Fast Optical Readout Chain for the Mu3e Experiment

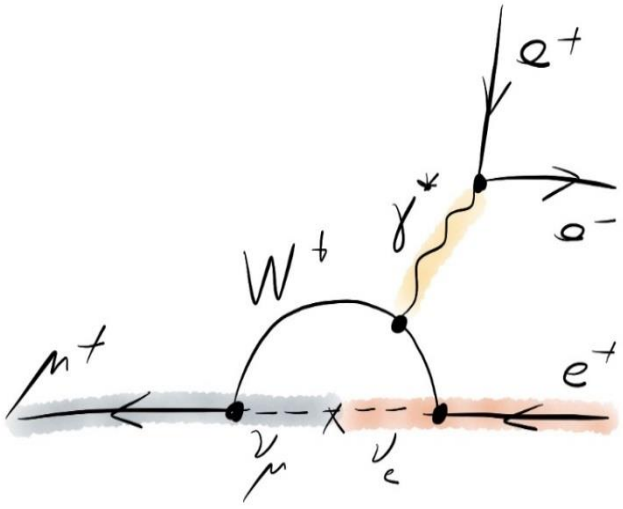
Qinhua Huang
on behalf of the Mu3e collaboration

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

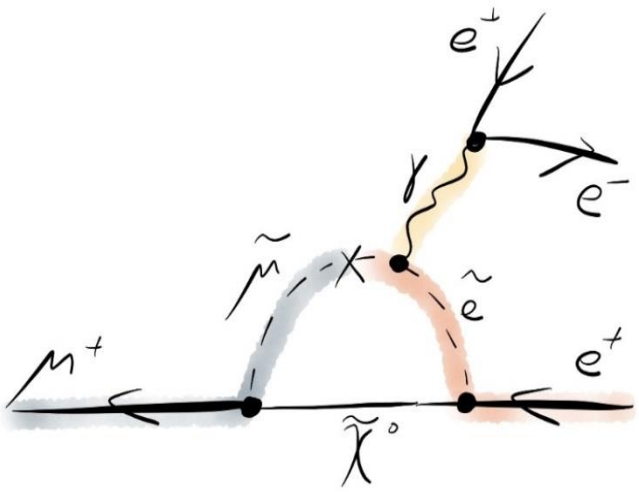
The Mu3e Experiment

Main purpose :

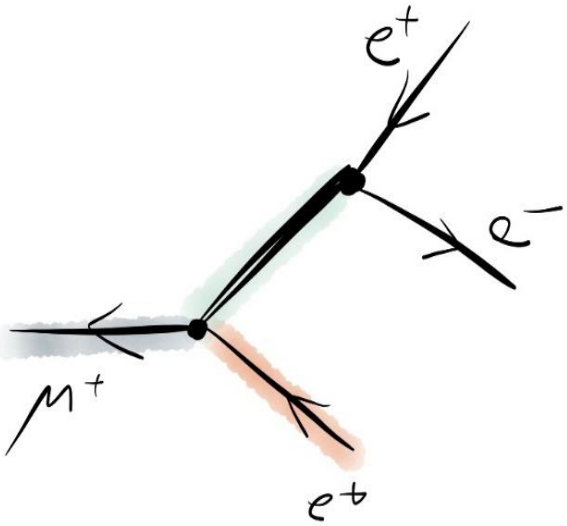
search for $\mu^+ \rightarrow e^+e^+e^-$ decay with sensitivity of 10^{-16}



