RGB-SL7

RGB-SL7

PRODUCT DESCRIPTION

The RGB-SL7 faithfully reproduces in RGB monitors the video modes that your computer is capable of displaying in NTSC or composite monitors. The RGB output of the RGB-SL7 improves the video quality of your Apple computer by removing the extraneous colors that occur during transitions in LORES and in mix mode text, and allowing you to select the text color by setting switches on the board.

The RGB-SL7 generates the following video modes (in order of increasing resolution):

1. 40 column switch selectable color text.
2. 80 column switch selectable color text.
3. 16 color LORES with option of mixing mode 1.
4. 16 color LORES with option of mixing mode 2.
5. 16 color MERES with option of mixing mode 2.
6. 2 color 280X192 (monochrome HIRES) with option of mixing mode 1.
7. 6 color HIRES with option of mixing mode 1.
8. 6 color HIRES with option of mixing mode 2.
9. 6 color DOT HIRES with option of mixing mode 1.
10. 6 color DOT HIRES with option of mixing mode 2.
11. 2 color 560X192 (monochrome double HIRES) with option of mixing mode 2.
12. 16 color 140X192 (color double HIRES) with option of mixing mode 2.

Modes 2,4,5,8,10,11, & 12 are available in Apple //es with an 80 column card installed in the Auxiliary slot. Modes 11 and 12 are available only if the 80-column card has 64K of resident RAM.

CONTENTS

1. RGB-SL7
2. RGB-SL7 manual
3. Cable with micro-probes
4. Internal RGB cable kit

SYSTEM REQUIREMENTS

1. Apple][or //e Rev B
2. RGB monitor with Apple ///-compatible cable

NOTE: This product is incompatible with Apple //e REV A computers.

MADE IN U.S.A.

FCC ID: D2A62LSLOT-7

12340 Saratoga Sunnyvale Rd, Suite 1
Saratoga, CA 95070

video 7

64K RGB INTERFACE MANUAL

FOR THE APPLE //e COMPUTER

Video-7 Incorporated

Notice

VIDEO-7 INCORPORATED reserves the right to make improvements in the product described in this manual at any time and without notice.

Copyright Notice

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (RGB Monitor) certified to comply with the Class B limits may be attached to this equipment. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Radio and Television Interference

The equipment described in this manual generates radio-frequency energy. If it is not installed properly it may cause interference with radio and television reception. The RGB or NTSC monitor you purchase must comply with the limits for Class B computing devices in accordance with the specifications in Subpart J, Part 15, of FCC rules. These rules are designed to provide reasonable protection against such interference in a residential installation.
The cable connection between the computer and the monitor must be a shielded cable with the shield properly grounded. You can determine if your equipment is the cause of interference by turning it off. If the interference stops, it was probably caused by the computer or the monitor. To correct the problem try:

1. Turning the TV or radio antenna until the interference stops.

2. Moving the computer to one side or the other of the TV or radio.

3. Moving the computer farther away from the TV or radio.

4. Plugging the computer into an outlet that is on a different circuit breaker or fuse than the TV or radio.

5. Installing a rooftop antenna connected to your TV and radio with a coaxial cable.

6. Identifying the offending piece of hardware by selectively turning them off one at a time and checking for interference.

If necessary, you should consult your computer dealer for additional suggestions. You may find the booklet "How to Identify and Resolve Radio-TV Interference Problems" prepared by the Federal Communications Commission helpful. The booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402, stock number 004-000-00345-4.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

64K RGB INTERFACE MANUAL

CHAPTER 1

1.1 PRODUCT FEATURES

Video-7's 64K RGB Interface not only gives you 80-column text capability for word processing, but also gives you new video modes, an extra 64K bytes of user memory, and allows you to interface your Apple //e to an NTSC or RGB monitor, or both.

For NTSC monitors, the video generated is found in the video output jack on the back of the Apple //e. Video-7's 64K RGB Interface is 100% compatible with both of Apple's 80-column cards when used in this configuration.

For RGB monitors, the video output is found in a DB-15 connector (supplied), and is completely pin compatible with the RGB output found in Apple ///s, therefore, any RGB monitor that is Apple ///-compatible will now work with your Apple //e.

1.1.1 APPLE //e-COMPATIBLE VIDEO MODES

The following standard video modes will be available whether you are interfacing your Apple //e to an NTSC or RGB monitor.

1. 40-column text.
2. 80-column text.
3. 16 color LORES with option of mixing mode 1.
4. 16 color LORES with option of mixing mode 2.
5. 6 color HIRES with option of mixing mode 1.
6. 6 color HIRES with option of mixing mode 2.
7. MERES (80X48) in 16 colors with option of mixing 80-column text (Double Density LORES).
8. 560X192 in Black and White with option of mixing 80-column text (Monochrome Double Density HIRES).
9. 140X192 in 16 colors with option of mixing 80-column text (Color Double Density HIRES).

The RGB output of Video-7's 64K RGB Interface improves the video quality of the Apple //e by removing the extraneous colors that occur during color transitions in LORES, MERES and Color Double Density HIRES. The text color may be selected by setting two switches on the board (see section 1.1.4) and is always free of the annoying color "tinge-ing" problem present in NTSC monitors.

1.1.2 SOFTWARE COMPATIBILITY

The hardware of Video-7's 64K RGB Interface is designed such that when you power up your Apple //e the video modes 1 through 8 are always chosen and remain in effect until you decide to change to one of the new video modes. The latter feature, coupled with the fact that the Interface uses the resident Apple //e 80-column firmware, allows for complete software compatibility with all existing (Apple Writer //e, Quick File //e, etc.) and future Apple //e software.

1.1.3 NTSC (COMPOSITE) MONITORS

Only if your Apple //e is a Rev B or higher will you obtain video modes 7 through 9. These modes are referred to as the "Double Density Video Modes."

MERES may, of course, be displayed only with color composite monitors. The NTSC video output of modes 8 and 9 are indistinguishable by composite type monitors. Either mode will appear in color when displayed on color composite monitors or in two colors when displayed on monochrome composite monitors.

Due to the limited resolution of color composite monitors, the above video modes are unsatisfactory, and are not recommended.

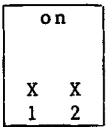
1.1.4 RGB MONITORS

With RGB monitors you will obtain excellent response for video modes 7 through 9 as well as the following new video modes:

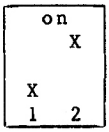
1. 40-column foreground background text in 16 colors.
2. LORES mixed with mode 1.
3. Foreground/background HIRES in 16 colors with limitations.
4. Foreground/background HIRES mixed with mode 1.
5. 160X192 in 16 colors.
6. 160X192 in 16 colors mixed with 80-column text.
7. Mix Mode (mix 140 and 560 anywhere on the screen).
8. Mix Mode mixed with 80-column text in bottom four lines.

Due to an RGB monitor's inherent higher resolution, all video modes available from the Video-7's 64K RGB Interface yield crisp and legible displays, and are enthusiastically recommended for word processing, as well as color graphic displays, or combinations thereof.

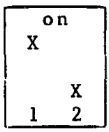
With RGB monitors Video-7's 64K RGB Interface is also able to simulate NTSC monitors of different color phosphorous. Highlighting of text (inverse mode) however, is always done in white to clearly accentuate the highlighted text. If you are accustomed to doing your word processing in either green, blue, amber, or white, you may select that text color by setting the two switches on the board as follows:



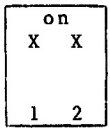
green 40 or 80-column text.



blue 40 or 80-column text.



amber 40 or 80-column text.



white 40 or 80-column text.

Text will now be displayed in the color of your choice. If at any future time you decide to change the text color, be sure to power down the computer before removing the Interface to change the switch positions.

1.2 INSTALLATION

Your Video-7 64K RGB Interface kit should include the following items:

1. The 64K RGB Interface enclosed in its antistatic bag.
2. A DB-15 cable kit assembly.
3. This manual.

If any of the above items are missing, please contact your dealer. To install the RGB Interface in your Apple //e, carry out steps 1 through 10

1. Remove the cover from your Apple //e by pulling up on the rear edges until the cover snaps off.

2. Touch the power supply cover (the big metal box to the left of the auxiliary slot) to discharge any static charge you may have accumulated on your clothes or body.

