RICH experience with MiniDAQ2

Antonino Sergi

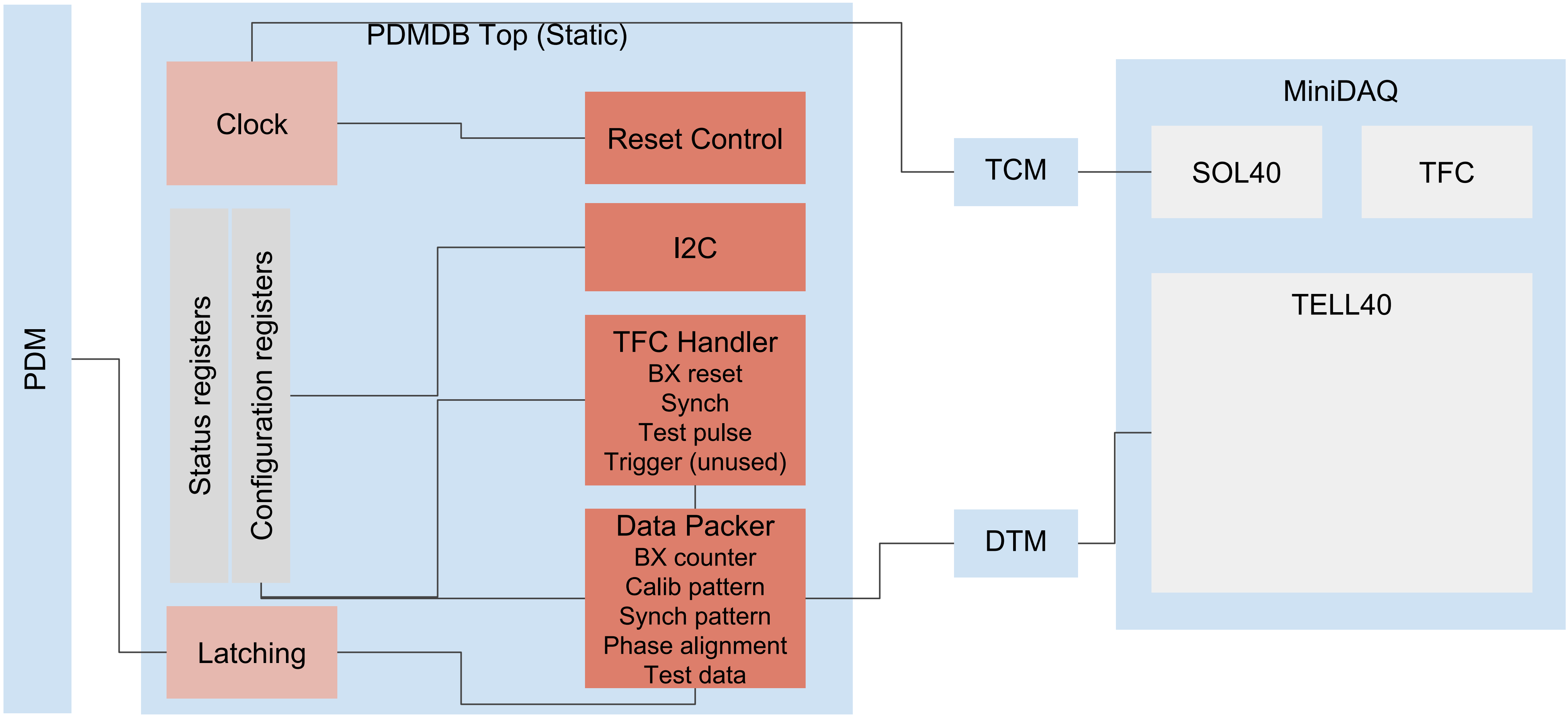
Outline

- Few (recycled) RICH slides:
  - An overview of our application
  - A couple of useful details
- Inter-system clock phases
  - Annoying stuff
- WinCC amenities
  - Our panels
  - Packaging and repo

Overview

- Successful test beam, but:
  - Phase alignment for the e-links can be annoying and unstable
  - Clock generation could be improved
  - Added input FIFO to delay 1 orbit (3564 clock cycles) to align data and trigger in the MiniDAQ1; just a patch: it will not be in the final design
- Plans for a final design, rad-hard:
  - Move the orbit synchronization in the MiniDAQ (up to them) Done, but not fully tested
  - Move most of the functionality in the TELL40 (Steve's suggestion) Done
  - Reorganize the clock distribution:
    - Drop the MMCM Done
    - Try to make it synchronous with the 40MHz master clock Done
  - Try to exploit Xilinx Partial Reconfiguration (PR) to minimize configuration time (15s/FPGA on MiniDAQ1 Done
  - Introduce Triple Module Redundancy (TMR), integrated with PR
    - Hierarchical design and implementation Done
    - Thin down the firmware as much as possible Done
    - Identify critical signals and apply triple voted logic Done
    - Exploit TELL40 to react to failures Done, but not fully tested

