U L T R A T E R M

ORIGINAL EQUIPMENT MANUFACTURER

PROGRAMMING GUIDELINES

JANUARY 1984

1105 N.E. Circle Blvd.
Corvallis, Oregon
(503) 758-0521

ULTRATERM SOFTWARE GUIDELINES

This document has been prepared to assist programmers to develop software for the UltraTerm. It briefly covers the UltraTerm's CRTC registers and memory usage and lists screen drivers for controlling the card's display. The information presented here is primarily limited to that necessary to understand and implement the UltraTerm screen drivers. Using these assembly-language routines is the easiest way to control the UltraTerm's screen display outside of its ROM routines. Complete information on the card's operation is contained in the UltraTerm Installation and Operation Manual. Page references is this document refer to the UltraTerm manual.

Note. In this document, the first reference to an address may be given as $C0B2 + n0 or $6F8 + n, where n is the slot number in which the UltraTerm resides. In all examples, the card is assumed to be in slot 3, the recommended slot for the UltraTerm.

CRTC Registers (Appendix C, page C.1 - C.3)

$C080 + n0 : CRTC Register Selection Port. Writing a CRTC register number to this address selects that register.

$C081 + n0 : CRTC Register Data Port. The data for the selected CRTC register is written to this address.

For the most part, you may never need to change the CRTC registers. However, as it is easy to change the UltraTerm's cursor and doing so shows how to modify any register, the cursor registers will here be discussed and manipulated. For complete, detailed information on the CRTC, refer to the Hitachi Microcomputer Data Book.

Bits 5 and 6 of Register 10, Cursor start, can be set or clear in four combinations resulting in four possible cursor types: fast-, slow-, or non-flashing, or no visible cursor at all. The four bit combinations are listed below with the hexadecimal and decimal conversions and the resulting cursor types.

BINARY	HEXADECIMAL	DECIMAL	CURSOR TYPE
000 ____	0 + x	0 + x	non-flashing
001 ____	20 + x	32 + x	not visible
010 ____	40 + x	64 + x	fast flashing
011 ____	60 + x	96 + x	slow flashing

In the above table, the values of the first four bits are left blank and their combined value is indicated by an 'x' in the conversions. These bits determine the number of lines which will make up the displayed cursor. (Think of the lines as being numbered from 0 to M, where Line 0 is the top line of the cursor cell and M is the maximum number of displayable lines in

the cell. M is determined by the display mode selected; for the non-interlaced modes, M = 8, for the interlaced display modes, M = 16.) The value of x in the Cursor start register determines the top-most displayed line of the cursor. If x = 0, a full-sized cursor is displayed. In a non-interlaced display mode, an x value of 8 produces an underscore cursor (lines 0-7 will not be displayed). When changing register 10, be aware that two different aspects of the cursor are affected. For example, if a slow-flashing, full sized cursor is to become a slow-flashing, underscore cursor, poking in a value of 8 will not alone suffice, bits 5 and 6 must also be set.

In Register 11, Cursor end, only the first four bits are of concern. The value in this register determines the bottom-most displayed line of the cursor. A value of 4 in Register 11 produces a four-line cursor in the upper half of the cursor cell, all else being equal. The value in Register 11 must be less than or equal to the value of x in Register 10 or no cursor will be displayed. If the values are equal, a single-line cursor will be displayed.

Note. While the cursor in the interlaced display modes can have 16 lines, the default cursor in these modes has only 12 lines.

To change the standard, slow-flashing cursor of the Videoterm emulation mode to a solid cursor, do the following.

From the monitor:

```
* C0B0: A          ; select Register 10
* C0B1: 0          ; write new value to selected register
```

This may also be entered as

```
* C0B0: A 0
```

Form Applesoft:

```
] POKE -16208,10 : POKE -16207,0     :REM SELECT CRTC REGISTER 10 AND
                                         POKE THE VALUE 0 THERE
```

To change this to a fast-flashing, underscore cursor, enter

```
* C0B0: A 48
```

or

```
] POKE -16208,10 : POKE -16207,72
```

Switch to the 80x24, interlaced, mode. The cursor will be full-sized again. Changing display formats will reset the cursor to the default settings. To increase the length of the cursor to fill the character cell, enter

```
* C0B0: B F
```

or decrease it to a one-line overscore by entering

```
] POKE -16208,11 : POKE -16207,0
```

To produce a slow-flashing, dash cursor, enter

```
* C0B0: A 64          or          ] POKE -16208,10 : POKE -16207,100
* C0B0: B  4                      ] POKE -16208,11 : POKE -16207,4
```

Note. Under MBASIC, the $C000 range is moved to $E000. The CRTC register selection and data addresses become $E0B0 and $E0B1 (-8016 and -8015), respectively.

Besides the CRTC registers, two other registers are used to control the UltraTerm's display. These are the Mode Control Register and the Video Attribute Register.

$C082 + n0 : Mode Control Register

This port is used to set the operating mode of the UltraTerm. The bits of this register determine whether the Apple or UltraTerm video is displayed, what range of memory can be read, which clock rate is used, and how the screen RAM can be accessed.

