N2PK VNA / myVNA timing data

This document gives supplementary timing data for the USB and parallel interfaces

The data was gathered with the following configuration:

- PC: Athlon 64X2 3800+ CPU, 2.5 GByte RAM
- OS: Windows XP Home Edition 2002, SP3
- Parallel: built in ECP port (LPT1)
- USB: V0.4 PCB, drivers v0.22
- Parallel Driver WinIO for part, inpout32 for the rest.
- myVNA V0.44c

The traces have the following data lines:

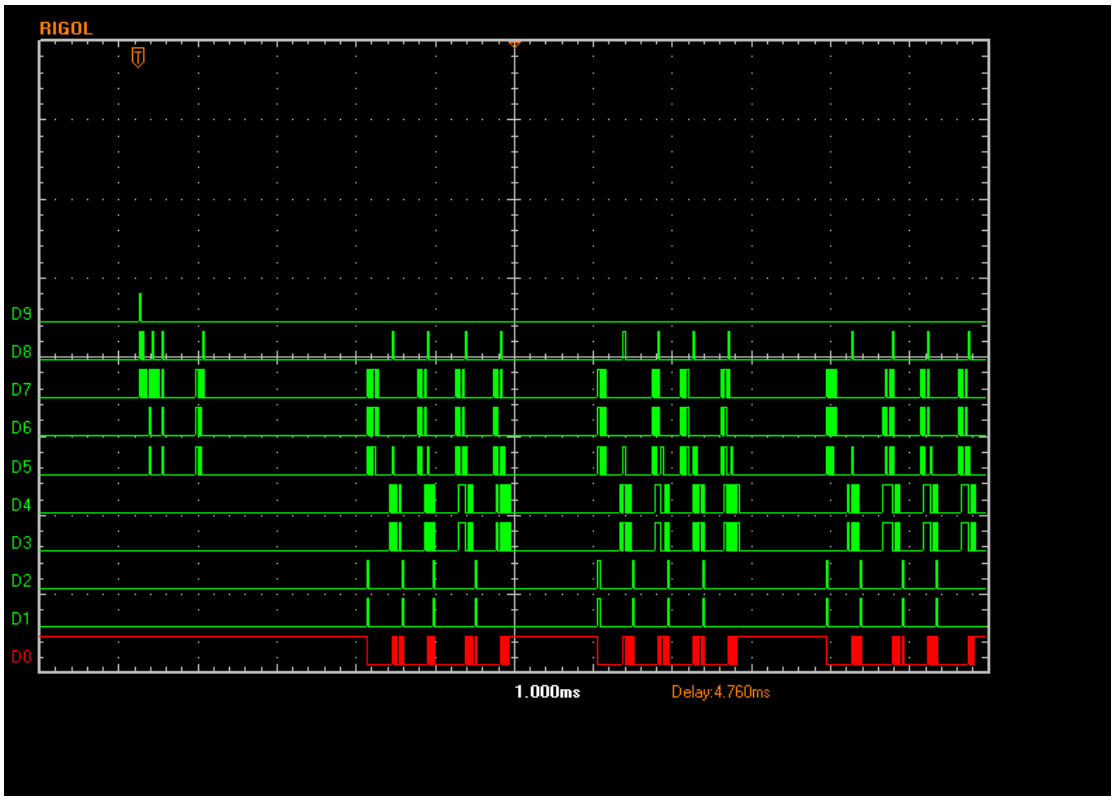
Analyser Data Line	Function
D9	DDS Reset
D8	DDS FQ_UD
D7	DDS WCLK
D6	DDS LO DATA
D5	DDS RF DATA + ADC SDI
D4	ADC SDO2
D3	ADC SDO1
D2	ADC CS2
D1	ADC CS1
D0	ADC SCLK

To make it easier to understand the traces, the program settings were as follows:

- Sweep Start delay 2000 us (not in all cases – sorry)
- ADC Step delay 1000 us
- Phase change step delay 0
- Scan 200 point, dual detector displaying S11 and S21
- Overlap DDS and ADC operation

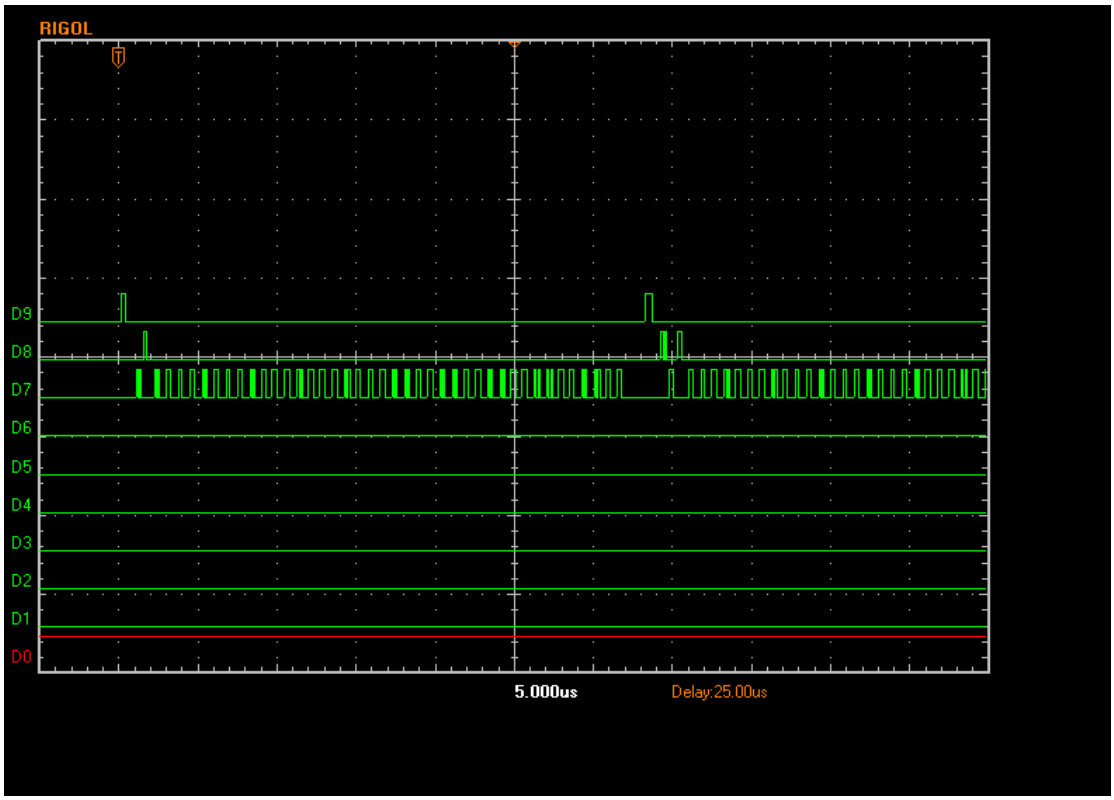
USB Port

Here is the start of a sweep showing DDS reset, sweep start delay and the first 3 points in the scan, where each point comprises 4 conversions for the 4 CDS phase points 0, 90, 180 and 270 degrees.

