1. Introduction

Here follows my modification to a FE-5680B Rubidium frequency standard made by Frequency Electronics Inc.. There are many different models available and they began appearing on Ebay a few years ago as second hand equipment. As they've been snapped up, so the price has increased also and today the FE-5680A with 10MHz output sells for approx. $200 USD. I didn't have $200 so I bought the budget version FE-5680B with 1 pulse-per-second output for $34 USD. The following details how I modified it to output a 10MHz signal suitable for use as an external clock reference for lab test equipment.

In essence it builds on Ruby to provide the following:

- Rack mount case and power supply so it can be plugged straight into mains
- 4 x 10MHz buffered square wave outputs
- 1 x output of 1 pulse per second
- LEDs indicators for Power, Frequency Lock and a 1pps blip
- Protection for connected test equipment

Andre.lubbock@gmail.com, Wellington, New Zealand, September 2015

Contents

2. References

Credit to the creators of the following information that helped me with the project

1. Finding the 20MHz signal. Thanks to a contribution from h572
http://www.eevblog.com/forum/testgear/fe-5680b-1pps-to-10mhz/

2. Idea for the case
http://gerrysweeney.com/build-a-10mhz-rubidium-frequency-standard-and-signal-distribution-amp-for-my-lab/

3. Circuit values
http://www.rhodiatoce.com/pics/time-nuts/FE-5680A/FE-5680A_schematics_v0.1.pdf

4. Ruby specs.
http://freqelec.com/rb_osc_fe5680a.html

5. Technical Manual
http://www.wa6vhs.com/Test%20equipment/FREQUENCY%20STANDARDS/FE-5680A/5680%20TECH%20MANUAL.pdf

3. The Design

It arrived through the post looking like this; it has a DB15 connector that carry all the signals and no RF connector. Image found on google of FE-5680A.

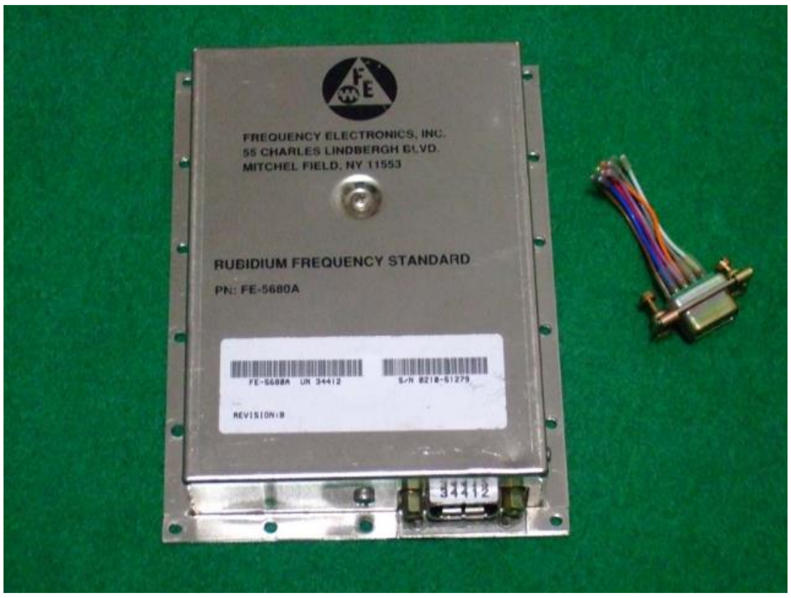

FREQUENCY ELECTRONICS, INC.
55 CHARLES LINDBERGH BLVD.
MITCHEL FIELD, NY 11553
RUBIDIUM FREQUENCY STANDARD
PN: FE-5680A
FE-5680A UN 34412
S/N 0210-51279
REVISION:B

Figure 1. Ruby Original

There are 3 areas to the project:

- Modifications to the frequency standard itself
- External circuitry to condition the signal and drive LEDs
- Case hardware

3.1 Frequency Standard Modifications

The only signal natively available on Ruby is a 1 pulse-per-second. The following modifications filter a 20MHz and bring it out on the external DB15 connector. The internal 5V supply is also brought out to power external circuitry.

3.1.1 *5V Output*

The add-on circuitry that I built is powered from 5V generated within Ruby. This 5V needs to be brought externally on the DB15 connector so just bridge the highlighted pads.

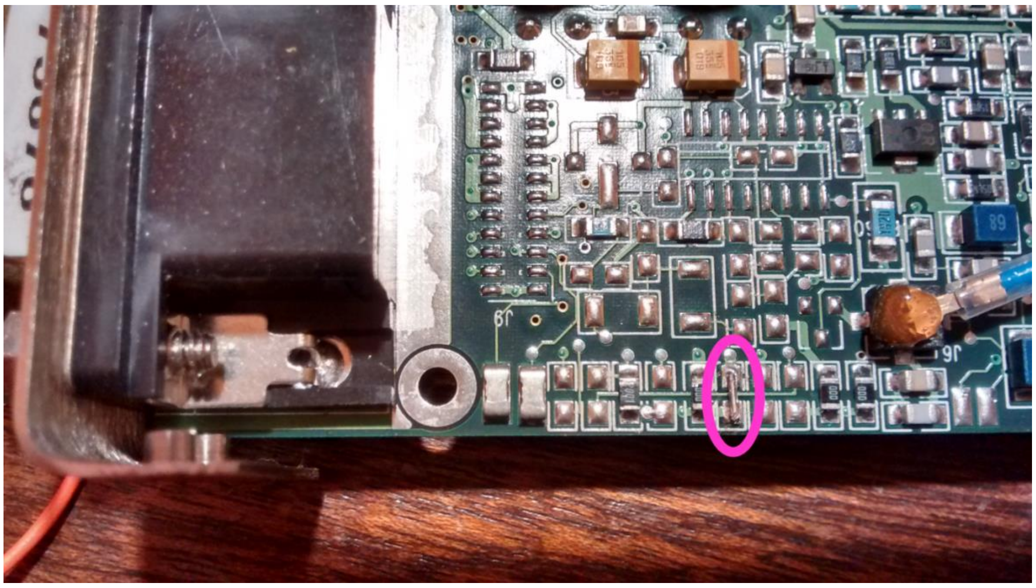


Figure 2. 5V output

3.1.2 *20MHz output*

Disconnect 10MHz from DB15 and connect 20MHz in its place (10MHz only appears briefly at startup). Rotate the 15Ω resistor on its pads to disconnect from 10MHz and connect it to 20MHz with a wire to J8. Refer to Figure 4.

3.1.3 *Band Pass Filter*

Change frequency of band pass LC filter from 10MHz to 20MHz by changing C1 to 56pF and L1 to 1µH. (Note original filter is actually tuned for 9.8MHz with a series LC of 2.2µH and 120pF).

The 10MHz square wave from the CPLD pin 49 is modified to a sine wave with this filter before appearing at the DB15. Without this mod a 20MHz filter acting on the 10MHz signal attenuates the signal to 500mV pk-pk which won't trigger the D-types.

The original filter with 10MHz input swings from -4V to +8V. The 9.8MHz filter in combination with 15 Ω series may have been chosen to attenuate the signal.

I could have bypassed the filter and connected the 20MHz square direct to the D-types but it was a really messy signal and the filter was already on the PCB. See Figure 6.

