THIS utility program supplies one of the "missing" sets of commands from Apple Basic. With its use text can be written at any point on the hi-res pages using the usual Basic PRINT statement constructions. The text may be written horizontally, vertically or rotated, and may be normal or in inverse.

It will be especially useful to those readers who plot graphs or need hi-res demonstrations such as described in last month's Applecart program, CAT not CAL.

Finally, simple animations can easily be carried out from Basic on the hi-res pages, and an example is given.

GIVE YOUR HI-RES TEXT

ONE utility missing from Applesoft Basic is the ability to put text easily on the hi-res pages at any position. I used to accomplish this using shape tables, but it is slow and difficult, usually involving string manipulation subroutines.

After I had realised how to calculate the base address and required offset for any pixel on the graphics screens (see *Windfall,* January 1983, page 22) it was an easy step to putting text on at any position.

I decided to create the characters as bit patterns based on an eight deep by seven wide array for two reasons. Firstly it was easy because the Apple's text is created thus, enabling just seven bits of a hi-res byte to be significant. Secondly, it would be easy to create reasonable looking lower case characters with descenders.

To this end I decided that the "bottom" row of any upper case character would be full of "nulls". These would allow separation of text between lines and give room for the descenders. Using bit patterns rather than shape table-like structures also has the advantage that manipulations of the text can easily be carried out since a "constant" array structure has to be processed. The big disappointment is that the on-board character generator cannot be accessed from software, and so valuable memory has to be used to store the character set.

							Hexadi
.	.	1	1	1	1	.	1E
.	1	.	.	.	1	.	22
.	1	.	.	.	1	.	22
.	.	1	1	1	1	.	1E
.	.	.	1	.	1	.	0A
.	.	1	.	.	1	.	12
.	1	.	.	.	1	.	22
.	.	.	.	.	.	.	00

Figure I

My first task therefore was to design the characters. The hi-res screen pixels are controlled by the seven least significant bits of a byte in the correct area of memory. The most significant bit partially controls the colour and as far as text goes was not important to me. The controlling bits are arranged from left to right on the screen so that if set the pixel is lit and if reset the pixel is not lit.

Traditionally numbers are arranged with the more significant digits to the left of the less significant, and so to design the set I simply draw the mirror image of the characters on an eight by eight array, leaving the bottom row and side columns empty. This is shown by considering the letter R. My array was designed as shown in Figure I.

I arranged my bit patterns in Ascii order, with the top row of each pattern at the lower memory location. This enabled me to access any character from its Ascii code by subtracting 32 ($20) and multiplying the result by 8 to give the position in the table of patterns of the top row of the character.

By simply adding the address of the table start and using indirectly addressed indexing I could access any row of the required pattern. For example, the Ascii code of A is $41. Subtracting $20 gives $21 and multiplying this by 8 (three shifts to the left) gives $108. With the program assembled to sit under DOS in a 48k machine the start of the text table is at $9400 and so the top row of letter A is situated at $9508.

I wanted to EOR the patterns on to the hi-res bytes so that animation could be accomplished and I also wanted to be able to stack vertically the text and to rotate it through 90 degrees so that graph axes could be easily labelled as in Figure II, which was created by the example at the end of this article.

I decided therefore to move the bytes of each bit pattern into a temporary storage area so that this could be manipulated before EORing with the

Stringing along the hi-res page with MAX PARROTT

screen. In particular I wanted to be able to put INVERSE text on the screen and to be able to start the text at any screen position, which would necessitate shifting the pattern between bytes.

It was easy to put the patterns into a store for horizontal and vertical text by a straight copy, but a little more difficult to "rotate" the patterns into the same storage area for the rotated text. Finally I decided to copy the pattern into yet another temporary storage area and to manipulate this into the required area of memory. I decided to use the top of page 2 for these temporary areas in order to save memory higher up.

My final problem was to decide on the syntax of the commands. It seemed easiest to use the ubiquitous ampersand for the main commands and to follow this with standard tokenisable Basic commands. I decided to signal whether upper or lower case was to be printed by embedding control characters within the Basic string.

After perusing page 121 of the manual I decided that to print text at screen position x,y (the lower left hand corner of the first character) the commands should be

& PRINT AT x,y, ...

where the dots indicate any legitimate PRINT type statement.

To make all following text appear in inverse the command would be

& INVERSE,

To make the text stack vertically downwards the command would be

& VLIN,

To make the following text rotated the command would be

& ROT=,

Each of these commands would be cleared by the issue of

& NORMAL.

It was clear by this point that quite a lot of memory would be consumed by the program and so I decided to assemble it to sit under DOS and to have two forms, one with lower case ability and one without. It is the latter I have presented here, as generally it is more useful.

In this form the program is BRUNned at $9235, whereupon it sets the ampersand vectors, and sets HIMEM to protect its main routines. The first part of the program is not protected and so if you type in the hexadecimal dump, SAVE it first. The length is $3CB.

I then only had to connect these program parts together so that the Basic line would be properly parsed and error messages given if necessary to complete the program. I used as many Applesoft routines as I could (I am eternally grateful to John Crossley and "The Apple Orchard," Fall 1980, for the entry points of these) for parsing.