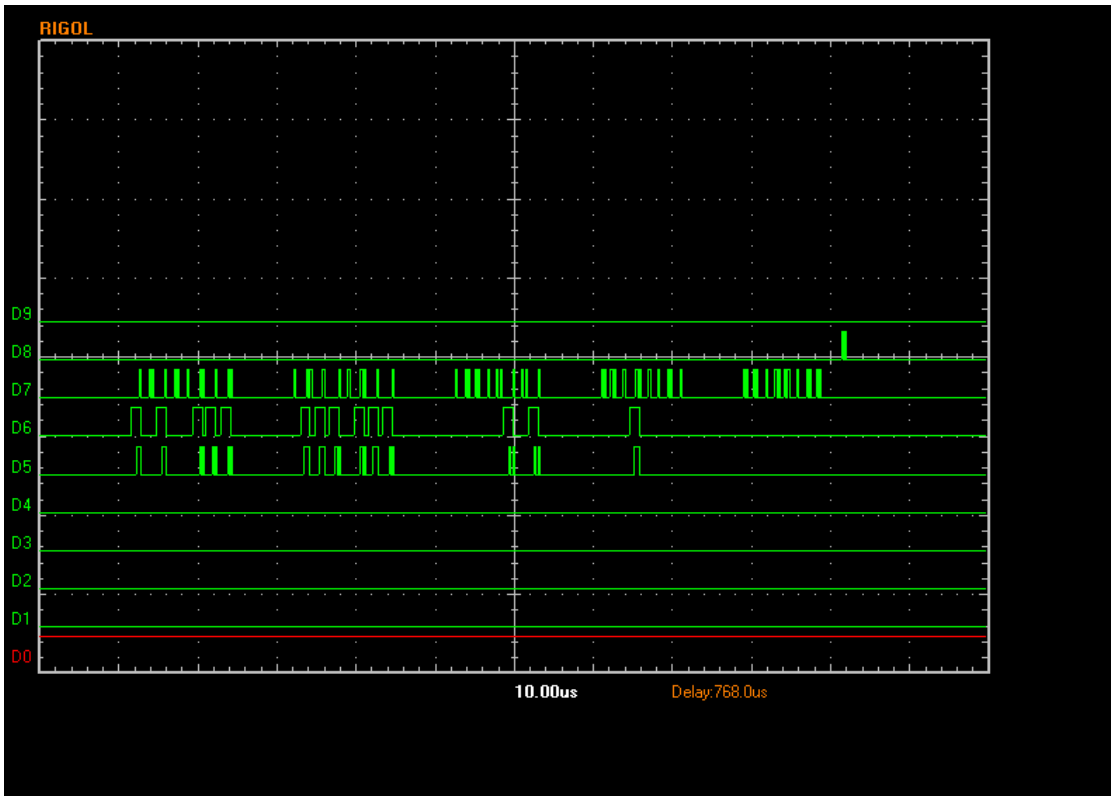


The first event is a DDS reset followed by setting of the frequency for the first point, 0 degrees phase. There is then the 2000 us programmed sweep start delay followed by 4 ADC conversions in quick succession, no inter phase step delay, then a 1000 us frequency step delay, the next 4 phase points etc.

With the USB port, the first event at the start of a sweep is the reset of the DDS. The top trace is the DDS Reset, the D8 line is FQUD and D7 is DDS clock. This is a zzoom in on the first events visible in the previous traces.

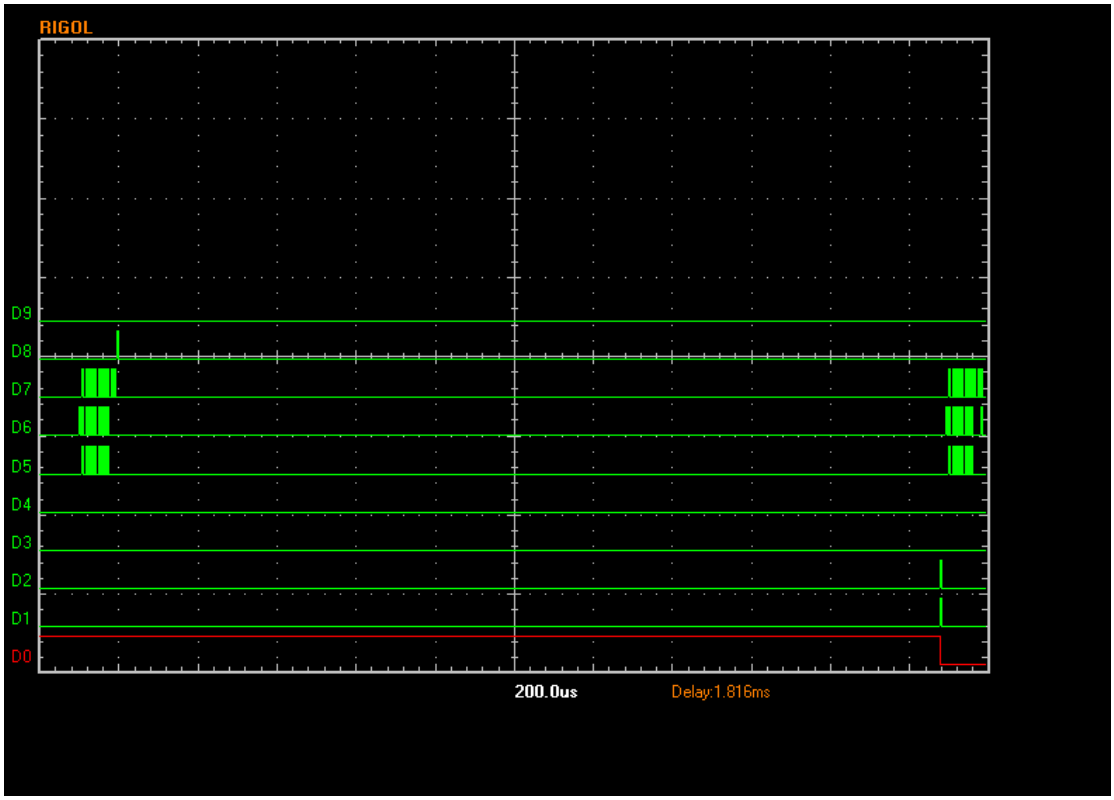


After this the DDS is set to the first frequency. D7 is the clock, and 5 bytes can be seen being clocked it. D5 and D6 are the data lines. After the last byte, FQUD is pulsed (D8) to load the data and set the frequency.

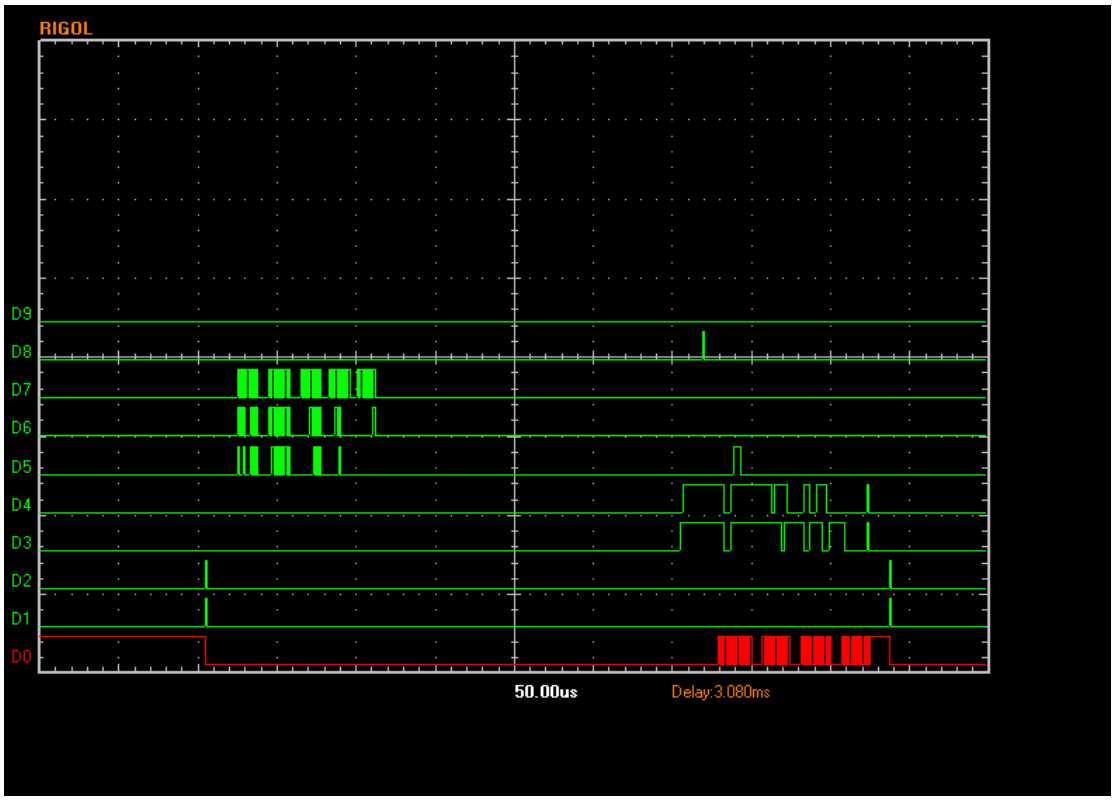


DDS load takes 90 us and finishes with an immediate load (FQ_UD pulses high)

Zooming out, the initial DDS load is visible on the left and is followed by the sweep start delay of just over 2000 us. The ADC clock had been kept high to keep the ADC on hold; an ADC conversion is started by dropping the clock signal. CS is also pulsed high and low. Immediately after this (ADC / DDS overlap is enabled) the DDS is set for the next phase point.