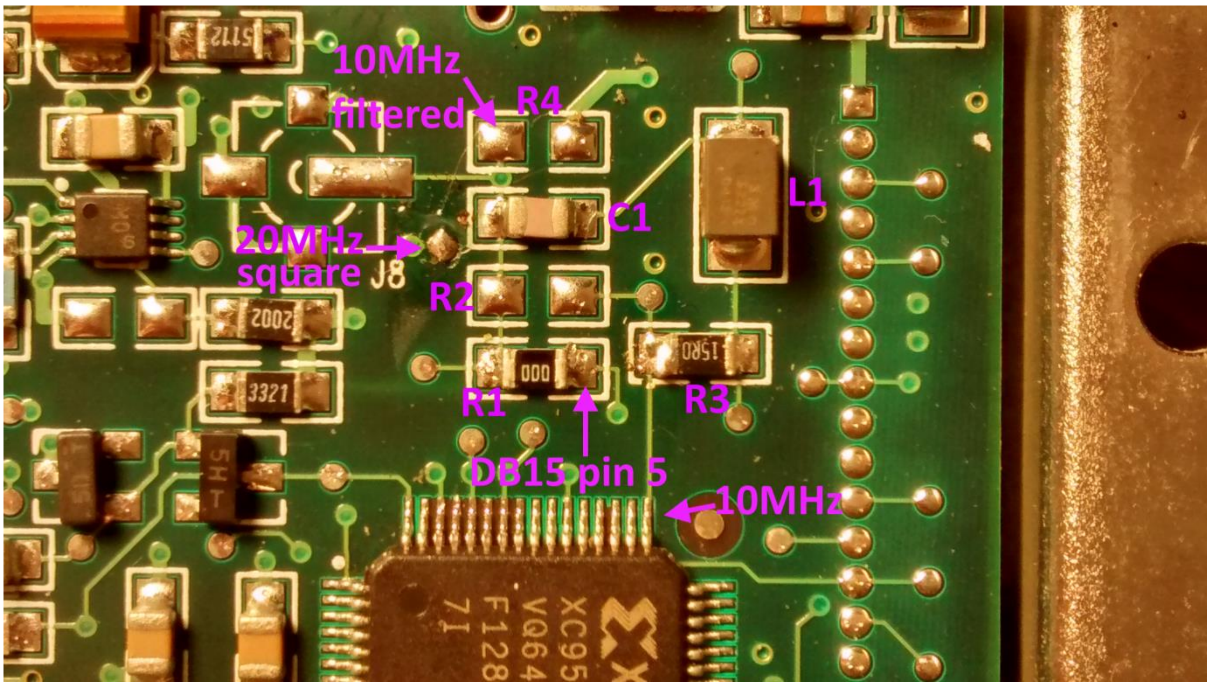


Figure 3. **Original Ruby Circuit**

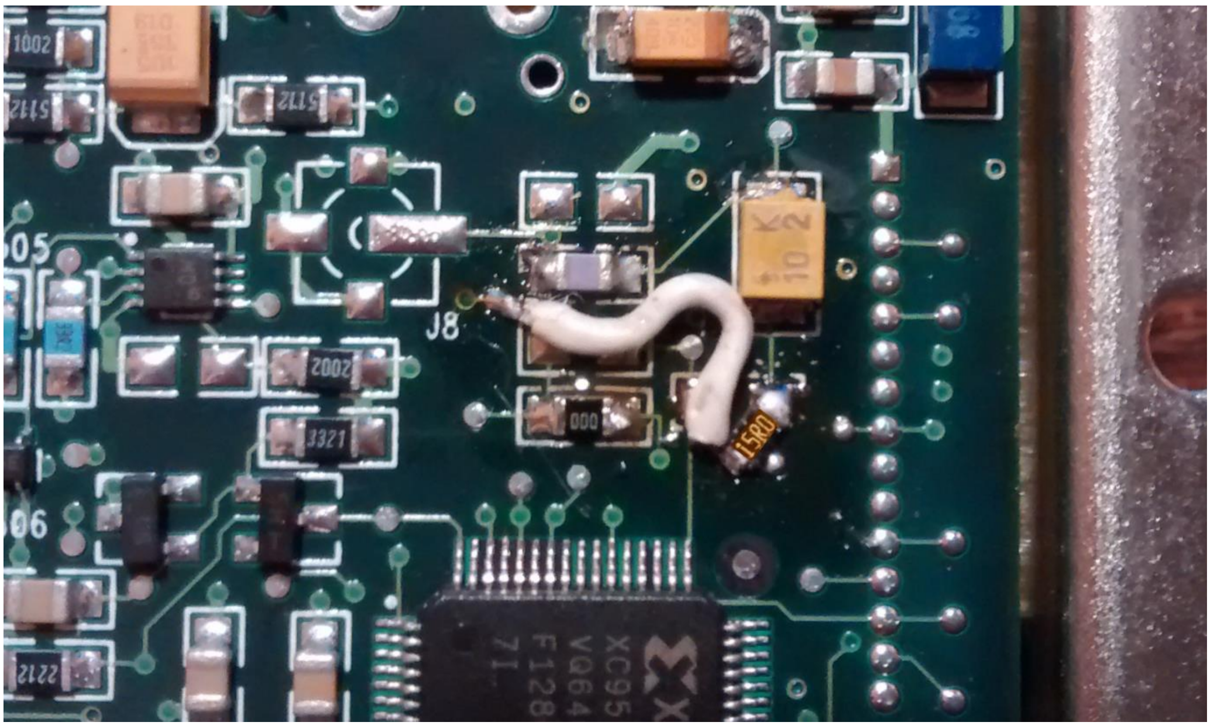


Figure 4. **Ruby Modified for 20MHz**

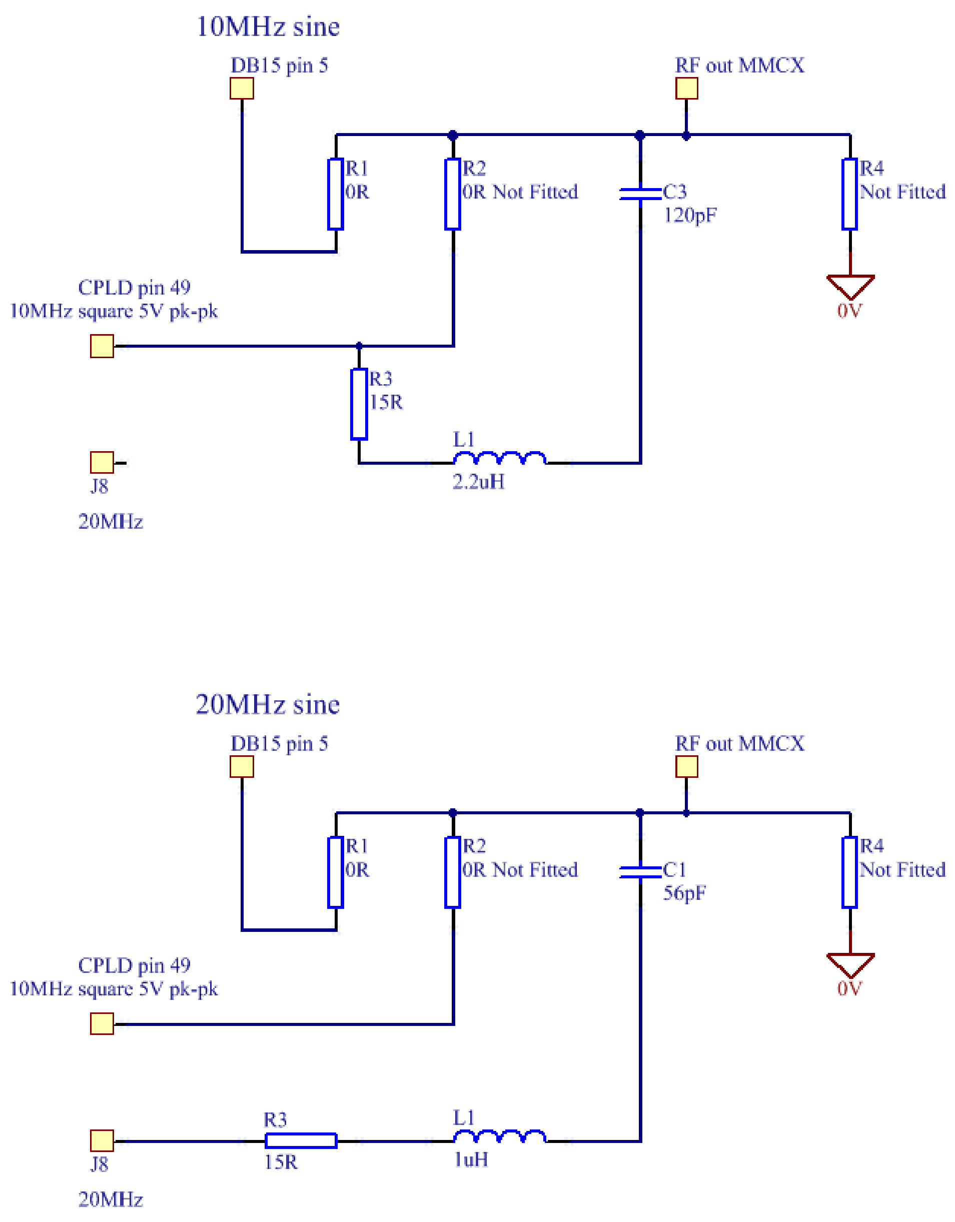


Figure 5. Ruby Schematic Before and After

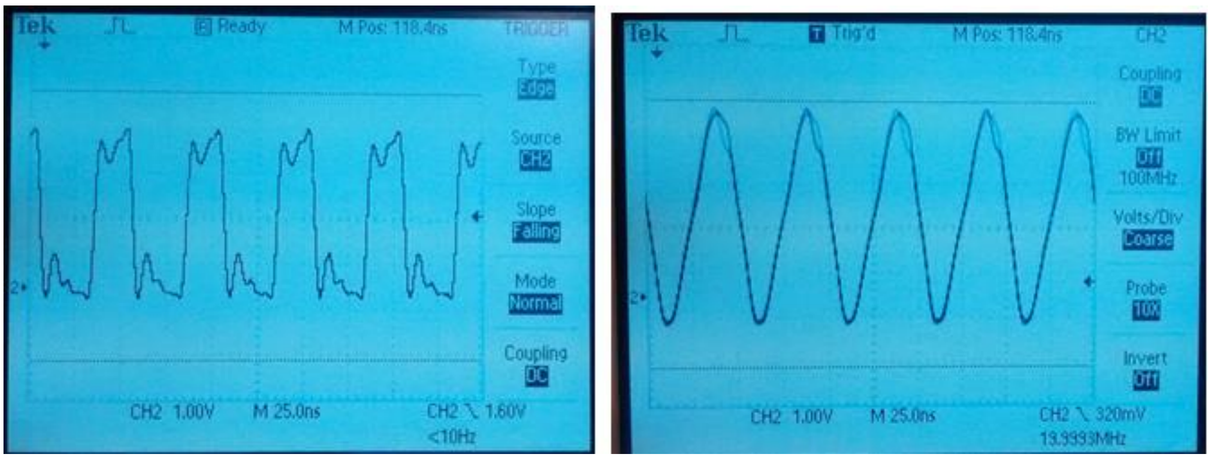


Figure 6. 20MHz Waveforms - Before and After Filter

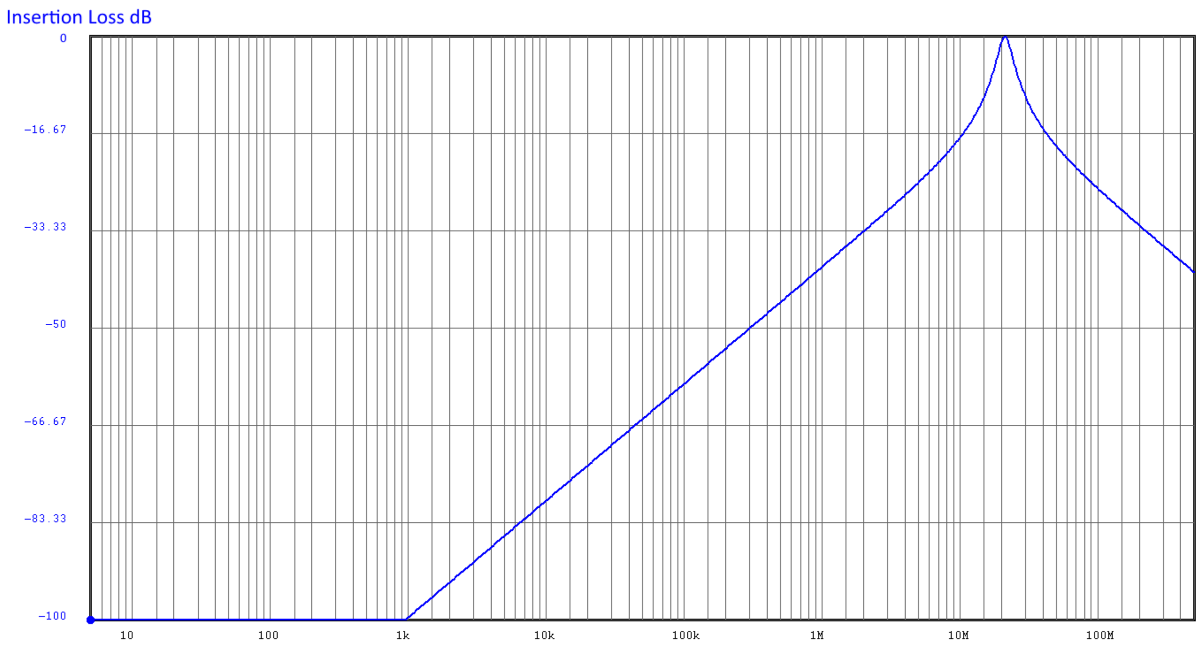


Figure 7. 20MHz band pass filter (into 1kΩ)

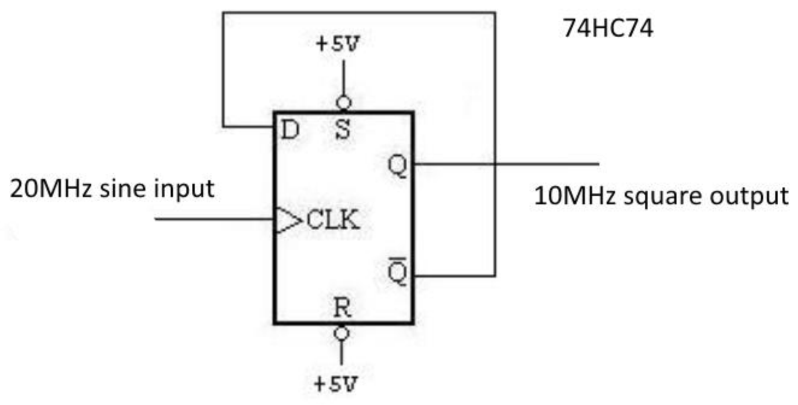
3.2 External Circuitry

The following mods take the 20MHz signal, convert it to 10MHz and condition it suitably for the test equipment I'm going to connect. They also drive indicator LEDs.

3.2.1 *Divide by 2*

Divide the 20MHz by 2 using a 74HC74 D-type flip flop. This is replicated 4X for the 4 outputs.

5V is used for the flip flop supply as they switch faster (compared to the 3.3V alternative) and it was easy to route it to the DB15. I hoped it would have less noise as the CPLD with all its heavy switching runs from 3.3V but unfortunately this didn't seem to be the case.



3.2.2 *10MHz Conditioning*

Condition the 10MHz output from the D-types. A resistor divider reduces the signal to approx. 300mV peak and the 100n cap AC couples the signal. A limiting diode is also fitted for belt n braces to protect the downstream equipment if the resistor divider fails. The diode is non-conducting during normal operation and doesn't affect the signal.

