHIFT-N	space !	- LINE 1
CTRL SHIFT-N	space "	- LINE 2
CTRL SHIFT-N	space #	- LINE 3
CTRL SHIFT-N	space $	- LINE 4
CTRL SHIFT-N	space %	- LINE 5
CTRL SHIFT-N	space &	- LINE 6
CTRL SHIFT-N	space '	- LINE 7
CTRL SHIFT-N	space (	- LINE 8
CTRL SHIFT-N	space)	- LINE 9
CTRL SHIFT-N	space *	- LINE 10
CTRL SHIFT-N	space +	- LINE 11
CTRL SHIFT-N	space !	- LINE 12
CTRL SHIFT-N	space -	- LINE 13
CTRL SHIFT-N	space !	- LINE 14
CTRL SHIFT-N	space /	- LINE 15
CTRL SHIFT-N	space 0	- LINE 16
CTRL SHIFT-N	space 1	- LINE 17
CTRL SHIFT-N	space 2	- LINE 18
CTRL SHIFT-N	space 3	- LINE 19
CTRL SHIFT-N	space 4	- LINE 20
CTRL SHIFT-N	space 5	- LINE 21
CTRL SHIFT-N	space 6	- LINE 22
CTRL SHIFT-N	space 7	- LINE 23

To move the cursor to an extended horizontal position, just change the KEY1 "space" character in the VTAB table above to a character higher up in the ASCII table. Use the table of ASCII values in Appendix A to learn which key(s) will move the cursor to the position you want. Experiment and see the effects you get.

Cursor Size And Flash Rate

You can control the cursor in ways not previously possible with the APPLE II 40-character screen. You may select from two cursor flash rates, two cursor sizes, a no-flash mode and a no-cursor mode.

To alter the cursor display:

Press CTRL-TC n - where n is a number in the range 0 - 7.

COMMAND KEY RESULT

CTRL-TC 0 - cursor is an underline flashing 4 times a second.
CTRL-TC 1 - cursor is a box flashing 4 times a second.
CTRL-TT n - WINDOW TOP (n = 0 - 23)
CTRL-TB n - WINDOW BOTTOM (n = 1 - 24)
CTRL-TL n - WINDOW LEFT (n = 0 - 79)
CTRL-TW n - WIDOW WIDTH (n = 1 - 80)

CONTROL CHARACTER FUNCTION TABLE

KEYS	FUNCTION
CTRL-A -	Upper and lower case switch
" **B** -	Unassigned
" **C** -	Stops BASIC programs
" **D** -	Disk Operating System flag
" **E** -	Unassigned
" **F** -	Unassigned
" **G** -	BELL
" **H** -	Backspace
" **I** -	Cursor column tab (next multiple-of-8 column)
" **J** -	Line Feed
" **K** -	Left bracket "["
" **L** -	Form Feed (HOME and clear)
" **M** -	RETURN
" **N** -	Unassigned
" **O** -	Unassigned
" **P** -	Unassigned
" **Q** -	Unassigned
" **R** -	Unassigned
" **S** -	Stoplist
" **T** -	Special function flag
" **TCn**	Alters cursor shape or flash rate
" **TFchar**	Load compressed character font
" **TR** -	SUP'R'TERMINAL cold-start RESET
" **TTchar**	Sets screen window top
" **TBchar**	Sets screen window bottom
" **TLchar**	Sets screen window left
" **TWchar**	Sets screen window width
" **U** -	Forward space
" **V** -	Switches SHIFT-key modification on or OFF
" **X** -	Cancels input line
" **Y** -	HOME cursor (does note scree

SECTION 3 - PROGRAMMER'S GUIDE

For the most part you may program with SUP'R'TERMINAL without paying special attention to its features. Only those areas dealing with screen formatting are affected.

There are three main areas of SUP'R'TERMINAL to understand in order do applications programming.

1. PROGRAM MOIFICATIONS

Program commands which are no longer effective and the commands or sequences to replace them with.

2. SUP'R'TERMINAL MEMORY MAP

Memory areas on the main circuit board including:
- a) Character RAM
- b) Screen RAM
- c) Program EPROM
- d) Cursor Control Registers

3. CHARACTER SETS
- a) Format
- b) Compression
- c) Storage & retrieval

These subjects are covered in detail below.

1. PROGRAM MODIFICATIONS

Command Replacement

The commands which need to be replaced in any program are those which make use of APPLE II monitor routines which specifially reference addresses in the range (hex $400 to $7FF). Since SUP'R'TERMINAL provides an entirely separate video display area, these commands do not work properly and must be replaced.

INTEGER BASIC	**APPLESOFT BASIC**
CALL -936	**HOME**
VTAB	**VTAB**
TAB	**HTAB**
	TAB
","	**","**

CALL -936 or HOME

The replacement command to blank the video screen and leave the cursor in the upper left corner of the screen is:

FROM THE KEYBOARD - CTRL-L

NOTE: This command causes a SYNTAX ERROR message unless followed by a CTRL-X or Backspace.

The standard alternative to this, ESC-@ also works in the immediate mode.

FROM WITHIN PROGRAMS - PRINT CHR$(140)

TAB or HTAB

The BASIC commands TAB, and HTAB, used with VTAB, normally allow the programmer or operator to direct the cursor to a specific location.

The TAB command still works in the normal fashion up to column 40. Thereafter use:

POKE 36,X

Where X is a number greater than the current cursor position and less than or equal to 80.

Notice that this means that tabbing backward will not work. In any situation where the programmer does not know the present cursor position, forcefully return the cursor to the horizontal zero position before tabbing to the destination location.

Absolute Cursor Positioning

VTAB no longer works. To replace VTAB and TAB-VTAB combinations, SUP'R'TERMINAL provides a single command sequence to position the cursor to any screen location. To direct the cursor to any absolute position:

PRINT CTRL-TSHIFT-N KEY1 KEY2
or
PRINT CHR$(148);CHR$(222);CHR$(ASCII);CHR$(ASCII)

The CTRL-T SHIFT-N prepares SUP'R'TERMINAL to interpret the ASCII value of the next two characters received or transmitted. The ASCII for a "space" (160) marks the zero point. Thus:

"CTRL-T^ space space" (CTRL-T^CHR$(160)CHR$(160))
sends the cursor to Line 0, horizontal position 0.
"CTRL-T^ !!" (CTRL-T^CHR$(161)CHR$(161))
sends the cursor to Line 1, horizontal position 1.

Below are further examples:

	COMMAND	POSITION
CTRL-T^	CHR$(160) CHR$(170)	- line 0, horizontal position 10
"	CHR$(161) CHR$(190)	- line 1, horizontal position 30.
"	CHR$(162) CHR$(200)	- line 2, horizontal position 40.
"	CHR$(163) CHR$(210)	- line 3, horizontal position 50.
"	CHR$(164) CHR$(220)	- line 4, horizontal position 60.
"	CHR$(165) CHR$(230)	- line 5, horizontal position 70.
"	CHR$(166) CHR$(240)	- line 6, horizontal position 80.

