IDLab mRICH Documentation

By Tommy Lam

Date: August 2018

Emails: [tommy9@hawaii.edu](mailto:tommy9@hawaii.edu)

[tommylam@vt.edu](mailto:tommylam@vt.edu)

Link to view: <https://docs.google.com/document/d/1eFXReqCd54kIyC2t8_Qbj3XF2IRHjioi1KzKH3wEyRI/edit?usp=sharing>

**Table of Contents**

[**Hodoscope**](#_1oreqg56s8p2) **1**

[Procedure:](#_tvgby1pbehdz) 1

[BIT files:](#_i2gjm3pl2cim) 7

[Notes on the SCROD BIT files](#_haxkmn4cskvs) 7

[Notes on the Hodoscope DC BIT files](#_tjjzwvc7pd4g) 8

[Problems and Solutions (when available)](#_iewod5v7gsdc) 8

[Register Maps](#_hlj8otb57zge) 10

[**PMT Readout**](#_6jajlinmo7kf) **13**

[Procedure](#_aihg6edxuhv7) 13

[Hardware Notes](#_9got4g8xfyd8) 14

[Firmware Notes](#_ilipynwfbnvi) 15

[Signals (UCF related stuff):](#_d14j4mkm61y8) 15

[Technical Changes](#_iuif89oogz74) 16

[Triggering](#_76it4axzzpr) 17

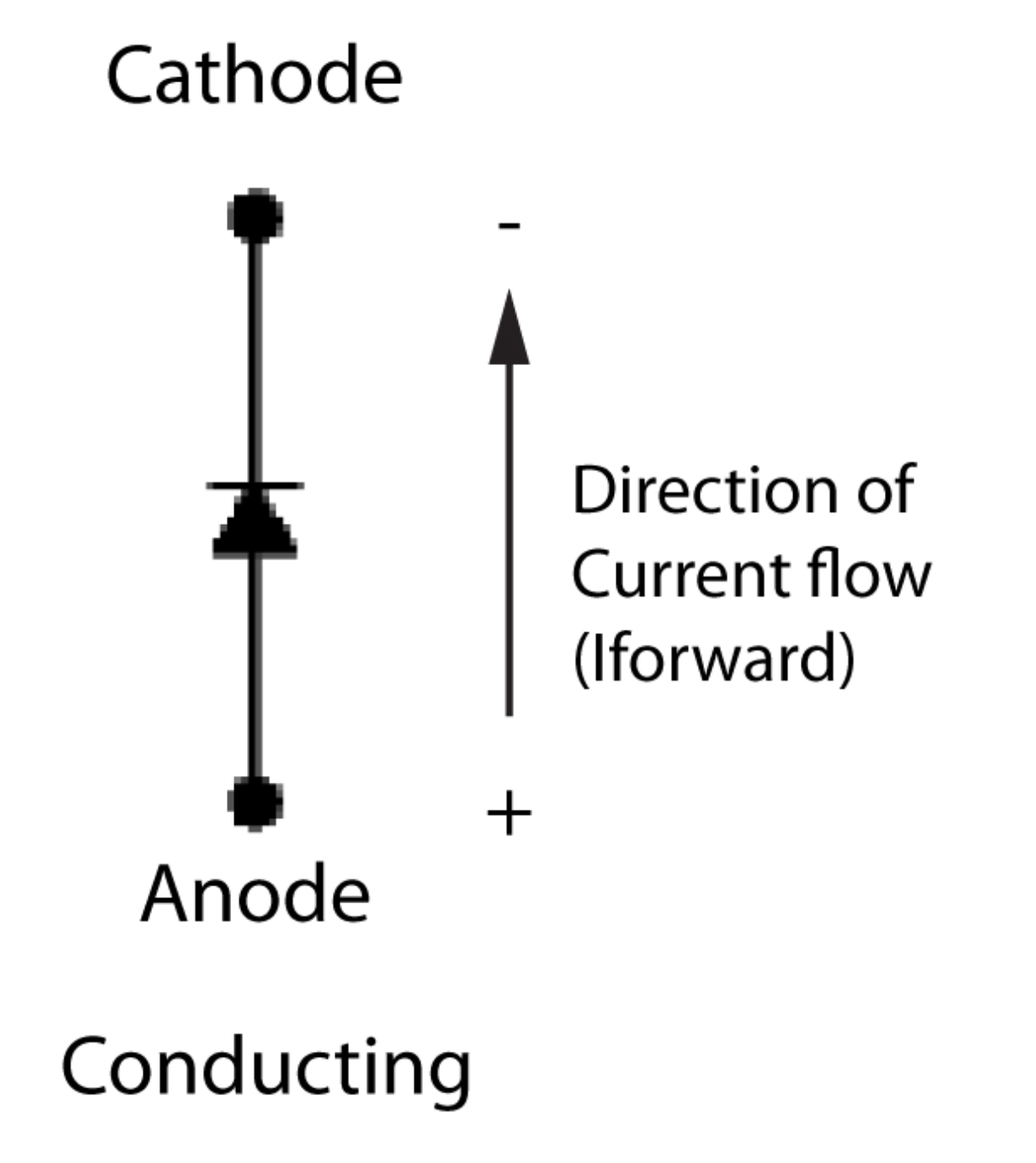
[Software Notes](#_76lt3aohftll) 17

[Problems and Solutions (if available)](#_f52s48mdyg6w) 18

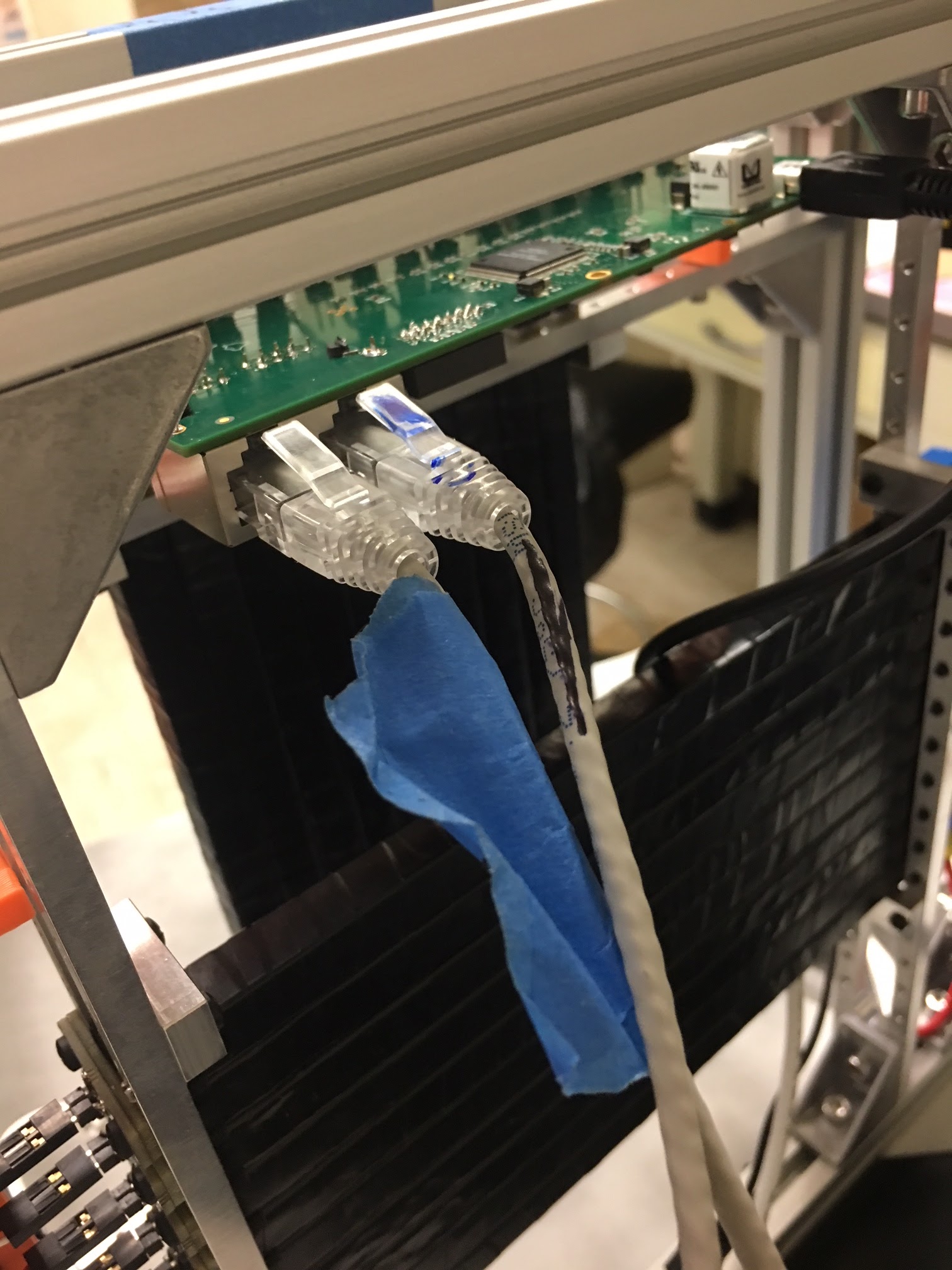
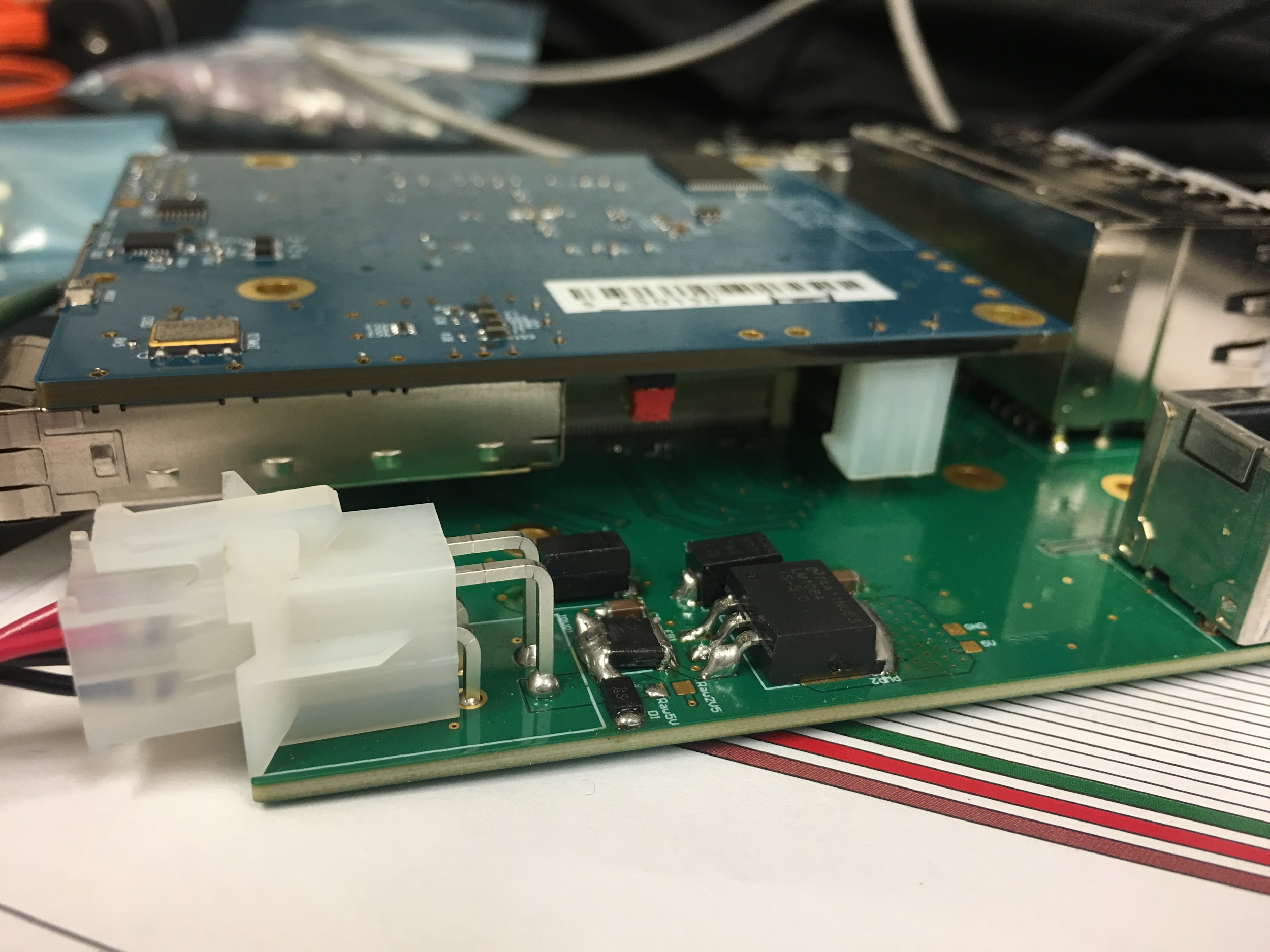
[**PCI Express Documentation (X710-DA4)**](#_ip31iedvkqlu) **19**