SM Neutrino mixing
BR < 10^{-54}



SUSY

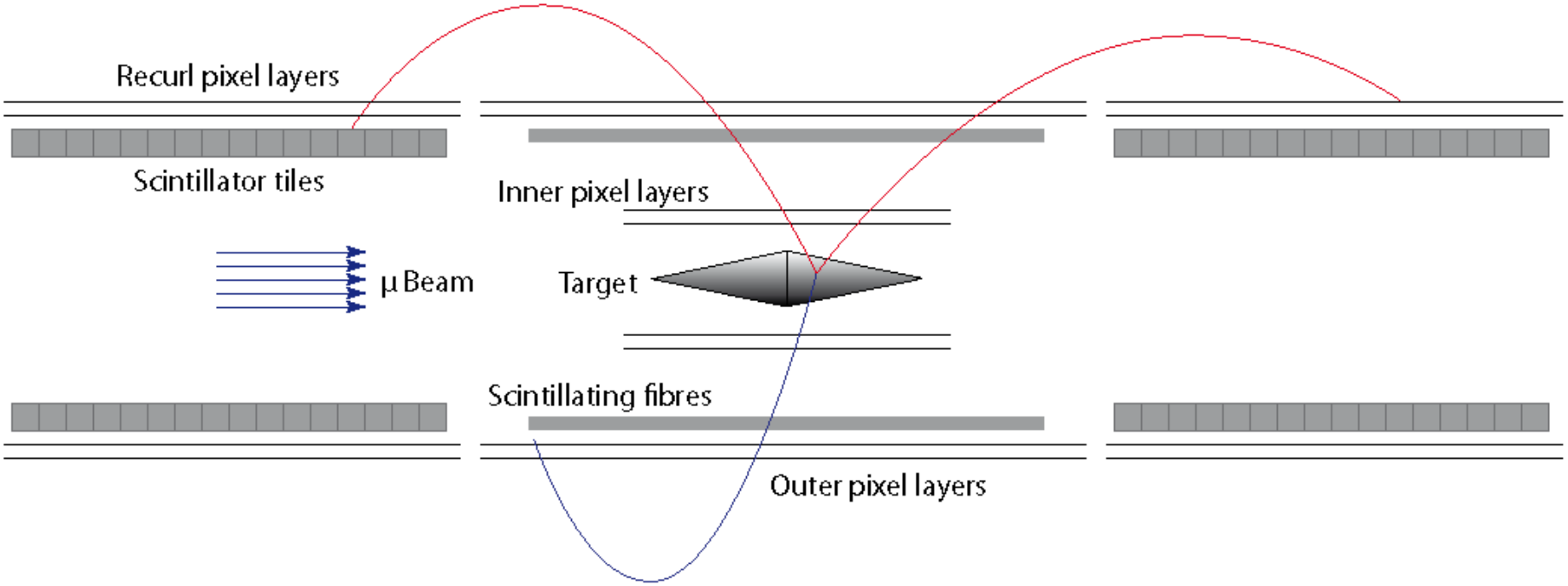


Tree-level lepton flavor violation

The Mu3e Experiment

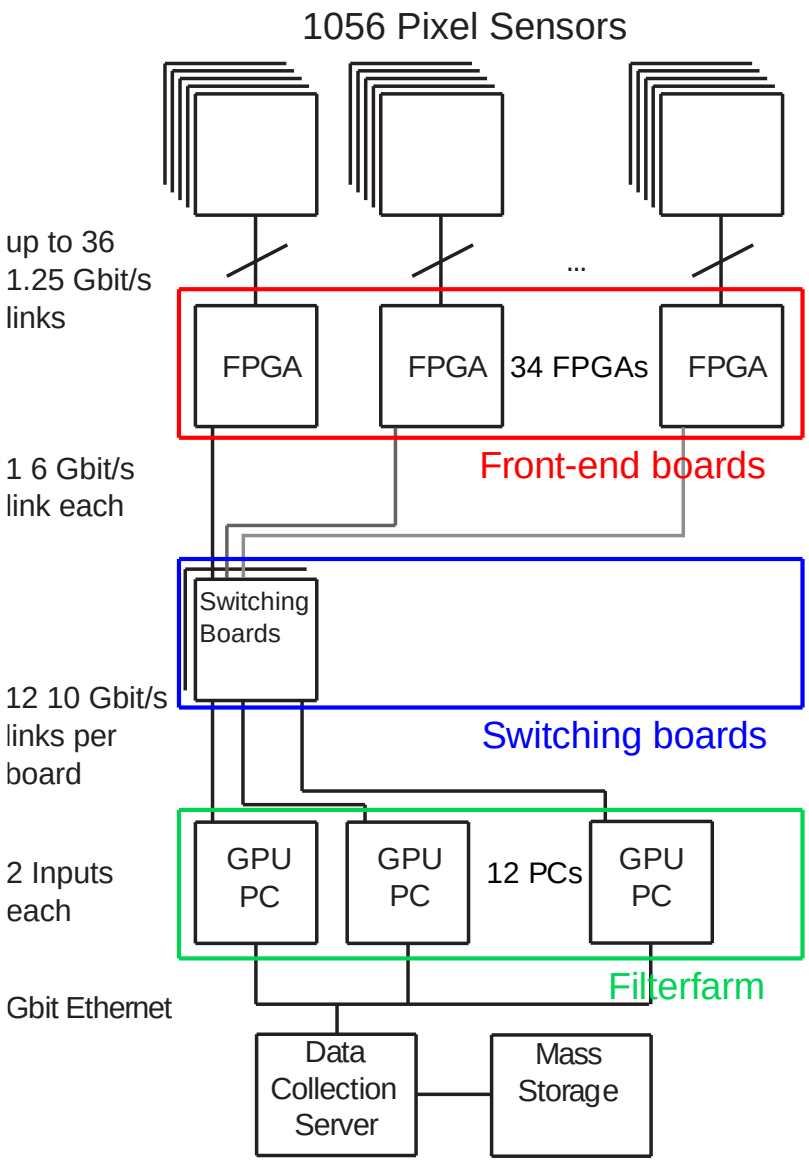
The Mu3e Detector

- Tracking detector : pixel sensors
- Timing detector : scintillating fibers and scintillator tiles



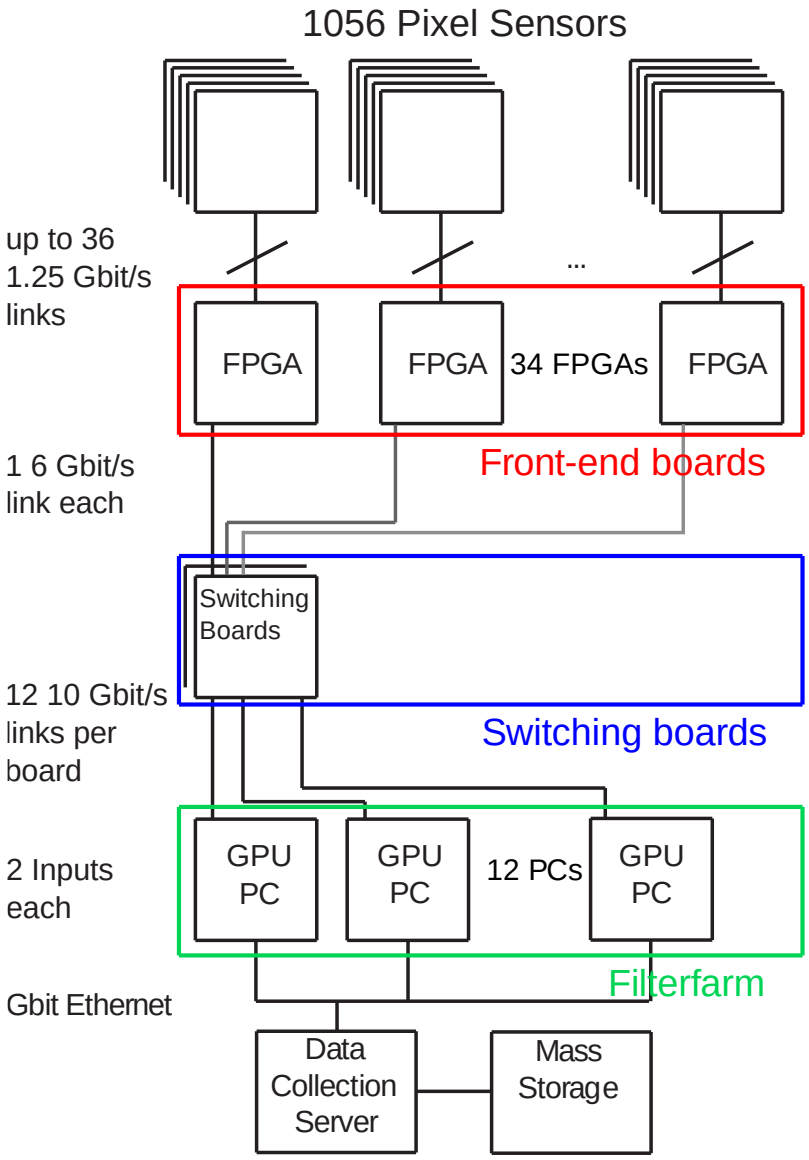
Data acquisition

- Triggerless DAQ
  - Front-end boards (T22.4)
    - Decode and merge data
    - Time-sorting
  - Switching boards (This talk)
    - Space-sorted to Time-sorted
  - GPU Filter farm (T42.5 T42.6)
    - Fast track finding
    - online reconstruction
    - Reduce data rate by a factor ~1000



