FELIX LPGBT Readout for RD53A

Ismet Siral , Carlos Solans , Laura Jeanty,
Roman Müller, Marco Travato

Foreword

- These slides are combination of my experience and the information listed in this detailed document provided by the Argonne Lab

- If you want to find more detailed information, please check the following documentation souces:

  - Argonne's readout documentation: https://cds.cern.ch/record/2690124/files/ATL-COM-DAQ-2019-160.pdf

  - RD53A Manual: https://cds.cern.ch/record/2287593/

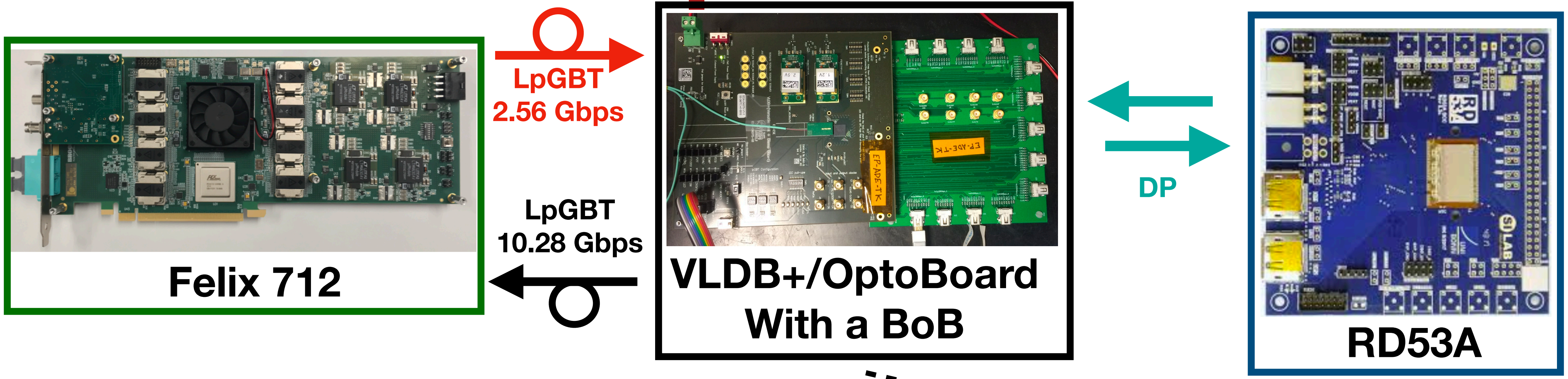
  - PiGBT Manual: https://pigbt.web.cern.ch/manual/overview.html

  - Felix Manual: https://atlas-project-felix.web.cern.ch/atlas-project-felix/user/felix-user-manual/versions/4.0.6/felix-user-manual.pdf

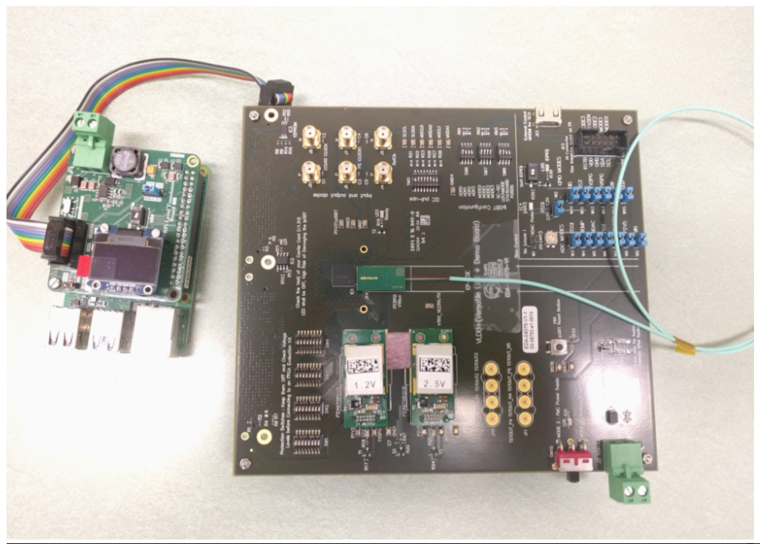
  - YARR Manual: https://yarr.web.cern.ch/yarr/

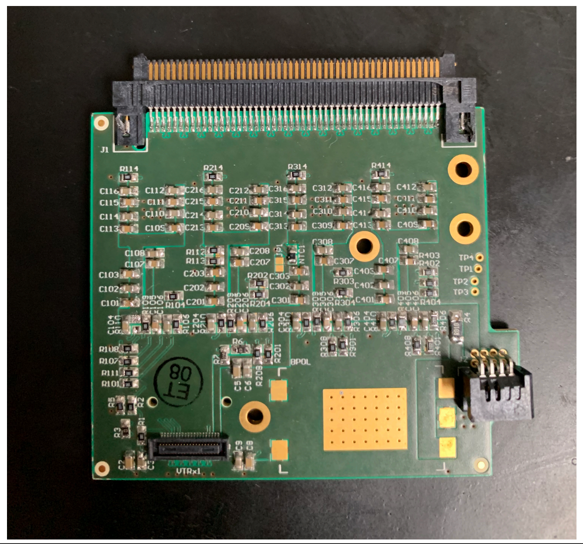
Hardware Info & Requirments

Hardware Setup



- The current setup requers the following hardare:
  - A Felix 712 Card connected to a Felix PC
  - An LpGBT Chip enclosed in eithers: VLDB+ board or an OptoBoard
  - A Break out board that accomponies the LpGBT chip board.
  - RD53A Chip and various cables.





VLDB+	OptoBoard
• Currently Configured by raspberry pi through I2C or using USB-I2C through PC. • The VLDB+ (EDA-04075) is a board designed by CERN's EP-ESE	• Three versions exist (V0,V1,V2) • Configured via IC (Fiber) • Hosts multpile LPGBT chip

Details About Opto Boards

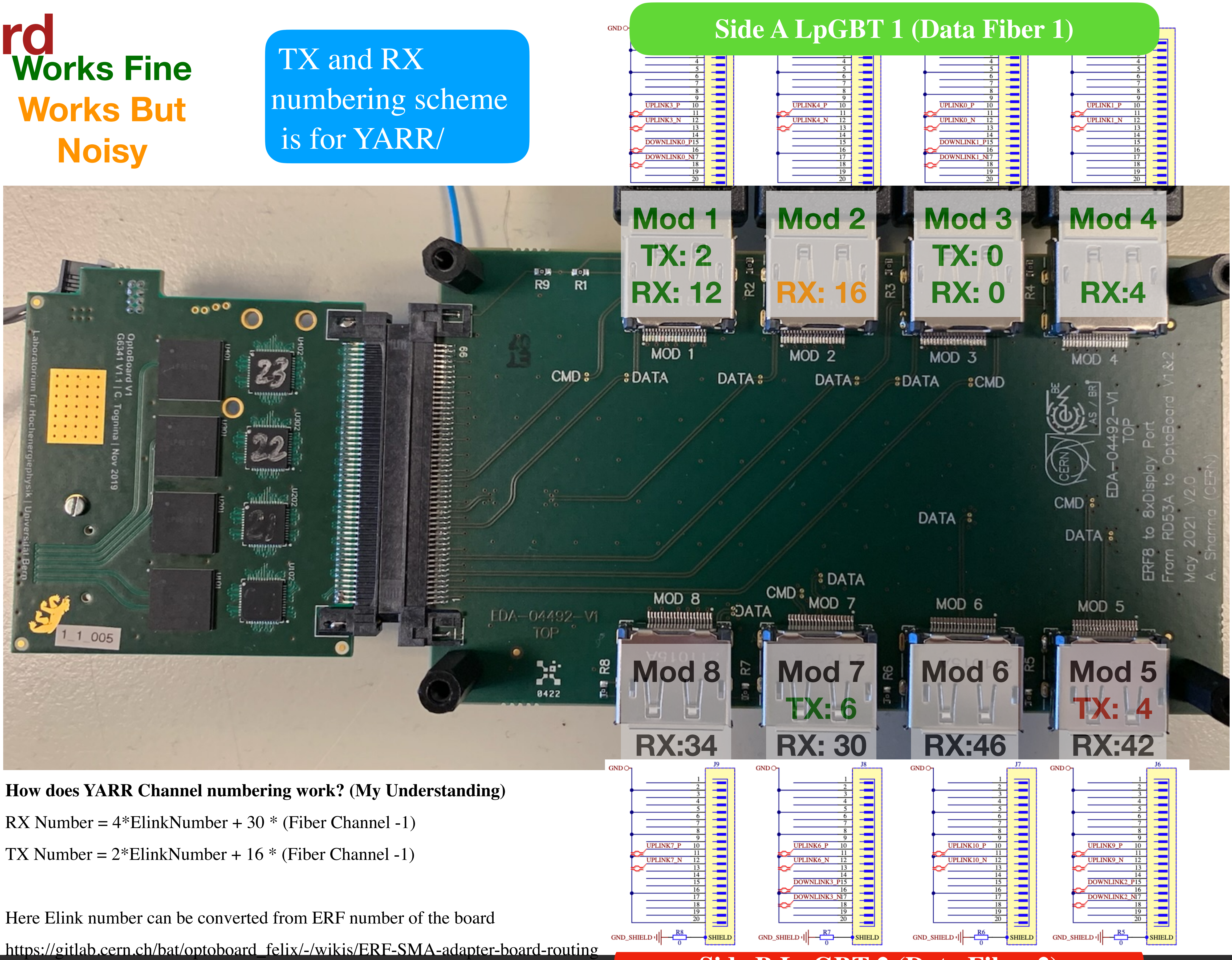
- Optoboard V1 and V2 consists (if fully populated) of nine active chips:
    - 4 lpGBTs (manual)
        - lpGBTs are numbered 1 to 4, lpGBT1 is the lpGBT master.
        - lpGBT1 is the I2C controller on the Optoboard and the only lpGBT in transceiver (TRX) mode. All other devices on the board are controlled through it. lpGBT1 receives the commands over IC (2.56 Gbps fibre) and distributes it to max. 8 CMD downlinks.
        - lpGBT2,3,4 are in transmitter (TX) mode and don't receive any commands from the fibres.
        - depending on the detector subsystem and layer the Optoboard has either 1, 2, 3 or 4 lpGBTs populated, lpGBT1 is always present.
        - lpGBT1 will be pre-configured (efused) for system tests with these values. lpGBT2,3,4 need to be configured.
        - Each LpGBT has its independent data fiber. So an opto-board with 4 LpGBT's would need 4 data fibers.
    - 4 GBCRs (Gigabit cable receivers) (manual)
        - GBCRs are needed for recovering the signal after twinax cables.
        - GBCRs can not be pre-configured, they need to be configured through the GUI or a script.
    - 1 VTRx+
        - two versions of the VLplus quad laser driver are in circulation (v1.2 and v1.3). The register maps differ between these versions, both are pre-configured to some extend but v1.3 has only 1 uplink (10 Gbps fibre) enabled by default.