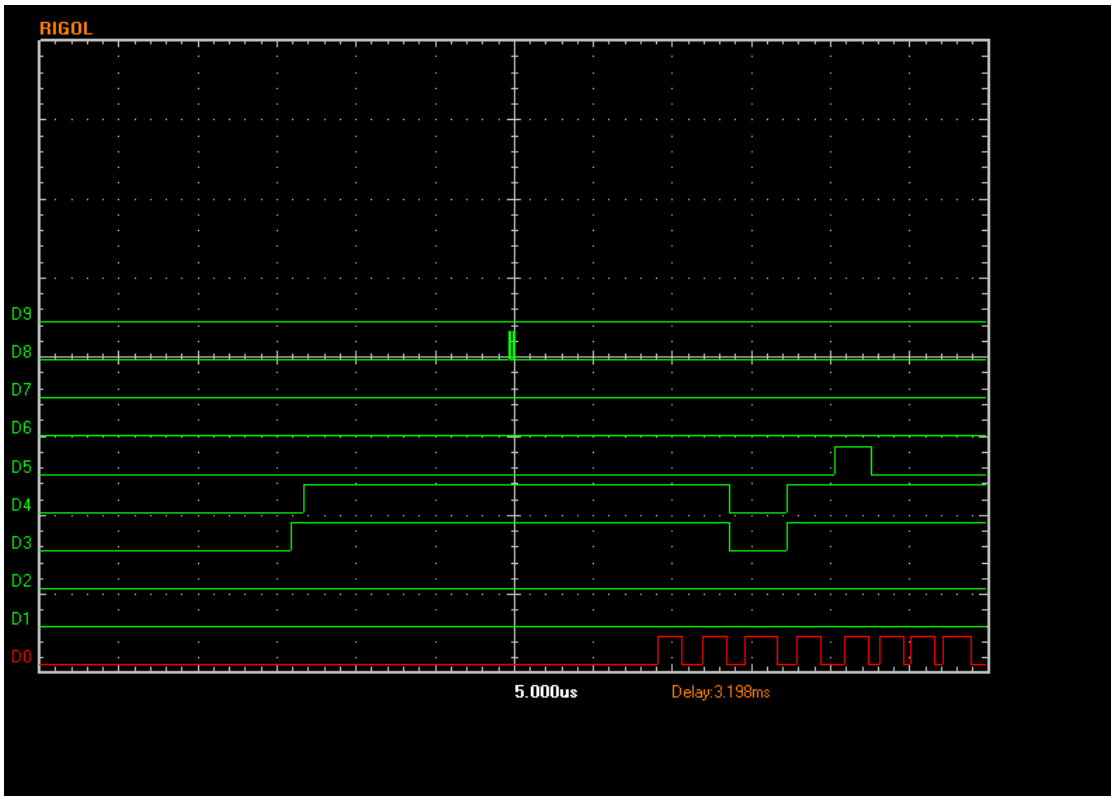


Below is the first ADC conversion. The ADC conversion starts with a CS pulse and clock going low. Whilst the ADC is converting, the DDS is set for the next frequency but the FQUD is not pulsed at this stage.



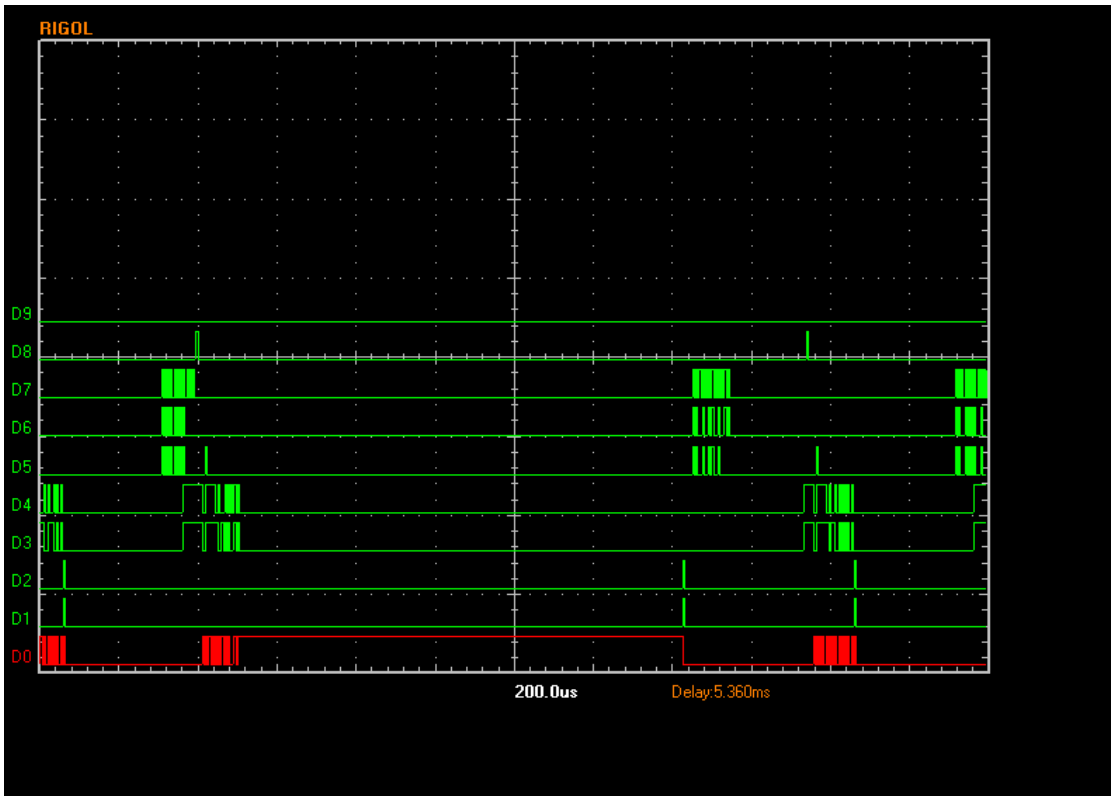
The USB converter then waits for the ADC conversion to complete (signalled by the data lines going high). This happens about 300 us after the ADC conversion starts. Now that the ADC is complete, but before the result is read, the FQUD (D8) is pulsed to set the DDS and allow time for the signals to settle. After this the ADC results are clocked out; this takes just over 100 us. After a short delay of just under 20 us, the next ADC conversion starts as the phase step delay was set as 0.

Zooming in on the start of the reading of the ADC clocking, the two ADC ready signals go high (don't forget the transistor inverter) and 13 us later the FQUD signal is pulsed.

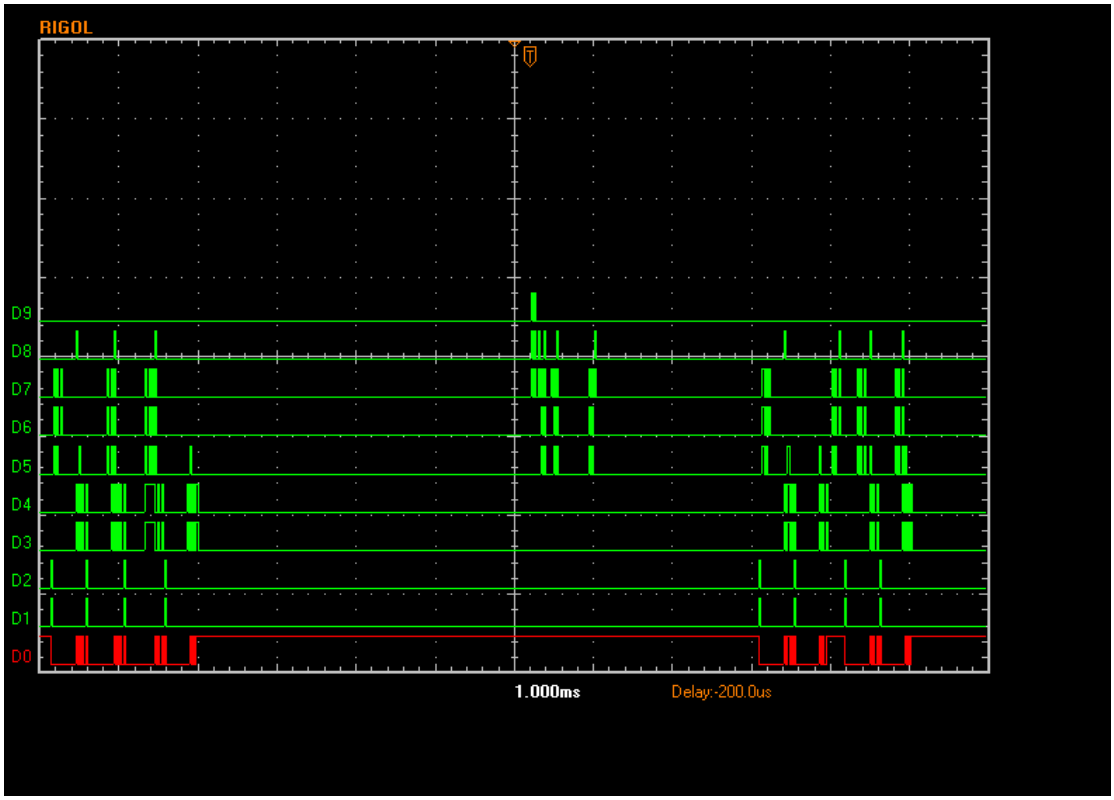


9 usec later the result starts to get clocked out of the ADC. Again, note the ADC speed getting clocked in on the first 6 clock edges; this causes the clock cycles to be slightly slower than the following clocks.