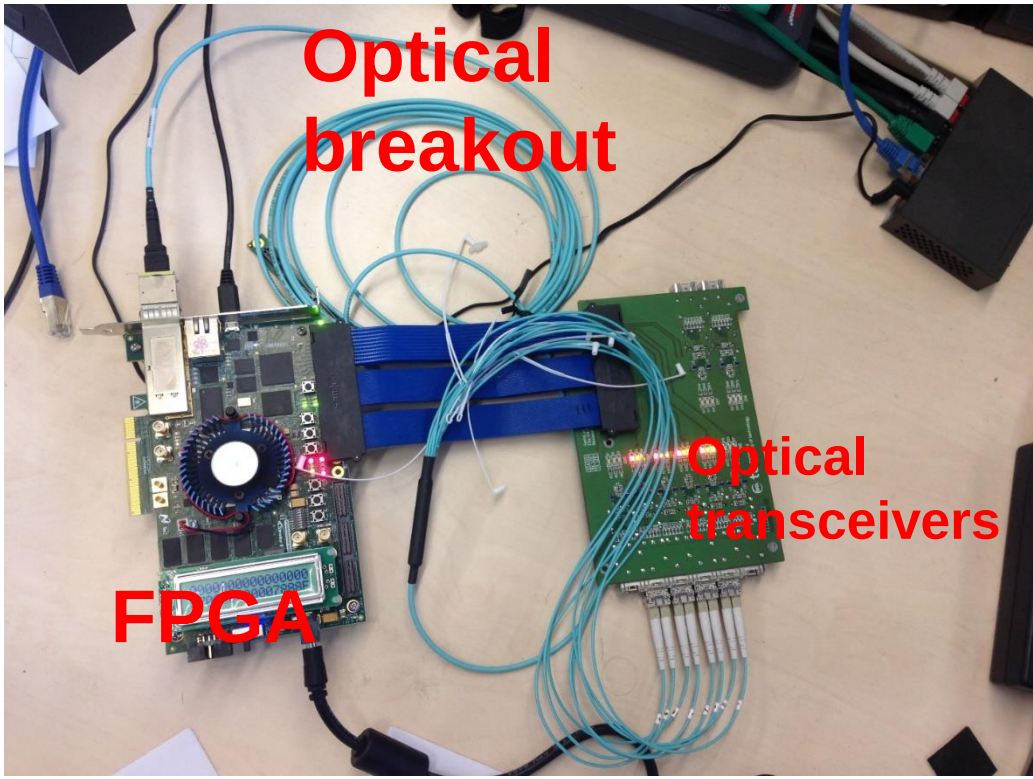
Data acquisition

- Switching boards
  - Switch the data stream between front-end FPGAs and the filter farm
  - Merge the data of sub-detectors and the data from different front-end boards
  - Time-sorting for the data of the whole detector
- Optical links
  - Reliability of the links at high speed > 6 Gpbs



Fast optical link Bit Error Rate Test

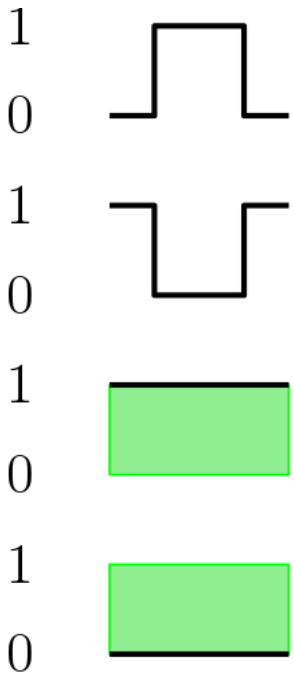
- Components for the real detector partially available
- A test chain using the available electronic components in our lab
- BER < 10^{-13} in the specification of the optical transceiver
- Bi-directional data stream
- Measure for roughly 11×24 hours, and see no error

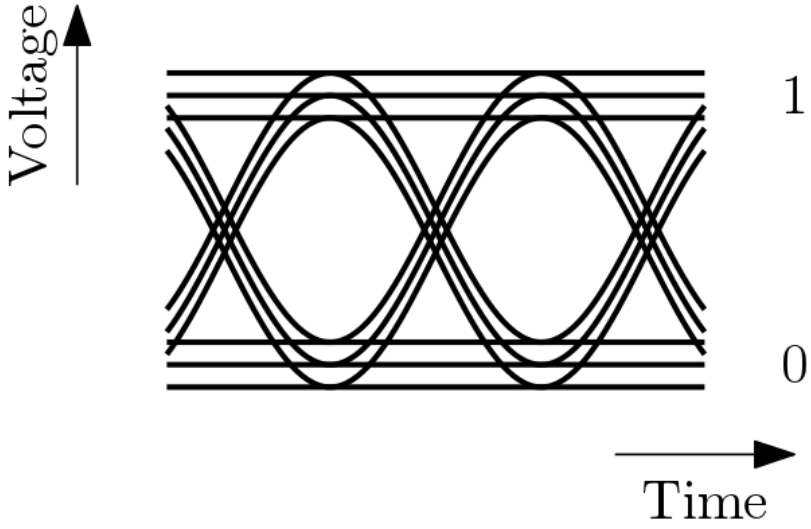


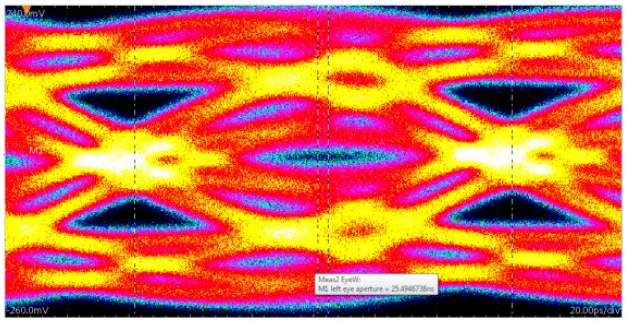
Fast optical link BERT : Result

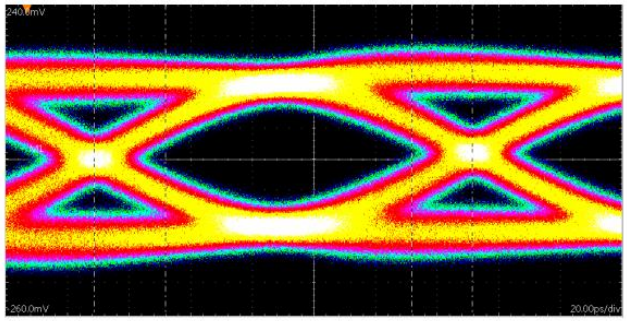
- BER < 7×10^{-16} (90% CL) at data rate 5650 Mbps with the breakout, better than the BER given in the specification
- Better BER because of the tuned transceiver, tuning can give signals with better quality

