Tiny BASIC for Micro8085 board (manual)

This implementation is based on Palo Alto Tiny Basic. Credits to Dr. Li-Chen Wang and Roger Rauskolb, 1976. Additions and updates by Anders Hjelm, 2020.

Connect to PC using USB port. Open a terminal emulator, e.g. TeraTerm using comm. setup 19200 bps, 8 data bits, no parity. Setup a 50 ms delay at end of line for sending BASIC text files. Board is powered by USB cable. Alt. source is 5VDC from DC plug.

All numbers are integers. Range -32768 to 32767 in signed mode, or 0 to 65535 in unsigned mode.

Variables are A through Z and array @(0)..@(N) where N depends on amount of free memory.

Direct commands (type direct at prompt)

```
LIST [n] Lists all statements (Option from line number)
RUN Execute the current program
NEW Purge all lines of the current program
SAVE Save current program to non volatile memory
LOAD Load program from non volatile memory
EDIT n Edit or change program line with number n
MAN This manual
? Short help
```

Most recent typed line or command can be retrieved by pressing arrow up (or down). Then use left/right arrow to move cursor, delete or backspace for corrections, and type new text where needed. End the EDIT session by pressing <Enter>.

If the first line of a program which is saved in non volatile memory is 'nn REM AUTORUN', the program will automatically be loaded and executed after power on (nn is line number).

Program execution can be terminated by pressing <Ctrl><C>

Statements (type direct at prompt or put them on a numbered line)

```
FOR v=expr TO expr [STEP expr] Setup a repetitive loop, e.g.

FOR I=1 TO 10 (STEP is optional)

NEXT v End delimiter of FOR loop

LET v=expr Assign a value to a variable, e.g. LET A=2*B

IF expr If expr is true, execute rest of line

GOTO expr Jump to line number pointed out by expr

GOSUB expr Jump to subroutine on line pointed out by expr

RETURN Return from subroutine

INPUT ["Text",]v Lets user input a value to a variable

PRINT "Text",v Prints text, values of variables, etc.

PUTC expr Prints the ascii char of expr

POKE addr,value Perform CPU memory write operation

OUT addr,value Perform CPU io write operation

SOD value Serial output data, SPI bus transmit

WAIT expr Mait/halts execution for expr millisecs

BEEP expr Makes sound with duration expr millisecs

XTAL expr Manipulate the freq.division (default 6144)

SIGNED Set signed mode (default)

UNSIGN Set unsigned mode (use for time calculations)

STOP Stop execution of program

REM Use remark for comments
```

Print options

```
"_" or '_' Place text between double or single quotes
#n Numbers printed decimal using n positions (default #6)
$n Numbers printed hexadecimal using n pos (must be $2 or more)
\b Print the ascii char of value b
, (comma) Argument delimiter.
e.g. PRINT "DECIMAL=",#5,A,\10,"HEX=",$5,A
```

Comma at end of PRINT statement means no new line generated. \$ also used for INPUT of hexadecimal values, e.g. \$7FA. Single quotes for assignment of ascii value, e.g. A='!' Use colon for more statements on same line. Parentheses around expressions are optional, use when necessary.

Functions (always returns a value)

```
RND expr A random number between 1 and expr
ABS expr Absolute value of expr
PEEK addr Value of CPU memory at addr
IN addr Value of CPU io read operation
SID Serial input data, SPI bus read

GETC ASCII value of most recent pressed key (or zero)

TIME Value of millisec counter (use UNSIGN mode)
USR addr[,HL,DE,BC,A] Call assembler routine on addr
                         Option preload CPU registers
           Length of current program
FREE
            Amount of free memory
```

```
Arithmetic operands
+ Add > Greater than ! Not
- Subtract >= Gr. than or equal ~ Invert
* Multiply < Less than & And
/ Division <= Less than or equal | Or
% Modulus == Equal ^ Xor
<- Left shift
> Not equal
```

>> Right shift