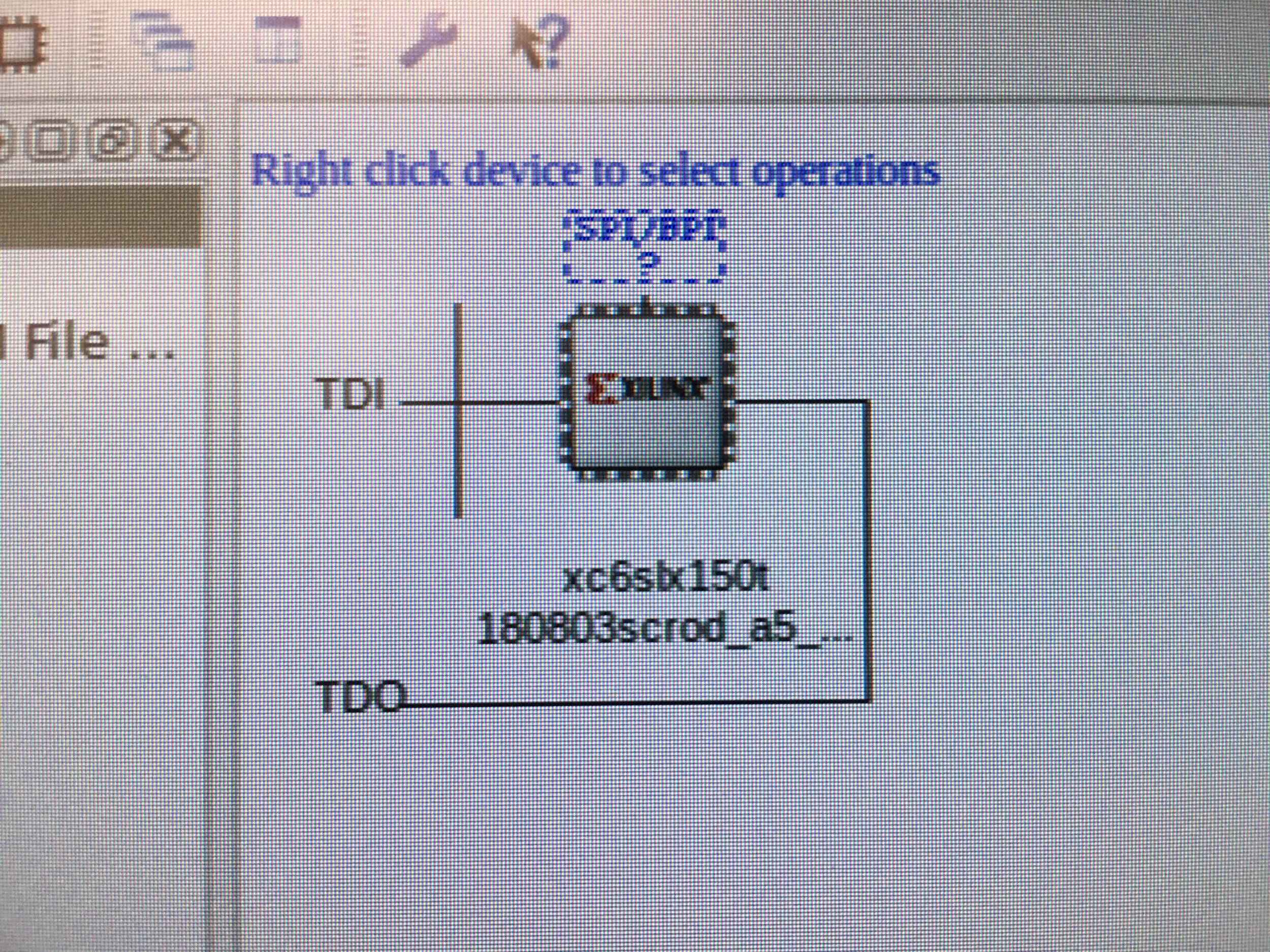
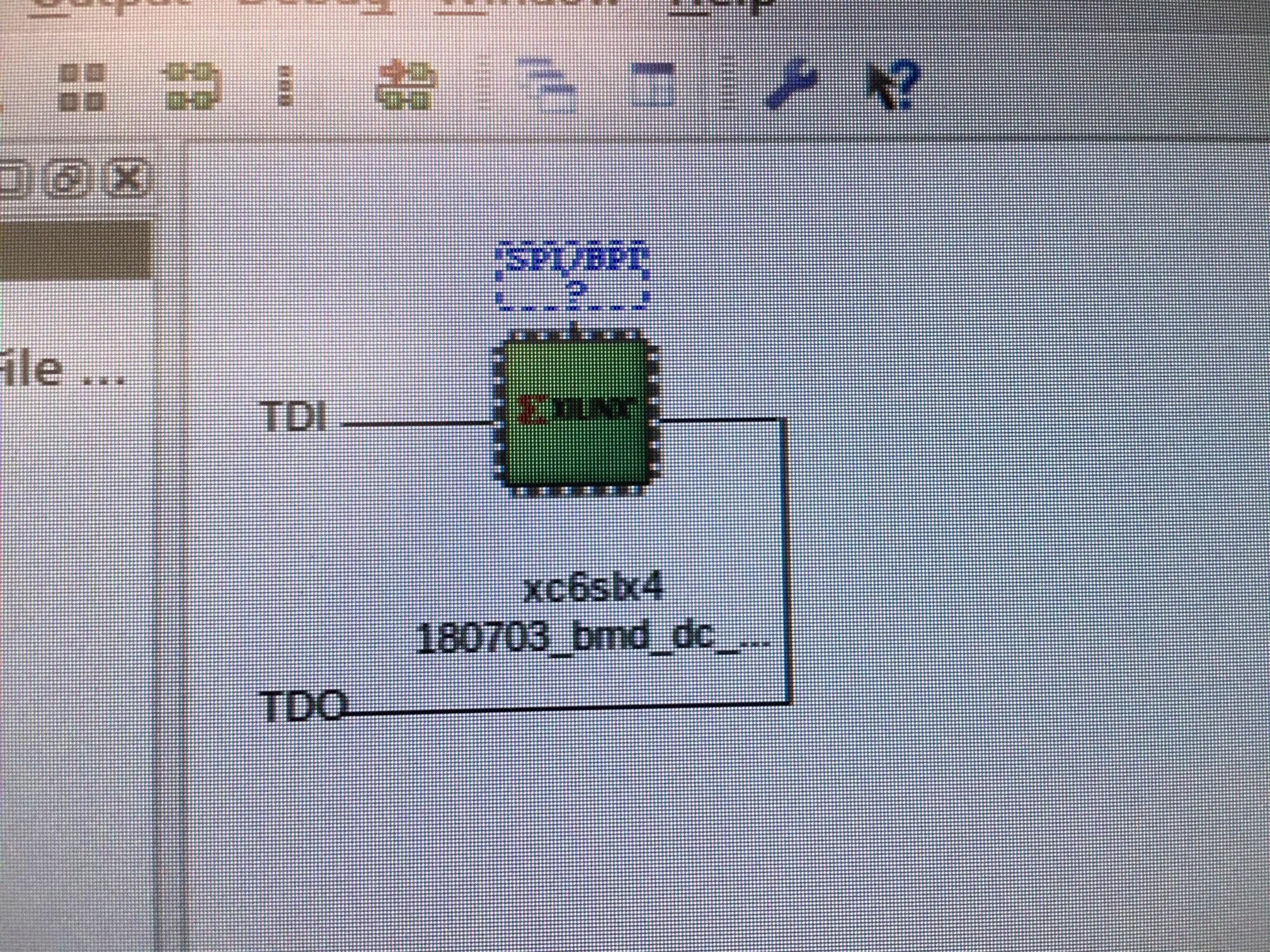
# Hodoscope

## **Procedure**:

1. What files to use:
   1. In dropbox, use the EIC-Beamtest\_FW/Fermi SCROD RJ45/SCROD\_A5\_RJ45
      1. Bit file should be in the main file
   2. mRICH\_hodo\_DC\_V1 should do the trick.
      1. Debug core shoud be in ./source 
      2. The bit file should be in ./ise
      3. The ISE .xise file is in ./hodo\_dc\_v1
2. Setup of the boards and hodoscope themselves:
   1. MPPC cathode and anode

The cathode (K) is on the minus sign (the tip of the arrow heads towards the cathode side. The convention we chose is the red part of the wire is connected to the cathode and the white side is connected to the anode.

