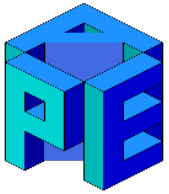


A FPGA-based Network Interface Card with GPUDirect enabling real-time GPU computing in HEP experiments.

Alessandro Lonardo
(INFN Roma1)
On behalf of the NaNet collaboration

GPU Computing in High Energy Physics
Pisa 10-12 September 2014

GPUs in HEP Low Level Triggers

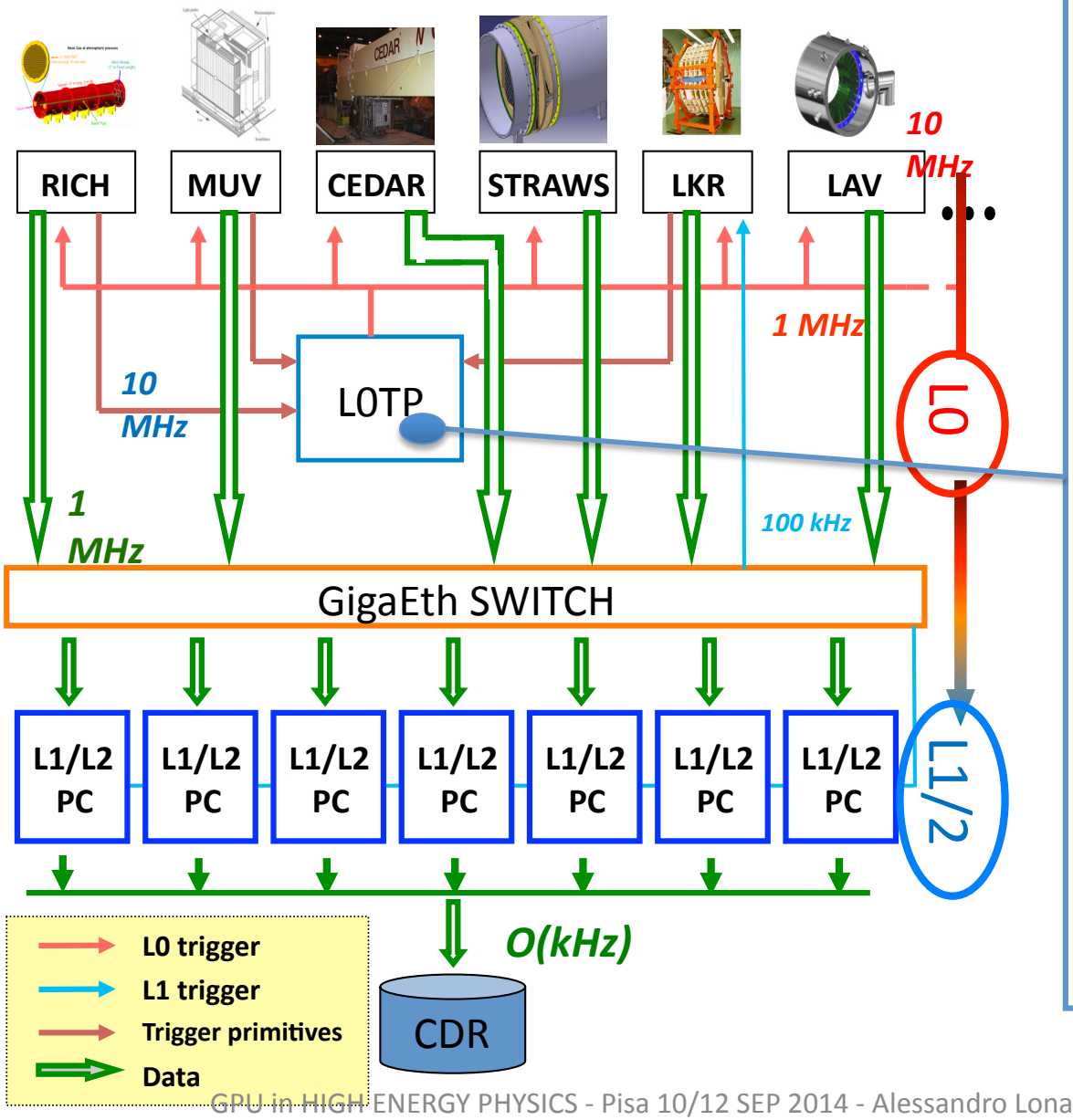
- Hard real-time constraints, system has to respond strictly within a given time budget to avoid data loss.
- System Throughput: have GPUs enough computing power to take trigger decision at tens of MHz events rate?

  Yes. (Choosing the appropriate algorithm and number of GPUs in the system…)
- System Latency:
  - Is GPUs processing latency small and stable enough?

    Yes. (Once data are available in their own internal memories…)
  - Is the latency of data communication from read-out systems to GPUs memories small and stable enough?

    In our experience this is the most critical task in terms of latency fluctuations of whole system.

Using GPUs in the NA62 L0 Trigger



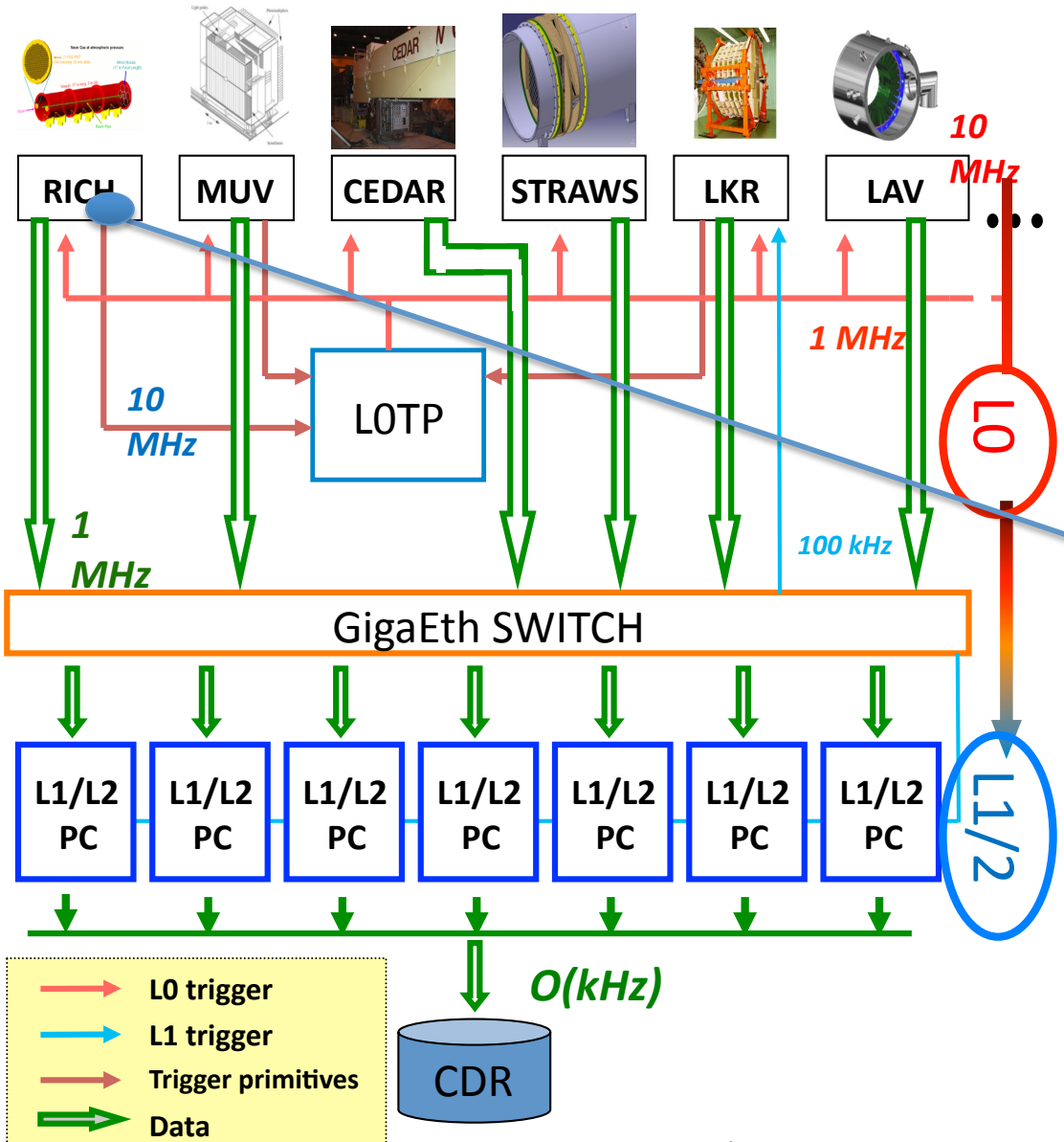
L0 trigger:

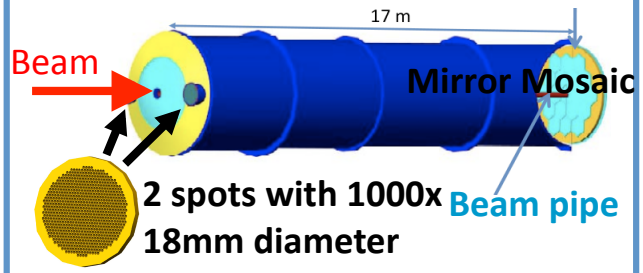
- FPGA custom-design
- 10 MHz to 1 MHz
- Max latency 1 ms

Replace custom hardware with a GPU-based system performing the same task but:

- Programmable
- Upgradable
- Scalable
- Cost effective
- Exploit GPUs computing power to implement more selective trigger algorithms.

Using GPUs in the NA62 L0 Trigger: the RICH Detector Case Study

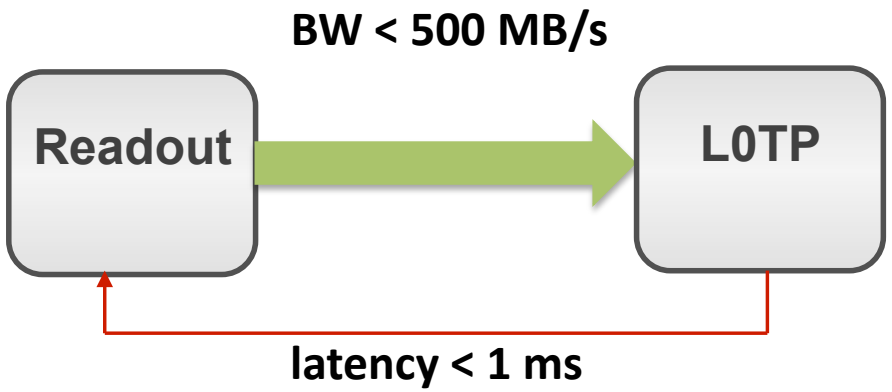




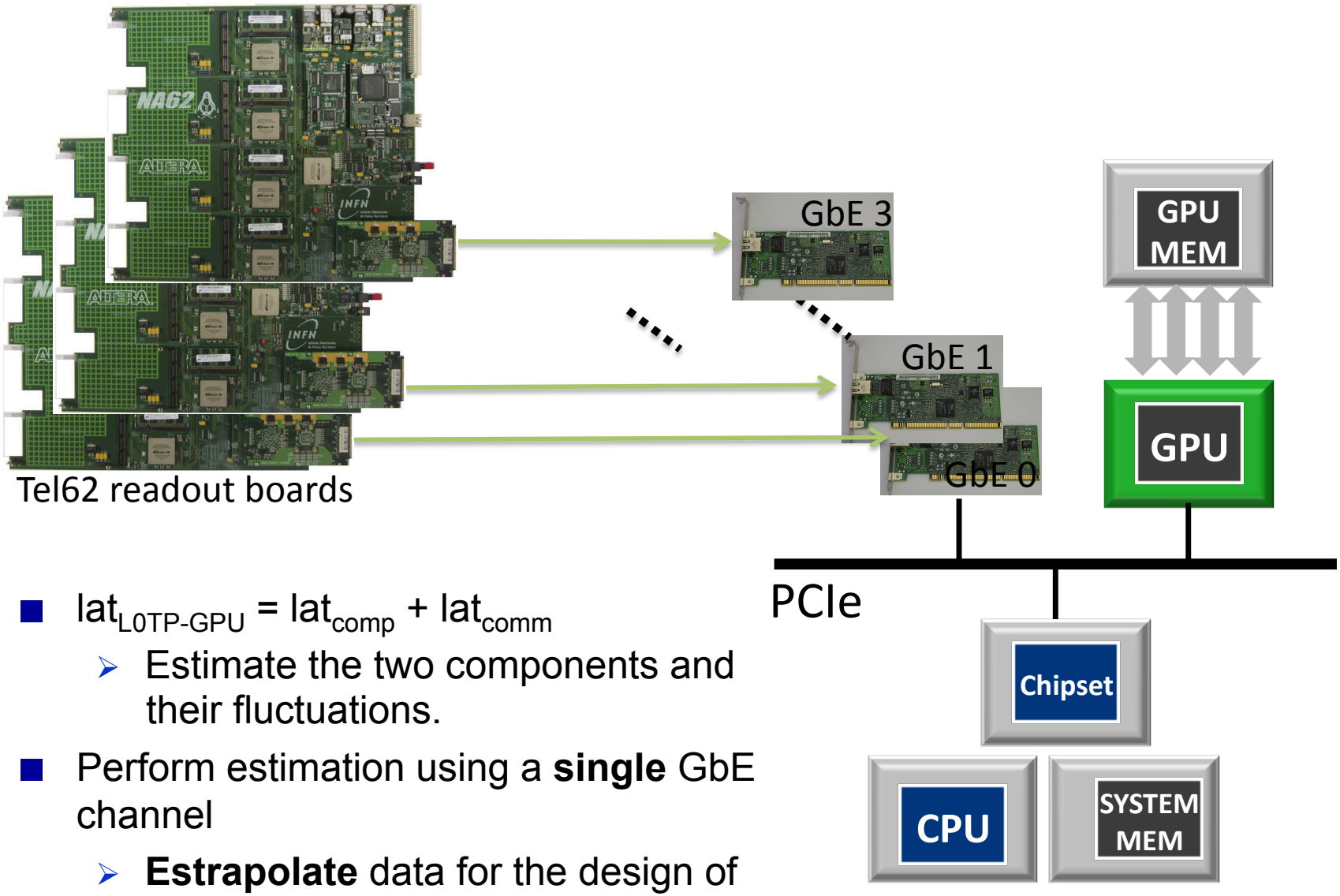
- Light focused by two mirrors on the two spots.
- **100 ps** time resolution.
- **10 MHz** event rate.
- **~20 hits** per ring on average.
- **36 byte** per event (max 32 hits per event).
- Computing on GPU: rough detection of particle speed (radius) and direction (centre) through matching of partial circular hit patterns.

NA62 RICH L0 Trigger Processor Requirements

- Network Protocol: UDP on GbE channels (4).
- System Throughput: 10 M events/s
  - Network bandwidth < 500 MB/s
- System response latency < 1 ms
  - determined by the size of Readout Board memory buffer storing event data to be passed to higher trigger levels.