The input impedance on my test equipment ranges from 50Ω on an HP 8920B RF test to 1kΩ on a Keysight 53131A Frequency Counter. This makes it hard to achieve a consistent signal level so I made the lower leg of the divider 47Ω low so it wouldn't get loaded down too much by the external equipment.

Originally I used 330Ω instead of 1k Ω but the ringing (I think caused by impedance mismatches) was too great. Note this signal is NOT impedance matched.

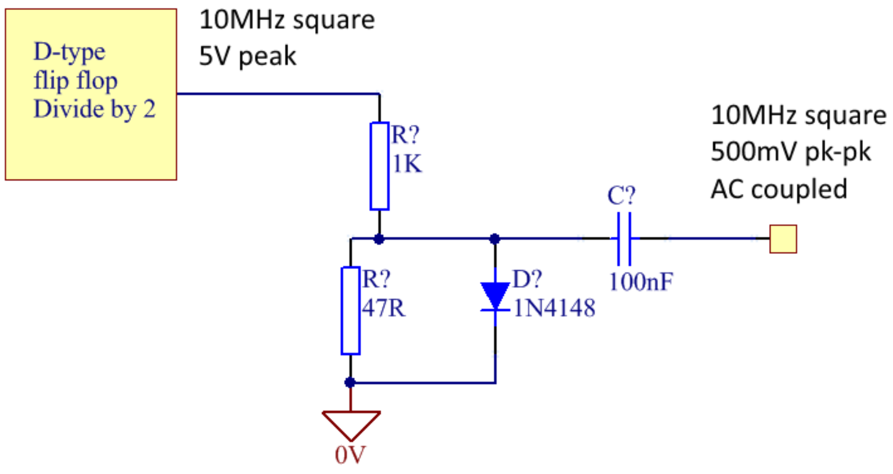


Figure 8. **10MHz conditioning**

The final signal is about 300mV peak to peak. The "noise" is 80MHz which I still believe is ringing from impedance mismatch but unfortunately I couldn't get rid of it. The test equipment doesn't seem to object to so I'm not too fussed.

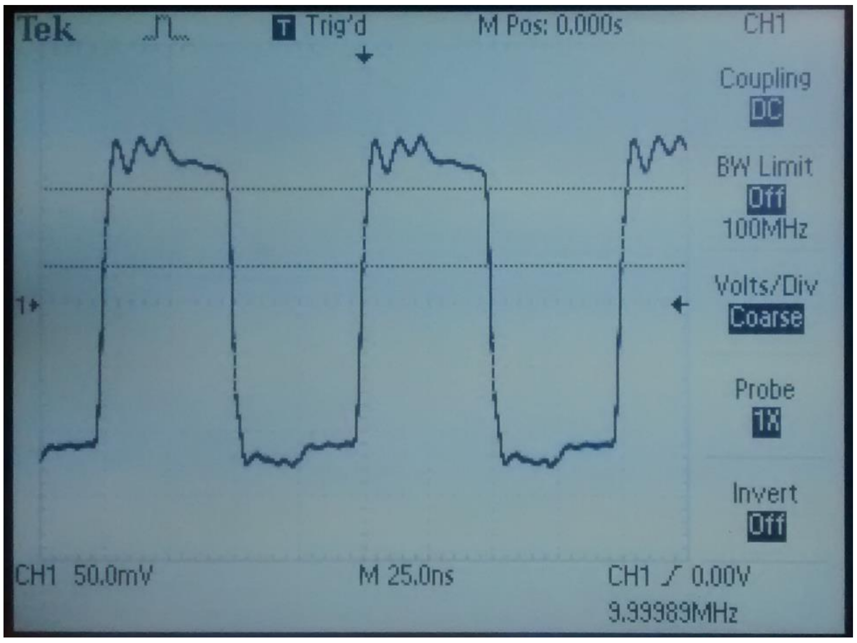


Figure 9. **Final 10MHz Output**

3.2.3 *Indicator LEDs*

There are 3 LEDs for Power, 1pps and Frequency Lock. I read somewhere that the Lock needs buffering so I used a 74HC125, it had some spare inputs so I buffered all signals as it just seemed like a good idea. C4, R8 stop the Lock LED from giving a brief blip at power on. The 1pps LED also blips but I couldn't stop this as it's longer than the 20µs 1pps wanted signal from Ruby. The lock signal is open collector so it needs R7 pull-up.

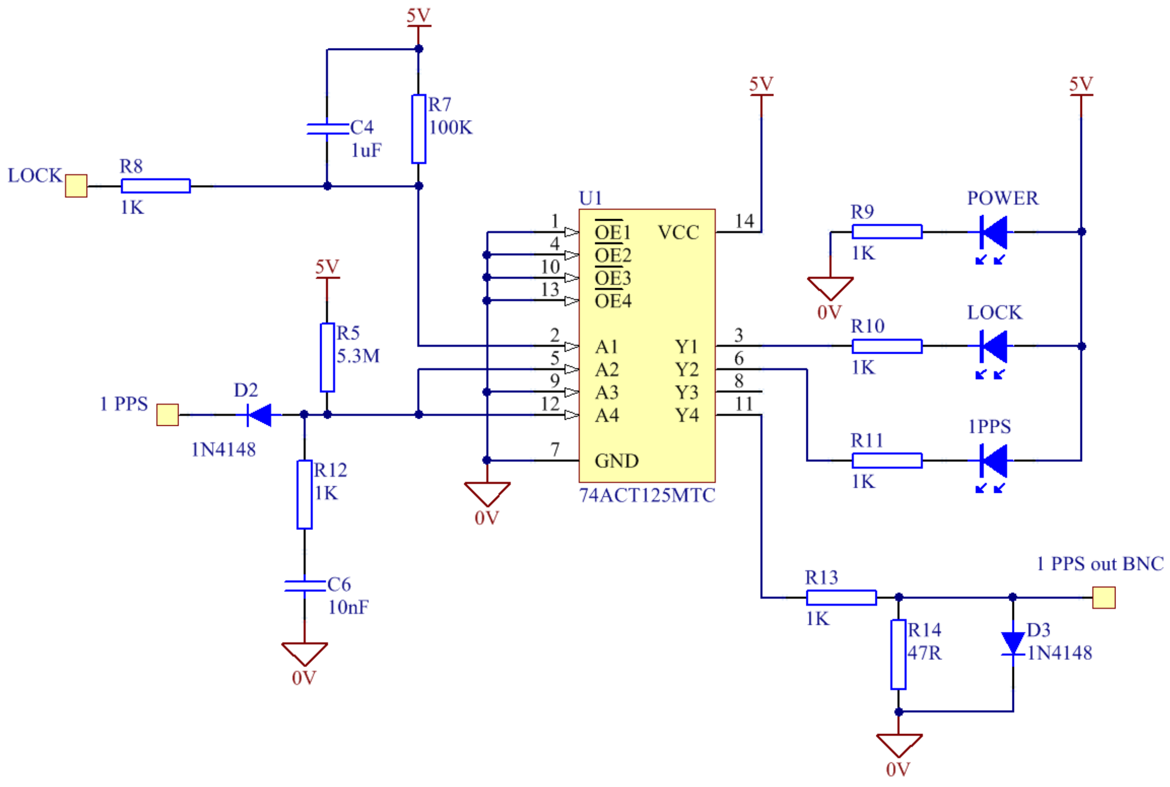


Figure 10. LED Buffering

3.2.4 *1 Pulse-Per-Second Buffer*

The 1pps output from Ruby is normally high then goes low once every second for 20µs. This is far too short to be visible on the LED so before buffering I needed to stretch it. D2, C5, R12, C6 stretch it to 10ms and it's actually quite visible for such a short pulse.

