**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 of integers or strings.

(Evaluates to 0 (false) -1 (true))

+ - Arithmetic

\* / % >> << Arithmetic and logical shift.

**Unary functions**

Rnd() Random integer

Sgn(a) Returns 0 if a = 0, -1 if a <0 1 if a > 0

Abs(x) Returns |x|

Val(a$) or Val(a$,base) Converts string to integer

Str$(n) or Str$(n,base) Converts integer to string

Chr$(n) ASCII code to 1 character string

Asc(a$) Returns ASCII code of first char of a$

Spc(n) Returns n size string of spaces

Peek(a) Deek(a) Leek(a) Byte,Word,Long memory reading.

Upper$(a$) Lower$(a$) Case conversion.

**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).

Rem “Comment” Comment : *MUST* be in quotes.

[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)

End Ends program. Also exits emulator. If you want to see

results use STOP.

Run Run the program / Clear variables & stacks etc.

Stop Stop the program, also allows viewing of results in emulator.

Clear Clear variables and stacks.

Dim <name>(size),<name>(size) Dimension arrays, only one dimension at present.

Collect Force garbage collection (will happen automatically later)

Cls Clear Screen

Print <expression>’;, Print things. ‘ is a new line and , does a tab

Goto <Line> Standard GOTO

Gosub <Line>/Return Standard GOSUB/RETURN.

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 – this has no protection at \*all\*)

While <expr>: …. : Wend While loop

Repeat: ….. : Until <Expr> Repeat Loop

If <expr> Then <commands> One line test (no ELSE)

If <expr> If then else that can spread over several lines

..

Else

..

Endif

For <var> = <from> to <to> [step <s>] Standard For Loop

Next [<var>]

**Running**

The interpreter is not interactive yet. To run it, edit the basic.bas file in the build directory and run build.bat

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.

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