**PANOSETI Quabo Firmware Release #8.1**

RR November 19, 2019

This is a follow-on to the version that Wei installed at UCSD a couple weeks ago. That version (I think we called it 8.0) was the first to incorporate separate data connections for White Rabbit and Data. This version corrects some problems of that one.

**First**, there was a problem with the elapsed-time clock starting up with random offsets of a few ns from the 1PPS synchronizing signal from the WR master (a Seven Sols WR-LEN in my tests). The cause of this was that we were using the wrong clock from the WR core to clock the rest of the system. There are two 62.5MHz clocks generated in the WR-core; one of them is used to clock the final flip-flop that generates the one\_pps signal (and much else). This clock is called clk\_ref\_62m5. There is another one called clk\_sys\_62m5 which was used to generate all of the other system clocks in v8.0. Changing this to clk\_ref\_62m5 resolves the “jumping time”problem.

**Second**, there were large timing violations in this design. There were no timing constraints at all on the main clock, and the source-synchronous ADC clock (for the pulseheight channel) which runs at 120MHz was showing timing violations on the order of 4ns. Jerome saw the effect of this in some of his tests.

The problem here cropped up because of an odd behavior of Vivado. Previous versions of the design, prior to including the WR core, permitted the use of the ADC\_clk input buffer to clock all of the serial-to-parallel flipflops; when the WR core was included this was no longer possible (this was due to the necessity of using “global” rather than “local” synthesis for the IP cores). So the tools tried to use global clock buffers, and incurred large routing delays causing the timing violations. To fix this, I directly instantiated local clock buffers in the “maroc\_dc.v” (maroc data capture) core.

There were other timing violations just due to the size of the design, and I used some different synthesis options to get to timing closure on the 100MHz system clock. In the end, the design meets timing. I’ve separated out the timing constraints from the physical ones (the pinout) to make things more easily understood.

**Finally** I implemented four-phase elapsed time logic, using a 250MHz clock, to achieve 1ns elapsed-time ticks.

I retained the use of J3pin2 as a jumper to set the IP address (jumper it to the adjacent ground pin to set the ‘0’ address, and leave it open to set the ‘1’ address), and I retained the other functions that Wei assigned on J3:

J3pin1 1PPS out

J3pin2 IP address set

J3pin3 WR\_UART\_RXD (input to the WR core)

J3pin4 WR\_UART\_TXD (output from the WR core)

J3pin5 WR\_UART\_RXD (input to the MIcroBlaze)

J3pin6 WR\_UART\_RXD (output from the MicroBlase)

I brought out a synchronized version of the OR of all of the **pulseheight triggers on SMA J1**. This was previously the 1PPS input (pre-WR) and I thought it might be nice to make it the 1PPS output, since the boards are already labeled this way, but that’s not a very useful thing once WR is working reliably. An indication of all of the system triggers might be useful.

I changed the functionality of the system a bit. We have a set of trigger masks, applied in the FPGA. Previously these masks affected both the PH mode and the Image mode. In this version the masks affect only the PH mode, which is I think the way it should be. We should talk through this.