

# SMARTERM Intelligent Interface



Installation,  
Tutorial,  
and  
Reference Manual

**ADVANCED**  
**LOGIC** SYSTEMS

Mike Scott

224-3363

re: [unclear] [unclear] for [unclear]

THIS EQUIPMENT IS MARKETING PURSUANT TO A WAIVER OF FCC RULES PART 15 SUBPART J. OPERATION OF THIS COMPUTER IN A RESIDENTIAL AREA MAY CAUSE OBJECTIONABLE INTERFERENCE TO RADIO AND TV RECEPTION, BECAUSE IT EMITS MORE RADIO FREQUENCY ENERGY THAN THE FCC RULES ALLOW. IF INTERFERENCE OCCURS, THE USER WILL BE REQUIRED TO TAKE ALL STEPS NECESSARY TO CORRECT THE INTERFERENCE.

825-0205-A

SMARTERM  
INTELLIGENT INTERFACE  
INSTALLATION, TUTORIAL AND REFERENCE MANUAL

BY  
ADVANCED LOGIC SYSTEMS, INC.

ALS INTELLIGENT INTERFACES



ACKNOWLEDGMENTS

The following people and organizations have contributed in some manner to the making of this product, and without whose help this undertaking would most certainly have been an impossibility. Many thanks go to each and every individual.

At Advanced Logic Systems, Inc.: Grant Jasmin, Ray LaBarbera, Claudette Mauro.

At Apple Computer, Inc.: Steve Wozniak (Woz), Walt Broedner, Andy Hertzfeld, Randy Wigginton, Dick and Cliff Huston, Rick Auricchio, Wendell Sander, Thomas Root, Brian Gordon, John Viega, George Johnson, Bob Paratore and special thanks to Tom Whitney. Also, many thanks to the rest, whose list of names is so numerous, it could go on for pages...

At Brooks Technical Group, Inc.: Harley Licht.

At Computer Plus, Inc.: Dick and Lucy Applebaum, Mark Wozniak.

Finally, all the personal friends of those at Advanced Logic Systems Inc. whose comments and suggestions have helped make this a better product: Chris March, Chuck (Sr.), Georgia, and most of all, very special thanks go to Bruce Kinney, whose innovations and encouragement helped us make it through to the end.

This manual was written and printed using the Moonshadow Text Formatter, exclusively distributed for APPLE ][ computers by Computer Plus, Inc.



TABLE OF CONTENTS

|                                                                |    |
|----------------------------------------------------------------|----|
| Introduction.....                                              | 4  |
| Installation.....                                              | 5  |
| Operational Adjustment.....                                    | 6  |
| Input Features.....                                            | 8  |
| Output Features.....                                           | 18 |
| SMARTERM Graphics.....                                         | 29 |
| SMARTERM Graphics for Pascal.....                              | 36 |
| BASIC Updates, and using SMARTERM with Pascal 1.1 and CPM..... | 39 |
| Reference List of Features.....                                | 40 |
| Diagnostic Troubleshooting.....                                | 45 |

INTRODUCTION

Congratulations on purchasing your new ALS SMARTERM (TM) Intelligent Interface. SMARTERM is a member of a growing family of intelligent plug compatible interfaces under development at Advanced Logic Systems, Inc. We at ALS are committed to excellence in the design and manufacture of APPLE (TM) compatible intelligent interfaces. With the purchase of this peripheral, you have increased the INPUT/OUTPUT processing capabilities of your APPLE ][ (TM) computer to levels previously unattainable. SMARTERM is truly a feature-packed, highly intelligent interface. With your new SMARTERM, you can display 80 columns of text per line as compared to the standard 40 columns your APPLE normally displays. You can input and display upper and lower case characters as compared to only upper case normally found on your APPLE.

You have at your command a full 128 (256 including inverse) ASCII character set that can be displayed, including controls! You now have a new graphics mode (medium resolution, 160 X 72) in addition to your APPLE's standard graphics modes, and this new graphics mode is complimented by a built in set of vector and character graphics support routines. Virtually all common screen commands used by BASIC programs [INTEGER and APPLESOFT (TM)] are supported. With the addition of the SMARTERM Interface in your APPLE computer, Pascal will automatically recognize it as the system console with virtually no extra software reconfiguring on your part! Also, CPM and the new Pascal 1.1 are supported, with keypress and typeahead available to Pascal 1.1 (a SMARTERM exclusive feature). By using this peripheral, you in no way preclude the use of APPLE's standard text and color graphics modes. In fact, using SMARTERM's software selectable video source switching, you may elect to display SMARTERM's 80 column video or your APPLE's standard 40 column video on one monitor without the need to change video cables or flip special switches. Add to all of this a powerful new set of cursor editing functions; software selectable cursor definitions; a debug mode; absolute (GOTOXY) cursor addressing; support of the often used "one wire shift key" mod; 8 "NEW" keys previously unavailable on the APPLE keyboard and much more. Your new SMARTERM is a natural compliment to your APPLE ][ computer, and brings new dimensions to I/O processing that were not previously available in one peripheral.

NOTE: SMARTERM is a trademark of Advanced Logic Systems, Inc. APPLE, APPLE ][, and APPLESOFT are trademarks of Apple Computer, Inc.



INSTALLATION

The SMARTERM Intelligent Interface consists of only two parts: the actual printed circuit board, and a 16 inch video cable. The cable that is supplied is intended to connect the video output of the APPLE ][ to the SMARTERM Interface. A second video cable is necessary to connect the output of the SMARTERM Interface to the video monitor you have chosen. Your dealer should be instrumental in helping you obtain the proper cable needed for your particular monitor. We do not recommend modifying a standard television or using an RF modulator for displaying the 80 column output of SMARTERM. Standard television sets do not have the necessary bandwidth required to obtain a sharp, smear free 80 column display. We do recommend you use any 9,12,13 or 15 inch video monitor with a minimum bandwidth of 9 megahertz. Twelve and thirteen inch monitors generally give the most pleasing display. Nine inch (or smaller) monitors tend to be somewhat of an eyestrain, while fifteen inches and up usually results in characters that are larger than necessary for an appealing display.

Installation of the Interface card is relatively straightfoward and easy. Follow these simple steps.

1. Turn off the power switch on the back side of the APPLE ][. This step is essential to prevent damage to both the APPLE ][ and the SMARTERM Interface.

2. Remove the APPLE ][ cover. This is accomplished by pulling up on the cover at both corners of the rear edge until the two fasteners pop apart. Slide the cover backward until it is free of the system.

3. Notice that inside along the rear of the APPLE ][ is a row of 8 sockets or "slots". It is in one of these slots that you insert the SMARTERM Interface card. With the keyboard facing you, the leftmost slot is slot #0, and the rightmost slot is slot #7. The "fingers" portion of the SMARTERM Intelligent Interface will slide with some friction into the slot you have chosen and then seat firmly. SMARTERM is designed to work in slots 1 through 7. We suggest it be installed in slot #3, the fourth one from the left. The reason for this is because the APPLE Pascal Language System will recognize SMARTERM as the system console device only in this slot. All examples given in this manual assume the Interface is installed in slot #3.

4. Connect the 16 inch video cable provided by plugging one end in the APPLE ][ video output jack and the other end in the upper (marked J1) of the two video jacks on the SMARTERM board. The cable should thread through one of the three vertical openings in the rear of the APPLE ][ case without any extreme tension at either connector.

5. Install a second video cable (user provided) from the lower (marked J2) video jack on the SMARTERM board through one of the three vertical openings in the rear of the APPLE ][ case to your monitor.



This completes all the physical installation and interconnect required.

### OPERATIONAL ADJUSTMENT

1. With the cover of the APPLE ][ removed, and the SMARTERM Interface installed in slot #3, turn the system power on. Make sure the monitor connected to SMARTERM is also turned on.

2. After the monitor has had time to warm up, you should see a stable display of the APPLE standard 40 column video. If you do not, make the following adjustments: A) turn the video level pot on the APPLE motherboard clockwise until it stops. The pot is located by slot #7. This sets the video output level of your APPLE to maximum. B) Adjust the video level pot located on the upper edge of the SMARTERM card until a bright and acceptable image is seen on the monitor. If the picture rolls vertically, adjust the vertical hold control on the monitor until the picture becomes stable. Repeat this procedure to remove any possible horizontal tearing or rolling by adjusting the horizontal hold control on the monitor. Adjust the monitor's brightness control for maximum or near maximum brightness. Finally, adjust the monitor's contrast control until a sharp picture is obtained.

3. If your APPLE ][ is set up to use the Pascal Language System, you should boot up the APPLE1 diskette that comes with the language system. If you are not set up to use the language system, use your normal procedure to enter either INTEGER or APPLESOFT BASIC, type PR#3 and hit the RETURN key. The display should now show the normal APPLE1 boot message if you are operating from Pascal, or should be cleared and displaying a single BASIC prompt character in the upper left corner along with a flashing cursor underline (block for Pascal). If necessary, adjust the horizontal hold control on your monitor to obtain a non-torn and centered display. If the picture rolls vertically, adjust the vertical hold control just enough to obtain a stable display.

4. It will probably be necessary to adjust the brightness and contrast controls on your monitor to obtain a pleasing 80 column display. We recommend that the brightness be turned close to or at maximum, and the contrast be adjusted to obtain a clear, smear free image.

5. Your monitor should now be correctly adjusted for an optimum display. No further adjustments should be necessary to view either the standard 40 column APPLE ][ display or the new 80 column SMARTERM display. Since the correct video display for you is largely a matter of personal judgment and taste, the previous steps taken for initial adjustment should be "fine tuned" to obtain the kind of display you feel comfortable viewing. Fine tuning is largely a matter of adjusting the brightness and contrast levels of your monitor once a balance of the video output levels of the APPLE ][ and the SMARTERM Interface have been made.

As a final note on display adjustments, the objective you are trying to achieve is to obtain an acceptable 40 or 80 column display without having to readjust the monitor controls every time the video source is switched.

6. If you can not obtain a video output on your monitor, go directly to the chapter entitled "Diagnostic Troubleshooting" and follow the given procedures until you have obtained an acceptable output from the SMARTERM Interface. At this point, the SMARTERM Interface is correctly installed, adjusted and verified.

7. Replace the cover of the APPLE ][, remembering to start by sliding the cover into position front edge first. Push down on the two rear corners until they pop into place.

#### COMPATIBILITY WITH OTHER PERIPHERALS

SMARTERM has been designed to meet all Apple peripheral hardware and software protocols, insuring that the Interface will work correctly with all currently available Apple Computer, Inc. peripherals. In addition, the SMARTERM Interface should work correctly in tandem with any original equipment manufactures peripheral interface card, provided that the other card is known to observe all Apple hardware and software protocols. Proper operation of the SMARTERM Interface with non-Apple Computer, Inc. peripherals can not be insured if the interface card in question does not meet all Apple peripheral hardware and software protocols.