* 1. MPPC to DC Channel mapping
     1. The channel mapping was chosen based on the length of the twisted pair wire. If one makes more twisted pairs with the right connectors, one can choose their own mapping.
     2. The previous mapping is here: <https://docs.google.com/spreadsheets/d/1aKKkmxEyzDeevKpB5wcshCR7V3Bk7OztbBq3_HpI1d8/edit?usp=sharing>
  2. Ethernet Cables
     1. Hodoscope DCs
        1. The taped ethernet cable should be in the port furthest away from the JTAG programmer slot
        2. Thus, the other cable should be closest to the programmer slot
     2. SCROD
        1. The taped ethernet cable should go on the side closer to the transition board (if the SCROD programmed light or the blue part of the scrod is facing up, the transition side of the board is on the bottom)
        2. Thus, the other cable should be on top.
  3. Powering the board
     1.  The left left (from the perspective of the picture) should be receiving about 4V (current draw of 0.8A preprogrammed, 1A programmed) while the right twisted pair is 6V (found that the current draw seems to be very low).
     2.  Note that the jumper is on JS-D
     3.  Values used during testing by Tommy (Yes, the voltage on this is higher than it should be -TL)

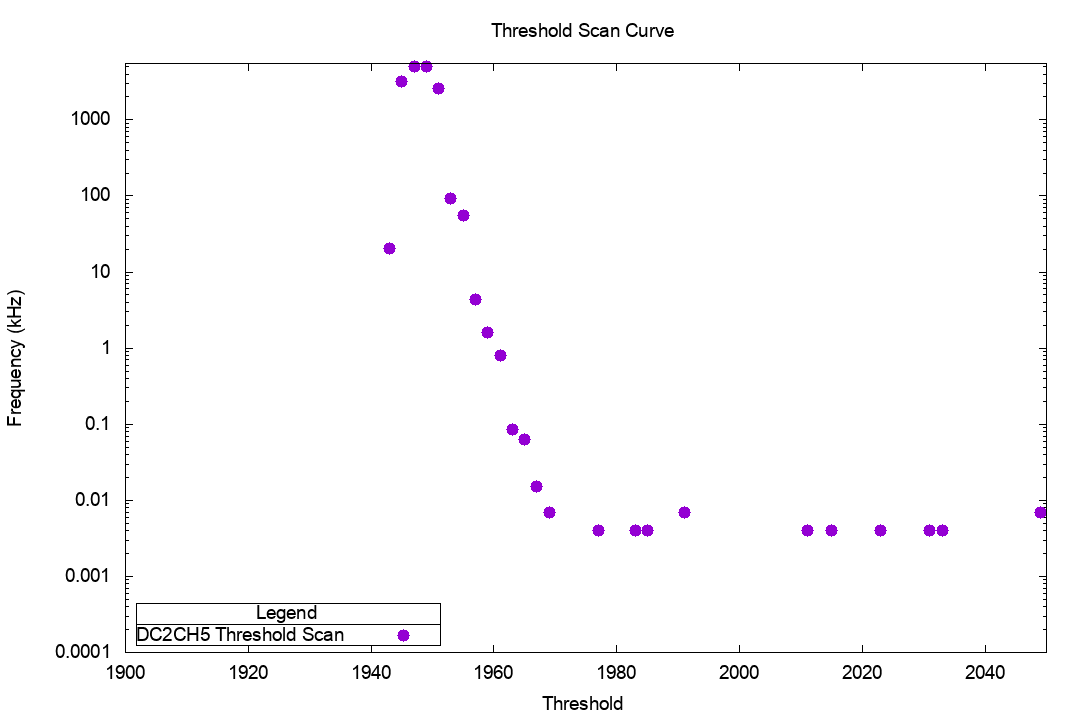
1. After powering up and plugging in all the appropriate fiber optics and JTAG Programmer(s) (programming the SCROD must happen first), source $PATH\_TO\_ISE/settings64.sh [in terminal], run impact or analyzer (literally type in “impact” or “analyzer” in terminal) and program the SCROD and the Hodoscope Daughter Cards (DCs) with their appropriate bit files (there should a be a warning if the wrong bit file was used since the bit files are FPGA dependent). 

Caption: (Sorry for the phone picture) -- The left is for the TX configuration. The right picture is for the SCROD configuration. The left one is green because it’s about to be programmed while the one of the right is white because it needs to be programmed (or forgot that it was programmed)

* 1. On the mRICH test bench, the path is /opt/Xilinx/14.7/ISE\_DS/settings64.sh
  2. Analyzer is good when wanting to run chipscope to monitor signals. Impact is good when wanting to program things quickly.
     1. To use impact, go to boundary scan, hit CTRL+I (or initialize chain), program the FPGA with the right bitfile, and you win
     2. On analyzer, connect with the jtag programmer, connect to the fpga, and configure the device with the bitfile and right debug\_core (extension: .cdc)
  3. Information on what bit files are mentioned in a later section. However, they should be relatively clear from the testbench
  4. HOW TO CHECK IF THINGS ARE PROPERLY PROGRAMMED
     1. It takes maybe 10 seconds for the SCROD to establish a connection with the computer. If it takes more than 30 sec., something’s wrong.
        1. Once one see’s “Connection Established” for the ethernet ports (not the one for internet)
     2. Check to see if you can readback from the SCROD (send BMD\_sendCMD.py 0 d 0 0.
     3. After this, continue with the other SCRODs one at a time (or all at once?)
     4. If there is any permutations of the ethernet cables, recall that the recalibration should probably be done.

1. Run **BMD\_TXConfig2.py** like so: “sudo ./BMD\_TXConfig2.py f 1 56 100 0” where
   1. “f” indicates that all DCs are being configured (can be changed to 1 = 0001 so DC 1 only, 2 = 0010 so DC 2 only, 3 = 0011 so DC 1 and 2 only, etc)
   2. “1” is calibration mode 1, where the threshold scan occurs
   3. 56 == 56V set on the board. Further mention of this will come in an incoming section (This step might not matter, using Khanh’s BMD setup, we could still threshold and see ADC counts with no actual HV or step 2)
   4. 100 == 100Hz or the ideal noise rate, randomly chosen. It doesn’t really matter at this point. This could change in the future.
   5. 0 == Extra threshold, I didn’t know we had an extra threshold anywhere but that’s the trailing zero at the end
   6. You should see thresholds at ~1900 or 2000 (which might correspond to 1.2V = Vped)

Output of step 4:



Example output of step four (after you take the data and plot it). Note that the threshold should somewhat be in this range.

1. “sudo ./BMD\_TXConfig2.py f 2 56 100 0” for ADC counts above threshold
   1. “[A]ssuming that the allowable incident angle is 180 degrees, and 0.04 m2 for the active area of one tracker plane (20 cm by 20 cm). The expected muon event rate should be roughly about 17.6 Hz. Using 17.6 Hz as an ideal rate the secondary scan will start at the baseline found from the first two stages and add increments of 1 ADC count to all baseline values for every channel in the system until a event trigger rate closes to the ideal rate is found.“ - Khanh, pg. 34
2. “sudo ./BMD\_TXConfig2.py f 3 56 100 0” to set the ideal values from previous steps

For the setting bias voltages script, the documentation for that is from here:

<https://docs.google.com/spreadsheets/d/1aKKkmxEyzDeevKpB5wcshCR7V3Bk7OztbBq3_HpI1d8/edit?usp=sharing>

NOTE: Section 5.1 of Khanh’s thesis explains what is going on in the last three steps of this procedure. Doing a threshold scan searching for the baseline (finding where zero is), doing a fine scan (finding accurately where the baseline is), and offset scan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| all in V | A | B | C | D |
| 1 | 54.9 | 54.66 | 54.42 | 54.72 |
| 2 | 54.75 | 54.6 | 54.54 | 54.54 |
| 3 | 54.93 | 54.48 | 54.39 | 54.48 |
| 4 | 54.78 | 54.33 | 54.36 | 54.45 |
| 5 | 54.78 | 54.45 | 54.51 | 54.45 |
| 6 | 54.69 | 54.39 | 54.63 | 54.66 |
| 7 | 54.87 | 55.29 | 54.66 | 54.78 |
| 8 | 54.87 | 54.39 | 54.84 | 54.69 |
| 9 | 54.84 | 54.42 | 54.63 | 54.72 |
| 10 | 54.9 | 55.2 | 54.57 | 54.6 |

Caption: Documentation of MPPC optimal voltage setting (breakdown voltage = V\_{ij} - 3V)

## BIT files:

This section refers to the bit files potentially used before and after the fermilab testing.