With the 1pps high, input A4 is pulled high through R5 and C6 is fully charged. As the 1pps goes low, C6 discharges quickly through R12 and the diode. When the 1pps goes high, the diode is reverse biased so C6 charges slowly through R5 and R12. This slow charge time keeps A4 low for about 10ms hence the pulse stretching.

R13, R14, D3 limit the signal to 500mV same as the 10MHz signals, notice though that it isn't AC coupled. I'm not sure if, or how, this signal would be used so the attenuation is only to avoid mishap if I connect something without thinking - I'll change it later if ever needed

3.2.5 *Power Supply Filtering*

Power for the whole device comes from a 16V laptop plugpack. It has a large 10MHz signal present which I wanted to keep away from Ruby so filtering was needed. It could have been done a million ways but I opted for a LC filter using the parts I had and it gave pretty good improvement. And no, the noise definitely wasn't from the rest of the circuitry. Unbelievably also, the PSU isn't isolated and has mains neutral connected through to DC 0V. Shame on you Vaio.

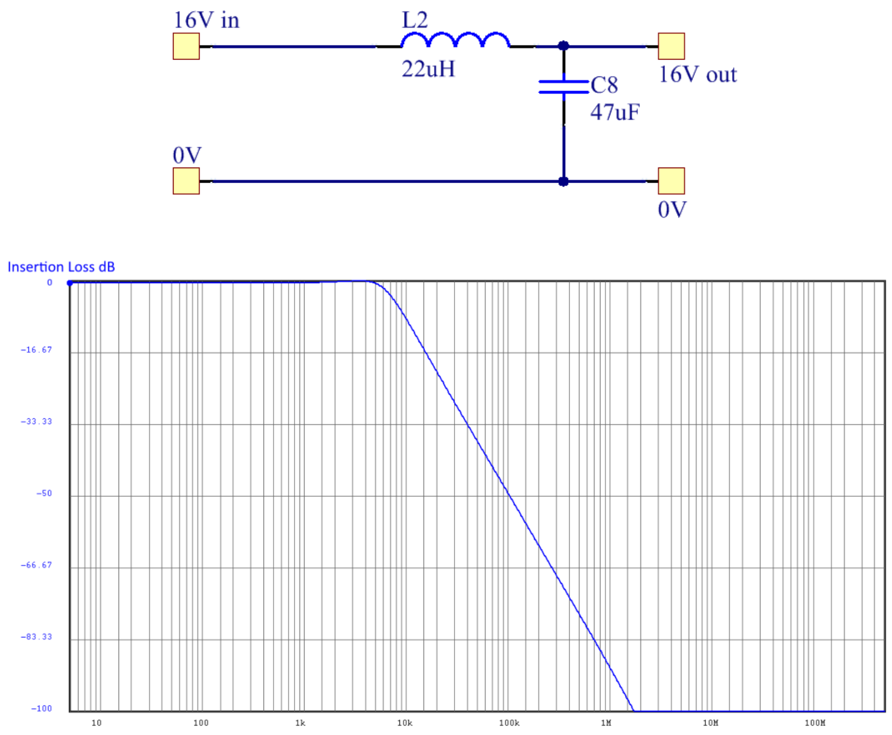
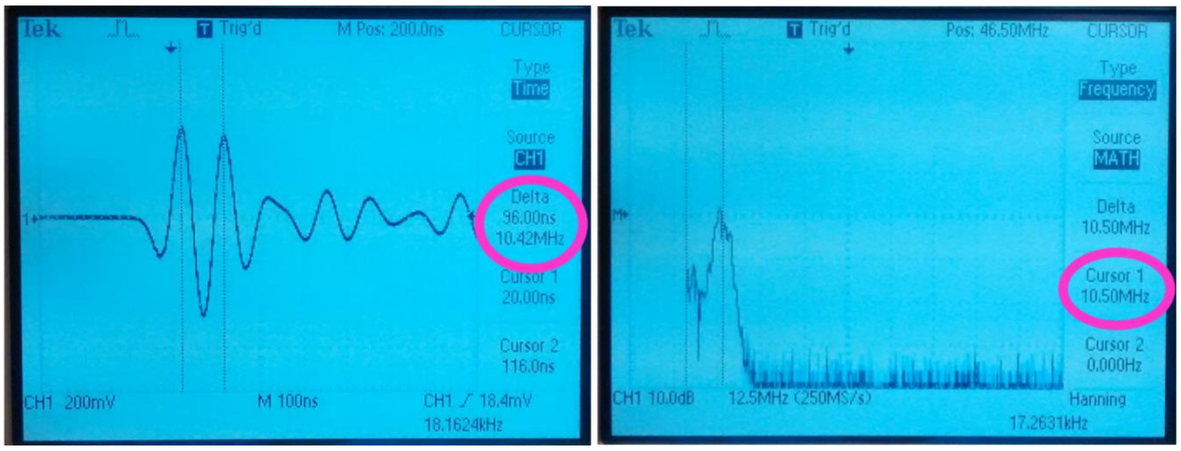
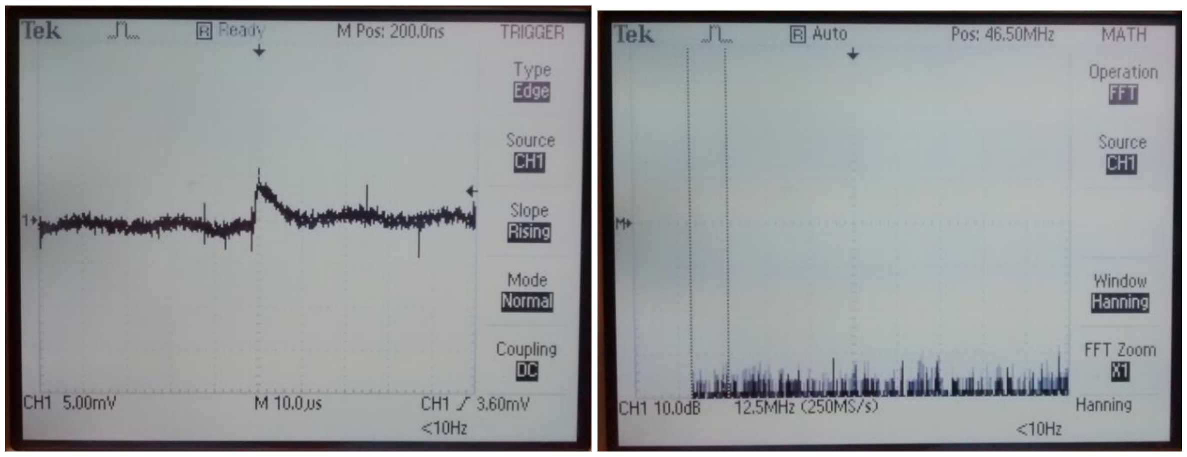


Figure 11. **Filter and Simulated Response**


Tek Trig'd M Pos: 200.0ns CURSOR
Type Time
Source CH1
Delta 96.00ns 10.42MHz
Cursor 1 20.00ns
Cursor 2 116.0ns
CH1 200mV M 100ns CH1 ∕ 18.4mV 18.1624kHz

Tek Trig'd Pos: 46.50MHz CURSOR
Type Frequency
Source MATH
Delta 10.50MHz
Cursor 1 10.50MHz
Cursor 2 0.000Hz
CH1 10.0dB 12.5MHz (250MS/s) Hanning 17.2631kHz