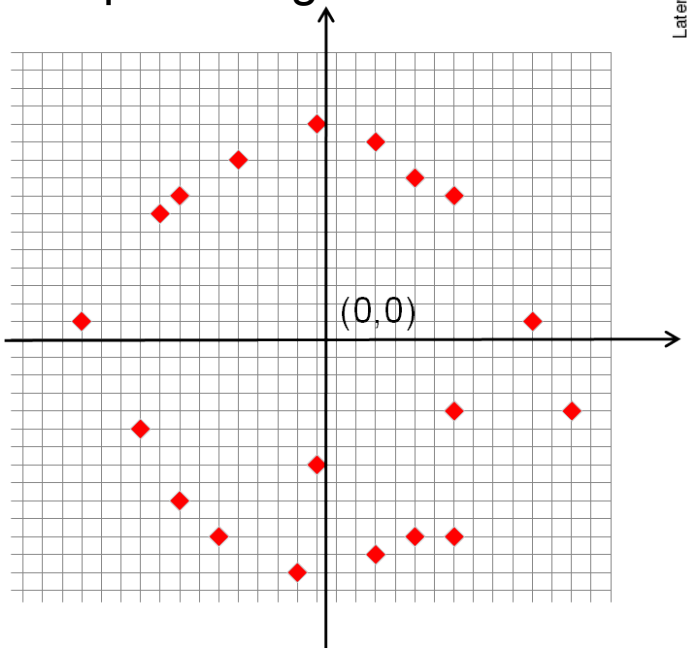
NA62 GPU-Based RICH Level 0 TP System Latency Estimation

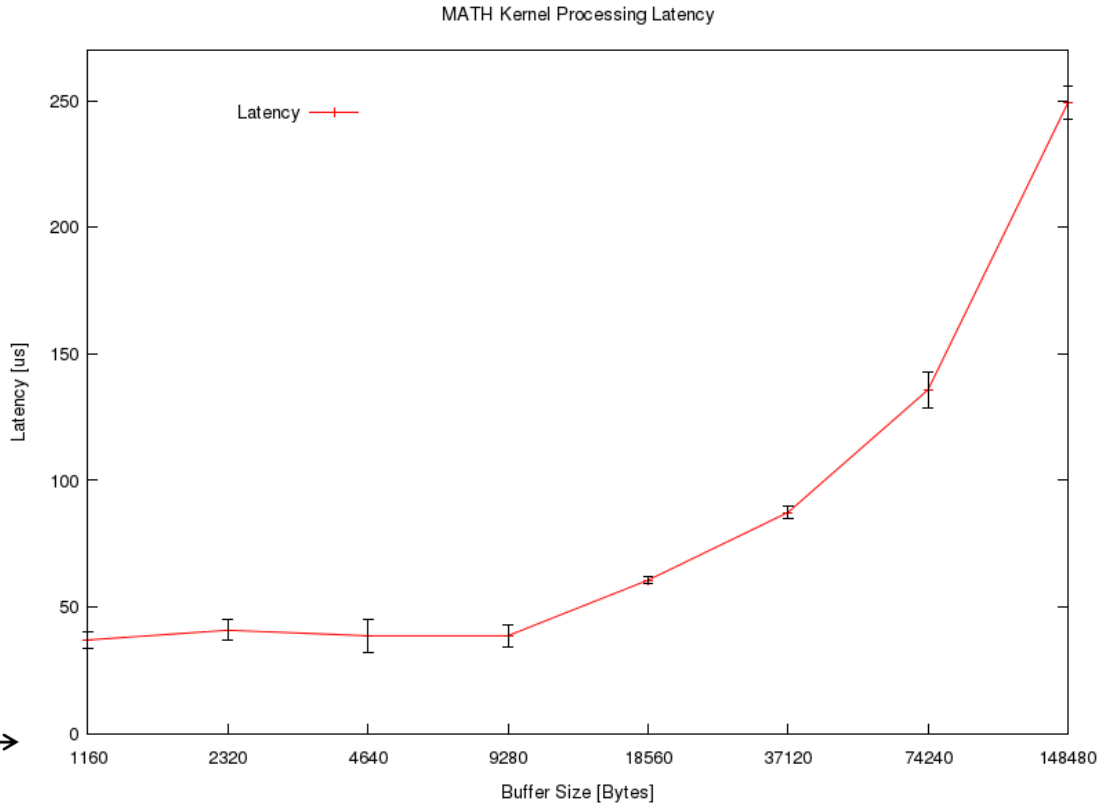


- $lat_{L0TP-GPU} = lat_{comp} + lat_{comm}$
  - Estimate the two components and their fluctuations.
- Perform estimation using a **single** GbE channel
  - **Estrapolate** data for the design of the “full bandwidth” system.

System Latency Estimation Processing Latency

- **lat_{comp}** : time needed to perform rings pattern-matching on the GPU with input and output data on device memory using the **MATH** least squares algorithm.

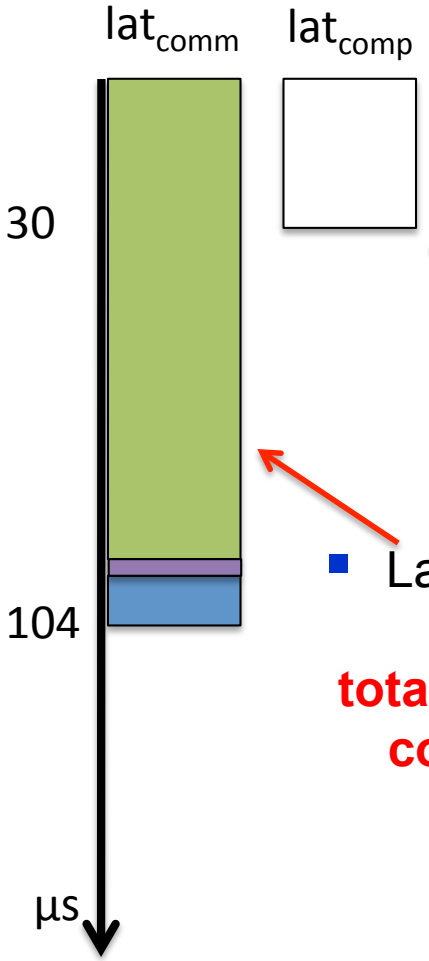




Communication Latency
Standard GbE NIC / SW Stack

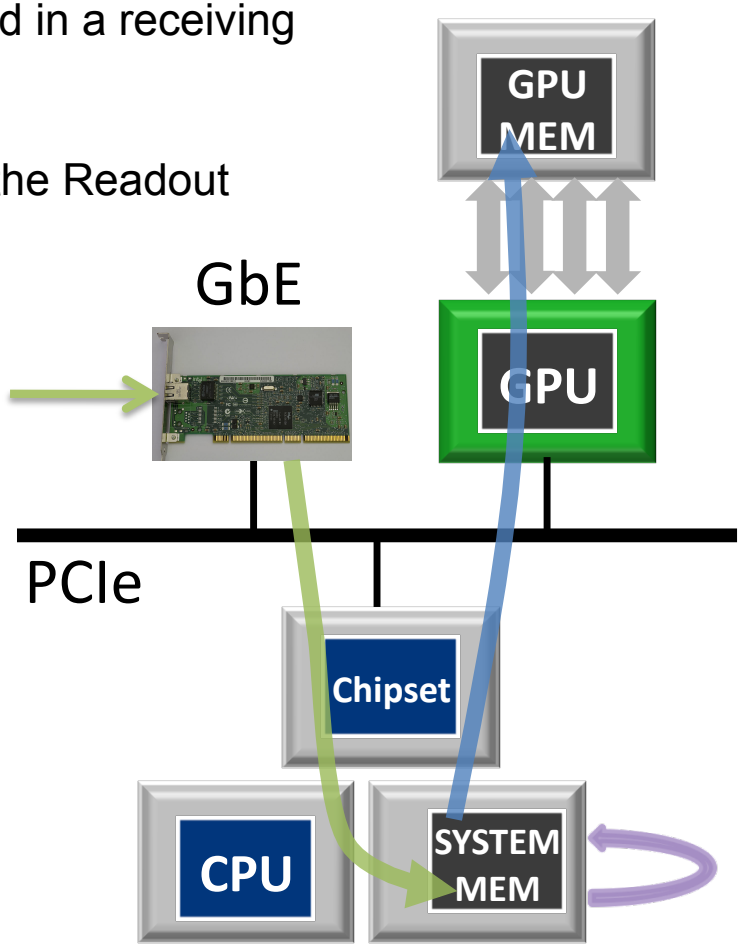
lat_{comm} : time needed to receive input event data from GbE NIC to GPU memory.

- 40 events data (1400 bytes) sent from Readout board to the GbE NIC are stored in a receiving GPU buffer.
- T=0 start of send operation on the Readout Board.

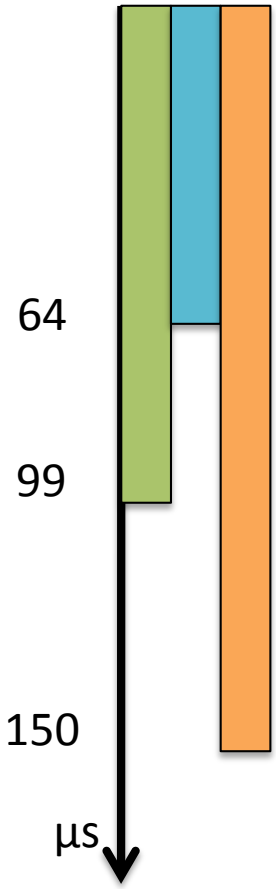


- Lat_{comm} = 104 us ≅ 3.5 x Lat_{comp}

total latency is dominated by communication component.



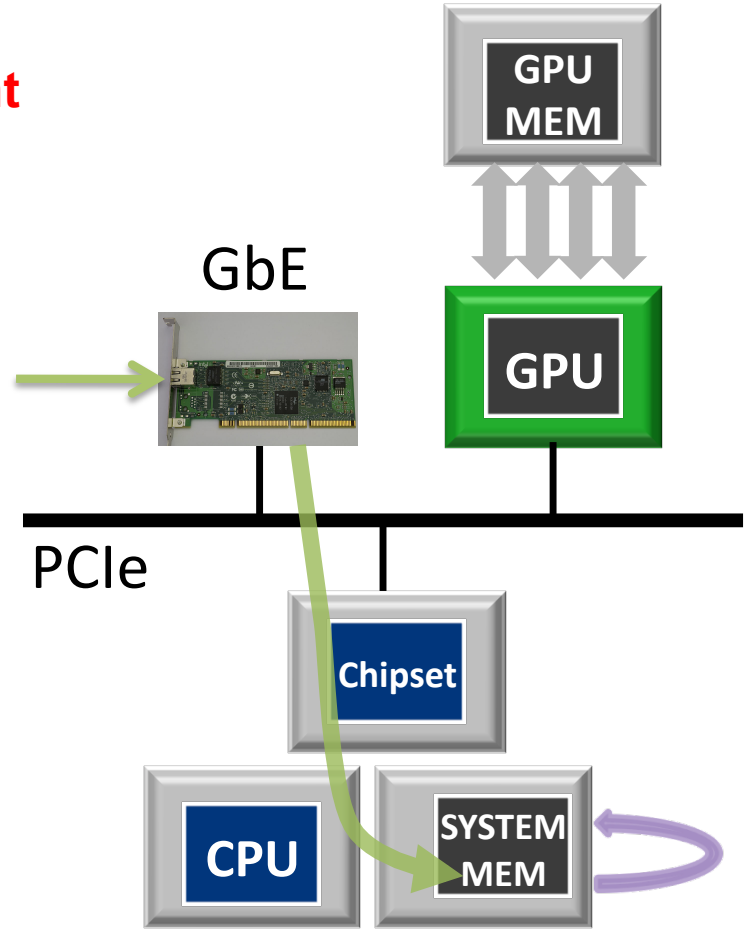
Standard GbE NIC / SW Stack Communication Latency Fluctuations



- Fluctuations on the **GbE component** of lat_{comm} may hinder the real-time requisite of the system.

```
sockperf: Summary: Latency is 99.129 usec
sockperf: Total 100816 observations; each
percentile contains 1008.16 observations

sockperf: ---> <MAX> observation = 657.743
sockperf: ---> percentile 99.99 = 474.758
sockperf: ---> percentile 99.90 = 201.321
sockperf: ---> percentile 99.50 = 163.819
sockperf: ---> percentile 99.00 = 149.694
sockperf: ---> percentile 95.00 = 116.730
sockperf: ---> percentile 90.00 = 105.027
sockperf: ---> percentile 75.00 = 97.578
sockperf: ---> percentile 50.00 = 96.023
sockperf: ---> percentile 25.00 = 95.775
sockperf: ---> <MIN> observation = 64.141
```



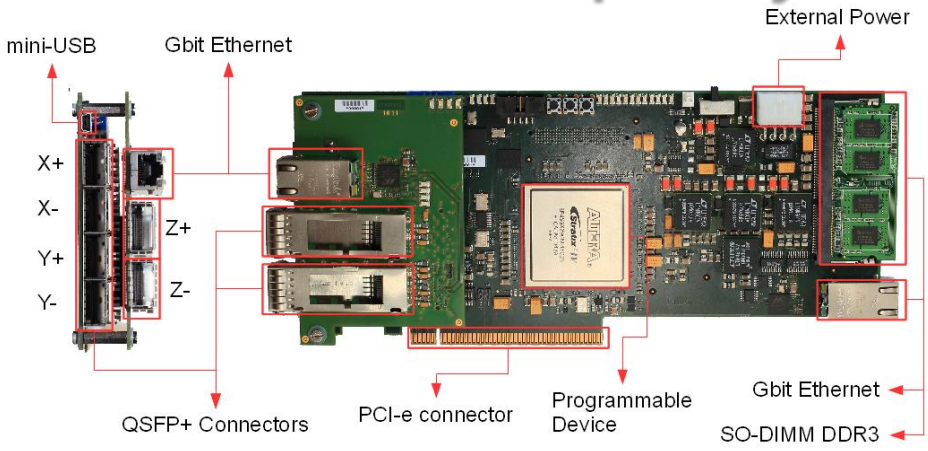
NaNet Design

- Challenge:
  - Lower data communication task latency and its fluctuations.
- How?
  1. Injecting directly data from the NIC into the GPU memory **without intermediate buffering.**
  2. **Offloading** the CPU from network stack protocol management, eliminating possible OS jitter effects.
- **NaNet design**:
  - Re-use the part of the **APEnet+** design, implementing a **PCIe** interface with **GPUDirect P2P/RDMA** capability.
  - Support **multiple link** technologies and network protocols.
  - Provide a network stack protocol management unit (e.g. **UDP Offloading Engine**).
  - Use FPGA resources to perform **processing on data stream** (e.g. reformat data on-the-fly in a GPU-friendly fashion).

APEnet+: a 3D NIC for HPC with GPUdirect P2P/RDMA capability

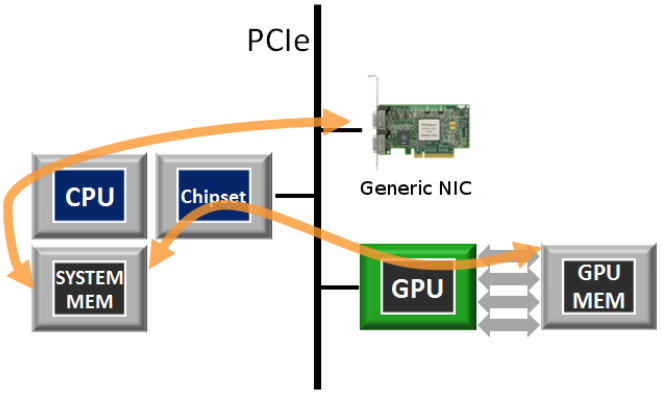
APEnet+ Card:

- FPGA based (ALTERA EP4SGX290)
- PCI Express x16 slot, x8 gen2 signaling
- Fully bidir 3D torus links, 34 Gbps/channel



APEnet+ Logic:

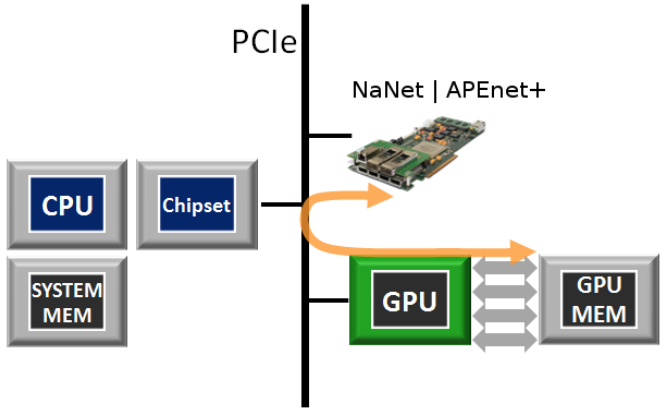
- Network Interface
  - NIOS II 32 bit uP
  - RDMA engine
  - Virtual Memory Management support
  - GPU I/O accelerator.