Commands used: “sudo ./BMD\_sendCMD.py DC# f reg# reg\_value” to change a register value and “sudo ./BMD\_sendCMD.py DC# d reg# 0” to read back the register value

### Notes on the SCROD BIT files

* + (control DC bit file: 180630\_bmd\_dc\_top\_v2\_csp3\_good.bit)
  + A5\_top\_001.bit -- it was possible to change the DC registers with the correct readback
  + A5\_top\_002.bit -- it was possible to change the DC registers with the correct readback
  + A5\_top\_003\_good.bit -- it was possible to change the DC registers with the correct readback
  + A5\_top\_004.bit -- DEADBEEF (regardless of DC bitfile) using the same interface
  + A5\_top\_005.bit -- it was possible to change the DC registers with the correct readback
  + A5\_top\_006.bit -- DC registers were not changable (the “f” command was sent, but the register always read back 0)
    - This could be the reason of the issue we had at Fermilab with the registers, at least towards the end game
  + 180803\_a5\_top\_01.bit -- new one with the modified ASIC mask protocol

### Notes on the Hodoscope DC BIT files

* + There was a UCF change on the BMD DC bit file during the FERMILAB beam (this affected TX, AUX, Data output, and SC\_DC\_CLK [or MC\_CLK])
    - This was clear from the triggers not showing up on Chipscope
  + Recommended: the 180703\_bmd\_dc\_top\_v2\_compile3\_dbg2.bit
    - New firmware might have been generated but further testing is required
  + Chipscope Debugging tools:
    - Debug\_core\_bmd\_dc\_v2.cdc
    - A new one was generated to look at readout values, readout register numbers, etc.

## Problems and Solutions (when available)

* Software Issues:
  + Issue: Scripts not running or they just don’t proceed
  + Potential Solutions:
    - Interface is not correct (can be checked by looking for the interface with the correct IP address (192.168.20.1) -- One could in theory change this in the firmware but it apparently breaks things so best to just leave it like this
    - Make sure the IP address is correct in the software
    - Check the SFP module: The AFBR-5710PZ modules WORK while the AFBR-57R5AEZ did not work
    - SCROD Address: 192.168.20.5 (you don’t ping this, only hear back from it)
    - The board is not powered properly (if the raw 6V or 5V regulator is not drawing any current, like about 0.183A, then you won’t get any communication with the boards
  + Issue: A programmed DC and SCROD can’t **change registers** (using sendCMD)
  + Potential Solutions:
    - Make sure you’re using the right bit files (please see the bit file section above)
    - Make sure you have the ethernet cables in the right orientation. A good way to check this is with a multimeter and compare it to the schematic
    - Reprogram the setup or power cycle (if after a long period of time)
  + Issue: **DEADBEEF**
  + Potential Solutions:
    - Wrong BIT file or not programmed
    - Ethernet cable isn’t plugged in
    - DC not powered on
  + Issue: Where is the **threshold**?
  + Answer: We found the threshold is about 1900 where anything above and below at does not trigger. This is evident from looking at the trigger bits. If you trigger on trigger bits (ex. 10010 or any variation thereof), you’ll see the baseline. We note that even at 1950 and 1850, we see nothing.
    - This was changed by using BMD\_sendCMD.py DC# b ch 1900 where DC# = 1,2,3, or 4, and ch = 2\*n for n = 0,1,2,3,4…, 13, 14
    - As a note, the default threshold Khanh had set up was 2048. When a 5mVpp signal (from a pulser) was put in (with a gain of 7, 35mVpp), we were still seeing trigger bits. Why? Because there was a rising edge that was much larger than noise or background (which is hopefully a good thing)
    - To do this, one can use “./setting\_TXregisters.sh” to set the registers to some values pre-defined in the register folder. (shell script that uses the BMD\_sendCMD above)
  + Issue: Not seeing anything during **threshold scan** (i.e. finding the baseline)
  + Comments and Potential Solutions:
    - SOLUTION: The issue seemed to be that there was a communication error since the firmware only accepted ASIC\_MASK 0001, 0010, 0100, and 1000. We added 1111 and that seemed to be the trick.
      * The change was made on the scrod side
    - Biggest thing: we know we can see the triggers bits on chipscope for the TX and that trigger bits is related to pulse count in rising\_edge and pulse\_count => trig\_count => rb\_value => register 69 & 96 (this is either with the baseline or the pulser so this is concerning)
      * Attempt 1: is RB\_ENA = 0 a problem? If so, why not use force it to be 1? What will the result of that be (can do this using the DC registers in the table below)
    - Notes: We think it’s a firmware issue since the scripts still work on Khanh’s BMD setup
    - Check Vped and make sure it equals something reasonable like 1.2V (on the board itself). This value also determines what the internal trimDAC is inside the targetX and thus, where your baseline is (seeing as the comparator is comparing any trigger signals to the internal trimDAC)



Caption: The threshold scan should look something like this while it’s covered well-enough.

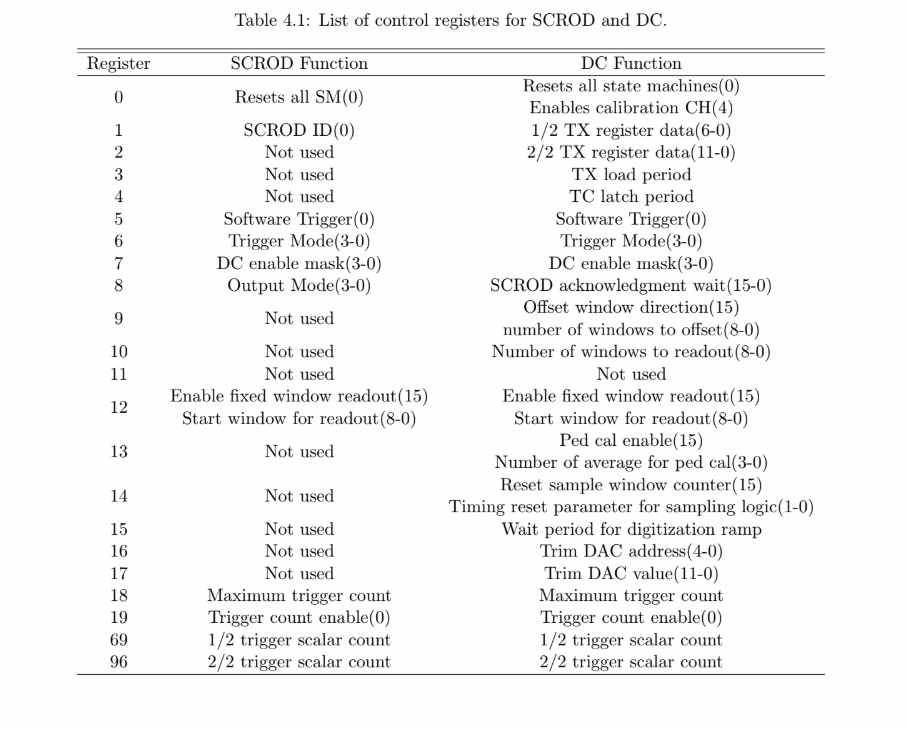
* + Issue: Can I add new trigger ports to the BMD\_DC firmware **debug core for chipscope**?
  + Answer: The answer appears to be no since it seems difficult to add the actual connections inside the FPGA. It is easier to remove some of the signals one might not be interested in and just change the connection.
  + Issue: Not all four DCs can be connected (the fourth one deadbeefs on you)
  + Comments
    - This seemed to be a hardware issue
  + Issue: DEADBEEF appears, doesn’t appear, and then appears
  + Comments
    - This happened after I tried to use jumpers and change the voltages. For SCROD II (the II sould be marked on the SCROD and RJ45), no jumpers are needed.
      * For SCROD I and SCROD II, a jumper on J5-D might be needed since the regulators are working differently
    - I was able to get an instance with some of the boards working and then one by one, it all turned to deadbeef but slot 3 (just the third slot, not any particular DC)

