Readout Column Plans

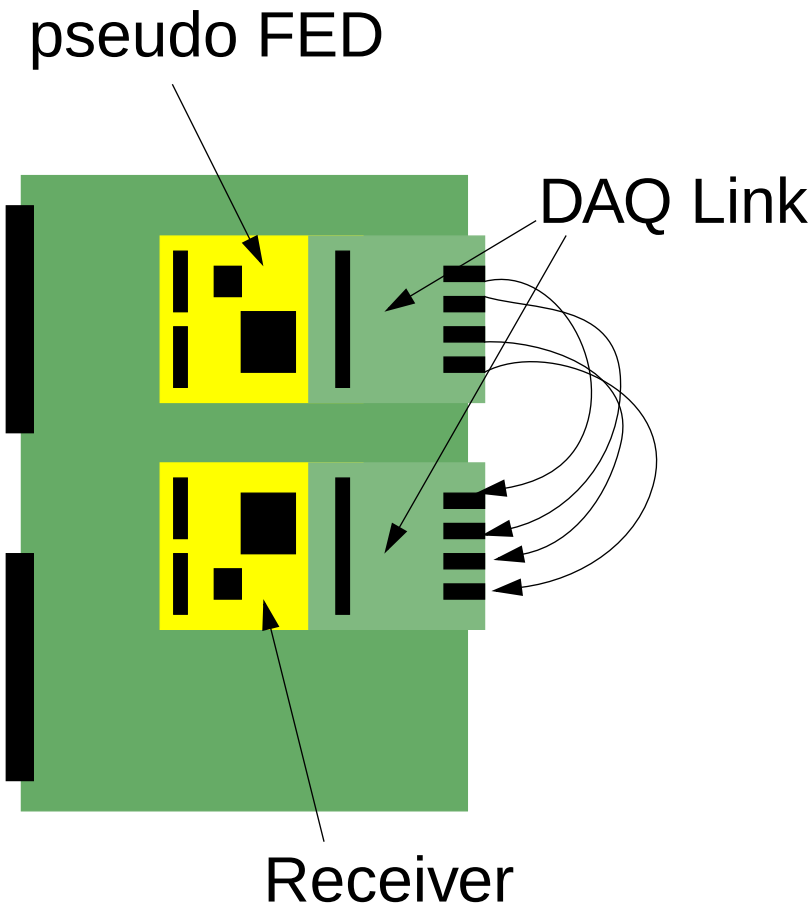
stage 1 setup (february)

stage 2 setup (june)

implementation ideas

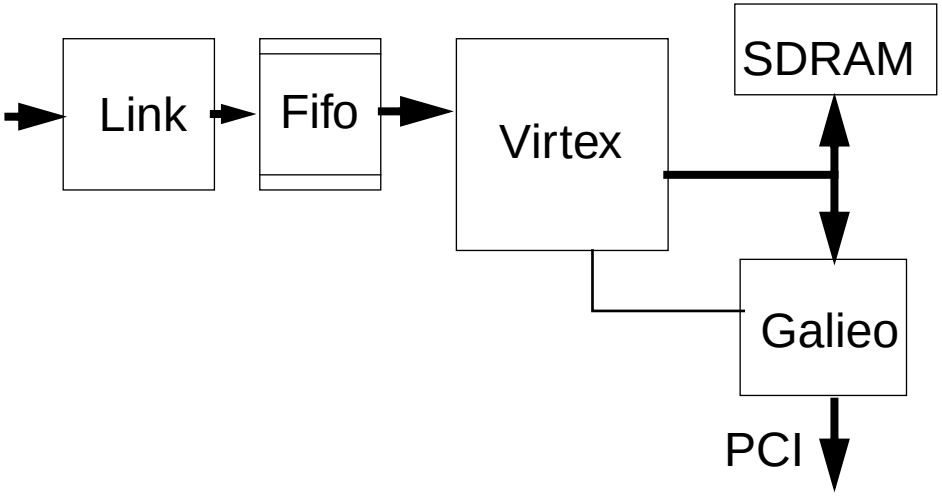
Stage 1: RUI - DAQ-link

- Purpose
  - data transfer over DAQ link
  - bandwidth of link
- Hardware
  - 2x generic II
  - 2x Siemens Link
  - 1x VME-processor with two PMC slots



stage 1 : items to do

- pseudo FED
  - generate dummy events on PCI - trigger
- Receiver
  - develop “MMU” with descriptor table
  - set up host-to-PCI bridge (Galileo) for DMA (FPGA --> memory)
  - readout of Receiver by PCI (without being ambitious...)