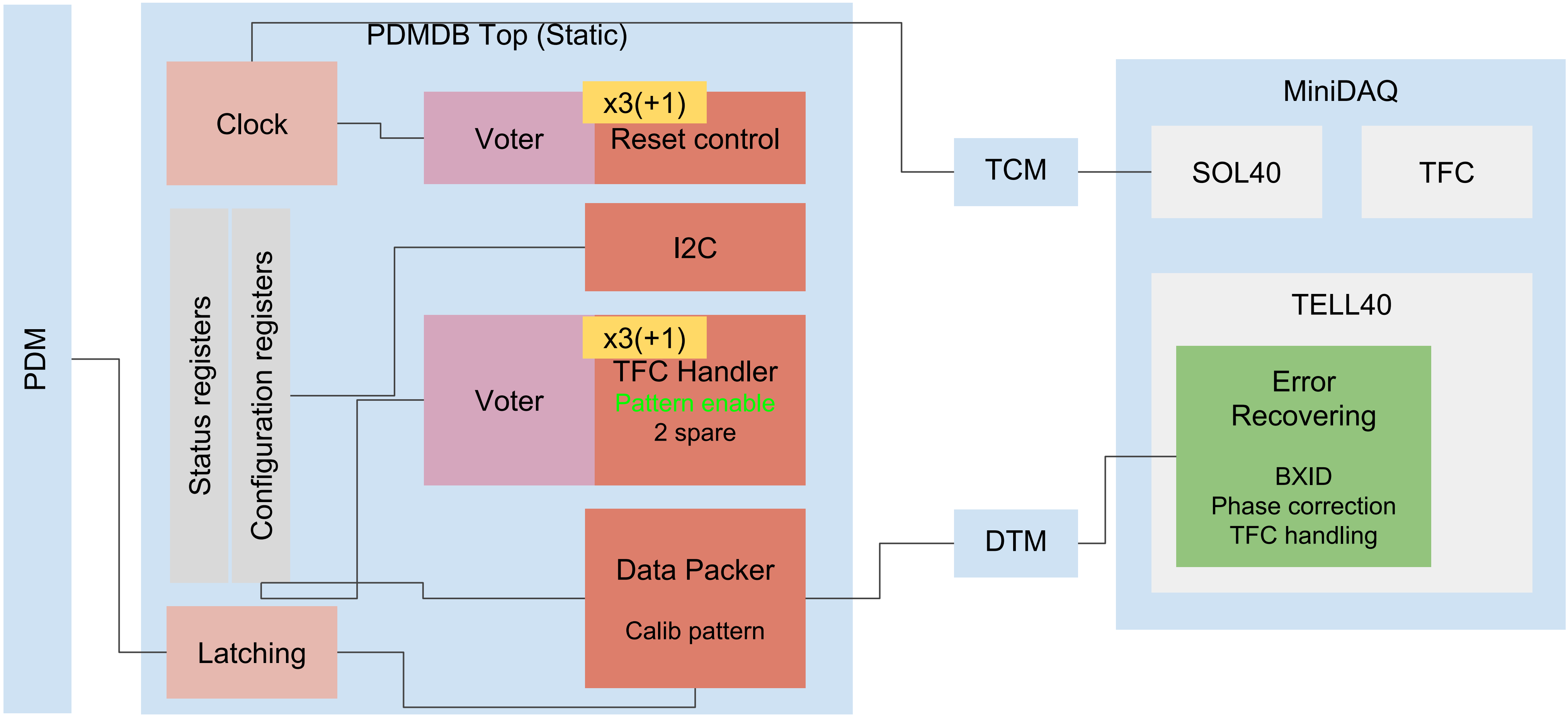
Old architecture (full functionality)



Full functionality on detector:

- Reading TFC commands
- Flagging data
- Not rad-hard

Thinned architecture with PR and TMR



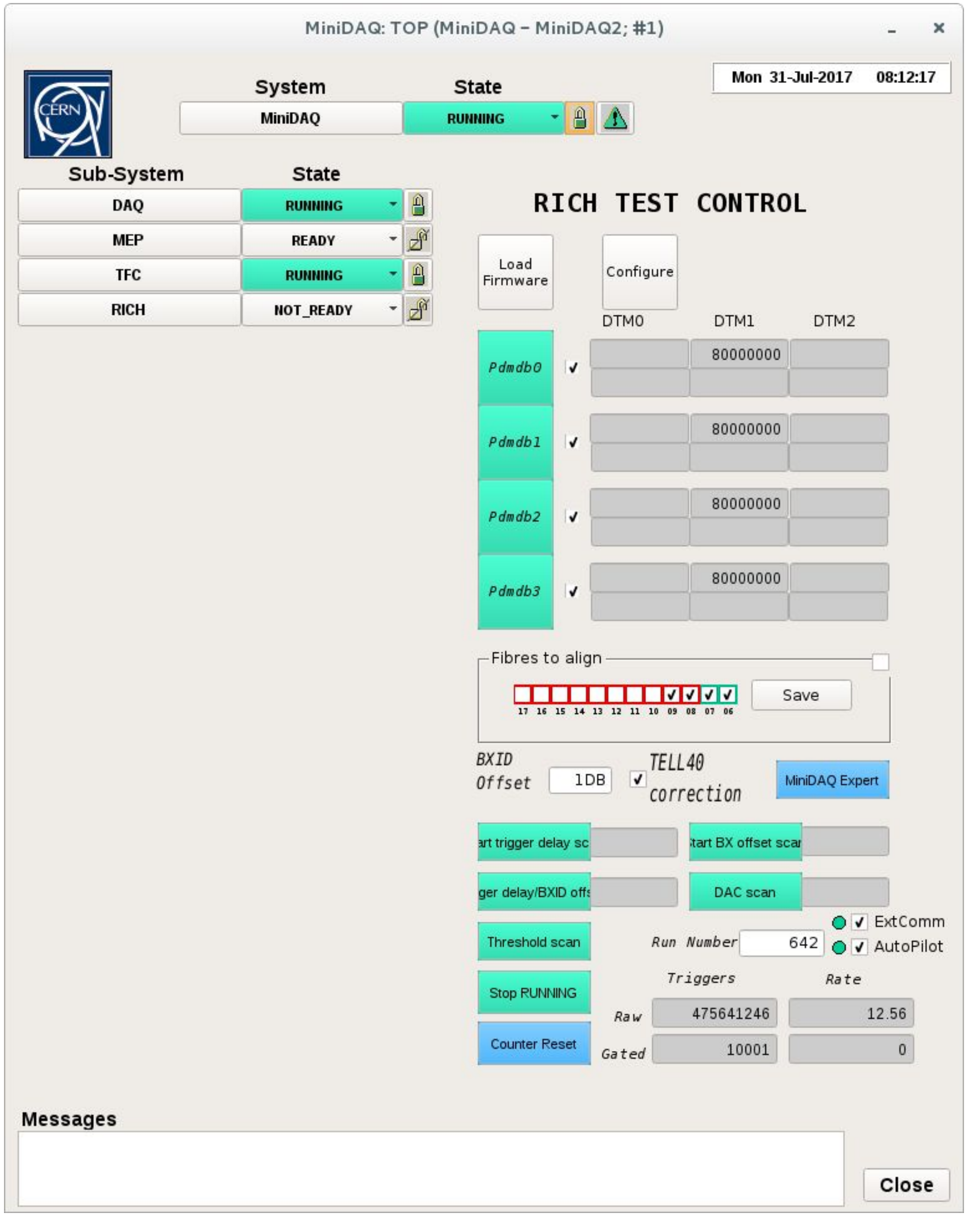
Individually reconfigurable modules:

- Voters can be part of them or not

Firmwares interaction and integration

- Master clock at 40MHz via GBTx
  - Multiplied by GBTx to 160MHz (we use e-links at 320Mb/s DDR) and delivered to FPGAs
  - We don’t use any PLL to get 40MHz, but a clock buffer divider
  - We don’t use the clock lines from the e-links
- 40MHz is used to pack the data
- 160MHz is used to serialize data to send via e-links
  - Individual serializers for e-links need to be aligned to the correct phase
- We managed to guarantee that the internal 40MHz is in phase with the master 40MHz
  - Otherwise .... the misalignement of data changes at each startup
  - We use the incoming e-links (for TFC) at 80Mb/s DDR

WinCC panels



MiniDAQ: TOP (MiniDAQ – MiniDAQ2; #1)