## Register Maps

Address Register Map for TARGETX

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **PCLK #** | **Reg #** |  |  |  |  | **Default setting** | **[hex]** |
| 1 | 0 | Trigger Threshold | Ch. 1 |  |  | 2048 | 800 |
| 2 | 1 | Wbias | Ch. 1 |  |  | 985 | 3D9 |
| 3 | 2 | Trigger Threshold | Ch. 2 |  |  | 2048 | 800 |
| 4 | 3 | Wbias | Ch. 2 |  |  | 985 | 3D9 |
| 5 | 4 | Trigger Threshold | Ch. 3 |  |  | 2048 | 800 |
| 6 | 5 | Wbias | Ch. 3 |  |  | 985 | 3D9 |
| 7 | 6 | Trigger Threshold | Ch. 4 |  |  | 2048 | 800 |
| 8 | 7 | Wbias | Ch. 4 |  |  | 985 | 3D9 |
| 9 | 8 | Trigger Threshold | Ch. 5 |  |  | 2048 | 800 |
| 10 | 9 | Wbias | Ch. 5 |  |  | 985 | 3D9 |
| 11 | 10 | Trigger Threshold | Ch. 6 |  |  | 2048 | 800 |
| 12 | 11 | Wbias | Ch. 6 |  |  | 985 | 3D9 |
| 13 | 12 | Trigger Threshold | Ch. 7 |  |  | 2048 | 800 |
| 14 | 13 | Wbias | Ch. 7 |  |  | 985 | 3D9 |
| 15 | 14 | Trigger Threshold | Ch. 8 |  |  | 2048 | 800 |
| 16 | 15 | Wbias | Ch. 8 |  |  | 985 | 3D9 |
| 17 | 16 | Trigger Threshold | Ch. 9 |  |  | 2048 | 800 |
| 18 | 17 | Wbias | Ch. 9 |  |  | 985 | 3D9 |
| 19 | 18 | Trigger Threshold | Ch. 10 |  |  | 2048 | 800 |
| 20 | 19 | Wbias | Ch. 10 |  |  | 985 | 3D9 |
| 21 | 20 | Trigger Threshold | Ch. 11 |  |  | 2048 | 800 |
| 22 | 21 | Wbias | Ch. 11 |  |  | 985 | 3D9 |
| 23 | 22 | Trigger Threshold | Ch. 12 |  |  | 2048 | 800 |
| 24 | 23 | Wbias | Ch. 12 |  |  | 985 | 3D9 |
| 25 | 24 | Trigger Threshold | Ch. 13 |  |  | 2048 | 800 |
| 26 | 25 | Wbias | Ch. 13 |  |  | 985 | 3D9 |
| 27 | 26 | Trigger Threshold | Ch. 14 |  |  | 2048 | 800 |
| 28 | 27 | Wbias | Ch. 14 |  |  | 985 | 3D9 |
| 29 | 28 | Trigger Threshold | Ch. 15 |  |  | 2048 | 800 |
| 30 | 29 | Wbias | Ch. 15 |  |  | 985 | 3D9 |
| 31 | 30 | Trigger Threshold | Ch. 16 |  |  | 2048 | 800 |
| 32 | 31 | Wbias | Ch. 16 |  |  | 985 | 3D9 |
| 49 | 48 | Sbbias | Vramp | Dbbias |  | 1300 | 514 |
| 50 | 49 | Vdisch | Vramp | Dbbias |  | 0 | 0 |
| 51 | 50 | Isel | Vramp | Dbbias |  | 2500 | 9C4 |
| 52 | 51 | Dbbias | Vramp |  |  | 1100 | 44C |
| 53 | 52 | Qbias | PLL | Vqbuff |  | 1500 | 5DC |
| 54 | 53 | Vqbuff | PLL |  |  | 1300 | 514 |
| 55 | 54 | VtrimT | PLL | Vqbuff |  | 3500 | DAC |
| 56 | 55 | Misc Digital Reg |  | 12 bit |  | 0 | 0 |
| 57 | 56 | VadjP | Timebase | VAPbuff |  | 1152 | 480 |
| 58 | 57 | VAPbuff | Timebase |  |  | 1300 | 514 |
| 59 | 58 | VadjN | Timebase | VANbuff |  | 0 | 0 |
| 60 | 59 | VANbuff | Timebase |  |  | 0 | 0 |
| 61 | 60 | unused (T7 legacy) | Trigger | Itbias |  |  |  |
| 62 | 61 | Vbias | Trigger | Itbias |  | 900 | 384 |
| 63 | 62 | TRGGbias | Trigger | Itbias |  | 1100 | 44C |
| 64 | 63 | Itbias | Trigger |  |  | 1100 | 44C |
| 65 | 64 | SSPin LE | Timebase |  | 8 bit time | 143 | 8F |
| 66 | 65 | SSPin TE | Timebase |  | 8 bit time | 163 | A3 |
| 67 | 66 | WR\_ADDR\_Incr1 LE | Timebase |  | 8 bit time | 163 | A3 |
| 68 | 67 | WR\_ADDR\_Incr1 TE | Timebase |  | 8 bit time | 178 | B2 |
| 69 | 68 | WR\_STRB1 LE | Timebase |  | 8 bit time | 20 | 14 |
| 70 | 69 | WR\_STRB1 TE | Timebase |  | 8 bit time | 35 | 23 |
| 71 | 70 | WR\_ADDR\_Incr2 LE | Timebase |  | 8 bit time | 35 | 23 |
| 72 | 71 | WR\_ADDR\_Incr2 TE | Timebase |  | 8 bit time | 50 | 32 |
| 73 | 72 | WR\_STRB2 LE | Timebase |  | 8 bit time | 148 | 94 |
| 74 | 73 | WR\_STRB2 TE | Timebase |  | 8 bit time | 163 | A3 |
| 75 | 74 | MonTiming SEL | Timebase |  | 8 bit time | 40 | 28 |
| 76 | 75 | SSToutFB | Timebase |  | 8 bit time | 60 | 3C |
| 77 | 76 | CMPbias2 | Wilk | Sbbias |  | 737 | AAA |
| 78 | 77 | Pubias | Wilk | Sbbias |  | 3112 | AAA |
| 79 | 78 | CMPbias | Wilk | Sbbias |  | 1000 | AAA |
| 80 | 79 | TPGreg |  |  | 12 bit pattern | 2730 | AAA |

The following was taken from Khanh’s thesis (some of the signals were confirmed by looking at the firmware)



