Pixel Phase 1 DAQ Documentation

Contents

Chapter 1

Phase 1 DAQ Test Stand

This chapter gives a brief summary about all the components neccessary to setup a test stand for the Pixel Phase 1 data acquisition system (DAQ). It is shown how all components are connected. The specifications of the shown parts and how to run them can be found in their respective chapter.

1.1 Minimal set of components needed for the Phase 1 DAQ Test Stand

For the DAQ system

- μTCA Crate – hosts all the μTCA cards, distributes signals over back plane
- MCH – MicroTCA Carrier Hub, manages all cards in crate
- AMC13 – interface to CMS TCDS (trigger & clock distribution system)
- Optical clock source – small card that creates a 40.08MHz clock
- FC7/CTA – CMS Tracker AMC, base board for most DAQ components
  - tkFEC – Tracker Front End Controller, sends control signals to the CCU board
  - pixFEC – Pixel Front End Controller, sends control signals to the detector modules
  - FED – Front End Driver, receives signals from the detector modules

For the test bench

- PortCard – equipped with POHs (Pixel Opto-Hybrids), conversion from electrical to optical signals
- DOH mother board – Digital Opto-Hybrids, conversion from optical to electrical signals
- CCU board – Communication & Control Unit
- Pixel Module
- Fibre adaptors
- Optical fibres
- A diverse set of power supplies and maybe an oszilloscop

1.1.1 Useful tools for debugging and setting up

- uSD card reader for JTAG-free CTA FW management
- Xilinx USB cable for debugging (FW upload)
- Lab PC with 2 Ethernet ports (SLC6)
- 1Ghz Oscilloscope
- opto-electrical converter for debugging of CLK lines
- differential probe, single ended probes for debug FMC on FED

1.1.2 Software tools

- Wireshark for spying on TCP traffic & determining MAC addresses (via yum)
- ic_mmc tools (imgtool for FW upload to SD card) (see FC7 user guide)
- AMC13 development tools (to set RARP mode on AMC13)
- natVIEW for crate-backplane configuration
- IPMItool (via yum)
- FED CMD line tools (`https://github.com/gauzinge/FED_burnin/tree/FIFO`)
- modified FEC SW including modified BPixelTools (available on FEC SW SVN)

1.2 Setting up infrastructure

1.2.1 Getting started

- uTCA Shelf, MCH & Power module should be plug & play
- MCH has a default IP address that can be changed via web interface (see NAT documentation)
  - simplest solution is to use a static IP & subnet mask
  - could also use DHCP server running on local lab PC for IP assignment (similar to P5 but maybe overkill)
- Lab PC needs to have 2 ethernet ports (1 for global network, one for local uTCA LAN)
  - configure the local LAN port to have static IP & subnet same as MCH
  - install uHAL (`cactus.web.cern.ch`)
  - install RARPD (daemon) for AMC13 / CTA IP assignment
- verify that you can ping the MCH (a USB format serial cable delivered with MCH can be used to connect via miniCOM terminal application in case ethernet does not behave at all)

1.2.2 Preparing FC7 cards

Before pluging the card in

- FC7 manual, although not complete, provides enough pointers to get started: `https://svnweb.cern.ch/cern/wsvn/ph-ese/be/fc7/trunk/fc7/doc/`
- 2 configurations to be done on the board:
  - set the powering switch to crate operation
  - push the switches 1 & 5 of the CPLD DIP switch down (includes FGPA in JTAG chain, sets FW loading from SD card after power on)
- carefully insert card in crate - attention, the switch that is actuated by the hot-swap handle is rather fragile
- if the card is properly seated in the crate, the blue LED at the bottom should light up and the corresponding LED on the MCH should blink
- insert the hot-swap handle - the card should power up & load the FW image from SD card (see 1.2.2)

Once the card is plugged in

- use an external card reader to format the SD card following the instructions from the FC7 manual p13
- upload an FC7 GoldenImage and / or a pixel specific image following the instructions
- caveat 1: one image must always be called “GoldenImage.bin” - this is what gets loaded on power-up
- caveat 2: FW files *must* be .bin format

Network configuration with RARP

- all pixel-specific FW uses RARP mode for the CTA
- MAC address is easily found out using wireshark
- requires edit of /etc/ethers & /etc/hosts & 'sudo /etc/init.d/rarpd restart'
- arp -e command is your friend
- see `https://twiki.cern.ch/twiki/bin/viewauth/CMS/Ph1PixelDAQTest`

Chapter 2

AMC13

2.1 AMC13 prerequisites

- TTC links from AMC13 to AMCs needs to be enabled explicitly on per-slot basis
- AMC13 needs a LHC-like input (clock) signal for QPLL to lock:
  - internal 40MHz oscillator does not work
  - either TTC data-stream from TTCci (80.16 MHz bi-phase encoded)
- external (BU) clock board (80.16 MHz clock)
- these need to be provided via the TTC SFP (Rx)
- AMC13 TTC simulator mode will not work for pixels