After the first 4 phase readings, the next reading steps the frequency. Here the inter step delay takes effect; the program was set for 1000 us and was slightly longer than this. The method used to create this was simply to hold the ADC clock high.

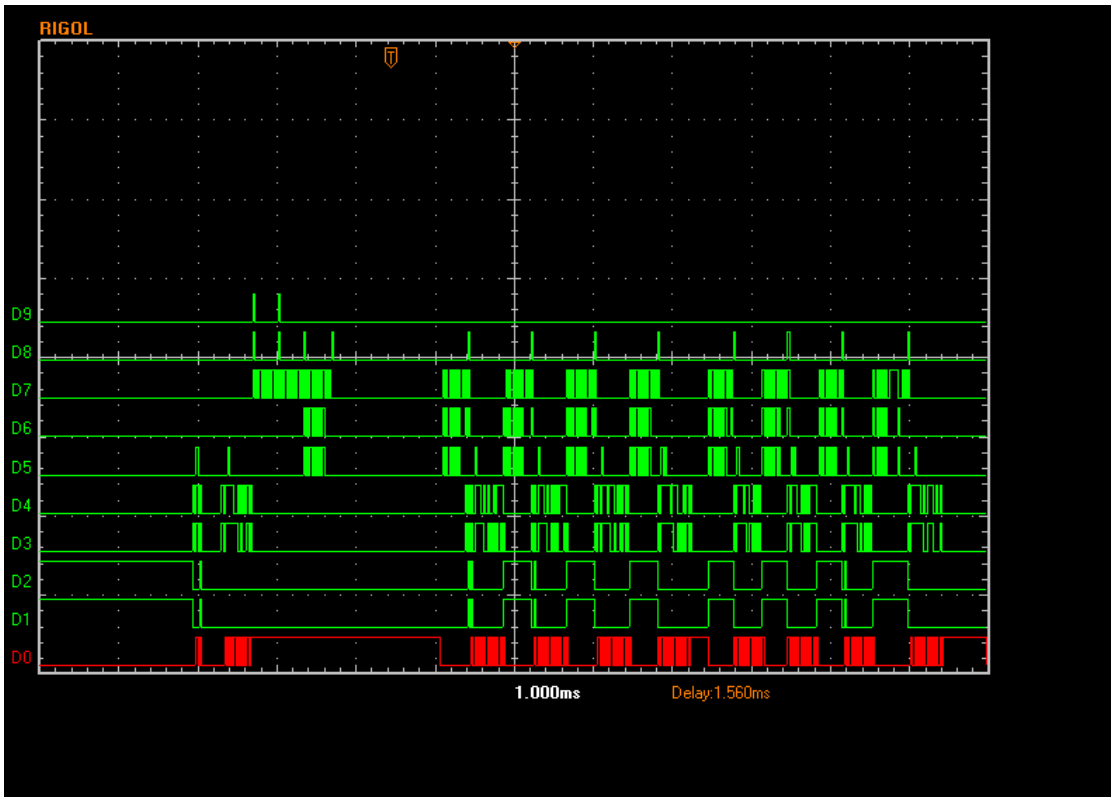


At the end of the sweep, there is a delay caused by the PC before the next sweep starts. This depends on sweep points, PC CPU speed etc etc. Here it is just over 4 ms.



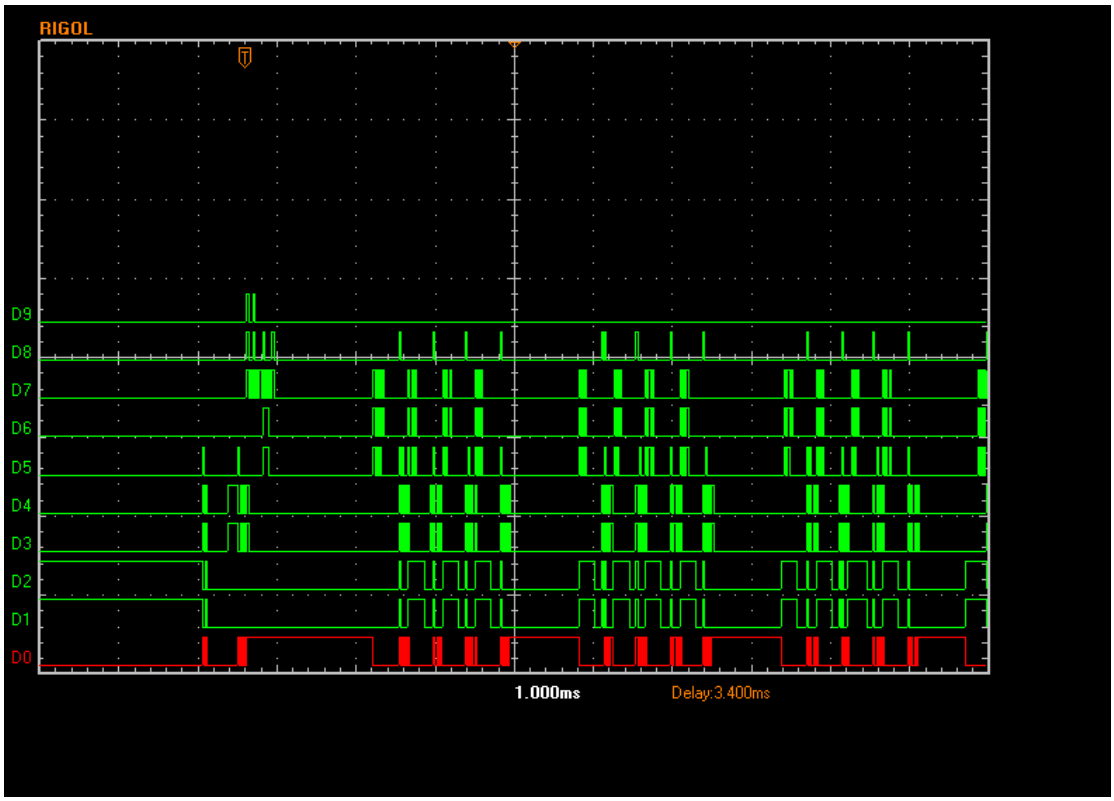
Parallel port

The first trace shows the startup of a sweep, from the first event to the first few frequency points. Two divisions from the left, the activity commences with a reset of the ADCs. There is then a test read of the ADCs, the DDSes are reset and set to the first frequency. The delay is the selected sweep start delay followed by the start of the scan. Visible on the screen are 8 ADC conversions as two groups of 4 (this is CDS). The slightly longer delay between the groups is the selected step delay.



