A Review of Tom Pittman's Tiny BASIC

Richard Rosner
Pocono Rd
Brookfield CT 06804

While at the Personal Computing 76 Trade Show at Atlantic City I bought a copy of Tom Pittman's *Tiny BASIC Interpreter* for my home computer, a 6502 with the MOS Technology TIM program in read only memory. Tom Pittman has several different versions of Tiny BASIC for different systems. I purchased the KIM version even though my computer uses a Teletype and the TIM instead of a keyboard. This allows me to load the interpreter starting at 0200 hexadecimal instead of at 1000 hexadecimal, leaving more memory space for my user programs.

Tiny BASIC is a subset of Dartmouth BASIC. It can handle only 16 bit signed integer numbers between −32768 and +32767 and recognizes the following commands:

CLEAR	REM	GOSUB
INPUT	LET	RETURN
LIST	IF . . . THEN	END
PRINT	GOTO	RUN

The only operations understood by Tiny BASIC are addition, subtraction, multiplication, and division (+, −, *, and /). There are two intrinsic functions. One generates a random number and the other, USR, allows subroutine jumps to machine language programs. More about USR later.

Even with these limitations, Tiny BASIC is very powerful. It is easy to learn and use, and programs can be written very quickly. To use Tiny BASIC you first load the paper tape of the interpreter into the programmable memory. This takes about ten minutes on a Teletype. Then write the addresses of the IO routines into the proper locations and jump to the start of the interpreter. In less than an hour I had it up and running. I wrote and debugged a plot routine one Friday night and made several revisions by the time the weekend was over.

The Tiny BASIC plot program shown in listing 1 accepts the X and Y origin and the X and Y increments as inputs. Then, using a function placed in a subroutine, the program plots a 50 by 50 point graph.

The first input, on line 10, determines the form of the output. A 1 input stops the program after the LIST command at line 20 causes lines 1000 to 1999 to be printed. These lines are the subroutine containing the function to be plotted.

A 2 input causes a request for two more inputs: the X origin and the X increment, T and V in line 200. The printed outputs are the function to be plotted and a table of X and Y values. This table is helpful in choosing the Y origin and Y increment to be used for the graph.

A value of 3 as the first input causes a request for four more inputs: the X origin, the X increment, the Y origin, and the Y increment; T, V, U and W in line 300. The outputs are the function to be plotted and the graph.

Lines 205 through 290 generate and print

```
:LIST

10 INPUTX
20 LIST1000,1999
30 IFX=1GOTO100
40 IFX=2GOTO200
50 IFX=3GOTO300
100 END
200 INPUT T,V
205 X=T
210 PR"X","Y"
215 PR"----","----"
220 GOSUB1000
230 PRX,Y
240 X=X+5*V
250 IFX<T+55*VGOTO220
290 END
300 INPUT T,V,U,W
310 PR
320 PR"  Y =",U,"","",U+25*W,"","",U+50*W
330 PR"","!     .     !     .     !     .     !     .     !"
340 PR"    X=    ---------------------------------------------------"
350 X=T
360 GOSUB1000
365 Y=(Y-U+W/2)/W
370 IFY<0GOTO490
375 IFY=0GOTO500
380 PR"    "X,"!";
390 IFY<=50GOTO430
400 PR
410 GOTO510
430 Y=Y-1
440 IFY=0GOTO470
450 PR" ";
460 GOTO430
470 PR"+"
480 GOTO510
490 PR"    "X,"<"
495 GOTO510
500 PR"    "X,"+"
510 X=X+V
520 IFX<T+51*VGOTO360
590 END
1000 REM THE EQUATION TO BE PLOTTED IS
1010 Y=X*X*X/4-3*X*X+2*X+1
1999 RETURN

:
```

Listing 1: Tiny BASIC listing for a plot routine using a function. The equation to be plotted is in the subroutine located at lines 1000 through 1999. The routine has the ability to print three types of outputs. If a value of 1 is input only the function to be plotted is printed. If a value of 2 is input the program requests the X origin and increment to generate and print a table of X and Y values for the equation on line 1010. The X increment is always 5 times the input increment value. If a value of 3 is input the program requests the X, Y origins and increments. The output consists of the equation to be plotted and the graph.

the table of X and Y values. Starting with the first X, which has the value of T, and for ten other X values spaced at five times the X increment, eleven Y values are calculated by calls to the subroutine at line 1000. In Tiny BASIC, PRINT statements are abbreviated "PR." The PRINT statements at lines 210 and 215 print the heading and the PRINT statement at line 230 prints the X and Y values. The IF statement at line 250 continues the loop until the eleventh X and Y values are calculated and printed.

