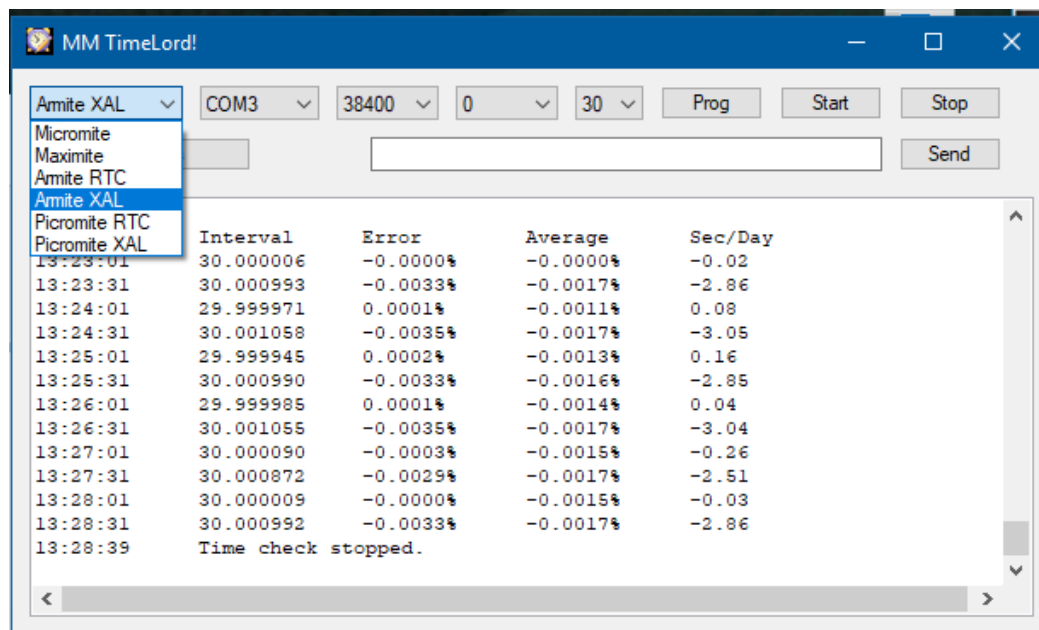


MMTimeLord – when every (milli)second counts.

MMTimeLord is a Windows application for testing and calibrating you micromite family devices.

MMTimeLord uses “QueryPerformanceCounter” which is the Windows equivalent to TIMER. The accuracy obtained is only as good as the reference used but there are ways of calibrating it.

Place the EXE in any convenient location. It will create an INI file so the folder has to be writable.



Run MMTimeLord and choose the device type.

Select the COM port from the pulldown list. If the device wasn't plugged in when you started TimeLord, press Scan Ports to refresh the list.

Set the baud rate.

If calibrating a micromite or Armite RTC, select the calibration value.

For micromites this will cause “OPTION CLOCKTRIM x” to be sent to the micromite when programming, where x is the value chosen from the list.

For Armite RTC, this will cause “OPTION RTC CALIBRATE x” to be sent to the micromite when programming, where x is the value chosen from the list.

Select the desired timing interval (in seconds). For RTC calibration, the only valid options are 30 or 60 seconds.

“SEND” will send any text in the adjacent box to the connected micromite.

“PROG” will send the program to the micromite after deleting any existing program and set the clock option if appropriate.

“START” will RUN the micromite program and start waiting for the data to arrive.

There is approximately a one second wait and during that time your PC may run flat-out. Once the first data has arrived, the program waits until 500mS before the next due data before going into a tight loop again.

There are two different programs sent depending on whether you chose a RTC test. For the normal timing test SETTICK is used:

```
>
OPTION CLOCKTRIM 0
>
NEW
>
DIM interval, n
interval = 30
SETTICK 1000, ticker
DO:LOOP
END
SUB ticker
n = n + 1
IF n MOD interval = 1 THEN PRINT "*"
END SUB
Saved 129 bytes
>
```

For the RTC test, the micromite program waits for the RTC to tick over every 30 or 60 seconds then sends “*” to the PC

```
>
NEW
>
t$=time$
secsa = (val(right$(t$,2))+1)mod 60
secsb = (secsa + 30) mod 60
sec$ = str$((secsb + 1) mod 10)
DO
do : t$=time$ : loop until val(right$(t$,2)) = secsa or val(right$(t$,2)) = secsb
do : t$=time$ : loop until right$(t$,1) = sec$
PRINT "*"
LOOP
Saved 208 bytes
>
```

There are different ways of sending the program depending on the device.