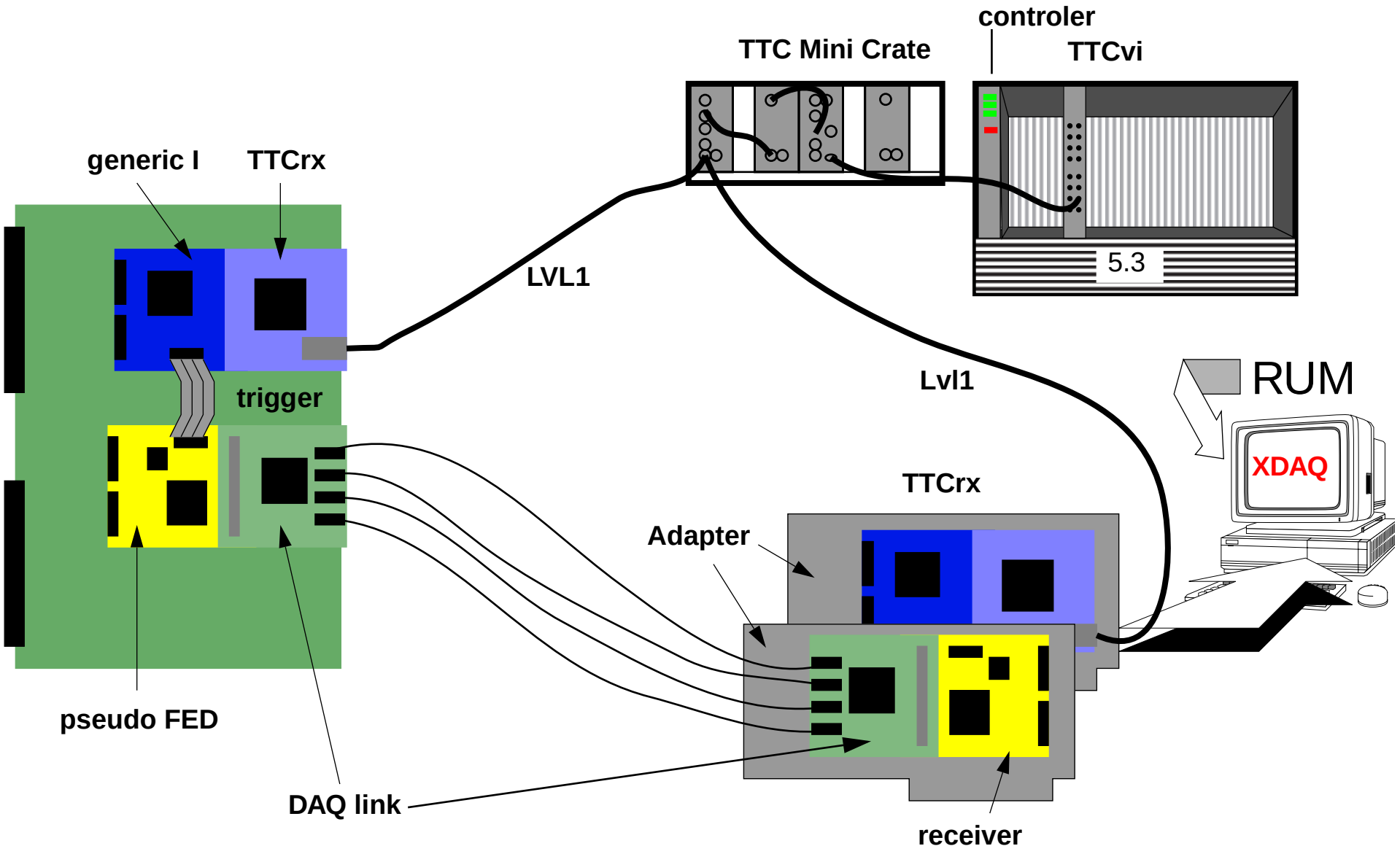
- Software
  - if available: use i2ocore library to access hardware.