Mon 31-Jul-2017 08:12:17

System	State
MiniDAQ	RUNNING

Sub-System	State
DAQ	RUNNING
MEP	READY
TFC	RUNNING
RICH	NOT_READY

RICH TEST CONTROL

		DTM0	DTM1	DTM2
Pdmdb0	✓		80000000	
Pdmdb1	✓		80000000	
Pdmdb2	✓		80000000	
Pdmdb3	✓		80000000	

Fibres to align

17	16	15	14	13	12	11	10	09	08	07	06
□	□	□	□	□	□	□	□	☑	☑	☑	☑

BXID Offset 1DB

☑ TELL40 correction

● ☑ ExtComm

● ☑ AutoPilot

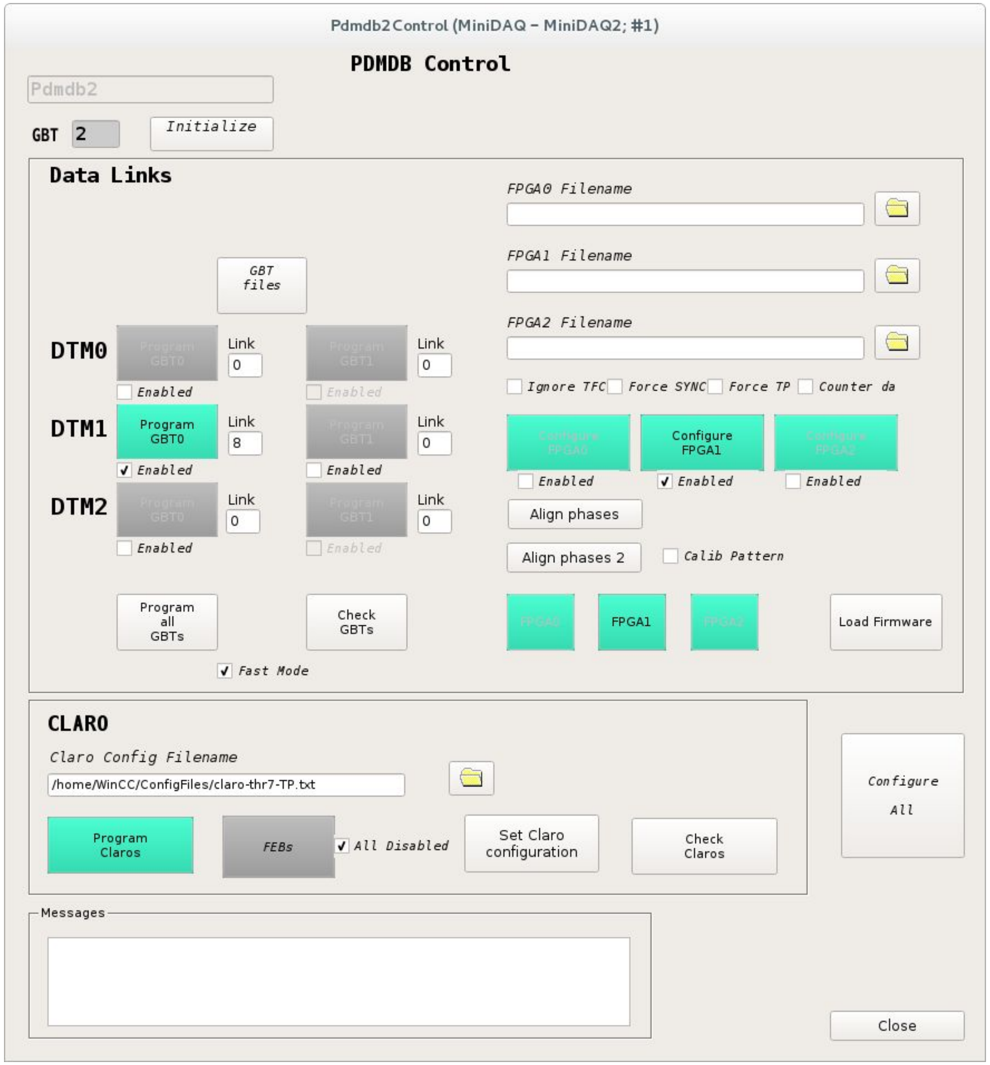
Run Number 642

	Triggers	Rate
Raw	475641246	12.56
Gated	10001	0

Messages

- What is provided is ok for quick tests, but starting to use it for measurements we needed more:
  - run number
  - step runs
    - threshold scan
    - DAC scan
    - latency scans
  - FE configuration:
    - CLARO config
    - loading firmware
  - Recipes
- Mostly compatible with MiniDAQ1

