Next I decided to abandon the idea of the polystyrene insulation - insulation doesn't stop the temperature inside the VFO compartment from rising, all it does is slow that the rise down. Ultimately, when the VFO is used inside a transceiver and everything boxed up inside an enclosure, and after waiting for the temperature everywhere to reach its final value, eventually the VFO will end up at the same temperature, regardless of whether it is in a polystyrene-insulated box or not. The polystyrene just makes the whole thing take longer. It seems to me that it would

be much better to minimise the temperature dependency by balancing the temperature coefficients, than to try to slow down the temperature rise.

This new version abandons the polystyrene insulation, the box contains just two DF96, one to be used as 6.0-6.5MHz VFO and the other intended to be used as a buffer. The rest of the oscillator (20MHz crystal oscillator, mixer, buffer) are less critical and can be outside on a separate subchassis.

As it turned out, the project diverged slightly at this point, and became more about data logging, and less about stable VFO's

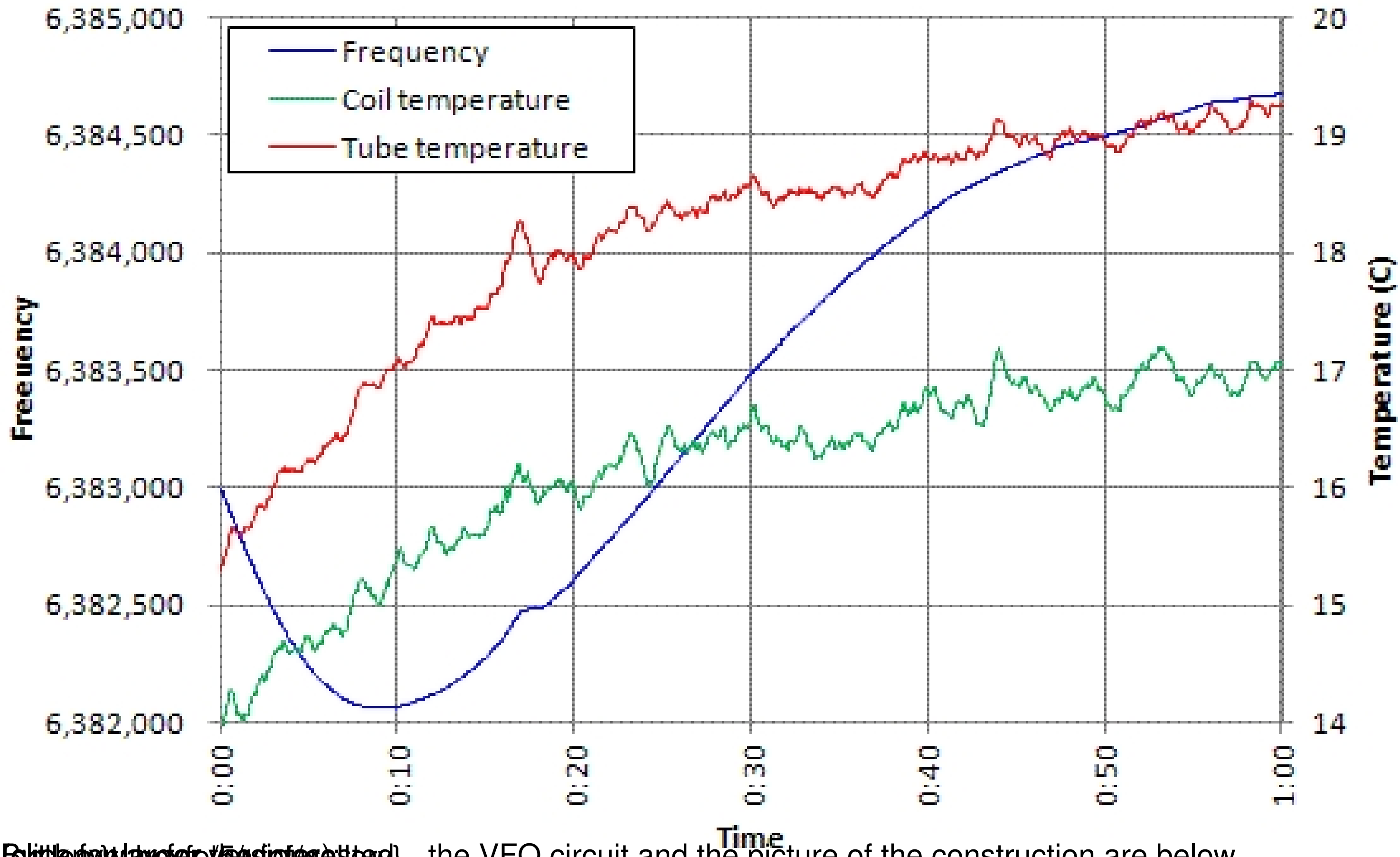
Now finding myself tired of an hour sitting next to the frequency counter and writing values into the computer, I automated this task. An Arduino Uno board is used to log frequency and six analog channels at 2 second intervals, to a micro-SD card (see photo). Subsequent data processing and charting of the resultant .csv format data file can be done in MS Excel or OpenOffice.