Lines 310 through 590 generate the graph. The PRINT statements at lines 310 to 340 print the heading. Line 320 prints three values of Y. U is the value of Y at the origin, U + 25 * W is the value of Y at the midpoint of the Y axis, and U + 50 * W is the greatest value of Y plotted. The subroutine call at line 360 brings in a value for Y which is scaled and rounded up at line 365. Lines 370 through 500 print the X value, the X axis, and a "<" or "+" to indicate the value of Y. If the scaled value of Y is less than zero, the IF statement at line 370 causes a jump to the PRINT statement at line 490 where a caret is printed on the X axis to indicate that the value of Y is below the lower limit of the Y axis. If the scaled value of Y equals zero, the IF statement at line 375 causes a jump to line 500 where a plus sign is printed on the X axis. If the scaled value of Y is greater than zero, the PRINT at line 380 prints the value of X and an exclamation mark for the X axis. The semicolon at this PRINT statement causes the next PRINT statement to write on the same line without any spacing. If the scaled value of Y is greater than 50 the PRINT at line 400 closes this line and causes the next PRINT statement to start printing at the beginning of the next line. If the scaled value of Y is between 1 and 50 the statements on lines 430 through 470 cause a plus sign to be printed Y spaces above the X axis. The statements at lines 510 and 520 increment X and continue the loop until the last (51st) values of X and Y are computed and plotted.

The example of listings 2 and 3 are for the cubic equation:

$$Y = \frac{X^3}{4} - 3X^2 + 2X + 1$$

Listing 2a is the table of values for X and Y starting at X equal to −50 with an X increment of 2. The jump to negative values of Y (at X equal to 40) and the exceptional values of Y at X equals −50 and −40 are due to computations exceeding the range of Tiny BASIC. Listing 2b is the same table with the first term rearranged. This keeps the value of Y within the range of precision for variables for a larger range of X. Listing 3 is the plot of the equation with the starting value of X equal to −25, the X increment equal to 1, and the starting value of Y and the Y increment equal to −100 and 4, respectively.

```
:RUN 2, -50, 2

1000 REM THE EQUATION TO BE PLOTTED IS
1010 Y=X*X*X/4-3*X*X+2*X-1
1999 RETURN

X         Y
----      ----
-50       -6083
-40       -4497
-30       -9511
-20       -3241
-10       -571
0         -1
10        -31
20        839
30        4109
40        -5105
50        -8919

:
```

*Listing 2a: The 2 command has been read telling the program to request a value for the X origin and increment. A table of X and Y values for the function of line 1010 is also output. In this example the X origin is —50 and the X increment is 5*2. For the X values of —50, —40, 40 and 50 the incorrect Y value is given. This is because the value that is computed is outside the range of the Tiny BASIC's capabilities. After looking at this table the origins of the program can be changed so that values capable of being used are generated. Another use of the table is to determine the values of the Y axis.*

A small program change in lines 1000 to 1999 allows discontinuous functions to be generated and plotted as in listing 4. These lines are the ones that contain the function to be plotted. Instead of using a simple expression, conditions are set up to determine the starting Y value of the graph on line 1060, the upper and lower limits between which the graph will oscillate on lines 1020 and 1000, and the risetime and falltime of the graph on lines 1040 and 1051. By examining lines 1000 to 1999 in listing 4 it can be seen that the graph is independent of X but is a function of its own previous value.

If you want to use the entire graphing routine as a subroutine you may want to renumber each line. Be sure that there are no statement number conflicts with your routines and that all GOTO, GOSUB, and LIST statements are properly changed. As an alternative you can use statement numbers from 2000 through 32767 and put a "GOTO (first line number of your routine)" at statement number 1.