WinCC panels



GBT 2

DTM0: Link 0, □ Enabled; Link 0, □ Enabled

DTM1: Link 8, ☑ Enabled; Link 0, □ Enabled

DTM2: Link 0, □ Enabled; Link 0, □ Enabled

☑ Fast Mode

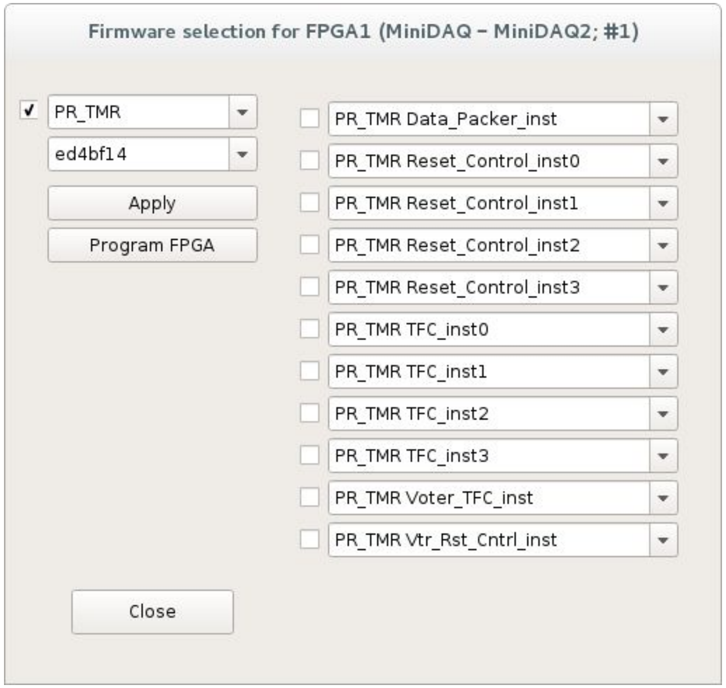
□ Ignore TFC □ Force SYNC □ Force TP □ Counter da