TAB & ","

The Applesoft TAB function (equivalent to the CTRL-I function provided with SUP'R'TERMINAL) is correctly translated. The cursor is advanced to the next multiple-of-8 column position.

The PRINT COMMAND "," function of both INTEGER and APPLESOFT languages are correctly interpreted for the respective languages. The TAB will operate correctly to the 80 column maximum rather than the standard 40-column.

2. SUP'R'TERMINAL MEMORY MAP

SUP'R'TERMINAL memory is divided into three "banks", all of which begin at the same base address ($C800). In order to access one of these memory banks, all banks must first be switched off. Any Read (PEEK) or Write (POKE) in the area of $CFF0 to $CFFF will turn all three banks OFF. Once all banks are turned off, the appropriate bank may be turned on.

The MEMORY areas are mapped as follows:

CHARACTER RAM - (HEX $400 bytes, $C800 to $CBFF)

Read or Write $CFFF (-12289) - Switches out all banks
Read or Write $C0B2 (-16206) - Switches in Character RAM bank

NOTE: This is "WRITE-ONLY" RAM. Character sets may be written into this area but may NOT be read back.

SCREEN RAM - (HEX $7C0 bytes, $C800 to $CFDF)

Read or Write $CFFF (-12289) - Switches out all banks
Read or Write $C0B4 (-16204) - Switches in Screen RAM bank
Directly store to $C800 to $CFDF (-14336 to -12352)

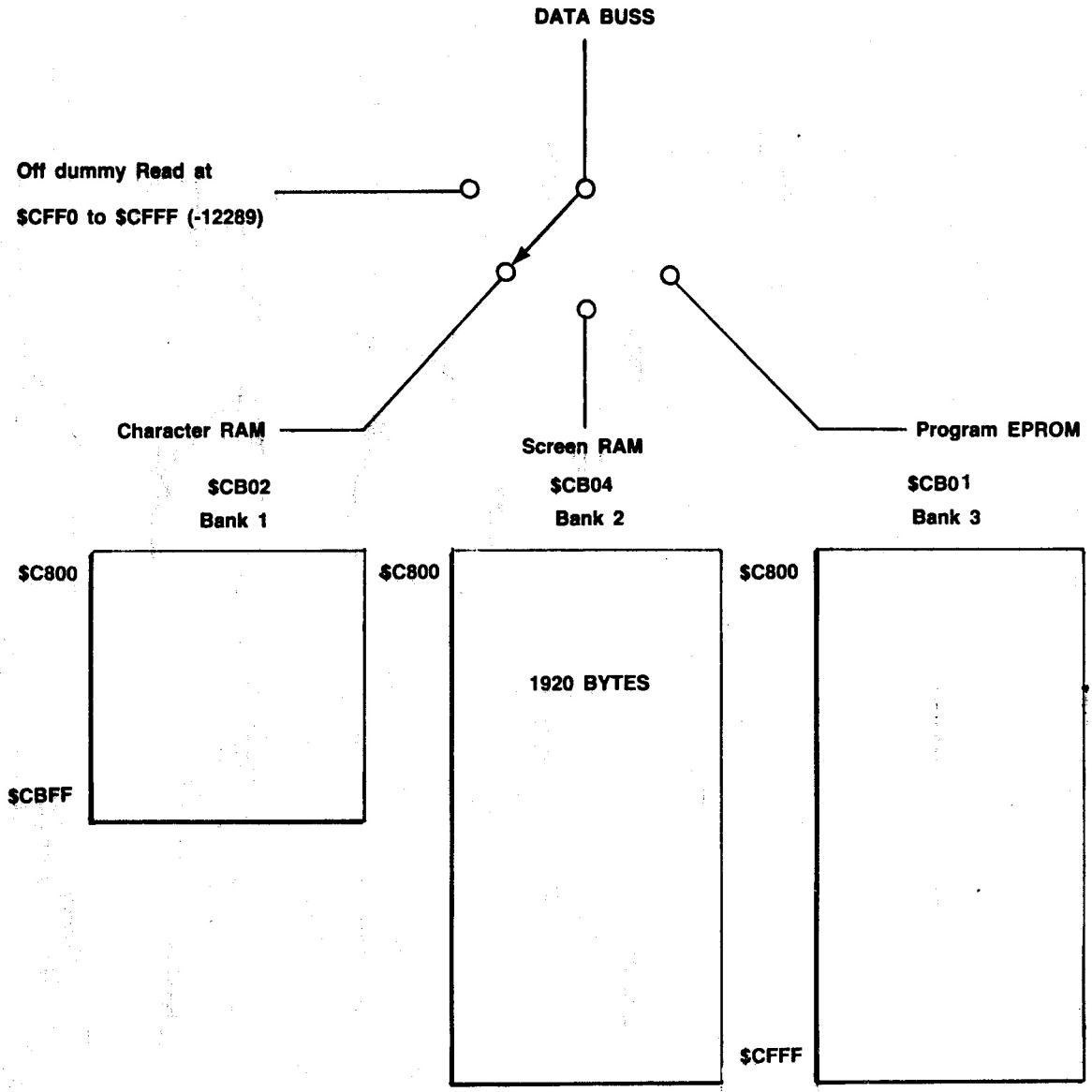
The bytes from $CF80 to $CFDF are available to the users. The user is cautioned, however, that MOST of these bytes ARE used by the controlling FIRMWARE. Experiment before committing code or data to any of these locations.

The RAM dedicated to the screen image is **LINEARLY** mapped (i. e. characters stored to seqential Screen RAM locations will appear on the screen in the proper visual sequence). See the program "SCREEN POKER" in Section 4 for an example.

PROGRAM EPROM - (HEX $800 bytes, $C800 to $CFFF; -14336 to -12289)

Read or Write $CFFF (-12289)
Read or Write $C3XX will enable ROM
Read or Write $C0B6 (-16202)
Read or Execute code in range $C800 - CFC0 (-14336 to -12352)

The accompanying diagram shows the memory "map" of SUP'R'TERMINAL.


DATA BUSS
Off dummy Read at
$CFF0 to $CFFF (-12289)
Character RAM
Screen RAM
Program EPROM
$CB02
Bank 1
$CB04
Bank 2
$CB01
Bank 3
$C800
$CBFF
$C800
1920 BYTES
$C800
$CFFF

Memory Map

Cursor Control Registers

The cursor may be directly manipulated under program control.

Four registers are used to control screen character-print location, the absolute character position and the condition of the cursor.

In order to access a register, store the register to be accessed in $C0B8 (-16200) Then store the new contents of the register in $C0B9 (-16199).