Bit 7. Memory read select.
0 - Read screen memory ($CC00-$CDFF)
1 - Read second page of ROM ($CC00-$CFDF)

The UltraTerm code is stored in ROM from $C800 to $CFDF (the range $CFE0-$CFFF is not used). This range is divided into two pages: $C800-$CBFF and $CC00-$CFDF. The second page, $CC00-$CFDF, is paged in on top of the Video Refresh Memory. With Bit 7 set, a READ accesses the second page of ROM; with it clear, a READ accesses the screen memory. A WRITE to this range in either case writes to the screen memory. Since most machine language programs will be written to bypass the ROM routines, Bit 7 should normally be set to 0. For more on the firmware interface and VRM see Appendix Y of the UltraTerm manual.

Bit 6. Video select.
0 - Display Apple video (40 columns)
1 - Display UltraTerm video

Bit 5. Clock select (Page Y.4, Section Y.6).
0 - Select 80-column clock
1 - Select 132-column clock

Bit 4. Addressing mode select.
0 - Block addressing, 512 byte (Videoterm emulation)
1 - Page addressing, 256 byte

Bit 4 is cleared when format 0 is selected, set when formats 1-7 are selected. In Videoterm emulation mode the entire range of screen RAM ($CC00-$CDFF) is used. The two pages of RAM are available in each of four banks: $C080, $C084, $C088, and $C08C (+ n0). UltraTerm mode uses only the first page of RAM ($CC00-$CCFF). There are sixteen banks of one page each. Memory access is much simpler and quicker in this mode. Therefore, if you intend to use format 0, you should invert this bit, setting the 256-byte page mode.

Note that if you mix your own software which uses the 256-byte page mode with our firmware for format 0, which uses 512-byte block addressing, you will get very strange results if the firmware scrolls the screen. We strongly recommend that you do not mix your software with our firmware in format 0 unless you use the 512-byte block addressing. If you want to maintain complete Videoterm compatibility in your software, you must use the 512-byte block mode. In this case you will neither need to write to the Mode Control Register after you initialize the UltraTerm nor need the MODE value.

Mode Control Register, continued.

Bits 0-3. Page select.

When in the 256-byte page address mode (Bit 4 set), these bits correspond to the character RAM address bits 8-11. In the Videoterm emulation mode (Bit 4 clear), these bits have no effect.

$C083 + n0 : Video Attribute Register

This register is used to set the display attributes for the characters stored in the display RAM. Each character may be displayed on the screen with one of two sets of attributes. One set, Inverse or Highlight, is selected if the high bit of the character in RAM is set, the other, Normal or Lowlight, if the high bit is clear. The Attribute register also determines which character font, Standard or High Density, will be used for display.

Bit 7. 0 - Select attributes as defined by Bits 5, 4, 1, 0
1 - Select attributes as defined by DIP switches (default)

This bit is always looked at. When set, Bits 5, 4, 1, and 0 will be ignored, the display attributes being determined by the DIP switches. The DIP switches on the UltraTerm allow the user to select his or her own combination of highlight/lowlight and normal/inverse attributes (see Section 5.d, page 5.6). Generally, this bit is not changed from the default so the user's choice of is maintained.

Bit 6. Unused.

Bit 5. 0 - Normal video if bit 7 of character = 1
1 - Inverse video if bit 7 of character = 1

Bit 4. 0 - Lowlight video if bit 7 of character = 1
1 - Highlight video if bit 7 of character = 1

Bit 3. Unused.

Bit 2. 0 - Use High-Density character set
1 - Use Standard charcter set

This bit is always looked at. It is not affected by the other bits of this register. The character set that you select must be chosen in concert with the display format or you may either cut off the bottom of the characters or have extra spaces between the lines making up the characters.

Bit 1. 0 - Normal video if bit 7 of character = 0
1 - Inverse video if bit 7 of character = 0

Bit 0. 0 - Lowlight video if bit 7 of character = 0
1 - Highlight video if bit 7 of character = 0

UltraTerm Variable and Permanent Memory Locations

The UltraTerm uses variable locations in the Apple screen memory area to maintain certain temporary values for its own use. These variables are guaranteed to be valid only immediately after you exit from our firmware I/O and initialization routines. Other peripheral cards may alter the values in these locations when they execute their own I/O routines. The temporary variables are defined below.

$0478	MODE	Mode mask for the Mode Control Port. It can be used to get upper four bits for the MCP. You should use one of the initialization routines described later, then fetch the MODE value and store it in a location unique to your program.
$04F8	HEIGHT	Displayed screen height.
$0578	SWDTH	Displayed screen width.
$05F8	PWDTH	Printed screen width. This variable is necessary because multiples of 16 must be used to scroll properly but the 132-column width is not a multiple of 16. The screen width for this format is actually 160 while the printed width is 132.
$0678	OLDCHAR	Used for lowercase correction in Apple][and][+.
$06F8	NO	Slot * $10. Used to index device select.
$0778	TEMPX	General temporary usage throughout code.
$07F8	MSLOT	Slot + $C0. Used for indexing permanents. Must be set before entering $C800 code for interrupt protocol to be maintained. If a program causes an interrupt, it will know how to reselect the proper $C800 area.