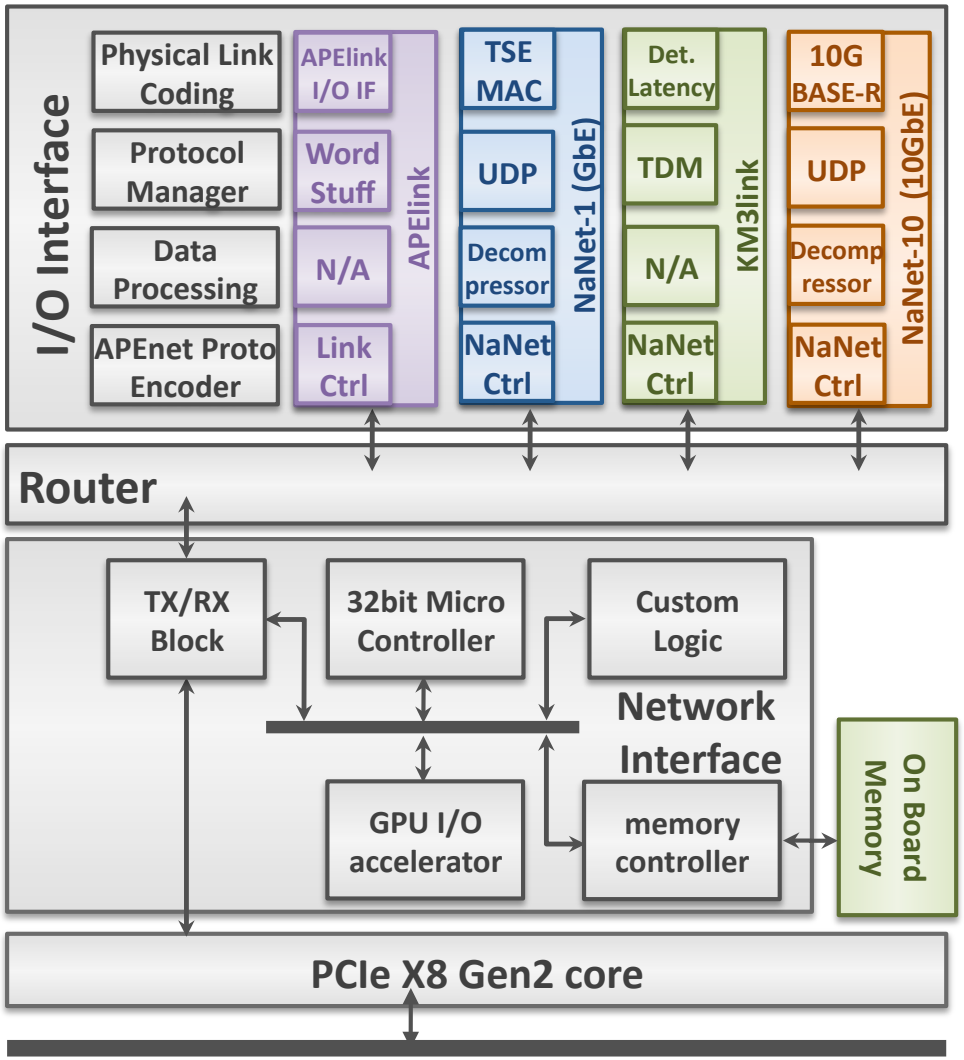
GPUDirect P2P (and RDMA)

- PCIe P2P protocol between Nvidia Fermi/ Kepler devices and APEnet+
- GPUDirect P2P allows direct data exchange on the PCIe bus with no CPU involvement

→ *Latency reduction for small messages*

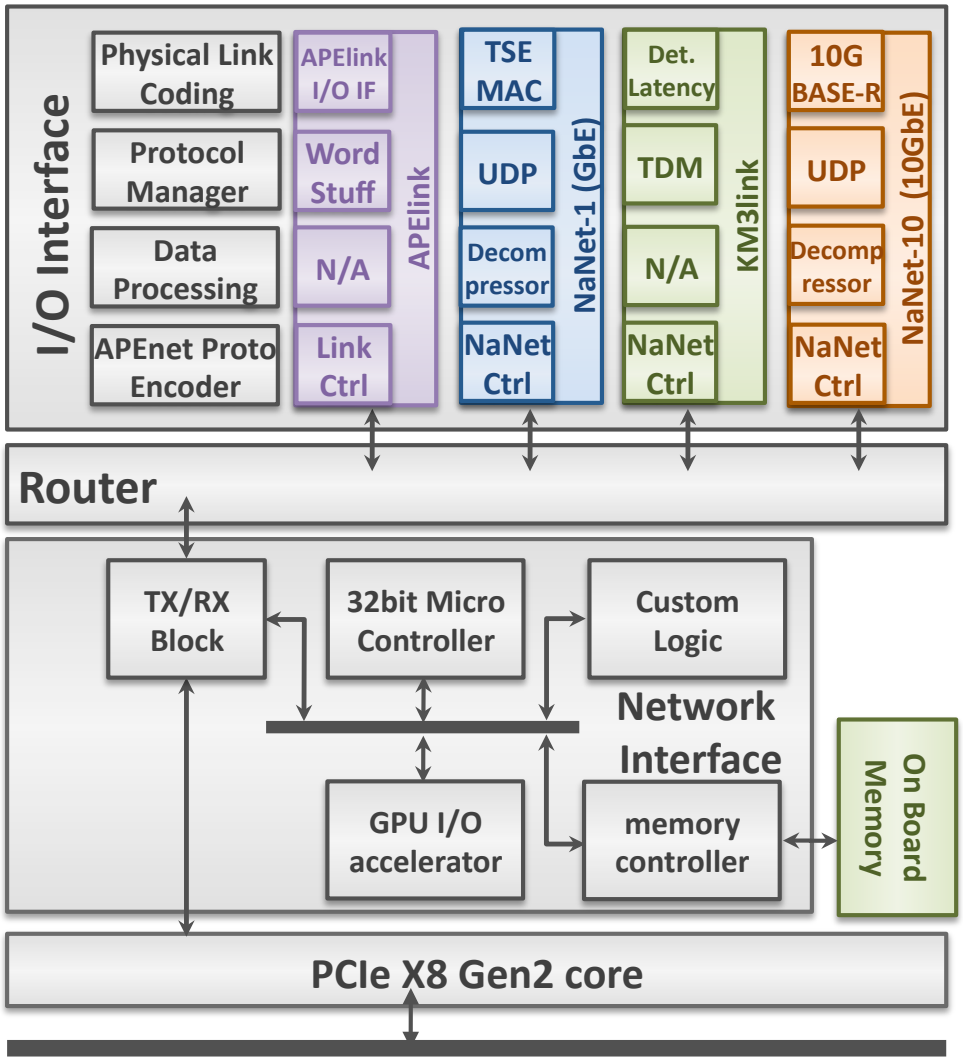


NaNet Design – I/O Interface



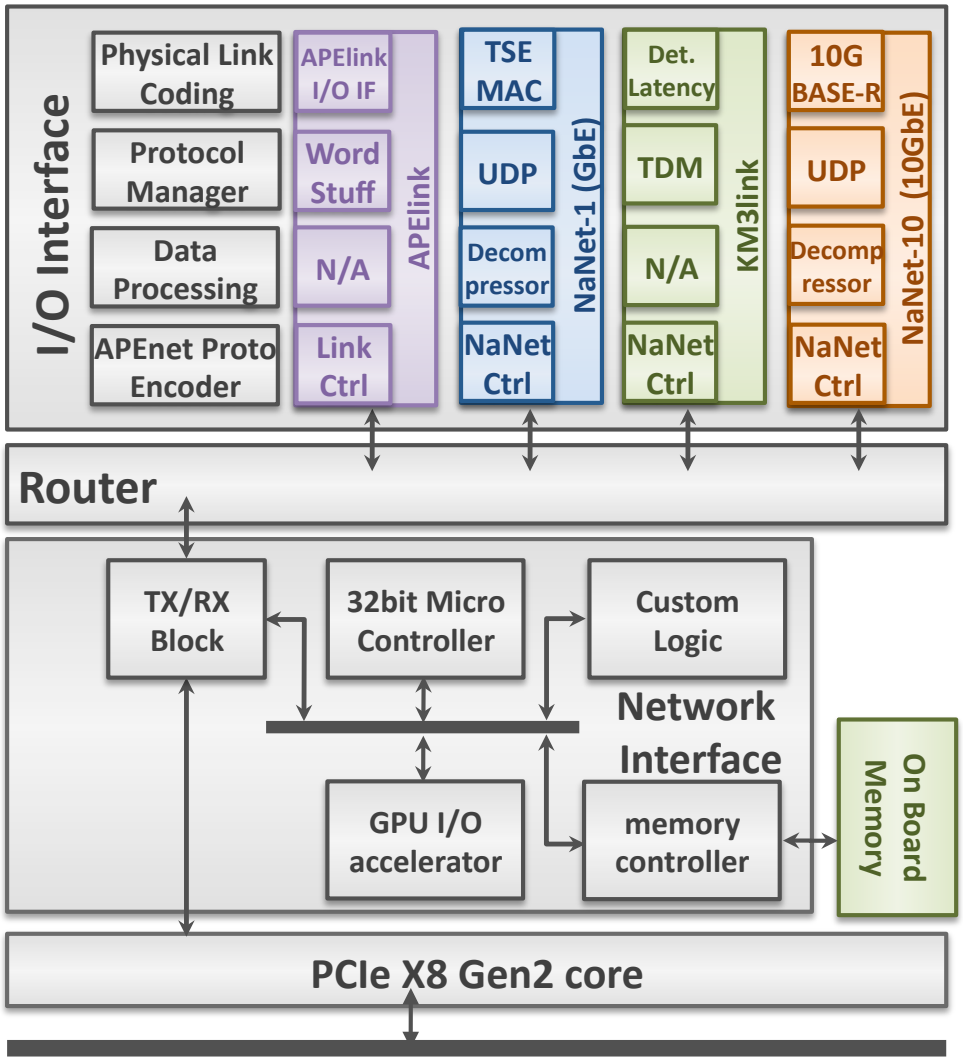
- Physical Link Coding
  - Standard: 1GbE (1000Base-T), 10GbE (10Base-R).
  - Custom: APElink (20 Gb/s QSFP), KM3link Det. Lat. (2.5 Gb/s optical).
- Protocol Manager
  - Data/Network/Transport layers off-loader module.
  - UDP, Time Division Multiplexing.
- Data Processing
  - Application dependent processing on data stream.
  - NA62 Decompressor: re-format event data (data size and alignment).
- APEnet Protocol Encoder
  - protocol adaptation between on-board and off-board protocol.
  - In RX path generates CPU/GPU destination memory addresses for incoming pkts.

NaNet Design – Router



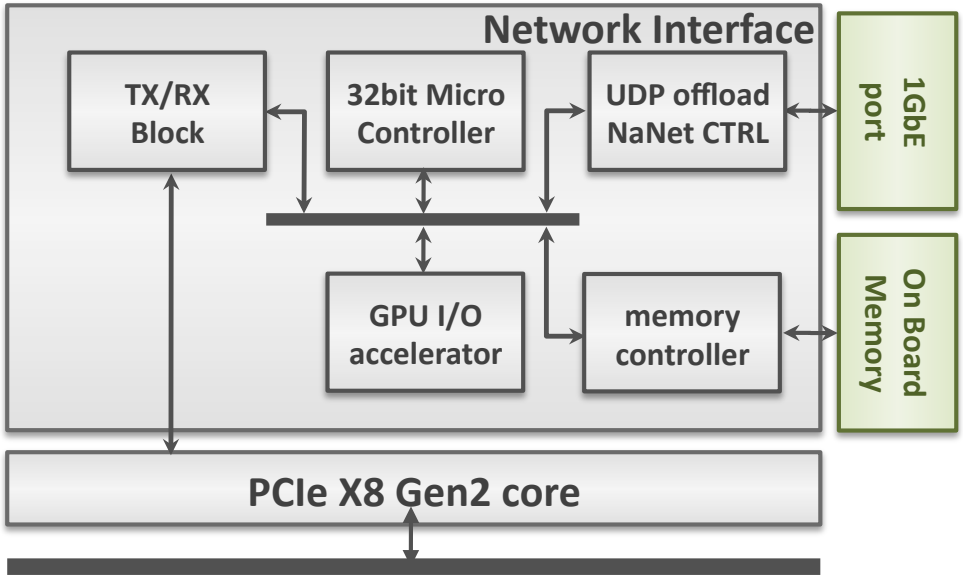
- 5 ports full crossbar switch
  - Router: dynamically interconnects ports.
  - Arbitration: resolve contention on ports requests (static, round robin).
  - 2.8 GB/s data streams.

NaNet Design – Network Interface

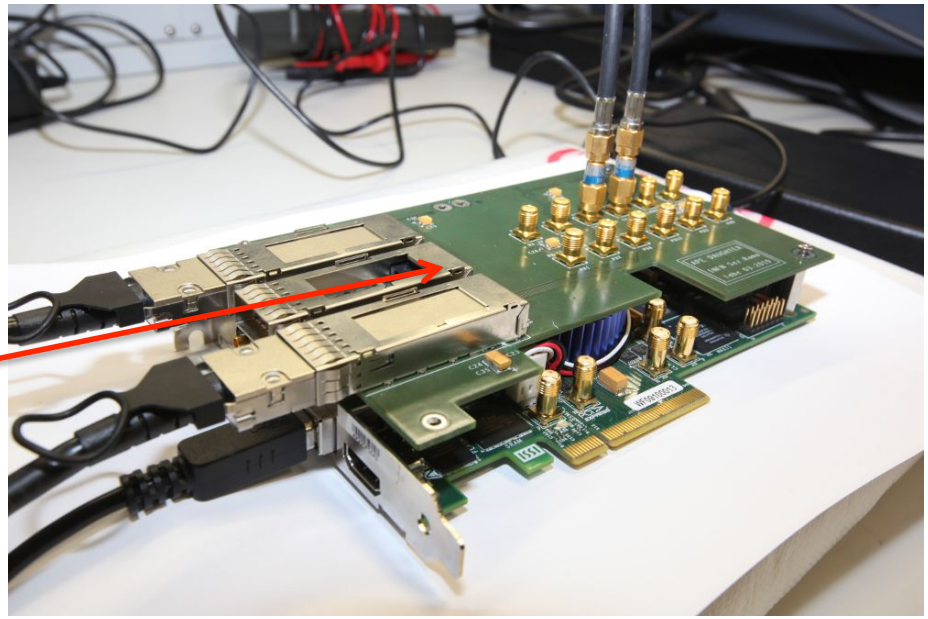


- In TX gathers data from PCIe interface and forwards them to destination port.
- In RX provides support for RDMA receive operation managing Virtual to Physical address translation.
  - CPU/GPU receive buffers addresses virtual to physical mapping.
  - Translation Lookaside Buffer (associative cache).
  - Microcontroller in case of miss.
- GPU I/O accelerator (P2P protocol)
- TX/RX Block
  - DMA engines, PCIe transactions.
- Altera NIOS II microcontroller.
- External memory controller
- Custom Logic (application specific)
- PLDA based PCIe X8 Gen2 Core

NaNet-1 Implementation

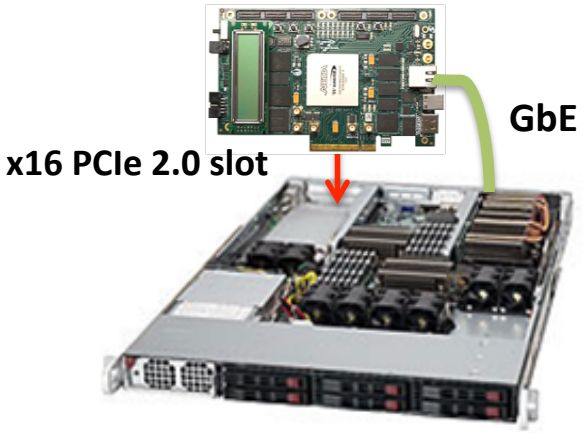


- Implemented on Altera Stratix IV dev board (EP4SGX230KF40C2)
- GbE PHY Marvell 88E1111
- Supports additional 3 APElink channels (20 Gb/s each) with HSMC daughtercard