Maximates use AUTO ... ctrl-C

Micromites and Armites use AUTOSAVE ... ctrl-Z

Picromites use AUTOSAVE ..., ctrl-C

Armites and Picromites read the RTC when TIME\$ is used so we are testing the actual clock.

Micromites read the RTC (where fitted) on start-up and with GETTIME so a RTC test is really only doing the same as SETTICK.

The best settings are 30 seconds and let it run for 60 minutes. That allows the timing fluctuations caused by USB to average out.

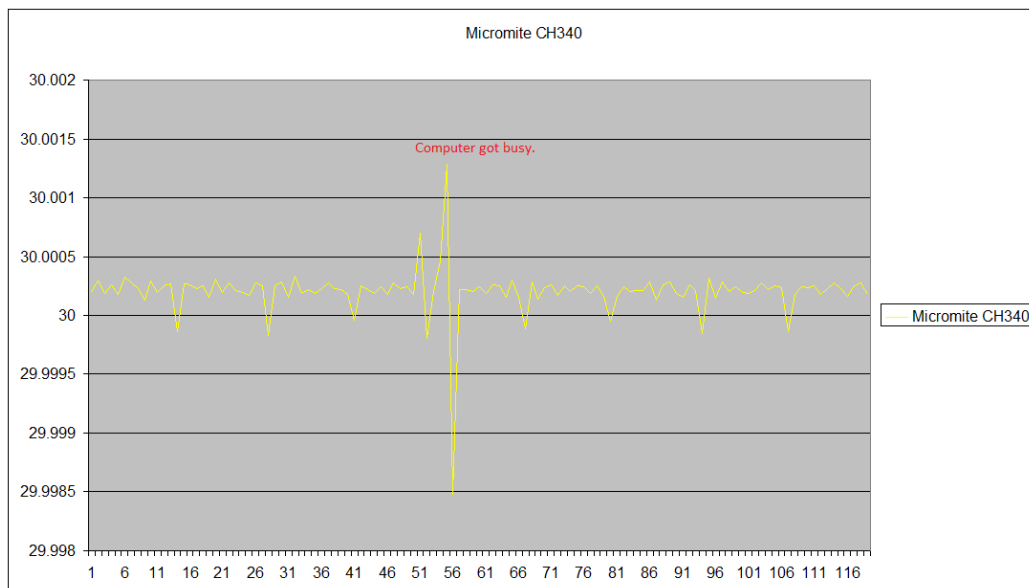
You can highlight the data and copy it then paste into a spreadsheet for analysis.
A sample spreadsheet is included and you can paste over the data for a quick chart.
Adding charts to MMTimeLord is next on the list.

The sample data is from a micromite which I heated up with a finger before hitting it with the 'freeze' spray.



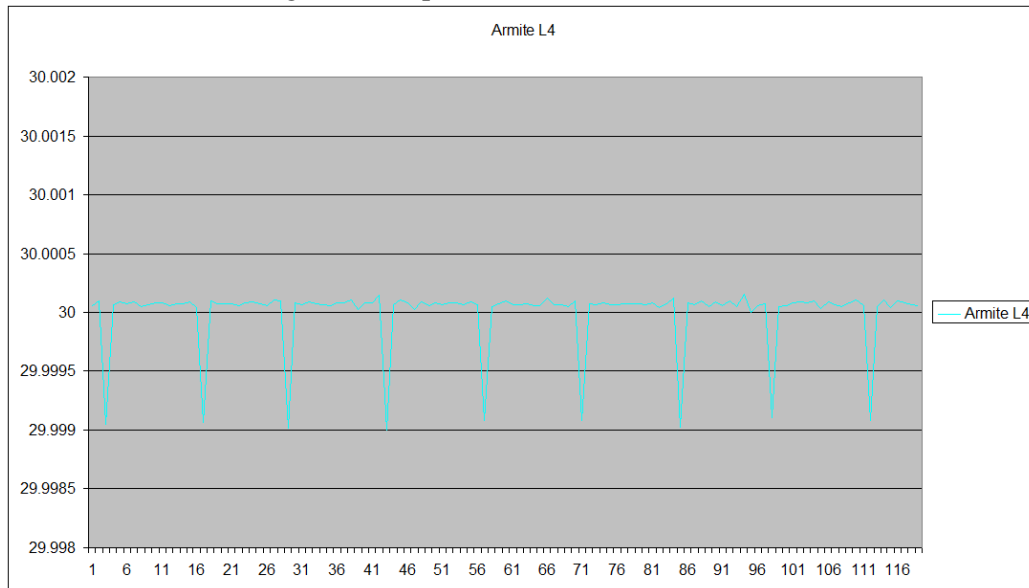
The accuracy of the readings depends on the reliability of the USB chain.
This is where the problems begin.
Different USB – serial modules perform very differently.