The following registers are accessible:

REGISTER 10 - Starting "RASTER" scan line for any character
BITS: 2-0 control start line number of 8-line character

BITS: 4 and 3 are not decoded
BITS: 6-5 are decoded as follows:
0 0 - no blink
0 1 - no cursor
1 0 - 4 blinks per second
1 1 - 2 blinks per second

REGISTER 11 - Ending raster scan line for any character (always 7).
REGISTER 14 - Cursor position high byte
REGISTER 15- Cursor position low byte

NOTE: Registers 14 and 15 form a two-byte number which assigns the cursor position relative to address $C800 (-14336) - which is line zero, horizontal position zero. The "Absolute Cursor Position" command already described performs the function of updating this register.

The contents of the registers may NOT be read back.

Cursor Display Mode

The CTRL-TC commands provide easy cursor-mode control:

CTRL-TC 0 - Cursor is an underline flashing 4 times a second.
CTRL-TC 1 - Cursor is a box flashing 4 times a second.
CTRL-TC 2 - Cursor is an underline with no flash.
CTRL-TC 3 - Cursor is a box with no flash.
CTRL-TC 4 - Cursor is an underline flashing 2 times a second.
CTRL-TC 5 - Cursor is a box flashing times a second.
CTRL-TC 6 - No cursor is displayed.

3. CHARACTER SETS

Format

Character sets are made up of 128 8-byte groups. Each 8-byte group is a "bit-map" which is interpreted by the SUP'R'TERMINAL hardware into signals representing a character.

The character format is 5 by 8. Bits 0-4 of each byte form the basis of the character. Bit 5, when set to 1, causes inverse display. Bits 6 and 7 are not decoded.

Character Storage

Characters are stored sequentially from $C800 to $CBFF. Thus:

ASCII 0 - $C800-$C807 (-14336 to -14329)
" **1 - $C808-$C80F (-14328 to -14321)**
.
.
.
" **128 - $CBF8-$CBFF (-12359 to -12351)**

Creating A Character

To create the character "A" in the normal (white on black) mode:

BITS:	7	6	5	4	3	2	1	0
BYTE 0 -	**0**	**0**	**0**	**0**	**0**	1	0	0
BYTE 1 -	0	0	0	0	1	0	1	0
BYTE 2 -	0	0	0	1	0	0	0	1
BYTE 3 -	0	0	0	1	0	0	0	1
BYTE 4 -	0	0	0	1	1	1	1	1
BYTE 5 -	0	0	0	1	0	0	0	1
BYTE 6 -	0	0	0	1	0	0	0	1
BYTE 7 -	0	0	0	0	0	0	0	0

Integer BASIC "POKE 50,63 or 255" and Applesoft INVERSE and NORMAL commands work in the standard manner. FLASH does not work since SUP'R'TERMINAL does not provide that feature.

In the inverse mode the same character format is used but bit 5 is also set or reset by the INVERSE or NORMAL command. All "1s" are interpreted as black, all "0s" as white. It is a possible to think of an extra bit 5, just to the right of bit 0, appearing when bit 5 is set. This extra bit fills the gap between letters:

BITS:	7	6	5	4	3	2	1	0	5
BYTE 0 -	0	0	1	1	1	**0**	1	1	1
BYtE 1 -	0	0	1	1	**0**	1	**0**	1	1
BYtE 2 -	0	0	1	**0**	1	1	1	**0**	1
BYtE 3 -	0	0	1	**0**	1	1	1	**0**	1
BYtE 4 -	0	0	1	**0**	**0**	**0**	**0**	**0**	1
BYtE 5 -	0	0	1	**0**	1	1	1	0	1
BYtE 6 -	0	0	1	**0**	1	1	1	**0**	1
BYtE 7 -	0	1	1	1	1	1	1	1	

In the example above, the character would appear as inverse (black on a white background).

NOTE: The inverse mode is interpreted only for the alpha characters. All numbers, symbols and the space character appear in the normal mode only.

Character Set Compression

Character sets may be converted into a compressed mode. This compressed mode saves disk space and allows the programmer or user to quickly change display fonts. This may be done in the middle of a program under program control. With a little more difficulty, it may also be performed from the keyboard.

Compression Technique

Each byte in a character definition is compared with the byte which follows. If a byte is found to be identical, a count is started. When the first different byte is encountered the count is encoded into the top three bits (5-7) as a binary count. This is, in effect, a pattern repetition count. Thus the letter "A" can be compressed as follows:

BITS:	7	6	5	4	3	2	1	0	
BYTE 0 -	0	0	0	0	0	1	0	0	- 0 repetitions, send as is.
BYTE 1 -	0	0	0	0	1	0	1	0	- 0 repetitions, send as is.
BYTE 2 -	0	0	0	1	0	0	0	1	- initiate count
BYTE 3 -	0	0	0	1	0	0	0	1	- repetition, send 00110001.
BYTE 4 -	0	0	0	1	1	1	1	1	- 0 repetitions, send as is
BYTE 5 -	0	0	0	1	0	0	0	1	- initiate count
BYTE 6 -	0	0	0	1	0	0	0	1	- 1 repetition, send 00110001.
BYTE 7 -	0	0	0	0	0	0	0	0	- 0 repetitions, send as is.

Thus the 8-byte letter "A" can be stored as the following six BYTEs:

			count--character				
BITS:	7	6	5--4	3	2	1	0
BYTE 1 -	0	0	0--0	0	1	0	0
BYTE 2 -	0	0	0--0	1	0	1	0
BYTE 3 -	0	0	1--1	0	0	0	1 - repetition count = 1
BYTE 4 -	0	0	0--1	1	1	1	1
BYTE 5 -	0	0	1--1	0	0	0	1 - repetition count = 1
BYTE 6 -	0	0	0--0	0	0	0	0

Thus The "space" character can be compressed to the single byte "E0".

For a program which compresses full fonts into the compressed mode, see FONT COMPRESSOR in Section 4.

Compressed Character Set Storage

Compressed character fonts may not be directly loaded into the Character RAM bank. They must be loaded into one of 10 staging areas and subsequently loaded into Character RAM via the CTRL-TF routines.

The character set and storage areas are:

FONT	BASE ADDRESS
0 -	$1000
1 -	$1400
2 -	$1800
3 -	$1C00
4 -	$2000
5 -	$2400
6 -	$2800
7 -	$2C00
8 -	$3000
9 -	$3400

For easiest loading into the staging area, first move the compressed set into the area in which it will normally be staged (i. e. font 3 in $1C00, font 7 in $2C00, etc.)

Next, BSAVE the font from its intended location:

BSAVE FONT2,A$1800,L(number of bytes)
BSAVE FONT7,A$2C00,L(number of bytes)

Remember that the number of bytes in a compressed font is variable, depending on how many pattern repetitions there are.