INPUT FEATURES

Before reading this manual and attempting to utilize the SMARTERM Intelligent Interface, it is highly recommended that you be familiar and accustomed with your APPLE ][ computer and most of its commonly used features. If you are a first time user of the APPLE ][, we recommend you start by reading the manuals that come with the system before utilizing the more involved features of SMARTERM. This will help you to better understand the procedures and examples given in this manual.

SMARTERM is designed to work in all APPLE ][ language environments. By leaving the Interface installed in slot #3, Pascal will automatically recognize it as the system console. The only change you may want to make is to one parameter in the SYSTEM.MISCINFO file on your APPLE1 diskette. This is accomplished by executing the program SETUP.CODE supplied on any APPLE3 diskette. This program allows changes to be made to the SYSTEM.MISCINFO file on your APPLE1 diskette. All original APPLE1: SYSTEM.MISCINFO files have, as part of their information, system console setup parameters. APPLE COMPUTER, INC. has initially set the console SCREEN WIDTH parameter to 79 columns, which allows abbreviated prompt lines on their standard 40 column screen. Changing the SCREEN WIDTH parameter to 80 columns insures that all Pascal prompt lines will appear in their unabridged forms. Refer to the Apple Pascal Preliminary Reference Manual for the proper steps to take in changing the APPLE1 SYSTEM.MISCINFO file. This change is an option on your part: not making it will in no way affect the operation or performance of SMARTERM.

We now assume that the SMARTERM Interface has been correctly installed and adjusted in slot #3 of your APPLE ][ computer. The following tutorial begins by assuming the system is configured for BASIC: INTEGER or APPLESOFT. If so, turn your system on and enter the BASIC of your choice. If your APPLE ][ is configured for Pascal, then you should boot the APPLE1 diskette and wait for the familiar command prompt line to appear. Type H and hit the RESET key. We are now ready to begin the tutorial.

Notice that the standard 40 column APPLE ][ display is present on your monitor. SMARTERM is designed to give you the standard 40 column display: 1) upon power up, 2) whenever RESET is hit, or 3) under your discretion by either:

- A) typing a few special characters during input, or
- B) printing those same special characters during output.

This gives any program the power of switching between APPLE 40 column text, LORES and HIRES graphics, or SMARTERM 80 column text and medium resolution graphics displays.



INITIALIZATION

First, let's initialize SMARTERM by typing either one of the following two commands from the keyboard:

PR#3      or      IN#3

Complete the command by hitting the RETURN key. Either one of the two commands will work correctly since SMARTERM initializes itself to process both keyboard input and display output.

There should now be a clear screen with the exception of a BASIC prompt character and a flashing cursor underline in the upper left corner of the display. Hit the RETURN key a few times and you will notice that the output behaves just as if this were the 40 column screen.

Try typing PR#3 or IN#3 again and you will notice that SMARTERM does not reinitialize itself. SMARTERM can distinguish between cold (first time) and warm (all subsequent) startups. This is to prevent current screen data from being lost (cold starts force a clear screen) when a program is switching between devices during I/O. (This is not a problem for Pascal since it handles I/O device switching in a much different manner.) This feature allows a BASIC program to PRINT something to SMARTERM, switch output devices by issuing a different PR#N, do some output to the second device, then finally switch back to SMARTERM by executing another PR#3. All subsequent PR#3 (or IN#3) commands will not reinitialize SMARTERM unless the RESET key is pressed or the specific SMARTERM exit procedure is executed.

NOTE: because of the method used by SMARTERM to detect device switching, there is an exception to the above rule. Whenever a (BASIC or machine language) program simultaneously switches both input and output to another device (for example a remote terminal), SMARTERM will assume its most recent entry is a cold start because both the input and output pointers in the APPLE are pointing to some device other than itself.

USING SMARTERM WITH DOS

The interfacing of SMARTERM to APPLE's DOS is practically automatic, and requires very little in the way of software overhead. To cold start SMARTERM from an APPLESOFT program with DOS active, use the following (we assume the Interface is installed in slot #3):

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

Since the CHR\$( ) function is not available to INTEGER BASIC, one must remember to start the "PR#3" string with a CTRL-D. The second empty PRINT statement is absolutely necessary, and may not be omitted. If you

try to print something to SMARTERM before executing this mandatory empty PRINT statement, unpredictable results will occur. This null PRINT statement is required only after the initial cold start "PR#3" command. All subsequent device switching through DOS will cause no problems, and an empty PRINT statement after all warm start "PR#3" commands is not necessary.

## EXITING THE INTERFACE

As mentioned above, proper exiting of SMARTERM is done by executing a special procedure. The specific SMARTERM exit procedure is invoked by typing:

ESC CTRL-Q

ESC is an abbreviation for ESCape, and is a single keystroke. CTRL-Q means ConTRoL-Q, and implies that the the ConTRoL key is pressed down simultaneously with the following key (in this case a Q). To verify this, type in this command now. The video monitor should now be displaying 40 column video. This is apparent by a significant change in character size. Also, after exiting SMARTERM, always type "TEXT" followed by the RETURN key to insure you will always have a full 24 line text window.

Initialize SMARTERM again by typing:

PR#3 or IN#3

and finish by hitting the RETURN key. Become comfortable with these entry and exit procedures by trying them a few times and verifying that they do indeed work. Next we shall look into the input control features of SMARTERM.

## SWITCHING BETWEEN UPPER AND LOWER CASE

Initialize SMARTERM by one of the previously described methods, and type a few random characters. Notice that all alphabetic characters (alpha's) are in upper case (capitals). This is the normal state of the APPLE keyboard, and is also the default mode when SMARTERM is initialized. SMARTERM can also generate and display lower case characters. To obtain lower case characters from the keyboard, begin by typing:

CTRL-A

Now type a few characters. You should be seeing lower case characters on the display. Experiment by typing all of the alphabets. Verify that the numbers and punctuation marks are unaffected by this new change



to the keyboard. We say that the keyboard is in lower case, or that it is downshifted. To get upper case characters again, type:

#### CTRL-Z

Verify that the keyboard is back to upper case again by typing a few more characters. We say that the keyboard is in capitals, or shift-lock mode.

#### THE SOFT SHIFT KEY

We now have at our disposal a means for obtaining lower and upper case characters by simply typing CTRL-A and CTRL-Z to lock the keyboard into downshift and shift-lock modes. This is a handy feature, but could become quite tiresome if we were typing a letter or report where capitals are only needed for beginning sentences and proper nouns. To give us capitals, there should be a different, simpler method other than by constantly shifting cases. As an example of how to use this new feature, reenter lower case mode, and type the following:

#### CTRL-A hello

Note: the 'h' in 'hello' immediately follows the CTRL-A prefix, we have put a space between them in the example above for greater legibility. Notice that the first letter, H, becomes capitalized! The CTRL-A is our new soft shift key feature. It behaves just like a shift key found on all typewriters, with the only difference being that it must be struck and released before typing the capital letter desired. SMARTERM knows when it is in lower case mode, and interpretes any CTRL-A as meaning "shift the next and only the next character to upper case". Notice that this feature only works when the keyboard is locked into lower case (downshift) mode. If you try to use this from the upper case mode, SMARTERM interprets a CTRL-A as the command to downshift the keyboard: not to capitalize the next keystroke.

The CTRL-A key is known as a "dual function" key. This is the only keystroke SMARTERM recognizes as having dual function characteristics. It is defined a dual function key because the results obtained by pressing it are interpreted in two different ways and are directly related to the mode the keyboard is in when it is pressed.

Now that we have a mechanism for generating capitals from lower case mode, spend some time becoming familiar with the soft shift key feature by typing a few sentences and capitalizing some of the words.

Since both INTEGER and APPLESOFT BASIC do not understand lower case characters, you will need to press CTRL-Z to lock the keyboard to upper case before entering or editing programs. There is an exception to this rule. When entering literal data between the quote marks of a string, you may use both upper and lower case characters. BASIC does



not care about the type of characters that are assigned to a string variable or string literal.

Here is an example. Begin by locking the keyboard to upper case and enter the following short program:

```
10 PRINT "Hello ";GOTO 10
```

Notice that in this example it is easier to issue a keyboard downshift command before typing "ello " in Hello than it is to type CTRL-A CTRL-A Hello because the former method requires one less keystroke.

At this point, RUN the program and let it execute for a few seconds, then press CTRL-C to stop it. This simple demonstration shows that BASIC will accept upper and lower case characters in strings. If you wish, reenter the above program all in lower case and verify that a SYNTAX ERROR will result. This is because BASIC recognizes keywords and variables in upper case only. This is not true for Pascal programs, because the Pascal compiler was designed to make sense out of most combinations of upper and lower case characters, provided the program lines follow proper Pascal syntax.

BASIC programs that stop to input string data will also gladly accept mixed case responses, as long as the programs are written to recognize them. If your programs have not been written to digest lower case characters as input, then you must keep the keyboard locked to upper case, or modify your programs to understand lower case data.

#### THE HARD SHIFT KEY

There is a one wire modification that can be made to your APPLE computer called "The one wire shift key" mod. This simple modification can be done by your dealer (or by yourself, if you feel so inclined). It consists of soldering a wire from the keyboards SHIFT key and running the other end over to PB2 (paddle button 2) on the GAME I/O connector inside your APPLE. If you have this done to your system, you now have at your disposal a REAL shift key! SMARTERM recognizes this modification if you have it. This is done by typing the following:

```
CTRL-V CTRL-A
```

The CTRL-V invokes the Shift Mod feature, (explained under the section entitled SMARTERM EXCLUSIVE INPUT FEATURES) and reads the state of the shift key through PB2. If you do not make this modification, don't worry: SMARTERM is smart enough to figure this out, and prevents anything strange from happening if you put it into Shift Mod mode. The reason for typing CTRL-A is to force the keyboard to lower case mode in case it is not currently in it. To use this "HARD" shift key feature, start typing as if the APPLE keyboard were that of a standard typewriter, and enjoy!

## SCREEN EDITING

SMARTERMAN duplicates all APPLE ][ screen editing features and introduces its own unique set of powerful editing features as well.

All of the following commands represent APPLE ][ standard nondestructive cursor positioning directives:

|       |                                 |
|-------|---------------------------------|
| ESC-A | Move right one character column |
| ESC-B | Move left one character column  |
| ESC-C | Move down one character row     |
| ESC-D | Move up one character row       |

To execute one of these commands, press the ESCape key and release it, then follow with the character that correspondes to the direction in which you desire to move.

Try a few of these commands. Notice that to repeate any given command you must type both ESC and the direction control character each time you wish to move the cursor.