The first token on the line of Basic is checked for 'NORMAL', 'VLIN', 'ROT=', or 'INVERSE' and if it is there the appropriate flags are set or reset. If none of these are present SYNCHK (line 94) checks to see if 'PRINT' is present and if it is a check is made for 'AT'. HFNS (line 97) retrieves the starting co-ordinates and the X and Y directions are dealt with as previously described. The presence of a comma is checked and the rest of the line is parsed.

I used FRMEVL to evaluate the formula found. This flags whether the result is a string or a number, VALTYP (line 140) is checked to see which. If it is a number I used FOUT to create a string equal to the

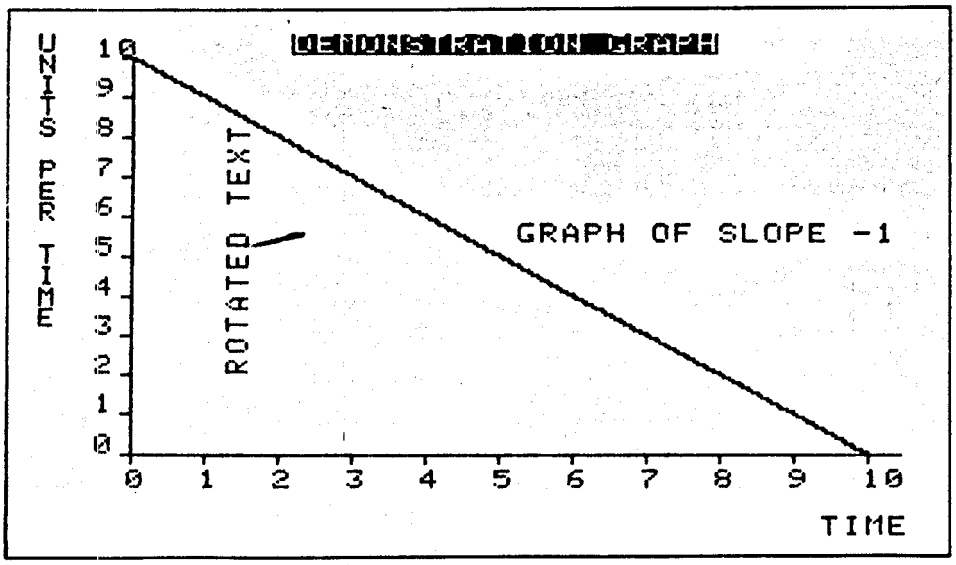

Figure II

GRAPHICS

Floating Point Accumulator at $100-$110 (it leaves a pointer to it in Y,A). I then set DSCTMP+1 to point at it.

If the formula is a string, FRMEVL and the routines it falls into leave a pointer to it in FACMO. I moved this to DSCTMP+1 and JSRed to FREEFAC to free up the temporary string descriptor which had been created.

The string is then dissected, character by character, making checks for zero bytes and double quotes which signify the ends of strings. Any printable character is transformed into the index to the table, the address is looked up and the transfers made.

Its use is best illustrated by two short Basic programs. The first draws a pretend graph and labels the axes and line etc. The second performs a simple piece of animation. Note that writing a string to screen a second time removes it. Both Basic programs assume that the program is resident on disc under the name GRAPHICS TEXT.

```
10  PRINT  CHR$ (4)"BRUN GRAPHICS TEXT"
20  HGR2 : HCOLOR= 3
30  HPLOT 30,10 TO 30,160: HPLOT 30,160 TO 270,160
40  FOR I = 160 TO 10 STEP  - 15: HPLOT 27,I TO 30,I: &  PRINT  AT
18,I,(160-I)/15: NEXT
50  FOR I = 30 TO 260 STEP 23: HPLOT I,160 TO I,163: &  PRINT  AT I -
2,173,(I-30)/23: NEXT
60  &  VLIN : &  PRINT  AT 0,8,"UNITS PER TIME"
70  &  ROT= : &  PRINT  AT 60,130,"ROTATED TEXT"
80  &  NORMAL : &  PRINT  AT 245,191,"TIME"
90  HPLOT 30,10 TO 260,160
100  &  PRINT  AT 150,80,"GRAPH OF SLOPE -1"
110  &  INVERSE : &  PRINT  AT 80,8,"DEMONSTRATION GRAPH"
120  &  NORMAL
130  GET T$: TEXT
```

Program I

```
10 PRINT CHR$(4)"BRUN GRAPHICS TEXT"
20 HGR : HCOLOR= 3:A$ = "HELP"
30 FOR I = 0 TO 279 STEP 7: HPLOT I,0 TO I,161: NEXT
40 FOR I = 0 TO 279 STEP 2
50 & PRINT AT I,50,A$: REM WRITE A$
60 FOR J = 1 TO 20: NEXT
70 & PRINT AT I,50,A$: REM REMOVE A$
80 NEXT
90 GET T$: TEXT
```