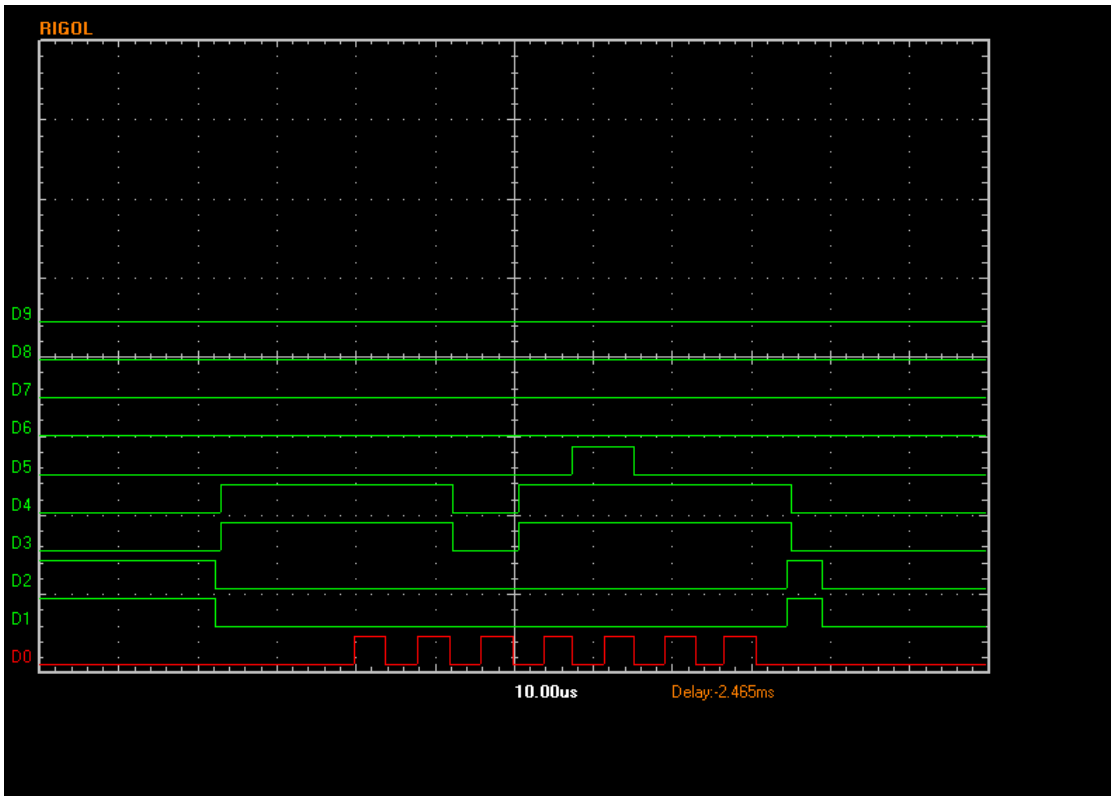
The bottom trace in red is the ADC clock. Above that (D1 and D2) are the two DDS CS signals. Above these (D3 and D4) are the two ADC data outputs with the serial data of an ADC reading being clocked out. D5 is the RF DDS data line which also carries the OSR speed selection data to the ADCs. D6 is the corresponding LO DDS data (which is similar but lacks the ADC programming data). D7 is the DDS clocks, D8 is the DDS FQUD which pulses every time the DDSes are loaded. Finally D9 is the DDS reset.

Here is the same trace with WinIO



Note that there is a significant speed increase. It is also easier here to see the overall pattern. After the initial reset sequence and the sweep start delay can be seen 3 groups of 4 ADC conversions which correspond to 4 points in the sweep where each point takes 4 ADC conversions for the CDS operation (0, 90, 180, 270 degrees) with both ADCs being read in parallel. A 1000 us delay between each set of 4 can be seen as can the initial delay of 1400 us.

Zoom in on start-up. The first thing that happens is that the presence and operation of the ADCs is checked. A partial clocking sets the ADC speed and commences a conversion. This is inpout32 again.



The chip selects go low, and the ADC is clocked 7 times (rising edge). On the first 6 edges the OSR data is clocked in. In this case the data bits are 0 0 0 0 0 1 corresponding to OSR(4:0), or ADC speed 10. It ends by raising and lowering CS lines to start an ADC conversion. Note that the ADC output data is inverted by the transistor buffer on the VNA PCB. This shows an extract from the ltc2440 datasheet for comparison

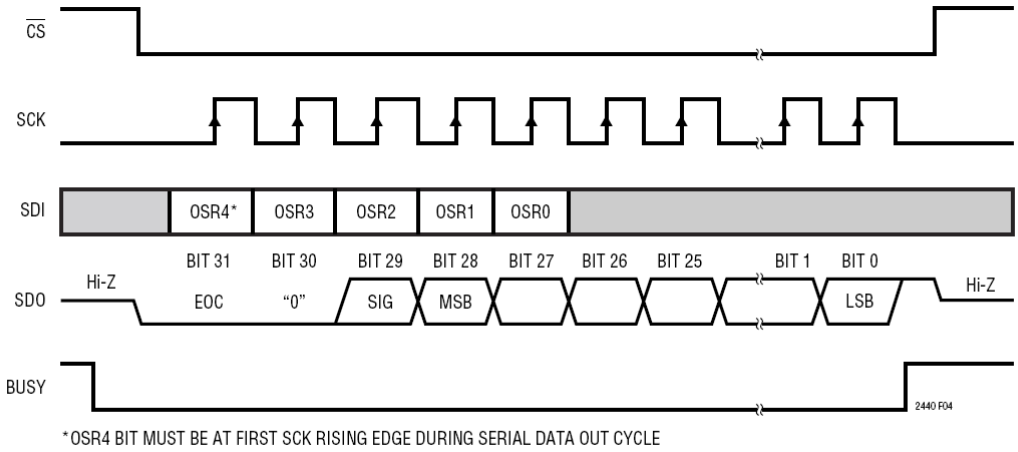
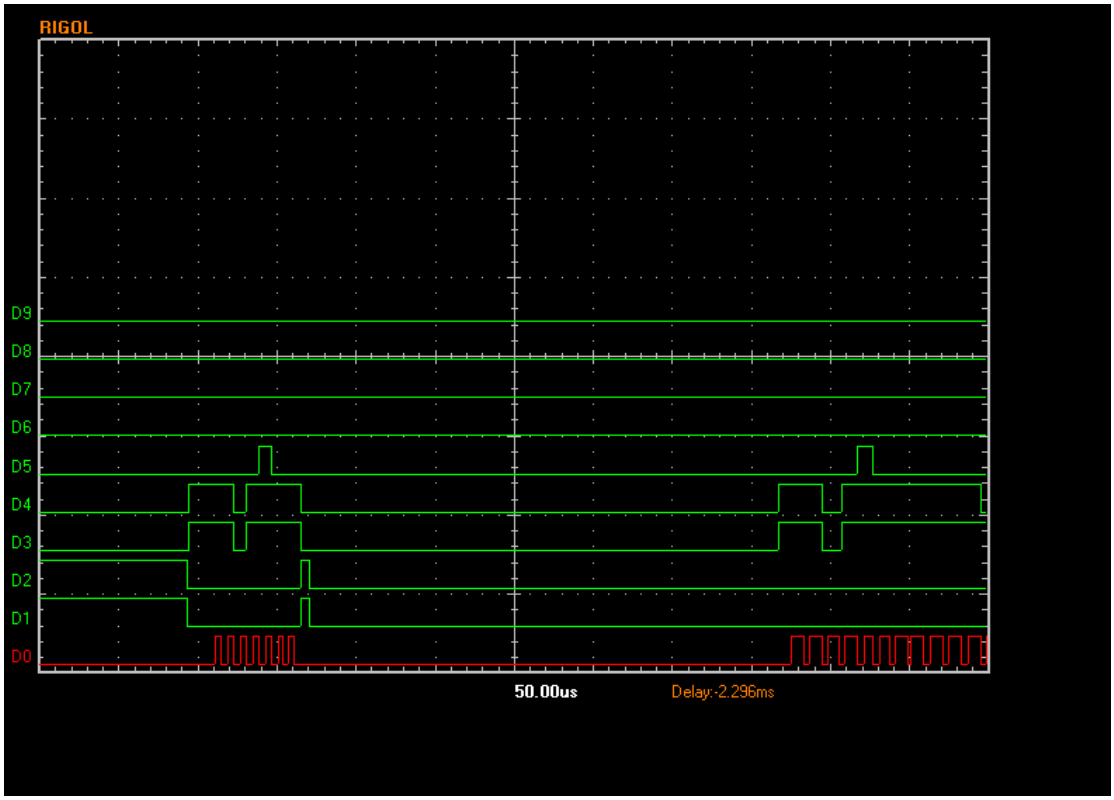