Details About OptoBoard ERF to DP Board

Works Fine

Works But Noisy

TX and RX numbering scheme is for YARR/



How does YARR Channel numbering work? (My Understanding)

RX Number = 4*ElinkNumber + 30 * (Fiber Channel -1)

TX Number = 2*ElinkNumber + 16 * (Fiber Channel -1)

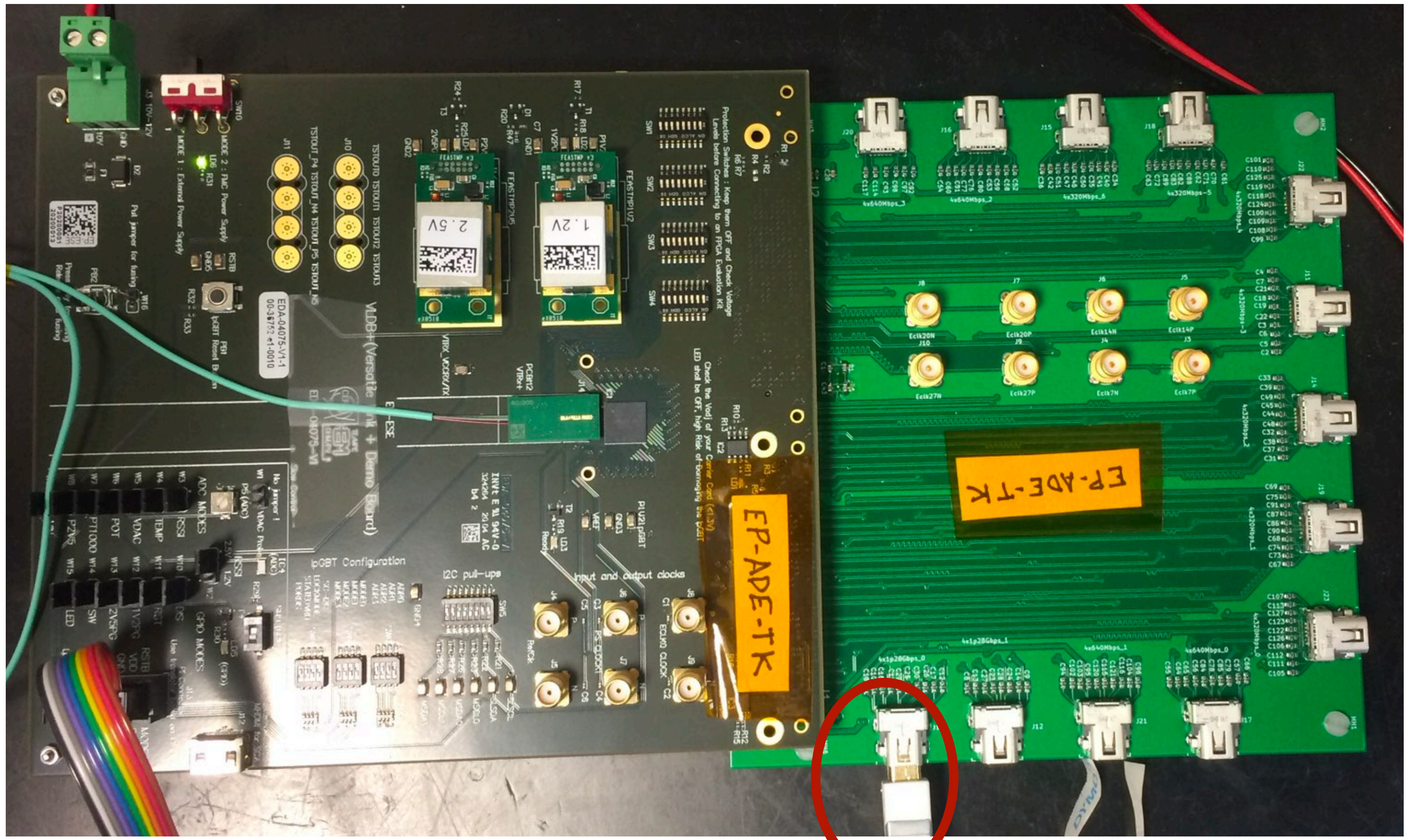
Here Elink number can be converted from ERF number of the board

https://gitlab.cern.ch/bat/optoboard_felix/-/wikis/ERF-SMA-adapter-board-routing

Details of VLDB+

- The VLDB+ has single LpGBT chip without a GBCR chip.

- So it means it has 1 CMD lane fiber and 1 Data fiber lane. This means on paper it can a quad but currently out-of the box the BoB of VLDB plus can only support 1 data and 1 cmd lane. (A hack is possible to improve this)