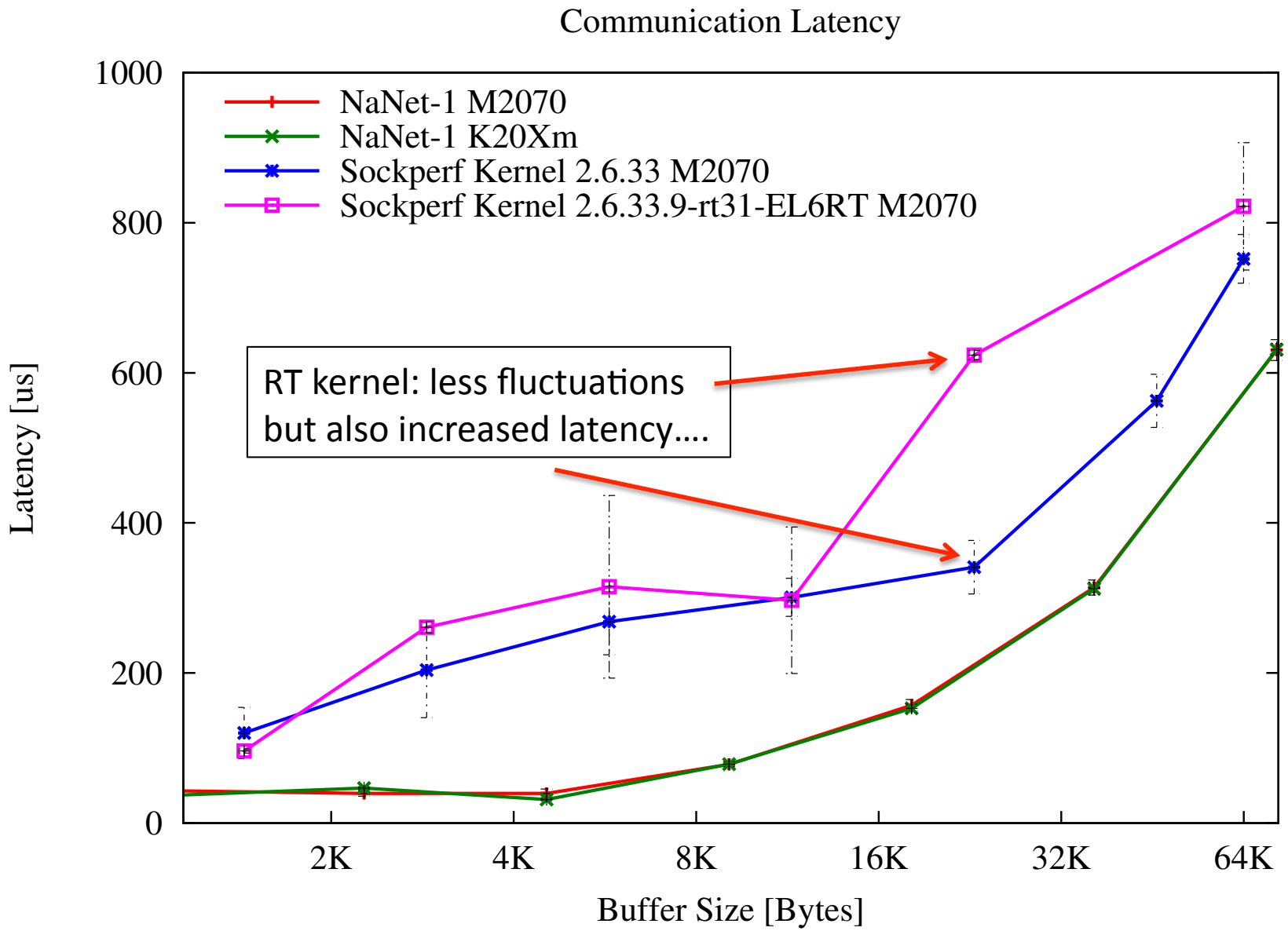


NaNet-1 Test & Benchmark Setup

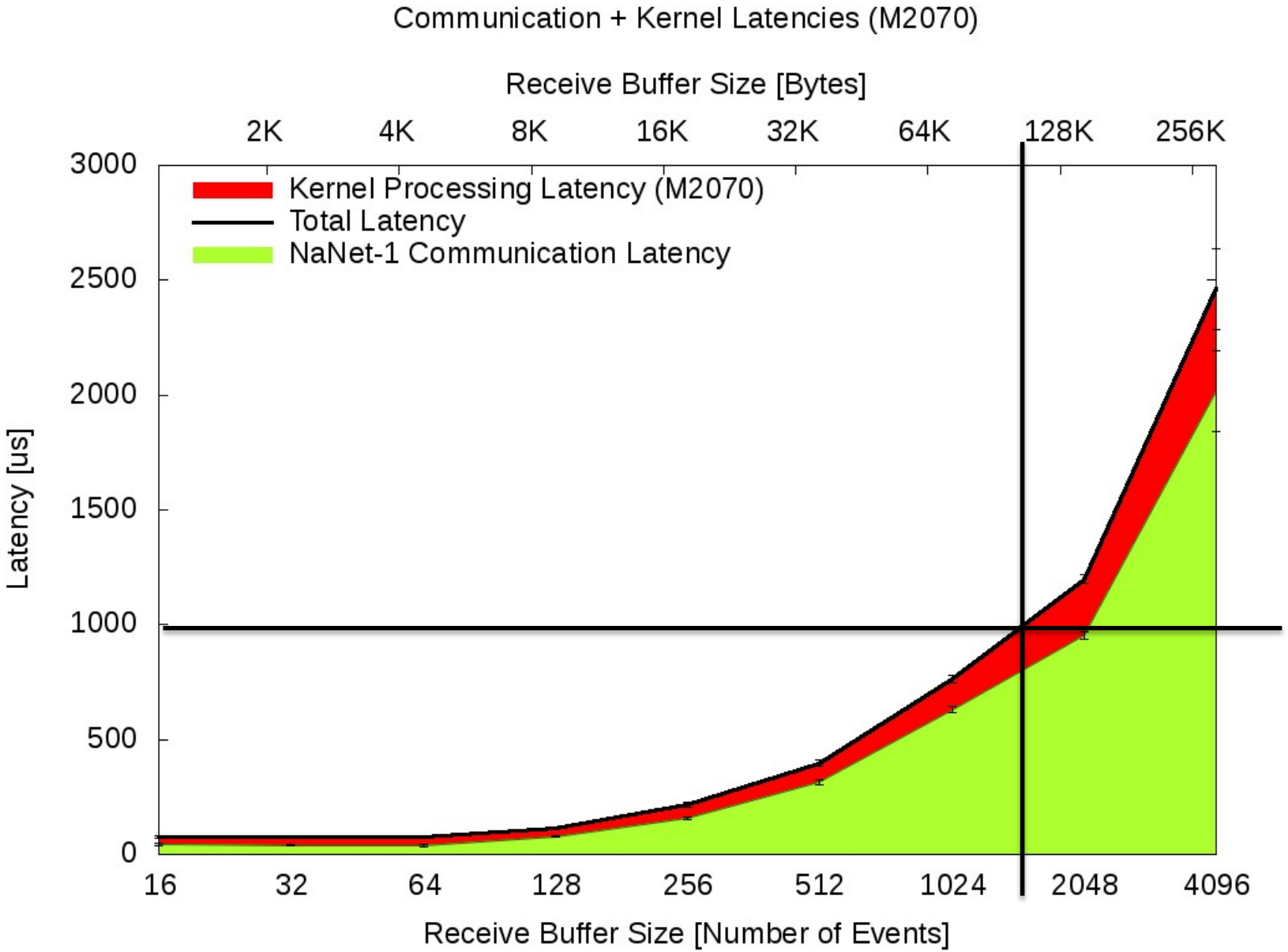
- Supermicro SuperServer 6016GT-TF
  - X8DTG-DF motherboard (Intel Tylersburg chipset)
  - dual Intel Xeon E5620
  - Intel 82576 Gigabit Network Connection
  - Nvidia Fermi M2070 and Kepler K20Xm
  - kernel 2.6.32-358.6.2.el6.x86_64, CUDA 4.2, Nvidia driver 325.15.
- NaNet-1 board in x16 PCIe 2.0 slot
- NaNet-1 GbE interface directly connected to one host GbE interface
- Common time reference between sender and receiver (they are on the same host).
- Ease data integrity tests.



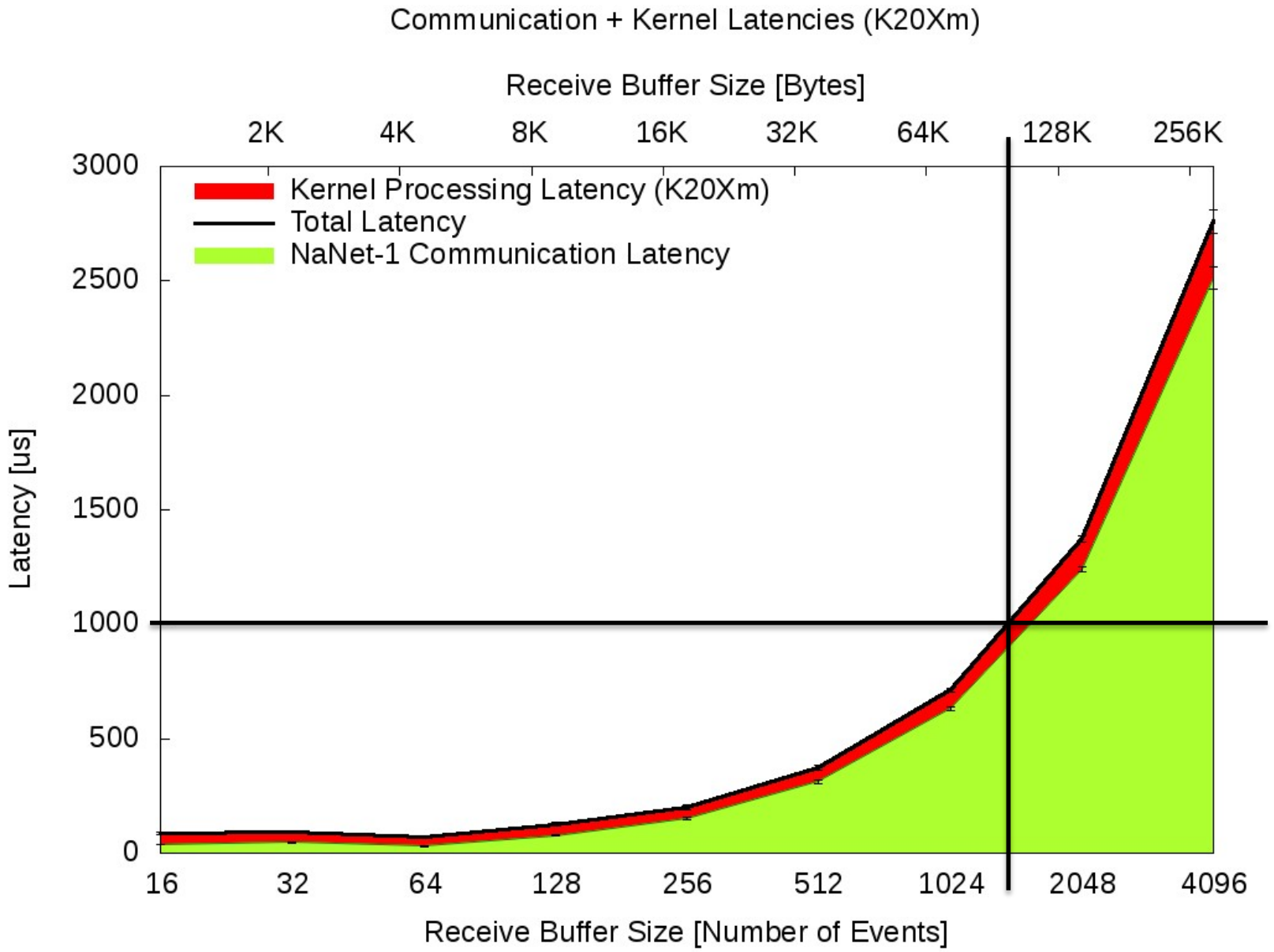
NaNet-1 Communication Latency



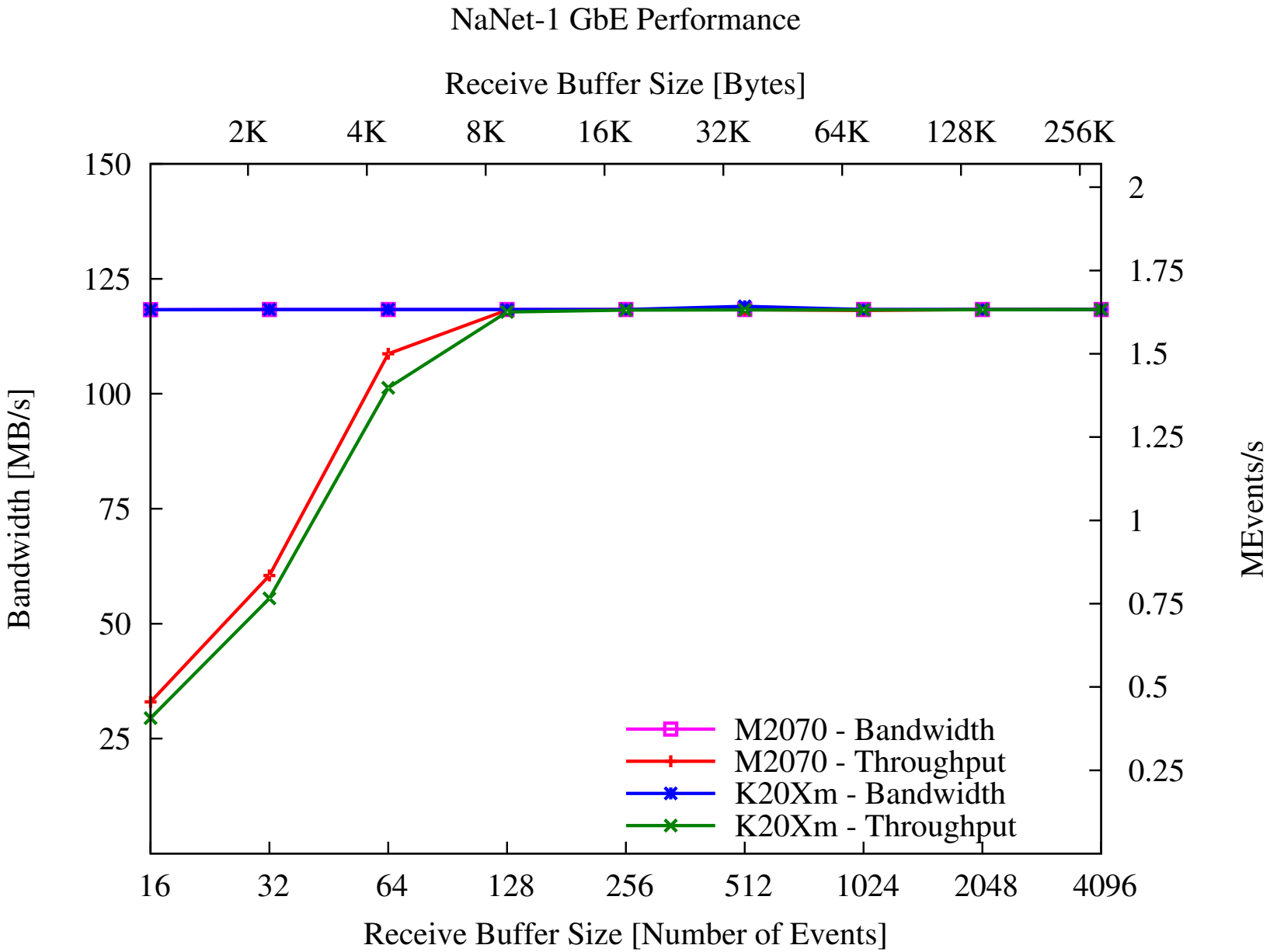
Total latency of the GPU-Based RICH L0-TP using NaNet-1 (Fermi M2070)