Aim: results end of february

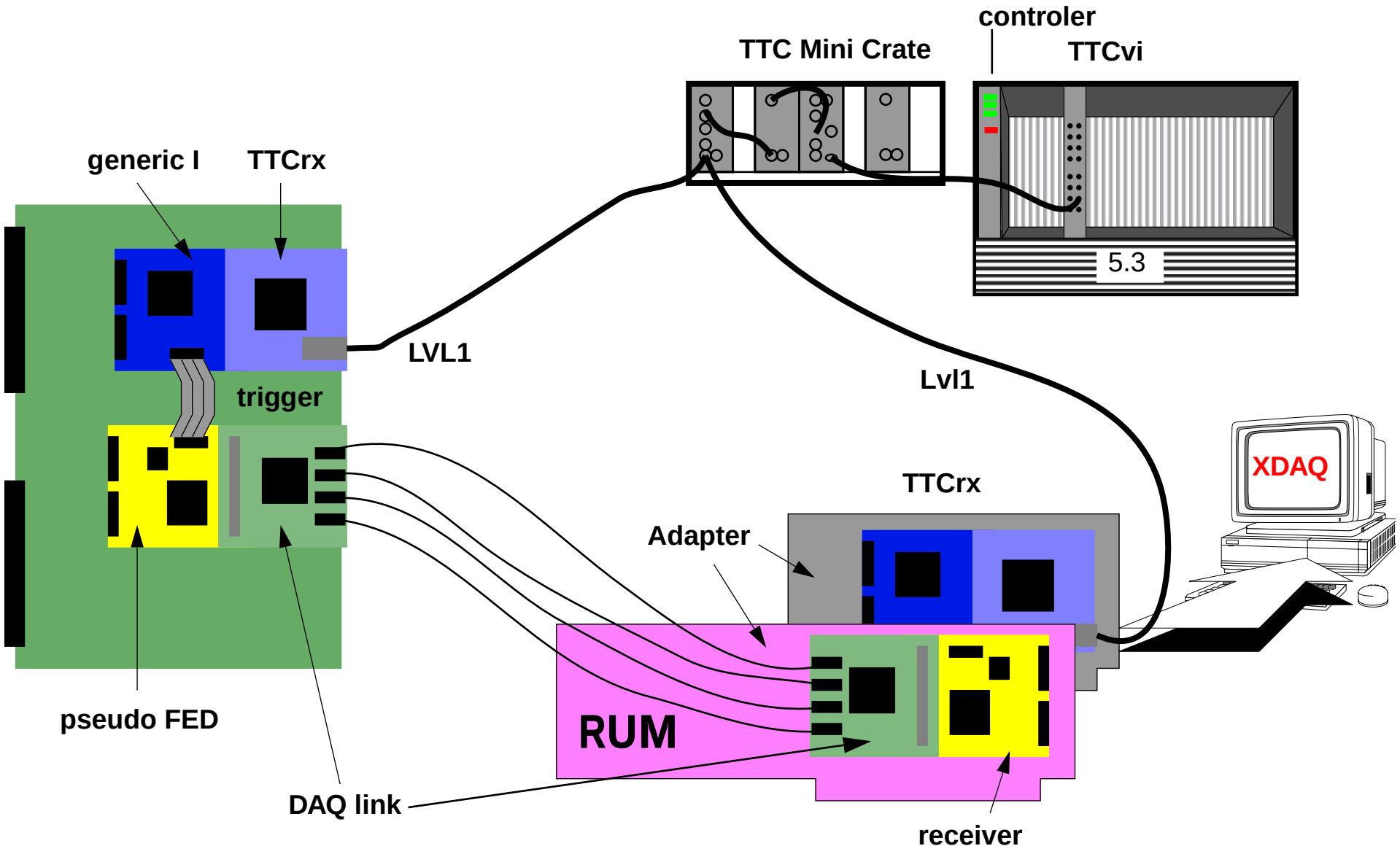
Stage 2 : Interface LINK - RUM

- Purpose:
  - build a DAQ chain Pseudo-FED - RUM
  - use software and hardware RUM
  - develop interface DAQ-Link RUM
  - measure performance Daq-Link RUM
  - identify bottlenecks