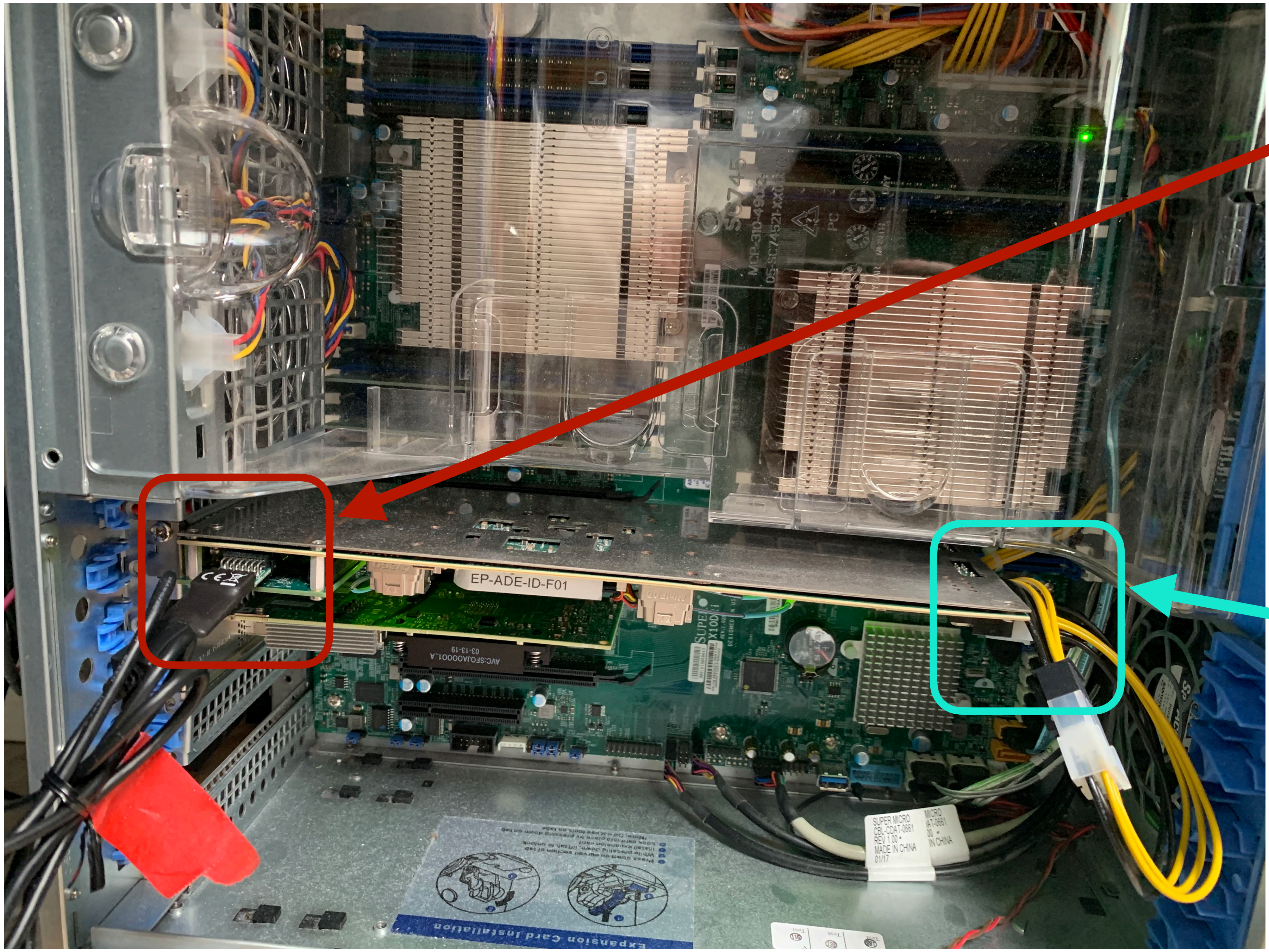


Only working channel of the BOB

Cabelling Details

Correct Cabelling Setup

FELIX JTAG Cable And Powering

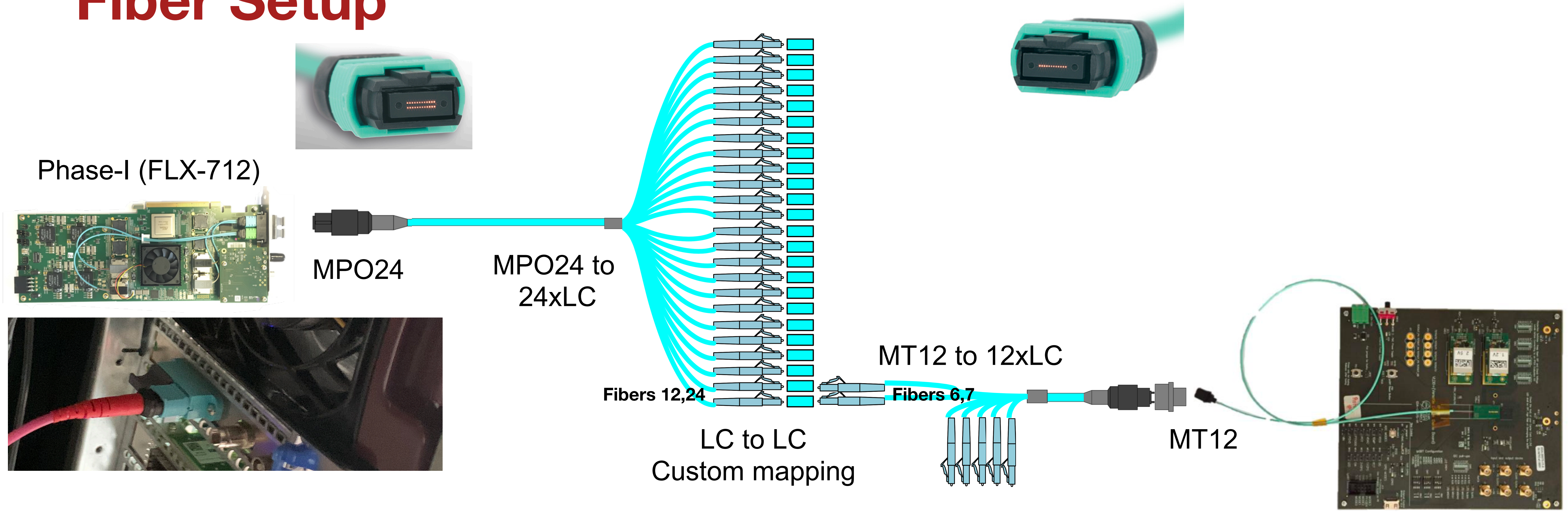


JTAG Cable Attached here to program felix

Felix Board powered through here (6pin cable)

Correct Cabelling Setup

Fiber Setup



- Currently FELIX uses a MPO24 connector. We connect to the port away from the motherboard. (Closer to the black petrusion) Then from the MPO24 we use only channels 12,24.

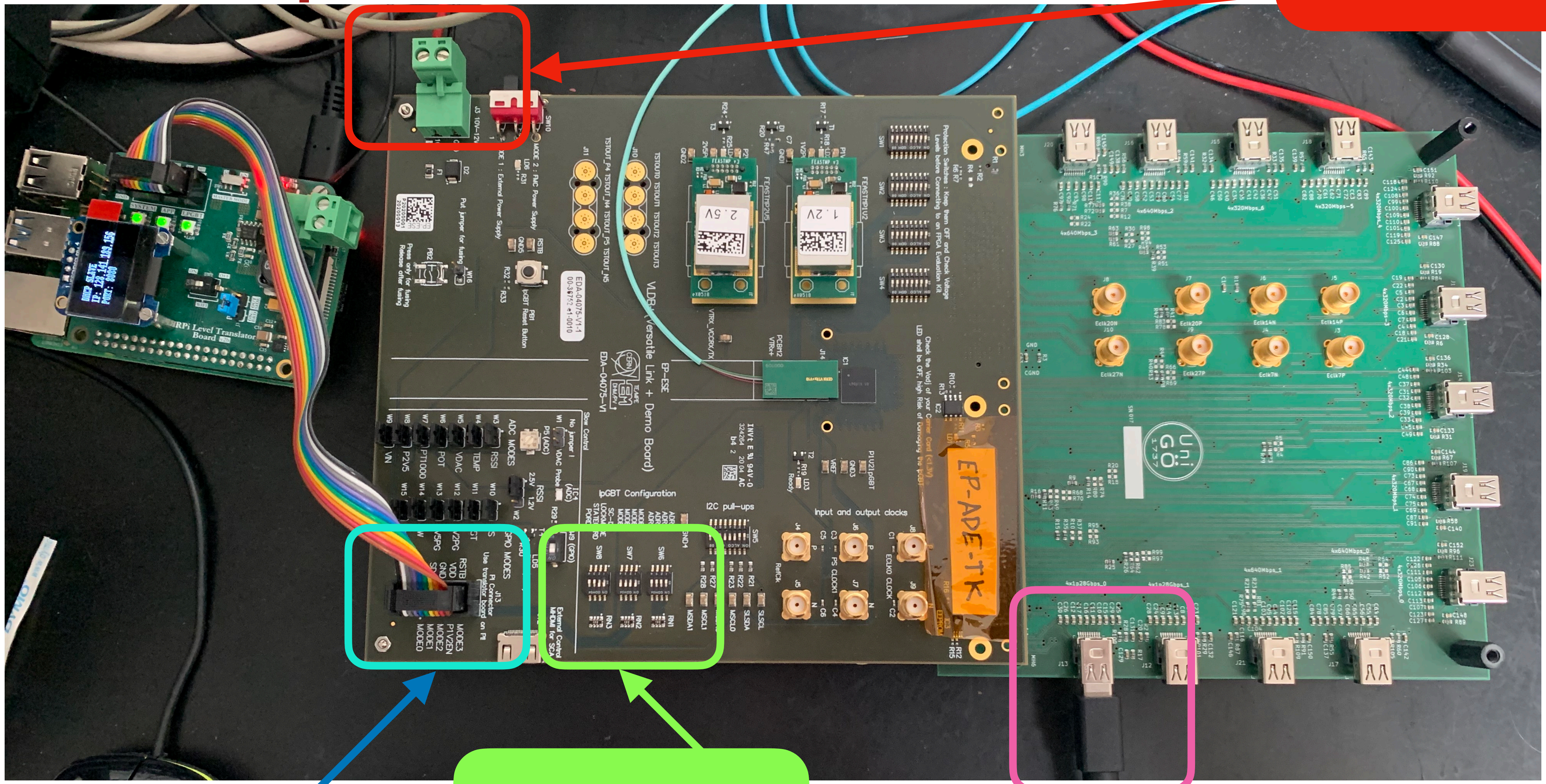
- The VLDB+/OptoBoard uses a MT12 connecter where only two channels are used. (6,7)

- The way we connect is we use channel 12/24 of comming from Felix 712 to channel 6,7 of the OptoBoard fibers. You need to match the dark fibers to the ones with lights. (Input/output)

Correct Cabelling Setup

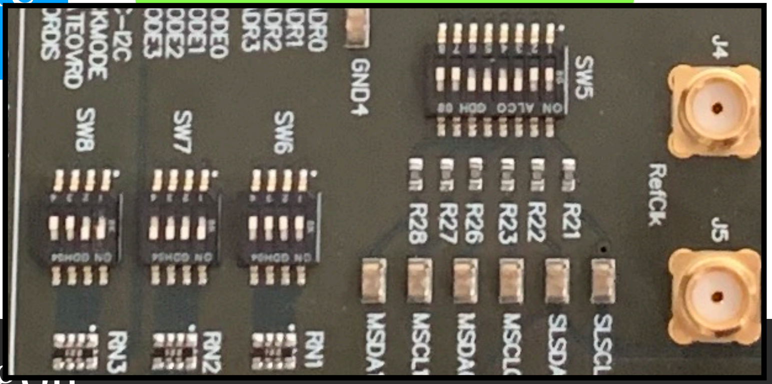
VLDB+ Setup

Should be powered here with: 10v 0.2A



Raspberry Pi (PiGBT) Connector connects here

The switches should be given as:



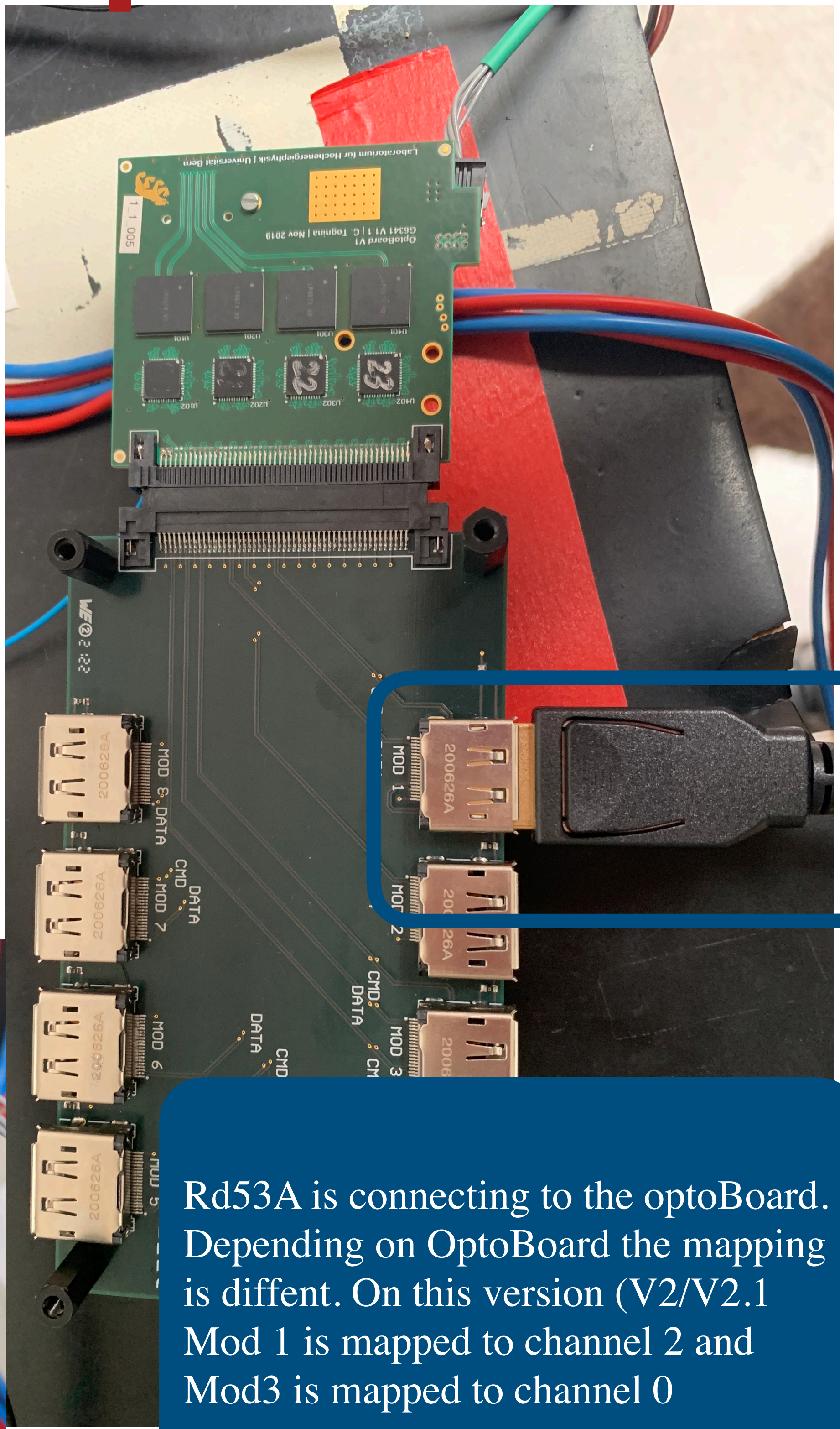
This data channel is used for connecting to RD53A

Correct Cabelling Setup

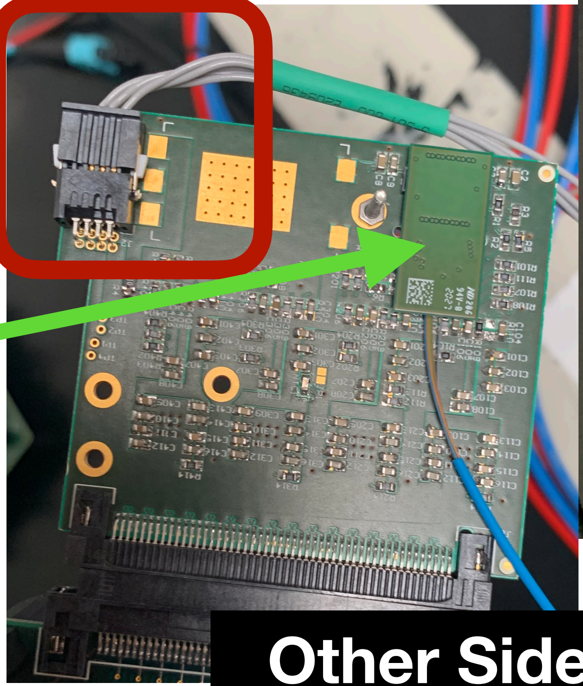
Opto Board Setup with BoB

OptoBoard needs to be powered in two channels:
Channel 1: 2.5V - 0.1 A
Channel 2: 1.2 - 2.5A

The cable should be labeled on which channel is connected at what voltage, don't assume the number is correct.



This is the Vrtx chip, connecting to the fiber bundle comming from the FELIX board.



Other Side of OptoBoard

Rd53A is connecting to the optoBoard. Depending on OptoBoard the mapping is diffent. On this version (V2/V2.1 Mod 1 is mapped to channel 2 and Mod3 is mapped to channel 0

More details on the right mapping is on the next slide

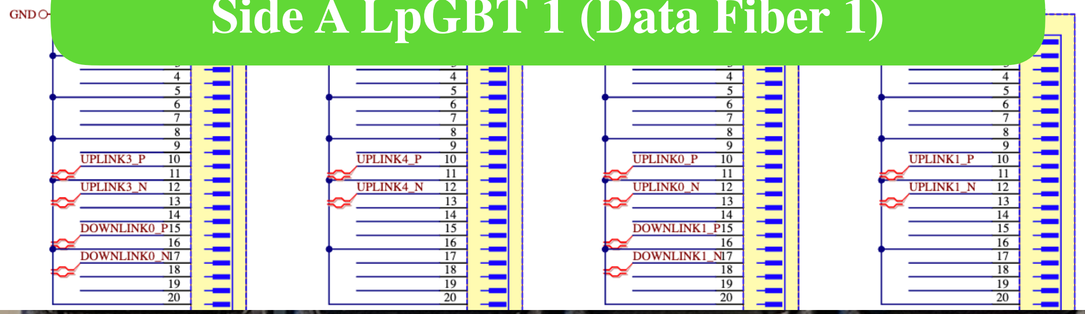
Details About OptoBoard ERF to DP Board

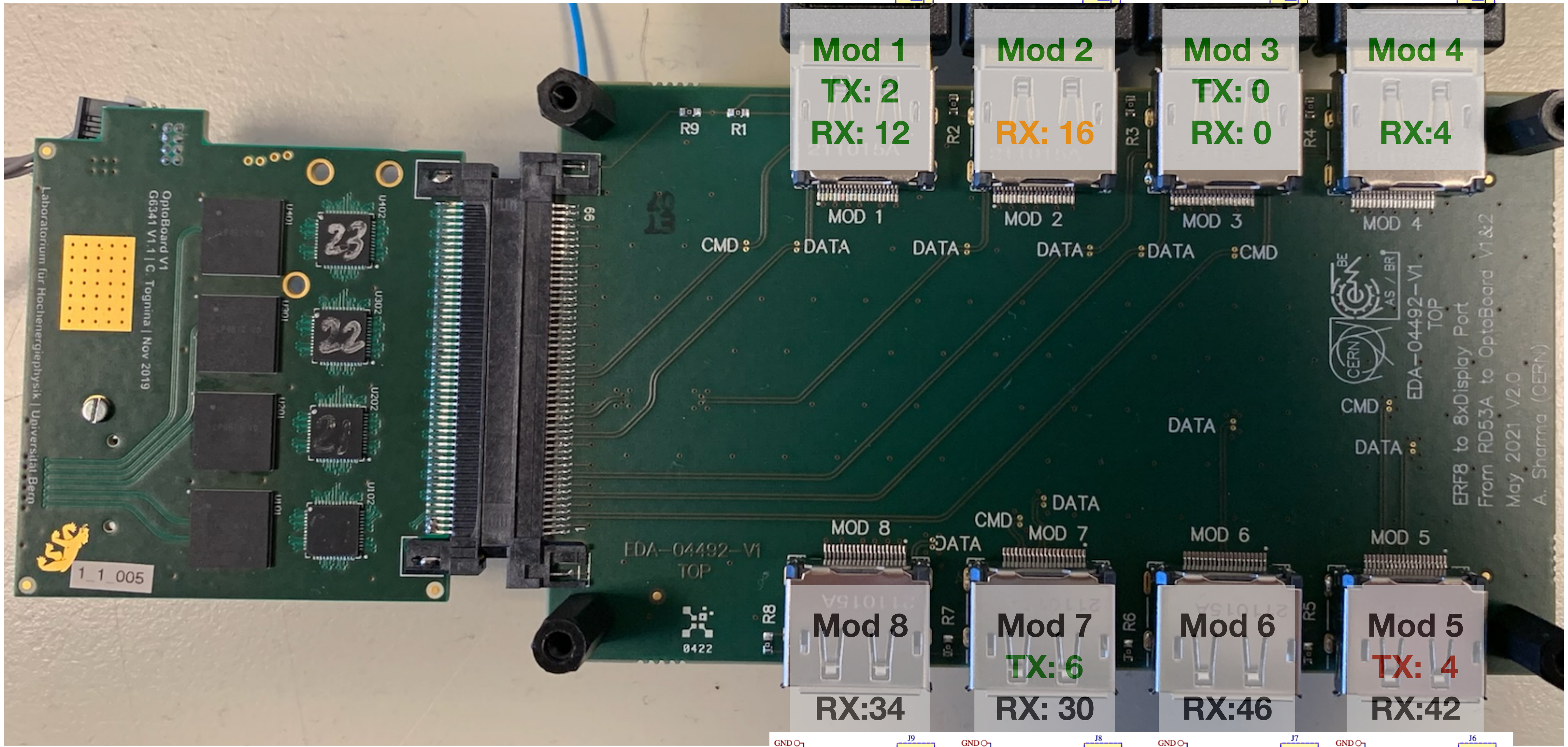
Works Fine

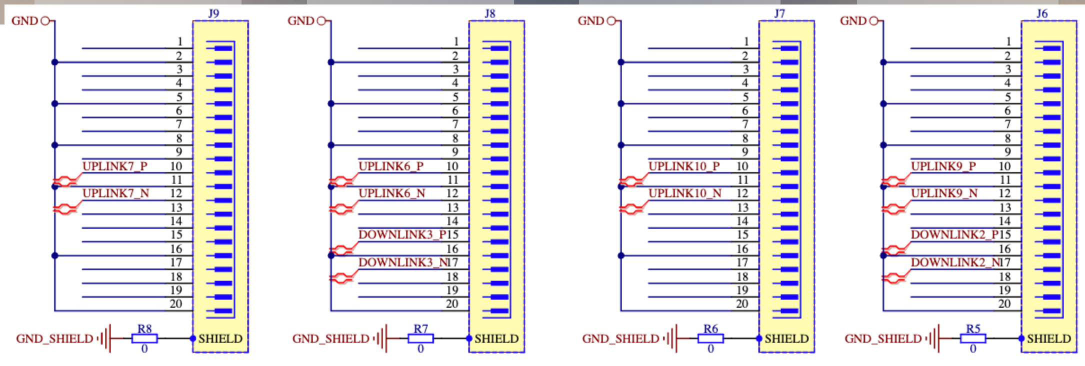
Works But Noisy

TX and RX numbering scheme is for YARR/

Side A LpGBT 1 (Data Fiber 1)