Three of the analog channels are used for measurement of voltages: the HT voltage from the rectifier tube, the 150V regulated voltage line from the VR150/30 gas-filled regulator tube, and the 1.4V ac filament voltage (obtained from the 6.3V ac transformer winding with a series resistor). A half-wave germanium diode rectifier is used for the filament (ac) measurement.

The other three analog channels are the most useful, for these I connected a TMP36 temperature sensor chip. A sensor was taped to the DF96 oscillator valve, one fixed near the inductor, and one taped to the PSU rectifier.

This chart (see below) shows a frequency drift curve over 1 hour (left Y-axis) alongside the temperatures at the inductor and DF96 tube (right Y-axis). The pattern is similar to that seen in previous experiments - an initial drift while the tube internals reach their stable temperature, followed by temperature dependencies in the oscillator components. The temperature rise in one hour was only 4C at the valve envelope and even less at the inductor. A long way from the original boxed mixer-oscillator using 12AT7 and such! Yes there's 1kHz drift down then 2.5kHz up drift... but remember, this spin-off project stopped being about the VFO and was all about the logger and getting that working nicely before proceeding with more VFO tests.

The logger works very nicely, I think it will be a great help having the hard work of the logging done accurately and automatically.

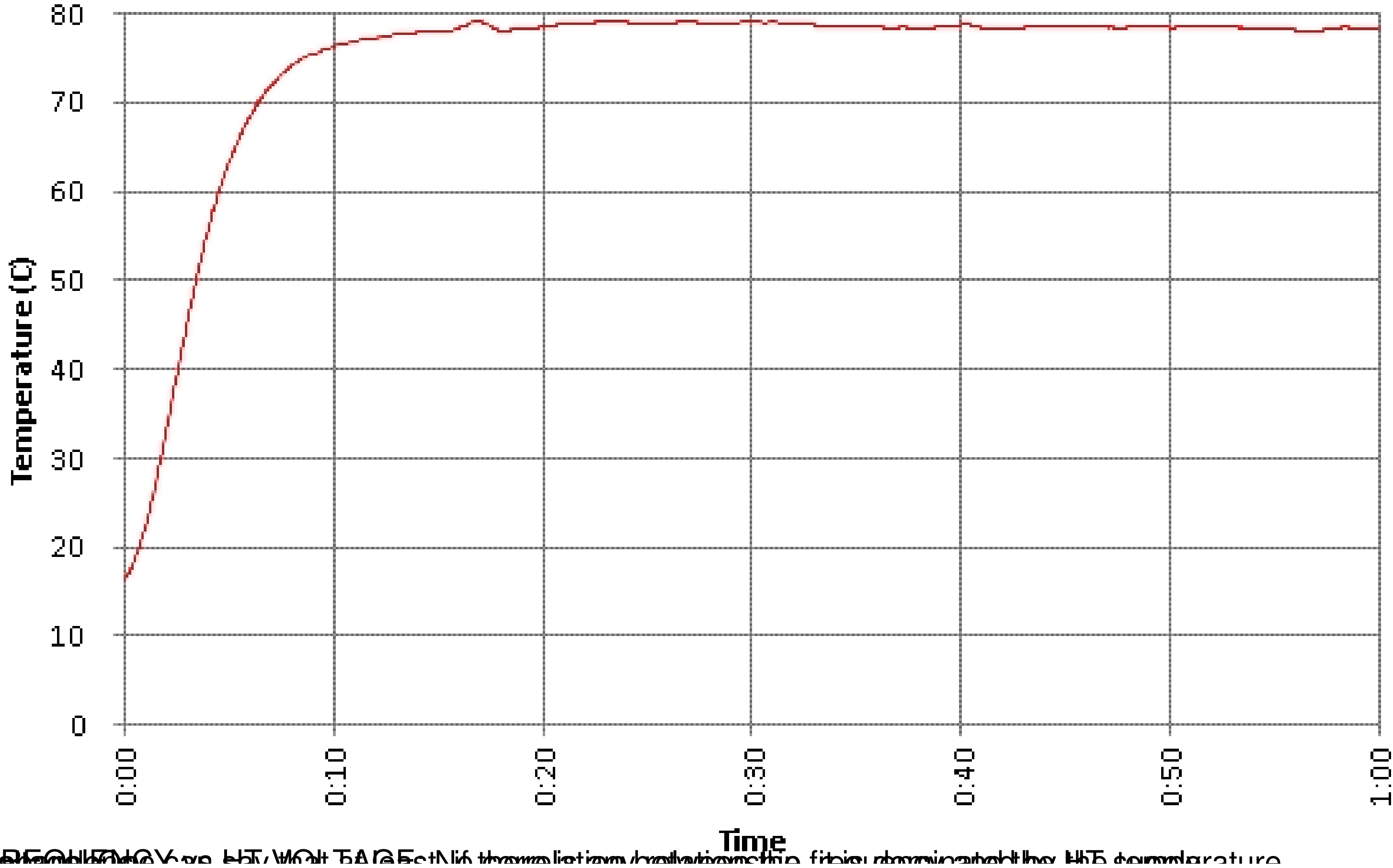


Click for larger version(s)... the VFO circuit and the picture of the construction are below