Program II

```
0000              1  * Copywrite (C) M.J. Parrott 1983.
0000              2  *
0003              3  Y        EPZ 3
0004              4  BASE     EPZ 4
0006              5  XRES     EPZ 6
0007              6  QUOT     EPZ 7
0008              7  YTEMP    EPZ 8
0009              8  TEMP     EPZ 9
0011              9  VALTYP   EPZ $11
005E             10  INDEX    EPZ $5E
0073             11  HIMEM    EPZ $73
009D             12  DSCTMP   EPZ $9D
00A0             13  FACMO    EPZ $A0
00B1             14  CHRGET   EPZ $B1
00B7             15  CHRGOT   EPZ $B7
00BB             16  TXTPTR   EPZ $BB
00E6             17  HPAG     EPZ $E6
0800             18  *
0800             19  * TOKENS
0800             20  *
008F             21  VLIN     EPZ 143
009B             22  ROT      EPZ 152
009D             23  NORMAL   EPZ 157
009E             24  INVERSE  EPZ 158
00BA             25  PRNTOK   EPZ 186
00C5             26  AT       EPZ 197
0800             27  *
0800             28  * SUBROUTINES ETC.
0800             29  *
02F0             30  TSTORE   EQU $2F0
02F8             31  STORE    EQU $2F8
02EF             32  XSTORE   EQU $2EF
02EE             33  YSTORE   EQU $2EE
03F5             34  AMPER    EQU $3F5
D412             35  ERROR    EQU $D412
D995             36  UPDATE   EQU $D995
DD7B             37  FRMEVL   EQU $DD7B
DEBE             38  CHKCOM   EQU $DEBE
DEC0             39  SYNCHK   EQU $DEC0
E3ED             40  STRLT2   EQU $E3ED
E600             41  FREEFAC  EQU $E600
ED34             42  FOUT     EQU $ED34
F6B9             43  HFNS     EQU $F6B9
0800             44  *
0800             45  *SET UP AMPERSAND
0800             46  *
9235             47          ORG $9235
9235             48          OBJ $1000
9235 A9 4C       49          LDA #$4C          ;JMP COMMAND
9237 8D F5 03    50          STA AMPER
923A A9 49       51          LDA #START
923C 8D F6 03    52          STA AMPER+1
923F 85 73       53          STA HIMEM
9241 A9 92       54          LDA /START
9243 8D F7 03    55          STA AMPER+2
9246 85 74       56          STA HIMEM+1
9248 60          57          RTS               ;AND BACK TO BASIC
9249             58 START:
9249 C9 9D       59          CMP #NORMAL       ;IS FIRST CHAR THE
924B D0 0D       60          BNE ROTIT         ;NORMAL TOKEN?
924D A9 00       61          LDA #0
924F 8D B6 92    62          STA IFLAG
9252 8D B8 92    63          STA VFLAG
9255 8D B7 92    64          STA RFLAG
9258 F0 29       65          BEQ RETURN
925A             66 ROTIT:
925A C9 BF       67          CMP #VLIN
925C D0 0C       68          BNE ROTT
925E A9 00       69          LDA #0
9260 8D B7 92    70          STA RFLAG
9263 A9 FF       71          LDA #$FF
9265 8D B8 92    72          STA VFLAG
926B D0 19       73          BNE RETURN
926A             74 ROTT:
926A C9 98       75          CMP #ROT
926C D0 0C       76          BNE INV
926E A9 FF       77          LDA #$FF
9270 8D B7 92    78          STA RFLAG
9273 A9 00       79          LDA #0
9275 8D B8 92    80          STA VFLAG         ;ZERO IT IN CASE
9278 F0 09       81          BEQ RETURN
927A             82 INV:
927A C9 9E       83          CMP #INVERSE
927C D0 0B       84          BNE PRINT
927E A9 7F       85          LDA #$7F
9280 8D B6 92    86          STA IFLAG
9283             87 RETURN:
9283 4C 95 D9    88          JMP UPDATE        ;MOVE TXTPTR AND RETURN
9286             89 IFLAG    DFS 1,0
9287             90 RFLAG    DFS 1,0
9288             91 VFLAG    DFS 1,0
9289             92 PRINT:
9289 A9 BA       93          LDA #PRNTOK       ;CHECK FOR PRINT
928B 20 C0 DE    94          JSR SYNCHK        ;TOKEN
928E A9 C5       95          LDA #AT           ;AND FOR AT TOKEN
9290 20 C0 DE    96          JSR SYNCHK
9293 20 B9 F6    97          JSR HFNS          ;GET PLOTTING COORDS
9296 85 03       98          STA Y             ;STORE THE Y COORD
9298 86 04       99          STX BASE          ;AND LSB OF X COORD
929A            100  *
929A            101  * CALC STARTING COORDS FOR STRING
929A            102  *
929A 98         103          TYA               ;TAKE THE MSB OF X COORD
929B A0 07      104          LDY #7            ;AND DIVIDE BY 7 TO CALC
929D 84 06      105          STY XRES          ;BYTE FOR PLOTTING OFFSET
929F 38         106          SEC
92A0 E5 06      107          SBC XRES
92A2 08         108 LOOP     PHP
92A3 26 07      109          ROL QUOT
92A5 06 04      110          ASL BASE
