**Variables**

Variable names can be any length, using the letters A-Z and the numbers 0-9, but must begin with a letter.

Numeric variables are all integer.

String variables use the standard postfix ‘$’ and can be up to 255 characters long.

Single dimension arrays of either are supported.

To create a single value, refer to it on the left hand side of assignment (or FOR). It is not created as part of an expression.

Arrays can only be created using a DIM statement. Trying c$(4) without it will fail.

A-Z are fast variables stored in a fixed place in memory. They are always present and are not zeroed when the program starts. This is deliberate

**Operators**

In precedence order

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| & | | | ^ |  |  |  | Bitwise AND OR and XOR of integers |
| < | > | = | <= | >= | <> | Comparison (integer or string), Return -1 or 0 |
| + | - |  |  |  |  | Arithmetic, strings can be added (concatenation) |
| \* | / | % | >> | << |  | Arithmetic and logical shift. |

**Unary functions**

|  |  |
| --- | --- |
| abs(x) | Returns |x| |
| asc(a$) | Returns ASCII code of first char of a$ |
| chr$(n) | ASCII code to 1 character string |
| instr(s1$,s2$) | Find position of s2$ in s1$ ; returns 0 if not found. |
| left$(a$,n) right$(a$,n) mid$(a$,n) mid$(a$,n,m) | String split up commands. |
| peek(a) deek(a) leek(a) | Byte,Word,Long memory reading. |
| rnd() | Random integer |
| sgn(a) | Returns 0 if a = 0, -1 if a <0 1 if a > 0 |
| spc(n) | Returns n size string of spaces |
| str$(n) or str$(n,base) | Converts integer to string |
| upper$(a$) lower$(a$) | Case conversion. |
| val(a$) or val(a$,base) | Converts string to integer |

**Commands (to date, e.g. these work)**

Commands can be separated by colons, and have line numbers. Every line has a number, but they aren’t necessary (except for GOSUB/RETURN, because PROC isn’t yet implemented).

|  |  |
| --- | --- |
| [let] <var> = <expression> | Assign a value to a variable. LET is optional. |
| assert <expression> | Causes error if expression is false (e.g. assert count=22) |
| clear | Clear variables and stacks. |
| cls | Clear Screen |
| collect | Force garbage collection (will happen automatically later) |
| dim <name>(size),<name>(size) | Dimension arrays, only one dimension at present. |
| end | Ends program. Also exits emulator. If you want to see results use STOP. |
| for <var> = <from> to <to> [step <s>] …. next [<var>] | Standard For Loop |
| gosub <line>/return | Standard GOSUB/RETURN. |
| goto <line> | Standard GOTO |
| if <expr>  …  else <optional>  …  endif | If then else that can spread over several lines |
| if <expr> then <commands> | One line test (no ELSE) |
| link <addr> | Call routine at <addr>. The fast variables A,X and Y are loaded into the processor registers before calling and saved on exit. |
| list [<from>][,<to>] | List program |
| on <expr> goto l1,l2,l3,l4 | Selector, l1 is for 1, l2 for 2 etc. |
| poke a,b doke a,b loke a,b | Write byte/word/long to memory (note, there is no protection at all here) |
| print <expression>’;, | Print things. ‘ is a new line and , does a tab |
| rem “comment” | Comment : *MUST* be in quotes. |
| repeat: ….. : until <expr> | Repeat Loop |
| run | Run the program / Clear variables & stacks etc. |
| stop | Stop the program, also allows viewing of results in emulator. |
| while <expr>: …. : wend | While loop |

**Running**

The interpreter is not interactive yet. To run it, edit the basic.bas file in the build directory and run build.sh (or copy the windows stuff from the directory out)

The line numbers should be in order, even though this BASIC doesn’t need them.

If you use END it will exit the emulator, this is for unit testing and speed tests. If you want to Print stuff and stop it running so you can see it, use STOP (and ESC then exits the emulator)

The benchmarks for speed, based on the very retro PCW Benchmarks are in the Benchmarks file. The interpreter runs at about the same speed as Acorn Archimedes 3010 BASIC *if* the estimate I found of 3.1 MIPS for a 14Mhz 65816 is correct. I haven’t cycle counted it, just instruction counted it, so it may be different in practice.

The TAB key functions as a “Break” key to stop programs running, or interrupt list, as ESC is used to exit the emulator entirely.

Please report all bugs to [paul@robsons.org.uk](mailto:paul@robsons.org.uk) or put them on the git tracker.