Figure 12. **16V Noise – Original – Time and Frequency Domains**

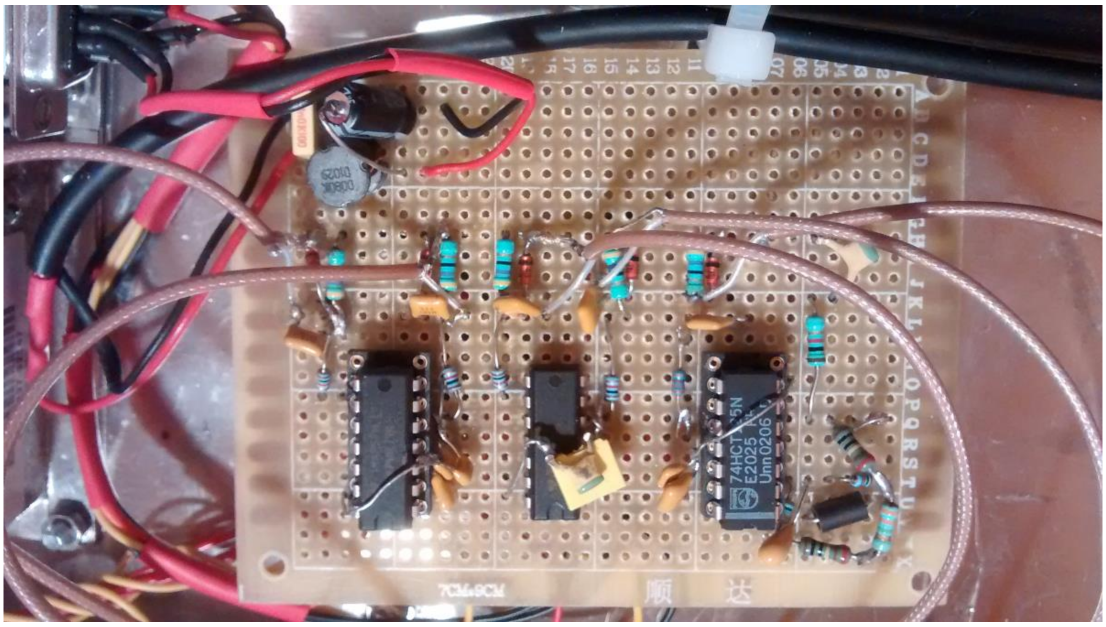

Tek Ready M Pos: 200.0ns TRIGGER
Type Edge
Source CH1
Slope Rising
Mode Normal
Coupling DC
CH1 5.00mV M 10.0µs CH1 ∕ 3.60mV <10Hz

Tek Auto Pos: 46.50MHz MATH
Operation FFT
Source CH1
Window Hanning
FFT Zoom X1
CH1 10.0dB 12.5MHz (250MS/s) Hanning <10Hz

Figure 13. **16V Noise with filter (note voltage scale is now 5mV)**

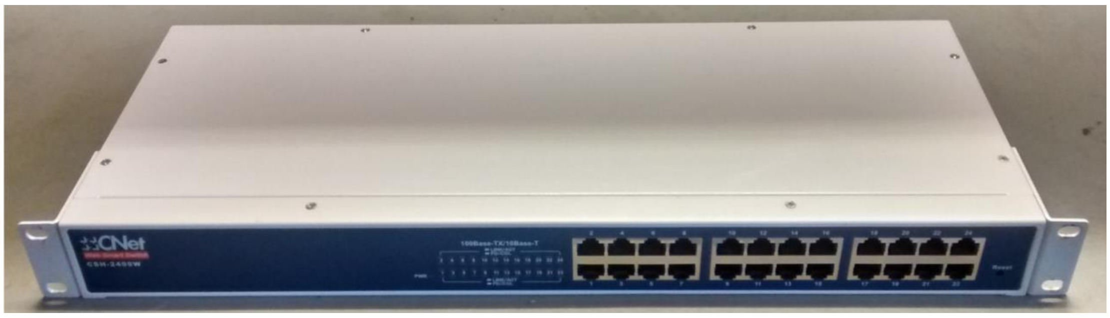
3.2.6 *Stripboard Layout*

Decoupling is a bit inconsistent but that was just what I had; a selection of 10n and 100n.



3.3 The Case

It was a video by Gerry Sweeney that first inspired me to think about packaging up the whole standard into a more user friendly device. It's all very well hooking up a bench PSU but there are better ways. Gerry had used a video splitter as it essentially offered much of what was needed: case, PSU, signal buffering, BNC connectors etc.. I couldn't find anything similar so I started with an ATX supply but unfortunately destroyed the PCB while drilling and that was the end of that. So I started looking elsewhere when I found this old router for $7 on Trademe. Bargain! On the face of it, it offered case and power socket and maybe a switch and power supply too. The power supply unfortunately was 3V3 and as the connector was mounted on the PCB I couldn't use that either; it also had a 3V3 fan currently unused.



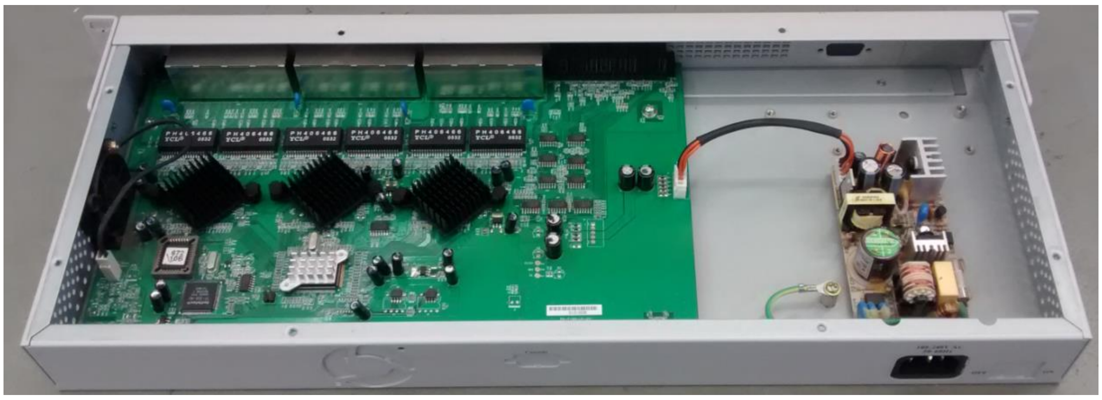


Figure 14. Router Case

The obvious problem was the front panel so I picked up a 19” blanking plate and glued it on. First though I had to mount LEDs, power switch and BNC connectors.