You will notice that no unnecessary blanks are included within each statement. This makes it slightly more difficult to read the program but does conserve space. Each blank requires a byte of storage. At an average of 2 blanks per line, this saves 90 bytes. As shown in listing 1, the program requires about 650 bytes of programmable memory storage.

The USR instruction allows a Tiny BASIC program access to machine language subroutines and to the rest of the memory space including IO ports. The format for using USR is:

D = USR (P, L, A)

where P is the address of the machine language subroutine, L is a variable passed through the X and Y processor registers, and A is a variable passed through the processor's accumulator. Upon exiting from such a user subroutine and returning to Tiny BASIC, the high order eight bits of D are taken from the Y register and the low order eight bits of D are taken from the accumulator. The D, P, L and A values are in decimal.

Two subroutines addressable by the USR function are included in Tiny BASIC. These are the PEEK and POKE routines. The PEEK routine reads the byte at location L, converts this byte to its decimal equivalent and sets D to this value. The POKE routine loads the low order eight bits of A into the memory location specified by L. The PEEK routine is located at P = 532 for the KIM version and the POKE routine is located at decimal address P = 536. The addresses for the TIM version are decimal addresses P = 4116 for PEEK and P = 4120 for POKE. For the 6800 these addresses are P = 276 for PEEK and P = 280 for POKE.

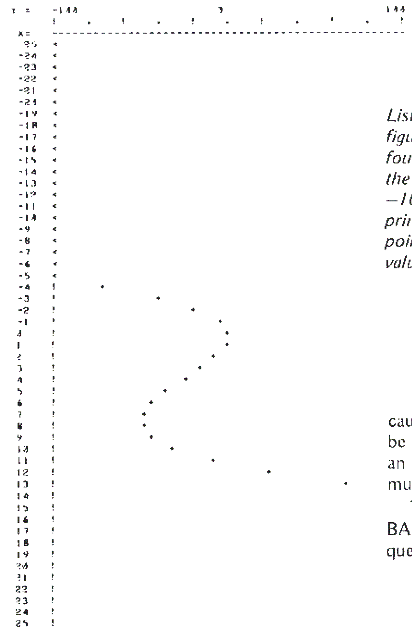
With these routines you can read and write into your computer's IO ports. Set L to the port address, decimal, P to the PEEK address and execute

D = USR (P, L).

```
:RUN 2, -50, 2

1000 REM THE EQUATION TO BE PLOTTED IS
1010 Y=X*X/4*X-3*X*X+2*X-1
1999 RETURN

X         Y
----      ----
-50       26685
-40       -20881
-30       -9511
-20       -3241
-10       -571
0         -1
10        -31
20        839
30        4109
40        11279
50        23849

:
```

Listing 2b: This is a run of the program under the 2 command using the same origin and increment values as in listing 2a. The function to be graphed, however, is still a cubic. Rearranging the order of operations helps avoid overflow errors. With this equation the only value that falls outside the ability of Tiny BASIC appears when X = —50.

```
:RUN 3, -25, 1, -100, 4

1000 REM THE EQUATION TO BE PLOTTED IS
1010 Y=X*X*X/4-3*X*X+2*X-1
1100 RETURN
```



Listing 3: The plot of the cubic equation whose table of values is shown in figure 2a. The graduations on the Y axis are from −100 to +100. A caret is found on the X axis from the values of −25 to −5. This mark indicates that the point on the graph is less than the Y origin, in other words is less than −100 for this example. If the value of Y is equal to 0, a plus sign, +, will be printed on the X axis. If the value of Y is greater than 0, an exclamation point, !, will be printed on the X axis, and a plus sign will be printed at the value.

The port data is loaded into D. Set L to the decimal port address, P to the POKE address, A to the value to be transferred and execute

D = USR (P, L, A).

This causes the value of A to be written into the port.

By using the PB7 pin on a 6530, very accurate pulses from two microseconds to about a quarter of a second in width can be generated directly from Tiny BASIC. If L is set to the base address of the 6530 timer, then the addresses L+12, L+13, L+14 and L+15 determine a factor by which A is multiplied to get the pulse width. These factors are:

Address	Pulse Width
L + 12	1 * A
L + 13	8 * A
L + 14	64 * A
L + 15	1024 * A

If the 6502 is using a one megahertz clock, executing the line

D = USR (536, L + 13,216)

causes a pulse of width 1728 µs (8 * 216) to be generated at PB7. PB7 must be set up as an input (not output) and the 6530 timer must be dedicated to this function.