The next group of cursor motion commands provide the same functions as the previous but differ in two respects:

1) The ESC key need only be pressed once to obtain multiple steps of cursor motion in any direction. By the same token, combinations of different motion directives also need to have the ESC key pressed only once.

2) The keys that make up this group of cursor commands are arranged as a cluster in the shape of a + sign, implying the four directions of motion. These keys are:

|       |                                 |
|-------|---------------------------------|
| ESC-I | Move up one character row       |
| ESC-J | Move left one character column  |
| ESC-K | Move right one character column |
| ESC-M | Move down one character row     |

Here is an example of the added usefulness of these cursor command keys. To move three columns left and four rows up, press the following sequence of keys:

ESC JJJIII

As can be seen, this set of cursor motion keys is highly suited for quick cursor positioning and allows for flexibility in screen edits.



## LEFT AND RIGHT ARROW KEYS

The left arrow key found on the APPLE ][ keyboard has both destructive and nondestructive features. It is destructive in that each time it is pressed, the last character typed into the current input buffer is erased and the buffer count is decremented. (An input buffer is a temporary place in memory that stores what you are currently typing on the keyboard. Only when the RETURN key is hit does the entire buffer get sent to the program asking for keyboard input.) Pressing the left arrow key is seen as a backing up motion of the cursor on the screen. Start a new line (by hitting the RETURN key) and type a few characters. Press the left arrow key until the cursor backs up to the first character typed on this line. Pressing the left arrow key again will start a new line showing that indeed the APPLE was keeping track of erasing the current input buffer. The APPLE cannot back up over an empty input buffer, so it simply generates a new prompt line for you. The nondestructive nature of the left arrow key is that it does not actually destroy what is on the screen when backing up, except when using Pascal. The left arrow and CTRL-H keys behave identically and both are interpreted as the ASCII backspace command.

Two more keys that behave identically are the CTRL-U and right arrow. Pressing either of these keys results in a nondestructive screen read, or 'screen pick'. This enables the copying of data currently on the screen. For instance, if we were to list a particular BASIC program line and then use the cursor motion commands to position the cursor at the listed line, we can copy the line over again by simultaneously pressing the right arrow or CTRL-U with the REPEAT key up to the point where edits or additions need to be made. This cuts down on the actual number of keystrokes needed to copy over most (or all) of a large program line. If you are not familiar with the screen read capabilities of the APPLE ][, enter a few simple BASIC text lines and list them. Use the cursor motion commands and right arrow feature to copy over an entire program line, and finish the entry by hitting the RETURN key. Verify that the same line was reentered by listing it out again.

## DESTRUCTIVE SCREEN EDITING FEATURES

The following are destructive screen edit commands:

```
ESC-@  Go home and clear the screen
ESC-E  Clear to the end of line
ESC-F  Clear to the end of screen
```

The first, ESC-@, positions the cursor at the upper left corner (Home) and clears the entire screen.

ESC-E will clear from the current cursor position to the end of the line.



ESC-F clears from the current cursor position to the end of the screen. Both this command and ESC-E leave the cursor at the current position.

The above three destructive edit commands require the ESC key to be pressed before each desired operation.

### ADVANCED SCREEN EDITING

These next four screen edit commands are new cursor motion directives and are available only when SMARTERM is being used as the current input device. They are:

|       |               |                       |
|-------|---------------|-----------------------|
| ESC-G | Forward tab:  | multiple of 8 columns |
| ESC-H | Backward tab: | multiple of 8 columns |
| ESC-L | Down tab:     | multiple of 4 rows    |
| ESC-N | Up tab:       | multiple of 4 rows    |

These commands are advanced cursor motion directives and are said to possess accelerated motion. With these new functions it is possible to move the cursor rapidly across the screen in any direction. Horizontal directives move the cursor to column numbers that are multiples of eight (zero, eight, sixteen, etc), with a maximum velocity of 8 columns per move. Vertical directives move the cursor to row numbers that are multiples of four (zero, four, eight, etc), with a maximum of four rows per move. Regardless of the cursor's current row and column number, a destination row or column will always be the proper multiple as cited above. These commands are all nondestructive (except of course when the action taken causes the screen to scroll) and need to have ESC pressed only once for multiple directives.

### NOTES ON ALL EDITING FEATURES

All of the ESC prefixed cursor motion directives discussed up to this point are available to all languages except Pascal or interpretive languages written in Pascal. Pascal has its own protocols for cursor motion directives when using the screen oriented editor and when utilizing normal console input and output. We suggest you read the section in the APPLE Pascal Preliminary Reference Manual concerning the screen oriented editor before attempting to use it. All standard features of the Pascal screen oriented editor are available when editing under SMARTERM (this entire manual was written using the Pascal screen oriented editor and a SMARTERM Intelligent Interface).

As a final note on editing features, all directives will work from both upper and lower case, although it must be remembered that this set of features is input oriented. The discussed commands can only be

invoked during keyboard input, i.e. when the cursor is visible on the screen.

## SMARTERM EXCLUSIVE INPUT FEATURES

SMARTERM includes as part of its standard firmware support various exclusive features that are highly useful and not otherwise found on a standard APPLE II computer. Descriptions of these features follow.

1) CTRL-K. SMARTERM generates from the keyboard the otherwise unattainable left bracket character, [, whenever CTRL-K is typed.

2) CTRL-V. This is the Shift Mod feature. It supports the often made "one wire shift key" mod. It converts the APPLE SHIFT key to a real one. When invoked, in addition to providing a standard shift key, it provides the following extra keyboard changes. The M, N and P keys on the APPLE keyboard usually provide the ], ^ and @ characters whenever the SHIFT key is held down simultaneously with one of them. When CTRL-V is pressed, one of the following actions will take place.

A) If the keyboard is currently locked to upper case, the M, N and P keys do not get translated into the ], ^ and @ characters when the SHIFT key is simultaneously held down. M, N and P remain the same.

B) If the keyboard is currently locked to lower case, pressing the SHIFT key simultaneously with either the M, N or P key will generate a true capital M, N or P. This allows these three capitals to be generated using only the SHIFT key instead of typing the usual CTRL-A prefix. Note: this special feature works only with the M, N and P keys.

To regain the normal ], ^ and @ characters during input when in Shift Mod mode, see the "NEW KEYS" feature [#5] explained below. To exit Shift Mod mode, simply type CTRL-V again. This is a toggled feature, invoking it once enables it if it is disabled, and vice-versa. As with any toggled feature of SMARTERM, invoking the feature an even number of times has the same effect as not invoking it at all.

3) ESC CTRL-Q. This is the previously discussed SMARTERM exit procedure. Invoking this procedure returns I/O control back to the standard 40 column APPLE screen. This feature is not available to Pascal.

4) ESC-V. Dual video feature. This is a toggled feature, and provides output to the APPLE 40 column screen simultaneous to output on SMARTERM's 80 column screen. If a monitor or color television is hooked up to the APPLE's auxillary video output (using an RF modulator, if necessary), you can now obtain text in both 40 and 80 columns. If any



lower case characters are typed or outputted, the information will appear meaningless on the 40 column screen since your APPLE ][ has no provisions for lower case character generation. If only one monitor is available, then the standard APPLE 40 column video output can still be viewed without necessitating hardware changes. To see the 40 column display, refer to the section in the output features chapter entitled "TERMINAL ESCAPES", subsection "VIDEO SOURCE SWITCHING". The dual video mode is not available to Pascal (although video source switching can still be utilized from Pascal for viewing HIRES and LORES graphics displays).

If this feature is used, it is important to remember that SMARTERM's screen may appear as though it were only 40 columns wide. That is, output generated by the APPLE (for instance, by listing a BASIC program) will tend to force SMARTERM to behave like a 40 column screen. This is because the APPLE 40 column screen firmware usually takes precedence over output formatting, forcing SMARTERM to duplicate the actions that it is taking; the final result usually is similar displays.

5) CTRL-N. This is the prefix character for the NEW KEYS feature and the VERBATIM mode feature. The NEW KEYS feature expands your APPLE keyboard to include characters not previously typeable:

\ \_ ` { | } ~ DEL ] ^ @

These characters are (in order): backslash, underline, accent, left brace, logical or, right brace, tilde, delete, right bracket, carrot (up arrow), and the at-sign. All new keys are a two key sequence and start with CTRL-N. They are completed with one of the following keystrokes:

0 1 2 3 4 5 6 7 8 9 :

Only the above 11 characters will generate the NEW KEYS. Typing any other character immediately after CTRL-N (including another CTRL-N) sends a VERBATIM character to the system. A VERBATIM character is one that is not processed by SMARTERM in any way, but is instead simply passed along to your APPLE. This is useful for sending control characters normally used by SMARTERM for keyboard processing to a program that needs it for its own control purposes. For example, since CTRL-K is processed by SMARTERM during input as a left bracket, typing CTRL-N CTRL-K will not generate a left bracket but will instead send a CTRL-K character to the program asking for input. Also, any normal alpha-numeric character typed after CTRL-N will be passed along to your APPLE as it normally would.

OUTPUT FEATURES

This chapter is a tutorial on how to use the output features of SMARTERM. We recommend that the previous chapter on SMARTERM's input features be read first before attempting to utilize its advanced output capabilities. Most of the output features of SMARTERM are available in all APPLE ][ language environments.

The SMARTERM screen is a matrix or grid made up of rows and columns. There are 24 rows of 80 columns each for a total of 1,920 separate characters per screen. This is twice the density of the standard APPLE screen. The top most row (or line) shall be designated row 0, and the left most column shall be designated column 0, making the top left corner of the screen absolute character location 0,0. It is necessary for us to view the screen in this manner because Pascal (and BASIC) can address the screen as an absolute coordinate system for maximum flexibility in output formatting. This will be examined later in the GOTOXY control character subsection.

The output processing firmware of SMARTERM is designed around a concept commonly called a "STATE MACHINE". That is, SMARTERM decides what to do with a particular character that is output to it by remembering what the last operation, or state was. In this way, it is possible to output various combinations of characters (special multiple character command sequences) that would normally appear to be nonsense, but in reality tell SMARTERM to do some powerful things. (These special commands are called "TERMINAL ESCAPES", and we shall see later just how useful they are to us; one of these terminal escapes lets us change the cursor into one of eight different definitions!)

To begin, let us consider the first, or normal output state. In this state, all of the alphabetic (upper and lower case), numeric, punctuation and special characters are simply printed on the screen at the current print position. There are a total of 128 different characters you can print: the entire standard ASCII set (although some are only visible in the debug mode; this will be explained later). Most control characters (bell, line feed, tab, form feed, etc.) are processed, and the output state machine knows to continually stay in the first, or normal state.