Mod 1
TX: 2
RX: 12
Mod 2
RX: 16
Mod 3
TX: 0
RX: 0
Mod 4
RX:4
Mod 8
RX:34
Mod 7
TX: 6
RX: 30
Mod 6
RX:46
Mod 5
TX: 4
RX:42
OptoBoard V1 G3341 V1.1 | C. Tognina | Nov 2019
Laboratorium für Hochenergiephysik | Universität Bern
EDA-04492-V1 TOP
ERF8 to 8xDisplay Port From RD53A to OptoBoard V1&2 May 2021 V2.0 A. Sharma (CERN)



Side B LpGBT 2 (Data Fiber 2)

How does YARR Channel numbering work? (My Understanding)

RX Number = 4*ElinkNumber + 30 * (Fiber Channel -1)

TX Number = 2*ElinkNumber + 16 * (Fiber Channel -1)

Here Elink number can be converted from ERF number of the board

https://gitlab.cern.ch/bat/optoboard_felix/-/wikis/ERF-SMA-adapter-board-routing

Software Installations/Setups

Setting up Software

- To use the RD53A you need different software:
  - Vivado to flash felix to firmware and the firmware itself
  - felix driver
  - Felix software package (To run FelixCore and communicate with Felix)
  - YARR (To run proper scans)
  - LpGBT Programer (Changes depending the LpGBT hardware)

Setting up Software

Installing Driver

- Drivers can be found here: https://atlas-project-felix.web.cern.ch/atlas-project-felix/user/dist/software/driver/

- We are currently using the following driver: tdaq_sw_for_Flx-4.7.0-2dkms.noarch.rpm

- You can instlal the driver following the instruction here: https://atlas-project-felix.web.cern.ch/atlas-project-felix/user/felix-user-manual/versions/4.0.6/5_software_installation.html#_5_2_1_driver_rpm_installation_instructions

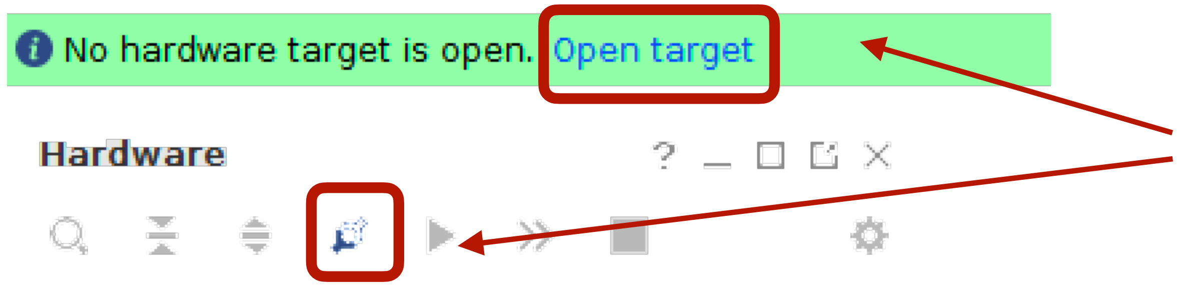
Setting up Software

Programming the FELIX

- To program felix we would need vivado or felix software, you need to install and license it from your institution.
- In addition you would need the firmware from us:
  - Latest firmwares can be found at: https://its.cern.ch/jira/browse/FLXUSERS-433 or https://cernbox.cern.ch/index.php/s/jUKu6ayuPVqD5qB?path=%2FBitFiles
  - We are currently using : FLX712_PIXEL_4CH_CLKSELECT_GIT_phase2-master_rm-5.0_2030_211118_20_23.
- To program Felix you need to run VIVADO
  - To start programing, click on the Open Hardware Manager in vivado, then follow instructions on the next slide.
- Our source felix-sw (Details on the later slides): (We only need one of the commands but I haven't double checked which one we need.)
  - fflashprog -f 2 ${FELIXFWDIR}/${FELIXFW} prog
  - fflashprog -f 3 ${FELIXFWDIR}/${FELIXFW} prog
  - sudo restart now
- At our CERN setups, you can find the fw's in ~/felix-fw directory

Setting up Software

Programming the FELIX with Vivado Visualised

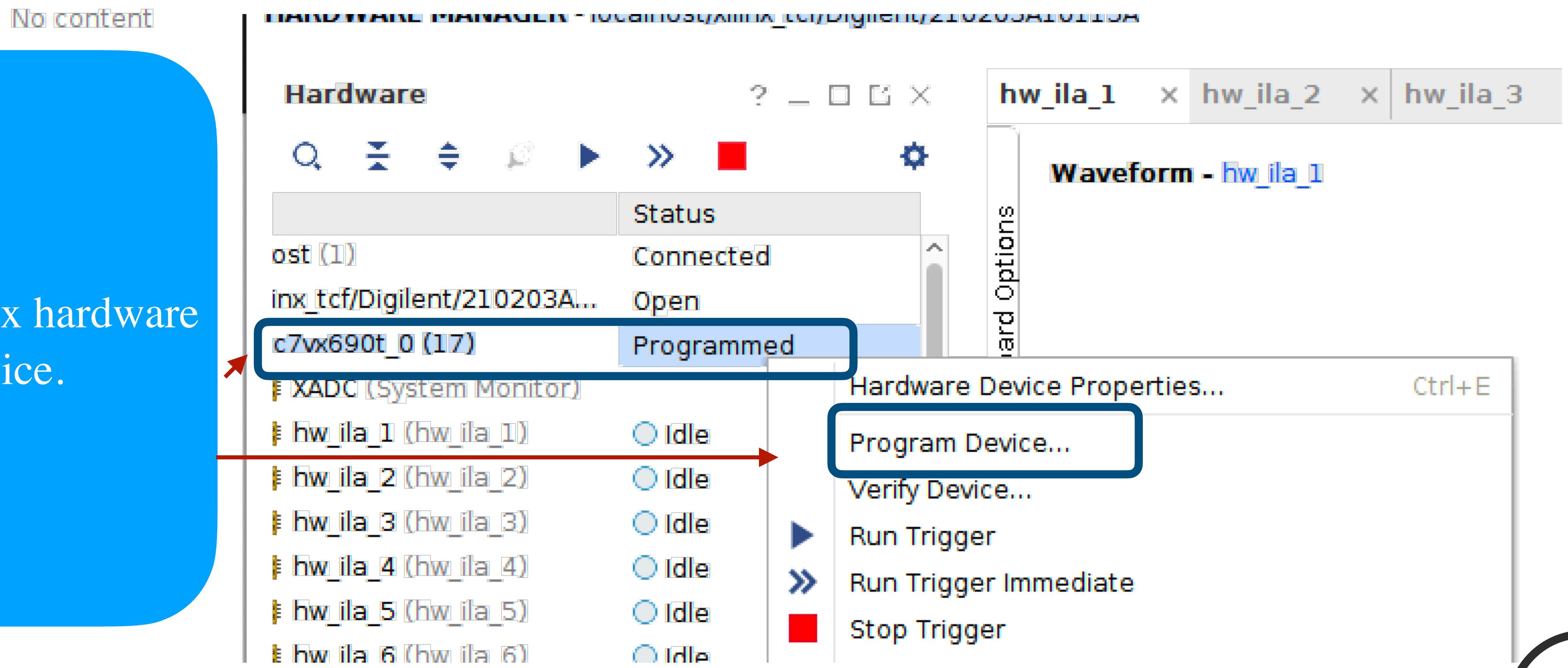


No hardware target is open. Open target

Hardware

Click one or the other to load Felix hardware

Right click on the Felix hardware and click program device.