Loading Compressed Fonts Into Staging Area

If you have used the font-saving method described above, loading compressed character sets into a staging area will be done by typing:

BLOAD FONT0
BLOAD FONT2
BLOAD FONT7

But, if a font is to be loaded to a non-standard area then type:

BLOAD FONT0,A$2400 - Loads FONT0 to storage location for FONT5
BLOAD FONT7,A$1000 - Loads FONT7 to storage location for FONT0

Transferring Compressed Fonts Into Character RAM

This will usually be done by program command:

PRINT CHR$(148);"F";CHR$(160–) - FONTn
or
PRINT CHR$("CTRL-T");"F";CHR$(167) - FONT7
From the keyboard:

CTRL-TF "0" - FONT0
CTRL-TF "1" - FONT1
CTRL-TF "2" - FONT2
CTRL-TF "3" - FONT3
.
.
.
CTRL-TF "9" - FONT9

The key sequences above will transfer a compressed font from its staging area in the APPLE memory, into the SUP'R'TERMINAL Character RAM storage area. The compressed font is expanded to its attern during the transfer process. It immediately changes the displayed font.

Loading An Uncompressed Character Font

It is also possible to load a non-compressed font directly into Character RAM.

Under Program Control

1. Read or Write $CFFF (-12209) - Switches out all banks
2. Read or Write $C0B2 (-16206) - Switches in Character RAM.
3. BLOAD CHARSET,A$C800

Remember that the character set must be in the non-compressed mode in order for this to work.

Direct From The Keyboard

From the keyboard, the method to use is slightly more complex. You must turn SUP'R'TERMINAL off.

1. Press ESC-Q to exit SUP'R'TERMINAL mode. This means you must have a separate TV monitor attached to the APPLE II standard VIDEO OUT jack or be able to perform this routine without seeing the characters.

2. Read or Write $CFFF
3. Read or Write $C0B2
4. BLOAD charsetname,A$C800
5. PR #3

CONTROL ("CTRL") CHARACTERS

KEY	HEX	DECIMAL	FUNCTION
CTRL-A	$81	129	Upper-Lower case switch
CTRL-B	$82	130	Unassigned
CTRL-C	$83	131	Keyboard Program Interrupt
CTRL-D	$84	132	D.O.S Attention flag
CTRL-E	$85	133	Unassigned
CTRL-F	$86	134	Unassigned
CTRL-G	$87	135	Bell
CTRL-H	$88	136	Backspace
CTRL-I	$89	137	Tab to next multiple of 8
CTRL-J	$8A	138	Line feed
CTRL-K	$8B	139	Clear to end of screen
CTRL-L	$8C	140	Form feed (home and clear)
CTRL-M	$8D	141	Carriage return (generates a line feed).
CTRL-N	$8E	142	Unassigned
CTRL-O	$8F	143	Unassigned
CTRL-P	$90	144	Unassigned
CTRL-Q	$91	145	Cancel terminal mode. Return standard APPLE display (DOS intact).
CTRL-R	$92	146	Unassigned
CTRL-S	$93	147	Unassigned
CTRL-T	$94	148	Terminal "escape" mode
--CTRL-TF char			- "load" compressed font into character RAM.
	CHAR RANGE		
	$A0	160	From $1000 - $13FF
	$A1	161	From $1400 - $17FF
	$A2	162	From $1800 - $1BFF
	$A3	163	From $1C00 - $1FFF
	$A4	164	From $2000 - $23FF
	$A5	165	From $2400 - $27FF
	$A6	166	From $2800 - $2BFF
	$A7	167	From $2C00 - $3000
	$A8	168	From $3000 - $33FF
	$A9	169	From $3400 - $37FF
--CTRL-TR	Cold Start reset		
--CTRL-TC n	Alter Cursor Size & Flash rate		
	NUMBER RANGE		
	$B0	176	underline - 4 flashes per sec.
	$B1	177	box - 4 flash per sec.
	$B2	178	underline - no flash.
	$B3	179	box - no flash.
	$B4	180	underline - 2 flashes per sec.
	$B5	181	box - 2 flashes per sec.
	$B6	182	no cursor displayed.
--CTRL-TT char			Top of window (0 - 23)
-- CTRL-TB char			Bottom of window (0 - 23)
-- CTRL-TW char			Width of window (0 - 79)
-- CTRL-TL char			Absolute left margin (0 - 79)
CTRL-U	$95	149	Right arrow (BASIC ONLY)
CTRL-V	$96	150	Switch TO or FROM shift-key modification mode
CTRL-W	$97	151	Unassigned
CTRL-X	$98	152	Cancels Input line
CTRL-Y	$99	153	Home cursor (doesn't blank screen)
CTRL-Z	$9A	154	Clears line
*	$9C	156	Non-destructive (non-copy) forward space.
CTRL-SHIFT-M	$9D	157	Clear to end of line
CTRL-SHIFT-N	$9E	158	GOTO XY (X = HORIZONTAL, Y = VERTICAL) absolute cursor positioning)
*	$9F	159	Reverse line feed (cursor up)

* Not available from keyboard. Can be used only as part of a program.

ESCAPE ("ESC") COMMANDS
(BASIC only)

ESC @	Clears screen and homes cursor -- immediate mode only.
ESC A	Non-copy forward space
ESC B	Backspace
ESC C	Line feed
ESC D	Reverse line feed
ESC E	Clear to end of line
ESC F	Clear to end of screen
ESC I	Moves cursor up one line and locks screen into cursor-move mode
ESC J	Moves cursor left one space and locks screen as does ESC I
ESC K	Moves cursor right one space and locks screen as does ESC I
ESC M	Moves cursor down one line and locks screen as does ESC I
ESC Q	Exits SUP'R'TERMINAL mode

NOTE: The I, J, K, and M keys form a diamond which points the direction of cursor movement:

I

J **K**

M

SECTION 4 - SAMPLE PROGRAMS

WINDOW MAKER

```
  0 REM INTEGER VERSION: WINDOW MAKER
  1 REM WRITTEN BY MORGAN P. CAFFREY
  2 REM  SAN FRANCISCO, CALIFORNIA
  3 REM  JANUARY 23,1980 (V. 2)
  5 X$='X': REM REQUIRED TO ESTABLISH A 'CHR$' ROUTINE
  6 CHR=2053: REM LOCATION OF VALUE OF X$
  40 POKE CHR,140: PRINT X$: PRINT '                DEMONSTRATION PROGRAM              ': PRINT
  41 PRINT 'THIS PROGRAM ALLOWS YOU TO SET YOUR OWN VIDEO SCREEN LIMITS AND SEE THE RESULTS': PRINT :
PRINT
  49 REM DEFINE VARIABLES AND ARRAYS
  50 DIM A(4): FOR I=1 TO 4:A(I)=0: NEXT I
  60 DIM B(4):B(1)=23:B(2)=23:B(3)=79:B(4)=79
  70 DIM C(4): FOR I=1 TO 4:C(I)=0: NEXT I
  80 DIM C$(4):C$='TBLW'
  85 BASE=160: REM EQUALS '1' AFTER A CTRL-T T HAS BEEN ENTERED OR PRINTED
  86 MAX=BASE+80: REM MAX POSSIBLE WINDOW WIDTH
  87 CTRLL=140:CTRLT=148
  90 DIM D$(255): REM  STRING TO PRINT FOR DEMONSTRATION
100 REM FIND OUT WHERE TO PUT THE PRINT OUTPUT
110 INPUT 'WINDOW TOP (RANGE 0 - 23) ';A(1): IF A(1)<=B(1) AND A(1)>=0 THEN 120: PRINT 'OUTSIDE THE RANGE - TRY
AGAIN': GOTO 110
120 PRINT 'WINDOW BOTTOM (RANGE ';A(1);' - 23) ';: INPUT A(2): IF A(2)>=A(1) AND A(2)<=B(2) THEN 125: PRINT 'OOPS -
TRY AGAIN': GOTO 120
125 PRINT
130 INPUT 'WINDOW LEFT MARGIN (RANGE 0 - 79) ',A(3): IF A(3)<=B(3) AND A(3)>=0 THEN 140: PRINT 'OUTSIDE OF
RANGE - TRY AGAIN': GOTO 130
140 PRINT 'WINDOW WIDTH (RANGE 1 - ';79-A(3);') ';: INPUT A(4): IF A(4)>=1 AND A(4)<=79-A(3) THEN 150: PRINT ' RANGE
ERROR': GOTO 140
150 REM
160 PRINT : PRINT : PRINT ' WHAT IS TO BE PRINTED IN THE WINDOW?': PRINT 'PRESS RETURN TO GET A STANDARD
SAMPLE DISPLAY': INPUT D$
170 IF D$='' THEN D$='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'
180 INPUT 'HOW MANY REPETITIONS',Z: IF Z>255 THEN Z=255
200 REM COMPUTE NEW WINDOW VALUES
210 FOR I=1 TO 4
220 C(I)=BASE+A(I)
230 NEXT I
300 REM SET WINDOW
305 POKE CHR,CTRLT: PRINT X$;'R': REM  RESET ALL
310 FOR I=1 TO 4
320 POKE CHR,CTRLT: PRINT X$;C$(I,I);: POKE CHR,C(I): PRINT X$: PRINT : PRINT
330 NEXT I
335 PRINT : PRINT
400 REM NOW PRINT IT
410 FOR I=1 TO 25: PRINT D$;: NEXT I: PRINT : PRINT
415 INPUT 'PRESS RETURN TO CONTINUE',E$
420 REM NOW RESTORE TO NORMAL
450 POKE CHR,CTRLT: PRINT X$;'R': PRINT
460 INPUT 'ANOTHER? (Y/N) ',E$: IF E$='Y' THEN 100
500 REM RESTORE EVERYTHING AND END
510 POKE CHR,CTRLT
520 PRINT X$;'T';: POKE CHR,BASE: PRINT X$: PRINT : POKE CHR,CTRLT
530 PRINT X$;'B';: POKE CHR,BASE+23: PRINT X$: PRINT : POKE CHR,CTRLT
540 PRINT X$;'L';: POKE CHR,BASE: PRINT X$: PRINT : POKE CHR,CTRLT
550 PRINT X$;'W';: POKE CHR,MAX: PRINT X$: PRINT : POKE CHR,CTRLT
600 PRINT X$;'R': PRINT : PRINT 'BYE!': END
```

SCREEN 'POKER' DEMONSTRATION

```
  0 GOTO 1000
  1 REM ROUTINE TO POKE CHARACTERS TO INDIVIDUAL SCREEN LOCATIONS
  3 REM GO TO INDIVIDUAL LOCATIONS, REMOVE LINES 23,24,50,55 AND MAKE(55 GOTO 10)
  5 A=0
  8 BASE=-14336:LAST=-14335:A=0
  10 POKE CHR,CTRLL: PRINT X$: PRINT
  15 INPUT 'CHARACTER YOU WOULD LIKE TO SEE? ',A$
  23 FOR LINE=0 TO 23
```

```
 24 FOR POS=1 TO 80
 25 POKE -12287,0: REM  BANK SELECTS OFF
 26 POKE -16204,0: REM  SCREEN RAM ENABLE
 40 SPOT=BASE+(LINE*80)+POS: POKE SPOT, ASC(A$): POKE LAST,160:LAST=SPOT
 41 REM SPOT=LINE+POS. SET SPOT=CHAR. SET LAST = SPACE. SET LAST = SPOT.
 50 NEXT POS
 55 NEXT LINE
 60 FOR I=1 TO 200: NEXT I: GOTO 1050
500 POKE CHR,CTRLL: PRINT X$: PRINT
505 PRINT 'NOTE: THIS SAME FUNCTION MAY BE DONE WITH THE 'GOTO X,Y' COMMAND': PRINT : PRINT
510 INPUT 'CHARACTER YOU WOULD LIKE TO SEE ',A$
520 INPUT 'IN WHICH LINE SHOULD IT APPEAR (0 - 23) ',LINE: IF LINE>=0 AND LINE<=23 THEN 530: PRINT '': GOTO 520
530 INPUT 'IN WHICH POSITION (0 - 79) ',POS: IF POS>=0 OR POS<=79 THEN 550: PRINT '': GOTO 530
550 POKE -12287,0: REM    BANK SELECTS OFF
560 POKE -16204,0: REM  SCREEN RAM ENABLE
570 SPOT=BASE+(LINE*80)+POS: POKE SPOT, ASC(A$)
580 REM  SPOT=LINE+POS
590 INPUT 'ANOTHER (Y/N)',A$: IF A$='Y' THEN 500: GOTO 1050
1000 REM WRITTEN BY MORGAN P. CAFFREY
1010 REM SAN FRANCISCO, CALIFORNIA
1020 REM JANUARY 20,1980
1040 X$='X':CTRLT=148:CTRLL=140:CHR=5+ PEEK (74)+ PEEK (75)*256: REM  LOMEM + 5 = LOCATION OF X$
1045 BASE=-14336:LAST=-14335:A=0
1050 POKE CHR,CTRLL: PRINT X$: PRINT
1060 PRINT 'YOU MAY SEE SINGLE CHARACTERS MOVING TO EVERY SCREEN POSITION (ENTER 1)': PRINT
1070 PRINT 'YOU MAY SEND THE CHARACTER OF YOUR CHOICE TO ANY SCREEN POSITION (ENTER 2)': PRINT
1075 PRINT 'YOU MAY CHOOSE TO QUIT (ENTER 3)': PRINT
1080 INPUT '? ',A$:A= ASC(A$)-176
1100 IF A=1 THEN 10
1110 IF A=2 THEN 500
1120 IF A=3 THEN END
1130 IF A>3 OR A<1 THEN PRINT 'OUTSIDE OF LEGAL RANGE - PLEASE TRY AGAIN ': FOR I=1 TO 450: NEXT I: GOTO 1050
```