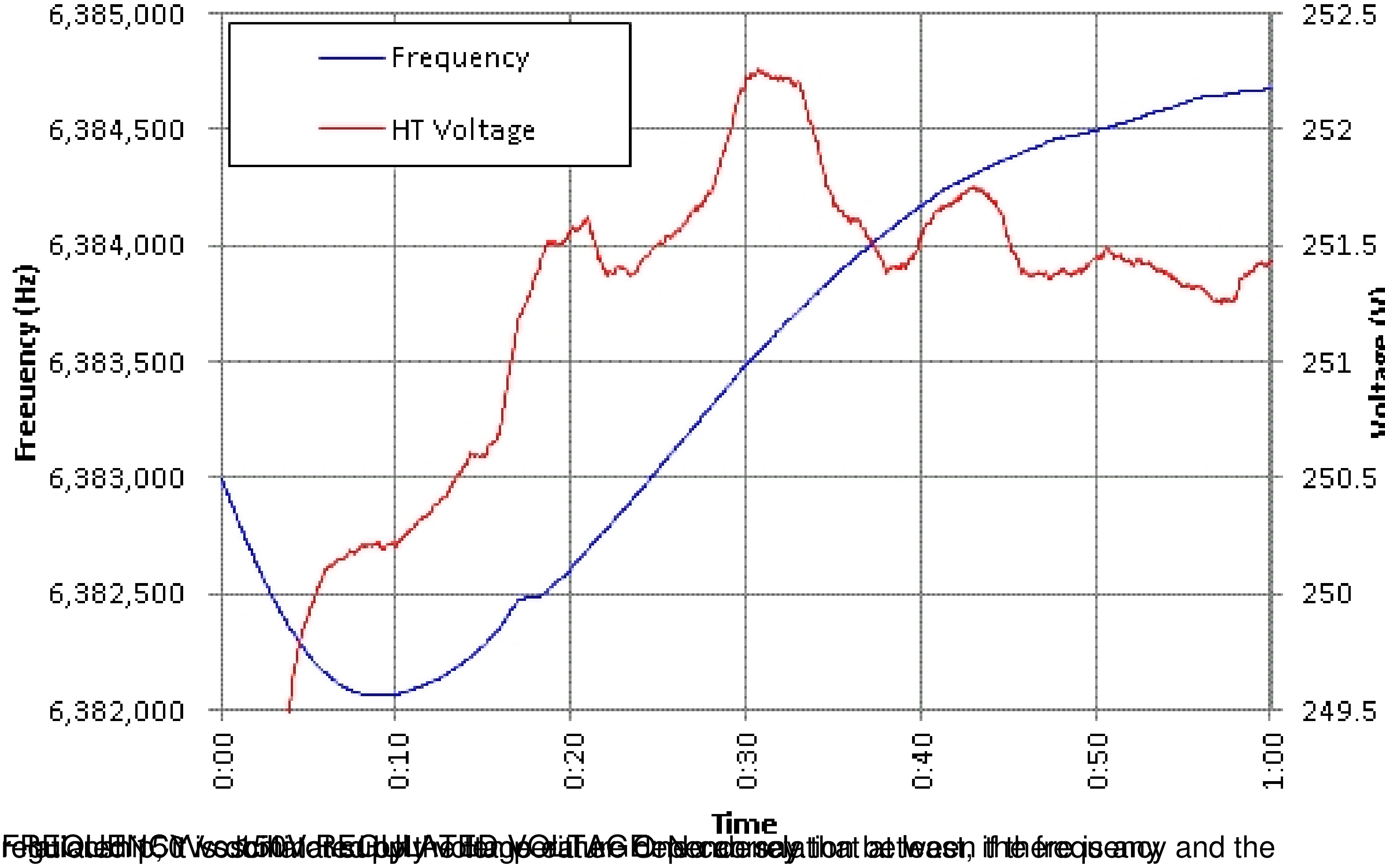
More measurements...

Now some other charts, from the same 1-hour VFO monitoring log:

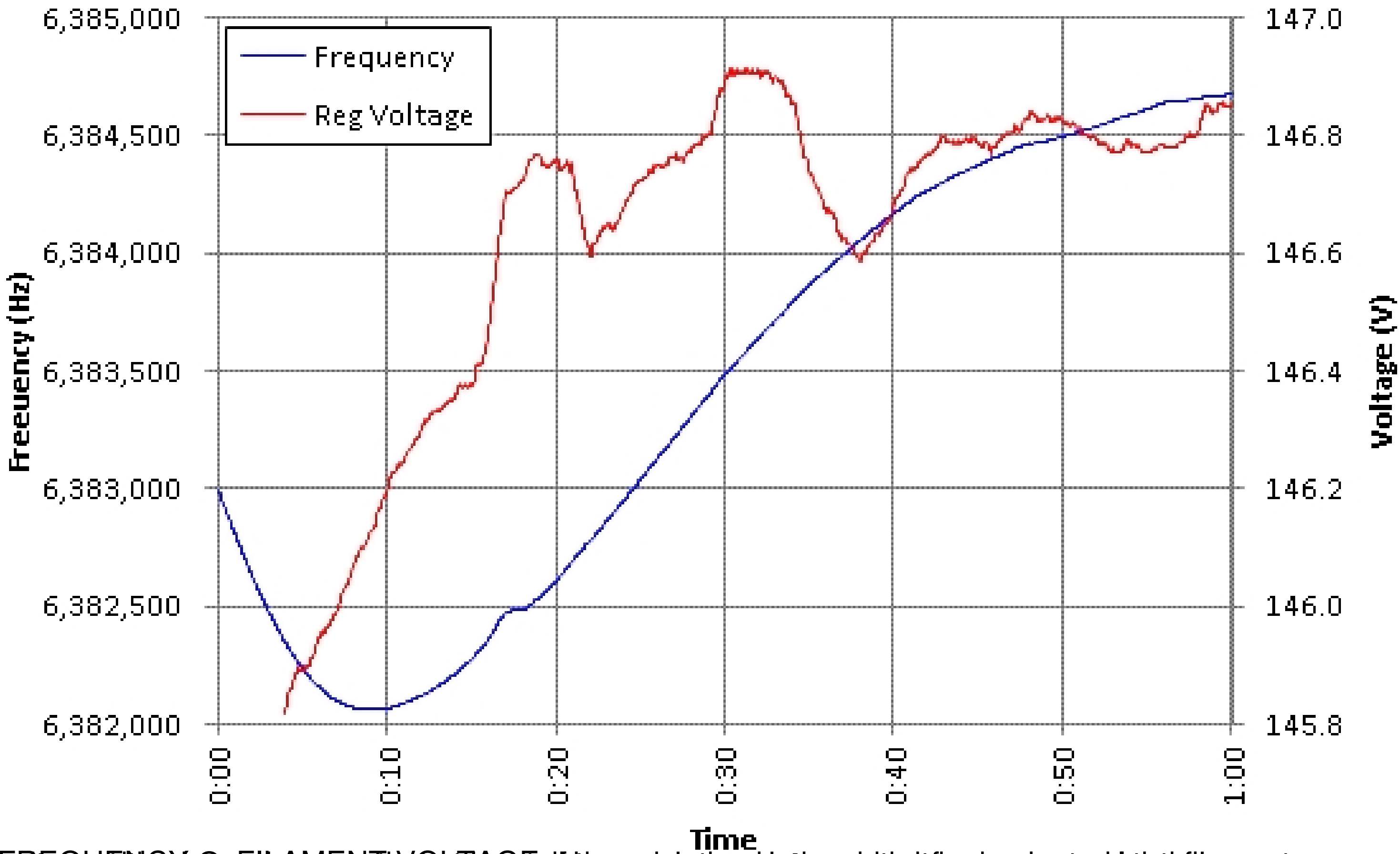
RECTIFIER TEMPERATURE: surprised and interested, to see that the temperature at the rectifier valve envelope rose to nearly 80C!