No content

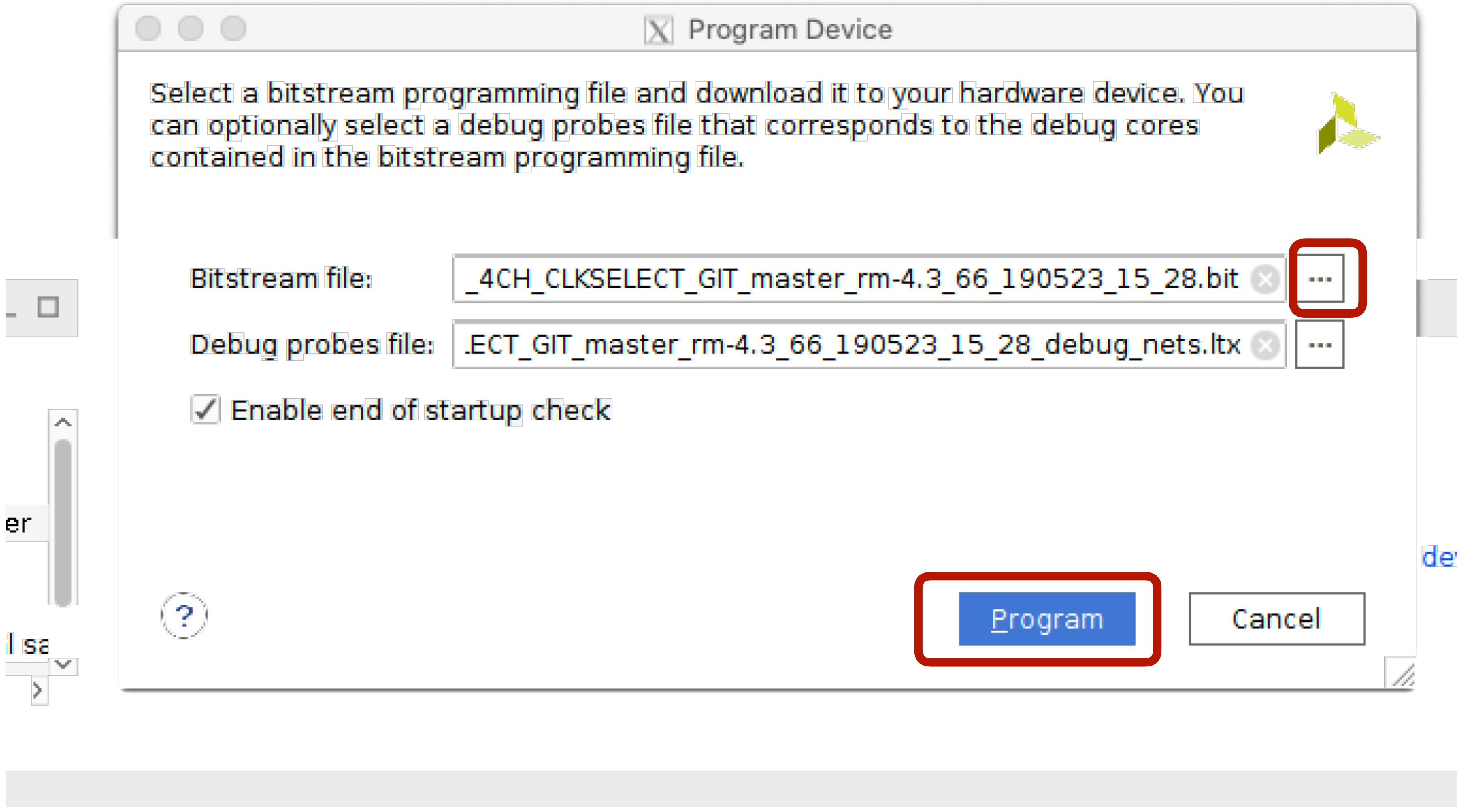
HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210203A0115A

Hardware

	Status
ost (1)	Connected
inx_tcf/Digilent/210203A...	Open
c7vx690t_0 (17)	Programmed
XADC (System Monitor)	
hw_ila_1 (hw_ila_1)	Idle
hw_ila_2 (hw_ila_2)	Idle
hw_ila_3 (hw_ila_3)	Idle
hw_ila_4 (hw_ila_4)	Idle
hw_ila_5 (hw_ila_5)	Idle
hw_ila_6 (hw_ila_6)	Idle

Setting up Software

Programming the Felix with Vivado



Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: _4CH_CLKSELECT_GIT_master_rm-4.3_66_190523_15_28.bit

Debug probes file: .ECT_GIT_master_rm-4.3_66_190523_15_28_debug_nets.ltx

☑ Enable end of startup check

Select the "…" and load the provided firmware, then program the Felix.

After programing the Felix, restart the PC. (The Felix shouldn't loose power in this processes)

Setting up Software

Installing Felixsoftware

- There is two ways of installing felixsoftware:
  - Compiling SF from scratch (My recommended method)
  - Using pre-compiled packaged.
  - The recommended version in CERN setups are:
  - 161: /home/itkpix/felix-5.0-FLX-1613/
  - SR1: /home/itkpix/felix-sw/felix-05-FLX-1613
- Using a compiled version:
  - Download from the website (Doesn't contain the FLX-1613 patch)
    - https://atlas-project-felix.web.cern.ch/atlas-project-felix/user/dist/software/latest/
    - You should download a reg 5.0 version which looks like: felix-05-00-xx-stand-alone-x86_64-centos7-gcc8-opt.tar.gz
  - un tar and start using (tar -xzf)

- Compiling from scratch: (Recommended)
  - Download the Felix software following the instructions at the git site.
    - https://:@gitlab.cern.ch:8443/atlas-tdaq-felix/software.git
    - Recommended Tag: Master
  - Use the documentation on the git site to clone all dependencies and compile the software with regmap 5.0 or follow the instructions below
    - cd software
    - ./clone_all.sh
    - ./checkout_all.sh <Recommended Tag> #### currently the tag we use is master
    - Minor patch to fix felixcore to get better timming. This won't be needed in the future is but recommended
      - cd felixcore
      - git checkout FLX-1613
    - export REGMAP_VERSION=0x0500 ###Most important step
    - source setup.sh
    - cmake_config x86_64-centos7-gcc8-opt

Felix Configuration

- To configure the Felix into a working mode, you would need to do following setups on each PC restart.
  - You would also need some extra scripts that can be found here (Generated by Marco): https://cernbox.cern.ch/index.php/s/jUKu6ayuPVqD5qB
- Commands
  - cd <FelixSoftwareDirectory>
  - export REGMAP_VERSION=0x0500 ### If you put this line in setup.sh, you wouldn't need to call this each time
  - source setup.sh
  - flx-init -c X
  - cd <directory from CERNBoxAbove>/scripts
  - source config_encoding_decoding.sh 0
  - source config_encoding_decoding.sh 1
- If you are running opto-board, you would need to run the following commands as well:
  - flx-config set GBT_RXPOLARITY=0xF
    - Here each bit stands for each fiber channel , you might need to change the value F on each polarity channel you want. If you cannot get link aligment it's probably because you didn't setup the polarity correctly.
  - flx-config set GBT_DATA_RXFORMAT2=0xFF #Changes to FEC12 Encoding
    - Here each bit in FF stands for different elink channels, you can change this for VLDB+ channels if you want to focus on VLDB+ only channels

Setting Up Software

Installing YARR

- To install YARR, we need to first checkout YARR:
  - git clone https://gitlab.cern.ch/YARR/YARR.git Yarr
- Then follow the following instructions to install YARR.
  - cd Yarr
  - Git checkout <Branch> # We are curretnly using the following branch: devel_rd53a_felixNetio_multichip_rebase
  - mkdir build
  - cd build
  - echo source scl_source enable devtoolset-7 > setup.sh
  - source setup.sh
  - cd build
  - cmake3 -DYARR_CONTROLLERS_TO_BUILD="Spec;Emu;NetioHW" ..
  - make install -j 4

Setting Up Software

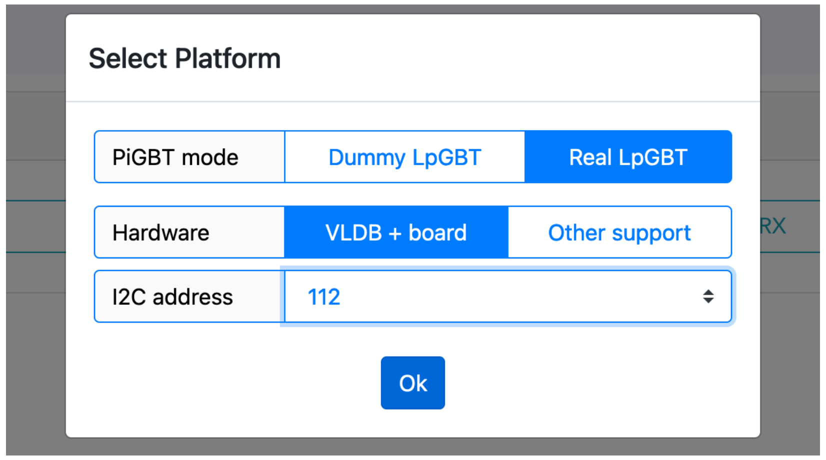
YARR Configuration

- You are now ready to use and run stuff.
- Before you run, there is one more important item.
- You need to edit the RD53A configuration card for:
  (Note, the configuration can be found at configure/rd53a_test.json. If not just make a new Felix run at it will be auto generated) (This might or might not be set depending on YARR version)
  - "OutputActiveLanes": 1 //Needed to readout the rd53a through a single fiber.
  - CdrSelSerClk: 0 //Needed to adjust the RD53A clock.
- Now you should be able to run Felix as normal.
- If you observe that you can run configure the RD53A but cannot read-back, please check the backup slides on debugging. Also it's recommended to check the official documentation given on slide 1.

Setting Up Software

Setting up configuration software for VLDB+

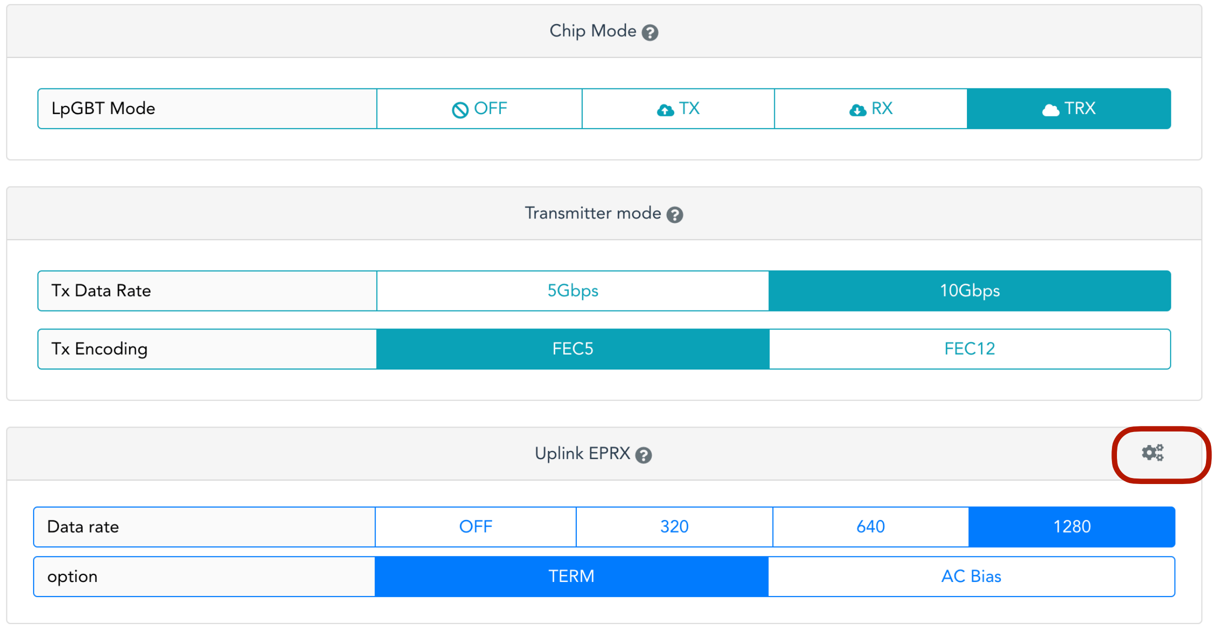
- Use the raspberry pi gui website:
- ep-ade-pigbt-01:8080
- Real VLDB, select VLDB+ , select IC and connect



