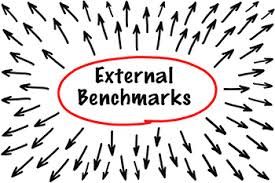
**Software 31**

So, I now can GOTO, or as VTL-2 does it # = - the only outstanding “left special” is the array code.

Back in the dim and distant past, the first thing that magazines would do with the latest home computer is benchmark its internal ROM BASIC. Virtually every machine released over a period of about 1978-1988 ish had a built in ROM BASIC, with a few exceptions – I owned a Sharp MZ-80K where it had to be loaded in from tape, and the Jupiter Ace had FORTH – but most had BASIC. When the machines arrived that had floppy disks as standard – primarily the Amiga and Atari ST – the BASIC interpreter became something provided on a floppy disk.

The UK magazine Personal Computer World had its own set of benchmarks, which were BASIC programs that were run and measured to see how long they took. The simplest one was this :

100 FOR I = 1 TO 1000

110 NEXT I

I’ve actually found them here <http://www.geocities.ws/peterochocki/computers/pcwbm.html>

To give some idea ; a ZX81 4.5s, a Speccy 4.4s, a C64 1.2s,a BBC Micro 0.8s. The slowest machine would be a ZX81 in slow mode which would be about 18s or so (estimate).

The variation is quite surprising. The first thing on the list is a 68000 at 8Mhz (HP Integral PC) which takes 1.9s ; it is quite remarkable that the BBC Micro (6502 at 2Mhz) should be over twice as fast given that both are operating in floating point. The ZX Spectrum BASIC is based on Nine Tiles original ZX80 BASIC which is designed to operate with 1k of RAM and doesn’t care about speed – a typical Z80 machine of the time clocked at 3.5-4Mhz would probably run the first benchmark in about a second.

So, I have benchmarked it. I can’t actually run Benchmark 1 – no FOR/NEXT loop, but I can run Benchmark 2, which is the same thing with an IF … GOTO instead , in VTL-2,

100 A = 0

110 A = A + 1

120 # = (A < 1000) \* 110

*Line 120: Returns 110 if A < 1000 and 0 otherwise ; # (GOTO) does not execute if its r-expr is zero.*

Which runs in about 40 seconds. This isn’t bad, really – it’s a slow processor, it’s running at a quarter of its actual speed anyway (just sticking an SC/MP 2 in and clocking it up to 4Mhz will get that down to 10 seconds) and it’s not written for efficiency by any means, it’s written more with clarity in mind (honest – an optimised one would look like spaghetti – and it wouldn’t be that much quicker anyway). The obvious optimisation is to take the + and – calculations out of the Maths library which saves a whole pile of push/pop/call/load Px immediate code I would think. Might try it ☺.

By comparison : ZX81 6.9s, Speccy 8.2s, C64 9.3s, Atari 800 7.3s

So not too bad really. If I conveniently forget they are all doing it in floating point and I’m doing it in 16 bit integer ☺ The SC/MP is a notoriously slow processor though – having attempted to emulate processors on an Arduino, it can cope with 4004/4040/8008, the RCA1802 and the SC/MP and that’s about it. I wrote an Apple 1 emulator (more as a protest than anything else) and it ran at about 15-20% of the speed of the real thing (which didn’t matter in most cases because of the slooooow TV Typewriter output).

For depressing comparison ; 2400Mhz Celeron, running QBASIC, 0.011s. I have reduced this PC to a wreck of its original finery.