Klmscint control registers

|  |  |  |  |
| --- | --- | --- | --- |
| Register | bit(s) | DC |  |
| 0 | 11:8 | LED |  |
| 1 | 0 | Update | DAC  control |
| 2 | 6:0 | 1/2 TX register data |
| 3 | 11:0 | 2/2 TX register data |
| **4** | **9:0 -> 15:0** | **Control TDCNUM** |
| 5 | 15:0 | TX load period |
| 6 | 15:0 | TC latch period |
| 10 | 0 | Reset sampling logic | sampling |
| 11 | 15:0 | Sampling logic reset params |
| 31 |  | SROUT TPG |  |
| 32 | 15:0 | RAM ADDR | RAM |
| 33 | 21:16 | RAM ADDR [5:0] |
| 34 | 7:0 | RAM DATA WR |
| 35 | 0 | RAM UPDATE [0] |
| 1 | RAMRW [1] |
| 37 | 2:0 | USE EXTRIG [2:0] | Status Reg |
| 15:11 | EXTRIG Period[15:11] |
| 38 | 3:0 | Ped cal cn avg not used |
| 15 | Pedcalcreset not used |
| 14 | Ped man Enable < =>readctrl\_trigger\_raw> |
| 13:12 | Peddemuxfifooutputselet not used |
| 11 | Waveform fifo rst not used |
| 11 | Buffer ctrl fifo reset not used |
| 10:7 | Ped sub calc mode not used |
| 6 | Klm trig cal readout mode < =>readctrl\_trigger\_raw> |
| 5 | Use klm trig |
| 4 | Use scrod link |
| 39 | 15 | Use trig dec |
|  | 14:0 | Trigdec trig mask |
| 40 | 0 | CMDREG Waveform fifo rst | Event |
| 41 | 15 | Cmdreg ped calc start | ? |
| 9:0 | Cmdreg ped calc ASIC en | ? |
| 42 | 15:0 | Cmdreg ped calc winl en | ? |
| 43 | 15:0 | Cmdreg readctrl chan enable bit | ? |
| 44 | 0 | Cmdreg evtbuild start building event | Event |
| 45 | 0 | Cmdreg evtbuild make ready |
| 46 | 0 | Cmdreg evtbuild packet builder busy |
| 47 | 15:0 | Cmgreg trig scaler clk max | ? |
| 48 | 15:0 | Cmgreg trig scaler clk max trigdec | ? |
| 50 | 0 | Cmdreg software trigger | Readout  Control |
| **51** | **# of DC:0** | **Cmdreg readctr asic enable bits** |
| 52 | 0 | Cmdreg hardware trigger enable |
| 53 | 11:0 | Cmdreg readctrl trig delay |
| 54 | 8:0 | Cmdreg readctrl dig offset |
| 55 | 0 | Cmdreg readctrl radout reset |
| 56 | 6:0 | Readctrl lkbk width |
| 57 | 8:0 | Cmdreg readctrl win num to read |
| 58 | 0 | Cmdreg readctrl readout continue |
| 59 | 0 | Cmdreg readctrl reset event num |
| 60 | 7:0 | dac value [7:0] |
| 11:8 | dac addr [11:8] |
| 15:12 | dac number [15:12] |
| 61 | 15:0 | Cmdreg readctrl ramp length |
| 62 | 15:0 | Cmdreg readctrl use fixed dig start win |
| 63 | 3:0 | Amux\_s[3:0] | ??? |
| 7:4 | Amux\_s[7:4] |
| 8 | runADC |
| 9 | Current ADC reset |
| 10 | I dac update |
| 70 | 0 | Trig count ena | Trigger  Control |
| 71 | 0 | Trig count rst |
| 72 | 9:0 | Trigger ASIC control word [9:0] -> not used |
| 74 | 7:0 | Cmdreg wave trigASIC dump cfg [7:0] | Readout  Control |
| 11:8 | Cmdreg pedsub dataout mode[11:8] |
| 75 | 3:0 | Cmdreg pdaq data mode[3:0] |
| 76 | 15:0 | Cmdreg pdaq data chmask |
| 79 | 3:0 | Cmdreg txttb format |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Note that Registers 69 and 96 are related to Calibration Mode 1&2, and trig\_count / pulse count

Recall that the data output does not matter for this iteration, we only want triggers

Register 18 deals with maximum trigger count (how long you have to get scalers)

Register 19 deals with allowing things to trigger.

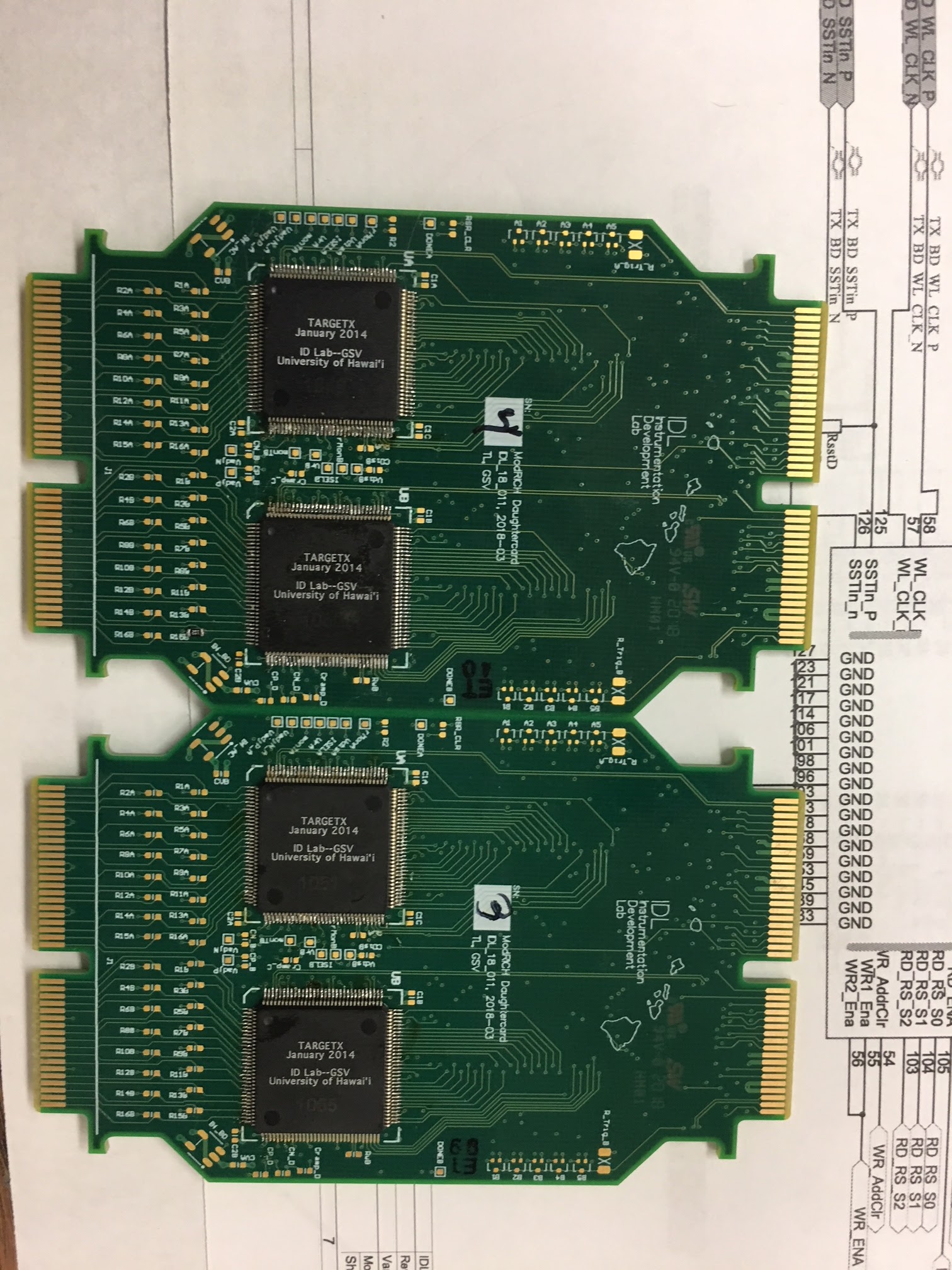
BMD\_sendCMD.py can be used to manually change the registers to read out reg. 69 and 96.