Figure 4. SDI Speed/Resolution Programming

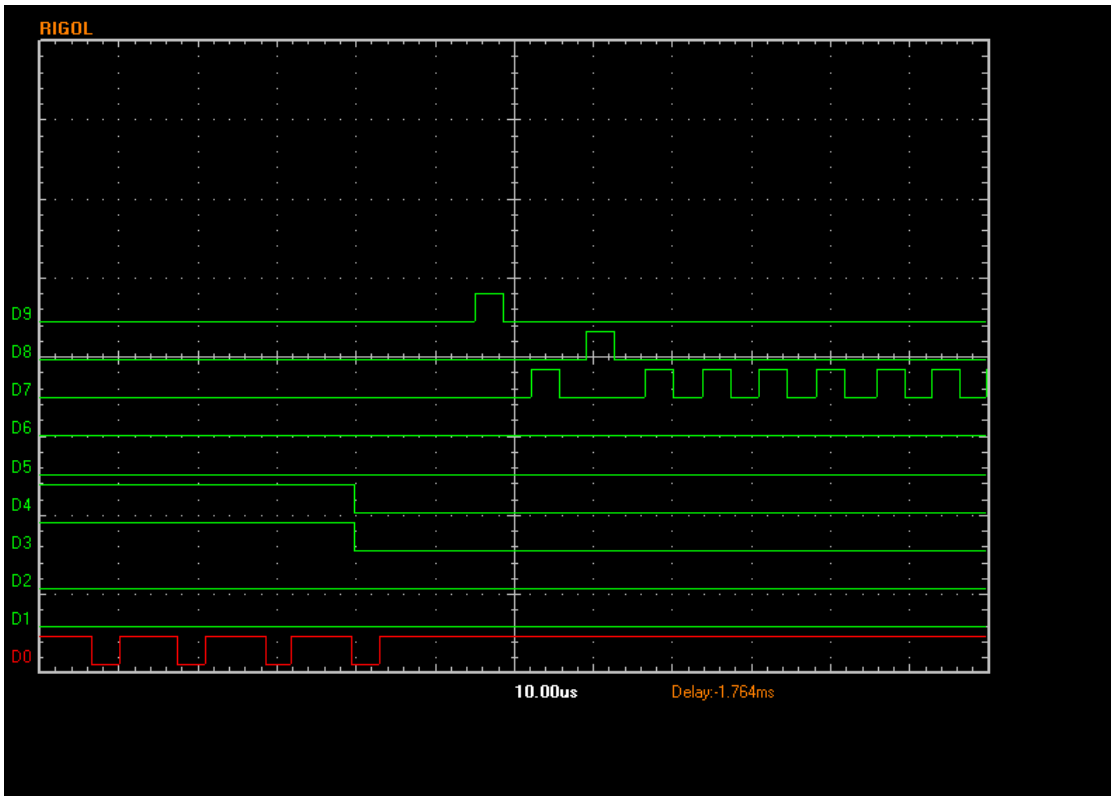
As can be seen, the corresponding ADC data is EOC low, Bit 30 low, SIG high and the other data bits from MSB low.

This is the delay between the initial ADC sync, which started the ADCs converting, to the completion of the ADC conversion and the reading of the ADC data. Inpout32.

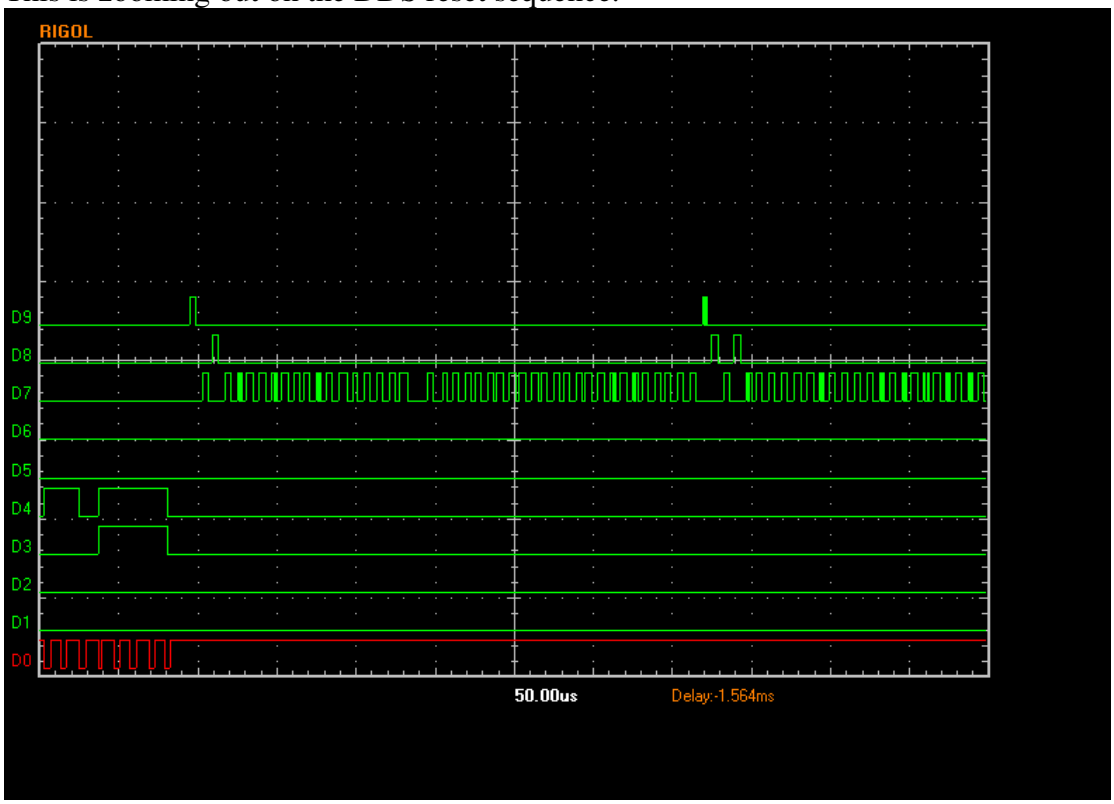


The ADC conversion takes about 330 us. The program detects that it has completed by the two ADC data lines going high; it then starts clocking out the result. Note also that the ADC speed selection (OSR) is again clocked into the ADCs on the first 6 rising edges. This time the full result is clocked out.

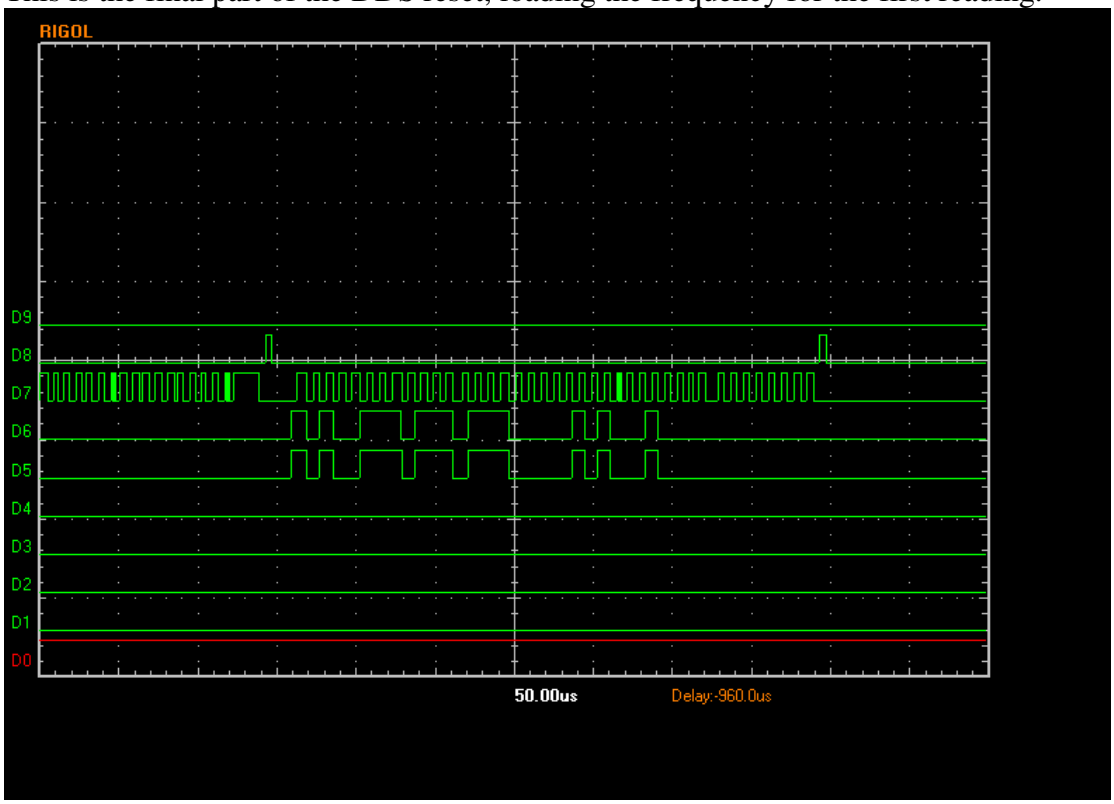
At the end of the clocking out of this first ADC conversion, the clock is left high; this stops the ADC from starting another conversion and holds its state idle. The DDS reset sequence then starts. The following series of traces are all inpout32.