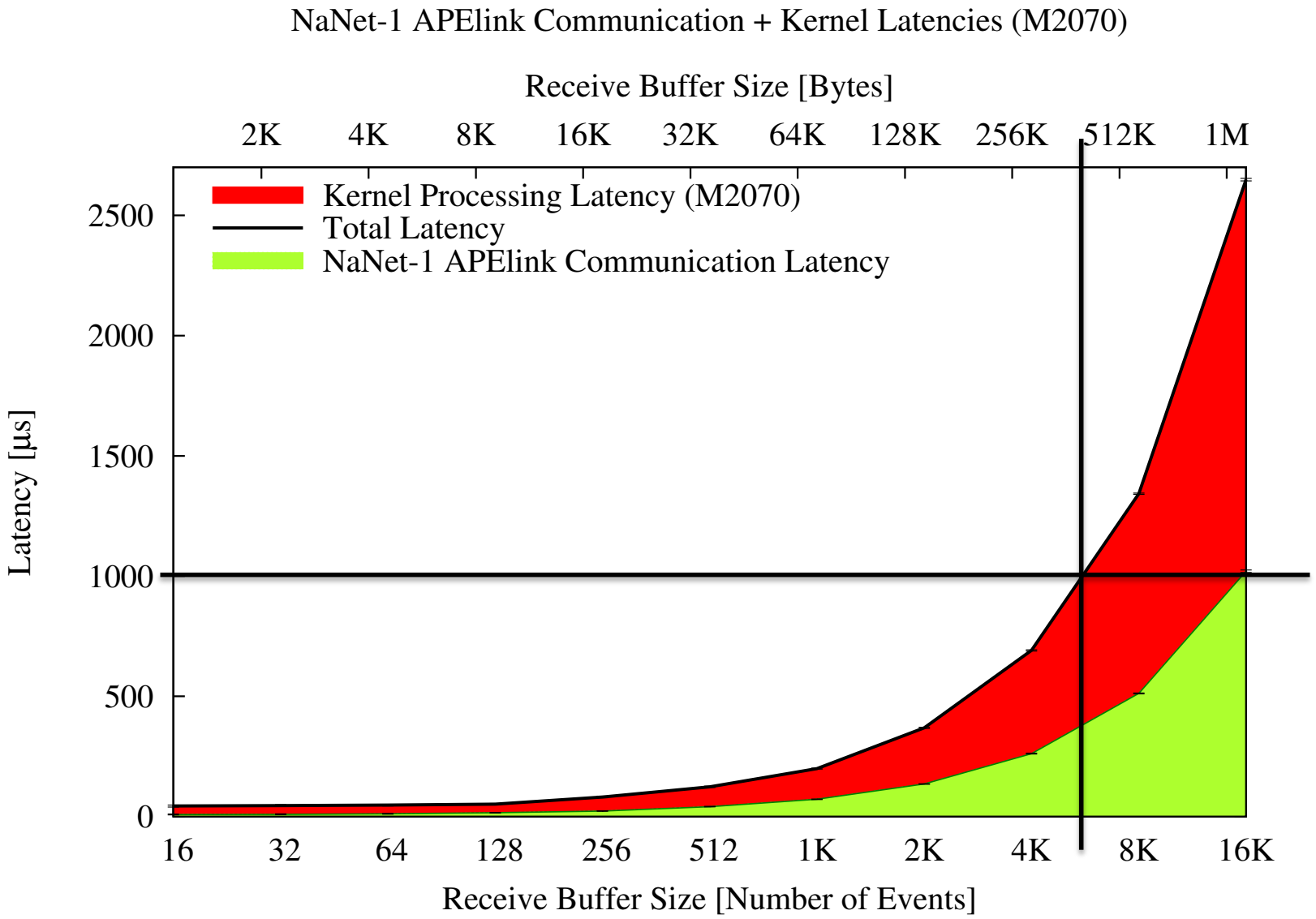
Total latency of the GPU-Based RICH L0-TP using NaNet-1 (Kepler K20X)



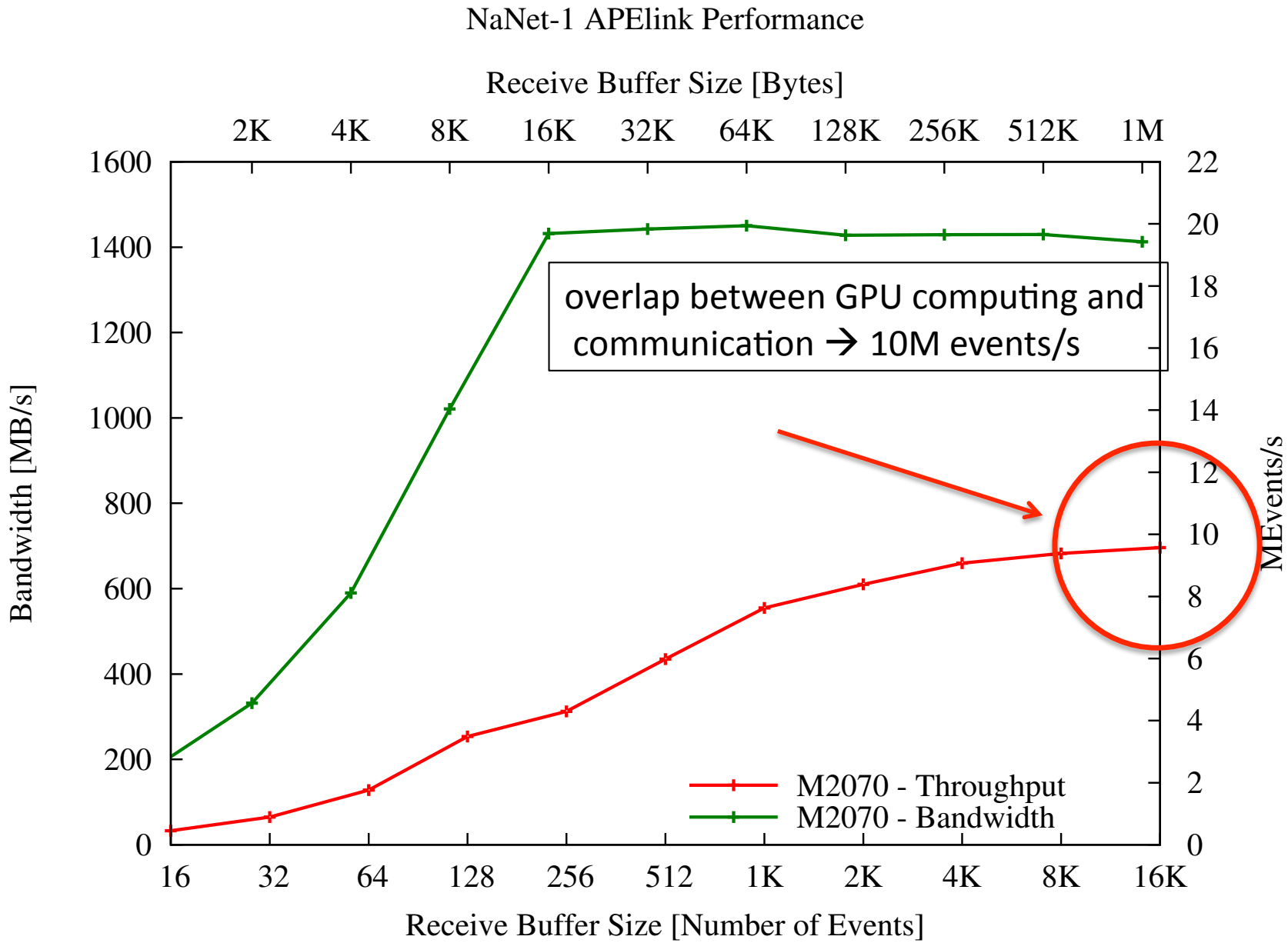
NaNet-1 1 GbE Bandwidth & Throughput



Total latency of the GPU-Based RICH L0-TP using NaNet-1 (APElink)



NaNet-1 APElink Bandwidth & Throughput



Total latency of the GPU-Based RICH L0-TP using NaNet-1 - Comments

- Note:
  - we performed latency measures in worst case conditions…
    - communication and processing tasks were serialized
  - while, in normal operation, data communication and GPU processing are overlapped
    - Circular list of receive buffers: when one of the buffers is full, the GPU kernel is launched, data arriving from the network are accumulated in the next buffer in the circular list concurrently with processing.
- Summary:
  - A RICH detector Level 0 Trigger Processor GPU-based system using NaNet-1 (1 GbE link) performing the simple MATH processing is able to sustain a throughput of **~1.6M events/s** with *real-time behaviour*.
  - Using a single APElink channel instead of the GbE one, the system shows a throughput of **~10M events/s**

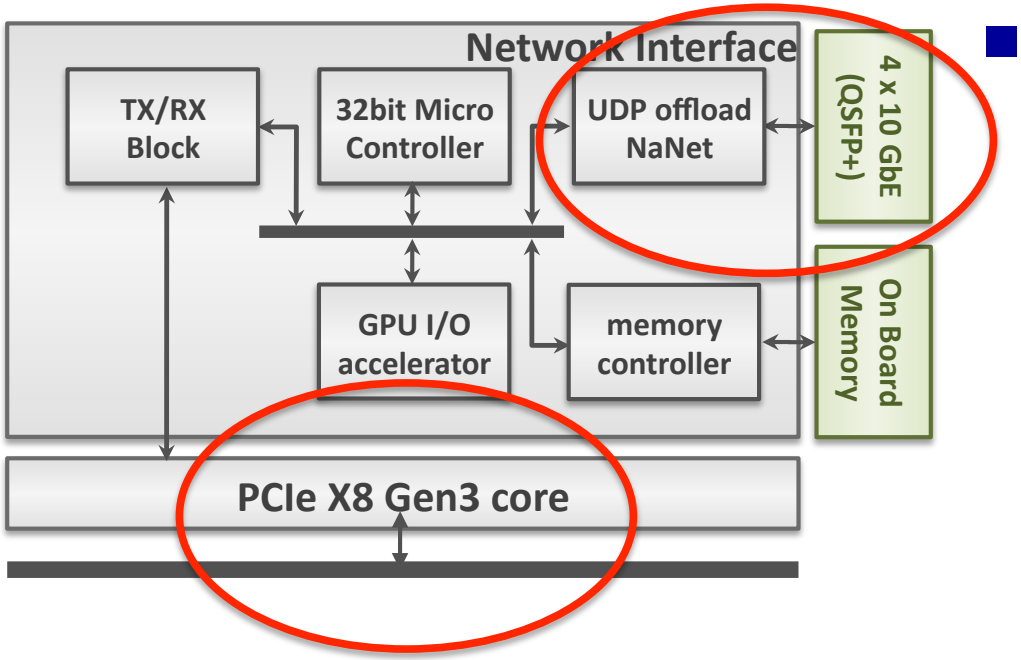
    ---> good scaling with link bandwidth.

NaNet-1 Integration in NA62 DAQ & Trigger System

- TTC daughtercard with HSMC connector (Ferrara)
- NaNet receives TTC stream with timing (40 MHz clock, SOB, EOB) and trigger signals from the experiment.
- Allows synchronous operation (accurate latency measurements)

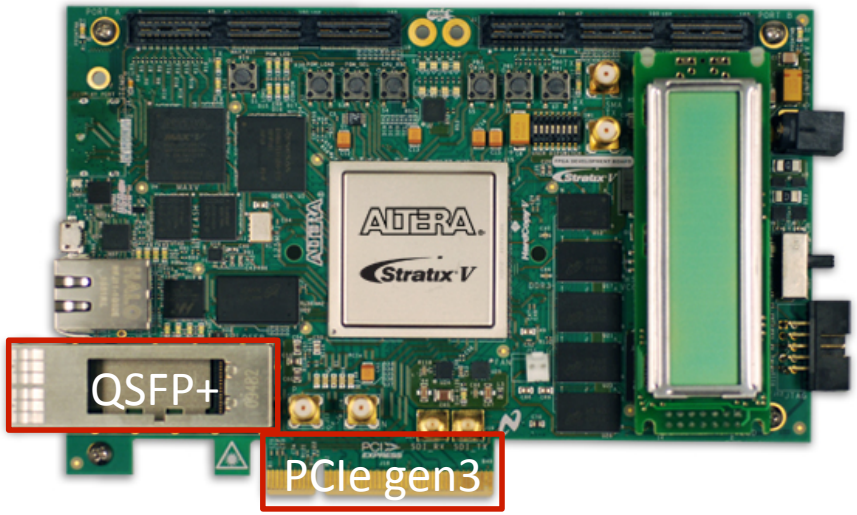
- First integration and tests activities during August technical run.
- Parasitic operation of GPU-based L0 trigger using NaNet-1 scheduled for October (with reduced event rate).

NaNet-10 and last generation FPGAs (Stratix V)

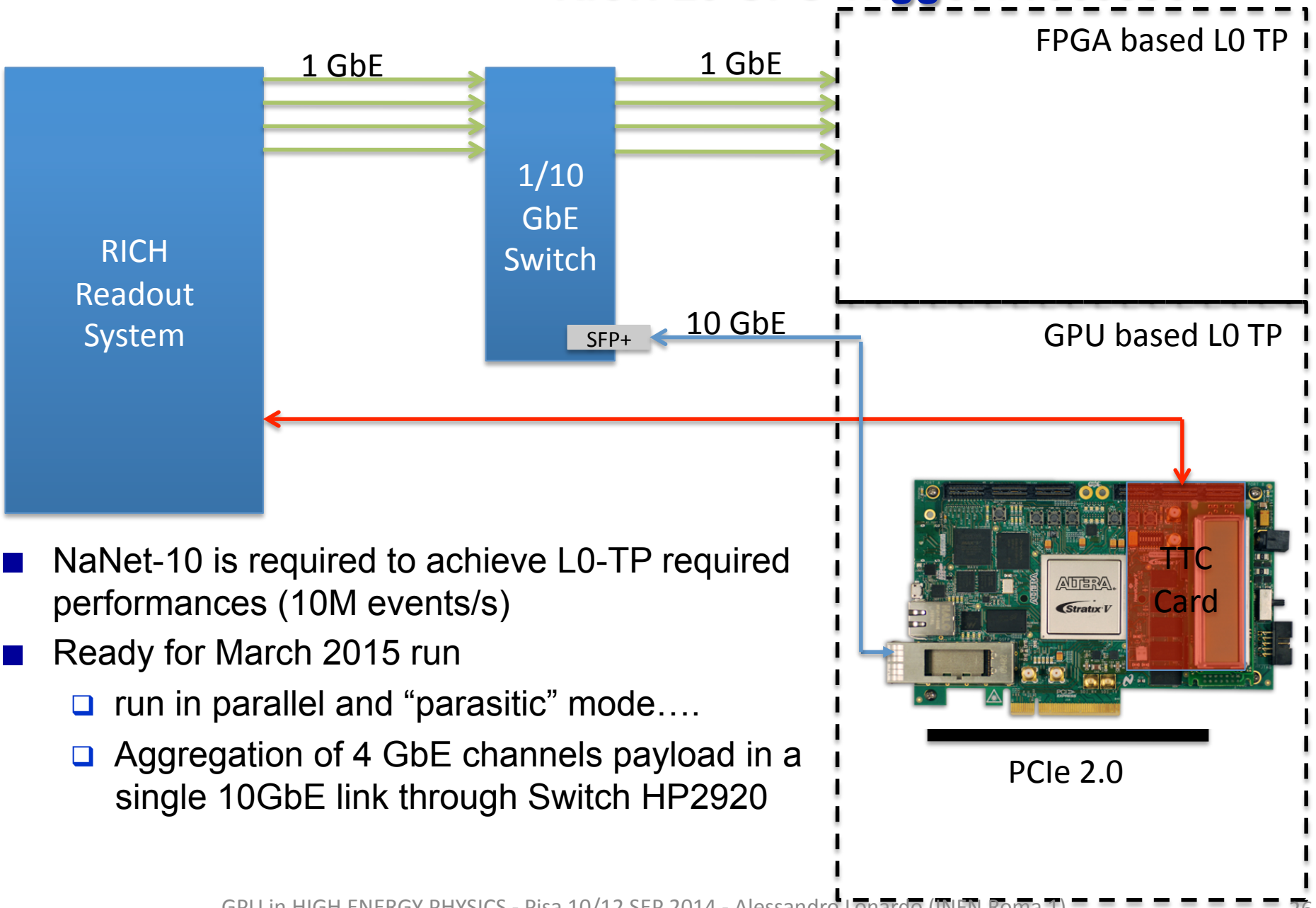


- Implemented on the Altera Stratix V dev board but porting on cheaper board (Terasic TR5-F40W) is possible
  - PCIe gen2 x8 - developing PCIe Gen3 (8 GB/s)
  - Faster embedded Altera transceivers (up to 14.1 Gbps)
  - hardened 10GBASE-R PCS features to support 10 Gbps

QSFP+ to 4 SFP+ cable



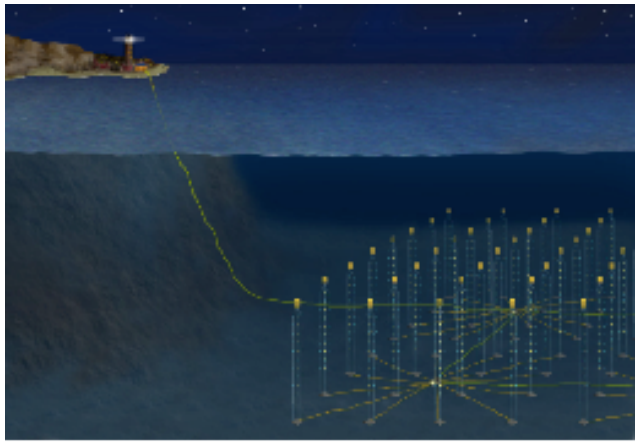
NaNet-10: path towards the final NA62 RICH L0 GPU Trigger Processor