2.2 Setting everything up for the test stand

2.2.1 TTC Configuration for test stand

- we usually use the following TTCci config for our uTCA system:
  - BC0 every orbit
  - BGO Ch15: 0x2c - charge inject @ BX 101, repetetive, optional prescale
  - cyclic L1A: BX 199 (98 BX after the charge inject), repetitive, optional prescale

2.2.2 AMC13 Configuration

- AMC13 offers 4 BGO channels (BC0 is sent automatically every orbit)
  - can configure any command and send single or cyclic BGOs (orbit prescale is possible)
  - this is rather flexible
  - EC0 and OC0 can be user enabled
- L1A is available in 3 modes:
  - 0: periodic triggers spaced by rate orbits at BX=500
  - 1: periodic triggers spaced by rate BX
  - 2: random trigger at 2 * rate Hz
- these are configured with rate and rules (`amc13.info`) attributes
- this config can be used to either send burst triggers (also single) or continuously

2.2.3 Pixel Test Stand Configuration for standalone AMC13

- restrictions on BGO & L1A config leave 1 possible config for the AMC13
- BGO Ch0: cmd=0x2c, repeat=1, prescale=0, BX=380 (sent 20 BX later than that, makes BX 400)
- L1A: mode=0, rate=1 burst=1 rules=0
  - this equals to: 1 L1A every orbit @ BX 500, 1 BGO (0x2c) every orbit at BX 401
  - 20 BX delay for the BGO have to be considered
- L1A is rescaled via 'rate' attribute, BGO via 'prescale' attribute

Chapter 3

Front End Driver (FED)

3.1 FED configuration

- 1 CTA + 1 FITEL Rx FMC (in upper slot for current PixFED beta FW - in lower slot in the future)
- Xilinx FMC105 supported on the other slot for debugging (`https://indico.cern.ch/event/482128/contribution/0/attachments/1211056/1766370/Phase_1_update_13_01_2016.pdf` for Pin mapping)
- current FW is rather basic (phase finding, FE DESER block, basic BE readout functionality)
  - single-event readout mode / TTS / C-DAQ implementation in progress
  - photocurrent measurement via ADC on FMC implemented
  - OSD read back functionality implemented
- integration in POS ongoing - standalone CMD line tool available: `https://github.com/gauzinge/FED_burnin/tree/FIFO` - gcc ¿ 4.7 required!!
- caveat: uTCA FED needs male MPO on input fibre (different from digital VME FED)

3.2 FED performance measurements

Here we could collect some results and 'How-to's for the performance measurements for the FEDs