To test the various devices I had available, I used a DS3231 RTC with a 1Hz output.
I ran the test for one hour with an interval of 30 seconds.

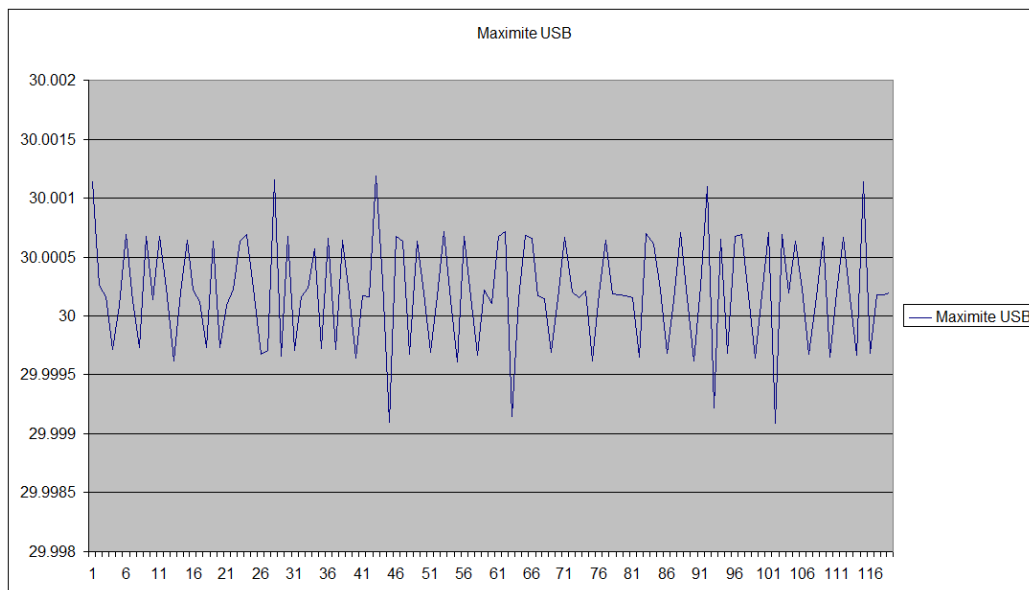


The best result was a micromite with a CH340 adapter – standard deviation 0.219mS

The glitch in the middle was caused by me using the computer during the run. It doesn't affect the average so not a problem in real life.