GOTO XY DEMONSTRATION

```
   0 GOTO 2000
  10 REM GOTO X,Y DEMONSTRATION
  20 POKE CHR,CTRLL: PRINT X$: PRINT : REM  CLEAR SCREEN
  25 BASE=160
  30 INPUT 'CHARACTER YOU WANT TO SEE? ',A$
  40 INPUT 'HORIZONTAL POSITION ',HRIZ:HRIZ=HRIZ+BASE: REM  GET PROPER OFFSET
  50 INPUT 'VERTICAL POSITION ',VERT:VERT=VERT+BASE: REM  GET PROPER OFFSET
  60 POKE CHR,CTRLL: PRINT X$: PRINT
  70 POKE CHR,GXY: PRINT X$;: POKE CHR,HRIZ: PRINT X$;: POKE CHR,VERT: PRINT X$;: PRINT A$: PRINT
  80 INPUT 'ANOTHER (Y/N) ',A$
  90 IF A$='Y' THEN 20
  95 GOTO 2050
 500 REM  EVERY POSITION
 505 ZERO=160: REM  THE 'SPACE' CHARACTER REPRESENTS 0: '!' = 1: '"' = 2: ETC.
 510 POKE CHR,CTRLL: PRINT X$: PRINT
 515 INPUT 'WHAT CHARACTER WOULD YOU LIKE TO SEE? ',A$
 520 PRINT X$: PRINT
 530 FOR VERT=0 TO 23
 531 C=VERT+ZERO
 532 FOR HRIZ=0 TO 79
 535 B=HRIZ+ZERO: REM  160+THE POSITION
 550 POKE CHR,GXY: PRINT X$;: POKE CHR,B: PRINT X$;: POKE CHR,C: PRINT X$;: PRINT A$;
 560 NEXT HRIZ
 570 NEXT VERT
 580 FOR N=1 TO 450: NEXT N: PRINT
 590 INPUT 'ANOTHER (Y/N) ',A$: IF A$='Y' THEN 510: GOTO 2050
1000 GOTO 2050
2000 REM  WRITTEN BY MORGAN P. CAFFREY
2010 REM  SAN FRANCISCO, CALIFORNIA
2020 REM  JANUARY 24,1980
2025 REM  CHR$ FUNCTION LOCATES X$ TO A FIXED POSITION - FIRST IN THE TABLE
2026 X$='0': REM  THE '0' IS A 'DUMMY' WHOSE ABSOLUTE POSITION IS DECIMAL 2053
2040 CHR=5+ PEEK (74)+ PEEK (75)*256: REM    LOMEM + 5 = LOCATION OF X$
2042 CTRLT=148:CTRLL=140:GXY=158
2045 BASE=-14336:LAST=-14335:A=0:GXY=158
2050 POKE CHR,CTRLL: PRINT X$: PRINT
2060 PRINT 'THIS ROUTINE USES THE 'GOTO X,Y' FEATURE OF SUP'R'TERMINAL': PRINT
2061 PRINT 'NORMALLY THIS IS ONLY DONE FROM WITHIN A PROGRAM': PRINT
```

```
2062 PRINT ' TO POSITION THE CURSOR - PRINT CHR$(158);CHR$(HORIZ);CHR$(VERT)'
2063 PRINT 'LIST THIS PROGRAM TO SEE AN INTEGER 'CHR$' ROUTINE': PRINT : PRINT : PRINT
2070 PRINT 'YOU MAY SEND THE CHARACTER OF YOUR CHOICE TO ANY SCREEN POSITION (ENTER 1)': PRINT
2071 PRINT 'YOU MAY CHOOSE TO SEND A CHARACTER TO EVERY POSITION ON THE SCREEN (ENTER 2) ': PRINT
2075 PRINT 'YOU MAY CHOOSE TO QUIT (ENTER 3)': PRINT
2080 INPUT '? ',A$: IF A$='' THEN 2050:A= ASC(A$)-176
2100 IF A=1 THEN 10
2110 IF A=2 THEN 500
2120 IF A=3 THEN END
2130 IF A>3 OR A<1 THEN PRINT 'OUTSIDE OF LEGAL RANGE - PLEASE TRY AGAIN ': FOR I=1 TO 450: NEXT I: GOTO 2050
10000 END
```