Select Platform

PiGBT mode	Dummy LpGBT	[✓ Real LpGBT]
Hardware	[✓ VLDB + board]	Other support
I2C address	112	

Ok



Chip Mode

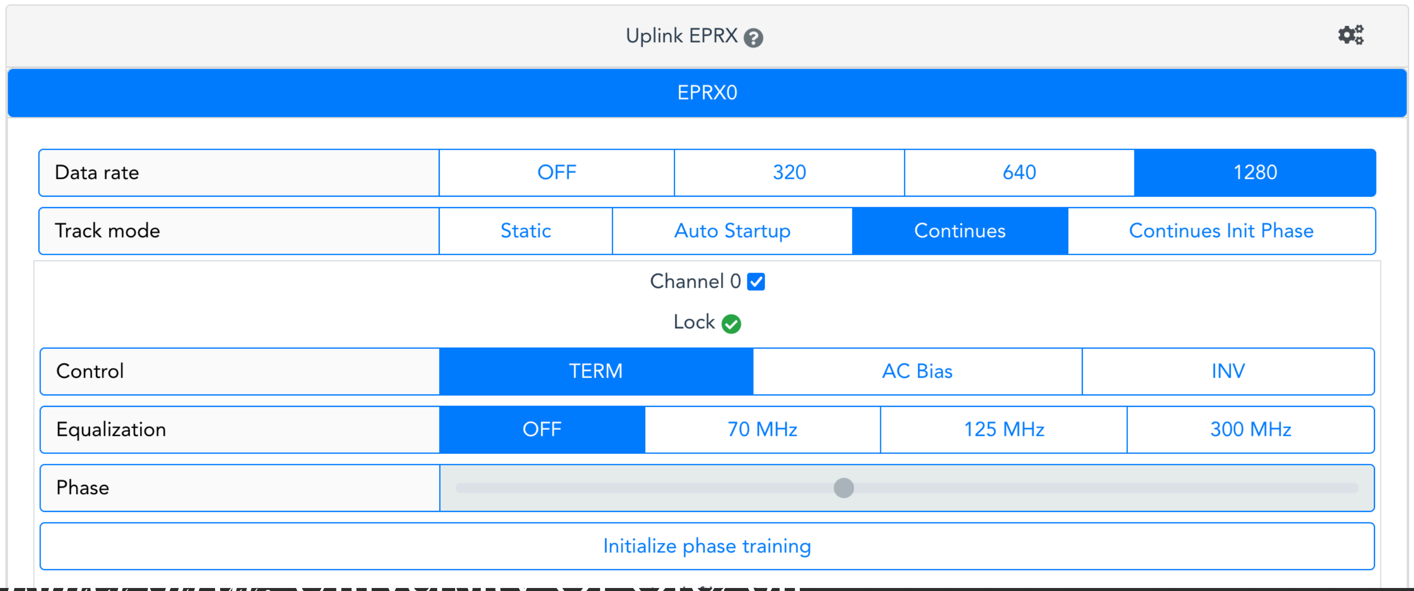
LpGBT Mode	OFF	TX	RX	[✓ TRX]

Transmitter mode

Tx Data Rate	5Gbps	[✓ 10Gbps]
Tx Encoding	[✓ FEC5]	FEC12

Uplink EPRX

Data rate	OFF	320	640	[✓ 1280]
option	[✓ TERM]	AC Bias		



Uplink EPRX

EPRX0

Data rate	OFF	320	640	[✓ 1280]	
Track mode	Static	Auto Startup	[✓ Continues]	Continues Init Phase	

Channel 0 ✓

Lock ✓

Control	[✓ TERM]	AC Bias	INV	
Equalization	[✓ OFF]	70 MHz	125 MHz	300 MHz
Phase				

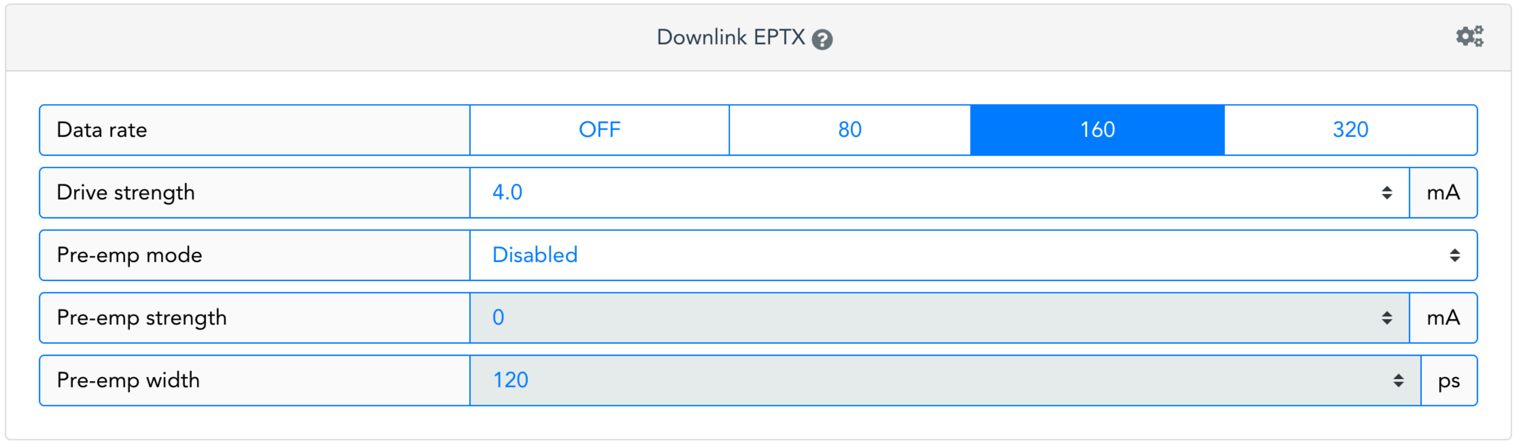
Initialize phase training

- TRX
  - TXdata rate = 10 Gbps, TX encoding = FEC5
- Uplink EPRX
  - EPRX0 (Click gear to excess EPRX0)
    - Data Rate = 1280
    - Track Mode = Continues
    - Control TERM
    - Equalization OFF
  - EPRX1-6 is off

Setting Up Software

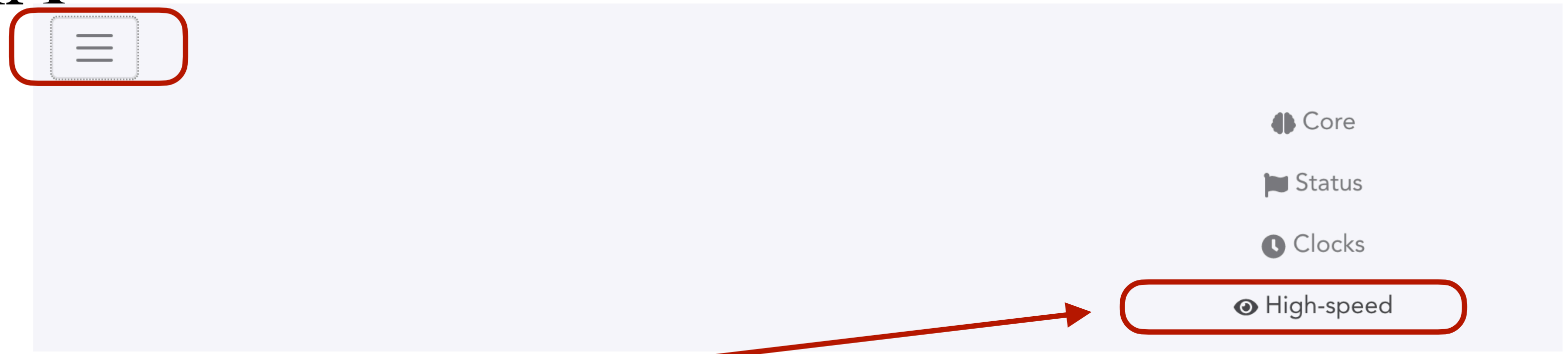
Setting up configuration software for VLDB+