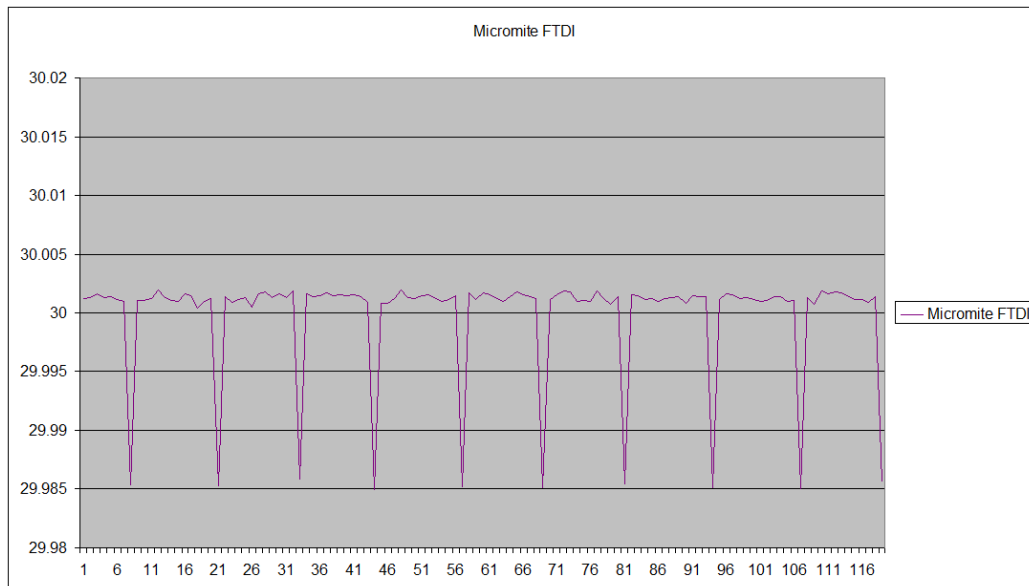


Next came the Armite with standard deviation of 0.273

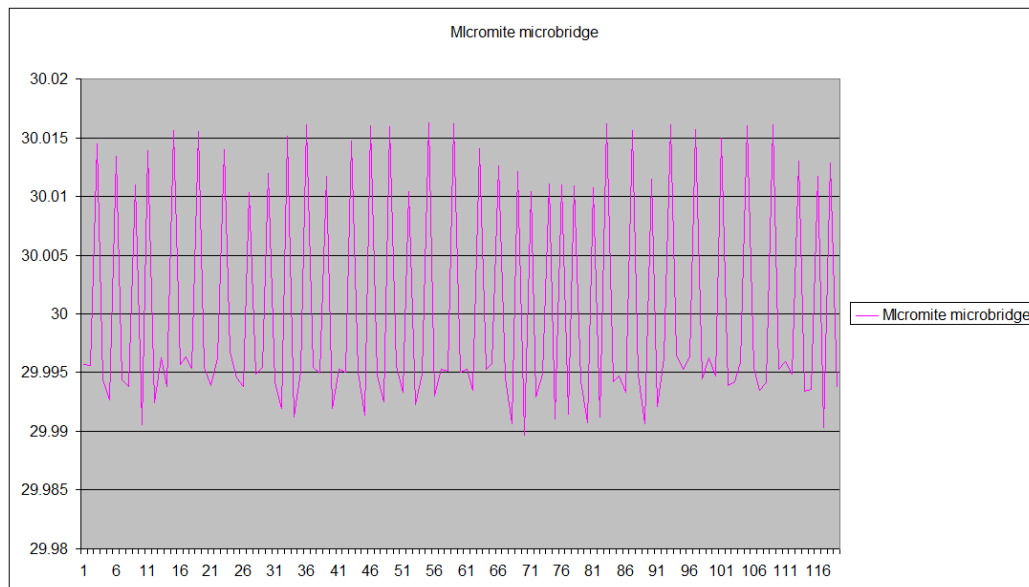


This is an original Maximite using it's built in USB. Standard deviation of 0.451mS

Now we get to the problem children.
Note the different vertical scales compared to the previous runs.



The FTDI adapter has a 16mS correction every 6-7 minutes. The average over the hour is OK but the correction doesn't look nice. SD 4.482mS
That's 10 times the Maximite!



The microbridge doesn't perform very well at all as far as timing tests go. Standard deviation was 9.286mS

Two things are of interest.
The microbridge and FTDI adapters don't use external crystals. All the other tested here do.

The 'correction' seems to be around 16mS each time. This is eerily close to the Windows tick time of 16.67mS

The 'good' adaptors seem to do a small correction every 6-7 minutes.

Calibrating you PC timer.

The easiest way to check the accuracy of your PC high performance counter is with a GPS 1Hz pulse.

Install this program on a micromite with the GPS pulse connected to pin 15.

```
1  ' GPS as pulse
2  DIM n, interval
3  interval = 30
4  ' GPS 1Hz pulse to pin 15
5  SETPIN 15, INTL, tick, PULLUP
6  ' OPEN "COM1:9600" AS #2 'only to check GPS is OK
7  DO
8  '   print input$(1,#2);
9  LOOP
10 ' close #2
11 SUB tick
12 IF n MOD interval = 1 THEN
13   PRINT "*"
14 ENDIF
15 n = n + 1
16 END SUB
17
```

In MMTimeLord, set the interval to 30 seconds and START. Do not overwrite the program you have just loaded.

After a while you can use the average percent error to set the PC clock ppm.

If the average error reads 0.0008% then put '8' into the ppm box.

Use a negative number if the % error is negative.

It is always best to allow the PC (and GPS) to warm up fully before doing the calibration and a good (timing wise) adaptor makes live easier.

You can spend a lot of time trying to chase a frequency that IS going to vary as conditions change.