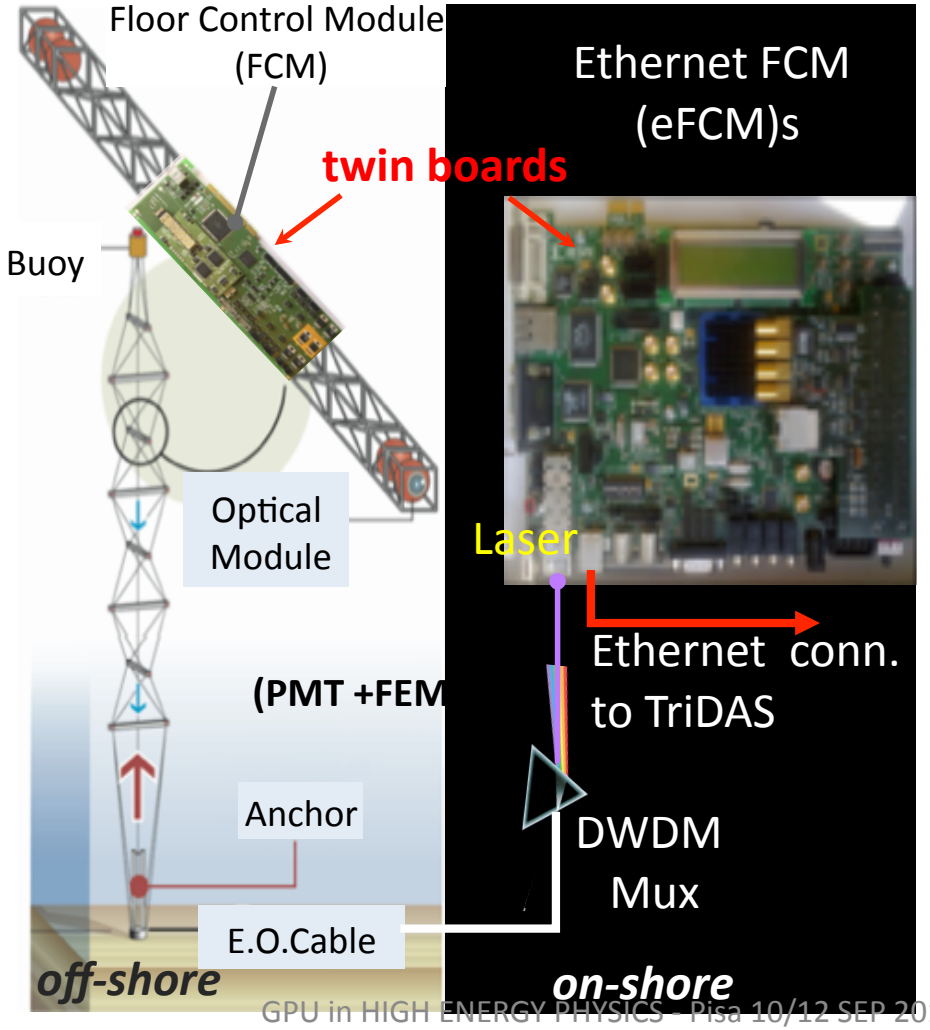


- NaNet-10 is required to achieve L0-TP required performances (10M events/s)
- Ready for March 2015 run
  - run in parallel and “parasitic” mode….
  - Aggregation of 4 GbE channels payload in a single 10GbE link through Switch HP2920

KM3Net-IT Experiment

- **KM3Net-IT**: an European deep-sea research infrastructure, hosting a neutrino telescope with a volume of cubic kilometer at the bottom of the Mediterranean Sea.





- Current read-out system design employs a large number of NO state-of-the-art components
  - 2.5 Gb/s optical link per 800Mb/s payload
  - 2 twinned (on and off-shore) FCM boards per floor (14 floors per tower)
  - Many PCs for HW read-out hosting
- When scaling to Km^3 many cost/size/power/reliability issues.

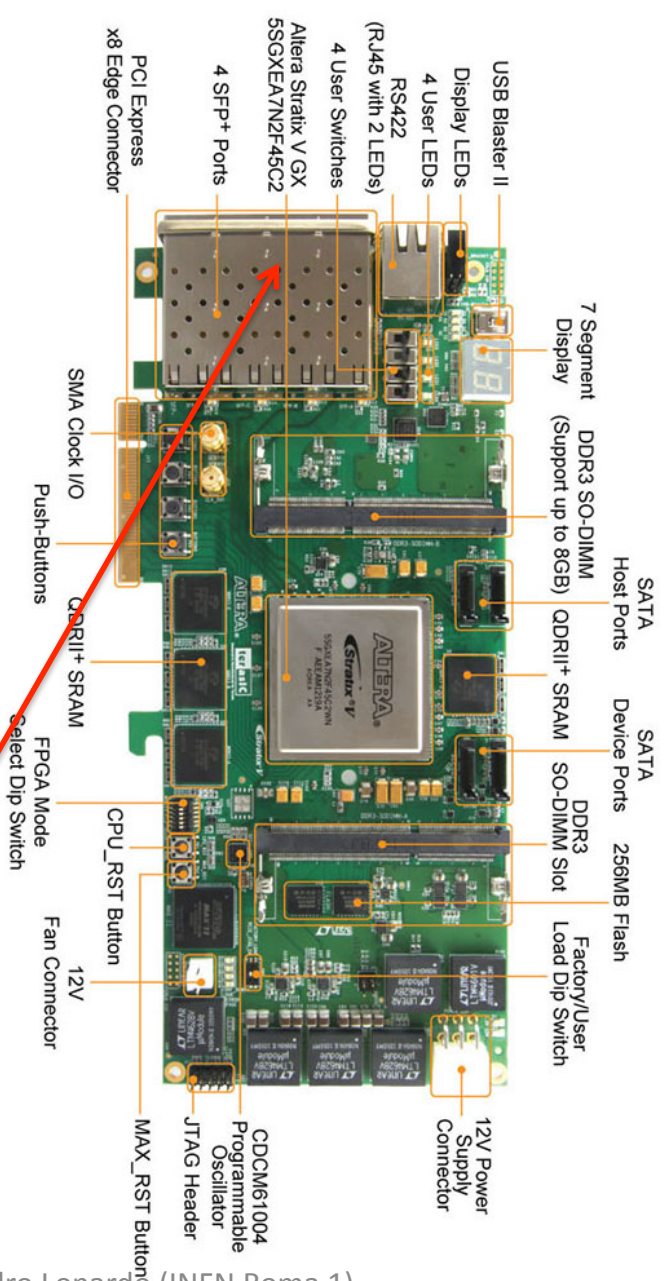
Why NaNet^3?

NaNet^3 specifications for enhanced read-out system:

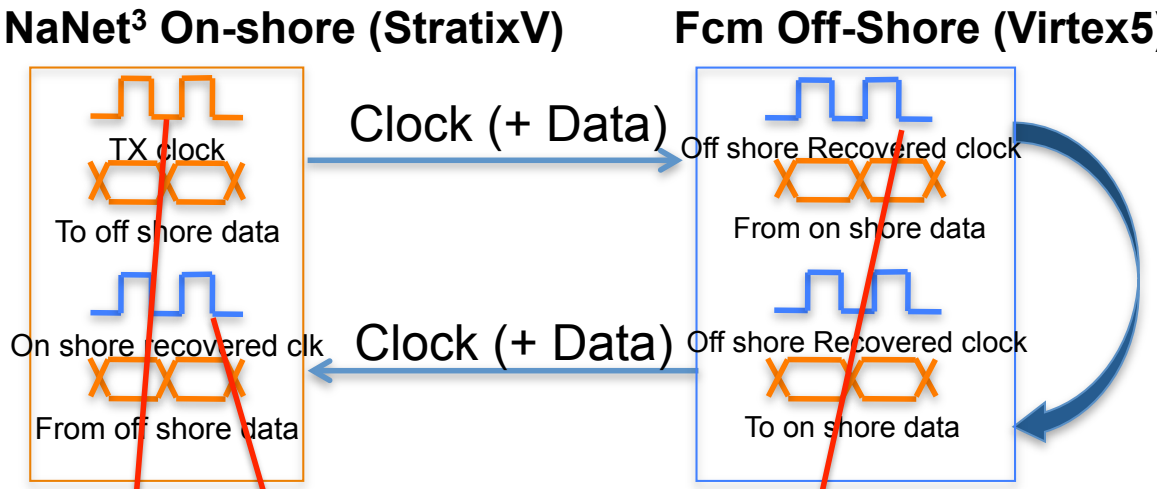
- **4 channels**/PCIe board
  - i.e. less PCs, less read-out boards,…
- Link speed up to **10Gb/s** ✔
  - Further reduction of a factor 4 in number of channel is made possible
- **GPUDirect P2P/RDMA** capability
  - to support future GPU-based trigger developments (see B. Bouhadef's Talk) ✔
- **Deterministic latency link**
  - "Fixed Latency" clock distribution for under-water events time-stamping ✔

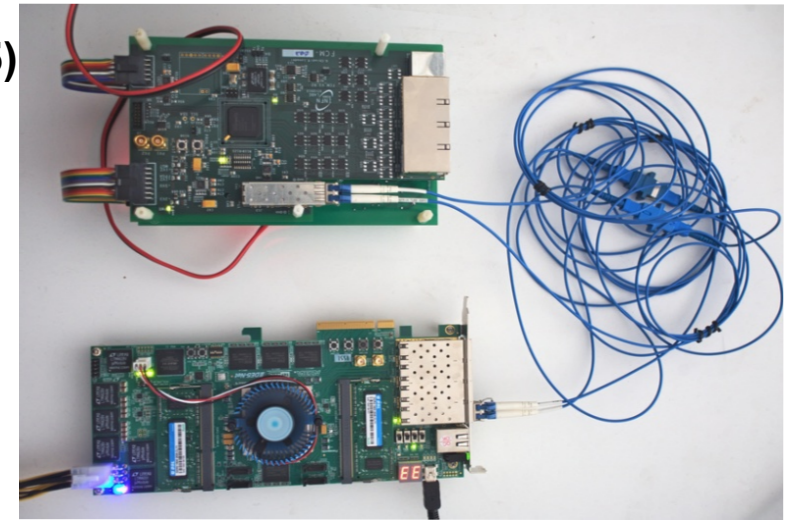
Implemented on **Terasic DE5-NET** board

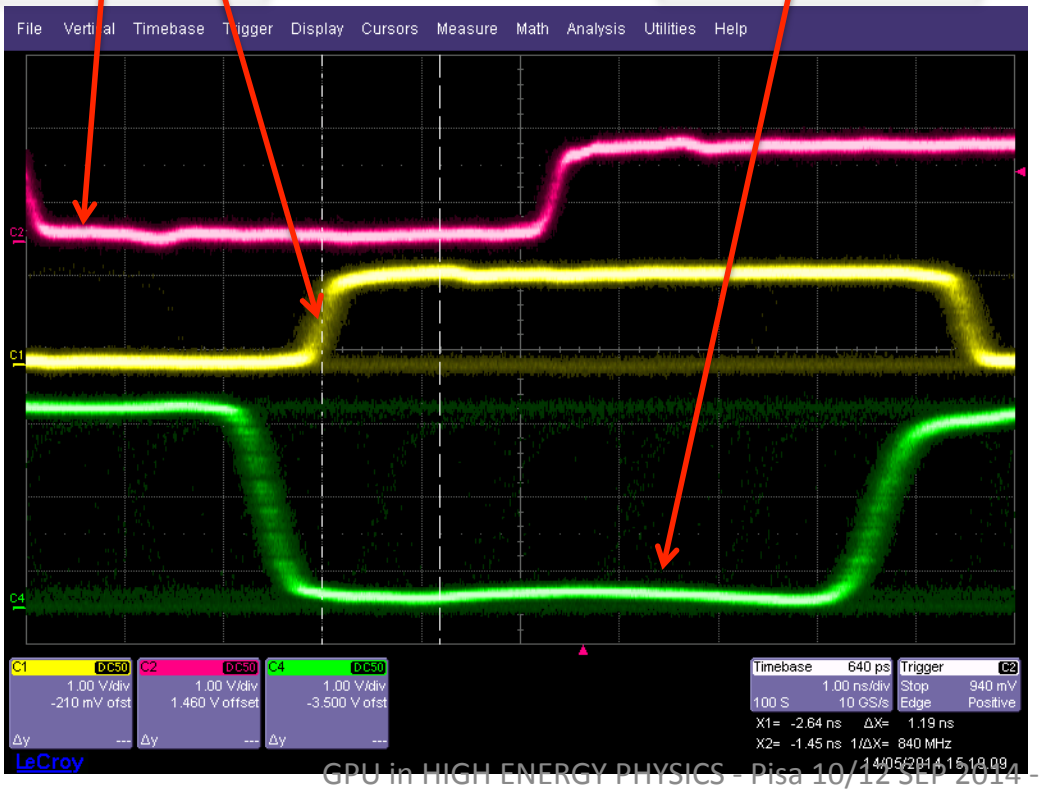
- Altera Stratix V based dev board
- PCIe Gen2 x8 (developing Gen3)
- 4 independent SFP+ ports (up to 10Gb/s)
- Deterministic Latency mode for transceiver



Deterministic latency link inter-operability



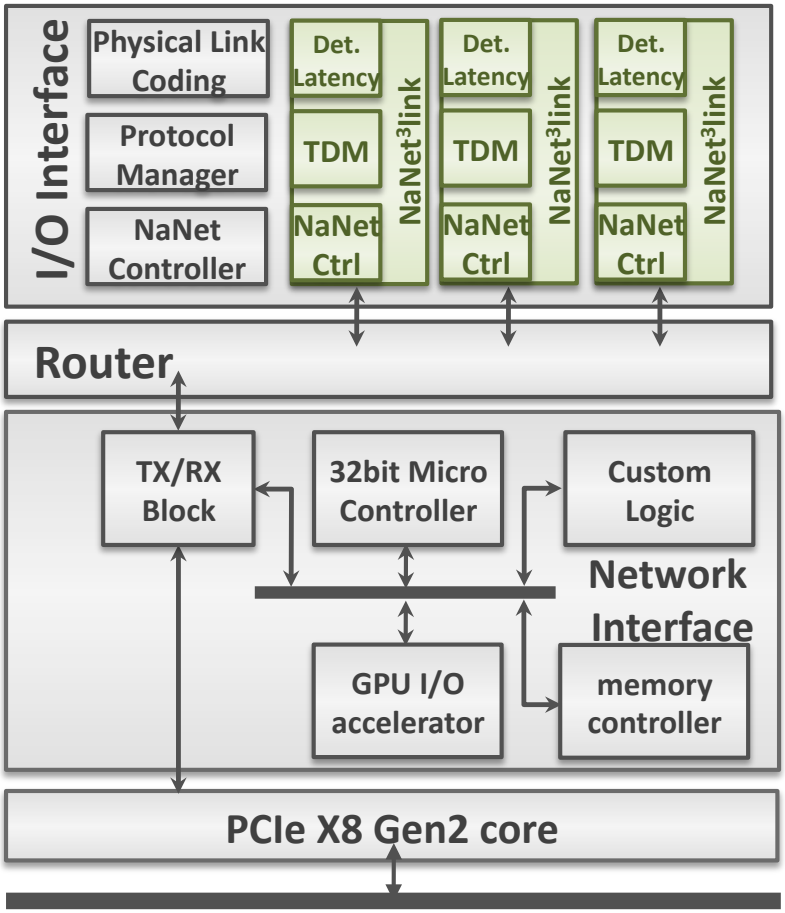




- Testbed: FCM vs Terasic DE5-Net
  - Custom hw mode for FCM Transceivers (Xilinx)
  - Latency deterministic mode for Stratix V Transceiver
  - 2mt copper and 2 mt long fiber
- Test:
  - 12 hours of periodic (~s) Tx clock reset to verify pll locking and rx word alignment