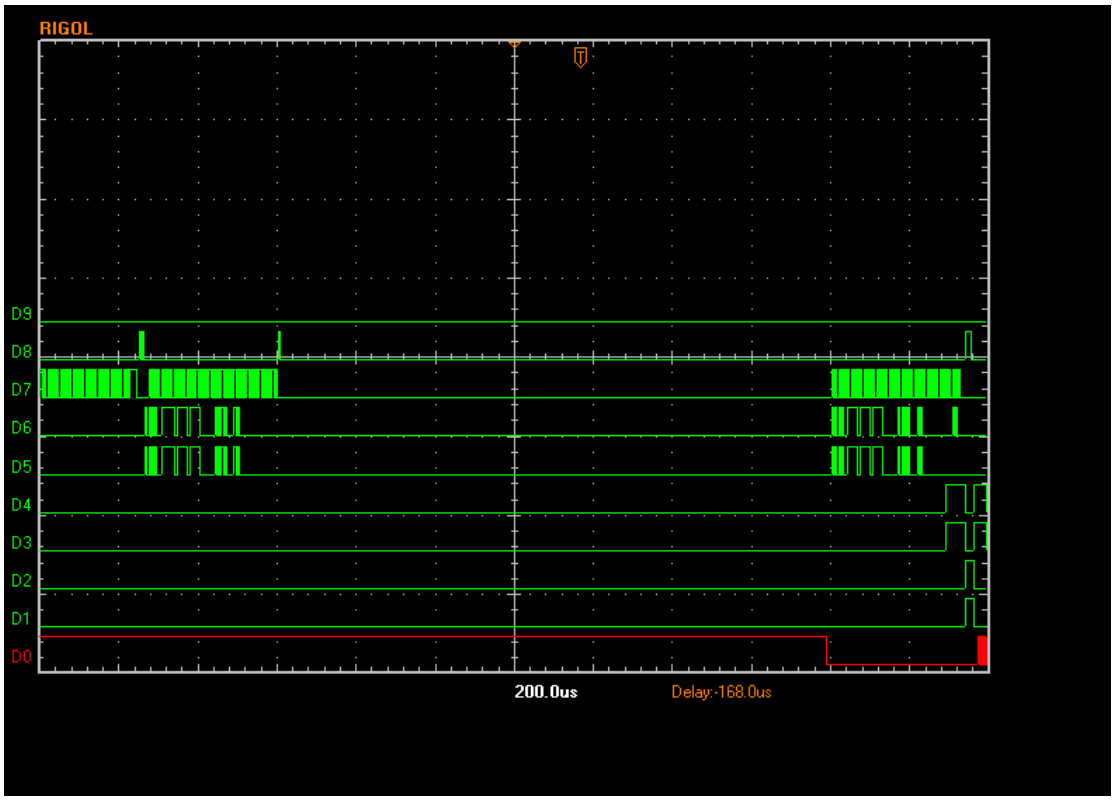
This is zooming out on the DDS reset sequence.



This is the final part of the DDS reset, loading the frequency for the first reading.

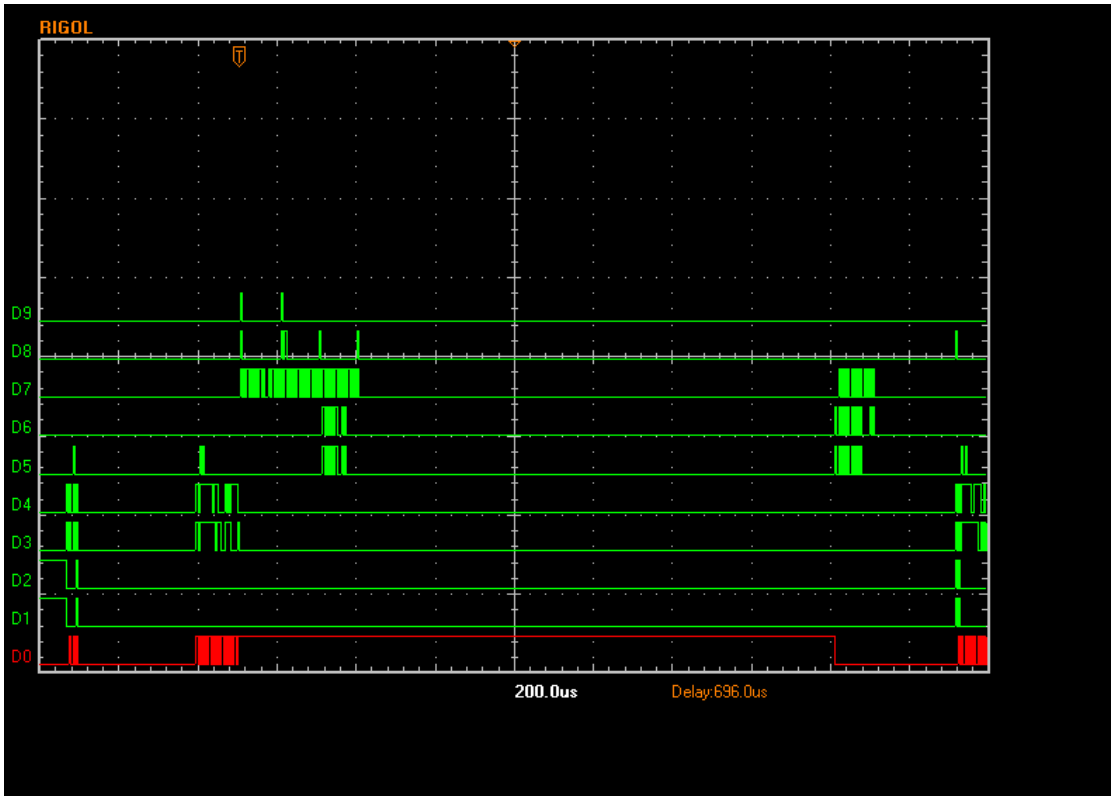


After this there is a sweep start delay, then ADC clock goes low to start the first conversion of the frequency point, here 1.4 msec. The delay is from the final pulse of FQ_UD (D8 trace) and the ADC clocks going low.



After this, the ADC conversion takes place (it finishes when the ADC data lines (D3 and D4) go high, following which the ADC data is clocked out. The first part of this is just visible on the edge of the trace. Whilst the ADC conversion is in progress, the next DDS setting is clocked in to the DDS. The FQUD signal however is not given until the ADC conversion is finished. Note that with inpout32 drivers on my PC, the time taken to load the DDSes is slightly longer than the time available during the ADC conversion at its fastest rate. This does not matter, it just we lose about 60 us on each conversion. WinIO removes this as can be seen below.

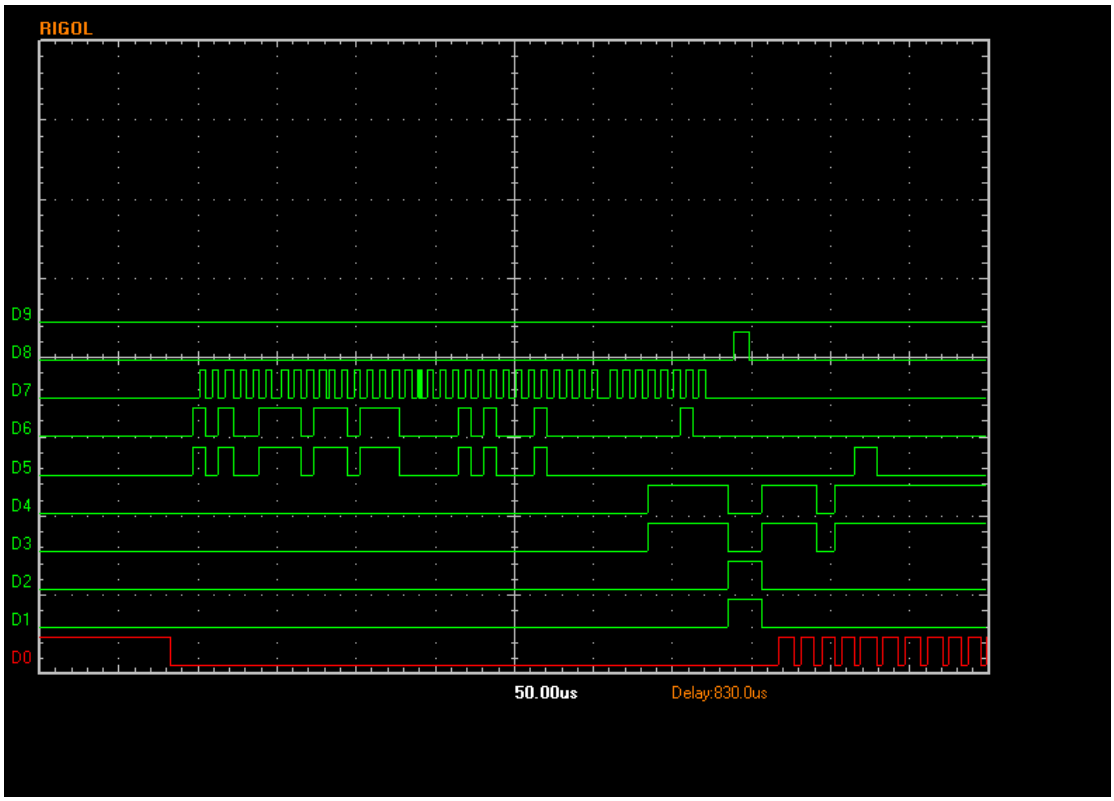
Here is the same trace using winIO. Note the significant speed increase (and slightly different sweep start delay – exact PC timing is always problematical)