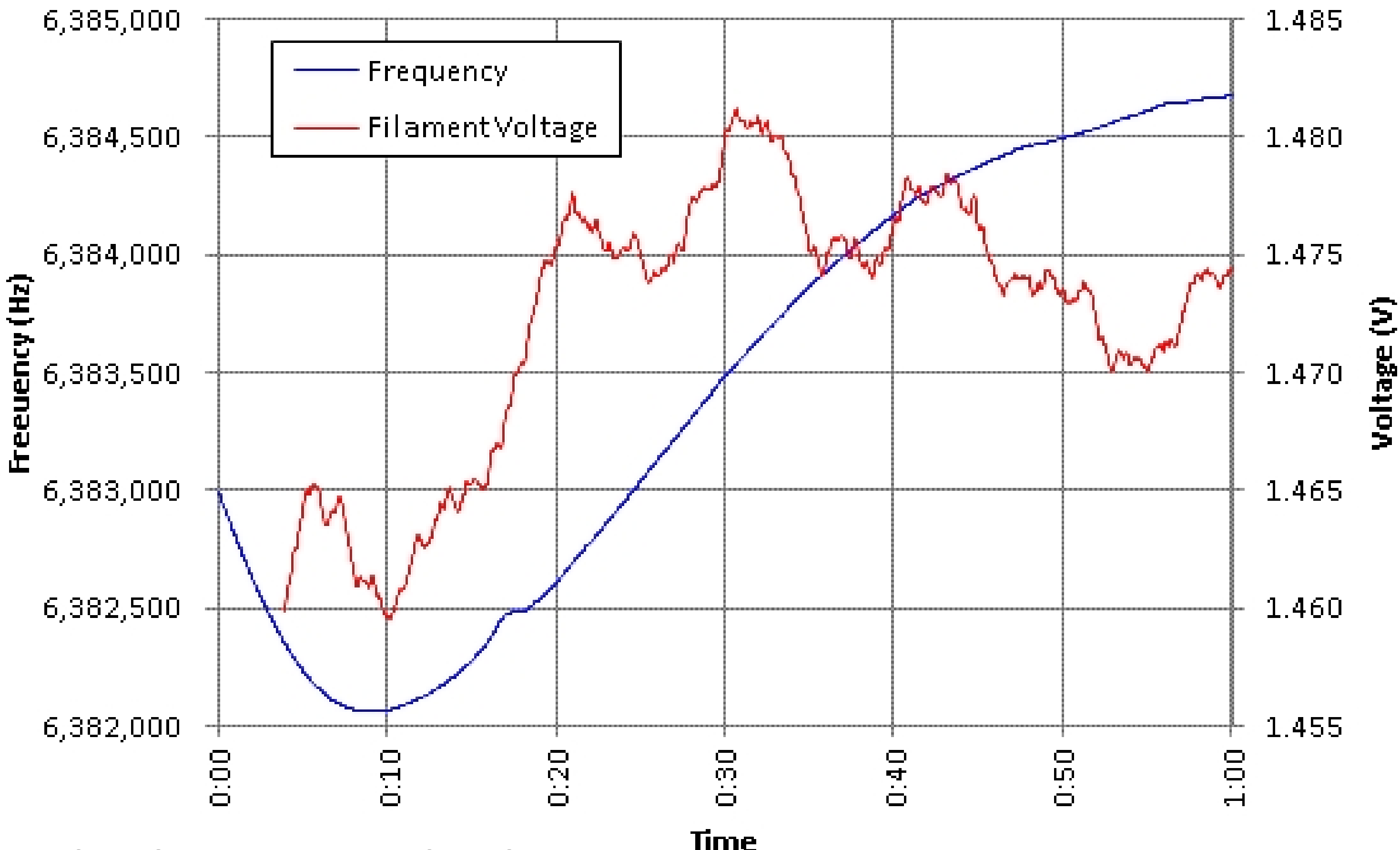
FREQUENCY vs HT VOLTAGE