To return to the TIM monitor the Tiny BASIC program should execute this sequence of statements:

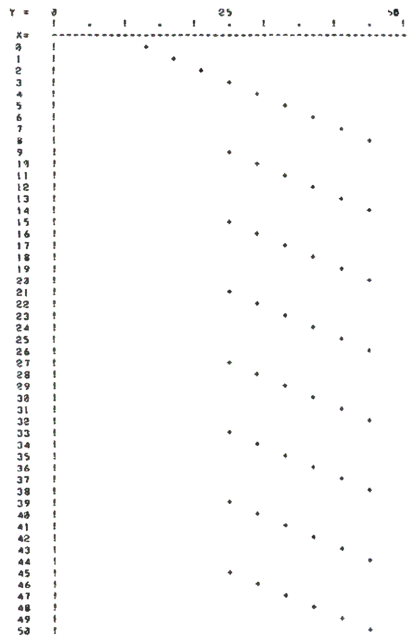
D = USR (536, 0, 0)
D = USR (536, 1, 96)
D = USR (0)

The first instruction loads 00, the 6502 BRK software interrupt instruction, into location 0000. The second instruction in the above sequence loads 96, 60 hexadecimal, the 6502 RTS return from subroutine instruction, into location 0001. The last instruction causes Tiny BASIC to jump to the subroutine at 0000 where the instruction executed by the 6502 causes the TIM monitor to take over. To return to Tiny BASIC, just have the TIM monitor execute the instruction at 0001 by typing "G" at the console. If the sequence of instructions above was executed from a Tiny BASIC program by preceding each instruction with a line number and commanding RUN, the next instruction in the user's Tiny BASIC program will be executed when returning from the TIM monitor.

Saving and reloading programs from paper tape or magnetic tape is easy with Tiny BASIC if you have a debugged version of the interpreter. The procedure is to save the program by typing LIST, turning on the paper tape punch, and pressing the RETURN key on the Teletype. To reload the program press LINE FEED, load the paper tape into the tape reader and turn the reader on. The LINE FEED command suppresses

```
:RUN 3, 0, 1, 0, 1

1000 L=25
1010 REM L IS THE LOWER LIMIT
1020 M=45
1030 REM M IS THE UPPER LIMIT
1040 S=4
1050 REM S IS THE RISETIME
1051 REM  THE FALLTIME IS  1/(M-L)
1060 K=12
1070 REM K IS THE STARTING Y VALUE
1080 IFX=T Z=(K-L)/S
1090 Y=S*Z+L
1100 Z=Z+1
1110 IFY>=M Z=0
1999 RETURN
```

Listing 4: Plot of a sawtooth function. The function is independent of X but is dependent on the previous value of Y. This type of graph may be plotted by changing lines 1000 through 1999 from those shown in listing 1 to those shown at the top of this listing. These changes allow you to specify the Y value that the graph should start at, the upper and lower limits between which the graph will oscillate, and the rise and fall times of the graph. Using this type of evaluation technique a wide variety of discontinuous or oscillating functions can be generated.



output from the interpreter to allow more than one line at a time to be read in without interfering line prompts which can cause the input through the tape reader to lose synchronization. If the LINE FEED command does not suppress line prompts then you may have a bug in the interpreter. In the KIM version check hexadecimal locations 06C7 and 06CB for 13 hexadecimal, X off, and 0A hexadecimal, LINE FEED, respectively. These bytes may be reversed. If they are, switch them and try again. In the TIM version check locations 14C7 and 14CB. In both cases the lower location should contain 13 hexadecimal and the higher location should contain 0A hexadecimal. Only tapes shipped before September 1976 should have this problem.

To obtain a copy of the Tiny BASIC interpreter for your machine, send a description of your system along with a check for $5 to Tom Pittman, itty bitty computers, POB 23189, San Jose CA 95153. In return you will receive the interpreter on a paper tape and a user's manual. The 6502 microprocessor, TIM, KIM, and 6530 are available from MOS Technology Inc, Morristown PA, as is a line of KIM-n products which can be assembled into a complete computer.■