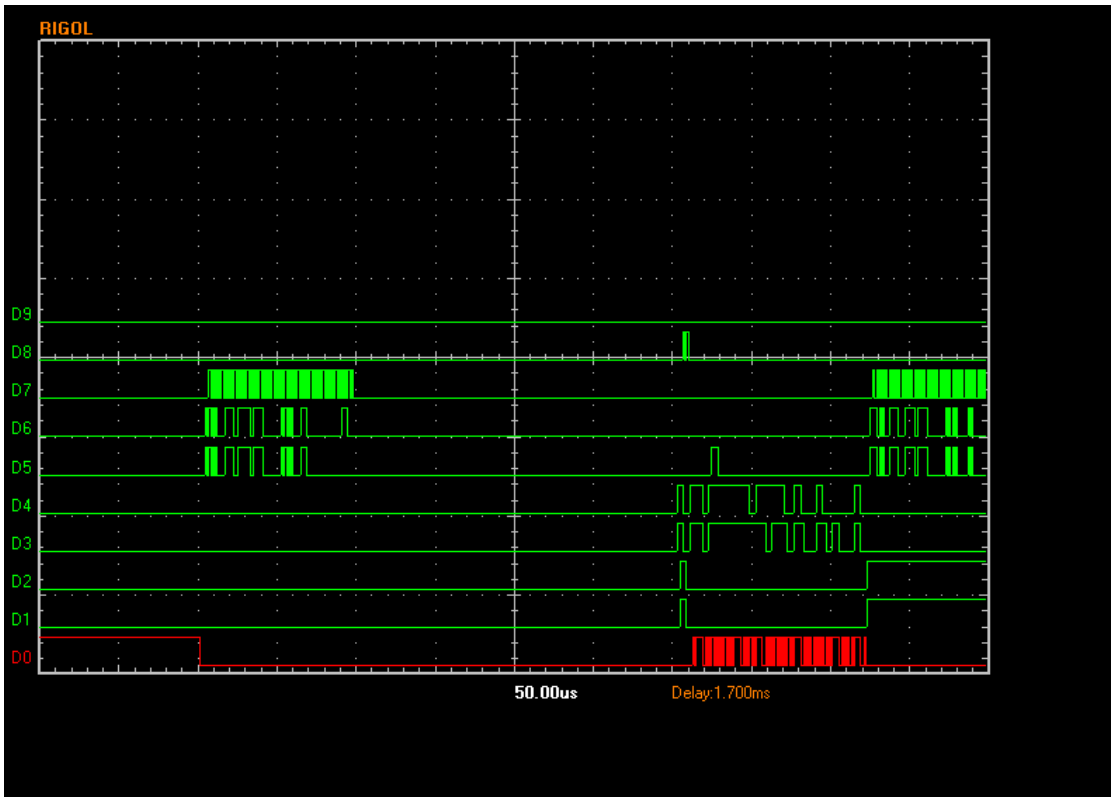
With the WinIO driver, the DDS load took 340 us, with winIO it takes 100 us.

Now the DDS load happens well before the end of the ADC conversion and the program waits until the ADC indicates completion before pulsing FQUD and reading the ADC data. The start of ADC data reading is just visible on the right.

Here is a zoom in on the first part of the ADC conver using inpout32 driver



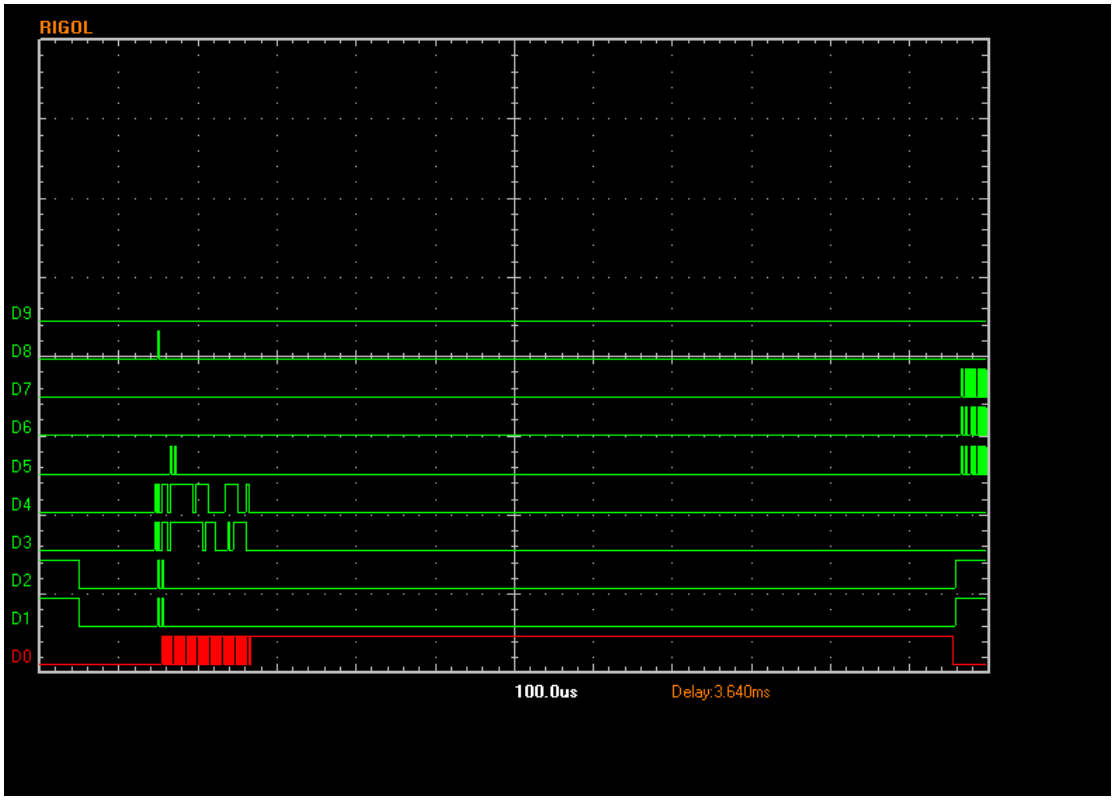
And the same with WinIO



Looking at the above, the ADC convert starts, the DDS is loaded but FQUD is not pulsed, so the DDS stays put until the ADC conversion is complete. The program waits until ADC signals completion by the data lines going high, at which point FQUD is pulsed and the ADC result clocked out. Again, the ADC speed setting is clocked in on the first few edges.

Clocking out the ADC result takes about 110 us using WinIO.

AT the end of a set of 4 phase readings, there is the programmed adc frequency step delay. Here the ADC clock is simply held high until the delay time is reached then the ADC clock goes low. Note that the delay is from the time that FQUD was pulsed, not from the time that the ADC reading was completed, in this case 10 divisions on the screen, or 1000 us.



Finally, at the end of a sweep, there is a delay before the next one starts.
This depends on the PC speed, number of data points in the scan etc. For this 200 point scan the delay was 3msec after which a new scan was started. With WinIO, in the following trace, can be seen the end of a sweep, the PC program delay in processing and displaying the trace followed by the start of a new scan and its sweep start delay and the start of the ADC conversion of the first data point.