3. Make sure your Apple //e has been powered down. The red light at the left rear corner of the slots should be off.

4. Remove the 64K RGB Interface from its antistatic bag and set the text color switches to the color of your choice as explained in section 1.1.4.

5. IMPORTANT: ONLY IF YOUR APPLE //e IS A REV A COMPUTER -- CUT THE BOW-TIE AS SHOWN IN FIGURE 1.

6. Attach the DB-15 cable to the RGB Interface, as shown in Figure 2.

7. Install the 64K RGB Interface in the auxiliary slot of the Apple //e. Make sure that the components are to your right as you install the Interface, see Figure 3.

8. Attach the DB-15 connector to the back of the Apple //e as shown in Figure 4.

9. Replace the cover of your Apple //e by inserting the front tip of the cover and pushing down firmly on its back corners until you feel it snap into place.

10. Connect your RGB monitor cable to the DB-15 connector --MAKE SURE YOU FASTEN THE SCREWS SUPPLIED WITH THE MONITOR CABLE. Connect your NTSC monitor cable (if used) to the video output jack of the Apple //e.

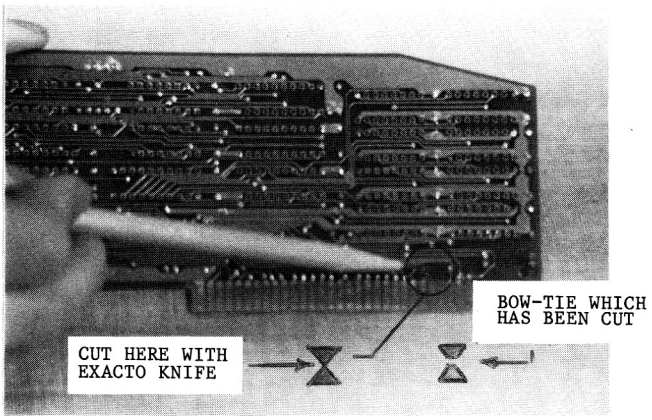


Figure 1. Cutting the bow-tie for Rev A Apple //es. The revision letter is found behind the slots. Rev A computers have the number 820-0064-A while Rev B computers have the number 820-0064-B.

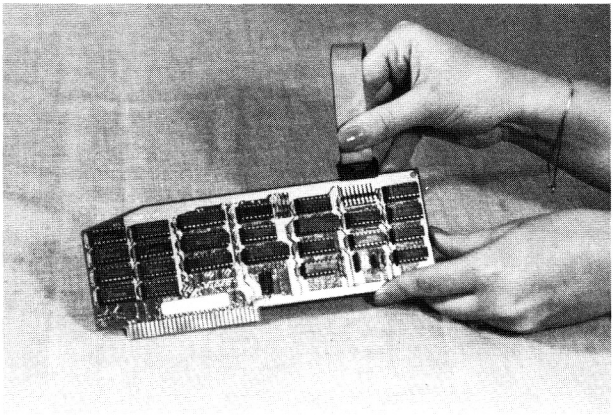


Figure 2. Installing the DB-15 cable to the 64K RGB Interface.

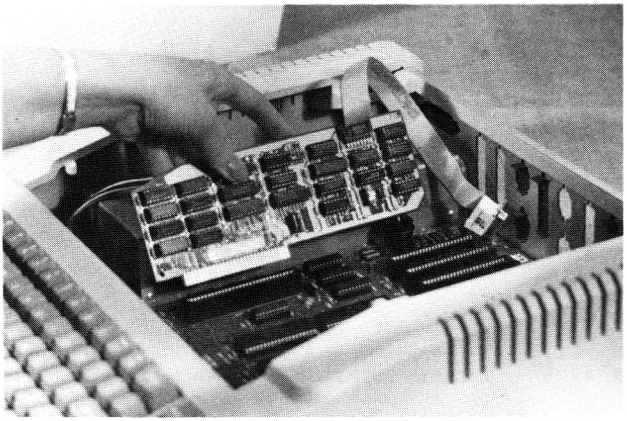


Figure 3. Installing the 64K RGB Interface in the Auxiliary slot.

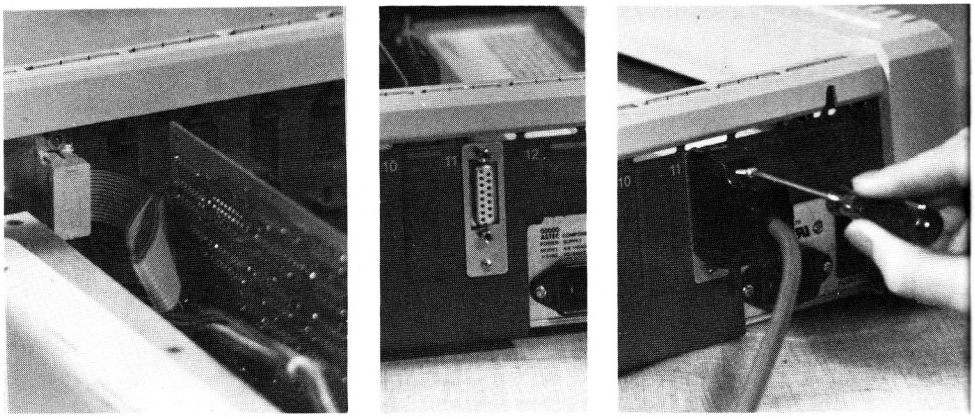


Figure 4. Attaching the DB-15 cable to the backplane of the Apple //e.

CHAPTER 2

2.1 INTRODUCTION

This chapter describes the firmware resident in the Apple //e that supports the 80-column feature of the machine. When the 80-column firmware is inactive the cursor is a checkerboard cursor. The firmware is activated automatically by Pascal or CP/M. From DOS either a PR#3 or IN#3 will activate the resident firmware. From the firmware monitor a 3 Cntl P or 3 Cntl K achieves the same results. When the 80-column firmware is active the cursor is a solid cursor.

Two sets of keyboard commands encompass all of the features found in the resident firmware, the Escape commands, and the Control commands as follows:

2.2 THE ESCAPE COMMANDS

As soon as you press the ESC key, the escape mode firmware is activated and the cursor changes to a "+" sign. Any subsequent key closure other than I,J,K,M or the left, right, up, and down arrow keys will deactivate the escape mode.

Esc @	Clears the video display and moves the cursor to its HOME position.
Esc A	Moves the cursor up one line.
Esc B	Moves the cursor right one space.
Esc C	Moves the cursor left one space.
Esc D	Moves the cursor down one line.
Esc E	Clears from cursor to the end of line.
Esc F	Clears from cursor to end of screen.
Esc I	Moves the cursor up one line.
Esc J	Moves the cursor left one space.
Esc K	Moves the cursor right one space.
Esc M	Moves the cursor down one line.
Esc R	Activates the uppercase-restrict mode. That is, independent from the setting of the caps lock or shift key, it forces all text entered to uppercase, except from within parenthesis, where the text entered is dependent on the caps lock or shift key.
Esc T	Deactivates the uppercase-restrict mode.
Esc 4	Switches from an 80-column display to a 40-column display without deactivating the 80-column firmware.
Esc 8	Switches from a 40-column display to an 80-column display only if the 80-column firmware was active.
Esc Cntl Q	Deactivates the 80-column firmware.

NOTES:

1. Esc A signifies pressing the escape key followed by the A key.
2. Esc Cntl Q signifies pressing the escape key followed by pressing the control key and Q key simultaneously.
3. Further cursor control may be obtained through the left, right and down arrows. The up arrow has to be activated through the Esc up arrow sequence to obtain cursor control.
4. The 80-column firmware may also be deactivated by the Cntl reset or Cntl open Apple reset sequence.

2.3 THE CONTROL COMMANDS

The ASCII decimal code is in parentheses.