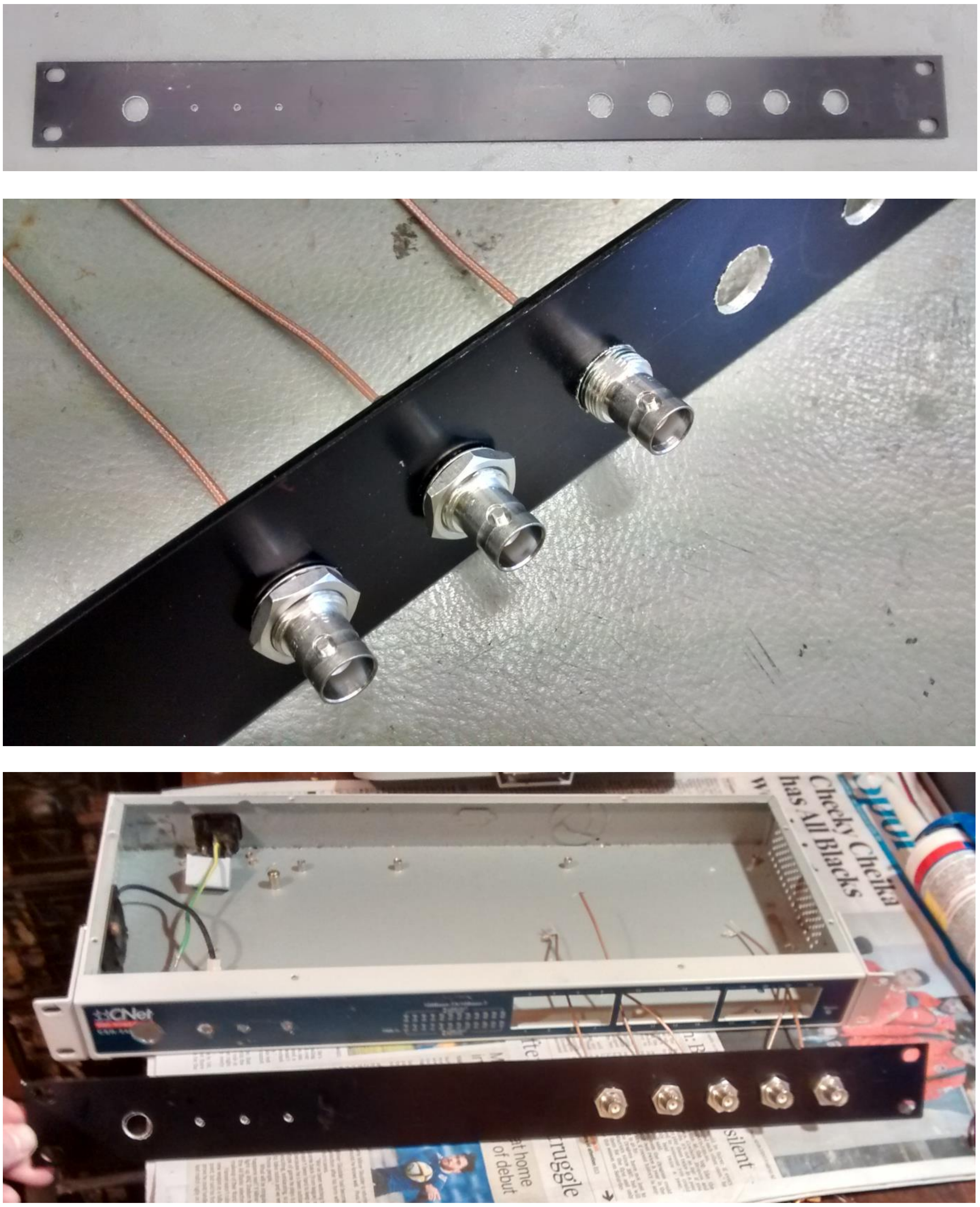
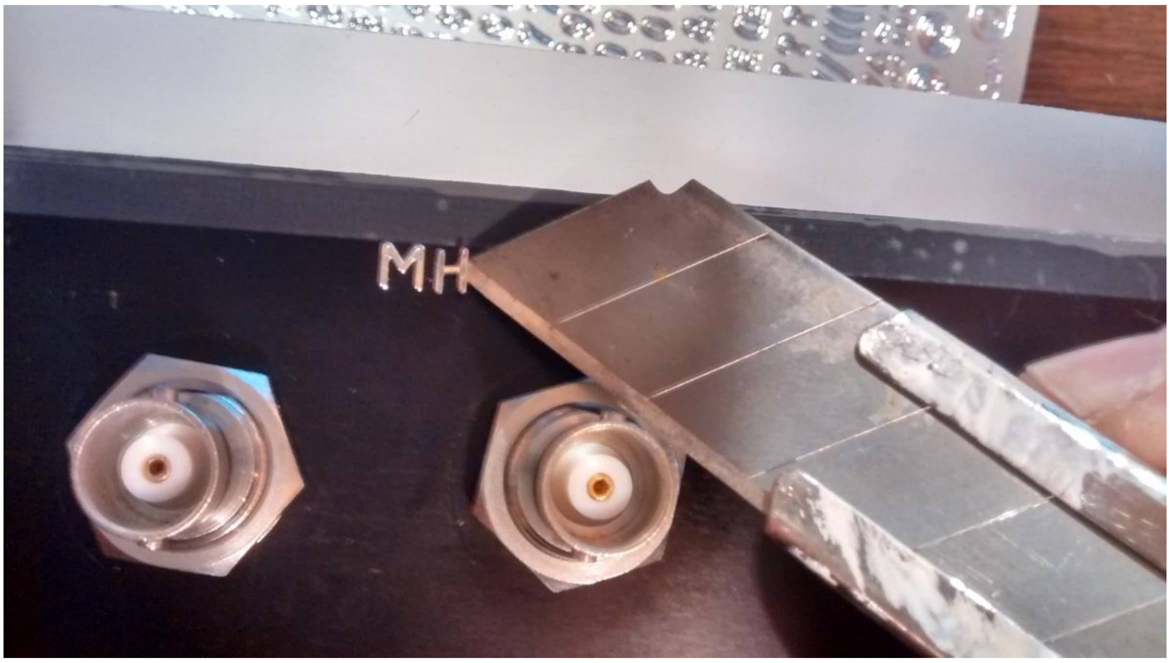


Figure 15. Front Panel Mount

For the legend I used stick on silver letters from an art shop – a whole sheet for $4. I had intended to spray the whole panel with lacquer afterwards to stop the letters scratching off but they've stuck really well so currently I'm going to leave as is. Sellotape helps with alignment and I used a stanley knife to handle them.



Someone kindly gave me a laptop supply with nominal 16V output to directly power Ruby. I had three main components to fit inside the case so I cut a metal plate which sits over the standoffs used by the original router PCB.

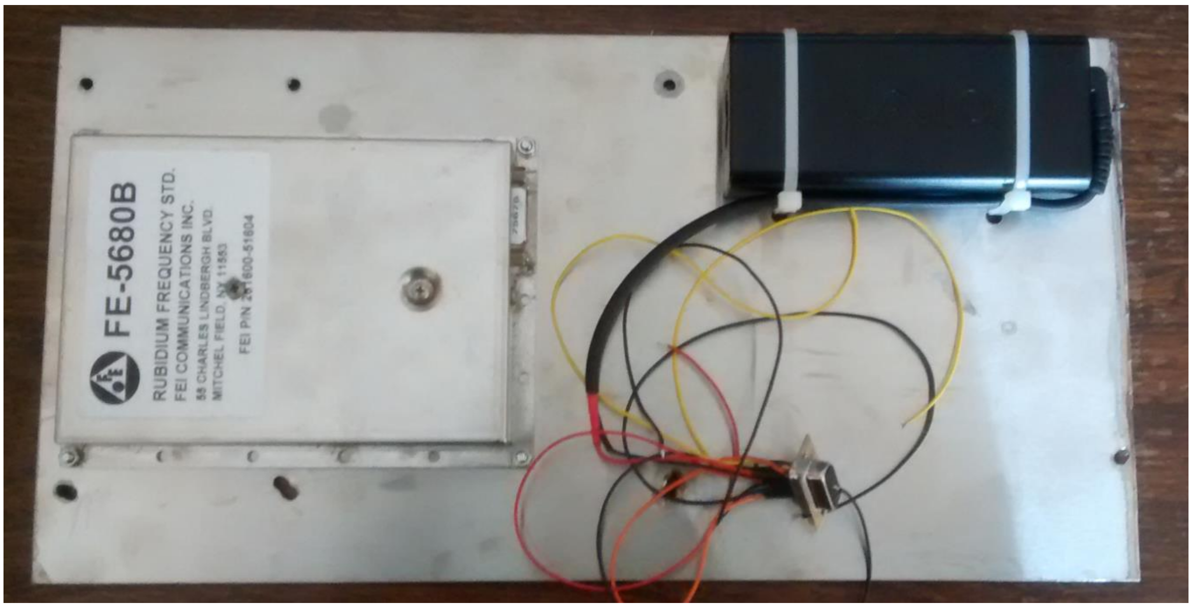


Figure 16. Mounting Plate

I mentioned the mains connector needed replacing. The replacement was smaller and had to be glued in. The pic also shows the X rated cap and wiring to the power switch. The heavy black lead is the supply for the laptop plugpack.