- Uplink EPTX
  - Data Rate 160 Mpbs
  - Drive strength 4.0 mA

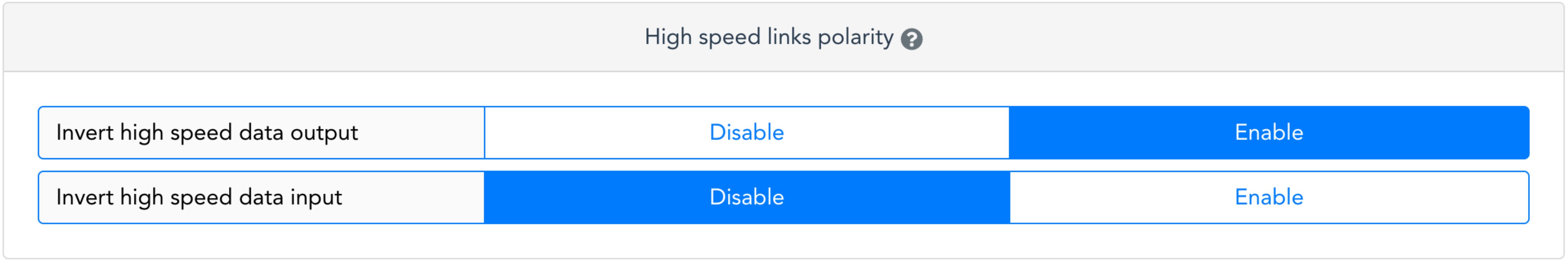


Downlink EPTX

Data rate	OFF	80	160	320
Drive strength	4.0			mA
Pre-emp mode	Disabled			
Pre-emp strength	0			mA
Pre-emp width	120			ps



- Click on the menu -> high speed
  - Invert high speed data output enabled (Invert polarity uplinks)



High speed links polarity

Invert high speed data output	Disable	Enable
Invert high speed data input	Disable	Enable

Setting Up Software

Setting up configuration software for OptoBoard

- for installation, clone the repository:
  http://gitlab.cern.ch/bat/optoboard_felix
- the software is developed by having a config file (JSON) with the register data for every script or the GUI. With this solution scripts don't need adjustment, one only changes the register values in the configs and the path to it.
- each config file has parameters to change in the topmost "Optoboard" dictionary. Please set as a minimum the serial (without spaces) written on the Optoboard and the version. From these the software accesses automatically the addresses and configuration of the Optoboard via the components.py file.
- python minimal_v1_v2.py is a good starting point for doing first tests. The script configures every chip according to minimal_v1_v2.json, which enables all possible up- and downlinks as well as the vital GBCRs.
- GUI: for quick debugging an Optoboard GUI has been developed. To run it use python optoboard_GUI.py in the GUI folder. An explorer window should open up where you can select a config. This config only initialises the GUI and doesn't send anything to the Optoboard. Use the various red send buttons to do a quick configuration of multiple registers.

- Latest recommended method is to use the Slim Gui:
  - git checkout slim_config
  - python quick_start.py -v 1.3 -s 1_1_005 -G 0 -d 0 -i 0
    - -s is the chip ID written on the optoBoard. -v is the VRTX chip number. At SR1 this is 1.2 at 161 this is 1.3
    - Fiber 0, -G 0 -i 0 -d 0 ; Fiber 1 -G 1 -i 1 -d 0 ; Fiber 1 -G 0 -i 0 -d 1 ; Fiber 1 -G 1 -i 1 -d 1

Running Scans

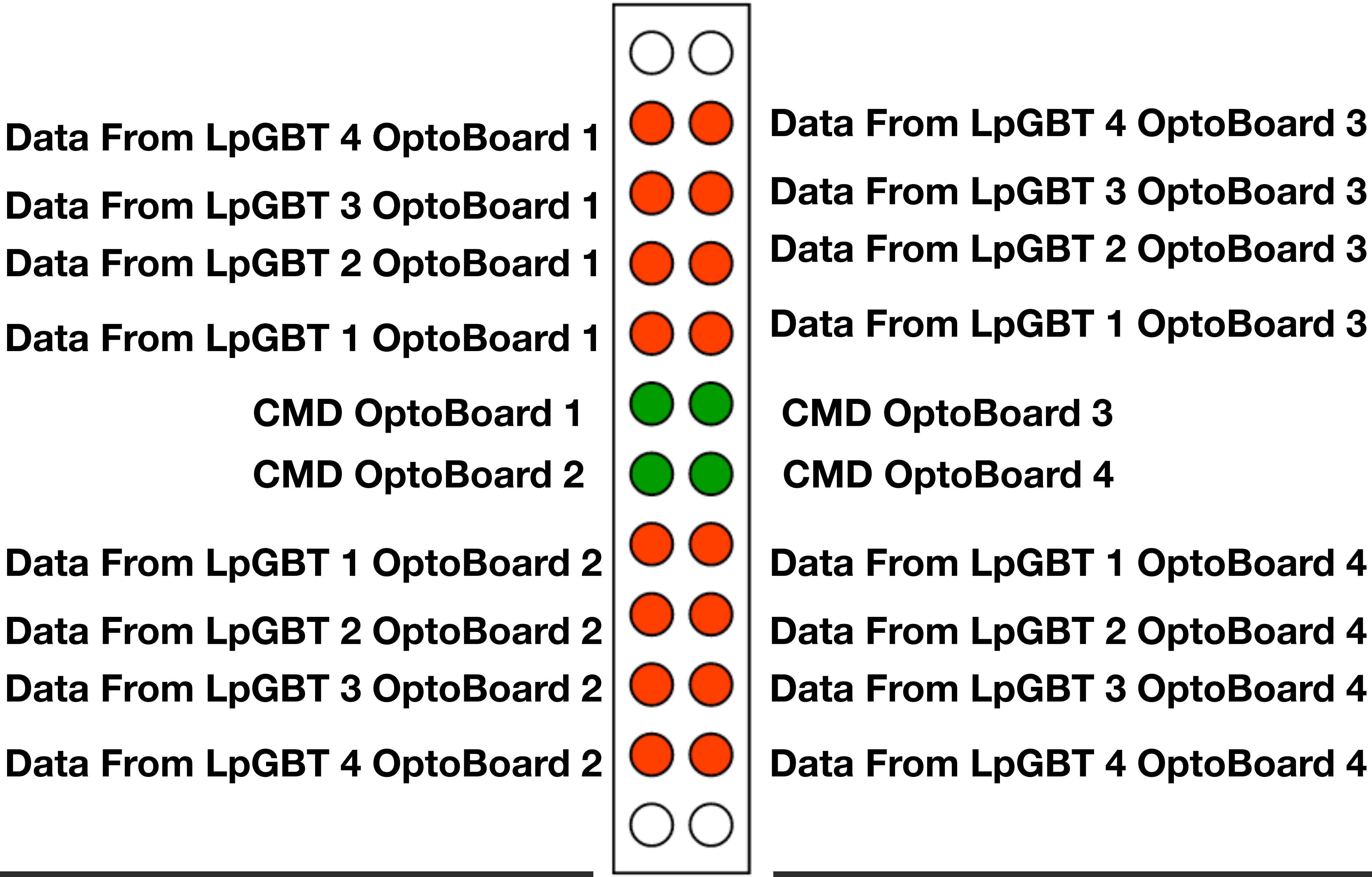
Running a Basic YARR Scan

- To make a scan with YARR, you would need to first run FelixCore, this program needs to be constantly running on the background.
  - cd <felix software directory>
  - export REGMAP_VERSION=0x0500 ###Most important step
  - source setup.sh
  - x86_64-centos7-gcc8-opt/felixcore/felixcore -d 0 --data-interface lo --elinks 0,4,8,12   —toflx-tlp 64
    - The command in red is only to be added if you have used the FLX-1613 patch when setting up felix-sw
    - You may need to edit the elink numbers depending on your needs.
- In alternative terminal, you need to run YARR.
  - cd <Yarr Directory>
  - source setup.sh
  - build/bin/scanConsole -c configs/connectivity/example_rd53a_setup.json -r configs/controller/felix.json -s configs/scans/rd53a/std_digitalscan.json
- When you run YARR, you give it 3 different json files, you would probably need to edit these files in the future depending on your software needs.
  - Like configs/connectivity/example_rd53a_setup.json, needs to be configured according to your fiber/e-link channel settings.
- Also before running YARR please don't forget to edit the register files as mentioned in "YARR Configuration" setup.

SR1 Specific Items

Fanout in SR1 Rack Area First Fanout

- At SR1, we have 7 optoBoards right now. So we will uses two fan outs with one with 4 optoBoards and 3 with the other.
- Each optoBoard only has 1 or 2 LpGBTs. So only LpGBT 1 and 3 is used. The exact LpGBT info per optoBoard can be found here: https://twiki.cern.ch/twiki/bin/view/Atlas/SR1DemonstratorOptoBox


Data From LpGBT 4 OptoBoard 1
Data From LpGBT 3 OptoBoard 1
Data From LpGBT 2 OptoBoard 1
Data From LpGBT 1 OptoBoard 1
CMD OptoBoard 1
CMD OptoBoard 2
Data From LpGBT 1 OptoBoard 2
Data From LpGBT 2 OptoBoard 2
Data From LpGBT 3 OptoBoard 2
Data From LpGBT 4 OptoBoard 2
Data From LpGBT 4 OptoBoard 3
Data From LpGBT 3 OptoBoard 3
Data From LpGBT 2 OptoBoard 3
Data From LpGBT 1 OptoBoard 3
CMD OptoBoard 3
CMD OptoBoard 4
Data From LpGBT 1 OptoBoard 4
Data From LpGBT 2 OptoBoard 4
Data From LpGBT 3 OptoBoard 4
Data From LpGBT 4 OptoBoard 4

Subtitle Text

Trouble Shooting Ideas

Felix Troubleshooting

- The trouble shooting is limited, but the first thing to do would be to run in Felix-software:
  - flx-info #Get the general Felix info
  - flx-info links #shows if the links are aligned, if not aligned either you LpGBT configuration or Felix configuration is problematic
  - fice -G 0 -i 0 -d 0 -I 70 -a 0x1c5 # Checks fice/programing communcation of optoBoard by trying to read registers. -I setting needs to match IC port of optoBoard, -G and -i - d needs to match the right felix fiber.
    - Fiber 0, -G 0 -i 0 -d 0 ; Fiber 1 -G 1 -i 1 -d 0 ; Fiber 1 -G 0 -i 0 -d 1 ; Fiber 1 -G 1 -i 1 -d 1
  - flx-config list | grep DECODING_LINK_ALIGNED # Shows individual rd53a links are aligned or not. More details in the Argone manual.