92A7 2A         111          ROL
92A8 28         112          PLP
92A9 90 05      113          BCC ADD
92AB E5 06      114          SBC XRES
92AD 4C B2 92   115          JMP NEXT
92B0 65 06      116 ADD      ADC XRES
92B2 88         117 NEXT     DEY
92B3 10 ED      118          BPL LOOP
92B5 B0 03      119          BCS LAST
92B7 65 06      120          ADC XRES
92B9 18         121          CLC
92BA 26 07      122 LAST     ROL QUOT          ;ON EXIT QUOT HAS #BYTE
92BC 85 06      123          STA XRES          ;& XRES HAS #BIT
92BE A5 07      124          LDA QUOT          ;CHECK TO SEE IF ON SCREEN
92C0 10 05      125          BPL CHECK
92C2 A2 35      126 ERR      LDX #53           ;'ILLEGAL QUANTITY'
92C4 4C 12 D4   127          JMP ERROR
92C7 C9 28      128 CHECK    CMP #40           ;IF OVER 39 ERROR
92C9 B0 F7      129          BCS ERR
92CB A5 03      130          LDA Y             ;GET THE Y COORD
92CD C9 C0      131          CMP #192          ;IS IT ON SCREEN?
92CF B0 F1      132          BCS ERR
```

```
92D1 E6 03          INC Y                 ;USED LATER
92D3                ;
92D3                ; PARSE REST OF LINE
92D3                ;
92D3 20 BE DE       JSR CHKCOM            ;IS IT A COMMA NEXT?
92D6          PRINTIT:
92D6 20 7B DD       JSR FRMEVL            ;EVALUATE WHAT'S THERE
92D9 24 11          BIT VALTYP            ;STRING OR NUMBER?
92DB 30 0A          BMI STR
92DD 20 34 ED       JSR FOUT              ;CREATE STRING FROM FAC
92E0 85 9E          STA DSCTMP+1          ;SET POINTER TO IT
92E2 84 9F          STY DSCTMP+2
92E4 4C F4 92       JMP STRING1           ;NOW PRINT IT
92E7          STR:
92E7 A0 02          LDY #2                ;NEED TO SET DSCTMP
92E9 B1 A0    TRANS LDA (FACMO),Y         ;FROM FACMO,LO
92EB 99 9D 00       STA DSCTMP,Y
92EE 88             DEY
92EF D0 F8          BNE TRANS             ;DON'T NEED LENGTH
92F1 20 00 E6       JSR FREEFAC           ;FREE TEMP DESCRIPTOR
92F4          STRING1:
92F4 20 0A 93       JSR WRITE1            ;PUT IT ON SCREEN
92F7          WHTNXT1:
92F7 20 B7 00       JSR CHRGOT            ;CHECK THE CHAR
92FA          WHTNXT:
92FA F0 0D          BEQ NOMORE
92FC C9 2C          CMP #$2C              ;IS IT A COMMA?
92FE F0 04          BEQ MVIT
9300 C9 3B          CMP #$3B              ;IS IT A ;
9302 D0 D2          BNE PRINTIT
9304          MVIT:
9304 20 B1 00       JSR CHRGET            ;GET NEXT CHAR
9307 D0 F1          BNE WHTNXT
9309          NOMORE:
9309 60             RTS                   ;BACK TO BASIC
930A          WRITE1:
930A A0 00          LDY #0                ;INIT CHAR COUNTER
930C          PRINT2:
930C A5 03          LDA Y                 ;GET COORD FOR BOTTOM
930E 85 08          STA YTEMP
9310 A9 00          LDA #0                ;CALC INDEX OF CHAR
9312 85 5F          STA INDEX+1
9314 8C EE 02       STY YSTORE            ;STORE CHAR COUNTER
9317 B1 9E          LDA (DSCTMP+1),Y      ;GET ASCII OF CHAR
9319 D0 01          BNE YES1
931B          NO1:
931B 60             RTS                   ;BACK TO CALLER
931C          YES1:
931C C9 22          CMP #$22              ;STOP ON A QUOTE
931E F0 FB          BEQ NO1
9320 38             SEC
9321 E9 20          SBC #$20              ;PUT IN RANGE 0 TO $5F
9323 85 5E          STA INDEX
9325 A2 02          LDX #2                ;TO MULTIPLY BY 8
9327          MULT:
9327 06 5E          ASL INDEX             ;LEAVING RESULT IN INDEX
9329 26 5F          ROL INDEX+1
932B CA             DEX
932C 10 F9          BPL MULT
932E 18             CLC
932F A5 5E          LDA INDEX             ;ADD THE ADDRESS OF TABLE
9331 69 00          ADC #TEXT             ;START
9333 85 5E          STA INDEX
9335 A5 5F          LDA INDEX+1
9337 69 94          ADC /TEXT
9339 85 5F          STA INDEX+1
933B A0 07          LDY #7                ;COUNTER FOR COLUMNS
933D A2 07          LDX #7                ;AND ROWS
933F AD B7 92       LDA RFLAG             ;WHAT'S ROT?
9342 F0 1E          BEQ HORIZ
9344          PUTIN:
9344 B1 5E          LDA (INDEX),Y         ;USE TEMP STORAGE
9346 99 F0 02       STA TSTORE,Y
9349 88             DEY
934A 10 F8          BPL PUTIN
934C A2 07          LDX #7                ;COUNTER FOR BYTES
934E A0 07    BYTES LDY #7                ;COUNTER FOR BITS
9350 B9 F0 02 BITS  LDA TSTORE,Y          ;GET PATTERN
9353 4A             LSR
9354 99 F0 02       STA TSTORE,Y
9357 3E F8 02       ROL STORE,X
935A 88             DEY
935B 10 F3          BPL BITS
935D CA             DEX