The following permanent variables are used by the UltraTerm. These values should not be altered unless you intend to affect the standard operation of the card.

$0478 + n	BASEL	Screen base address.
$04F8 + n	BASEH	Screen base address.
$0578 + n	CHORZ	UltraTerm cursor horizontal position.
$05F8 + n	CVERT	UltraTerm cursor vertical position.
$0678 + n	BYTE	I/O byte.
$06F8 + n	START	Top-of-screen address/$10.
$0778 + n	POFF	Power-off byte and state code.
	Bits 0-2	Output state code.
	Bits 3-7	Contains 0011 0xxx after initialization.
$07F8 + n	FLAGS	General-purpose flags.
	Bits 0-2	Display format number.
	Bits 3-7	Internal usage.

UltraTerm Format Selection (Page F.47)

The UltraTerm has eight different screen formats. The format numbers used in the screen drivers to set the display mode are offset by one from the numbers used with the firmware commands, CTRL-V or ESC. This is because the firmware must have the additional capability to accept a command to switch to the Apple 40-column video signal. The display modes, firmware commands, and the format numbers are listed in the following table.

Display mode	CTRL-V or ESC Command	Format Number
Apple video	0	None
80 x 24	1	0
96 x 24	2	1
160 x 24	3	2
80 x 24 int	4	3
80 x 32 int	5	4
80 x 48 int	6	5
132 x 24 int	7	6
128 x 32 int	8	7

('int' denotes a mode using interlace, requiring a monitor with high persistence phosphor.)

UltraTerm Screen Drivers

Routines to initialize the UltraTerm and drive its screen display are listed in the UltraTerm manual (Appendix F, Section F.3, pages F.47-51). A modified version of those routines is listed on the following pages. Listed below are the main differences between the two versions, where 'Original' is the manual version and 'Revised' is the version in this document.

Original	No check for nor support of the Videoterm mode (format 0) Slot independent; slot number passed to INIT routine Display format number passed to the INIT routine
Revised	Supports the card in Videoterm emulation mode Slot 3 dependent; recommended slot for the UltraTerm Display format determined by CHKFORM routine using tables