Cntl G (7)	Generates a 1000Hz tone for 0.1 second.
Cntl H (8)	Moves cursor one space to the left.
Cntl J (10)	Moves cursor one line down.
Cntl K (11)	Clears from cursor to end of screen.
Cntl L (12)	Homes cursor and clears screen.
Cntl M (13)	Moves cursor to left-most end of next line.
Cntl S (19)	Stops sending characters to the display until another key is pressed.
Cntl V (22)	Scrolls the display down one line, leaving cursor in current position.
Cntl W (23)	Scrolls the display up one line, leaving cursor in current position.
Cntl Y (25)	Homes cursor but does not clear screen.
Cntl Z (26)	Clears the cursor line.
Cntl \ (28)	Moves cursor one space to the right.
Cntl] (29)	Clears from cursor to end of line.

NOTE: Cntl G signifies pressing the control key and the G key together.

CHAPTER 3

3.1 THE 80-COLUMN FIRMWARE AND PASCAL OR CP/M

As was mentioned in Chapter 2, both Pascal and CP/M automatically sense the presence of Video-7's 64K RGB Interface and activate the 80-column firmware resident in the Apple //e. In Pascal, the SETUP program should be run to make the up and down arrows functional. Cursor positioning in Pascal is further supported by the Cntl ^ x,y sequence which uses X=x-32 and Y=y-32 to move the cursor to a horizontal position given by X and a vertical position given by Y. For further information consult the appropiate manual for the particular operating system.

3.2 THE 80-COLUMN FIRMWARE AND DOS

Also mentioned in Chapter 2, the 64K RGB Interface firmware is activated in DOS by performing a PR#3. The firmware can be made to automatically activate by inserting the following Applesoft statement somewhere in your Hello program:

```
PRINT CHR$(4);"PR#3"
```

CHR$(4) is the ASCII decimal code for Cntl D and informs DOS that the print statement is an operating system directive. In a similar fashion, any of the control commands listed in the following table may be used in a print statement to control the 80-column firmware. The ASCII decimal code is in parentheses.

Cntl N (14)	Sets video output to normal; i.e., white on black.
Cntl O (15)	Sets video output to inverse; i.e., black on white.
Cntl Q (17)	Sets display to 40-columns.
Cntl R (18)	Sets display to 80-columns.
Cntl U (21)	Deactivates 80-column firmware, homes cursor and clears the screen.

NOTES:

1. Cntl N signifies pressing the control key and the N key simultaneously.
2. These control commands work only if the 80-column firmware has been activated.
3. For RGB monitors normal text color is switch-selectable to colors other than white.

As an example: boot DOS, type in, then run the following Applesoft program:

```
      Statement                        Comments (do not type in)

  10  PRINT CHR$(4);"PR#3"             Activate 80-col firmware
  20  HOME:PRINT"This is 80"
  30  PRINT CHR$(17)                   Set display to 40
  40  PRINT"THIS IS 40"
  50  PRINT CHR$(18)                   Set display to 80
  60  FOR I=0 TO 100
  70  NEXT I:GOTO 20
```

3.2.1 TABBING

The 80-column firmware is completely compatible with BASIC's VTAB command. HTAB, however, wraps around to the next line if it reaches the 41st character. The HTAB problem may be bypassed by using the POKE command, as follows:

POKE 36,[y] where y is an integer between 1 and 255 that indicates the horizontal tab position with wrap around at every multiple of 80.

3.2.2 COMMA AND SEMICOLON TABBING

The 80-column firmware is completely compatible with the semicolon method of tabbing from within the BASIC print statements. The comma method, however, gives unpredictable results and should not be used. The comma method problem may be bypassed by inserting the appropiate number of blanks in the print statement.

3.2.3 ALTERING THE SCREEN SIZE

With the 80-column firmware active the horizontal width of the screen may be changed with the following BASIC command:

POKE 33,[y] where y is any integer number up to 80.

CHAPTER 4

4.1 INTRODUCTION

This chapter describes the demonstration diskette and its double density software drivers. The double density drivers are a powerful software tool to allow you to manipulate the Apple //e's double density graphics.

4.2 THE DEMONSTRATION DISKETTE

Upon booting the demonstration diskette your computer should display the following menu:

1. Business Demo
2. Word Processing
3. Art Show Case
4. Slide Show
5. Video Modes
6. Color Switches
7. Color Mouse Draw(tm)
8. Boot F/B HIRES
9. Quit

Run through the demonstration to become fully acquainted with the RGB Interface and its many capabilities.

Option 7 will allow you to make drawings using your Apple //e mouse in the 16 color 140X192 video mode. The software allows you to clear the screen to a specified color, draw lines, rectangles, and color fill portions of or the whole screen. You may also save the file to disk and retrieve it at a later time to modify your drawing. To obtain instructions on how to operate this software package move the cursor to the square containing a question mark (?) and press the mouse button.

Option 8 will set the Interface to the Foreground/Background HIRES mode and allow you to select a foreground as well as a background color. You may then run programs in regular HIRES that make use of intensive text. The text will now be completely legible--free of any color "tinge-ing".

You may also load some of the Applesoft programs found in the diskette and list them on your printer to see how the Applesoft double density drivers were used.

4.3 THE APPLESOFT DOUBLE DENSITY DRIVERS

The Applesoft BASIC double density drivers are implemented as a series of ampersand routines. This allows considerable flexibility in the command formats and gives you a very easy-to-use interface through Applesoft. The routines described below are implemented as machine language code segments located in main and auxiliary memory. The commands are activated by BRUN-ing the binary code file from an Applesoft Basic program as follows:

```
PRINT CHR$(4);"BRUN HIRES"
```

The double density drivers may also be activated directly from the keyboard by typing:

```
BRUN HIRES
```

When either of the above commands is executed, the code file is loaded into the system, and the new commands are installed into memory. The first command that must be given is the & GR command in order to initialize the desired video mode.

The double density drivers contained in the demonstration diskette operate under DOS only. ProDOS double density drivers may be purchased from:

Video-7 Incorporated
12340 Saratoga Sunnyvale Rd. Suite 1
Saratoga, California 95070
(408) 725-1433

4.3.1 & GR X; Where X may range from 0 to 6

This is the initial entry point for the graphic routines. When this command is executed, the current pen position is moved to location 0,0 (the top left corner of the screen), the pen color is set to white and the background or fill color is set to black.

This command must be followed by a mode specification from the following table:

0	Disconnect Drivers
1	Two-color 560X192
2	Sixteen-color 140X192
3	Mix Mode
4	Sixteen-color 160X192
5	Foreground/Background HIRES
6	MERES

If the mode specification is given as zero, i.e., & GR 0, the graphic routines are disconnected and the space allocated to them is returned to the system.

A positive mode specification causes the RGB Interface to display data according to the video mode given in the above table and to load the corresponding driver routine for that video mode. Mode 3 is defaulted with the drivers for mode 1. A negative mode specification will load the driver for that particular video mode without changing the video state of the RGB Interface. This is particularly useful when dealing with video mode 3 which allows you to mix, anywhere on the screen, modes 1 and 2. Executing the command & GR 3 causes the RGB Interface to display data as in mode 3 and to load the driver for mode 1. Executing the command & GR -2 will activate the mode 2 double density driver but not change the video mode. Any further commands will now be interpreted and transfered to the screen as in mode 2.

To paint portions of the screen in mode 1 you must issue the & GR -1 or & GR 3 command before you perform any further plotting.

4.4 THE GRAPHIC COMMANDS

Nine commands that support graphic generation have been created. The commands allow for plotting dots or lines and color filling portions of the screen or the whole screen.

4.4.1 & TEXT

This command switches the display back to the text screen and also ensures that the DOS print hooks are correctly set. You should always execute this command prior to terminating a program which uses any of the graphic commands.

4.4.2 & COL = X; Where X may be a variable or expression

This command specifies the pen color to be used for drawing, for example:

```
& COL = 2
```

The color number should be in the range 0 to 15, and specifies the following colors:

0	Black	4	Dark Green	8	Brown	12	Green
1	Magenta	5	Grey 1	9	Orange	13	Yellow
2	Dark Blue	6	Medium Blue	10	Grey 2	14	Aquamarine
3	Purple	7	Light Blue	11	Pink	15	White

All modes default to white as the plot color.