935E 10 EE          BPL BYTES
9360 30 08          BMI WRITE
9362          HORIZ:
9362 B1 5E          LDA (INDEX),Y         ;TRANSFER BIT PATTERN
9364 99 F8 02       STA STORE,Y           ;TO THE STORE AREA
9367 88             DEY
9368 10 F8          BPL HORIZ
936A          WRITE:
936A A2 07          LDX #7
936C          LOOP1:
936C A4 08          LDY YTEMP             ;GET THE Y COORD FOR THE
PATTERN
936E 88             DEY                   ;MOVE UP ONE ROW
936F 84 08          STY YTEMP             ;SAVE IT
9371 98             TYA
9372 20 C8 93       JSR YCALC             ;CALC NEW ADDRESS BASE
9375 A9 00          LDA #0                ;INIT TEMP TO
9377 85 09          STA TEMP              ;ZEROS
9379 AD B6 92       LDA IFLAG             ;INVERSE WANTED?
937C 5D F8 02       EOR STORE,X           ;SUPERIMPOSE PATTERN
937F 8E EF 02       STX XSTORE            ;SAVE ROW COUNTER
9382 A6 06          LDX XRES              ;WITH #BITS REQUIRED
9384 F0 07          BEQ NO                ;FOR THIS BYTE OF HIRES
9386          SHIFT:
9386 0A             ASL                   ;MOVE PATTERN AS REQUIRED
9387 26 09          ROL TEMP              ;INTO TEMP AS WELL AS A
9389 CA             DEX
938A 10 FA          BPL SHIFT             ;ONCE MORE FOR BIT 7
938C 4A             LSR                   ;AND SHIFT ACC BACK TO CLEAR 7
938D          NO:
938D A4 07          LDY QUOT              ;GET THE OFFSET FOR THE BYTE
938F 51 04          EOR (BASE),Y          ;ADJUST BYTE
9391 91 04          STA (BASE),Y          ;AND PUT IT IN
9393 C8             INY
9394 C0 28          CPY #40
9396 90 01          BCC COON
9398          END:
9398 60             RTS                   ;BACK TO BASIC
9399          COON:
9399 A5 09          LDA TEMP              ;GET PATTERN
939B 51 04          EOR (BASE),Y          ;FORCE THE PATTERN
939D 91 04          STA (BASE),Y          ;AND PUT ON SCREEN
939F AE EF 02       LDX XSTORE            ;RESTORE ROW COUNTER
93A2 CA             DEX                   ;MOVE UP ONE ROW
93A3 10 C7          BPL LOOP1             ;FINISH 8 ROWS
93A5 AD B7 92       LDA RFLAG             ;WHAT'S ROT?
93A8 F0 07          BEQ VRT
93AA A5 08          LDA YTEMP
93AC 85 03          STA Y
93AE 4C C4 93       JMP NEXT1
93B1          VRT:
93B1 AD B8 92       LDA VFLAG
93B4 F0 09          BEQ HORIZ1
93B6 18             CLC
93B7 A5 03          LDA Y
93B9 69 08          ADC #8
93BB 85 03          STA Y
93BD D0 05          BNE NEXT1
93BF          HORIZ1:
93BF A4 07          LDY QUOT              ;GET OFFSET FOR 1ST BYTE
93C1 C8             INY                   ;SAFE AS CHECKED BEFORE
93C2 84 07          STY QUOT
93C4          NEXT1:
93C4 AC EE 02       LDY YSTORE            ;GET STRING COUNTER
93C7 C8             INY
93C8 4C 0C 93       JMP PRINT2            ;NO! SO BACK FOR NEXT CHAR
93CB          YCALC:
93CB C9 C0          CMP #192              ;IS IT ON SCREEN?
93CD 90 03          BCC YES
93CF 68             PLA                   ;DISCARD A WORD
93D0 68             PLA
93D1 60             RTS                   ;BACK TO BASIC
93D2          YES:
93D2 29 30          AND #$30              ;Y COORD ALREADY IN A
93D4 4A             LSR
93D5 4A             LSR
93D6 4A             LSR
93D7 4A             LSR
93D8 05 E6          ORA HPAG              ;WHICH PAGE?
93DA 85 05          STA BASE+1
93DC A5 08          LDA YTEMP
93DE 29 07          AND #$7
93E0 0A             ASL
93E1 0A             ASL
93E2 65 05          ADC BASE+1
93E4 85 05          STA BASE+1
93E6 A5 08          LDA YTEMP
93E8 29 C0          AND #$C0
93EA 4A             LSR
93EB 85 04          STA BASE
93ED 4A             LSR
93EE 4A             LSR
93EF 05 04          ORA BASE
93F1 85 04          STA BASE
93F3 A5 08          LDA YTEMP
93F5 29 08          AND #$8
93F7 F0 06          BEQ DONE
93F9 A9 80          LDA #$80
93FB 65 04          ADC BASE
93FD 85 04          STA BASE
93FF          DONE:
93FF 60             RTS
9400          TEXT:
9400 00 00 00       HEX 0000000000000000 ;space
```

```
9403 00 00 00
9406 00 00
940B 08 08 08    320        HEX 0808080808000800 ;!
940B 08 08 00
940E 08 00
9410 14 14 14    321        HEX 1414140000000000 ;"
9413 00 00 00
9416 00 00
9418 14 14 3E    322        HEX 14143E143E141400 ;#
941B 14 3E 14
941E 14 00
9420 08 3C 0A    323        HEX 083C0A1C281E0800 ;$
9423 1C 28 1E
9426 08 00
9428 06 26 10    324        HEX 0626100804323000 ;%
942B 08 04 32
942E 30 00
9430 04 0A 0A    325        HEX 040A0A042A122C00 ;&
