Status Update on the CRU development/testing

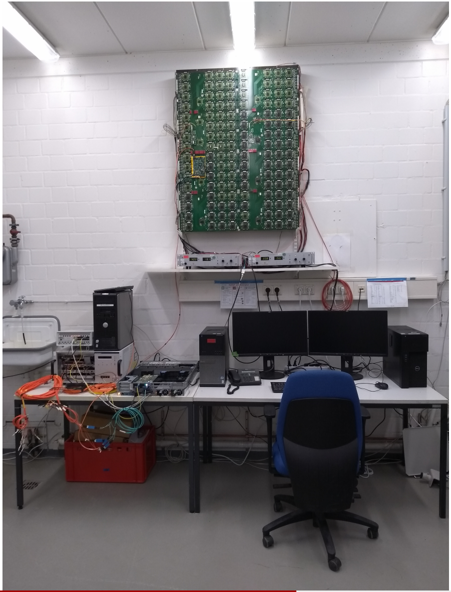
Guido Willems

10/12/2018

Preparatory work

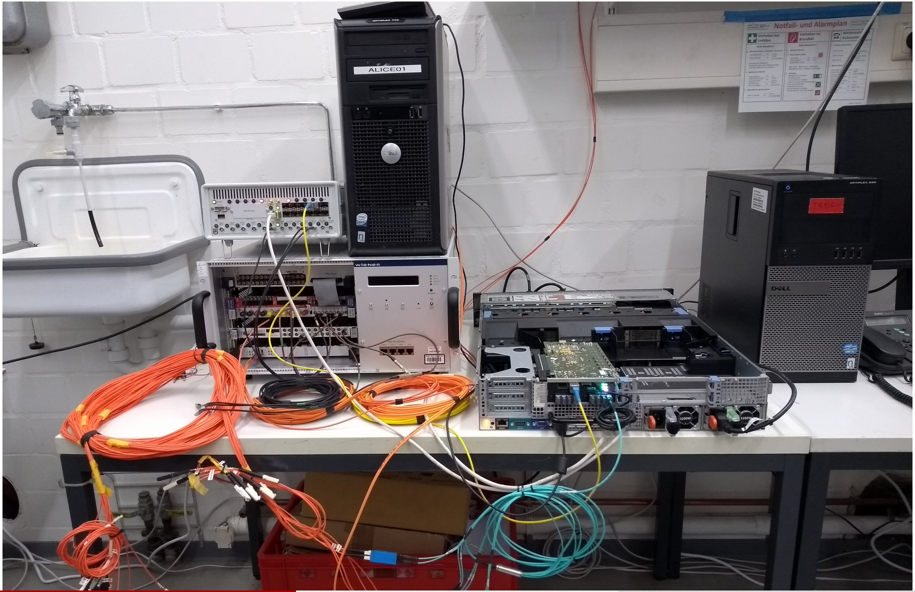
- Setup of a reverse proxy server on the laboratory network to gain access to the repositories hosted at physi.uni-heidelberg.de without SSL encryption
- Update of DCS board firmware to the versions currently used at P2
- Setup of the FLP with CRU
- FLP is on loan, but it can be kept
- Basic setup of lab LTU: All required fibers/cables are in place, software set up, OLT can sync with CRU ONU and old TTCex system is connected.
- No triggering of lab chamber possible yet (wait implementation from CTP)
- Old Münster cosmics trigger setup broken down to the essential functionalities of providing simple triggers (no photomultiplier input anymore, but sending of dedicated triggers possible through command line tools)

CRU lab test setup



- TRD CRU test setup consisting of:
- 1 detector chamber
- 2 ORI baords / 1 DCS board
- First Level Processor (FLP)
- Common Readout Unit (CRU)
- Run 2 trigger setup (TTCvi, TTCex...)
- New Local Trigger Unit (LTU) for Run 3
- Various control machines

CRU lab test setup



Preparatory work: Setup of the TRD infrastructure

- Setup of a virtual machine which imitates the TRD infrastructure at P2 in the lab
- Setup of a DIM DNS server for the lab network
- Setup of an Oracle database system housing the wingDB/gateDB (many thanks to Tom for his help!)
- Update of wingDB with all definitions from P2
- Setup of the Intercom layer (many thanks to Tom for his help!)
- Installation of various TRD command line tools (wing_tags, nginject, ...)
- Setup of a git repository with puppet scripts hosted at CERN gitlab to simplify such installations in the future (Tom)
- VM currently runs on FLP server
- VM and/or puppet scripts can be reused for tests at the surface at P2

Preparatory work: Adaption to the latest firmware and the CRUv2 card with new pinout

- Integration of the TRD custom link wrapper and user logic into the common CRUv2 firmware
- Writing of TRD specific python code compatible with the common cru-sw python framework