## NORMAL AND INVERSE VIDEO

Characters can be printed in normal or inverse video (normal only for Pascal). To obtain inverse characters from INTEGER BASIC, type:

POKE 50,63

and hit the RETURN key. The INTEGER BASIC prompt character should now



be in inverse video. Next, enter this short program:

```
10 PRINT "ABCDEFGG...":END
```

RUN this program and notice that it prints its output in inverse video. LIST the program and it should also output in inverse video. To obtain normal video again, type the following [from this point on we shall assume that the RETURN key will be used to complete a line of input, no further mention of it will be made] :

```
POKE 50,255
```

Normal video now resumes. Verify this if you wish by RUNNING the previously entered short program again and observing the output. Notice that even though SMARTERM may be instructed to display inverse video for all output, characters that are typed and echoed to the screen during input are always displayed as normal video. This is a function of the APPLE firmware that gets a line of keyboard input (the GETLIN subroutine) and is not caused by SMARTERM.

APPLESOFT provides us with a simpler method for obtaining inverse output. Refer to the APPLESOFT ][ tutorial manual for a complete description on the use of INVERSE and NORMAL for output.

NOTE: SMARTERM does not have character blink as an output option.

### SPECIAL BASIC PROCESSING

The customary output positioning commands TAB, HTAB and VTAB are fully processed by SMARTERM to retain compatibility with your current software. Since SMARTERM is capable of displaying a screen of 80 X 24 characters, a special method must be incorporated into BASIC programs to TAB (or HTAB) past the 40th character column. This method is as follows:

```
POKE 36,N
```

Where N is the column position that you wish output to start printing at. Remember, when using this method for tabbing, the valid column range is:  $0 \leq N \leq 79$ .

There are several restrictions when using TAB and HTAB in BASIC programs. No TAB (HTAB) can cause printing to occur to the left of the last printed character on the current line. Attempting to do so will cause the character to be printed to the right of the last printed character. A TAB (HTAB) of less than 18, if it ends directly on a character already printed, may simply be tabbed from that character's position.

VTAB provides identical results on SMARTERM's screen as compared

to the standard APPLE 40 column screen, so no modifications to existing programs are necessary.

In addition to retaining full vertical and horizontal positioning control for output, the print field separator ",", (comma tabbing) in a print statement now gives twice as many fields (maximum) on the screen: 10 for INTEGER BASIC and 6 for APPLESOFT.

Full indenting and field separation also occur when program lines greater than 80 characters in length are listed on the screen.

## RUN TIME FEATURES

SMARTERM gives you two special output run time features. The first is the SUSPEND output, or STOP LIST feature. Whenever a program is outputting large amounts of continuous data, or a program listing is rapidly scrolling by on the screen, it is possible to suspend the output by typing:

### CTRL-S

This effectively "hangs" the APPLE in a tight keyboard polling loop. The system remains suspended until a second key is struck (any key other than ESC, CTRL, SHIFT, REPT and RESET) at which time normal output processing then resumes.

The second run time feature is the FLUSH output command. to invoke FLUSH, type:

### CTRL-F

when a program is printing data to the screen. Invoking FLUSH cancels all output to the screen. It does not affect operation of the running program in any other way. The net effect of FLUSHing screen output is generally one of speeding up output bound programs. One would normally use this feature when a program generates large amounts of data that does not need to be viewed on the screen. To cancel FLUSH, simply type CTRL-F a second time. FLUSH is also automatically canceled whenever keyboard input is requested.

## THE DEBUG MODE

During the course of developing programs involving complicated screen output, many programmers find the need to view the actual control codes being sent to the screen. It has been found that this kind of feature is invaluable as an aid to expedite program debugging.

Typing or outputting a CTRL-W invokes the DEBUG mode. ALL



output to SMARTERM will be displayed. Control characters can be recognized by one of several ways. Some are miniature letters with an underline. Some, such as the bell, are symbols with an underline. Others are various combinations of single or double arrows. Generally, the one thing that makes all printed controls recognizable (except those that are special arrow symbols) is the characteristic underline. Control characters can be displayed in normal or inverse video. The only two screen functions that will occur during DEBUG mode are advance (by default), and scrolling of the screen. The only control character recognized by SMARTERM as having active status during DEBUG mode is CTRL-W. This provides us with a way out of DEBUG mode, hence; DEBUG is a toggled feature.

The following is a translation table for determining what control characters are being displayed during the DEBUG mode:

| CNTRL | SYMBOL   | CNTRL | SYMBOL   | CNTRL | SYMBOL   | CNTRL | SYMBOL   |
|-------|----------|-------|----------|-------|----------|-------|----------|
| A     | <u>A</u> | B     | <u>▶</u> | C     | <u>◀</u> | D     | <u>␣</u> |
| E     | <u>E</u> | F     | <u>F</u> | G     | <u>⬆</u> | H     | <u>⬇</u> |
| I     | <u>•</u> | J     | <u>⬇</u> | K     | <u>⬆</u> | L     | <u>⬇</u> |
| M     | <u>■</u> | N     | <u>⬆</u> | O     | <u>⬆</u> | P     | <u>⬆</u> |
| Q     | <u>1</u> | R     | <u>2</u> | S     | <u>3</u> | T     | <u>4</u> |
| U     | <u>u</u> | V     | <u>v</u> | W     | <u>w</u> | X     | <u>X</u> |
| Y     | <u>Y</u> | Z     | <u>Z</u> | [     | <u>[</u> | \     | <u>\</u> |
| J     | <u>J</u> | ^     | <u>^</u> | -     | <u>-</u> |       |          |

#### CONTROL CHARACTERS

SMARTERM recognizes 17 distinct control characters (not counting the three special controls for STOP LIST, FLUSH and DEBUG) for screen control. The following is a complete list of these control characters:

##### (ENQ) CTRL-E: DOWN TAB

This control character moves the vertical print position down to the next row that is a multiple of 4 (0, 4, 8 etc.). If the current position is within 3 line of the bottom of the screen, the screen will scroll.

##### (ACK) CTRL-F: UP TAB

This code moves the print position up to the previous row number that is a multiple of 4. If the current vertical position is already within 3 rows from the top, the print position will be forced to row 0.

(BEL) CTRL-G: BELL

This is the bell character. It beeps a 1 Kiloherz tone on the APPLE speaker.

(BS) CTRL-H: BACK SPACE

Each time a back space is processed, the horizontal print position backs up by one column. When column 0 is reached, the next back space will advance the print position to column 79 of the previous row, unless row 0 was the last row. In this case, the print position will be forced to the upper left corner of the screen (absolute screen location 0,0).

(HT) CTRL-I: TAB

The horizontal tab is a standard control character, and advances the horizontal print position to the next column number that is a multiple of 8. If the tab will send the current print position past column 79, the print position will advance to column 0 of the next line.

(LF) CTRL-J: LINE FEED

This is the line feed character. It advances the row position by one. If the action taken advances the print position to row 24 (the 25'th row), the screen will scroll.

(VT) CTRL-K: CLEAR TO THE END OF SCREEN

Control K will clear from the current print position to the end of the screen. The current print position does not change. Note: this character can only be generated from the keyboard by using the VERBATIM mode since CTRL-K is normally translated into the [ character. APPLESOFT should use a CHR\$(11) to output a CTRL-K.

(FF) CTRL-L: CLEAR SCREEN

This control code will force an automatic HOME of the print position, and then clear the entire screen.

(CR) CTRL-M: CARRIAGE RETURN

A carriage return forces the current horizontal print position to column 0 on the current line. All non-Pascal based programming environments will also cause an automatic line feed to be generated.



(DC2) CTRL-R: BACK TAB

This control character performs a reverse, or back tabbing of the horizontal print position. It forces it to the previous column number that is a multiple of 8. If the operation takes the print position to before column 0, then it will go to column 72 of the previous row. If the top of the screen had already been reached, then the print position is forced to the upper left corner (absolute screen location 0,0).

(EM) CTRL-Y: HOME

This is the home control code. It forces the current print position to the upper left corner of the screen (absolute screen location 0,0).

(SUB) CTRL-Z: CLEAR LINE

This is the clear line control character. It will cause the current line to be erased. The current print position is not changed. Like CTRL-K, control Z can only be generated from the keyboard by using the VERBATIM mode. APPLESOFT should use CHR\$(26) to output a CTRL-Z.

(FS) CTRL-\: FORWARD SPACE

This is the nondestructive forward space. It moves the current print position forward (right) one column. An automatic carriage return will take place if the forward space moves the print position past column 79. An automatic line feed will also occur if the forward space command is issued by any non-Pascal language. APPLESOFT can generate this command by using CHR\$(28) for the character.

(GS) CTRL-]: CLEAR TO THE END OF LINE

This control will erase from the current print position to the end of the line. The current print position is not changed.

(US) CTRL-\_: REVERSE LINE FEED

A reverse line feed decrements the current row position to the previous one. The current row position pins when the top of the screen is reached. This control character is the only other one that can not be generated from the APPLE keyboard. A CHR\$(31) will do nicely from APPLESOFT BASIC.

This completes the definitions for 15 of the 17 control characters recognized. Every character that we have discussed up to

this point (be it a printable or a control) has been processed at the first state of the output state machine. The next two control characters are special in that they cause the output state machine to change states. These two control characters require the user to send multiple character codes for the desired operation to take place. The first of these two control characters is known as GOTOXY. It is the topic of the next section.

### GOTOXY

SMARTERM has built into its repertoire of functions the capability of absolute (X,Y) screen addressing. This is commonly referred to as the GOTOXY function. As stated earlier, SMARTERM's screen can be viewed as an 80 (horizontal) by 24 (vertical) character cell grid, with the top left corner representing the absolute screen coordinate 0,0 (origin). Positive X is from left to right, and the absolute range is 0 to 79. Positive Y is from top to bottom, and the absolute range is 0 to 23. This imaginary matrix of 1,920 characters can be randomly accessed by simply issuing the GOTOXY control character, followed by an X value and a Y value. The X and Y values sent are simply two ASCII characters that are translated by SMARTERM into values from 0 to 79. If either the X or Y value sent is out of range, then only the direction in range will be processed. If both quantities are out of range, the GOTOXY is canceled. To initiate a GOTOXY, send the following sequence of characters:

CTRL-^ Xvalue Yvalue

These three characters are: the ASCII Record Separator (control ^) followed by the desired character that represents the appropriate X (column) position desired, followed by the character representing the desired Y (row) position.

The following table is a translation of the ASCII characters that are used as X and Y values for GOTOXY commands.