The screen drivers listed in this document will be used in the programs on the UltraTerm Utilities Disc. We therefore feel these routines may be useful to you.

```
                        1   ********************************
                        2   *                              *
                        3   *   STANDARD SCREEN DRIVERS     *
                        4   *   WITH VIDEOTERM EMULATION    *
                        5   *      FOR THE ULTRATERM        *
                        6   *       COPYRIGHT 1983         *
                        7   *          BY VIDEX            *
                        8   *                              *
                        9   *          01/02/84            *
                        10  *                              *
                        11  ********************************
                        12  *
                        13  *
                        14  YSAVE    EQU  $34         ; LOCATION USED TO PRESERVE Y
                        15  CH       EQU  36          ; SCREEN HORIZONTAL
                        16  CV       EQU  37          ; SCREEN VERTICAL
                        17  *
                        18  BASL     EQU  $08
                        19  BASH     EQU  $09
                        20  *
                        21  KBD      EQU  $C000       ; KEYBOARD BUFFER
                        22  KBDSTRB  EQU  $C010       ; KEYBOARD STROBE
                        23  *
                        24  * BASIC TOKEN EQUATES
                        25  *
                        26  NO       EQU  $6F8
                        27  MODE     EQU  $0478
                        28  *
                        29  START    EQU  $06FB       ;START ADDRESS (SLOT 3)
                        30  FLAGS    EQU  $07FB       ;PERMANENT FLAGS (SLOT 3)
                        31  * SLOT   EQU  $07FB
                        32  *
                        33  DEV0     EQU  $C0B0
                        34  DEV1     EQU  $C0B1
                        35  DEV2     EQU  $C0B2
                        36  DEV3     EQU  $C0B3
                        37  DISP0    EQU  $CC00
                        38  *
                        39  *
                        40  * CHKFORM WILL FIND OUT WHAT FORMAT THE ULTRATERM IS IN
                        41  * AND STORE THE NUMBER OF LINES AND THE NUMBER OF COLUMNS
                        42  * FOR FUTURE REFERENCE IN YOUR OWN MACHINE LANGUAGE
                        43  * ROUTINES.  THIS SHOULD BE FIRST ROUTINE CALLED.
                        44  *
                        45  CHKFORM
8000: 2C 00 C3          46          BIT  $C300       ;TURN ON SCREEN IN CASE APPLESOFT
                        47  *                        ;WROTE TO #CFXX
8003: AD FB 07          48          LDA  FLAGS
8006: 29 07             49          AND  #$07
8008: 8D 24 81          50          STA  FORMAT
800B: A8                51          TAY
800C: B9 0A 81          52          LDA  MODETBL,Y
800F: 8D 27 81          53          STA  MODEMASK
8012: B9 12 81          54          LDA  HIGHTBL,Y
8015: 8D 23 81          55          STA  HIGH        ; THE NUMBER OF LINES ON SCREEN
8018: B9 1A 81          56          LDA  WIDETBL,Y
801B: 8D 22 81          57          STA  WIDE        ; THE NUMBER OF COLUMNS ON SCREEN
801E: A0 00             58          LDY  #$00
8020: 84 08             59          STY  BASL        ; FOR VIDEOTERM EMULATION
8022: 60                60          RTS
```

```
                        63   *
                        64   * THIS WILL DISPLAY A CURSOR AT THE SCREEN POSITION
                        65   * SET UP BY THE GOTOXY ROUTINE.
                        66   *
                        67   CURSOR
8023: A9 0E             68            LDA  #$0E
8025: 8D B0 C0          69            STA  DEV0
8028: AD 26 81          70            LDA  PAGE
802B: 29 1F             71            AND  #$1F
802D: 8D B1 C0          72            STA  DEV1
8030: A9 0F             73            LDA  #$0F
8032: 8D B0 C0          74            STA  DEV0
8035: AD 25 81          75            LDA  CHPAGE
8038: 8D B1 C0          76            STA  DEV1
803B: 4C 7F 80          77            JMP  PAGSEL
                        78   *
                        79   * THIS ROUTINE ESTABLISHES A STARTING LOCATION FOR THE
                        80   * DISPLAY OF CHARACTERS USING STOADV.
                        81   *
                        82   * THE HORIZONTAL POSITION IS CH (36)
                        83   * THE VERTICAL   POSITION IS CV (37)
                        84   *
                        85   * ONE WAY OF SETTING THIS POSITION BEFORE USING GOTOXY
                        86   * IS TO POKE 36,HORIZOTAL AND POKE 37,VERICAL WITHIN
                        87   * THE APPLESOFT PROGRAM.  SEE PAGE 129 OF THE
                        88   * APPLESOFT MANUAL.
                        89   *
                        90   *
                        91   GOTOXY
803E: 84 34             92            STY  YSAVE      ; PRESERVE THE Y REGISTER
8040: A5 24             93            LDA  CH
8042: AC 24 81          94            LDY  FORMAT
8045: C0 06             95            CPY  #$06
8047: D0 03             96            BNE  NOT132