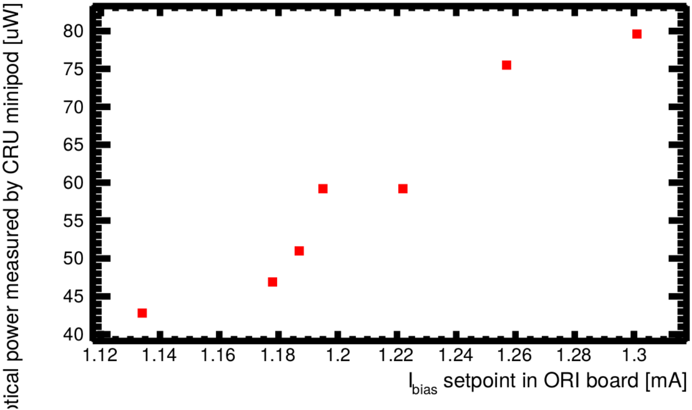
Current status

- Measurement of the optical power threshold by varying the ORI board configuration and reading error counters of the custom link wrapper

Now work in progress:

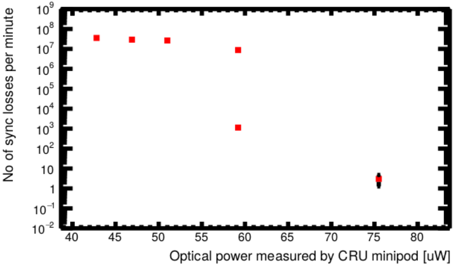
- Measurement of the final link BER using pseudo random data
- Lab test of the remaining parts of the readout chain including the trigger and header generation functionality (requires some implementations/discussions with CTP beforehand, see next slides)

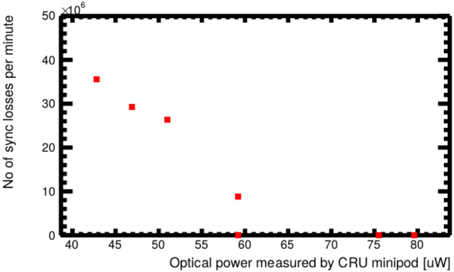
Results: ORI link stability tests



- Vary laser bias current
- Measure optical power at CRU with minipods via I2C bus

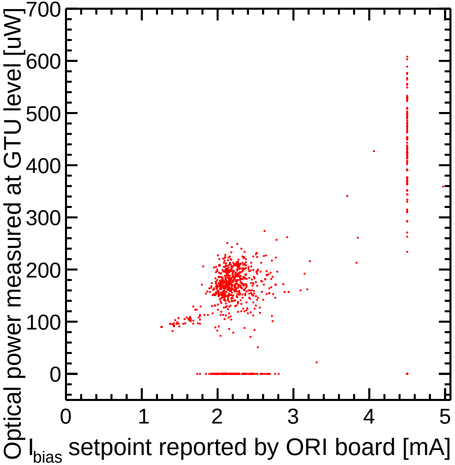
Results: ORI link stability tests





- Stable link operation possible with optical powers $> 80\ \mu$W at CRU input
- Laser diodes capable of delivering up to 1000 μW of optical power
- Total attenuation (cavern → CR1, conservative estimation): < 5.5 dB, translating to a power $> 281\ \mu$W at the CRU input

Results: ORI board settings at P2



- Majority of ORI boards currently between 2 mA and 3 mA laser bias current setpoint
- Absolute maximum rating for current setpoint: 12 mA
- Conclusions from plot:
  - ORI boards have enough power margin to be read out in CR1 (30% additional loss expected due to long fiber)
  - Optical connectors inside the central barrel apprently ok (no outliers with large power loss)

TRD CRU <-> CTP Interaction

- TRD CRU needs to communicate busy upstream via TTC-PON
- LTU needs to send triggers to FEE and CRU in parallel
- TRD specific functionalities to be implemented by CTP (but planned)
- CRU common firmware does not provide access to TTC-PON directly from user logic
- CRU common firmware currently tuned to work with continuous readout
- Currently no upstream messages planned after every event (only after full orbits (89.4μs))
- common logic takes care of flow control, HB accept/reject messages

TRD CRU <-> CTP Interaction

- New message type after every event will be introduced
- Common logic will be equipped with a busy input and take care of generating the appropriate busy upstream messages to the CTP with low latency
- TRD specific logic generates busy
- Additional 8 OLTs need to be purchased separately

Note on integration of TRD specific logic into common logic

- Currently firmware consists of link wrapper and user logic
- User logic takes care of trigger/busy handling and of packetization
- TRD would freeze without trigger/busy handling
- TRD user logic cannot be switched off or circumvented like for other detectors
- Therefore: Remove user logic completely and integrate all user logic functionalities into the link wrapper
- ⇒ Have only single TRD specific block in final firmware