Eye diagram : Not tuned and tuned







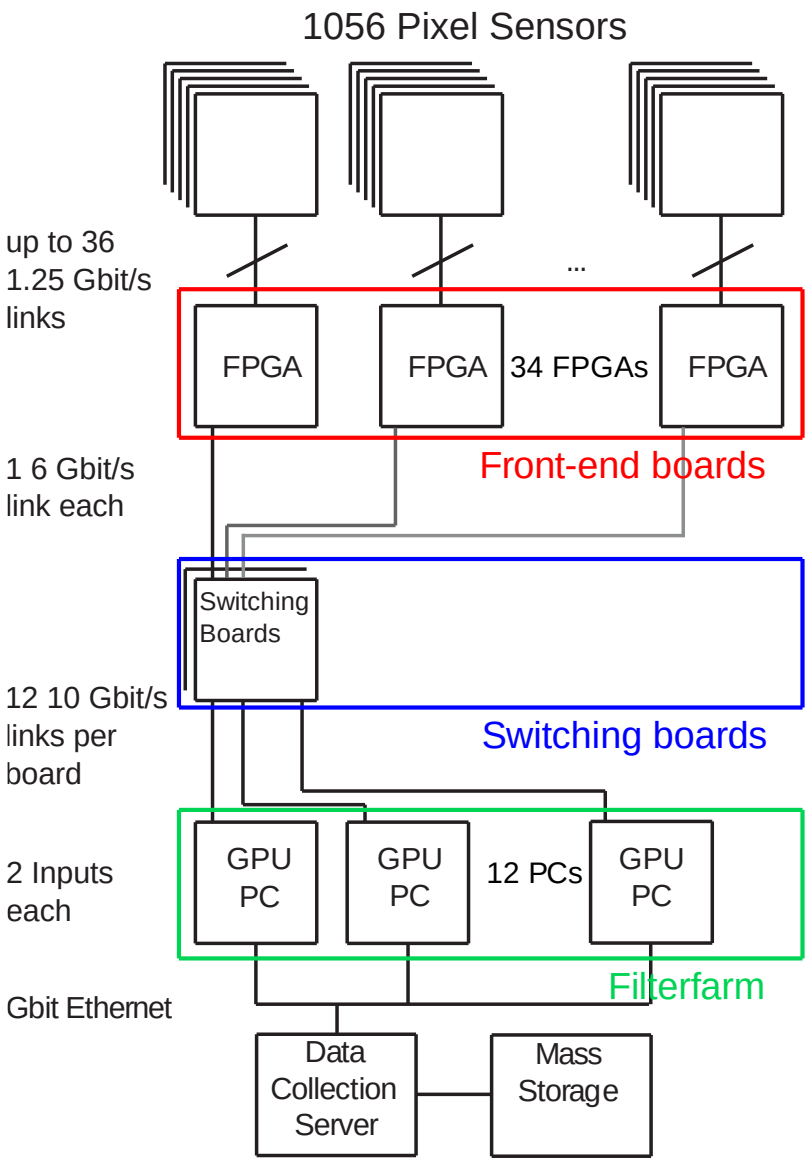


At 8 Gbps

Corrodi, S., Fast Optical Readout of the Mu3e Pixel Detector, Master thesis, 2014

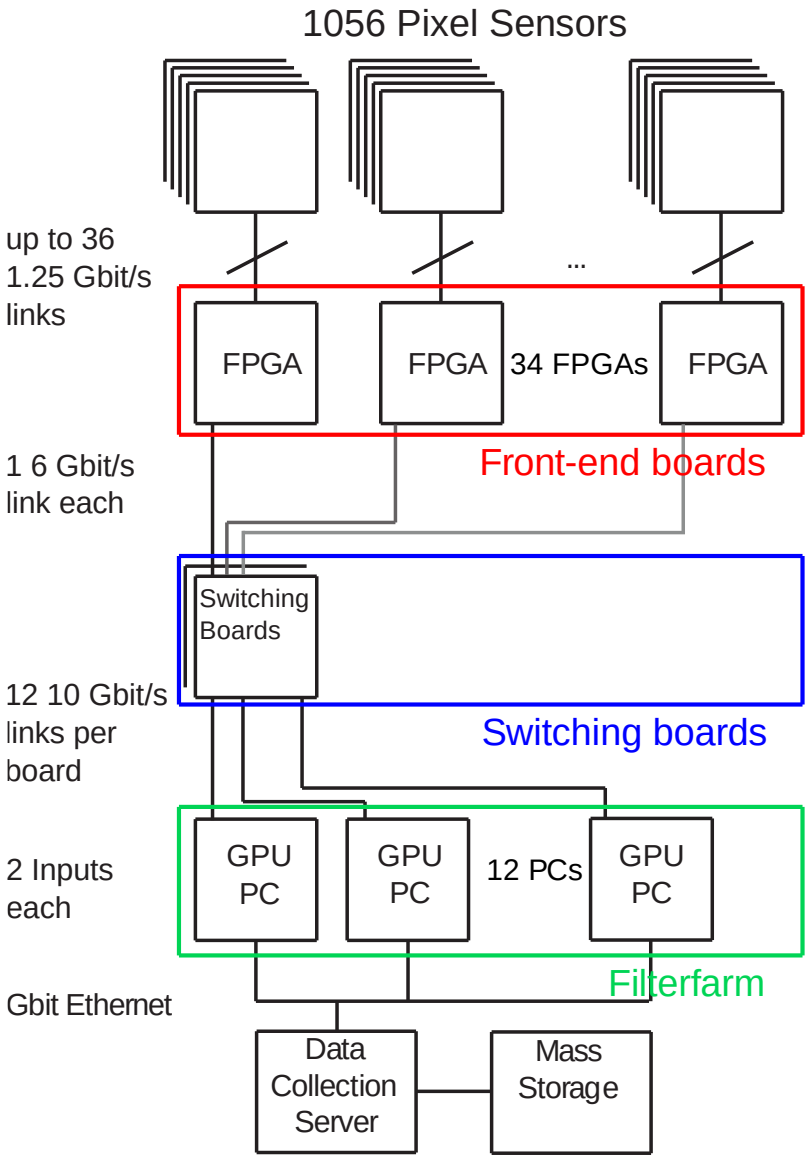
Switching FPGA

- Firmware tasks
  - Switch the data stream between front-end FPGAs and the filter farm
  - Merge the data of sub-detectors and the data from different front-end boards
  - Time-sorting for the data of the whole detector
  - Distribute the slow control signals to the front-end FPGAs
- Requirement
  - Reliable and extremely fast!
  - Throughput ~O(25Gbps)