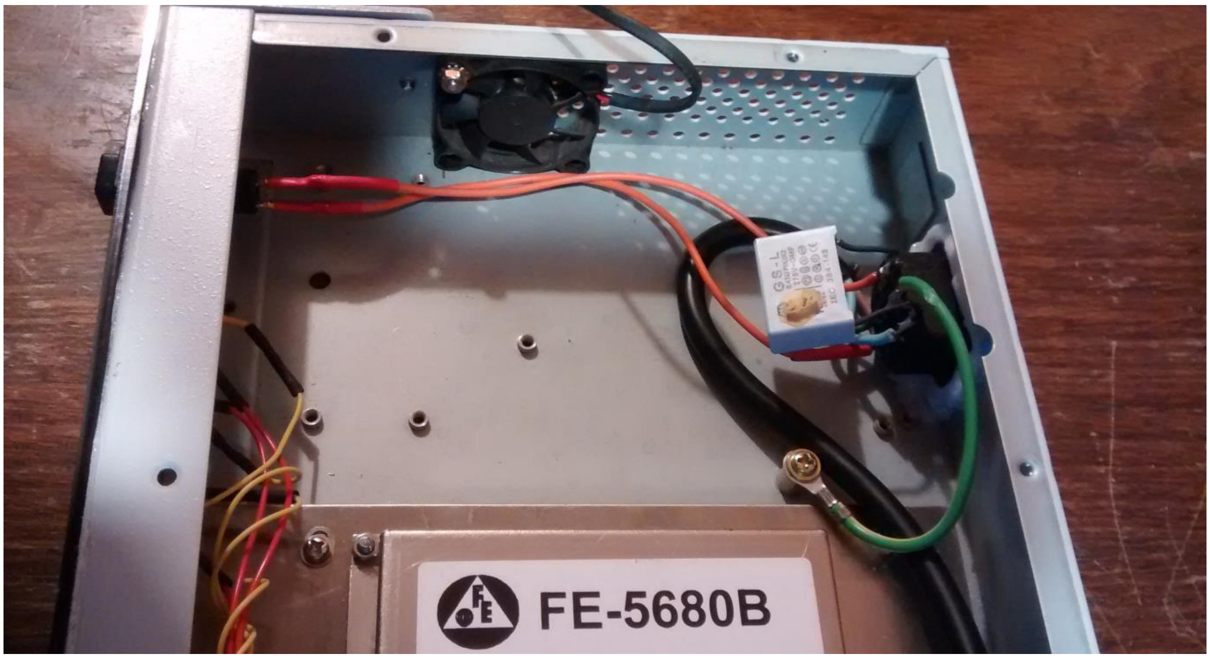


Figure 17. Mains Socket

The finished product. Twisted yellow and red wires carry LED signals. The circuit was built on stripboard; a PCB would have been better for noise and tidier overall but hey… it works.

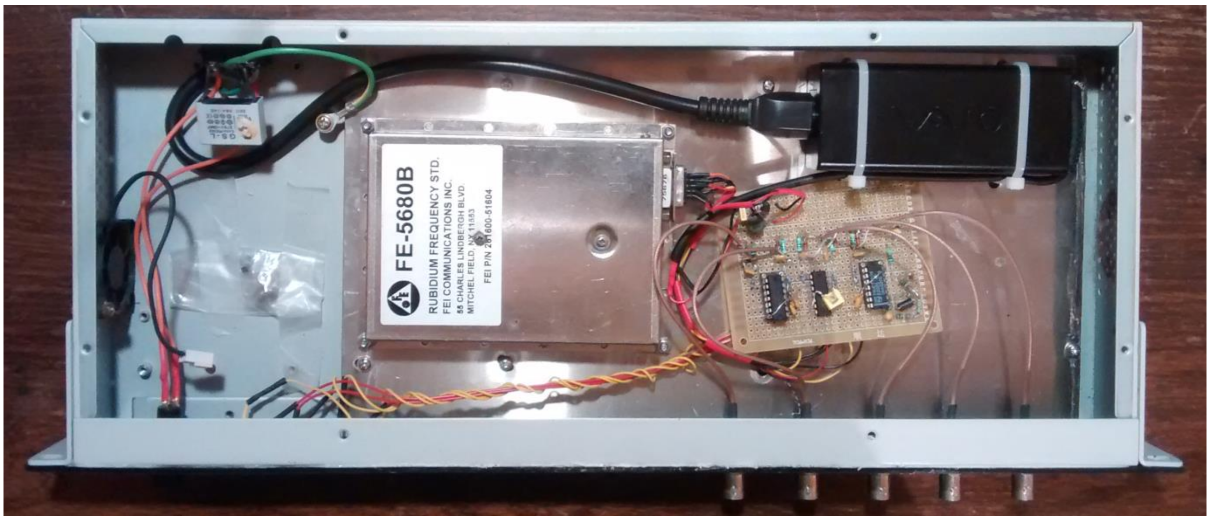


Figure 18. All the Bits

Temperature testing. Ruby gets to 60°C just sitting on the bench so I was concerned that this would rise further inside a case. I sealed it up and monitored various points but thankfully it only got to 65°C.

4. Further Comment

Ruby outputs 10Mhz but it disappears about 10 secs after power on. What causes it to disappear?

The signal comes from CPLD U403 pin 49. My gut feel is that the 10MHz is gated with a control signal inside the CPLD. Either a message or possibly just a logic signal from U400 µC acts as the control signal for this gate? I began looking for signals that change state when the 10MHz disappears but quickly gave up as just about all of them do. Perhaps someone reading this may care to carry on.

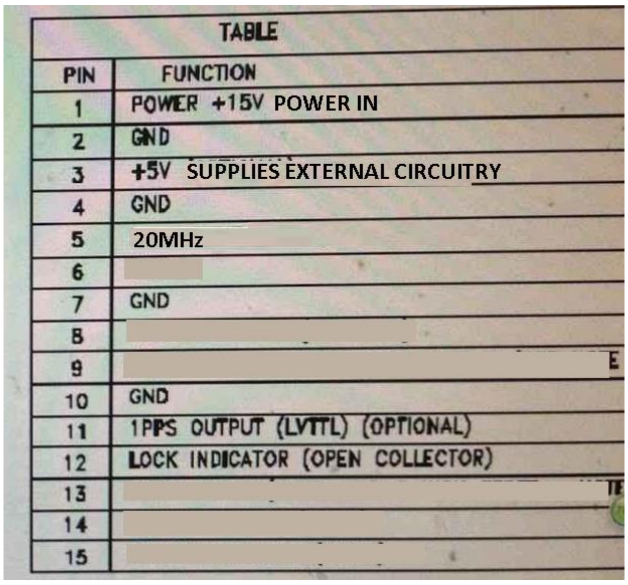
There are many Ruby standards so they may share common code and a link is fitted to indicate the 10MHz should disappear. This could be read by either the µC or the CPLD. I removed the 1kΩ between U400.3 and VCC without effect.

The 1pps used to be 2µs wide but is now 20µs. I have no idea what caused the change.

The bare Ruby used to take 3 minutes to lock when sitting on the bench and powered from a bench PSU. Now it's all boxed up with a laptop supply it takes 15 minutes. Why ??

An RS232 line driver is fitted and data correctly gets from the DB15 to the processor. I found documentation that recommends sending ASCII "S" and the message "Request Frequency Offset" (HEX 2D 04 00 29) but neither yielded a response and I didn't persevere further.

DB15 New Pinout



TABLE

PIN	FUNCTION
1	POWER +15V POWER IN
2	GND
3	+5V SUPPLIES EXTERNAL CIRCUITRY
4	GND
5	20MHz
6	[redacted]
7	GND
8	[redacted]
9	[redacted]
10	GND
11	1PPS OUTPUT (LVTTL) (OPTIONAL)
12	LOCK INDICATOR (OPEN COLLECTOR)
13	[redacted]
14	[redacted]
15	[redacted]

Figure 19. DB15 Modified Pinout