4.4.3 & BCOL = X; Where X may be a variable or an expression

This command specifies the background or fill color. This color is used for the & CLEAR and & VFILL routines and also for the background color when displaying characters or bit mapped shapes. The command is used as follows:

```
& BCOL = 1
& BCOL = X+1
```

The color specification is the same as for the & COL command. The default color for all modes is black.

4.4.4 & CLEAR

This command clears the whole screen to the color specified by the last & BCOL command (or black if no & BCOL command has been given since the last & GR command). It has the same effect as an & VFILL command which specifies the whole drawing area, except that it executes much faster. The command has no parameters.

4.4.5 & VFILL (X1,X2,Y1,Y2); Where Xi and Yi may be variables or expressions

This command clears a portion of the screen to the background color as specified in the last & BCOL command (or black if no & BCOL command has been given since the last & GR command). Four parameters are required to specify the screen bounds to be filled, for example:

```
& VFILL (10,30,20,100)
& VFILL (XL,XL+20,YL,YL-12)
```

The first two parameters specify the horizontal pixel coordinates, while the second specify the vertical coordinates. The values in each pair may be specified in any order.

4.4.6 & MOVE (X,Y); Where X and Y may be variables or expressions.

This command moves the current pen position to pixel coordinate (X,Y) without drawing on the screen. The command requires two coordinates, for example:

```
& MOVE (20,50)
```

The first parameter is the new horizontal position in terms of pixels counting from the left edge of the screen, and the second is the new vertical location of the pixel counting from the top of the screen.

4.4.7 & DOT

This command places a dot in the foreground color at the current pen position. No parameters are required for this command.

4.4.8 & PLOT (X,Y); Where X and Y may be variables or expressions.

This command draws a line in the foreground color from the current pen position to a new pen position specified as parameters to this statement. The following statements will, for example, draw a brown line across the top of the screen in mode 2:

```
& PRINT CHR$(4);"BRUN HIRES"
& GR 2
& MOVE (0,0)
& COL = 8
& PLOT (139,0)
END
```

4.4.9 & SCRN (X); Where X may be an integer or real variable.

This command returns a value to the variable X representing the color at the current pen position. One parameter is required to accept the returned value, for example:

```
& SCRN (VA)
```

In this case the returned value will be placed in the destination variable VA. The value returned is in the range 0 to 15 and represents a color as specified in the table of section 4.4.2

4.5 THE DISK COMMANDS

The Disk Commands allow for storage and retrieval of double density graphics files generated using the Applesoft double density drivers.

4.5.1 & SAVE FN$

This command saves the double density graphic area in main and auxiliary memory to a specified disk file, for example:

```
& SAVE "TEST PICTURE"
```

The disk file name may be specified as a literal or in a string variable. Slot, drive and volume parameters may be specified as described in the Apple //e DOS Users Guide. The file will appear to be a binary file in the disk directory, but is not stored in the standard binary format and can only be loaded using the & LOAD command described in section 4.5.2.

4.5.2 & LOAD FN$

This command loads a file from disk which has been saved using the & SAVE command. A filename must be specified in the same manner as in the & SAVE command, for example:

```
& LOAD "PIE.PIC"
& LOAD "MY GRAPH,S6,D2"
```

4.6 THE GRAPHIC TEXT AND SHAPE COMMANDS

The Applesoft double density drivers contain commands to place text as well as predefined shapes anywhere on the screen.

4.6.1 &GPRNT

This command replaces the standard screen output hooks with a pointer to a routine in the graphics package which replaces text on the graphics screen. The default character font is very similar to that used for standard text output by the Apple //e. After this command has been issued, all output via the standard Applesoft PRINT statements will be directed to the graphics screen until a command such as &TPRNT is given which restores the Apple //e's output hooks. While it is active the routine disconnects the Disk Operating System. All characters are printed in a matrix of 7 horizontal and 8 vertical pixels (the same as the standard text screen) and all characters are displayed downwards and to the right of the current pen position. After each character is written, the pen position is moved to the right seven pixels.

The following command sequence illustrates how characters may be placed on the screen:

```
& GR 2                  Initialize graphics mode 2
& BCOL = 2              Set fill color to dark blue
& CLEAR                 Clear the screen to dark blue
& COL = 13              Set pen color to yellow
& MOVE (30,96)          Move pen to center of screen
& GPRNT                 Redirect output to graphic screen
PRINT "Some text"       Print text on the graphic screen
& TPRNT                 Restore normal print output
```

Unless the command & CPRNT 1 has been executed (see section 4.6.2) all control characters are ignored except for the carriage return (ASCII code 13) and line feed (ASCII code 10) characters.

The line feed characters cause the current pen position to be moved down the screen by 8 pixels, the equivalent of the character row size on the text screen. If the downward movement results in an attempt to move off the bottom of the screen, the pen wraps around to the top to the screen. The carriage control character moves the pen position horizontally to the left edge of the screen and moves the pen vertically downward as described for the linefeed character. This is the exact equivalent of a carriage return in text mode.

4.6.2 & CPRNT X; Where X may be 0 or 1

In some circumstances the user may not wish control characters to be ignored, for example, extra graphic characters which may be defined in this area. The & CPRNT command governs the control character trap in the graphic print routines, and allows the user to output control characters. The routine requires one parameter, for example:

```
& CPRNT 1
& CPRNT 0
```

If the parameter is zero, control characters will not be printed; if it has a value of one it will enable printing of a control character. When the drivers are first activated an implied & CPRNT 0 is executed.

It is the responsibility of the user to ensure that, before executing an & CPRNT 1 command suitable characters are defined in the font that were loaded by the & NCHARS command explained in section 4.6.4. The default font has definitions for all control characters.

4.6.3 & TPRNT

This routine restores the print hook to normal when text is no longer to be sent to the graphics screen. It has no effect if the & GPRNT command is not active. The & TEXT command contains an implicit & TPRNT call.

4.6.4 & NCHARS F$

The & NCHARS and & SCHARS commands are used to change the font used for displaying text on the graphics display. The & NCHARS command changes to a user-defined font; & SCHARS changes back to the standard font.

The & NCHARS command requires one parameter which specifies the disk file containing the new font. For example:

```
& NCHARS "ROMAN.FONT"
& NCHARS "BYTE.FONT,S6,D2,V12"
& NCHARS F$
```

The system font is automatically loaded when the HIRES file is activated as described in section 4.3. Appendix B describes the character cell format that must be used when generating font tables.

4.6.5 & SCHARS

This command changes the font used for displaying text on the graphics screen back to the system standard font.

4.6.6 & DRAW (ba,br,cd,rd,bw,bh); Parameters may be variable or expressions

This procedure draws a predefined shape on the screen. It performs a bit-map transfer to the screen of a specified portion of a block of memory. The shape is placed below and to the right of the current pen position, which is not changed by this command. Each bit set to a "one" in the source block is put on the screen using the current pen color and each "zero" bit uses the current fill color. Six parameters are required for this procedure.

Block Address: The first parameter is an address pointer to the first element of the source block in memory.

Bytes per row: The second parameter specifies the number of bytes in each row of the source block.

Column Bit Displacement: The third parameter specifies the number of bits to skip in each source row before beginning the transfer.

Row Displacement: The fourth parameter specifies the number of source rows to skip before beginning the transfer process.

Width of Block: The fifth parameter specifies the bit width of the block portion to be transferred to the screen.

Height of Block: The sixth parameter specifies the total number of rows of the block portion to be transferred to the screen.

For example:

```
& DRAW (768, 3, 0, 0, 24, 8)
```

This would specify that the bit array starting at memory locations 768 ($0300 hexadecimal) is to be used as a block source, that each row is 3 bytes long, that drawing should commence at the start of the array and that the area to be drawn is 24 bits wide and 8 rows in height. Appendix C gives a detailed example of a shape block definition.