□ Enabled ☑ Enabled □ Enabled

□ Calib Pattern

Claro Config Filename: /home/WinCC/ConfigFiles/claro-thr7-TP.txt

☑ All Disabled



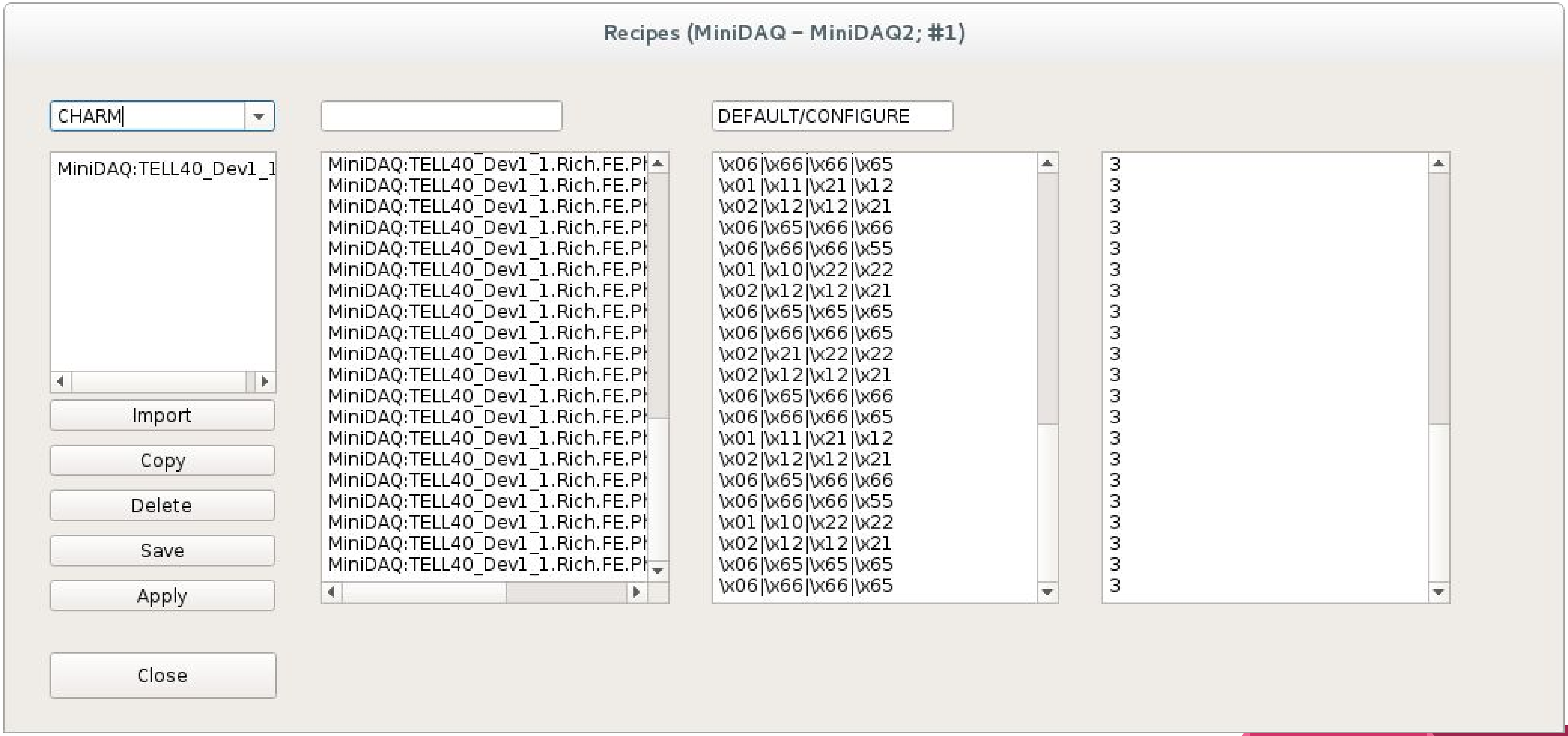
☑ PR_TMR

ed4bf14

□ PR_TMR Data_Packer_inst
□ PR_TMR Reset_Control_inst0
□ PR_TMR Reset_Control_inst1
□ PR_TMR Reset_Control_inst2
□ PR_TMR Reset_Control_inst3
□ PR_TMR TFC_inst0
□ PR_TMR TFC_inst1
□ PR_TMR TFC_inst2
□ PR_TMR TFC_inst3
□ PR_TMR Voter_TFC_inst
□ PR_TMR Vtr_Rst_Cntrl_inst

- SCA-I2C/SPI/JTAG are supported for a single instruction:
  - we needed a bunch of panels and scripts
- A lot faster (x10) on MiniDAQ2
  - CPU usage of GBtServ, WCCOAxx ~20%

WinCC panels



Recipes (MiniDAQ – MiniDAQ2; #1)

CHARM

DEFAULT/CONFIGURE

```
MiniDAQ:TELL40_Dev1_1
```

```
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
MiniDAQ:TELL40_Dev1_1.Rich.FE.P
```

```
\x06|\x66|\x66|\x65
\x01|\x11|\x21|\x12
\x02|\x12|\x12|\x21
\x06|\x65|\x66|\x66
\x06|\x66|\x66|\x55
\x01|\x10|\x22|\x22
\x02|\x12|\x12|\x21
\x06|\x65|\x65|\x65
\x06|\x66|\x66|\x65
\x02|\x21|\x22|\x22
\x02|\x12|\x12|\x21
\x06|\x65|\x66|\x66
\x06|\x66|\x66|\x65
\x01|\x11|\x21|\x12
\x02|\x12|\x12|\x21
\x06|\x65|\x66|\x66
\x06|\x66|\x66|\x55
\x01|\x10|\x22|\x22
\x02|\x12|\x12|\x21
\x06|\x65|\x65|\x65
\x06|\x66|\x66|\x65
```

```
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
3
```

- Recipes are still a dev feature:
  - only one recipe in the FW
  - we supplemented it cheating

Summary

- Detector side:
    - BXID flagging: offsets and latency may not be obvious to find, especially with asynchronous triggers (test beam case)
    - GBTx e-links phase alignment: it took some effort to make it reproducible after powercycles
- MiniDAQ2
    - After patching the LLI clock tree it is usable (standalone)
    - Significant improvement on the SCA side in terms of speed of operation
        - 1.1MB Xilinx (compressed) bitstream
            - MiniDAQ1: 15(30) seconds with GbtServ off(on)
            - MiniDAQ2: 2-3 seconds
        - Useful in tests
        - Very relevant in real applications
    - The first one broke after a few weeks of usage (DC-DC converters), and it’s being replaced
- WinCC
    - It usually works (besides versioning stuff)
    - To use it for test beams or measurements quite some work is required
    - Not very multi-developer friendly yet (subdetector components)