9433 04 2A 12
9436 2C 00
943B 08 08 00    326        HEX 0808000000000000 ;'
943B 00 00 00
943E 00 00
9440 08 04 02    327        HEX 0804020202040800 ;(
9443 02 02 04
9446 08 00
944B 08 10 20    328        HEX 0810202020100800 ;)
944B 20 20 10
944E 08 00
9450 08 2A 1C    329        HEX 082A1C081C2A0800 ;*
9453 08 1C 2A
9456 08 00
9458 00 08 08    330        HEX 0008083E08080000 ;+
945B 3E 08 08
945E 00 00
9460 00 00 00    331        HEX 0000000008080400 ;,
9463 00 08 08
9466 04 00
946B 00 00 00    332        HEX 0000003E00000000 ;-
946B 3E 00 00
946E 00 00
9470 00 00 00    333        HEX 0000000000000800 ;.
9473 00 00 00
9476 08 00
9478 00 20 10    334        HEX 0020100804020000 ;/
947B 08 04 02
947E 00 00
9480 1C 22 32    335        HEX 1C22322A26221C00 ;0
9483 2A 26 22
9486 1C 00
948B 08 0C 08    336        HEX 080C080808081C00 ;1
948B 08 08 08
948E 1C 00
9490 1C 22 20    337        HEX 1C22201804023E00 ;2
9493 18 04 02
9496 3E 00
9498 3E 20 10    338        HEX 3E20101820221C00 ;3
949B 18 20 22
949E 1C 00
94A0 10 18 14    339        HEX 101814123E101000 ;4
94A3 12 3E 10
94A6 10 00
94AB 3E 02 1E    340        HEX 3E021E2020221C00 ;5
94AB 20 20 22
94AE 1C 00
94B0 38 04 02    341        HEX 3804021E22221C00 ;6
94B3 1E 22 22
94B6 1C 00
94BB 3E 20 10    342        HEX 3E20100804040400 ;7
94BB 08 04 04
94BE 04 00
94C0 1C 22 22    343        HEX 1C22221C22221C00 ;8
94C3 1C 22 22
94C6 1C 00
94CB 1C 22 22    344        HEX 1C22223C20100E00 ;9
94CB 3C 20 10
94CE 0E 00
94D0 00 00 08    345        HEX 0000080008000000 ;:
94D3 00 08 00
94D6 00 00
94DB 00 00 08    346        HEX 0000080008080400 ;;
94DB 00 08 08
94DE 04 00
94E0 10 08 04    347        HEX 1008040204081000 ;<
94E3 02 04 08
94E6 10 00
94EB 00 00 3E    348        HEX 00003E003E000000 ;=
94EB 00 3E 00
94EE 00 00
94F0 04 08 10    349        HEX 0408102010080400 ;>
94F3 20 10 08
94F6 04 00
94FB 1C 22 10    350        HEX 1C22100808000800 ;?
94FB 08 08 00
94FE 08 00
9500 1C 22 2A    351        HEX 1C222A2A1A023C00 ;@
9503 2A 1A 02
9506 3C 00
9508 08 14 22    352        HEX 081422223E222200 ;A
950B 22 3E 22
950E 22 00
9510 1E 22 22    353        HEX 1E22221E22221E00 ;B
9513 1E 22 22
9516 1E 00
9518 1C 22 02    354        HEX 1C22020202221C00 ;C
951B 02 02 22
951E 1C 00
9520 1E 22 22    355        HEX 1E22222222221E00 ;D
9523 22 22 22
9526 1E 00
9528 3E 02 02    356        HEX 3E02021E02023E00 ;E
952B 1E 02 02
952E 3E 00
9530 3E 02 02    357        HEX 3E02021E02020200 ;F
9533 1E 02 02
9536 02 00
9538 3C 02 02    358        HEX 3C02020232223C00 ;G
953B 02 32 22
953E 3C 00
9540 22 22 22    359        HEX 2222223E22222200 ;H
9543 3E 22 22
9546 22 00
954B 1C 08 08    360        HEX 1C08080808081C00 ;I
954B 08 08 08
954E 1C 00
9550 20 20 20    361        HEX 2020202020221C00 ;J
9553 20 20 22
9556 1C 00
9558 22 12 0A    362        HEX 22120A060A122200 ;K
955B 06 0A 12
955E 22 00
9560 02 02 02    363        HEX 0202020202023E00 ;L
9563 02 02 02
9566 3E 00
956B 22 36 2A    364        HEX 22362A2222222200 ;M
956B 22 22 22
956E 22 00
9570 22 22 26    365        HEX 2222262A32222200 ;N
9573 2A 32 22
9576 22 00
9578 1C 22 22    366        HEX 1C22222222221C00 ;O
957B 22 22 22
957E 1C 00
9580 1E 22 22    367        HEX 1E22221E02020200 ;P
9583 1E 02 02
9586 02 00
9588 1C 22 22    368        HEX 1C2222222A225C00 ;Q
958B 22 2A 22
958E 5C 00
9590 1E 22 22    369        HEX 1E22221E0A122200 ;R
9593 1E 0A 12
9596 22 00
959B 1C 22 02    370        HEX 1C22021C20221C00 ;S
959B 1C 20 22
959E 1C 00
95A0 3E 08 08    371        HEX 3E08080808080800 ;T
95A3 08 08 08
95A6 08 00
95AB 22 22 22    372        HEX 2222222222221C00 ;U
95AB 22 22 22
95AE 1C 00
95B0 22 22 22    373        HEX 2222222222140800 ;V
95B3 22 22 14
95B6 08 00
95BB 22 22 22    374        HEX 2222222A2A362200 ;W
95BB 2A 2A 36
95BE 22 00
95C0 22 22 14    375        HEX 2222140814222200 ;X
95C3 08 14 22
95C6 22 00
95CB 22 22 14    376        HEX 2222140808080800 ;Y
95CB 08 08 08
95CE 08 00
95D0 3E 20 10    377        HEX 3E20100804023E00 ;Z
95D3 08 04 02
95D6 3E 00
95DB 3E 06 06    378        HEX 3E06060606063E00 ;[
95DB 06 06 06
95DE 3E 00
95E0 00 02 04    379        HEX 0002040810200000 ;backslash
95E3 08 10 20
95E6 00 00
95EB 3E 30 30    380        HEX 3E30303030303E00 ;]
95EB 30 30 30
95EE 3E 00
95F0 00 08 14    381        HEX 0008142200000000 ;^
95F3 22 00 00
95F6 00 00
95FB 00 00 00    382        HEX 0000000000003E00 ;_
95FB 00 00 00
95FE 3E 00
9600             383        END
```