TABLE 1

ABSOLUTE  
GOTOXY ADDRESS VALUES

| X OR Y | ASCII<br>CHAR | X  | ASCII<br>CHAR | X  | ASCII<br>CHAR |
|--------|---------------|----|---------------|----|---------------|
| 0      | SPACE         | 27 | ;             | 54 | V             |
| 1      | !             | 28 | <             | 55 | W             |
| 2      | "             | 29 | =             | 56 | X             |
| 3      | #             | 30 | >             | 57 | Y             |
| 4      | \$            | 31 | ?             | 58 | Z             |
| 5      | %             | 32 | @             | 59 | [             |
| 6      | &             | 33 | A             | 60 | \             |
| 7      | '             | 34 | B             | 61 | ]             |
| 8      | (             | 35 | C             | 62 | ^             |
| 9      | )             | 36 | D             | 63 | _             |
| 10     | *             | 37 | E             | 64 |               |
| 11     | +             | 38 | F             | 65 | a             |
| 12     | ,             | 39 | G             | 66 | b             |
| 13     | -             | 40 | H             | 67 | c             |
| 14     | .             | 41 | I             | 68 | d             |
| 15     | /             | 42 | J             | 69 | e             |
| 16     | 0             | 43 | K             | 70 | f             |
| 17     | 1             | 44 | L             | 71 | g             |
| 18     | 2             | 45 | M             | 72 | h             |
| 19     | 3             | 46 | N             | 73 | i             |
| 20     | 4             | 47 | O             | 74 | j             |
| 21     | 5             | 48 | P             | 75 | k             |
| 22     | 6             | 49 | Q             | 76 | l             |
| 23     | 7             | 50 | R             | 77 | m             |
| 24     | 8             | 51 | S             | 78 | n             |
| 25     | 9             | 52 | T             | 79 | o             |
| 26     | :             | 53 | U             |    |               |

Notice that the X arguments 60, 63 and 64 can be generated using the NEW KEYS mode by typing CTRL-N 0, CTRL-N 1 and CTRL-N 2 respectively. This is because these three characters are not available from the APPLE keyboard.

The GOTOXY function of SMARTERM is primarily intended for use with Pascal. Pascal programs can utilize the I/O intrinsic, GOTOXY(Xcoord,Ycoord: INTEGER) to utilize full control of the screen. In this case, the preceding table is not used, and the programmer need only send the desired coordinates in integer form: Pascal automatically handles the conversion. The preceding table is included to facilitate use from machine language and BASIC programs. It is more likely that most BASIC programs will use the standard TAB, HTAB and VTAB commands since this provides upward compatibility with most older programs. When writing machine language programs, the GOTOXY function will provide the easiest method of random screen addressing.

The GOTOXY function utilizes three states of the output state machine. Two of these states are unique to GOTOXY: the X and Y processing states. The GOTOXY initiator, CTRL-^, is interpreted in the first, or normal output state. When SMARTERM decodes this control character as a GOTOXY request, it switches to state 2, the X processing state. The next character is expected to be in the range given in table 1. Characters outside this range are ignored. If it is in range, it is then loaded into SMARTERM's internal X address variable. In either case, SMARTERM advances to state 3, the Y processing state. If the Y character is not in range, GOTOXY is cancelled. If it is in range, the value is loaded into SMARTERM's internal Y address variable and the function completes by returning to state 1 for normal output processing.

## TERMINAL ESCAPES

The final control character recognized and processed by SMARTERM is CTRL-T. Sending this character signals SMARTERM that we wish to initiate a Terminal Escape. Terminal Escapes are functions that control SMARTERM's special hardware and software options. There are eleven distinct Terminal Escape functions processed by SMARTERM. These range from simple video mode switching commands to complex graphics functions. Terminal Escapes, like the GOTOXY command, are multi-state functions. All Terminal Escapes, with the exception of the reset function, are three state and require three characters. The reset function is two state, and requires only two characters.

## CURSOR MODE

We may, at our option, change the video image of SMARTERM's cursor. We can do this by using the cursor definition Terminal Escape. There are eight user selectable cursor definitions:

TABLE 2

### CURSOR DEFINITIONS

|                                 |                                 |
|---------------------------------|---------------------------------|
| 0: 3.75 HZ BLINKING FULL BLOCK  | 4: 1.875 HZ BLINKING HALF BLOCK |
| 1: " " " HALF "                 | 5: " " " QUARTER "              |
| 2: " " " UNDERLINE              | 6: " " " UNDERLINE              |
| 3: 1.875 HZ BLINKING FULL BLOCK | 7: BLANKED (BLACK) CURSOR       |

For example, to obtain a 3.75 HZ (normal blink speed) blinking underline, type (or output) the following:

CTRL-T C 2



Three characters must be sent: control T to signal a Terminal Escape, C for cursor mode, and the definition number, 0 through 7.

## VIDEO SOURCE SWITCHING

In the input features chapter we discussed the ability to send output to the APPLE standard 40 column screen (the dual video mode) simultaneously with SMARTERM's screen. Unless two monitors are hooked up to the APPLE, it would seem that viewing the 40 column output is somewhat an impossibility without physically changing video cables. This is where software video source switching comes in handy. This feature allows program control over who's video will be displayed on the monitor: SMARTERM's or APPLE's. There are two switching commands available:

- 1) Switch APPLE video to the monitor
- 2) Switch SMARTERM video to the monitor

As an example, to view standard 40 character APPLE video (text, LORES or HIRES graphics), type or [ouput] the following:

CTRL-T A 1

That is, control T for Terminal Escape, A for APPLE video, and finally the numeric 1 (actually, any character will do for the third character, we have chosen the numeric 1 for convenience).

To switch back to SMARTERM video, a similar request is sent:

CTRL-T B 1

The B stands for Board video, and the numeric 1 may be any character you wish.

In this way it is possible, under software control, to switch to any video mode desired using just one monitor. Of course, the dual video command must be issued (ESC V, as explained in the special input features section) before attempting to display standard APPLE 40 column text.

## SMARTERM VIDEO MODE SWITCHING

SMARTERM, like the APPLE ][, is capable of displaying more than one video mode. SMARTERM offers as an addition to the 80 column text mode, a dual function graphics mode. The graphics screen is switched in by issuing a video mode switch command (not to be confused with a video source switch command).

To make SMARTERM display its graphics screen, type (or output) the following:

CTRL-T G 1

This is also a Terminal Escape, and requires the customary CTRL-T initiator. G stands for Graphics, and the numeric 1, as stated before, can be any character you wish.

To obtain the text screen once again, type (or output) the following:

CTRL-T T 1

Where T stands for text.

#### SYSTEM RESET

During the course of normal SMARTERM use, there may arise the need to reset the Interface under software control. This reset of the Interface is unlike a BASIC PR#N or IN#N initialization in that only SMARTERM and its internal variables are reset; the APPLE system variables are not disturbed. The reset is invoked by typing (or outputting) the following:

CTRL-T R

As can be seen, this Terminal Escape is different from all the rest because it only requires two characters be sent to invoke the function.

All of the Terminal Escapes discussed up to this point have been hardware option oriented. The next section describes the use of SMARTERM's built in graphics primitives, which, although they utilize the hardware graphics screen option, fall under the category of software oriented Terminal Escapes.



SMARTERM GRAPHICS

The built in graphics capabilities of SMARTERM place it in an intelligent peripheral category all its own. SMARTERM provides you with a new medium resolution, black and white graphics mode with a screen resolution of 160 X 72 (horizontal X vertical). The graphics screen, unlike the text screen, is a first quadrant representation of a cartesian coordinate system. This places the origin (0,0) at the lower left corner of the screen. Positive X is to the right and positive Y is up. The coordinate system is represented by:

$$0 \leq X \leq 159 \quad (\text{DOMAIN})$$

$$0 \leq Y \leq 72 \quad (\text{RANGE})$$

Any combination of X and Y with values in the above domain and range is considered a valid coordinate. Each coordinate of the 11,520 possible coordinates is a separate pixel (dot). All pixels on the screen are approximately the same size; some are slightly larger or smaller than others (a similar phenomena found in APPLE ][ LORES graphics). Each pixel can assume a 0 or 1 state. These states visually correspond to black or white dots, respectively.

The firmware routines provided with SMARTERM to support the graphics screen are defined as "graphics primitives". They are termed "primitives" because they provide the lowest level of support (hence primitive) necessary to make the graphics mode usable.

You have at your disposal the following primitive functions:

- 1) Initialize: Initializes the graphics screen and sets it to black.
- 2) Background: Sets the screen to black or white.
- 3) Plot: Plots a single point in black or white.
- 4) Line: Plots a line (vector) in black or white.
- 5) Chgraf: Draws a graphic character at the current text position.

All of these primitives are invoked as Terminal Escapes. The plot and line primitives require parametric data (the desired X,Y coordinates) be stored in special memory locations prior to invoking their Terminal Escape sequence. In this way all graphics primitives are consistent in definition, and once the X and Y values for a desired point or line have been stored in memory, they will remain unaltered (regardless of how many times the graphics primitives are invoked) until a new set of coordinates is stored.

When performing the graphics primitives examples from the keyboard you will notice that the cursor will still be active on the graphics screen. To effectively remove the cursor from the screen during keyboard input mode, simply change its definition to "blank".

## INITIALIZING THE SCREEN

To initialize the graphics screen, type or output the following:

CTRL-T I 1

As with other previously defined three character Terminal Escapes, the numeric "1" character may be any character you wish it to be. The result of executing this function will be a cleared (black) screen.

## SETTING BACKGROUNDS

The graphics screen can be set to a black or white background to provide enhancement effects for graphic images. To obtain a background, type or output the following data:

CTRL-T S [0,1]

As can be seen, we have added a new twist to the above Terminal Escape. The third parameter in this three character sequence may take on one of two values (characters). Sending a 0 or a 1 results in a black or white background, respectively. The [ ] notation is used to denote a choice must be made for a parameter. When setting a white background, it may be necessary to adjust the contrast control on some monitors to keep the video image from smearing.

## PLOTTING POINTS

We now have a mechanism for setting backgrounds, so it is now appropriate to examine the point plotting capabilities of SMARTERM.

To plot a point on SMARTERM's graphic screen we must satisfy three conditions (parameters):

- 1) the X coordinate;
- 2) the Y coordinate;
- 3) the color (black or white).

Plotting points (and lines as we shall soon see) involves more "housekeeping" than other Terminal Escapes we have discussed. There are 4 special memory locations assigned as temporaries in the APPLE memory that are used by SMARTERM to hold the X,Y coordinate pairs during point and line plotting. These locations are:



| TABLE 3 |         |       |
|---------|---------|-------|
| PARAM   | DECIMAL | HEX   |
| X0:     | 1144    | \$478 |
| Y0:     | 1272    | \$4F8 |
| X1:     | 1400    | \$578 |
| Y1:     | 1528    | \$5F8 |

Before plotting a point, the X0 and Y0 coordinate pair must first be loaded with the desired values. This is done by using simple POKE statements from INTEGER or APPLESOFT BASIC. Pascal applications must use the procedure

```
DOXY(X,Y:INTEGER;SELECT:INTEGER);
```

provided in the chapter entitled 'SMARTERM Graphics for Pascal'. This procedure provides the necessary memory loading, and simplifies the task of direct memory poking from Pascal.