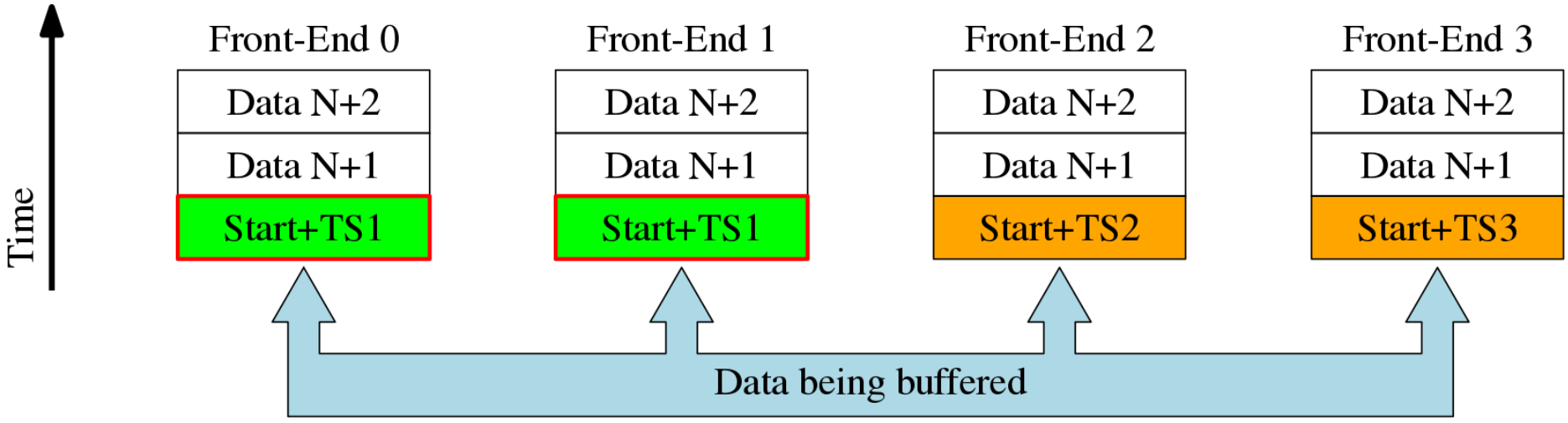
Switching FPGA

- Firmware tasks
  - Switch the data stream between front-end FPGAs and the filter farm
  - *Merge the data of sub-detectors and the data from different front-end boards*
  - Time-sorting for the data of the whole detector
  - Distribute the slow control signals to the front-end FPGAs
- Requirement
  - Reliable and extremely fast!
  - Throughput ~O(25Gbps)



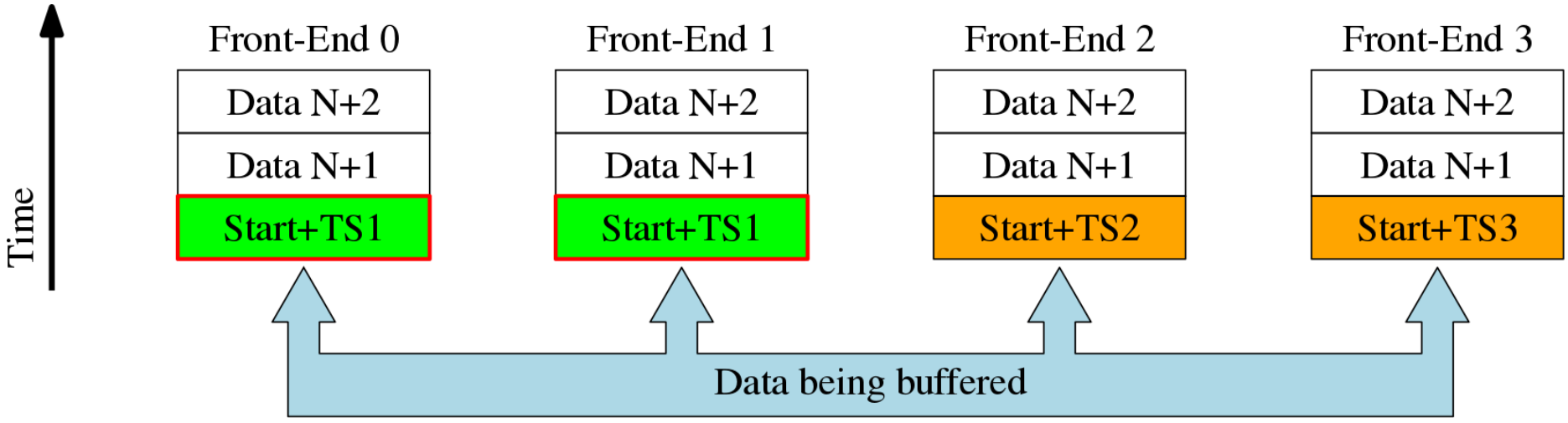
Switching FPGA : Merge algorithm on VHDL

- Synchronize the data from different front-end boards
- Buffer and check the incoming data
- The front-end data are already time-sorted



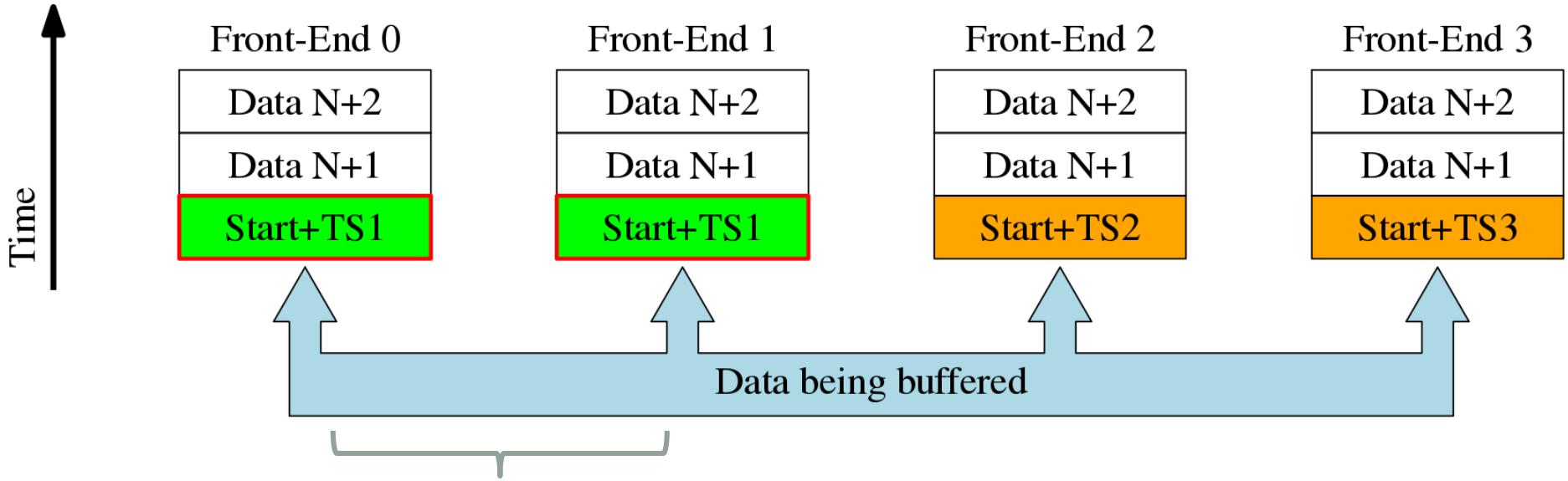
Switching FPGA : Merge algorithm on VHDL

- Start algorithm when all channels receive the frame headers



Switching FPGA : Merge algorithm on VHDL

- Check the “Hit Block Counter” by round-robin
- Start merging all frames with same minimal “Frame Time Stamp”