8049: 18                97            CLC
804A: 69 0F             98            ADC  #15
804C: 8D 25 81          99   NOT132   STA  CHPAGE
804F: B9 02 81          100           LDA  MTBL,Y
8052: A8                101           TAY
8053: A9 00             102           LDA  #$00
8055: 18                103           CLC
8056: 65 25             104  MLOOP    ADC  CV
8058: 88                105           DEY
8059: D0 FB             106           BNE  MLOOP
                        107  *
805B: 6D FB 06          108           ADC  START
805E: 20 A7 80          109           JSR  MULTIPLY
8061: 18                110           CLC
8062: 6D 25 81          111           ADC  CHPAGE
8065: 8D 25 81          112           STA  CHPAGE
8068: 90 15             113           BCC  PAGSEL
806A: EE 26 81          114           INC  PAGE
806D: 4C 7F 80          115           JMP  PAGSEL
```

```
                    118  *
                    119  * STORE AND ADVANCE
                    120  *
                    121  * BEFORE CALLING THIS ROUTINE PUT THE CHARACTER
                    122  * YOU WANT TO DISPLAY IN THE ACCUMULATOR.
                    123  *
                    124  * BEFORE USING THIS ROUTINE ESTABLISH AN INITIAL
                    125  * HORIZONTAL / VERTICAL POSITION ON THE SCREEN WITH
                    126  * THE GOTOXY ROUTINE.
                    127  *
                    128  * IN ADDITION TO DISPLAYING THE CHARACTER ON THE SCREEN,
                    129  * THIS ROUTINE WILL AUTOMATICALLY ADVANCE TO THE NEXT
                    130  * SCREEN POSITION.
                    131  *
                    132  * FOR EXAMPLE,
                    133  *        JSR  GOTOXY       ; SET INITIAL SCREEN POSITION
                    134  *        LDA  'V'          ; LOAD ACCUMULATOR WITH CHARACTER
                    135  *        JSR  STOADV
                    136  *        LDA  'I'
                    137  *        JSR  STOADV       ; NOTE THAT GOTOXY IS USED ONCE
                    138  *        LDA  'X'          ; AT START OF DISPLAY OF TEXT.
                    139  *        JSR  STOADV
                    140  *
                    141  * THIS WOULD DISPLAY     VIX    ON THE SCREEN
                    142  *
                    143  STOADV
8070: 84 34         144           STY  YSAVE        ; PRESERVE THE Y REGISTER
8072: AC 25 81      145           LDY  CHPAGE
8075: 91 08         146           STA  (BASL),Y
8077: EE 25 81      147           INC  CHPAGE
807A: D0 1D         148           BNE  EXIT
807C: EE 26 81      149           INC  PAGE
807F: AD 26 81      150  PAGSEL   LDA  PAGE
8082: 29 0F         151           AND  #$0F
8084: 0D 27 81      152           ORA  MODEMASK
8087: 8D B2 C0      153           STA  DEV2         ; NORMALLY DEV2,Y
808A: 48            154           PHA
808B: 0A            155           ASL
808C: 29 0C         156           AND  #$0C
808E: A8            157           TAY
808F: B9 B0 C0      158           LDA  DEV0,Y
8092: 68            159           PLA
8093: 29 01         160           AND  #$01
8095: 09 CC         161           ORA  #$CC
8097: 85 09         162           STA  BASH
8099: A4 34         163  EXIT     LDY  YSAVE        ; RECOVER THE Y REGISTER
809B: 60            164           RTS
                    165  *
                    166  CSROFF
809C: A9 0E         167           LDA  #$0E
809E: 8D B0 C0      168           STA  DEV0
80A1: A9 FF         169           LDA  #$FF
80A3: 8D B1 C0      170           STA  DEV1
80A6: 60            171           RTS
                    172  *
```

```
                    175  *
80A7: 48            176  MULTIPLY PHA
80A8: 6A            177           ROR
80A9: 4A            178           LSR
80AA: 4A            179           LSR
80AB: 4A            180           LSR
80AC: 8D 26 81      181           STA  PAGE
80AF: 68            182           PLA
80B0: 0A            183           ASL
80B1: 0A            184           ASL
80B2: 0A            185           ASL
80B3: 0A            186           ASL
80B4: 60            187           RTS
                    188  *
                    189  * NOTE THAT THE SCROLL ROUTINES WOULD EVENTUALLY
                    190  * RE-DISPLAY TEXT THAT HAD BEEN SCROLLED OFF THE SCREEN,
                    191  * BUT IT WOULD LOOK LIKE GARBAGE WHEN IT RE-APPEARED.
                    192  * IT IS RECOMMENDED THAT YOU PRINT A NEW LINE AT THE TOP OR
                    193  * BOTTOM OF THE SCREEN DEPENDING ON THE DIRECTION OF
                    194  * THE SCROLL.
                    195  *
                    196  * SCROLL THE SCREEN DOWN ONE LINE
                    197  *
                    198  SCROLLDN                  ; SCROLL DOWN
80B5: 38            199           SEC
80B6: AC 24 81      200           LDY  FORMAT
80B9: AD FB 06      201           LDA  START
80BC: F9 02 81      202           SBC  MTBL,Y
80BF: 4C CC 80      203           JMP  MORESCR
                    204  *
                    205  * SCROLL THE SCREEN UP ONE LINE
                    206  *
                    207  SCROLLUP                  ; SCROLL UP
80C2: 18            208           CLC
80C3: AC 24 81      209           LDY  FORMAT
80C6: AD FB 06      210           LDA  START