Values for X or Y that are out of range will result in the termination of any plot request. Once the required loading of X,Y values is complete, the following Termianl Escape is issued to plot the desired point:

```
CTRL-T P [0,1]
```

Bear in mind that all graphic commands issued to SMARTERM will be executed regardless of the current video mode. If 80 column text mode is currently selected, the result of a graphic command will usually be seen as meaningless text on the screen (often as printed control characters), and that to correctly view the results of a graphic command, one must remember to switch SMARTERM into the graphics display mode. The same is true when printing text during graphics display mode. All text will appear as combinations of graphic pixel blocks, and must not be confused as being block character graphic output.

The following APPLESOFT program demonstrates the plotting of random points on SMARTERM's medium resolution graphics screen, and is a good example of proper use of the plot function:

```

] 10 T$= CHR$(20) :REM CONTROL-T FOR TERMINAL ESCAPES
] 20 COLUR$= "1" :REM SET PLOT COLOR TO WHITE
] 30 HTAB 1 :VTAB 1 :PRINT T$;"G";"1";: REM SWITCH IN GRAPHICS
] 40 PRINT T$;"I";"1";: REM INITIALIZE GRAPHICS SCREEN
] 50 FOR I = 1 TO 500 :REM PLOT 500 POINTS
] 60 POKE 1144, RND(1)*160 :REM RANDOM X COORDINATE
] 70 POKE 1272, RND(1)*72 :REM RANDOM Y COORDINATE
] 80 PRINT T$;"P";COLUR$;: REM THIS PLOTS THE POINT
] 90 NEXT I

```

Notice in this example that there are semicolons at the end of all PRINT statements. This is necessary to prevent BASIC from issuing a carriage return to the screen. Semicolons must be placed at the end of all print

statements related to graphic commands; if they are not, there is a possibility that the screen will scroll, and portions of your graphic image will be lost. Notice also that none of the above PRINT statements cause the current print position to move from the initial HTAB 1 and VTAB 1 presets. This is because all Terminal Escape characters are digested internally and are not displayed (unless, of course, DEBUG mode is active).

## PLOTTING LINES

Plotting a line (vector) on the graphics screen is similar to plotting a point with the exception that a second coordinate pair (end point) is necessary to define a complete line segment. Generally, plotting lines is no more complex than plotting points. The only extra programming requirement is storing two more quantities (X1,Y1) in memory prior to invoking the line draw Terminal Escape. Enter and RUN the following APPLESOFT program to demonstrate the line plotting graphic primitive of SMARTERM:

```
] 10 T$=CHR$(20) :REM CONTROL-T FOR ALL TERMINAL ESCAPES
] 20 COLUR$="1" :REM SET PLOT COLOR TO WHITE
] 30 HTAB 1 :VTAB 1: PRINT T$;"G";"1";: REM SWITCH GRAPHICS IN
] 40 PRINT T$;"I";"1";: REM INITIALIZE GRAPHICS SCREEN
] 50 FOR I=1 TO 500 :REM PLOT 500 LINES (VECTORS)
] 60 POKE 1144,RND(1)*160: POKE 1272,RND(1)*72: REM X0,Y0
] 70 POKE 1400,RND(1)*160: POKE 1528,RND(1)*72: REM X1,Y1
] 80 PRINT T$;"L";COLUR$;: REM THIS PLOTS THE LINE
] 90 NEXT I
```

In comparing the above example to the previous for plotting points, notice that there are two differences. The first is in line numbers 60 and 70. It can be seen that there are 4 POKE statements representing 2 complete line segment endpoints: X0,Y0 and X1,Y1. In this example the values generated for these quantities will always be in the correct range. If any of the four parameters are out of range, then the line plot will be aborted. The second difference in the the above example is apparent in line number 80. Here, the letter "L" is used to indicate a line (vector) draw is being requested. The general form for the line draw Terminal Escape is:

CTRL-T L [0,1]

Like points, lines can be plotted in black or white.

Up to this point we have defined the use of [0,1] as the standard characters for determining the color of a background, point plot or line plot. For those programmers who are interested in such things, here is the reason why. SMARTERM actually strips all but the least significant bit off the third character sent to it in a background, point or line Terminal Escape. The least significant bit

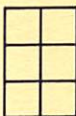


then determines if the graphic action to be taken shall be in black or white. Since the ASCII "0" and "1" characters actually correspond to their least significant bits, it is convenient to use them to represent black and white respectively. Actually, any ASCII character may be used for the third parameter. Consult an ASCII code chart for determining the least significant bit of a particular character. Character pairs such as B & C, D & E etc. provide results identical to the 0 & 1 pair.

## CHARACTER GRAPHICS

SMARTERM is capable of displaying block character graphics. Character graphics is a special operating mode that utilizes attributes from both the 80 column text and medium resolution graphic modes. Although character graphics is viewed in SMARTERM's graphic mode, its display matrix is identical to the text screen: an 80 X 24 (horizontal x vertical) grid (a total of 1,920 cells). Each separate cell is capable of displaying any one of 64 different graphic characters.

Table 4 contains the 64 block graphic characters along with the equivalent ASCII characters that translate into them. Each graphic character can be thought of as a 2 X 3 block of pixels (remember, a pixel is the fundamental graphic unit) as shown below:



6 pixels = 1 cell of the  
80 X 24 matrix

Since every pixel in a 2 X 3 block may be black or white, there are 64 distinct combinations, hence, a set of 64 graphic characters. Before plotting a graphic character, obtain the desired screen coordinate by issuing a GOTXY, or by issuing HTAB and VTAB commands. After positioning, use the following Terminal Escape sequence to plot a block graphic character:

CTRL-T D [any character in table 4]

For the third parameter, use the ASCII character in table 4 that corresponds to the graphic character you wish to plot. Since plotting block graphic characters is equivalent to printing normal text, there are a few rules to observe when using this feature:

- 1) The origin of the screen (0,0) is in the upper left corner (like the text mode), not the lower left corner.
- 2) Positioning on the screen prior to plot can be done through the use of HTAB and VTAB commands (BASIC) or GOTOXY commands (Pascal and BASIC).
- 3) Graphic characters are treated like regular text internal

































































to SMARTERM. Because of this, the current print position advances after every plot. Screen functions, such as carriage return and line feed, and scrolling of the screen will take place if the print position is allowed to increment past column 79 (absolute).

4) In general, this screen mode should be thought of as a special text screen: plotting graphic characters is identical to printing text characters on the text screen.



Table 4

## Translation Table for Block Character Graphics

| ASCII |       |                                                                                     | ASCII |        |                                                                                     | ASCII |           |                                                                                     |
|-------|-------|-------------------------------------------------------------------------------------|-------|--------|-------------------------------------------------------------------------------------|-------|-----------|-------------------------------------------------------------------------------------|
| 0     | SPACE |    | 22    | C      |    | 44    | q         |    |
| 1     | \$    |    | 23    | O      |    | 45    | u         |    |
| 2     | "     |    | 24    | CNTL-H |    | 46    | s         |    |
| 3     | &     |    | 25    | CNTL-L |    | 47    | w         |    |
| 4     | l     |    | 26    | CNTL-B |    | 48    | P         |    |
| 5     | %     |    | 27    | CNTL-F |    | 49    | T         |    |
| 6     | #     |    | 28    | CNTL-A |    | 50    | R         |    |
| 7     | '     |    | 29    | CNTL-E |    | 51    | V         |    |
| 8     | h     |    | 30    | CNTL-C |    | 52    | Q         |    |
| 9     | d     |    | 31    | CNTL-O |    | 53    | U         |    |
| 10    | b     |    | 32    | 0      |    | 54    | S         |    |
| 11    | f     |    | 33    | 4      |    | 55    | W         |    |
| 12    | a     |   | 34    | 2      |   | 56    | CNTL-P    |   |
| 13    | e     |  | 35    | 6      |  | 57    | CNTL-T    |  |
| 14    | c     |  | 36    | 1      |  | 58    | CNTL-R    |  |
| 15    | g     |  | 37    | 5      |  | 59    | CNTL-V    |  |
| 16    | @     |  | 38    | 3      |  | 60    | CNTL-Q    |  |
| 17    | D     |  | 39    | 7      |  | 61    | CNTL-S    |  |
| 18    | B     |  | 40    | p      |  | 62    | CHR\$(27) |  |
| 19    | F     |  | 41    | t      |  | 63    | CHR\$(31) |  |
| 20    | A     |  | 42    | r      |  |       |           |                                                                                     |
| 21    | E     |  | 43    | v      |  |       |           |                                                                                     |

SMARTERM GRAPHICS FOR PASCAL

In order for Pascal application programs to utilize the full graphics capabilities of SMARTERM, a machine language interface is needed to relieve the Pascal host procedure of performing the direct memory pokes that the vector graphics primitives expect. In this chapter, we present an assembly procedure that performs this simple function. It has been written for assembly under TLA (an acronym for The Last Assembler) which is provided with all Pascal language Systems. This assembly procedure, called DOXY, need only be entered once (as a text file) and should be saved on a backup diskette. After it has been correctly entered and saved, it must be A(ssembled). The resulting object file can then be referenced by any Pascal procedure utilizing SMARTERM's medium resolution graphics mode.

The procedure DOXY should appear in your Pascal programs in the following declaration form:

```
PROCEDURE DOXY (X,Y: INTEGER; SELECT: INTEGER);
EXTERNAL;
```

Note that DOXY is always passed with three arguments:

- 1) X: should be an integer in the range  $0 \leq X \leq 159$ .
- 2) Y: should also be an integer, in the range  $0 \leq Y \leq 71$ .
- 3) SELECT: this integer variable selects which coordinate pair to load:

```
0 (or any even integer) = X0,Y0
1 (or any odd integer) = X1,Y1
```

Don't forget to include the EXTERNAL; line when declaring procedure DOXY. Failing to do so will inevitably generate an error.

To prevent ambiguous situations from arising we ask you to save the textfile that contains the assembly procedure DOXY under the file name of SMTGRAF. All further references in this manual to the DOXY text and code files will be under the name of SMTGRAF.

Since the new procedure DOXY will be an entry into your Library of utility routines, all references to it from a Pascal program must be L(inked before the Pascal host procedure can execute. To do this, type L for L(ink from the system command level and a dialog similar to the following will take place:

Linking...