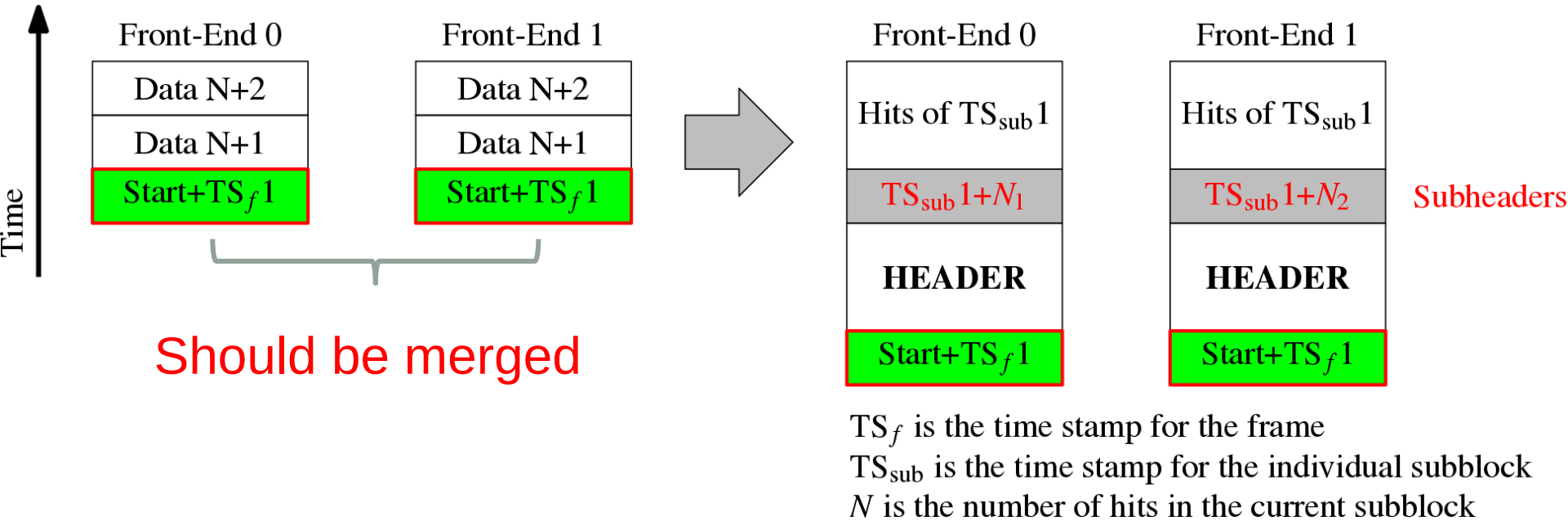


Example : TS1 < TS2 < TS3

Should be merged

Switching FPGA : Merge algorithm on VHDL

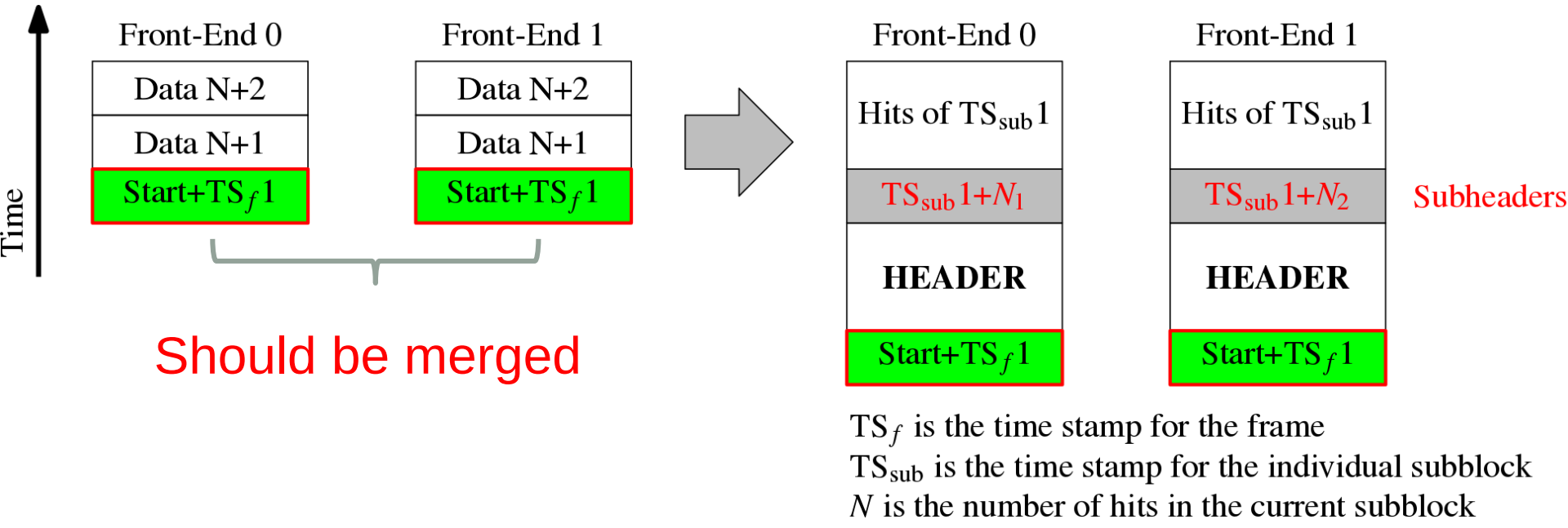
- Check the “Time Stamp” of the subblocks by round-robin



TS$_f$ is the time stamp for the frame
TS$_{sub}$ is the time stamp for the individual subblock
N is the number of hits in the current subblock

Switching FPGA : Merge algorithm on VHDL

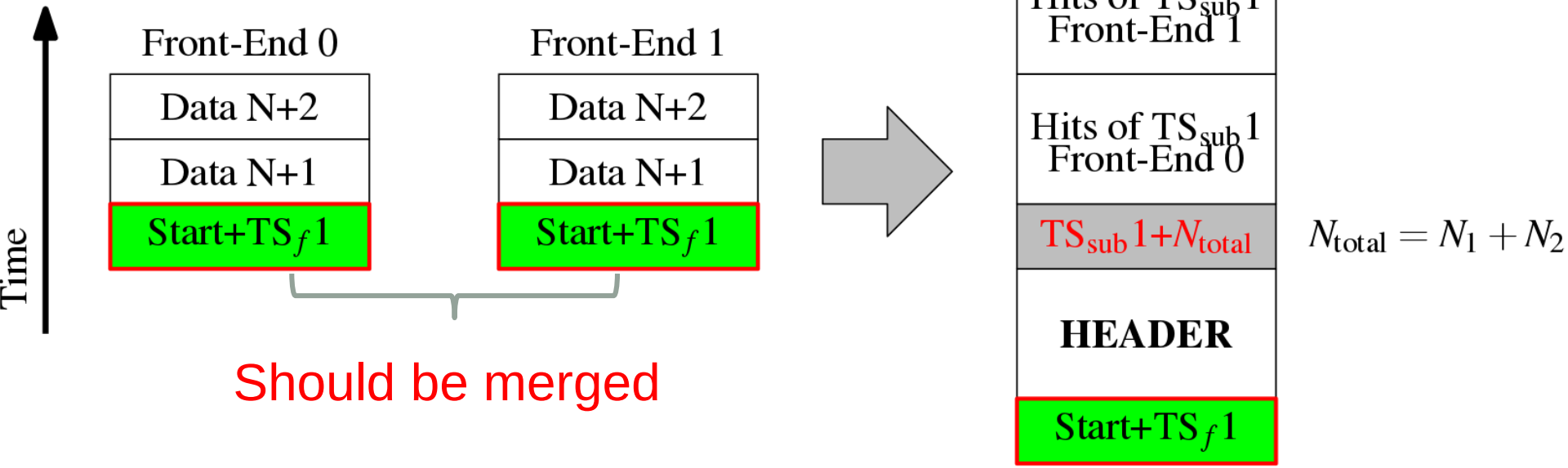
- Check the “Time Stamp” of the subblocks by round-robin
- If multiple subblocks have same “Time Stamp” then merge