5.0 INTRODUCTION

This chapter describes the Applesoft 16-color foreground/background text mode (console) driver. The console driver contained in the demonstration diskette operates under DOS only. ProDOS console drivers may be purchased from:

Video-7 Incorporated
12340 Saratoga Sunnyvale Rd. Suite 1
Saratoga, California 95070
(408) 725-1433

The ProDOS diskette contains both the Double Density (explained in Chapter 4) and the Console Drivers.

5.1 WHAT THE CONSOLE DRIVER DOES

The Video-7's 64K RGB Interface can display a text image on its video display in three modes:

1. 24 lines of 40 characters per line, monochrome only.
2. 24 lines of 40 characters per line, colored characters on colored backgrounds, 16 colors available.
3. 24 lines of 80 characters per line, monochrome only.

The routines supplied in the Apple //e ROM are available only for the two monochrome modes. In either of these two modes, the Video-7 RGB Interface will operate in exactly the same manner as the standard Apple display.

The standard Apple //e does not have the capability of handling text displays in color and consequently supplies no routines to support this mode. The Video-7 console drivers are designed to correct this problem when running under Applesoft.

When the screen is in the color text mode, each character position on the screen can display a character formed by dots (foreground) in one of 16 colors against a background of any other of the 15 colors.

The color text mode commands contained in the console drivers are activated by BRUN-ing the binary code file from an Applesoft Basic program as follows:

```
PRINT CHR$(4);"BRUN FB"
```

The console drivers may also be activated directly from the keyboard by typing:

BRUN FB

When either of the above commands is executed, the code file is loaded into the system, and the new commands are installed into memory. The console driver controls three aspects of the screen:

The viewport:

This is the rectangular area on the screen where the console driver places text characters. The console drivers place new text only in the viewport, so you may change the size and location of the viewport to protect data already placed on the screen. The default size of the viewport is the entire screen.

The cursor position:

This is the position in the viewport where the console drivers will place the next character output to the display. The cursor can also be moved around the screen.

The screen colors:

The color of the dots which form the characters (foreground) and the color of the background. A list of the parameter codes for each color is given in section 5.8 "Color Specifications".

5.2 SCREEN OUTPUT

The Video-7 console driver takes over the control of the console. All characters sent from the Apple //e Operating System or a program to the screen are passed through the console driver (there is an exception to this which is described in section 5.6 "Special Conditions").

If the character code is an ASCII control character (with a value of 31 or less), the console driver performs a special control function. These control characters and their functions are discussed in sections 5.3 and 5.4. Control characters are not displayed on the screen.

If the character code is a valid ASCII text character code (a value between 32 and 127), the console driver places the character in the viewport at the current cursor position.

After the character has been placed on the screen, the cursor normally moves one space to the right. If this would place the cursor outside the current viewport the console driver will place the cursor at the first position in the next line. If this line is below the bottom of the viewport, the viewport is scrolled up one line and the bottom line blanked before the cursor is placed at the start of this line. These automatic cursor movements can be controlled using special characters (see section 5.4).

When moving backwards, the cursor motion is analogous to its forward motion. If the cursor is moved past the left edge of the viewport, it is placed at the right edge of the viewport on the line above. If the cursor is moved past the top of the viewport, the contents of the viewport are scrolled down one line and the cursor is placed at the end of the newly blank top line.

5.3 SCREEN CONTROL CODES

The ASCII control characters with values between 0 and 31 are recognized by the console driver as screen control codes. When such a character is received during output, it is not placed on the screen; instead, it performs a control function as described in section 5.4 "Control Codes".

Some of the screen control codes must be followed immediately by one or more parameters. These parameters supply additional information relating to the function invoked by the control code. For example, setting a color for the foreground (code 19) requires one parameter which specifies the color to be used. For example:

```
10 PRINT CHR$(4);"BRUN FB"              Install drivers
20 SETFORE = 19: DARKBLUE = 2           Declare variables
30 PRINT CHR$(SETFORE)+CHR$(DARKBLUE); Set foreground color
40 SETBACK = 20: WHITE = 15             Declare variables
50 PRINT CHR$(SETBACK)+CHR$(WHITE);     Set background color
60 PRINT "Some Text"                    Print text
70 END
```

The print statement in line 60 will print the characters with a Dark Blue foreground on a White Background.

NOTE: It is recommended that ALL control codes be terminated by a semicolon (as shown on lines 30 and 40 above). If several control code instructions are to be included in one line, use the semicolon followed by a colon, technique.

5.4 CONTROL CODES

The following control codes have been implemented in the console driver. The control code values for keyboard, Basic and assembly language input are included in parentheses after the command name.

5.4.01 RESET VIEWPORT (Control-A, Decimal 01, Hex 01)

Resets the viewport to the size of the screen (40 horizontal characters by 24 vertical characters). The cursor is left in the same position on the screen, although its coordinates relative to the origin of the viewport may change.

5.4.02 CURSOR ON (Control-E, Decimal 05, Hex 05)

Makes the cursor visible. This is the default state. The cursor is never displayed during output, the console driver turns the cursor on whenever input is requested, and removes the cursor when the input request is complete.

5.4.03 CURSOR OFF (Control-F, Decimal 06, Hex 06)

Makes the cursor invisible. All input and output is still processed normally, except that the cursor is not displayed on the screen.

5.4.04 SOUND BELL (Control-G, Decimal 07, Hex 07)

Sounds a short beep on the Apple //e's built-in speaker.

5.4.05 MOVE CURSOR LEFT (Control-H, Decimal 08, Hex 08)

This is the backspace command. Moves the cursor position one space to the left. Screen wraparound and scrolling are performed unless inhibited by the Cursor Movement Control command (see below).

5.4.06 MOVE CURSOR RIGHT (Control-I, Decimal 09, Hex 09)

Moves the cursor one space to the right. Screen wraparound and scrolling are performed unless inhibited by the Cursor Movement Control command (see 5.4.17).

5.4.07 MOVE CURSOR DOWN (Control-J, Decimal 10, Hex 0A)

Moves the cursor down one line. Screen wraparound and scrolling are performed unless inhibited by the Cursor Movement Control command (see 5.4.17).

5.4.08 MOVE CURSOR UP (Control-K, Decimal 11, Hex 0B)

Moves the cursor up one line. Screen wraparound and scrolling are performed unless inhibited by the Cursor Movement Control command (see 5.4.17).

5.4.09 HOME CURSOR (Control-L, Decimal 12, Hex 0C)

Moves cursor to the upper left corner of the viewport. Does not clear any portion of the screen or change viewport setting.

5.4.10 RETURN CURSOR (Control-M, Decimal 13, Hex 0D)

Moves the cursor to the beginning of the next line. If specified by the Cursor Movement Control command this can be amended to move to the start of the current line by resetting the automatic line feed option.

5.4.11 VIEWPORT TOP (Control-N, Decimal 14, Hex 0E)

Sets the upper left corner of the viewport to the current cursor position. This position will now have the coordinates (0,0), and all cursor movements will be made relative to those coordinates.

5.4.12 VIEWPORT BOTTOM (Control-O, Decimal 15, Hex 0F)

Sets the lower right corner of the viewport to the current cursor position.

5.4.13 NORMAL (Control-Q, Decimal 17, Hex 11)

Specifies that all subsequent characters will be displayed as characters of the foreground color on a field of the backgrour color. Does not affect any characters already on the screen.

5.4.14 INVERSE (Control-R, Decimal 18, Hex 12)

Specifies that all subsequent characters will be displayed as characters of the background color on a field of the foreground color. The cursor character color will also be inverted. Does not affect any characters already placed on the screen.

5.4.15 FOREGROUND COLOR (Control-S, Decimal 19, Hex 13)

Sets the foreground color to the value specified by the next following parameter character. All subsequent characters will be displayed in the new color, but characters already on the screen will not be affected. The colors and their associated color values are described later in section 5.8 "Color Specifications".

As an example: PRINT CHR$(19)+CHR$(2) would set foreground to Dark Blue.

5.4.16 BACKGROUND COLOR (Control-T, Decimal 20, Hex 14)

Sets the background color to the value specified by the next following parameter character. All subsequent characters will use this color as a background, but characters already on the screen will not be affected. The colors and their associated color values are described in section 5.8 "Color Specifications".