# PMT Readout

## Procedure

Needs work

## Hardware Notes



Not shorted boards (3 and 4) that needs population (as of 8/8/18). Resistance between 2.5V and GND should be several kOhms (maybe a couple kOhms, which is still okay)



There should be a jumper on J5-D for the SCROD + Transition Board setup

Note that PMT DC #1 and 2 should be unshorted and currently on here

Not seen is the carrier board on the other side of the DC.

## Firmware Notes

### Signals (UCF related stuff):

* + Many of the signals were changed to fit the pre-existing firmware
    - Ex. RD\_CS -> RD\_COLSEL
    - EX2. RD\_RS -> RD\_ROWSEL
  + WR1 and WR2 have the same ports and should be treated the same
  + The following signals are not used in the PMT DCs
    - MON\_TIMING
    - DONE
    - RD\_ENA
    - SR\_CLR
    - CS\_DAC
    - SCK\_DAC
    - DIN\_DAC
  + Length of some signals
    - DO == 16 downto 1 (for BUS A and B)
    - SAMPELSEL\_S == 5 downto 1
    - RD\_RS\_S == 2 downto 0
    - RD\_CS\_S == 5 downto 0
    - SR\_CLOCK == 3 downto 0
    - SHOUT == 3 downto 0
    - WR\_ENA == 3 downto 0 (both WR1 and WR2)
    - SAMPLESEL\_ANY == 15 downto 0
    - PCLK == 15 downto 0
    - SSTIN\_P == 7 downto 0
    - SSTIN\_N == 7 downto 0
    - TDCi\_TRG == 4 downto 0\*\*\*
      * “i” goes from 0 to 15

### Technical Changes

* + One of the major changes from transitioning from the KLM Scintillator Low Voltage Firmware to the mRICH PMT readout was the transitioned from 10 DCs and 10 TXs to 4 DCs and 16 TXs.
    - How this was compensated (by Tommy): Originally, TDC\_NUM\_CHAN == 10. However, since there are 4 DCs, TDC\_NUM\_CHAN == 4 now and a new TX\_NUM\_SAN == 16 (I think keeping the numbers separate was a better idea, at least in terms of generalizing code). This apparently required a lot of changes from going through many different files changing 10 ⇒ TX\_NUM\_SAN to allocating more memory in the wave generation data to change dina(10\*5 - 1 = dina(49:0) to dina(79:0) = dina(16\*5 - 1: 0 )
      * This change was done by modifying trig\_bram
      * Other signals that needed to change were the internal trigger signals, but this was a simple copy and paste
      * Note: while TDC\_NUM\_CHAN is probably something similar to Daughtercard\_number\_channel, the SAN at the end of TX\_NUM\_SAN has no real meeting besides something like Fuji-san.
      * In theory, if everything was changed back to 10, it should be equivalent to how the KLM had it before (some small changes beyond this will be necessary)
      * These constants should exist in: PMT\_readout\_Firmware/source/interfaces/klm\_scrod/tdc/source/tdc\_pkg.vhd
    - One thing that should be personalized are the PCLKs but due to time constraints, PCLKs are being shared amongst neighbors (i.e. I think TXs on the same board are sharing PCLKs)
    - Similar changes to PCLK were made to SSTIN\_N and STTIN\_P
    - There are a few LVDS signals going to banks that cannot handle LVDS\_25 signals so BLVDS\_25 signals (bussed LVDS signals) can be used instead
      * The particular signals were like: WL\_CLK and SSTin
      * Resource: <https://www.xilinx.com/support/documentation/data_sheets/ds162.pdf> (table 34, pg. 44)

### Triggering

* + We shouldn’t care too much about the signals right now. I think the biggest thing is that we might want to transition to just software triggers for now and worry about triggering on the hodoscope much later
  + This can probably be done using the KLM software
  + End goal: use the hodoscope to trigger the PMT readout.

## Software Notes

* KLM Software comes from Chris Ketter and Richard Peschke, modifications made by Tommy Lam and Dawei Yang

## Problems and Solutions (if available)

* Issue: The regulators get very, very, very hot (w/ all the DCs, ~120 deg C)
* Comments:
  + Even without the PMT DCs, the board gets to be 60 degrees C with a current draw of 0.4A before programming. However, this is comparable to the SCROD on the RJ-45 (I.e. the Hodoscope side), where the current draw was ~0.4A before programming and ~0.7A after programming.
  + PLEASE CHECK T
  + With a signal populated board, the temperature increases to 80 deg C without programming. With an UNPOPULATED board, there is no change. Thus, there’s probably a short on the PMT DCs.
    - Voltage = 6, Current = 0.4 with one board, Current = 0.7A with two boards, Current ` 1A with three boards (unprogrammed)
  + **SOLUTION**: What’s probably happening is that the 3.3V and 5V are fighting to power the board. The best thing to do is to put a jumper on J5-D. What this does is shut off 3.3V so only the 5V power is supplying the board with voltage.
* Issue: The DCs are shorted and drawing way too much current
* Comments:
  + Check the resistance between GND and 2.5V (on the output side of the board, they should be adjacent to each other). The resistance should be on the kOhm scale (~1kOhm should do).
  + Standard current draw is on the order of 0.1-0.2A. Therefore, the total current draw for the daughter cards should be ~ 0.6A.
  + A nice thing to do is to use a thermal camera to check for where the shorts are. This doesn’t always work but can be a good indicator in the worst case scenario.
  + If not visual on the board, it could be a problem internally in the targetX (i.e. if the targetXs were dropped or damaged).

# PCI Express Documentation (X710-DA4)

* What was the main issue: There were permission issues with the one SFP module model. Using a different SPF module was allowed by the PCI Express and we proceeded forward.
* What we thought the issue was: The computer was having issues with the PCI express so we thought we needed to “turn off” the flag that checks for whether a device is licensed or not.
  + How was this done:
    - github link: <https://github.com/terpstra/xl710-unlocker>
    - Improved upon instructions:

<https://sourceforge.net/p/e1000/mailman/message/34988514/>

* + - Summary of instructions
      * Download the github files and nvmupdate (the latter comes from intel)
      * open it up, and modify mytool (requires the right ethernet name [can use ‘ifconfig’] and device number [X710-DA4’s number was 1572 or 0x1572])
      * make mytool (really good idea to read the instructions up above at this point)
        + to get a record of everything, “sudo ./mytool 0 0x8000> somefile”
        + This will be useful for when you mess up big time
      * go to “somefile” and look for four 000b’s in a row (separated by 6 other numbers)
        + it should be somewhere around 00006870
        + “ If your record is not at this location, run mytool 0 0x8000 > somefile, and start looking for 4 occurrences of "000b" that are separated by 0xc addresses and repeat four times. This is how I found my records, since the NVM image from Intel does not have pointers to the correct location.”
      * AFTER FINDING THE SIGNALS, make the appropriate modifications in mypoke.c (the interface [ex. Eth0], bias [ex. 0X6870], and what you want to change line to [example: changing 2b0c to 230c or in our case, we had 6b0c to 630c])
        + make mypoke and run it
        + NOTE: we had a checksum error that we couldn’t figure out but it proper signals were still modified
      * after that, run nvmupdate -u to see if anything seriously changes (you might not need ot do this but it seems to be a pretty good safety check)
      * when everything seems okay, reboot the system and see if the changes you make work
    - TROUBLESHOOTING
      * If you see “PXE: PCI Vendor and Device ID don’t match,” you’ve probably bricked the card and this requires a lot of work to change.
      * If you don’t see that error but don’t see the Intel X710, just go back to mypoke and change the appropriate bit back to how it was before and reboot.