80C9: 79 02 81      211           ADC  MTBL,Y
80CC: 8D FB 06      212  MORESCR  STA  START       ; CONTINUE THE SCROLL ROUTINE
80CF: 20 A7 80      213           JSR  MULTIPLY
80D2: 48            214           PHA
80D3: A9 0C         215           LDA  #$0C
80D5: 8D B0 C0      216           STA  DEV0
80D8: AD 26 81      217           LDA  PAGE
80DB: 29 0F         218           AND  #$0F
80DD: 8D B1 C0      219           STA  DEV1
80E0: A9 0D         220           LDA  #$0D
80E2: 8D B0 C0      221           STA  DEV0
80E5: 68            222           PLA
80E6: 8D B1 C0      223           STA  DEV1
80E9: 20 3E 80      224           JSR  GOTOXY
80EC: 60            225           RTS
                    226  *
```

```
                    229  *
                    230  * INSTEAD OF SOFTINIT, IT IS RECOMMENDED THAT YOU
                    231  * TURN ON THE ULTRATERM FROM THE APPLESOFT
                    232  * PROGRAM WITH    PRINT CHR$(4)"PR#3"
                    233  * THIS WILL PERFORM A SOFTINIT AND ALSO CONNECT
                    234  * THE I/O HOOKS.
                    235  *
                    236  SOFTINIT
80ED: A9 00         237          LDA  #$00
80EF: 85 08         238          STA  BASL       ; FOR VIDEOTERM EMULATION
                    239  *
80F1: AD FF CF      240          LDA  $CFFF
80F4: AD 00 C3      241          LDA  $C300
80F7: A2 C3         242          LDX  #$C3
80F9: A0 30         243          LDY  #$30
80FB: 8C F8 06      244          STY  N0
80FE: 20 00 C8      245          JSR  $C800
8101: 60            246          RTS
                    247  *
8102: 05            248  MTBL     DFB  80/$10
8103: 06            249          DFB  96/$10
8104: 0A            250          DFB  160/$10
8105: 05            251          DFB  80/$10
8106: 05            252          DFB  80/$10
8107: 05            253          DFB  80/$10
8108: 0A            254          DFB  160/$10
8109: 08            255          DFB  128/$10
810A: 40            256  MODETBL  HEX  40         ;  80X
810B: 50            257          HEX  50         ;  96X
810C: 70            258          HEX  70         ; 160X
810D: 50            259          HEX  50         ;  80X
810E: 50            260          HEX  50         ;  80X
810F: 50            261          HEX  50         ;  80X
8110: 70            262          HEX  70         ; 132X
8111: 70            263          HEX  70         ; 128X
8112: 18            264  HIGHTBL  DFB  24
8113: 18            265          DFB  24
8114: 18            266          DFB  24
8115: 18            267          DFB  24
8116: 20            268          DFB  32
8117: 30            269          DFB  48
8118: 18            270          DFB  24
8119: 20            271          DFB  32
811A: 50            272  WIDETBL  DFB  80
811B: 60            273          DFB  96
811C: A0            274          DFB  160
811D: 50            275          DFB  80
811E: 50            276          DFB  80
811F: 50            277          DFB  80
8120: A0            278          DFB  160
8121: 80            279          DFB  128
                    280  *
8122: 00            281  WIDE     HEX  00         ; WIDTH OF SCREEN
8123: 00            282  HIGH     HEX  00         ; HEIGTH OF SCREEN
8124: FF            283  FORMAT   HEX  FF         ; CURRENT SCREEN FORMAT
8125: 00            284  CHPAGE   HEX  00         ; CURRENT PAGE HORIZONTAL
8126: 00            285  PAGE     HEX  00         ; CURRENT PAGE OF SCREEN
8127: 50            286  MODEMASK HEX  50         ; CURRENT MODE
```

```
--End assembly, 296 bytes, Errors: 0

Symbol table - alphabetical order:

   BASH     =$09        BASL     =$08        CH       =$24     ?  CHKFORM =$8000
   CHPAGE   =$8125   ?  CSROFF   =$809C   ?  CURSOR   =$8023     CV       =$25
   DEV0     =$C0B0      DEV1     =$C0B1      DEV2     =$C0B2   ?  DEV3     =$C0B3
?  DISP0    =$CC00      EXIT     =$8099      FLAGS    =$07FB      FORMAT   =$8124
   GOTOXY   =$803E      HIGH     =$8123      HIGHTBL  =$8112   ?  KBD      =$C000
?  KBDSTRB  =$C010      MLOOP    =$8056   ?  MODE     =$0478      MODEMASK=$8127
   MODETBL  =$810A      MORESCR  =$80CC      MTBL     =$8102      MULTIPLY=$80A7
   N0       =$06F8      NOT132   =$804C      PAGE     =$8126      PAGSEL   =$807F
?  SCROLLDN=$80B5   ?  SCROLLUP=$80C2   ?  SOFTINIT=$80ED      START    =$06FB
?  STOADV   =$8070      WIDE     =$8122      WIDETBL  =$811A      YSAVE    =$34


Symbol table - numerical order:

   BASL     =$08        BASH     =$09        CH       =$24        CV       =$25
   YSAVE    =$34     ?  MODE     =$0478      N0       =$06F8      START    =$06FB
   FLAGS    =$07FB   ?  CHKFORM  =$8000   ?  CURSOR   =$8023      GOTOXY   =$803E
   NOT132   =$804C      MLOOP    =$8056   ?  STOADV   =$8070      PAGSEL   =$807F
   EXIT     =$8099   ?  CSROFF   =$809C      MULTIPLY=$80A7   ?  SCROLLDN=$80B5
?  SCROLLUP=$80C2      MORESCR  =$80CC   ?  SOFTINIT=$80ED      MTBL     =$8102
   MODETBL  =$810A      HIGHTBL  =$8112      WIDETBL  =$811A      WIDE     =$8122
   HIGH     =$8123      FORMAT   =$8124      CHPAGE   =$8125      PAGE     =$8126
   MODEMASK=$8127   ?  KBD      =$C000   ?  KBDSTRB  =$C010      DEV0     =$C0B0
   DEV1     =$C0B1      DEV2     =$C0B2   ?  DEV3     =$C0B3   ?  DISP0    =$CC00
```