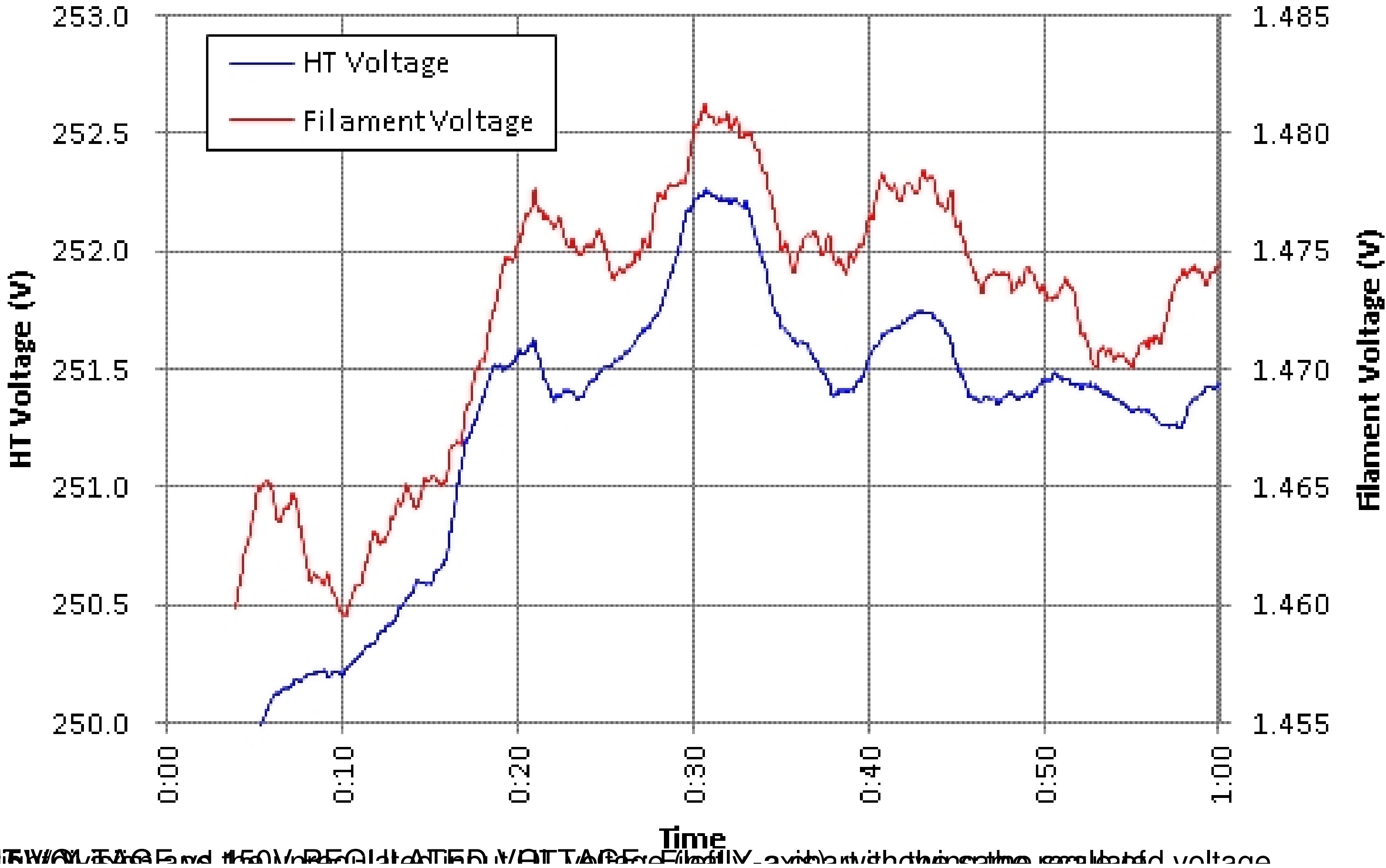
FREQUENCY vs 150V REGULATED VOLTAGE ... if there is any and the