TS$_f$ is the time stamp for the frame
TS$_{sub}$ is the time stamp for the individual subblock
N is the number of hits in the current subblock

Switching FPGA : Merge algorithm on VHDL

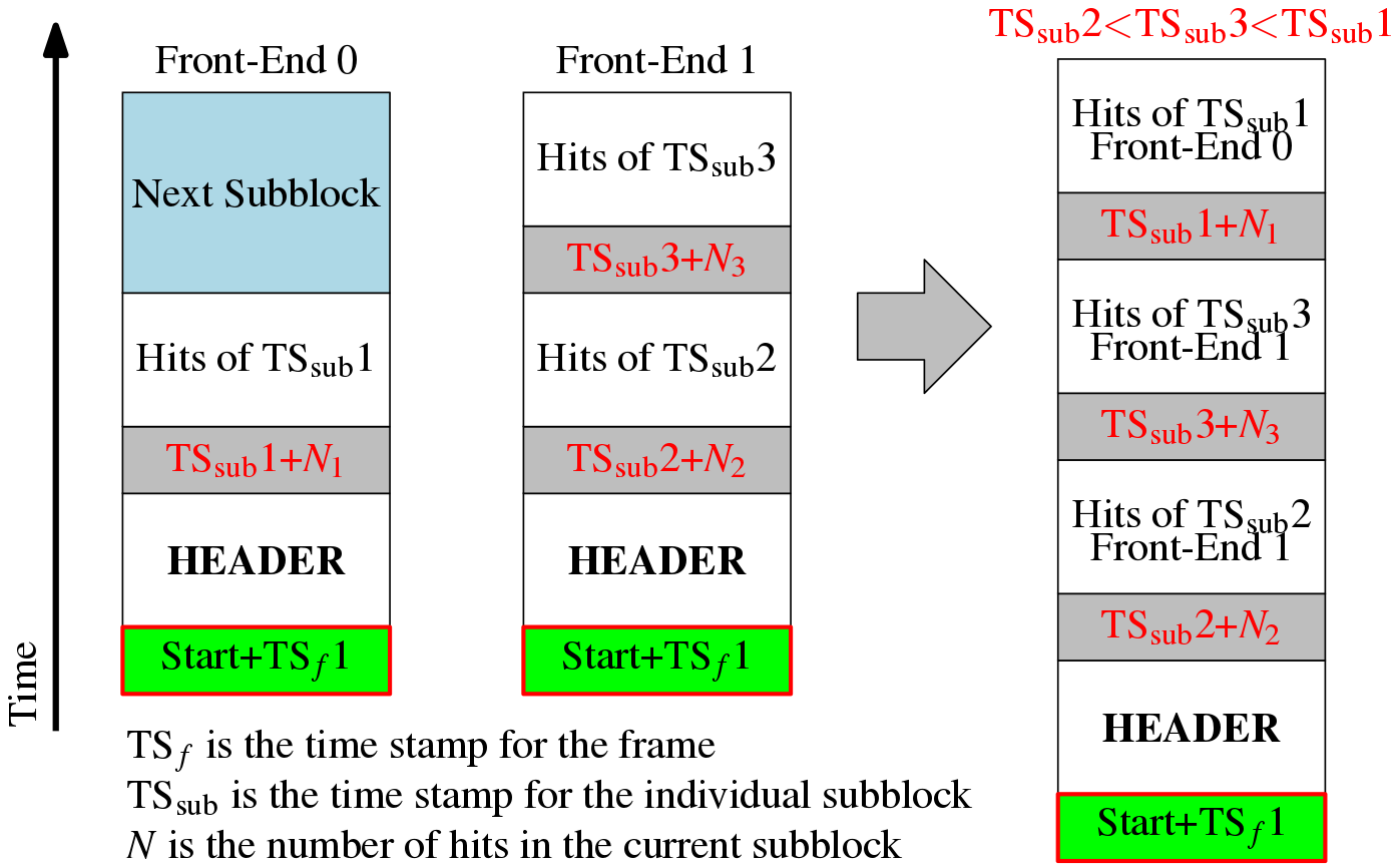
- Check the “Time Stamp” of the subblocks by round-robin
- If multiple subblocks have same “Time Stamp” then merge



TS_f is the time stamp for the frame
TS_{sub} is the time stamp for the individual subblock
N is the number of hits in the current subblock

Switching FPGA : Merge algorithm on VHDL

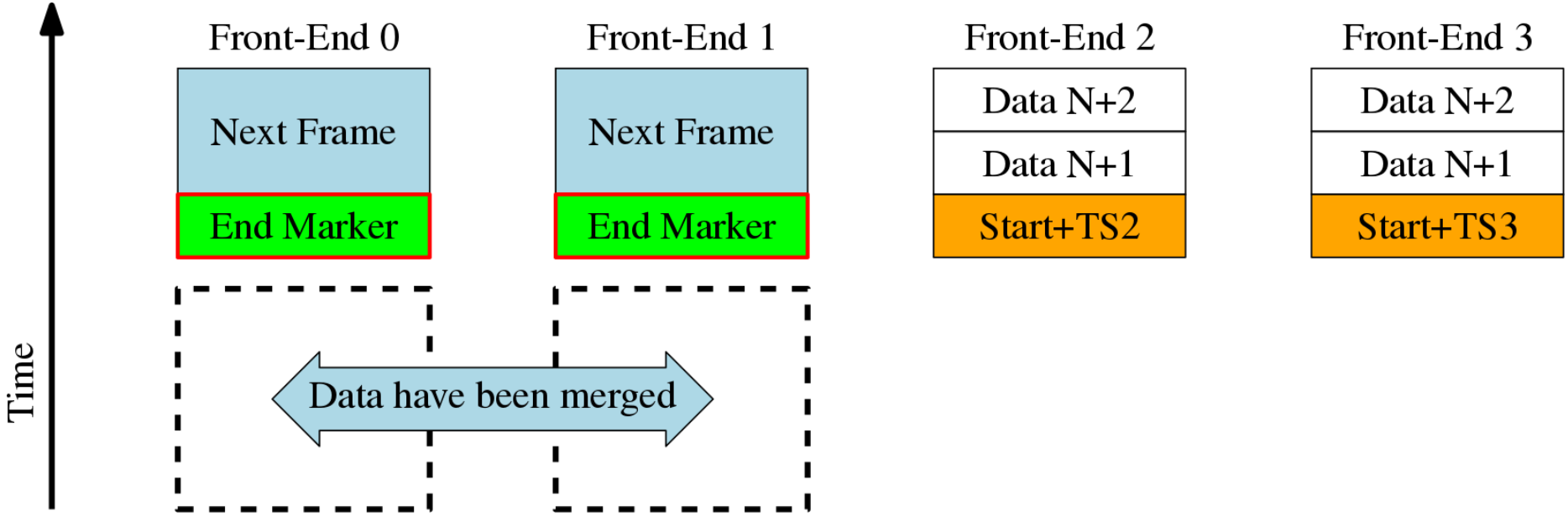
- Check the “Time Stamp” of the subblocks by round-robin
- If only one subblock has the minimal “Time Stamp” then copy