Using the screen drivers from BASIC

It is fairly easy to access the screen driver routines from BASIC. The following is a listing of a short assembly-language routine which reads the character at a specified screen location and stores the character value at an address that can be read from a BASIC program. The routine begins by making sure the UltraTerm is on and sets up certain variables. Next comes a JSR to the GOTOXY function which selects the screen location specified by the values in Cursor Horizontal (36) and Cursor Vertical (37). The character value at that location is then stored at an address accessible to Applesoft and returns to BASIC.

Also listed is a short Applesoft program which CALLs the assembly language routine (saved as 'GET.SCRN.PEEK'), activates the UltraTerm and produces a display to be read from, and returns the value of the character at the requested location.

While the program and routine are simple, they illustrate the power of the UltraTerm screen drivers. Most of the assembly-language routine is right out of the screen driver listing. The only change is the addition of a few commands to store a character at a particular memory location. In the same way, you could include a segment to take keyboard input and place it at a specified place on the screen, again using the GOTOXY routine. Subsequent input would be placed consecutively on the screen if the Store and Advance (STOADV) routine were incorporated.

If you are familiar with the Applesoft ampersand ('&') command, you will probably want to use it with the screen driver routines. It is beyond the scope of this document to cover the ampersand command, but many publications descibe it and its use. One of these is 'Call-A.P.P.L.E. In Depth, All About Applesoft'.

Note. The Applesoft PEEK command cannot be used to access the UltraTerm's screen memory. This command goes through the Apple's Floating-Point routine and may cause a read of the memory range from $CF00 to $CFFF. Accessing this range of memory will cause the UltraTerm to bank off. (The BIT $C300 command at the beginning of the routines ensures that the UltraTerm is on when the routines are entered regardless of what has gone on in the Applesoft program.)