As an example: PRINT CHR$(20)+CHR$(15) would set background to White.

5.4.17 CURSOR MOVEMENT (Control-U, Decimal 21, Hex 15)

Sets the controls that determine various movements of the cursor according to the value of the immediately following parameter character. Only the lower four bits of the parameter character are significant. The controls are Advance, Line Feed, Wrap-around, and Scroll.

When Advance is active, the cursor moves one space to the right after each character is placed on the screen; when inactive, the cursor stays at the same position. The default state is active.

When Line Feed is active, the cursor performs a line feed after each carriage return; when inactive, no automatic line feed is performed. The default state is active.

When Wrap-around is active, an attempt to move the cursor beyond the right or left edge of the viewport causes the cursor to be placed at the opposite edge of the next or previous line

respectively; when inactive, the cursor remains at the edge of the viewport. The default state is active.

When Scroll is active, an attempt to move the cursor beyond the top or bottom of the viewport causes the viewport to be scrolled down or up respectively, and the cursor is placed on the new top or bottom line; when inactive, the cursor remains at the top or bottom of the viewport. The default state is active.

The selection of which cursor control is to be activated is determined by the lower four bits of the parameter character as follows:

Control	Bit	Default
Advance	0	1
Line Feed	1	1
Wrap-around	2	1
Scroll	3	1

If a given bit has a value of 1, the associated control is active. As an example, PRINT CHR$(4)+CHR$(01) would only enable the Advance function. This is because the parameter character "01" contains a value of 1 in bit 0 and values of 0 in bits 1 through 3 (its binary representation).

5.4.18 SCREEN SYNCHRONIZATION (Control-V, Decimal 22, Hex 16)

Causes a delay until the video generator in the Apple //e has finished diplaying one complete frame on the screen. The video generator produces 60 frames a second, thus the delay can be up to 1/60 of a second.

5.4.19 ABSOLUTE POSITION (Control-Z, Decimal 26, Hex 1A)

Moves the cursor to the row and column specified in the two parameter characters immediately following this code. The first parameter is used to determine the new horizontal position; and the second parameter is used to determine the new vertical position. If the value of the first parameter is greater than the width of the viewport, the cursor is placed at the right of the viewport. If the value of the second parameter is greater than the height of the viewport, the cursor is placed on the bottom line of the viewport.

As an example: PRINT CHR$(26)+CHR$(2)+CHR$(5) would place the cursor on the second character position of the fifth line

5.4.20 CLEAR VIEWPORT (Control-\, Decimal 28, Hex 1C)

Moves the cursor to the upper left corner of the viewport and sets the contents to space characters.

5.4.21 CLEAR TO END OF VIEWPORT (Control-], Decimal 29, Hex 1D)

Clears to blanks from the current cursor position to the end of the viewport. The cursor is not moved.

5.4.22 CLEAR LINE (Control-^, Decimal 30, Hex 1E)

Clears the entire line the cursor is on and moves the cursor to the left edge of the viewport.

5.4.23 CLEAR TO END OF LINE (Control-_, Decimal 31, Hex 1F)

Clears from the current cursor position to the right edge of the viewport. The cursor is not moved.

5.5 COMPATIBILITY

The following standard Applesoft statements which control screen output are fully compatible with the console driver routines, and will have the expected effects if intermixed with the special console driver screen control codes in a program:

HTAB
VTAB
INVERSE
NORMAL

Refer to the Applesoft reference manual for a guide to the use of these commands.

5.6 SPECIAL CONDITIONS

When the console driver is initialized, the system colors are set as follows:

Foreground color - 15 (White)
Background color - 0 (Black)

When the console driver is initialized or whenever a system reset occurs, the following variables are set:

1. Cursor on.
2. All cursor motion controls on.
3. Viewport set to full screen.

The console driver can be deactivated by entering escape mode (pressing ESCAPE) and then pressing the Control and Q keys simultaneously. This will remove the console driver from the system and return the user to the standard 40-column monochrome mode.

There is a problem in the Apple //e monitor routines dealing with line oriented keyboard entry through the GETLN procedure. When the return key is depressed to indicate the end of data entry, the GETLN routine fills the rest of the line with spaces. Unfortunately, this is done within the monitor, and cannot be intercepted by the console driver routines. The effect is that the Apple //e monitor clears the rest of the line to spaces, but the color codes for these characters are not changed. This can cause some unpredictable effects in the characters that are overwritten.

5.7 KEYBOARD CONTROL CODES

The console driver controls input as well as output. In general, the console driver acts in exactly the same way as the standard Apple //e 80-column keyboard driver, except that a number of extra escape control codes have been added, and two escape control codes have not been implemented.

Escape mode is used for controlling the screen during data input. It is entered by pressing the ESCAPE key on the keyboard. The cursor will change from an inverse representation of the character under the cursor to an inverse plus sign. The cursor will remain in this form until escape mode is turned off by pressing any character which does not constitute a valid escape mode control. There are a number of valid escape controls. Each one corresponds directly to one of the control codes described in section 5.4.

The following escape control codes have been implemented:

Character	Escape Function	Control Code
@	Clear viewport and exit escape mode	28
A	Move cursor right and exit escape mode	09
B	Move cursor left and exit escape mode	08
C	Move cursor down and exit escape mode	10
D	Move cursor up and exit escape mode	11
E	Clear to end of line and exit escape mode	31
F	Clear to end of viewport and exit escape mode	29
H	Home cursor	12
I	Move cursor up	11
J	Move cursor left	08
K	Move cursor right	09
M	Move cursor down	10
R	Set viewport bottom-right	15
T	Set viewport top-left	14
V	Reset viewport to full screen	01
Left Arrow	Move cursor left	08
Right Arrow	Move cursor right	09
Up Arrow	Move cursor up	11
Down Arrow	Move cursor down	10

The following cursor escape code commands have not been implemented:

R	Set upper case entry restricted mode
T	Reset upper case restricted mode

5.8 COLOR SPECIFICATIONS

The following 16 colors can be displayed as background or foreground colors using the parameter characters indicated.

Black	0	Brown	8
Magenta	1	Orange	9
Dark Blue	2	Grey 2	10
Purple	3	Pink	11
Dark Green	4	Green	12
Grey 1	5	Yellow	13
Medium Blue	6	Aquamarine	14
Light Blue	7	White	15

5.9 SCREEN CONTROL CODES

The following table is a resume of all the screen control codes impplemented in the console driver.

Code	Hex	Parameters	Function
01	01		Reset viewport
05	05		Cursor on
06	06		Cursor off
07	07		Sound the bell
08	08		Move cursor left
09	09		Move cursor right
10	0A		Move cursor down
11	0B		Move cursor up
12	0C		Move cursor home
13	0D		Return cursor
14	0E		Set upper-left viewport
15	0F		Set lower-right of viewport
17	11		Set normal text
18	12		Set inverse text
19	13	color	Set foreground color
20	14	color	Set background color
21	15	move data	Cursor movement controls
22	16		Synchronize screen
26	1A	x, y	Absolute position
28	1C		Clear viewport
29	1D		Clear to end of viewport
30	1E		Clear line
31	1F		Clear to end of line

IMPORTANT NOTICE

The following chapters are highly technical in nature and are intended for those who have extensive knowledge of the Apple II or Apple //e hardware and also are familiar with 6502 machine language programming. They are not "required reading" for those who want only to use the card's 80-column capability or reap the benefits of the higher resolution acquired through the use of an RGB monitor with off-the-shelf programs or through the Applesoft Drivers supplied in the demonstration diskette.

6.1 THE APPLE //e VIDEO CONTROL

The Apple //e hardware configures the extension RAM on the 64K RGB Interface, as a mirror image (occupying the same address space) as the resident RAM on the Apple //e motherboard. Software-selectable switches bank-in the desired section of RAM. When in 40-column, all characters displayed are retrieved from resident RAM. When in 80-column all even characters are retrieved from extension RAM while the odd characters are retrieved from resident RAM. Four switches control all the different 80-column display modes, and are:

6.2 THE 80-COLUMN SWITCH

The 80-column switch doubles the horizontal resolution of the video screen. On power up and during a system reset the switch comes up reset, that is, inactive. To set the switch a microprocessor write to location $C00D (in HEX as denoted by the $ preceeding the number) must be performed. To reset the switch a microprocessor write to location $C00C must be performed. The status of the switch may be read from location $C01F.