FONT COMPRESSOR PROGRAM

```
  0 REM CHARACTER COMPRESSOR
  1 GOTO 2000: REM  INITS
100 REM  GET THE NEXT CHARACTER
105 CALL HOME
110 FOR I=I0 TO I7
120 OLD(I)= PEEK (OLDBASE+(CHAR*I8)+I): TEMP=OLD(I)
126 FOR J=I0 TO I7: REM GET BINARY IMAGE IN REVERSE
127 F=OLD(I) MOD I2: BW(J)=F
128 OLD(I)=OLD(I)/I2: NEXT J: PRINT
129 OLD(I)=TEMP: IF OLD(I)<I32 THEN 130: POKE I0,OLD(I): CALL I2:OLD(I)= PEEK (I1)
130 FOR J=I7 TO I0 STEP I1N: IF BW(J)=I1 THEN PRINT "O";: IF BW(J)=0 THEN PRINT ".";: NEXT J
133 NEXT I: PRINT
140 RETURN
199 REM COLLECT AND MOVE COMPRESSED CHARACTER
200 REM TOTAL
201 VTAB I1: PRINT "CHARACTER ";CHAR;"  COMPRESSION COUNT = ";CT+I1
202 Z=0: FOR X=0 TO CT: REM  DISPLAY STORAGE VERSION
205 VTAB Z+2: TAB 15
210 PRINT CMPR(X): Z=Z+ZZ(X)
220 NEXT X: VTAB 15
225 FOR N=0 TO CT: ZZ(N)=0: NEXT N
230 IF NO=I1 THEN INPUT "OK?",A$: IF A$="N" THEN END : REM  LAST REVIEW LINE
240 FOR X=I0 TO CT: REM  MOVE COMPRESSED CHAR
250 POKE NBASE+NPTR,CMPR(X)
255 NPTR=NPTR+I1
260 NEXT X
270 IF NO=0 THEN PRINT "CHARACTER ";CHAR;" AT ";NBASE+NPTR;" - ";CT+I1;" BYTES"
271 IF NO=I1 THEN 272: FOR AA=I1 TO 150: NEXT AA
272 CHAR=CHAR+I1
280 IF CHAR=E THEN GOTO FINISH: REM 20000
285 BYTE=0: ONEUP=BYTE+1: CT=0: RC=0: FOR I=I0 TO I7: CMPR(I)=I0: NEXT I
290 GOTO 1000: REM GET THE NEXT CHAR
399 REM COLLECT NEXT BYTE ROUTINE
400 REM FILL CMPR ARRAY WITH COMPRESSED CHARS
401 CMPR(CT)=OLD(BYTE)+O(RC): REM THE COMPRESSION STROKE!
410 CT=CT+I1: ZZ(CT-I1)=RC+I1: RC=0: FLAG=0
420 BYTE=BYTE+1: ONEUP=BYTE+1
430 IF BYTE=7 THEN FLAG=2
440 IF FLAG#I2 THEN GOTO 1010
445 REM LAST BYTE DIFFERENTLY
450 CMPR(CT)=OLD(BYTE)+O(I0): GOTO TOT: REM
999 REM MAIN PROGRAM ROUTINE
1000 GOSUB GETCHAR: REM  MAIN TEST - 128 TIMES FOR FULL CHARACTER SET
1010 IF OLD(BYTE)#OLD(ONEUP) THEN GOTO COLLECT: REM  400
1015 RC=RC+I1
1020 REM GOT A MATCH
1025 IF BYTE#6 THEN 1030: REM WHEN BYTE= 6 THEN LAST TWO ARE IDENTICAL
1026 CMPR(CT)=OLD(BYTE)+O(RC): GOTO TOT
1030 BYTE=BYTE+I1: ONEUP=BYTE+I1: IF BYTE=I8 THEN GOTO TOT: REM  200
1050 IF ONEUP<8 THEN GOTO 1010
1060 GOTO TOT: REM  200 SHOW RESULTS
2000 REM INITS
2001 DSP COUNT
2010 CHR$="X": I=0: I0=0: I7=7: I8=8: I2=2: I15=15: I1=1: HOME=-936: I1N=-1: I32=32
2011 POKE I2,169: POKE 3,31: POKE 4,37: POKE 5,0: POKE 6,133: POKE 7,1: POKE 8,96: REM MACH LANG ROUTINE
2019 REM SET ARRAY FOR TOP THREE BITS
2020 DIM O(7): O(0)=0: O(1)=32: O(2)=64
2021 O(3)=96: O(4)=128: O(5)=160: O(6)=192: O(7)=224
2024 REM  VARIOUS ARRAYS
```

```
2025 DIM A$(39),B$(39),C$(39),D$(39),E$(39)
2026 DIM BW(7),CMPR(7),ZZ(7),OLD(7)
2028 FOR I=I0 TO I7:BW(I)=I0:ZZ(I)=I0:CMPR(I)=I0:OLD(I)=I0: NEXT I
2040 REM CREDITS
2041 CALL HOME: GOSUB 30000
2048 REM  POINTERS
2050 OLDBASE=2048: REM  BASE OF OLD CHAR
2051 NBASE=16384: REM  BASE OF NEW CCHAR
2055 FOR I=0 TO 1024: POKE OLDBASE+I,I0: POKE NBASE+I,I0: NEXT I: REM BLANK CHAR AREAS
2066 NPTR=0: REM  POINTER INTO NEW OUTPUT FILE
2069 CHAR=0: REM  POINTER TO PRESENT CHAR
2070 CT=I0: REM  NUMBER OF BYTES IN NEW CHAR
2071 RC=I0: REM  RC= REPETITION COUNT
2072 BYTE=I0: REM  POINTER IN "OLD" ARRAY
2074 ONEUP=I1: REM  COMPARISON POINTER
2075 REM SUBROUTINE POINTERS
2076 GETCHAR=100:TOT=200:COLLECT=400:FINISH=20000
2078 NO=0: REM  DEFAULT= NO REVIEW
2079 FLAG=0: REM FIND OUT WHAT TO DO
2080 PRINT "WHAT CHARACTER SET SHALL I": PRINT "COMPRESS? (ENTER 'CAT' FOR CATALOG) ": INPUT A$
2082 IF A$#"CAT" THEN 2086: PRINT "CATALOG":PRINT:INPUT"(PRESS RETURN TO CONTINUE",A$:GOTO2080:
2086 INPUT "STOP TO REVIEW EACH CHARACTER ?",B$: IF B$="Y" THEN NO=1:B$=""
2087 PRINT "NUMBER OF CHARACTERS TO MOVE?": INPUT "DEFAULT (0) = 128",E: IF E=0 THEN E=128
2089 PRINT "BLOAD",A$
2099 GOTO 1000
19999 REM  LOG TO DISK
20000 INPUT "SAVE COMPRESSED FILE TO DISKETTE? ",A$: IF A$#"Y" THEN 20030
20010 INPUT "NAME OF FILE? ",A$
20020 PRINT "BSAVE";A$;".COMP,A16384,L";NPTR: REM  VARIABLE NUMBER OF CHARACTERS
20025 PRINT "CREATED FILE: ";A$;".COMP"
20030 PRINT "CATALOG": END
30000 REM CREDIT SCREEN
30001 CALL HOME
30002 A$="SUP'R'TERMINAL FONT COMPRESSOR"
30004 B$="BY MORGAN P. CAFFREY"
30010 C$="P.O. BOX 31324"
30011 D$="SAN FRANCISCO, CA. 94131":E$="JANUARY 22, 1980"
30020 VTAB 10: TAB (39- LEN(A$))/2: PRINT A$: PRINT : PRINT
30022 TAB (39- LEN(B$))/2: PRINT B$: TAB (39- LEN(C$))/2: PRINT C$: TAB (39- LEN(D$))/2: PRINT D$
30023 TAB (39- LEN(E$))/2: PRINT E$
30025 PRINT : PRINT : PRINT "CLEARING CHARACTER SET SPACE"
30040 RETURN
```

APPENDIX A - APPLE ASCII CODES

The normal range of American Standard Code for Information Interchange (ASCII) codes is from 0 to 127. The APPLE ASCII is what is known as 'negative ASCII'. Negative ASCII begins at 128 and goes to 255. The high bit of each character is set and the numbers are considered to be negative. This slight difference in representation has no effect on the meaning. 128 characters are coded into numerical representations.

The following table gives the normal and APPLE ASCII code in decimal and hexidecimal representation and the character represented. It also provides a quick guide to APPLE & SUP'R'TERMINAL meanings.

NOTE: The control characters from 0 to 31 have character names acquired from their original use in telecommunication systems.