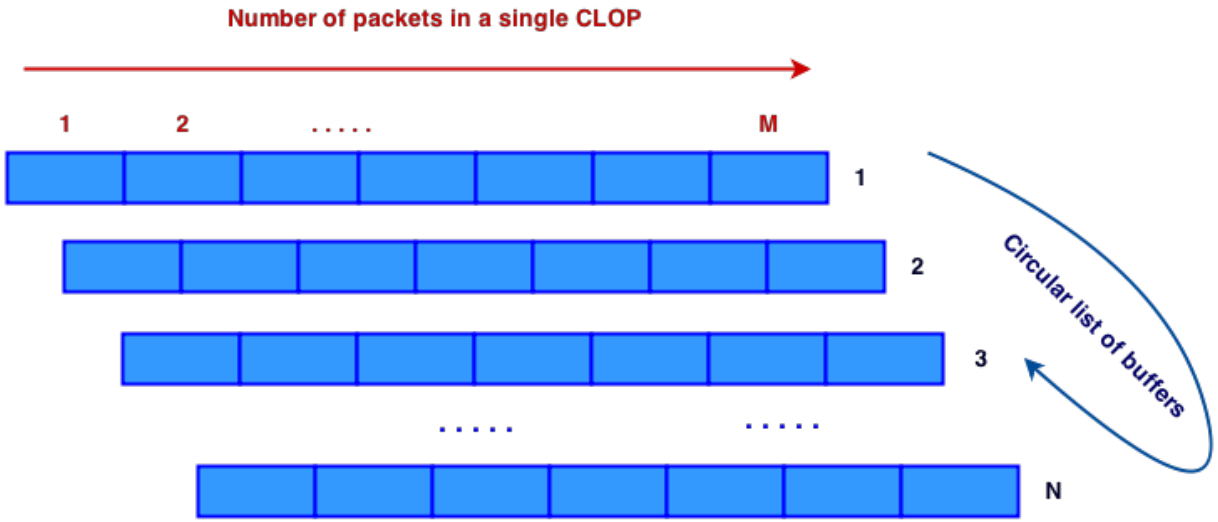
NaNet³ development status

- **NaNet³ link (currently 1 out of 4):**
  - Physical Layer: Altera Deterministic Latency Transceivers (8B10B encoding scheme)
  - Data Layer: Time Division Multiplexing (TDM) data transmission protocol.
    - RX path: payload of different off-shore devices, multiplexed on continuous data stream at fixed time slot
      - (PCIe DMA transaction to CPU/GPU memory)
    - TX path: limited data rate per FCM for slow control devices messages
      - (PCIe TARGET transaction from CPU/ GPU memory)
- **NaNet Ctrl:**
  - protocol translation: encapsulates TDM data stream in APElink packet protocol.
  - Destination Virtual Address generation
- **PCIe X8 Gen2**
  - CPU/GPU Memory Write Bandwidth ~ 2.5 GB/s



NaNet³ software stack

- **Linux Kernel Driver**
  - Status/Configuration registers.
  - TX registers interface.
- **NIOS II Firmware**
  - New BSP for NaNet³ board.
  - Initialization of NaNet³ channels.
  - Management of 4 concurrent data streams



- **Application Library**
  - nan3_t nan3_open(int card_id); int nan3_close(nan3_t nan);
  - int nan3_register_clop(nan3_t nan, nan3_chan_id_t chan, u32 pkts, u32 items, u8 is_gpu);
  - int nan3_wait_event(nan3_t nan, nan3_event_recv_t* event);
  - u32 nan3_write_slow_data(nan3_t nan, u8 channel, u8 deviceId, u32 data);

Conclusions

- NaNet design has proved to be effective in two different experimental contexts thanks to its modularity and high performance/low latency (RDMA, GPUDirect, network protocol offloading) features:
  - NA62 GPU-based Low Level Trigger
    - Demonstrated real-time data communication between the NA62 RICH read-out system and the GPU-based L0 trigger processor over a single GbE link.
    - Demonstrated scalability of the design up to multiple 10GbE read-out channels
  - KM3Net-IT on-shore read-out system:
    - Demonstrated different vendors, high-end FPGAs serial links inter-operability (with Deterministic Latency).
    - Compliant with Real-time constraints of TDM processing.
    - Ready for future enhancements (on-shore link aggregation, GPU-bases trigger…)
- Work in progress
  - 10 GbE support on multiple platforms (Terasic DE5-Net, Altera Stratix V dev. Kit,…)
  - Adoption of last generation FPGA devices for PCIe Gen3 integration and faster transceivers (→ higher I/O bandwidth)
  - NaNet^3 full-fledged (4 links) design for KM3Net-IT
- NaNet^3 design will be completed and tested on final site before the end of 2014.
- NaNet-10 will be completed during 1Q2015.

Thank you….

F. Ameli[(a)], R. Ammendola[(b)], A. Biagioni[(a)], A. Cotta Ramusino[(c)], M. Fiorini[(c)], O. Frezza[(a)], G. Lamanna[(d)], F. Lo Cicero[(a)], A. Lonardo[(a)], M. Martinelli[(a)], I. Neri[(c)], P. S. Paolucci[(a)], E. Pastorelli[(a)], L. Pontisso[(f)], D. Rossetti[(e)], F. Simeone[(a)], F. Simula[(a)], M. Sozzi[(f)], L. Tosoratto[(a)], P. Vicini[(a)]

(a) INFN Sezione di Roma
(b) INFN Sezione di Roma Tor Vergata
(c) Università di Ferrara e INFN Sezione di Ferrara
(d) INFN LNF and CERN
(e) NVIDIA Corporation, USA
(f) INFN Sezione di Pisa and CERN

Backup Slides

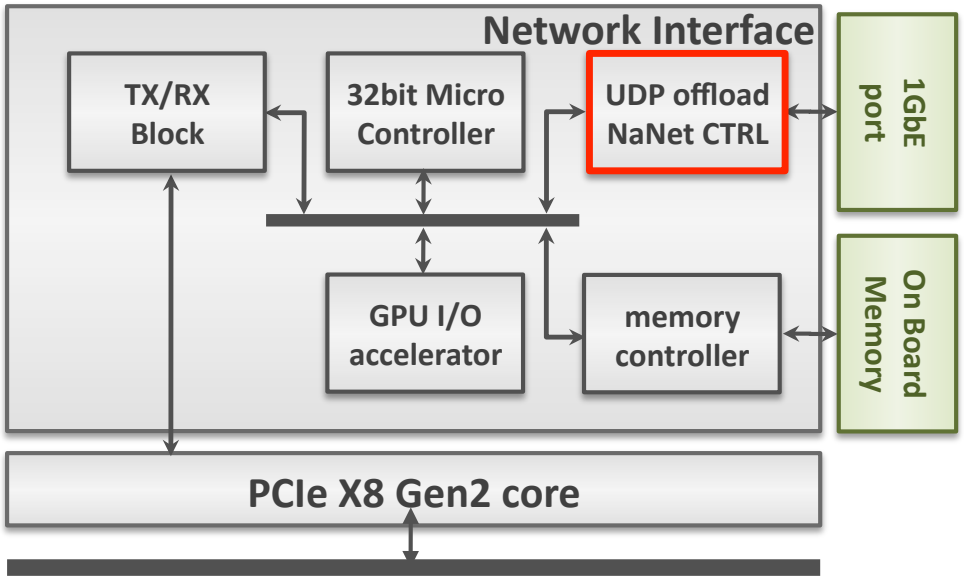
NaNet-1: architectural details

NiosII UDP Offload :

- Objective: **offloading the UDP protocol management** from the Nios II microcontroller.
- collects of data coming from the Avalon Streaming Interface of the Altera Triple-Speed Ethernet Megacore (TSE MAC) and redirects UDP packets into an hardware processing data path.
- Current implementation provides a single 32-bit width channel **@6.4 gbps** (six times greater than the GbE Specs)

NaNet Controller:

- encapsulation of GbE packets in the APEnet+ protocol (and viceversa for output stream)
  - 32-bit data words coming from the Nios II subsystem into 128-bit APEnet + data words.
  - Addition of Header/Footer words and redirect in corresponding FIFO



Standard GbE NIC / Real-Time Kernel

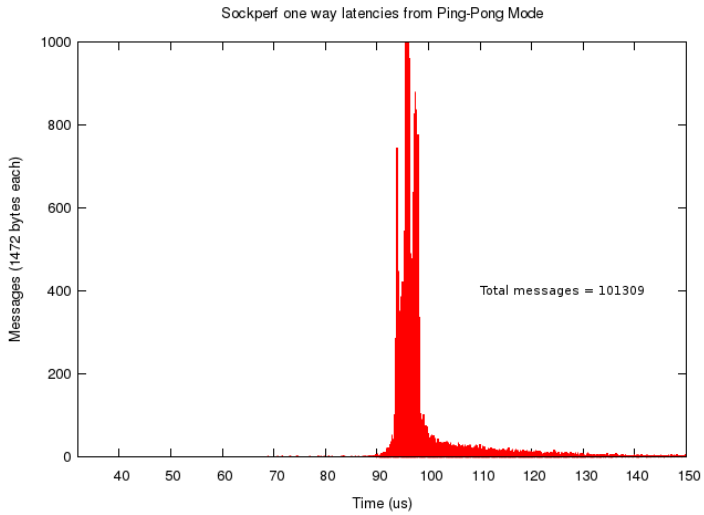
Might a Real-Time kernel come to our rescue?

Main features of such kernels:

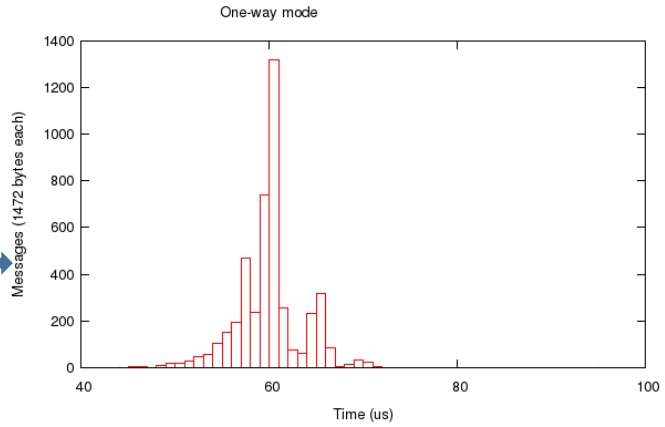
- predictability in response times
- reduced jitters
- microsecond accuracy
- improved time granularity

...but not a panacea...

vanilla kernel: 2.6.33

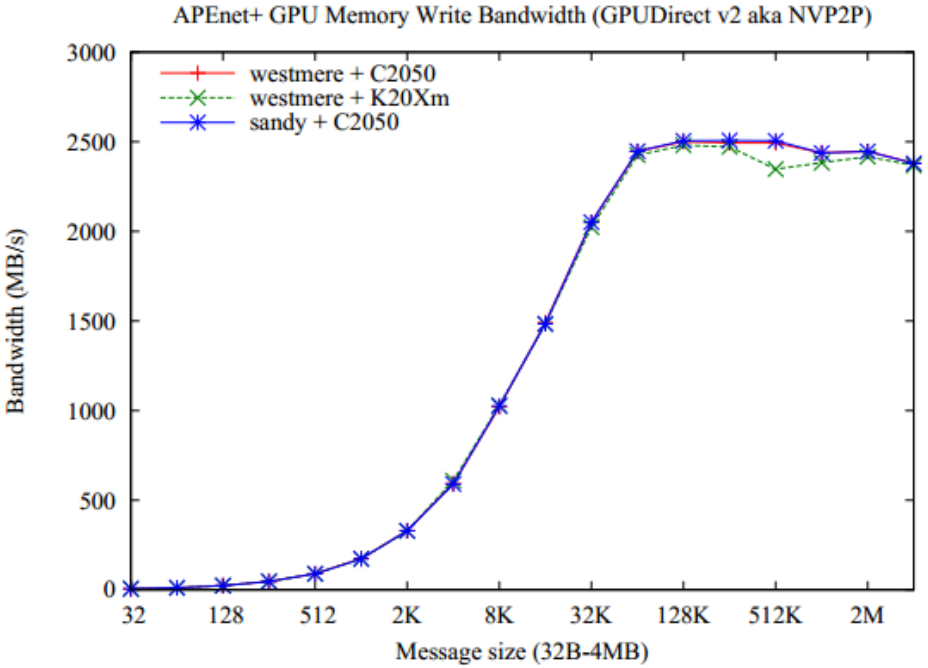


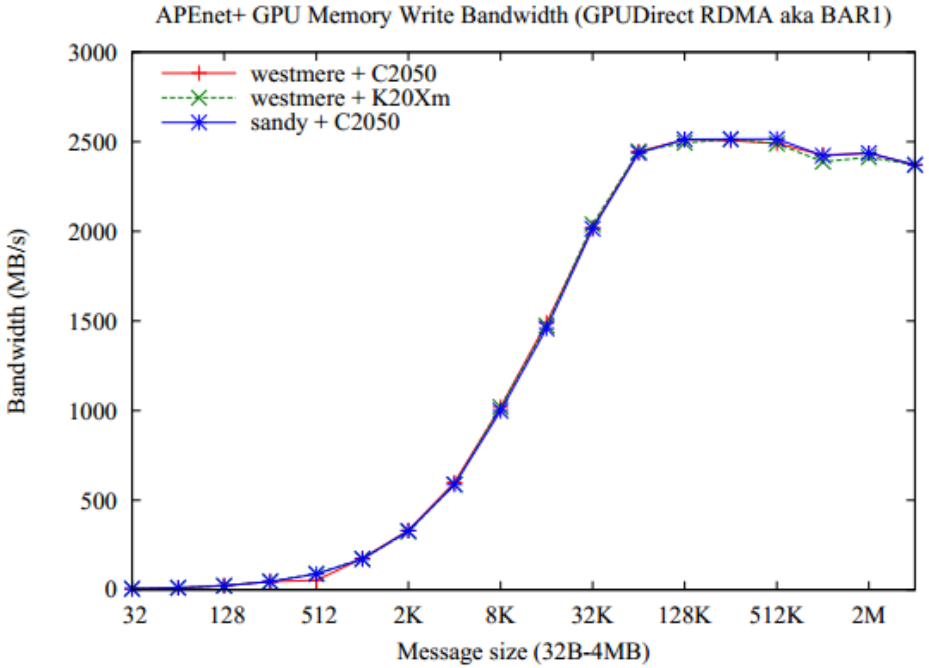
kernel 2.6.33.9-rt31-EL6RT



To avoid other possible sources for latency, the CPUSPEED and IRQBALANCE services has been stopped. The INTERRUPT moderation was disabled either.