TS_f is the time stamp for the frame
TS_{sub} is the time stamp for the individual subblock
N is the number of hits in the current subblock

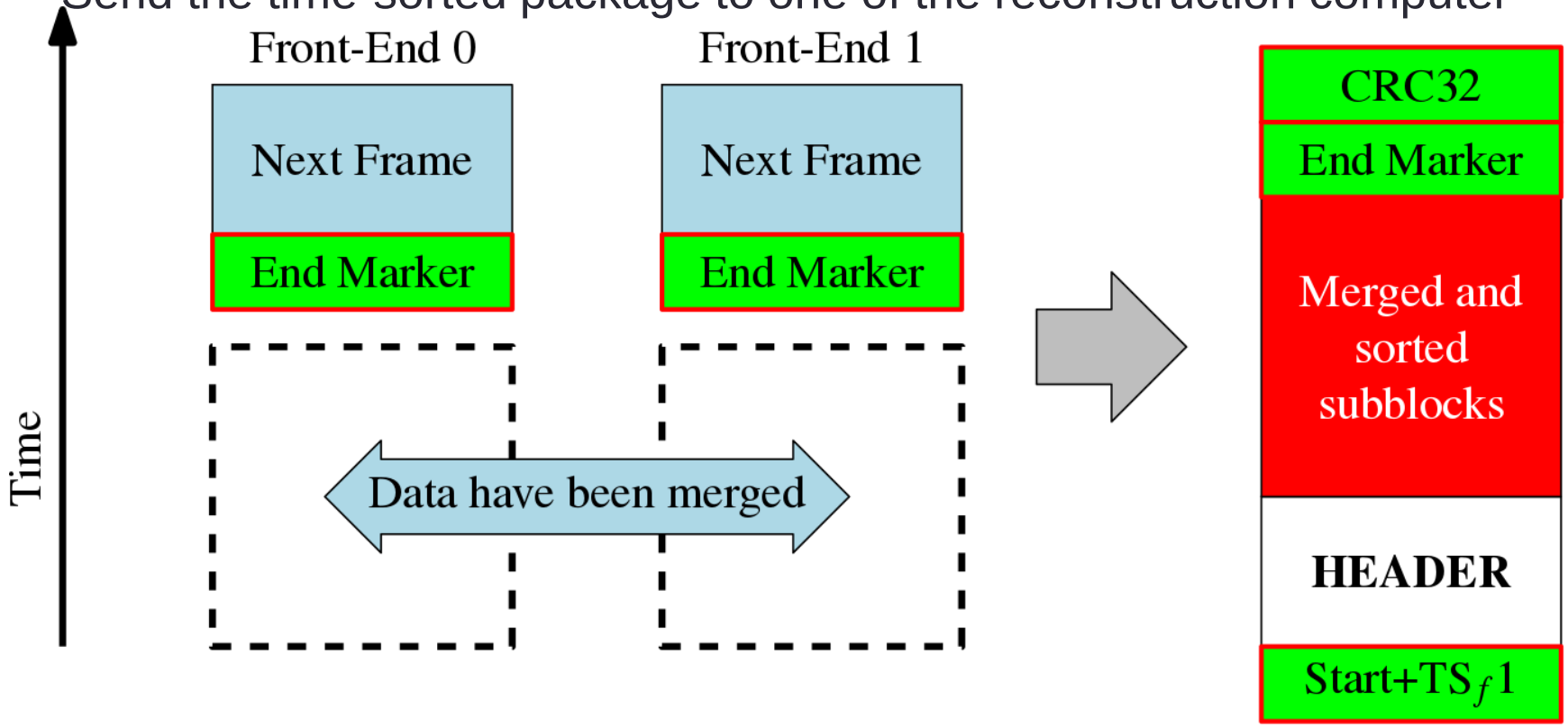
Switching FPGA : Merge algorithm on VHDL

- Repeat the above until all merging channels have the end marker
- Add CRC32 checksum and finish merging the frames with same “Frame Time Stamp”
- Send the time-sorted package to one of the reconstruction computer



Switching FPGA : Merge algorithm on VHDL

- Repeat the above until all merging channels have the end marker
- Add CRC32 checksum and finish merging the frames with same “Frame Time Stamp”
- Send the time-sorted package to one of the reconstruction computer



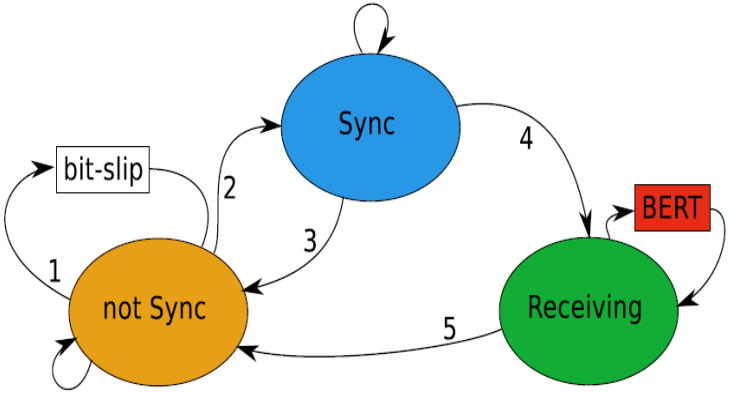
Summary

- Firmware development is ongoing, first prototype is available
- Bit Error Rate Tests are done for required speed, and prove that the optical links fulfill the experiment
- The preliminary protocol and algorithm should meet the speed requirement for the Mu3e experiment about O(25Gbps)

BACKUP

Fast optical link BERT : Firmware and software

- Firmware written by VHDL, with PRBS generators
- 8b/10b encoded, DC balance
- Altera® Quartus II 14.1
- Logic simulation : Modelsim 10.1
- Alternative : Altera Receiver Toolkit
  - Auto Sweep
  - EyeQ (support for Stratix IV up to 6 Gbps)
  - This toolkit allows BER test and transceiver tuning without external devices



Switching FPGA : a preliminary protocol

- Send Idle pattern to establish connection
- CRC32 checksum ensures error-free
- Easy to merge into a bigger packet in readout FPGA
- Same/Reduced overhead after merging the subframes with same block counter