6.3 THE 80 STORE AND PG2 SWITCHES

The 80 store switch instructs the hardware to access video RAM by using the PG2 switch as a pointer. If PG2 is set, the extension RAM found in the Video-7 64K Interface is selected and the resident RAM is deselected. If PG2 is reset the opposite condition is activated. The banking occurs always for the range $400 through $7FF, and when in HIRES it also occurs for the range $2000 through $3FFF. The switches come up inactive; i.e., reset, during power up and during a system reset. To set the 80 store switch, a microprocessor write to location $C001 must be performed. To reset the 80 store switch, a microprocessor write to location $C000 must be performed. The status of the 80 store switch may be read from location $C018.

To set the PG2 switch, a microprocessor read or write to location $C055 must be performed. To reset the PG2 switch, a microprocessor read or write to location $C054 must be performed. The status of the PG2 switch may be read from location $C01C.

6.4 THE AN3 SWITCH

In Rev B or later, Apple //es can generate double density graphics. Apple II-compatability in the Apple //e was achieved by always forcing graphics to be 40-columns wide, independent of the status of the 80-column switch. However, in Rev B or later computers only, the AN3 switch defeats this feature and allows the hardware to generate double density graphics. The switch comes up inactive, or set, during power-up or upon a system-reset.

To reset the switch a microprocessor read or write to location $C05E must be performed. To set the switch a microprocessor read or write to location $C05F must be performed. There is no way to read the status of the AN3 switch.

CHAPTER 7

7.1 THE APPLE //e DOUBLE DENSITY MODES AND NEW RGB VIDEO MODES

This chapter describes the new video modes generated by Video-7's 64K Interface and the Apple //e's Double Density video modes. To obtain any of the new modes the AN3 must be reset. Text mixing in the lower four lines of a graphics display will occur as follows: when in 40-columns the text mixed will be in foreground/background colors; when in 80-columns the lower four lines will be in hard-switch-selectable color on black background in normal video, and black on white in inverse video. The Apple //e and the RGB Interface's software switch settings for all different video modes are given in Appendix A. The text color switch settings are given in Chapter 1.

7.2 FOREGROUND/BACKGROUND TEXT (Available only in RGB)

In this mode the extension RAM in the 64K RGB Interface is used only for color information. The most significant nibble of each byte is used to generate any one of 16 foreground colors while the least significant nibble is used for the background color.
The 40-column text video information is retrieved from page one of resident RAM, that is, from locations $400 through $7FF. The color information is retrieved from extension RAM from location $400 through $7FF in a one-to-one correspondence with the video information.

The Apple //e must be in text mode with the 80-column video switches in the following state:

```
80-column switch: reset
80 store switch:  set
PG2 switch:       used as pointer:
                  when set, color information is stored;
                  when reset, video information is stored.
```

7.3 LORES MIXED WITH F/B TEXT (Available only in RGB)

In this mode the LORES graphics mode is mixed with 40-columns of foreground/background text. The bottom four lines of text follow the rules of Section 7.2.

7.4 16-COLOR MERES (NOT available in Rev A Apple //e)

In this video mode the screen is divided into 80X40 pixels, any one of which can be 16 different colors. The extension RAM is used to generate the color pixels in even columns, while the resident RAM is used to generate the color pixels in odd columns. The video mapping is identical to that of page one LORES, that is, from $400 through $7FF. The most significant nibble of each byte contains the color information for the even rows while the least significant nibble contains the color information for the odd rows. Text mixing in the lower four lines will be in 80-column format.

The Apple //e must be in the LORES video mode with the 80-column video switches in the following state:

80-column switch:	set
80 store switch:	set
PG2 switch:	used as pointer: when set, even horizontal pixels are accessed; when reset, the odd ones are accessed.

7.5 FOREGROUND/BACKGROUND HIRES (Available only in RGB)

This video mode is identical to the foreground/background text mode, with the exception that foreground color is determined by an "on" HIRES dot instead of an "on" text dot. The extension RAM is used to generate color information which is stored in $2000 through $3FFF giving 80X192 possible color combinations. The color-determining data (video) is stored in resident RAM at locations $2000 through $3FFF. A "one" chooses the foreground color (high nibble in extension RAM), while a "zero" chooses the background color (low nibble in extension RAM).

In mix mode, the text in the lower four lines will be foreground/background 40-column text, as described in section 7.2.

The Apple //e must be in HIRES with the 80-column video switches in the following state:

80-column switch:	reset
80 store switch:	set
PG2 switch:	used as pointer to store a) color information in extension RAM (set), or b) text and HIRES information in resident RAM (reset).

7.6 DOUBLE DENSITY HIRES MODES

The Double Density HIRES video modes are available only in REV B Apple //es. The 64K RGB Interface subdivides the Double Density HIRES video mode of the Apple //e into four different video modes. This subdivision is accomplished by two dedicated flags (Flg1 and Flg2) in the RGB Interface. The four video modes are:

7.6.1 560X192 IN BLACK AND WHITE

This video mode is the Double Density HIRES mode of the Apple //e as seen in a composite monochrome monitor.

This video mode is "bit-mapped", that is, any bit within the byte which is set to a logic "one" will generate a color pixel. The color displayed is white for bits set to a logic "one" and black for the bits that are set to a logic "zero". Only seven bits (bits zero through 6) of each byte are used to generate video, with the least significant bit of each byte being displayed first and bit 6 being displayed last. The even groups of seven bits are retreived from extension RAM, while the odd groups of seven bits are retreived from resident RAM. In mix mode, the text in the lower four lines will be 80 columns of switch-selectable color.

This mode is the default double HIRES video mode upon power-up. The software must go through the following sequence to select this video mode after any other double HIRES mode has been selected or if it is selecting the mode other than the first time via the power-up default (this sequence will set Flg1 and Flg2 to a logic "one"):

a) Set the HIRES flag
b) Reset the MIX flag
c) Reset the 80-column flag
d) Reset and then set AN3
e) Reset and then set AN3 once more
f) Reset AN3

The remainder of the Apple //e's flags must be in the following state:

80-column flag:	set
80 store flag:	set
PG2 flag:	used as a pointer: accesses odd bytes when set; accesses even bytes when reset.

7.6.2 140X192 IN SIXTEEN COLORS

This video mode is the Double Density HIRES mode of the Apple //e as seen in a composite color monitor.

This video mode separates the horizontal width into 140 locations any one of which can have 16 colors. Four bits of each byte are used to generate the 16 colors but only 7 bits out of each byte are actually used for display. Even bytes are retrieved from extension RAM while the odd bytes are retrieved from resident RAM. Table 1 shows how the first 10 pixels are retrieved from memory.

Table 1

Pixel	Bits	Byte	Extension	Resident
1	0-3	1	X	
2	4-6	1	X	
	0	1		X
3	1-4	1		X
4	5-6	1		X
	0-1	2	X	
5	2-5	2	X	
6	6	2	X	
	0-2	2		X
7	3-6	2		X
8	0-3	3	X	
9	4-6	3	X	
10	0	3		X
	1-4	3		X

The color code of each nibble must be rotated right one bit to maintain the same color as the nibble is placed horizontally across the screen.

The software must go through the following sequence to select this mode (this sequence will set Flg1 and Flg2 to a logic "zero"):

a) Set the HIRES flag
b) Reset the MIX flag
c) Set the 80-column flag
d) Reset and then set AN3
e) Reset and then set AN3 once more
f) Reset AN3

The rest of the Apple //e's flags must be in the following state:

80-column flag:	set
80 store flag:	set
PG2 flag:	used as a pointer: accesses odd bytes when set; accesses even bytes when reset.