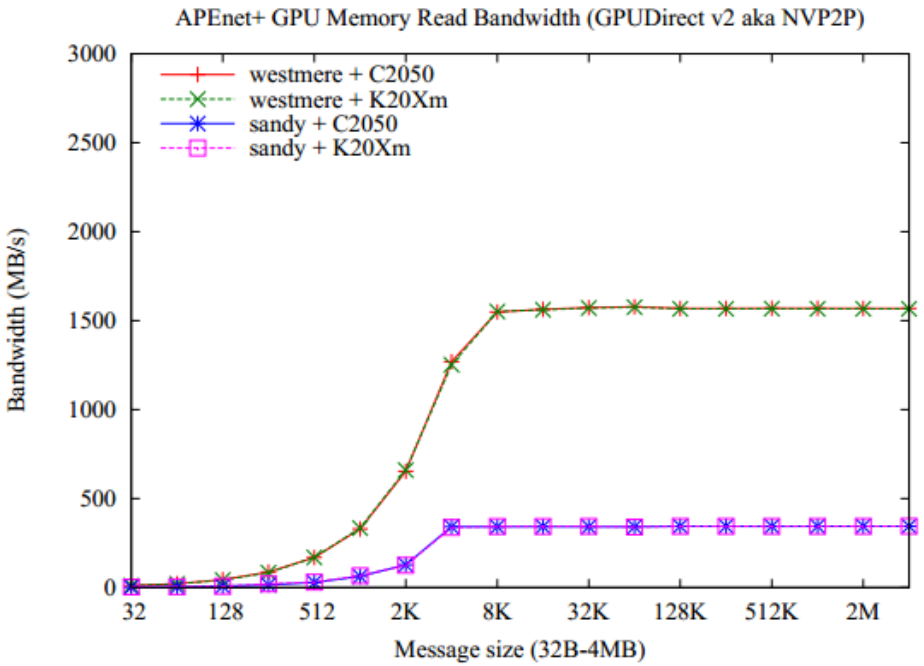
GPU Memory Write (RX)

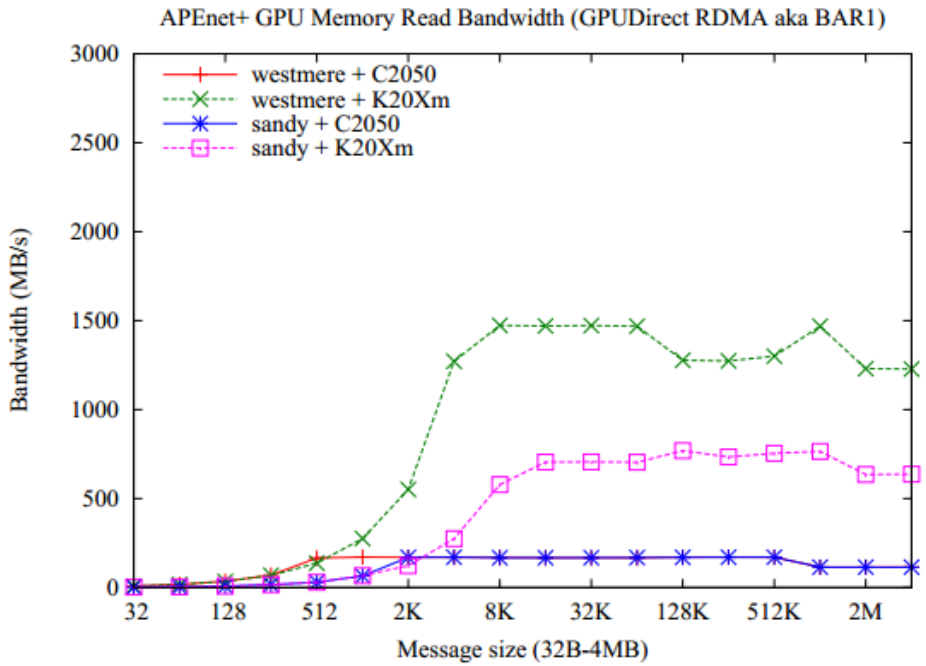




- GPU/CPU independent
- 2.5 GB/s
- The plateau is due to APElink channel

GPU Memory Read (TX)



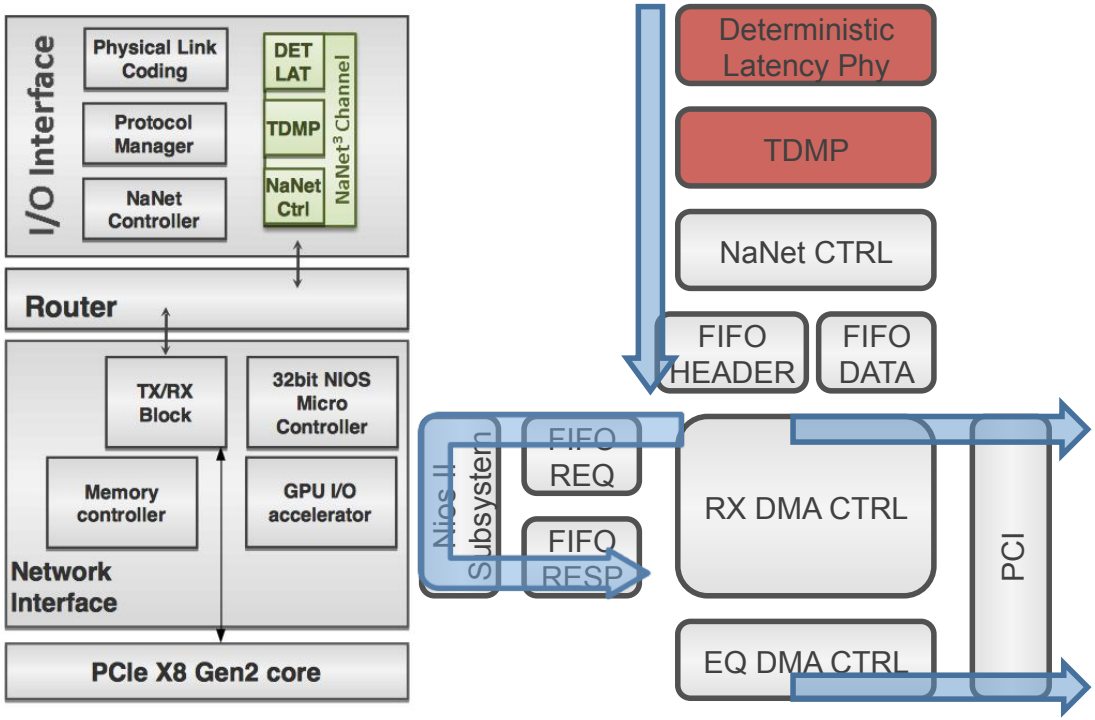


- GPUDirect P2P
  - Stability
  - Sandy Bridge limitations

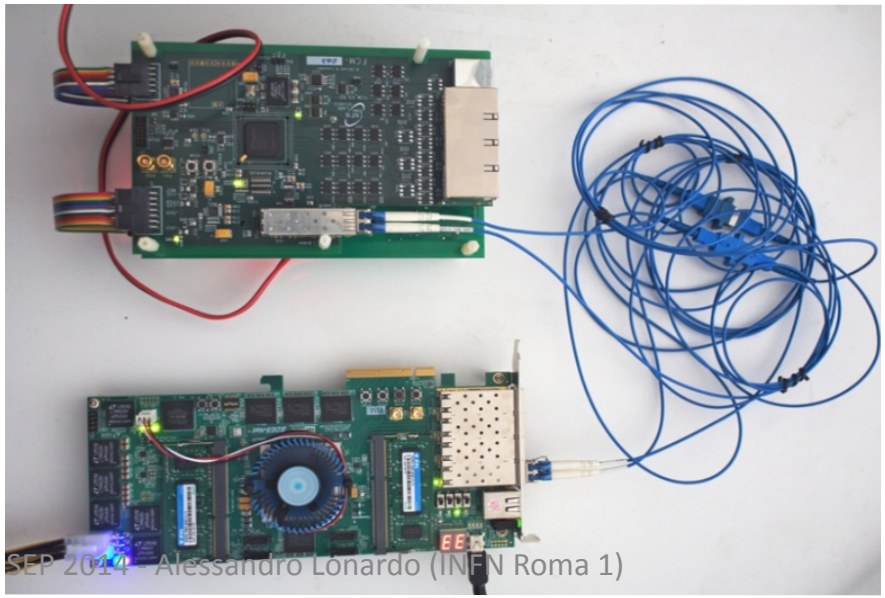
- GPUDirect RDMA
  - Kepler GPUs are OK!
  - Sandy Bridge is NOT OK!

NANET3 architecture and dataflow

- NaNet3 link phy:
  - Altera Deterministic Latency Transceivers
  - 8B10B encoding scheme
  - Time Division Multiplexing (TDM) data transmission protocol
    - payload of different off-shore devices, multiplexed on continuous data stream at fixed time slot



- Preliminary test of:
  - Xilinx – Altera transceivers interoperability
  - Interoperable link with Fixed (Deterministic) Latency

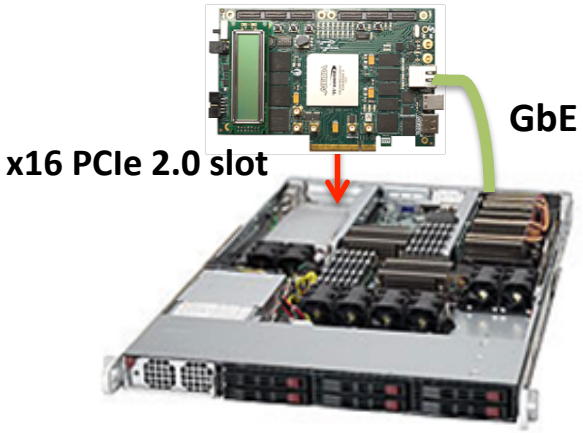


NaNet-1 Latency Benchmark

In a single host process:

- Allocate and register (pin) *N* GPU receive buffers of size *P* x sizeof(16 events) in a **circular list**.
- In the main loop:
  - Read TSC Register (*cycles_bs*)
  - Send *P* UDP packet containing 16 events over the host GbE intf
  - Wait for a received buffer event
  - Read TSC Register (*cycles_ar*)
  - Launch the GPU kernel on next buffer in circular list.
  - Synch Streams
  - Read *(cycles_ak)*
  - Record *latency_cycles_comm = cycles_ar - cycles_bs*
  - Record *latency_cycles_calc = cycles_ak – cycles_ar*

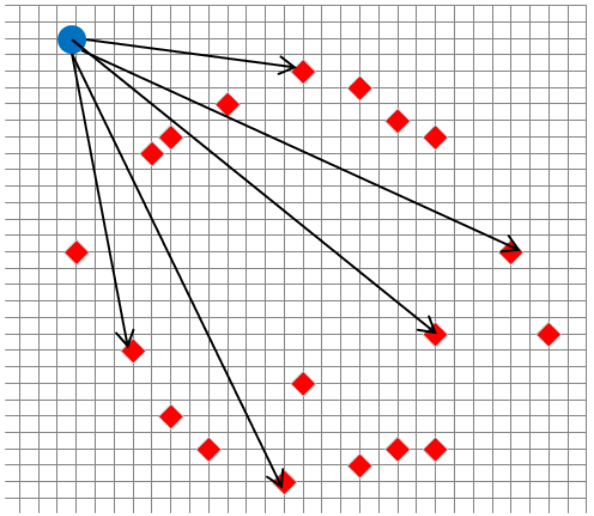
- Results in good agreement with oscilloscope measures (TEL62).



GPU Ring Search Algorithms

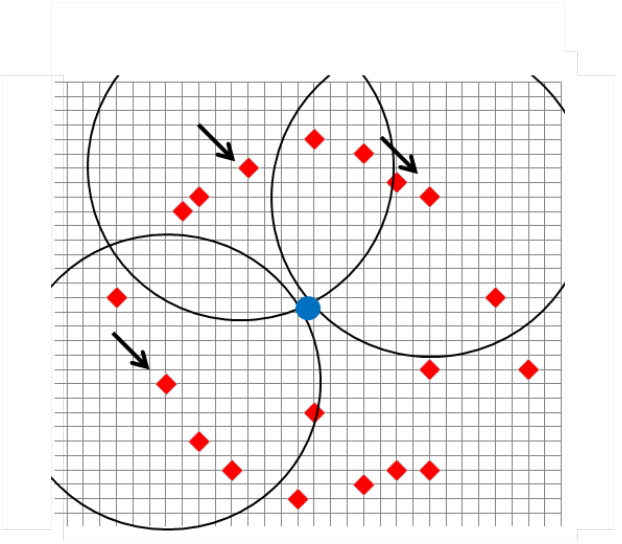
- Events can produce multiple ring hits patterns on PMs.
- 95% of generated hits patterns produce a single ring.
- Single ring search algorithms are used by multi-ring search algorithms.
- Let's focus on single ring pattern search algorithms.

- DOMH/POMH: Each PM (1000) is considered as the center of a circle. For each center an histogram is constructed with the distances btw center and hits.

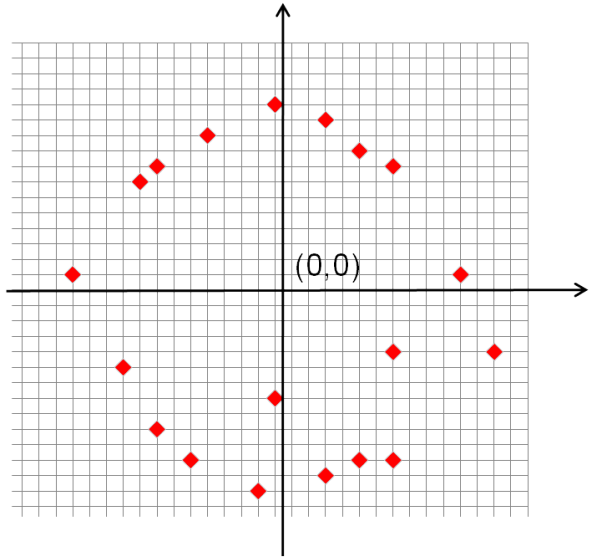


GPU Ring Search Algorithms (2)

- HOUGH: Each hit is the center of a test circle with a given radius. The ring center is the best matching point of the test circles. Voting procedure in a 3D parameters space.

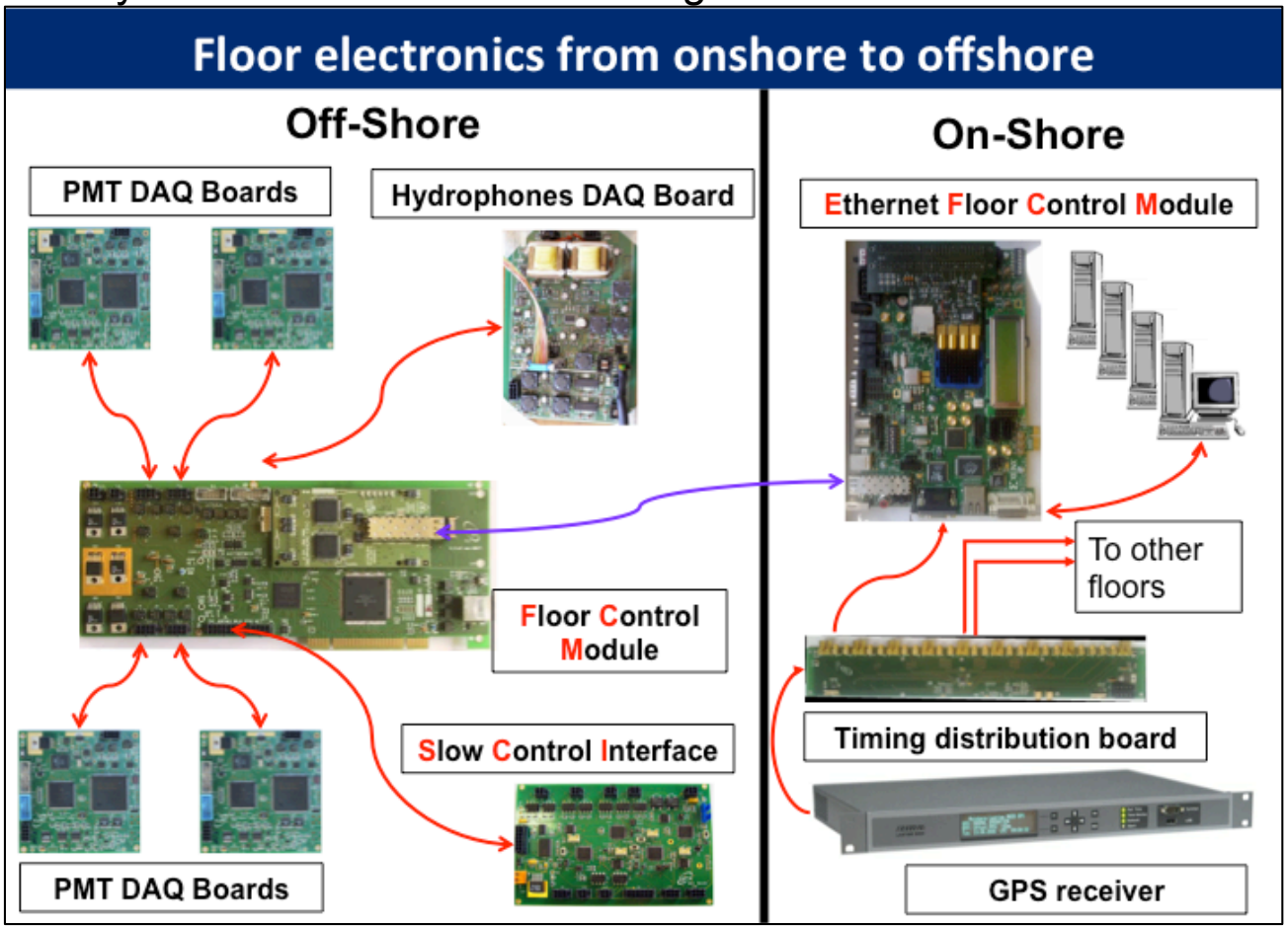


- MATH: Translation of the ring to centroid. In this system a least square method can be used. The circle condition can be reduced to a linear system , analitically solvable, without any iterative procedure.



KM3Net-IT experiment: read-out

- Current **read-out system** design employs a huge number, NO state-of-the-art components
  - 2.5 Gb/s optical link
  - 2 twinned FCM boards per floor
  - Many PCs for HW read-out hosting



F. Simeone
IEEE/NSS 2011

- When scaling to KM3 many cost/size/power/reliability issues!!!