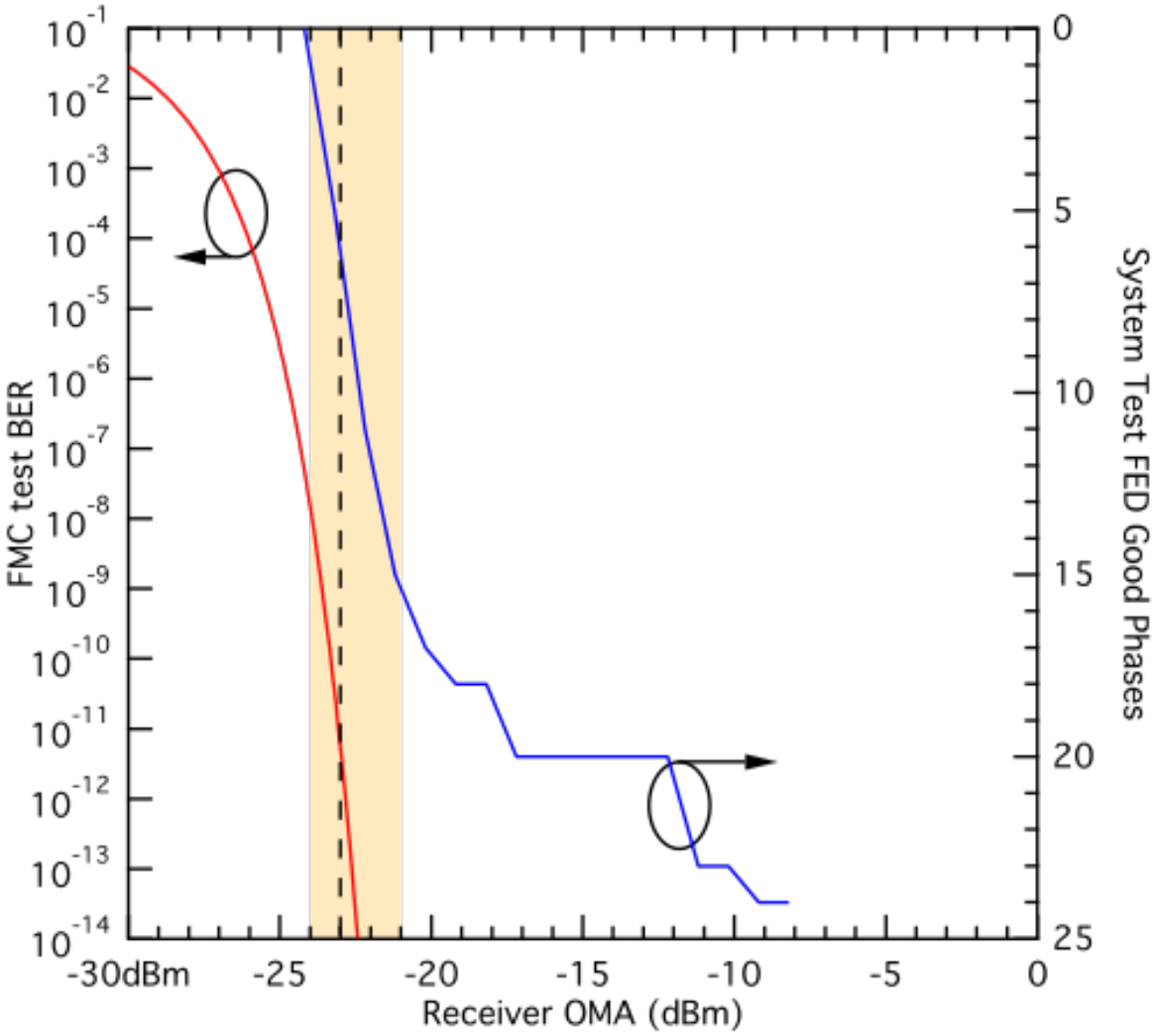

Figure 3.1: **Number of good phases vs optical signal attenuation**

Chapter 4

Pixel Front End Controler (pixFEC)

4.1 FEC configuration

- TkFEC: modified version of FEC SW available on SVN (make Generic, make APIConsoleDebugger for CMD line operation)
  - `https://svnweb.cern.ch/cern/wsvn/cmsptdaqup/CMS_UPGRADES_IPHC/SOFTWARE/PHASE_1/CTRL/tags/TK_FEC_FMC8SFP_4CTRLRING_V01/?#a9e50ea3b3bff0b870e7f5f38e685e2a`
  - `https://twiki.cern.ch/twiki/bin/viewauth/CMS/AddingAnUtcaRingDevice`
- PxFEC: SW adaptation in progress (no "release" yet)
- Ring 0 / Link 0 is on the bottom of bottom FMC
- Fibre mapping for 1 FMC can be seen in 5.1

SFP left					SFP right	Ring pxFEC	Ring tk FEC
H (Clock)	Rx			Tx	G (Data)	Ring 2 B	Link 4
	Tx			Rx			
F (Clock)	Rx			Tx	E (Data)	Ring 2 A	Link 3
	Tx			Rx			
D (Clock)	Rx			Tx	C (Data)	Ring 1 B	Link 2
	Tx			Rx			
B (Clock)	Rx			Tx	A (Data)	Ring 1 A	Link 1
	Tx			Rx			

Figure 4.1: **FEC fibre mapping**

Chapter 5

Tracker Front End Controler (tkFEC)

5.1 FEC configuration

- TkFEC: modified version of FEC SW available on SVN (make Generic, make APIConsoleDebugger for CMD line operation)
  - `https://svnweb.cern.ch/cern/wsvn/cmsptdaqup/CMS_UPGRADES_IPHC/SOFTWARE/PHASE_1/CTRL/tags/TK_FEC_FMC8SFP_4CTRLRING_V01/?#a9e50ea3b3bff0b870e7f5f38e685e2a`
  - `https://twiki.cern.ch/twiki/bin/viewauth/CMS/AddingAnUtcaRingDevice`
- PxFEC: SW adaptation in progress (no "release" yet)
- Ring 0 / Link 0 is on the bottom of bottom FMC
- Fibre mapping for 1 FMC can be seen in 5.1

SFP left					SFP right	Ring pxFEC	Ring tk FEC
H (Clock)	Rx			Tx	G (Data)	Ring 2 B	Link 4
	Tx			Rx			
F (Clock)	Rx			Tx	E (Data)	Ring 2 A	Link 3
	Tx			Rx			
D (Clock)	Rx			Tx	C (Data)	Ring 1 B	Link 2
	Tx			Rx			
B (Clock)	Rx			Tx	A (Data)	Ring 1 A	Link 1
	Tx			Rx			

Figure 5.1: **FEC fibre mapping**

List of Figures

List of Tables