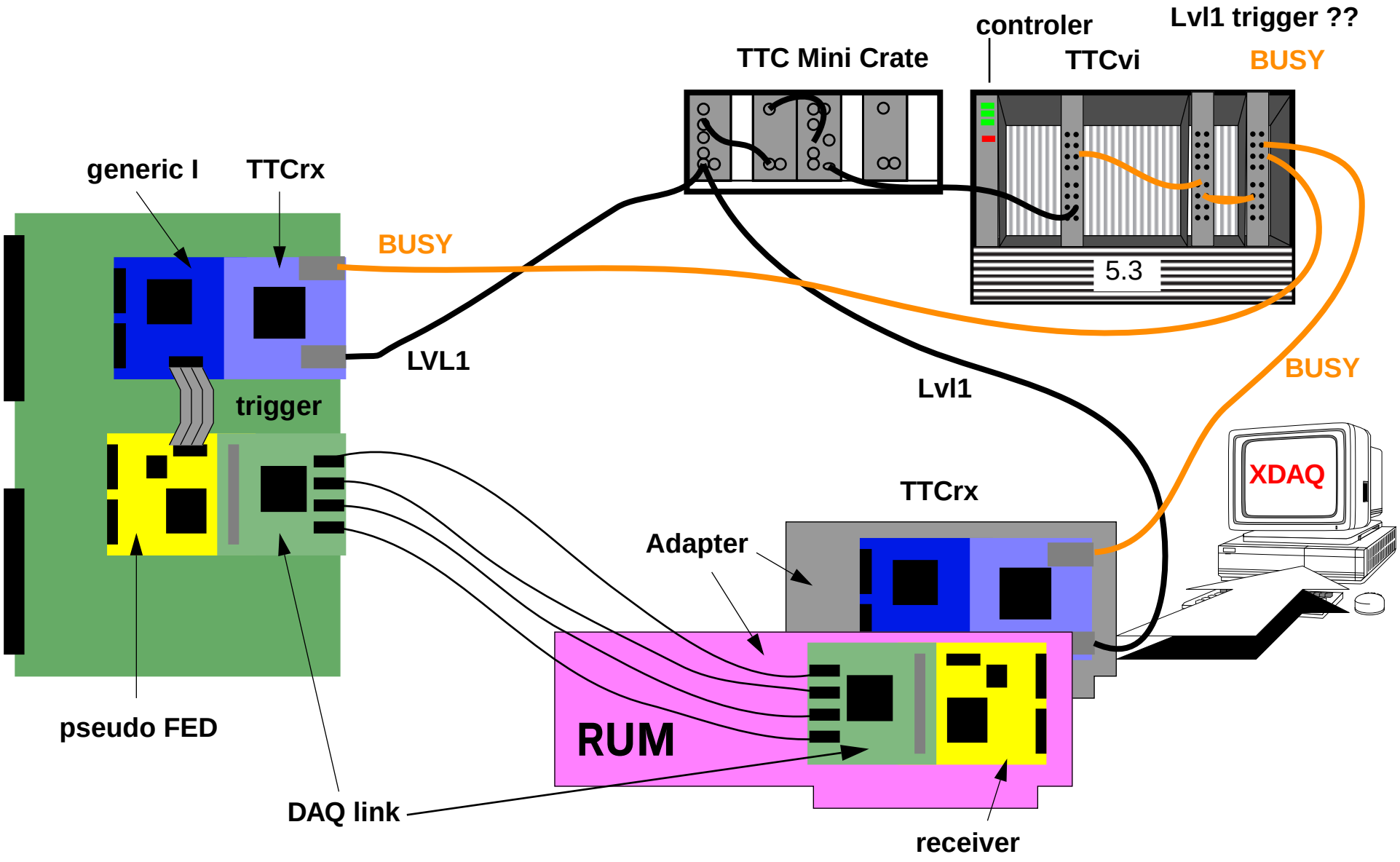
stage 2 a : with software RUM



stage 2b : with hardware RUM

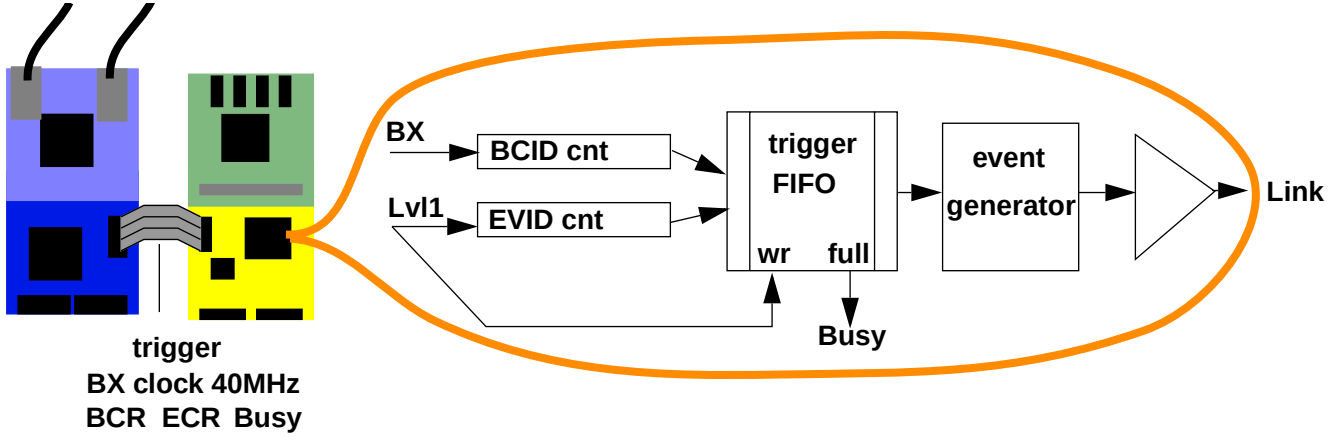


implementation of BUSY

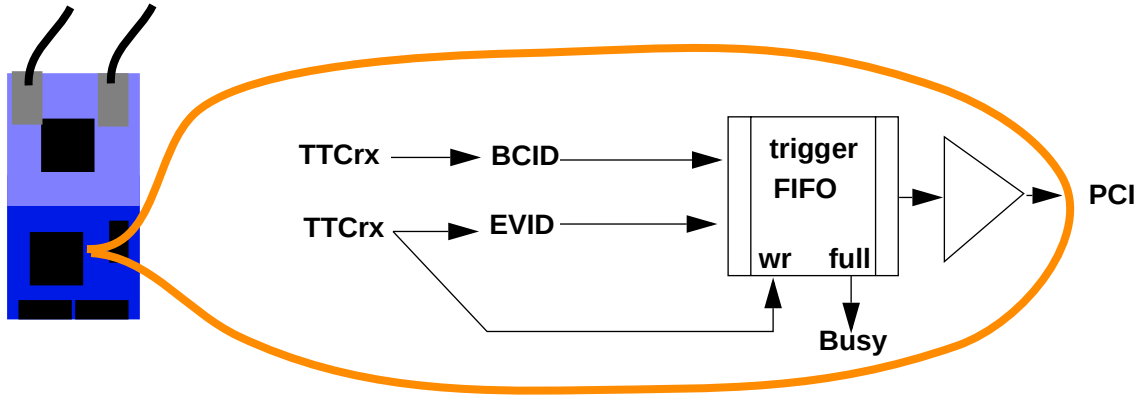


Trigger receiver board

- At pseudo FED:



- In RM PC:



Trigger details

- Trigger generated by software
  - \+ /- Busy easy to implement by software, but slow
  - \- only low trigger rates
- Trigger generated by TTCvi

  TTCvi provides internal trigger generator with various frequences
  - \+ easy to use
  - \+ no additional hardware needed
  - \- no inhibit (=busy) input
  - \- BUSY would have to be implemented with software ==> slow
- Trigger generated by external logic (NIM , VME-pulser, generic I)
  - \+ finer control on trigger rate
  - \+ easy implementation of an inhibit (=busy) input
  - \- extra hardware needed
  - \- only constant trigger rates easy to implement

Interface Daq Link - RUM

- Software
  - RUM software must PULL data.
    (RUI does not know where to write data)
- Hardware RUM

Three different possibilities

1) RUI pushes whenever it has data
   - retry in case the RUM has not yet received EVT tag
   - RUM only slave on this PCI bus
   - RUI design must be different when working with Software RUM
2) RUM requests data from RUI, then RUI pushes data
   - needs Master / Slave capability on bus
3) RCN arrives at RUI
   - RUI pushes data to RUM
   - RUM design simpler
   - RUI design more complicated (currently not implemented)

Stage 2 : items to do

- Interface Link - RUM
  - software RUM / hardware RUM
  - optimize for performance
  - identify factors which limit performance
- Trigger “bricolage”
  - Pseudo Trigger with BUSY input
  - trigger rate counting (in pseudo trigger or in TTCrx logic)
  - use setup of Claude
  - adapt Claudes FPGA to become RM input

build trigger throtteling system (BUSY module ?)

AIM : results in June