HEXADECIMAL DUMP OF
THE CHARACTER SET

```
*9400.95FF

9400- 00 00 00 00 00 00 00 00
9408- 08 08 08 08 08 00 08 00
9410- 14 14 14 00 00 00 00 00
9418- 14 14 3E 14 3E 14 14 00
9420- 08 3C 0A 1C 28 1E 08 00
9428- 06 26 10 08 04 32 30 00
9430- 04 0A 0A 04 2A 12 2C 00
9438- 08 08 00 00 00 00 00 00
9440- 08 04 02 02 02 04 08 00
9448- 08 10 20 20 20 10 08 00
9450- 08 2A 1C 08 1C 2A 08 00
9458- 00 08 08 3E 08 08 00 00
9460- 00 00 00 00 08 08 04 00
9468- 00 00 00 3E 00 00 00 00
9470- 00 00 00 00 00 00 08 00
9478- 00 20 10 08 04 02 00 00
9480- 1C 22 32 2A 26 22 1C 00
9488- 08 0C 08 08 08 08 1C 00
9490- 1C 22 20 18 04 02 3E 00
9498- 3E 20 10 18 20 22 1C 00
94A0- 10 18 14 12 3E 10 10 00
94A8- 3E 02 1E 20 20 22 1C 00
94B0- 38 04 02 1E 22 22 1C 00
94B8- 3E 20 10 08 04 04 04 00
94C0- 1C 22 22 1C 22 22 1C 00
94C8- 1C 22 22 3C 20 10 0E 00
94D0- 00 00 08 00 08 00 00 00
94D8- 00 00 08 00 08 08 04 00
94E0- 10 08 04 02 04 08 10 00
94E8- 00 00 3E 00 3E 00 00 00
94F0- 04 08 10 20 10 08 04 00
94F8- 1C 22 10 08 08 00 08 00
9500- 1C 22 2A 2A 1A 02 3C 00
9508- 08 14 22 22 3E 22 22 00
9510- 1E 22 22 1E 22 22 1E 00
9518- 1C 22 02 02 02 22 1C 00
9520- 1E 22 22 22 22 22 1E 00
9528- 3E 02 02 1E 02 02 3E 00
9530- 3E 02 02 1E 02 02 02 00
9538- 3C 02 02 02 32 22 3C 00
9540- 22 22 22 3E 22 22 22 00
9548- 1C 08 08 08 08 08 1C 00
9550- 20 20 20 20 20 22 1C 00
9558- 22 12 0A 06 0A 12 22 00
9560- 02 02 02 02 02 02 3E 00
9568- 22 36 2A 22 22 22 22 00
9570- 22 22 26 2A 32 22 22 00
9578- 1C 22 22 22 22 22 1C 00
9580- 1E 22 22 1E 02 02 02 00
9588- 1C 22 22 22 2A 22 5C 00
9590- 1E 22 22 1E 0A 12 22 00
9598- 1C 22 02 1C 20 22 1C 00
95A0- 3E 08 08 08 08 08 08 00
95A8- 22 22 22 22 22 22 1C 00
95B0- 22 22 22 22 22 14 08 00
95B8- 22 22 22 2A 2A 36 22 00
95C0- 22 22 14 08 14 22 22 00
95C8- 22 22 14 08 08 08 08 00
95D0- 3E 20 10 08 04 02 3E 00
95D8- 3E 06 06 06 06 06 3E 00
95E0- 00 02 04 08 10 20 00 00
95E8- 3E 30 30 30 30 30 3E 00
95F0- 00 08 14 22 00 00 00 00
95F8- 00 00 00 00 00 00 3E 00
```