ASCII DEC	ASCII HEX	APPLE ASCII DEC	APPLE ASCII HEX	CHAR	TYPE:	SUP'R'TERMINAL MEANING
0	$00	128	$80	NULL	CTRL@	HOME & CLEAR
1	$01	129	$81	SOH	CTRL-A	UPPER-LOWER CASE "SWITCH"
2	$02	130	$82	STX	CTRL-B	ENTER BASIC
3	$03	131	$83	ETX	CTRL-C	STOP BASIC PROGRAM
4	$04	132	$84	ET	CTRL-D	D.O.S FLAG
5	$05	133	$85	ENQ	CTRL-E	UNASSIGNED
6	$06	134	$86	ACK	CTRL-F	UNASSIGNED
7	$07	135	$87	BEL	CTRL-G	BELL
8	$08	136	$88	BS	CTRL-H	BACKSPACE
9	$09	137	$89	HT	CTRL-I	TAB MULTIPLE-OF-8 COLUMNS
10	$0A	138	$8A	LF	CTRL-J	LINEFEED
11	$0B	139	$8B	VT	CTRL-K	LEFT BRACKET
12	$0C	140	$8C	FF	CTRL-L	HOME & CLEAR
13	$0D	141	$8D	CR	CTRL-M	RETURN
14	$0E	142	$8E	SO	CTRL-N	
15	$0F	143	$8F	SI	CTRL-O	
16	$10	144	$90	DLE	CTRL-P	
17	$11	145	$91	DC1	CTRL-Q	
18	$12	146	$92	DC2	CTRL-R	
19	$13	147	$93	DC3	CTRL-S	STOPLIST
20	$14	148	$94	DC4	CTRL-T	SPECIAL FUNCTION KEY
21	$15	149	$95	NAK	CTRL-U	RIGHT ARROW
22	$16	150	$96	SYN	CTRL-V	SHIFT-KEY MOD. SWITCH
23	$17	151	$97	ETB	CTRL-W	
24	$18	152	$98	CAN	CTRL-X	CANCEL LINE
25	$19	153	$99	EM	CTRL-Y	HOME CURSOR
26	$1A	154	$9A	SUB	CTRL-Z	
27	$1B	155	$9B	ESCAPE	ESC	
28	$1C	156	$9C	FS	*	
29	$1D	157	$9D	GS	CTRL-]? (SHIFT-M)	
30	$1E	158	$9E	RS	CTRL-^ (SHIFT-N)	
31	$1F	159	$9F	US	*	
						CTRL-T INTERPRETIVE MEANING
32	$20	160	$A0	space	space	0
33	$21	161	$A1	!	!	1
34	$22	162	$A2	"	"	2
35	$23	163	$A3			3
36	$24	164	$A4	t$	t$	4
37	$25	165	$A5	%	%	5
38	$26	166	$A6	&	&	6
39	$27	167	$A7	'	'	7
40	$28	168	$A8	(	(	8
41	$29	169	$A9	)	)	9
42	$2A	170	$AA	*	*	10
43	$2B	171	$AB	+	+	11
44	$2C	172	$AC	,	,	12
45	$2D	173	$AD	-	-	13
46	$2E	174	$AE	.	.	14
47	$2F	175	$AF	/	/	15
48	$30	176	$B0	0	0	16
49	$31	177	$B1	1	1	17
50	$32	178	$B2	2	2	18
51	$33	179	$B3	3	3	19

Continued

ASCII		APPLE ASCII		CHAR	TYPE:	SUP'R'TERMINAL MEANING
52	$34	180	$B4	4	4	20
53	$35	181	$B5	5	5	21
54	$36	182	$B6	6	6	22
55	$37	183	$B7	7	7	23
56	$38	184	$B8	8	8	23
57	$39	185	$B9	9	9	24
58	$3A	186	$BA	:	:	25
59	$3B	187	$BB	;	;	26
60	$3C	188	$BC	<	<	27
61	$3D	189	$BD	=	=	28
62	$3E	190	$BE	>	>	29
63	$3F	191	$BF	?	?	30
64	$40	192	$C0	@	@	31
65	$41	193	$C1	A	A	32
66	$42	194	$C2	B	B	33
67	$43	195	$C3	C	C	34
68	$44	196	$C4	D	D	35
69	$45	197	$C5	E	E	36
70	$46	198	$C6	F	F	37
71	$47	199	$C7	G	G	38
72	$48	200	$C8	H	H	39
73	$49	201	$C9	I	I	40
74	$4A	202	$CA	J	J	41
75	$4B	203	$CB	K	K	42
76	$4C	204	$CC	L	L	43
77	$4D	205	$CD	M	M	44
78	$4E	206	$CE	N	N	45
79	$4F	207	$CF	O	O	46
80	$50	208	$D0	P	P	47
81	$51	209	$D1	Q	Q	48
82	$52	210	$D2	R	R	49
83	$53	211	$D3	S	S	50
84	$54	212	$D4	T	T	51
85	$55	213	$D5	U	U	52
86	$56	214	$D6	V	V	53
87	$57	215	$D7	W	W	54
88	$58	216	$D8	X	X	55
89	$59	217	$D9	Y	Y	56
90	$5A	218	$DA	Z	Z	57
91	$5B	219	$DB	[	[	58
92	$5C	220	$DC	\	\	59
93	$5D	221	$DD	]	]	60
94	$5E	222	$DE	^	^	61
95	$5F	223	$DF	_	_	62
96	$60	224	$E0	space	space	chr$(224) 63
97	$61	225	$E1	a	a	64
98	$62	226	$E2	b	b	65
99	$63	227	$E3	c	c	66
100	$64	228	$E4	d	d	67
101	$65	229	$E5	e	e	68
102	$66	230	$E6	f	f	69
103	$67	231	$E7	g	g	70
104	$68	231	$E8	h	h	71
105	$69	232	$E9	i	i	72
106	$6A	233	$EA	j	j	73
107	$6B	234	$EB	k	k	74
108	$6C	235	$EC	l	l	75
109	$6D	236	$ED	m	m	76
110	$6E	237	$EE	n	n	77
111	$6F	238	$EF	o	o	78
112	$70	239	$F0	p	p	79
113	$71	240	$F1	q	q	80 (last interpretable character)
114	$72	241	$F2	r	r	
115	$73	242	$F3	s	s	
116	$74	243	$F4	t	t	

Continued

ASCII		APPLE ASCII		CHAR	TYPE:	SUP'R'TERMINAL MEANING
117	$75	243	$F5	u	u	
118	$76	244	$F6	v	v	
119	$77	245	$F7	w	w	
120	$78	246	$F8	x	x	
121	$79	247	$F9	y	y	
122	$7A	248	$FA	z	z	
123	$7B	249	$FB	{	{	
124	$7C	250	$FC	|	|	
125	$7D	251	$FD	}	}	
126	$7E	252	$FE	~	~	
127	$7F	253	$FF	DEL	?	

Comput

BP 84 Centre Commerci
16, rue Linois
75740 PARIS CEDEX 1
TEL.(1) 575 76 78