Linker II.1 [A4]

Host File? (Enter the name of your main Pascal procedure here)

Opening (Your program).code

Libfile? SMTGRAF (This is the routine we wish to link)



```

Opening SMTGRAF.code
Libfile?                (Press RETURN key for last file)
Mapname?                (Press RETURN key for no map file)
Reading (Your main program)
Reading DOXY
Output File? (You choose an appropriate name).code

```

At this point the final linked version of your program is ready. You may now X[ecute this final version of your program, and fully utilize SMARTERM's built in vector and character graphics capabilities.

The assembly language procedure DOXY follows. Please refer to the APPLE Pascal Preliminary Reference Manual chapter on assembly language interfacing in case any questions arise with respect to content and structure of this small program. Please type it in using the Pascal screen oriented editor exactly as it appears below:

```

.TITLE SMARTERM GRAPHICS X,Y VARIABLE LOADING
.PROC DOXY,3 ;3 WORDS OF PARAMETERS.
.PAGE
;*****
;
; THIS ASSEMBLY PROCEDURE STORES AN X,Y COORDINATE PAIR IN ONE
; OF THE FOLLOWING TWO MEMORY PAIRS (GIVEN IN HEX) ACCORDING TO
; THE SELECT VALUE:
;
;      SELECT                X      Y
;      0 (OR ANY EVEN INTEGER) $478  $4F8
;      1 (OR ANY ODD INTEGER)) $578  $5F8
;
;
;  PROCEDURE DOXY(X,Y: INTEGER; SELECT: INTEGER);
;
;*****
RETURN .EQU 0                ;PASCAL RETURN ADDRESS TEMPORARY.
XLOC0 .EQU 478              ;X LOCATION 0.
YLOC0 .EQU 4F8              ;Y LOCATION 0.
XLOC1 .EQU 578              ;X LOCATION 1.
YLOC1 .EQU 5F8              ;Y LOCATION 1.
PLA                          ;SAVE PASCAL RETURN ADDRESS.
STA RETURN
PLA
STA RETURN+1
PLA                          ;POP PARAMETERS, LAST FIRST.
LSR A                       ;CARRY WILL DETERMINE MEM PAIR SELECT.
PLA                          ;DISCARD HIGH BYTE.
PLA                          ;GET LOW BYTE OF Y.
BCS $010                    ;BRANCH IF Y1 SELECTED.
STA YLOC0                   ;ELSE Y0 SELECTED, PUT Y AWAY.
BCC $020                    ;BR ALWAYS.
$010 STA YLOC1              ;Y1 IS SELECTED, PUT IT THERE.

```

```

$020  PLA          ;DISCARD HIGH BYTE OF Y.
      PLA          ;GET LOW BYTE OF X.
      BCS $030     ;BR IF X1 SELECTED.
      STA XLOC0    ;X0 SELECTED, PUT AWAY X.
      BCC $040     ;BR ALWAYS.
$030  STA XLOC1    ;X1 WAS SELECTED, PUT X AWAY.
$040  PLA          ;DISCARD HIGH BYTE OF X
      LDA RETURN+1 ;PUSH THE PASCAL RETURN ADDRESS
      PHA          ;BACK ON THE STACK.
      LDA RETURN
      PHA
      RTS          ;RETURN TO PASCAL.
      .END

```

After entering the above assembly procedure using the Pascal screen oriented editor, save the file as SMTGRAF.TEXT on more than one diskette. This will save you the drudgery of reentering the above code in the event that one of your diskettes is trashed, lost, or whatever. Assemble the SMTGRAF text file to produce an object code file for Linking to any Pascal program that will use DOXY. It is generally a good idea to store the object code on more than one diskette as well for the same reasons cited above, and one of them should be the diskette that contains your standard library procedures and functions most often used in the programs you write.



BASIC UPDATES, and USING SMARTERM with Pascal 1.1 and CPM

For users of Pascal (1.0 & 1.1) and CPM, one thing that cannot be done is the execution of Terminal Escapes from the keyboard. Terminal Escapes must be invoked by outputting them.

When running under Pascal or CPM, notice that the cursor is updated during output, and not during input as with both BASIC's. This more closely emulates the cursor function of many popular terminals on the market. It also allows you, the user, to replace SMARTERM's keyboard code with that of your own design while still maintaining a cursor on the screen.

An exclusive feature not available on any other 80 column product is the Pascal 1.1 keypress and typeahead support. This feature is only supported for Pascal 1.1.

CPM warm (CTRL-C) boots will cause the screen to clear, since CPM in effect is asking SMARTERM to cold start itself. Make sure any important screen information is written down before warm booting since the current screen will be destroyed.

When using SMARTERM with BASIC, you must convert all HOME and CALL -936 commands to PRINT CHR\$(12). This will print a CTRL-L to SMARTERM and will generate a HOME and CLEAR SCREEN sequence.

After exiting SMARTERM, remember to type TEXT (and hit the RETURN key) to insure a full 24 line text window.

## REFERENCE LIST OF FEATURES

This chapter is a synopsis of the features provided by SMARTERM. It should be used as a quick reference guide. The prior chapters of this manual explain in complete detail all of the features supported by this Intelligent Interface. Please refer to them for a detailed description of a particular feature in question. The discussion of SMARTERM's features in the previous chapters are generally accompanied by programming examples to show typical and correct use.

### KEYBOARD INPUT FEATURES

The following five features are obtained by holding down the control key while simultaneously typing the desired character.

#### CTRL-A

Control A provides two functions:

- 1) The keyboard is down shifted to provide lower case characters if it is currently in the capitals (shift lock) mode.
- 2) If the keyboard is currently in down shift (lower case) mode, typing a control A will result in the next alpha being capitalized.

#### CTRL-K

Control K translates into the "[" character.

#### CTRL-N

Control N is the lead in character for the NEW KEYS (\ \_ ` { | } ~ DEL ] ^ @) which are translated from the 0 1 2 3 4 5 6 7 8 9 : keys, respectively. Control N is also the lead in for the VERBATIM mode which passes any character following it VERBATIM to the system.

#### CTRL-U

This is the "screen pick" or screen read feature. Both the forward arrow and control U characters invoke this feature. It is not available to Pascal.

#### CTRL-V

Control V provides the "one wire shift-mod" feature. When activated, it provides real SHIFT key operation. Also, the "]" , "^" and "@" characters are translated into M, N and P respectively. This provides true shifted M, N and P characters when typing in lower case mode. CTRL-V is a toggled feature.



## CTRL-Z

Control Z locks the keyboard to upper case.

## ESCAPE FEATURES

ESCAPE features are obtained by pressing and releasing the ESC key before the desired character. ESCAPE functions are not available to Pascal, or any interpretive language environment written in Pascal. The ESC G through ESC N features are auto repeating. That is, once the ESC key has been pressed, the keys G through N may be pressed any number of times to obtain multiple step cursor movement.

ESC @: Homes the cursor and clears the screen.  
 ESC A: Moves the cursor right one character column.  
 ESC B: Moves the cursor left one character column.  
 ESC C: Moves the cursor down one character row.  
 ESC D: Moves the cursor up one character row.  
 ESC E: Clears from the current column to the end of the line.  
 ESC F: Clears from the current column to the end of the screen.  
 ESC G: Tabs the cursor forward to the next column multiple of 8.  
 ESC H: Tabs the cursor backward to the previous column multiple of 8.  
 ESC I: Moves the cursor up one character row.  
 ESC J: Moves the cursor left one character column.  
 ESC K: Moves the cursor right one character column.  
 ESC L: Tabs the cursor down to the next row multiple of 4.  
 ESC M: Moves the cursor down one character row.  
 ESC N: Tabs the cursor up to the previous row multiple of 4.

## ESC CTRL-Q

This is the SMARTERM exit function. It performs an automatic PR#0,IN#0 and switches the APPLE standard 40 column video to the monitor. One should type TEXT (and hit the return key) after exiting SMARTERM to insure a full 24 line text window.

## ESC V

This escape code toggles the APPLE simultaneous video feature. Output goes both to SMARTERM and to the APPLE standard 40 column screen. Either output may be viewed on the primary monitor by activating the appropriate video switch command.

## OUTPUT FEATURES

The following is a list of features that are processed during character output. Most features can be activated by typing them from the keyboard, because the APPLE normally echos all keyboard input to the screen. This is not true however, when operating from Pascal. Pascal usually filters all input (ignoring what it does not understand) before echoing it to the screen. In either case, all output features can be utilized from any language environment by simply PRINTing the appropriate ASCII codes to the screen. It has been found through research at Advanced Logic Systems, Inc. that by imbedding function control codes in normal PRINT statements, overall system output performance is not appreciably reduced. In fact, the use of printed commands to SMARTERM in most cases provides a small increase in output processing performance. This increase is due to efficient use of processor time allocation. Simple hardware switching tasks, graphics primitives, etc. would take slightly longer to execute in the host language because of increased system overhead requirements.

## RUN TIME FEATURES

Run time features are not printed to the screen. They may be activated only when the system is sending output to SMARTERM.

## CTRL-F

This is the FLUSH output feature. It is typed from the keyboard during program output. This causes all output to go undisplayed until:

- A) CTRL-F is typed once again, or
- B) Input is requested from the keyboard, causing an auto-cancel of FLUSH.

## CTRL-S

This is the STOP list or SUSPEND output feature. If it is typed during program output, program execution is suspended. Normal system operation will resume after the user hits a second key (any key other than SHIFT, CTRL or RESET). This feature is very useful for reading screen data that is being displayed faster than one can read.

## CONTROL CHARACTERS

Control characters are those that usually do not get displayed. Instead they are internally processed, and provide control over the total screen environment. There is a case that does provide for the actual displaying of control characters sent to SMARTERM. This is known



as the DEBUG mode of operation, and is explained in full detail under the section entitled "THE DEBUG MODE" in the output features chapter.

CTRL-E: Tabs down to the next row that is a multiple of 4.  
 CTRL-F: Tabs up to the previous row that is a multiple of 4.  
 CTRL-G: Generates a bell by beeping a 1KHZ tone on the speaker.  
 CTRL-H: Back spaces the current print position one column.  
 CTRL-I: Tabs forward to the next column that is a multiple of 8.  
 CTRL-J: Linefeed. Moves the print position down one row.  
 CTRL-K: Clears from the current column to the end of the screen.  
 CTRL-L: Sends the print position home and clears the screen.  
 CTRL-M: Carriage return. Resets the column position to 0.  
 CTRL-R: Tabs back to the previous column that is a multiple of 8.  
 CTRL-T: Initiates a Terminal Escape request.  
 CTRL-W: Toggles the debug mode. Prints all control characters.  
 CTRL-Y: Home. Resets print position to the upper left corner (0,0).  
 CTRL-Z: Clears the current row.  
 CTRL-\: Moves the print position to the next column.  
 CTRL-]: Clears from the current column to the end of the line.  
 CTRL-^: Initiates a GOTXY.  
 CTRL-\_: Reverse linefeed. Moves up one row.