HEXADECIMAL DUMP OF THE PROGRAM

```
*9235.93FF

9235- 55 4D 50
9238- 20 4F 46 20 54 48 45 20
9240- 50 52 4F 47 52 41 4D 04
9248- 20 C9 9D D0 0D A9 00 8D
9250- 86 92 8D 88 92 8D 87 92
9258- F0 29 C9 8F D0 0C A9 00
9260- 8D 87 92 A9 FF 8D 88 92
9268- D0 19 C9 98 D0 0C A9 FF
9270- 8D 87 92 A9 00 8D 88 92
9278- F0 09 C9 9E D0 0B A9 7F
9280- 8D 86 92 4C 95 D9 00 00
9288- 00 A9 BA 20 C0 DE A9 C5
9290- 20 C0 DE 20 B9 F6 85 03
9298- 86 04 98 A0 07 84 06 38
92A0- E5 06 08 26 07 06 04 2A
92A8- 28 90 05 E5 06 4C B2 92
92B0- 65 06 88 10 ED B0 03 65
92B8- 06 18 26 07 85 06 A5 07
92C0- 10 05 A2 35 4C 12 D4 C9
92C8- 28 B0 F7 A5 03 C9 C0 B0
92D0- F1 E6 03 20 BE DE 20 7B
92D8- DD 24 11 30 0A 20 34 ED
92E0- 85 9E 84 9F 4C F4 92 A0
92E8- 02 B1 A0 99 9D 00 88 D0
92F0- F8 20 00 E6 20 0A 93 20
92F8- B7 00 F0 0D C9 2C F0 04
9300- C9 3B D0 D2 20 B1 00 D0
9308- F1 60 A0 00 A5 03 85 08
9310- A9 00 85 5F 8C EE 02 B1
9318- 9E D0 01 60 C9 22 F0 FB
9320- 38 E9 20 85 5E A2 02 06
9328- 5E 26 5F CA 10 F9 18 A5
9330- 5E 69 00 85 5E A5 5F 69
9338- 94 85 5F A0 07 A2 07 AD
9340- 87 92 F0 1E B1 5E 99 F0
9348- 02 88 10 F8 A2 07 A0 07
9350- B9 F0 02 4A 99 F0 02 3E
9358- F8 02 88 10 F3 CA 10 EE
9360- 30 08 B1 5E 99 F8 02 88
9368- 10 F8 A2 07 A4 08 88 84
9370- 08 98 20 CB 93 A9 00 85
9378- 09 AD 86 92 5D F8 02 8E
9380- EF 02 A6 06 F0 07 0A 26
9388- 09 CA 10 FA 4A A4 07 51
9390- 04 91 04 C8 C0 28 90 01
9398- 60 A5 09 51 04 91 04 AE
93A0- EF 02 CA 10 C7 AD 87 92
93A8- F0 07 A5 08 85 03 4C C4
93B0- 93 AD 88 92 F0 09 18 A5
93B8- 03 69 08 85 03 D0 05 A4
93C0- 07 C8 84 07 AC EE 02 C8
93C8- 4C 0C 93 C9 C0 90 03 68
93D0- 68 60 29 30 4A 4A 4A 4A
93D8- 05 E6 85 05 A5 08 29 07
93E0- 0A 0A 65 05 85 05 A5 08
93E8- 29 C0 4A 85 04 4A 4A 05
93F0- 04 85 04 A5 08 29 08 F0
93F8- 06 A9 80 65 04 85 04 60
```