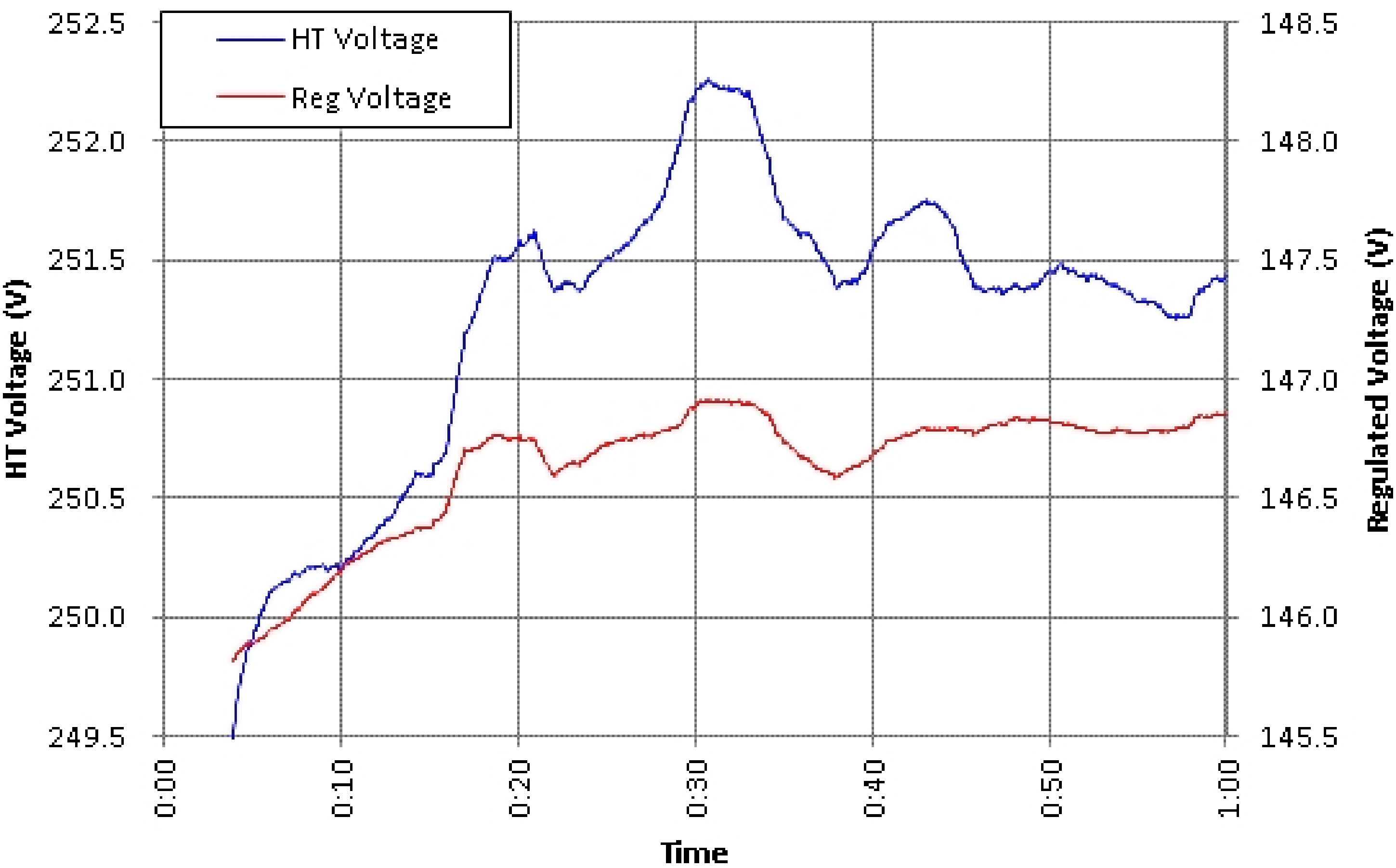
FREQUENCY & FILAMENT VOLTAGE: No correlation [illegible] between [illegible] the frequency [illegible] and the filament



HT VOLTAGE [illegible] FILAMENT VOLTAGE [illegible] (right Y axis)



HT VOLTAGE and 150V REGULATED VOLTAGE



Further observations

One worry I have, is that the analog voltage measurements are only as good as the voltage reference of the analogue to digital converter (ADC) - which in this case is just the 5V supply of the Arduino board, which itself comes from the USB line of my laptop computer. Any change in that 5V supply will directly show up proportionately in the measurements. Only a 1% change in the 5V supply voltage would change the measured HT voltage by 2.5V. Some of the above variations may well be due to fluctuations in this USB powered voltage.

However, this would also affect the temperature measurements. The TMP36 temperature sensor has an output voltage that changes by 10mV/C and is 0.75V at 25C. A 1% change in ADC reference would therefore cause a mis-reading of temperature by 0.75C. The overall temperature changes in the chart are 4C maximum and I think that this error in temperature would be visible.

I might still want to check this though - I would need to either use an accurate voltage reference to the Arduino microcontroller ADC, or to use one of the analog channels to measure an accurate known regulated voltage.

Another concern is the accuracy of the frequency counter timebase, which is a 32.768kHz watch crystal. To get more accurate frequency measurements, I should use my 10MHz GPS-locked frequency reference. In the case of this measurement the drift was +1.5/-1kHz which would be much greater than any crystal drift - but when I get to trying to balance temperature coefficients the drift will hopefully be a lot less so I should look at improving the timebase accuracy.