#### TERMINAL ESCAPE FUNCTIONS

The following is a list of the 11 available Terminal Escape functions. All functions (with the exception of RESET) are defined by a two character code, and are preceded by a CTRL-T (Terminal Escape initiator) resulting in a three character sequence. All of the following definitions assume a leading CTRL-T initiator. The [...] sequence found in some of the definitions signifies a choice is available for the third character. Some functions, however, are implicitly defined within their second character; this allows the use of any printable character for the third. These functions do not actually use the third character, but they must be sent just the same. We suggest the ASCII numeric 1 be used as the third character in such a case (this is a practice adopted by Advanced Logic Systems, Inc. to provide program consistency and to help avoid confusion).

A1: Switches the APPLE 40 column video display to the monitor.  
 B1: Switches the SMARTERM 80 column video display to the monitor.  
 C[0-7]: Changes the cursor definition. See table 2 for the eight different cursor definitions.  
 D[ ]: Outputs a graphic character (see table 4 for the 64 definitions) at the current print position.

- G1: Switches SMARTERM into the 160 X 72 graphics mode.
- I1: Initializes the SMARTERM graphics screen, and clears the background to black.
- L[0,1]: Plots a vector from X0,Y0 to X1,Y1 on the graphics screen in black or white. The color depends on the third character:
- A) 0 results in a black plot,
  - B) 1 results in a white plot.

The four memory locations that must be loaded with X0,Y0 and X1,Y1 before initiating this Terminal Escape are:

|    | DEC  | HEX   |    | DEC  | HEX   |
|----|------|-------|----|------|-------|
| X0 | 1144 | \$478 | X1 | 1400 | \$578 |
| Y1 | 1272 | \$4F8 | Y1 | 1528 | \$5F8 |

- P[0,1]: Plots a single point in the color chosen at X0,Y0. The X0,Y0 coordinate pair must be set up prior to initiating the plot request. The X1,Y1 pair is not used.
- R: Resets SMARTERM to its initial (cold start) state. This Terminal Escape does not require a third character.
- S[0,1]: Sets the graphics screen to black or white: 0:= black, 1:= white. This is the "Background" command.
- T1: Switches SMARTERM into the 80 column text mode.



DIAGNOSTIC TROUBLESHOOTING

This chapter is included to help you diagnose common errors made during the installation and normal use of your SMARTERM Intelligent Interface. Please read this chapter if you cannot obtain a proper output from SMARTERM. If after trying all of the suggested fixes your SMARTERM still fails to function, please bring it to the dealer it was purchased from. The dealer may elect at his option to service it himself or send it back to Advanced Logic Systems, Inc. for repair.

In this chapter, references are made to a control on both the APPLE motherboard and the SMARTERM Interface card. It is called a "pot" and is short for potentiometer. This control is easily recognized, as it is a small, black (or blue) rotating knob. This is the only moving control found on the SMARTERM Intelligent Interface card, and is located next to J1. The pot on the APPLE motherboard can be found to the right of slot #7 as you look into the system with the keyboard facing you.

NO VIDEO

This is obviously the most severe problem that can occur, so we have given it the highest priority in the list. We begin the diagnosis by asking a few basic questions. Is your APPLE ][ plugged in? Is it turned on? Is the monitor also plugged in and turned on? If so, open the APPLE case and inspect the SMARTERM printed circuit board. Is it seated firmly in its socket? If not, turn the APPLE off before reseating the Interface. Is there a video cable connecting the APPLE video out to the upper (J1) video jack on SMARTERM? There should also be a video cable connected from the lower (J2) video jack on SMARTERM to the monitor being used. Is the video output level high enough to produce any output? Make sure the video level pot on the upper edge of the SMARTERM p.c. board is turned at or near maximum to provide a strong video signal to the monitor.

WEAK OR LOW CONTRAST VIDEO

Check the video level pots on both the APPLE motherboard and the SMARTERM Interface card. The video level pot on the APPLE motherboard should be close to full clockwise travel for proper APPLE video. The video level pot on SMARTERM should be adjusted for a bright video image on the monitor. Check the contrast and brightness controls on the monitor. We suggest near maximum brightness and near minimum contrast for a quality, smear free image. If necessary, adjust the video level pot on the SMARTERM Interface for a bright image. A good output on the monitor is evident when horizontal bars and separate dots within character definitions are approximately equal in brightness.

Since the SMARTERM video level pot and the contrast and brightness controls of the monitor all interact with each other, one should try "fine tuning" the controls as described above to obtain a satisfactory display.

### ROLLING DISPLAY - HORIZONTALLY OR VERTICALLY

Make sure the video output levels of both the SMARTERM Interface and the APPLE are high enough. Adjust the horizontal hold control of the monitor until a centered or near centered display results with no horizontal rolling. Adjust the vertical hold control on the monitor until the same results can be achieved vertically.

### BENT OR TORN DISPLAY

A bent or torn display is usually seen in the upper left corner of the screen. Correct the problem by adjusting the horizontal hold on the monitor for a proper display. Make sure the video level of SMARTERM is high enough (adjust the video level pot on the Interface card if necessary) to prevent this problem from arising.

### APPLE VIDEO BUT NO SMARTERM VIDEO

Are you issuing the correct PR#, or IN# command to SMARTERM? Double check the slot number that SMARTERM is plugged into. Remember, whenever power is first applied to the APPLE ][, or the RESET key is pressed, SMARTERM switches its video output to display normal APPLE video. If SMARTERM still won't produce a display, turn off the APPLE and remove the Interface card. Using moderate pressure push down on all of the I.C.'s (Integrated Circuits) to insure none have worked loose in their sockets from mechanical vibration. Loose I.C.'s are a common cause of system trouble, and long term thermal cycling can have a similar effect of loosening them much the same as mechanical vibration. Also, clean the gold "fingers" of the Interface card with a new pencil eraser to insure good electrical contact with the socket when the card is plugged in. Install the Interface card back into one of the slots and insure it is seated with a snug and firm fit.

### ERRATIC OPERATION OF SMARTERM

Turn off the power to the APPLE and remove the Interface card. Clean the gold "fingers" portion of the card with a new pencil eraser to insure good electrical contact when it is plugged back into one of the slots. Avoid touching the "fingers" portion of the card because natural



skin oils tend to cause dirt and dust to accumulate on them, resulting in poor electrical contact. Also, apply an even, firm pressure on all of the I.C.'s on the Interface card to insure none are loose. Install the card back into one of the slots, checking for proper seating and a snug, firm fit.

**Advanced Logic Systems, Inc.**

1026 West Maude Ave., Suite 305  
Sunnyvale, CA 94086



# Expand Your Apple

## SMARTERM CHARACTER ENHANCEMENT INSTRUCTIONS

This sheet provides the instructions for installing the Smarterm Character Enhancement, as well as the documentation changes necessary to update the standard Smarterm manual. The kit contains only two parts, an 8002 character generator and an EPROM with a small sticker on it marked "2.0".

### INSTALLATION

Carefully follow these step-by-step instructions:

1. TURN OFF POWER TO THE APPLE ][[[
2. Open the cover, and remove the Smarterm
3. Carefully pry the 24 pin device marked 8002-003 from its socket and press it into the foam that the kit includes
4. Remove the 8002A from the foam, line up the notch with the notch painted on the circuit card, align the 24 pins with the socket and firmly press the device into the socket
5. Carefully pry the ROM or EPROM from its socket. This is the 24 pin device next to the character generator, marked 9316B or with a small sticker with "1.0" or "1.1". Press it into the foam.
6. Remove the new EPROM from the foam, align its notch and pins, and press it into its socket
7. Reinstall the Smarterm as described in the Smarterm manual, page 5

### VIDEO ADJUSTMENT PROCEDURE

The new character set generates much more vertical video information, due to the 7x11 dot matrix. Therefore, it will be necessary to adjust the vertical size and linearity controls on your monitor. These controls are a small knob or a small hole through which an adjusting screwdriver can be inserted either on the front or rear of the monitor. The procedure is based on creating a pleasing display. Turn the vertical size (perhaps labelled "height") counterclockwise to reduce the vertical display size enough to put all 24 lines of information on the screen.

If, after the above adjustment, you notice an unsatisfactory difference in size between the top and bottom lines on the screen, adjust the vertical linearity control A VERY SMALL AMOUNT. If necessary readjust size to fill the screen. Repeat these two adjustments until the screen is full with 24 lines displayed and the linearity of the display is satisfactory.



## SMARTERM FUNCTIONS -

All functions of the Smarterm are identical to the descriptions in the Smarterm manual. Keyboard controls have been enhanced for application to a wider variety of programs and especially for use with CP/M based word processing and business applications. The firmware assumes that the shiftkey modification has been installed when booting CP/M or Pascal diskettes, but maintains a provision for a single key shift if that is not the case. The special keys and their functions are as follows:

1. CTRL-SHIFT-P (press all three characters at once) is used as a shift key only if the shift wire has not been installed.
2. CTRL-SHIFT-M (press all three characters at once) provides a shift lock/unlock function.
3. "Literal" means that the Smarterm will not process the next character typed. Under CP/M and Pascal, the literal key is ESC (escape). Thus, to pass ESC to the program, press it twice. (See para 6 below for extra functions.)
4. In Apple DOS, CTRL-Q is used as the literal key, and ESC functions exactly as described in the Apple documentation. Thus, to switch from 80 column to 40 column, press ESC then CTRL-Q then CTRL-Q.
5. The ESCape function is passed to the system by pressing ESC two times.
6. Pressing ESC (CTRL-Q in Apple DOS) provides 12 extra keys, as follows:

|                     |   |   |   |    |   |   |   |   |   |   |   |   |
|---------------------|---|---|---|----|---|---|---|---|---|---|---|---|
| Escape then numeral | _ | ` | { |    | } | ~ | ^ | [ | ] | \ | @ | d |
| Shifted key         |   | " | # | \$ | % | & | ' | ( | ) | 0 | * | = |
| Unshifted key       | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9 | 0 | : | - |

d = DEL

7. CTRL-O sent to the screen will turn the inverse mode on, and CTRL-N will turn it off. Since CP/M "echoes" all characters to the screen, pressing these keys at the system level (A> on the screen) will turn inverse on and off. Since these are usually program controlled functions, this function can appear as a surprise.
8. In the event that RESET is pressed in error or for any reason the code to enable graphics is sent to the screen, the sequence CTRL-T B 1 (cr) will restart the 80 column mode.
9. CTRL-S is the stop list feature, to interrupt any listing to screen or printer. A stopped listing will wait for any other character to be pressed.
10. The CTRL-F (FLUSH) command is unnecessary and has been deleted.
11. The graphics mode is now a 160 x 96. (See the Smarterm manual, page 29)