**Software: Post 28**

Having written a bit more of the MINOL interpreter I’m not convinced by the language or the byte-ness, so I’m going to go back to VTL-2 , (probably). Most of the code is written (it’s on GitHub) and it’s got LET left to do, and implement some of the I/O and Memory Space management.

I’ll have a poke about for other micro-languages – Tiny BASIC was always a possibility, and M5 (an RPN language for the NASCOM-1), and of course FORTH.

But it will probably be VTL-2. I think I will rewrite what I’ve done already (the expression evaluation stuff), learning from mistakes, you know ☺ I might even go as far as writing some proper design stuff, rather than ad-hocing it.

Sometimes I’ve lost code and rewritten it, but it’s never taken very long, just handle turning, and it’s often a lot better. I have a Scelbi replica design for Arduino which works at the cycle level, i.e. you can operate it like a real Scelbi, loading programs in by executing code on the toggle switches, and somehow mucking around with GitHub it went completely AWOL. Somehow I hadn’t got another backup of it either …….. I rewrote it completely in a couple of hours or so and was much happier with it.

VTL-2 is quite interesting for all its brevity. Some neat ideas never take off. I remember in Personal Computer World, Issue 1 I think, there was a Z80 assembler which you programmed by writing things like C->(HL) ; it was quite neat. The only thing I’ve seen like it since (though I think it predates it) is in an obscure Bell Microprocessor which has a similar ‘C’ low level syntax.