7.6.3 160X196 IN SIXTEEN COLORS (Available only in RGB)

This video mode offers 20 more pixels of horizontal resolution than the 140X192 video mode. Because it uses all eight bits of each byte it "cleans up" the user interface; i.e., it is much more "user friendly" than the 140 mode. The ease with which the software can interface with this mode improves the speed with which animation may be performed.

Each nibble of each byte represents a color pixel that may represent any one of sixteen colors. The first two pixels are retrieved from extension RAM, while the next two pixels are retrieved from resident RAM; this algorithm is repeated across the screen. The color code does not need to be rotated and always represents the correct color.

The software must go through the following sequence to select this mode (this sequence will set Flg1 to a logic "one" and Flg2 to a logic "zero"):

a) Set the HIRES flag
b) Reset the MIX flag
c) Set the 80-column flag
d) Reset and then set AN3
e) Reset the 80-column flag
f) Reset and then set AN3
g) Reset AN3

The remaining Apple //e flags must be in the following state:

80-column flag:	set
80 store flag:	set
PG2 flag:	used as a pointer: accesses odd bytes when set; accesses even bytes when reset.

7.6.4 MIX MODE

This mode combines under software control the 560X192 and 140X192 video modes anywhere on the screen. The selection is performed through the most significant bit of each video byte because neither the 560 nor the 140 mode use this bit. When the software sets the most significant video bit to a logic "zero" the hardware displays the remaining seven bits as bit-mapped video; a logic "one" will instruct the hardware to display the next seven bits as color pixels. Because the color pixels require 4 bits to determine a color pixel, color aberrations will occur at the leading and trailing edges of the transitions from one mode to the other. The aberrations may be eliminated by blanking enough bytes at the beginning and end of the each transition to guarantee the four bit integrity of the corresponding color pixels.

The software must go through the following sequence to select this video mode (this sequence will set Flg1 and Flg2 to a logic "one"):

a) Set the HIRES flag
b) Reset the MIX flag
c) Reset the 80-column flag
d) Reset and then set AN3
e) Reset and then set AN3 once more
f) Reset AN3

The rest of the Apple //e's flags must be in the following state:

80-column flag:	set
80 store flag:	set
PG2 flag:	used as a pointer: accesses odd bytes when set; accesses even bytes when reset.

APPENDIX A

VIDEO MODE SOFTWARE SWITCH SETTINGS

Set software switches as follows to install the desired video **modes.**

AN3	TEXT	HIRES	80COL	Video mode
1	1	X	0	1. 40-column text.
1	1	X	1	2. 80-column text.
1	0	0	0	3. LORES mix with 1.
1	0	0	1	4. LORES mix with 2.
1	0	1	0	5. HIRES mix with 1.
1	0	1	1	6. HIRES mix with 2.
0	1	X	0	7. F/B text.
0	1	X	1	8. 80-column text.
0	0	0	0	9. LORES mix with 7.
0	0	0	1	10. MERES mix with 8.

F2	F1					
		0	0	1	0	11. F/B HIRES mix with 7.
0	0	0	0	1	1	12. 140X192 mix with 8.
0	1	0	0	1	1	13. 160X192 mix with 8.
1	0	0	0	1	1	14. Mix mode mix with 8.
1	1	0	0	1	1	15. 560X192 mix with 8.

NOTES:

1. X signifies a "do not care" state
2. TEXT is set (1) by microprocessor read or write to location $C051, it is cleared (0) by microprocessor read or write to location $C050.
3. HIRES is set (1) by microprocessor read or write to location $C057, it is cleared (0) by microprocessor read or write to location $C056.
4. For explanation of AN3 and 80COL see Chapters 2 and 3.
5. For RGB monitors text color for normal video mode is switch-selectable to colors other than white.

APPENDIX B

CHARACTER CELL FORMAT

Each cell is stored as a seven-by-eight bit array. This cell array is "bit-mapped" unto the screen. All bits containing a "one" are plotted as dots with the foreground color and all bits containing a "zero" are plotted in the background color. Each character array is stored as eight bytes--one byte for each row of seven dots. Only the low order seven bits are displayed; the high order bit is ignored.

Within the array, the least significant bit of each byte is the left-most dot displayed. The eight bytes (character block) making up one character are stored consecutively, with the top row first. The data bytes must be arranged as successive character blocks, of eight bytes each, corresponding to the ascending order of the 128 ASCII characters. The total font file containing the new character set must, therefore, contain 1024 bytes.

For example the letter "Y" would have to be coded as follows:

	Bit 0	1	2	3	4	5	6
Row 0	1				1		
1		1		1			
2		1		1			
3			1				
4			1				
5			1				
6			1				
7							

FORMAT OF A CHARACTER CELL FOR THE LETTER Y.

Note that all zero bits have been left blank.

SHAPE BLOCK GENERATION

The demonstration diskette contains a shape binary file called "plane". This file when BLOADED will reside in memory starting at address $0300 (decimal 768). You may examine the contents of this shape by performing the following:

```
RUN APLANE      Run Applesoft program to display the plane
PRESS A KEY     Program ends on a key press
LIST            List program when its done.
CALL -151       Enter the Apple //e monitor
300.317         Dump locations 300 through 317 to screen
3D0G            Return to Applesoft
```

The program listing should be:

```
10  PRINT CHR$(4);"BRUN HIRES"              Load Drivers
20  PRINT CHR$(4);"BLOAD PLANE,A$300"       Load Shape
30  &GR2                                    140X192 graphics
40  &DRAW(768,2,0,0,13,1?)                  Draw shape
50  GET A$                                  Wait for key press
60  &TEXT:&GR0                              Disconnect Drivers
70  HOME:END                                Clear screen and end
```

The 300 through 317 core dump should be:

```
300- 22 00 02 00 02 00 07 00
308- 27 20 2F A0 2F A0 3F E0
310- 3F E0 7F F0 FF F8 30 60
```

The shape and shape table were generated as follows:

1. The shape was first defined on paper:

```
    X   X
        X
        X
       XXX
    X  XXX  X
    X XXXXX X
    X XXXXX X
    XXXXXXXXX
    XXXXXXXXX
   XXXXXXXXXXX
  XXXXXXXXXXXXX
    XX     XX
```

2. The shape was then coded:

Bits

		3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0	Hex.
Row	0			1				1										22 00
	1							1										02 00
	2							1										02 00
	3						1	1	1									07 00
	4			1			1	1	1			1						27 20
	5			1		1	1	1	1	1		1						2F A0
	6			1		1	1	1	1	1		1						2F A0
	7			1	1	1	1	1	1	1	1	1						3F E0
	8			1	1	1	1	1	1	1	1	1						3F E0
	9		1	1	1	1	1	1	1	1	1	1	1					7F F0
	10	1	1	1	1	1	1	1	1	1	1	1	1	1				FF F8
	11			1	1						1	1						30 60

Note that the locations containing "zeroes" have been left blank. To enter and save the shape table perform the following:

```
CALL -151
300:22 0 2 0 2 0 7 0 27 20 2F A0 2F A0 3F E0 3F E0 7F F0 FF F8
30 60
3D0G
BSAVE PLANE,A$300,L$20
```

VIDEO-7 INCORPORATED LIMITED WARRANTY

VIDEO-7 INCORPORATED ("Video-7") warrants this product against defects in material and workmanship for a period of one year from the date of purchase. During the warranty period, Video-7 will repair or, at its option, replace this product at no charge.

In order to obtain warranty performance, return the product, transportation charges prepaid, to Video-7. Attach a copy of proof of date of retail purchase (such as bill of sale) and a description of the problem.

This warranty does not apply if the product has been damaged by accident, abuse, misuse or misapplication, or as a result of service or modification other than by Video-7.

VIDEO-7 IS NOT RESPONSIBLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM BREACH OF ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING DAMAGE TO PROPERTY AND, TO THE EXTENT PERMITTED BY LAW, DAMAGES FOR PERSONAL INJURY. THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow limitation of implied warranties, or the exclusion or limitation of incidental or consequential damages, so that the above limitations or exclusions may not apply to you.

12340 Saratoga Sunnyvale Rd. Suite 1
Saratoga, CA 95070

Video-7
550 Sycamore Drive
Milpitas, CA 95035