This is a listing of an Applesoft program to read a character from a specified location of the UltraTerm's screen. It calls the Read Screen routine listed on previous pages.

```
10 D$ = CHR$(4)
20 CH = 36                          :REM Cursor Horizontal
30 CV = 37                          :REM Cursor Vertical
40 LOC = 908                        :REM Routine stores ASCII here
50 PRINT D$ "BLOAD GET.SCRN.PEEK"
60 PRINT D$ "PR#3"                  :REM Turn on UltraTerm (also performs INIT)
70 CV$ = CHR$(22)                   :REM CTRL-V used to set screen size
80 CW$ = CHR$(23)                   :REM CTRL-W used to set attributes
90 PRINT CV$ "5"                    :REM Set 80x32 mode
100 PRINT CW$ "23"                  :REM Set low-light background

105         REM Put something on the screen we can read

110 HOME
120 FOR I = 0 TO 12
130 POKE CH,I: PRINT I
140 NEXT I

145         REM Input the CH, CV position you wish to read from the screen

150 POKE CH,20: POKE CV,20: INPUT "CH = "; HZ
160 POKE CH,20: POKE CV,21: INPUT "CV = "; VT
170 POKE CH,CV: POKE CV,VT
180 CALL 768                        :REM Call routine to read screen at CH, CV
190 A% = PEEK (LOC)                 :REM A% set equal to ASCII value
200 POKE CH,0: POKE CV,28
210 PRINT "ASCII VALUE "; A%
220 PRINT "CHARACTER IS "; CHR$(A%)
230 GOTO 150
```

```
                        1   ********************************
                        2   *                              *
                        3   *   ULTRATERM SCREEN DRIVER     *
                        4   *     READ SCREEN ROUTINE      *
                        5   *   FOR USE WITH APPLESOFT     *
                        6   *          BY VIDEX            *
                        7   *                              *
                        8   *          01/10/84            *
                        9   *                              *
                        10  ********************************
                        11  *
                        12  *
                        13  CH        EQU  36          ; CURSOR HORIZONTAL
                        14  CV        EQU  37          ; CURSOR VERTICAL
                        15  BASL      EQU  $08
                        16  BASH      EQU  $09
                        17  NO        EQU  $6F8
                        18  MODE      EQU  $0478
                        19  *
                        20  START     EQU  $06FB       ;START ADDRESS (SLOT 3)
                        21  FLAGS     EQU  $07FB       ;PERMANENT FLAGS (SLOT 3)
                        22  * SLOT    EQU  $07FB
                        23  *
                        24  DEV0      EQU  $C0B0
                        25  DEV1      EQU  $C0B1
                        26  DEV2      EQU  $C0B2
                        27  DEV3      EQU  $C0B3
                        28  DISP0     EQU  $CC00
                        29  *
                        30            ORG  $300
                        31  *
                        32  *
                        33  BEGIN
0300: 2C 00 C3          34            BIT  $C300       ; TURN ON SCREEN CASE APPLESOFT
                        35  *                          ; WROTE TO $CFXX
0303: AD FB 07          36            LDA  FLAGS
0306: 29 07             37            AND  #$07
0308: 8D 89 03          38            STA  FORMAT
030B: A8                39            TAY
030C: B9 80 03          40            LDA  MODETBL,Y
030F: 8D 8B 03          41            STA  MODEMASK
0312: A9 00             42            LDA  #$00
0314: 85 08             43            STA  BASL        ; FOR VIDEOTERM EMULATION
                        44  *
                        45  * NOW LET'S GET THE SCREEN ASCII VALUE
                        46  * AT LOCATION CH,CV
                        47  *
0316: 20 22 03          48            JSR  GOTOXY      ; POINTS US TO THE SCREEN LOCATION
0319: AC 88 03          49            LDY  CHPAGE      ; CHPAGE AND BASL SET BY GOTOXY
031C: B1 08             50            LDA  (BASL),Y    ; ASCII VALUE IS NOW IN ACCUMULATOR
                        51  *
031E: 8D 8C 03          52            STA  CHAR        ; STORE ASCII VALUE IN CHAR
                        53  *
0321: 60                54  GOBASIC   RTS              ; RETURN TO BASIC PROGRAM
```

```
                        57   GOTOXY
0322: A5 24             58          LDA  CH
0324: AC 89 03          59          LDY  FORMAT
0327: C0 06             60          CPY  #$06
0329: D0 03             61          BNE  NOT132
032B: 18                62          CLC
032C: 69 0F             63          ADC  #15
032E: 8D 88 03          64   NOT132 STA  CHPAGE
0331: B9 78 03          65          LDA  MTBL,Y
0334: A8                66          TAY
0335: A9 00             67          LDA  #$00
0337: 18                68          CLC
0338: 65 25             69   MLOOP  ADC  CV
033A: 88                70          DEY
033B: D0 FB             71          BNE  MLOOP
                        72   *
033D: 6D FB 06          73          ADC  START
0340: 20 6A 03          74          JSR  MULTIPLY
0343: 18                75          CLC
0344: 6D 88 03          76          ADC  CHPAGE
0347: 8D 88 03          77          STA  CHPAGE
034A: 90 03             78          BCC  PAGSEL
034C: EE 8A 03          79          INC  PAGE
034F: AD 8A 03          80   PAGSEL LDA  PAGE
0352: 29 0F             81          AND  #$0F
0354: 0D 8B 03          82          ORA  MODEMASK
0357: 8D B2 C0          83          STA  DEV2       ; NORMALLY DEV2,Y
035A: 48                84          PHA
035B: 0A                85          ASL
035C: 29 0C             86          AND  #$0C
035E: AA                87          TAX
035F: BD B0 C0          88          LDA  DEV0,X
0362: 68                89          PLA
0363: 29 01             90          AND  #$01
0365: 09 CC             91          ORA  #$CC
0367: 85 09             92          STA  BASH
0369: 60                93          RTS
                        94   *
036A: 48                95   MULTIPLY PHA
036B: 6A                96          ROR
036C: 4A                97          LSR
036D: 4A                98          LSR
036E: 4A                99          LSR
036F: 8D 8A 03          100         STA  PAGE
0372: 68                101         PLA
0373: 0A                102         ASL
0374: 0A                103         ASL
0375: 0A                104         ASL
0376: 0A                105         ASL
0377: 60                106         RTS
```

```
0378: 05        109  MTBL     DFB  80/$10
0379: 06        110           DFB  96/$10
037A: 0A        111           DFB  160/$10
037B: 05        112           DFB  80/$10
037C: 05        113           DFB  80/$10
037D: 05        114           DFB  80/$10
037E: 0A        115           DFB  160/$10
037F: 08        116           DFB  128/$10
0380: 40        117  MODETBL  HEX  40          ;  80X
0381: 50        118           HEX  50          ;  96X
0382: 70        119           HEX  70          ; 160X
0383: 50        120           HEX  50          ;  80X
0384: 50        121           HEX  50          ;  80X
0385: 50        122           HEX  50          ;  80X
0386: 70        123           HEX  70          ; 132X
0387: 70        124           HEX  70          ; 128X
0388: 00        125  CHPAGE   HEX  00
0389: FF        126  FORMAT   HEX  FF          ; CURRENT SCREEN FORMAT
038A: 00        127  PAGE     HEX  00          ; CURRENT PAGE OF SCREEN
038B: 50        128  MODEMASK HEX  50          ; CURRENT MODE
038C: 00        129  CHAR     HEX  00          ; LOCATION WHERE ASCII IS STORED

--End assembly, 141 bytes, Errors: 0

Symbol table - alphabetical order:

   BASH     =$09        BASL     =$08     ?  BEGIN    =$0300      CH       =$24
   CHAR     =$038C      CHPAGE   =$0388      CV       =$25        DEV0     =$C0B0
?  DEV1     =$C0B1      DEV2     =$C0B2   ?  DEV3     =$C0B3   ?  DISP0    =$CC00
   FLAGS    =$07FB      FORMAT   =$0389   ?  GOBASIC  =$0321      GOTOXY   =$0322
   MLOOP    =$0338   ?  MODE     =$0478      MODEMASK=$038B      MODETBL  =$0380
   MTBL     =$0378      MULTIPLY=$036A   ?  NO       =$06F8      NOT132   =$032E
   PAGE     =$038A      PAGSEL   =$034F      START    =$06FB

Symbol table - numerical order:

   BASL     =$08        BASH     =$09        CH       =$24        CV       =$25
?  BEGIN    =$0300   ?  GOBASIC  =$0321      GOTOXY   =$0322      NOT132   =$032E
   MLOOP    =$0338      PAGSEL   =$034F      MULTIPLY=$036A      MTBL     =$0378
   MODETBL  =$0380      CHPAGE   =$0388      FORMAT   =$0389      PAGE     =$038A
   MODEMASK=$038B       CHAR     =$038C   ?  MODE     =$0478   ?  NO       =$06F8
   START    =$06FB      FLAGS    =$07FB      DEV0     =$C0B0   ?  DEV1     =$C0B1
   DEV2     =$C0B2   ?  DEV3     =$C0B3   ?  DISP0    =$CC00
```