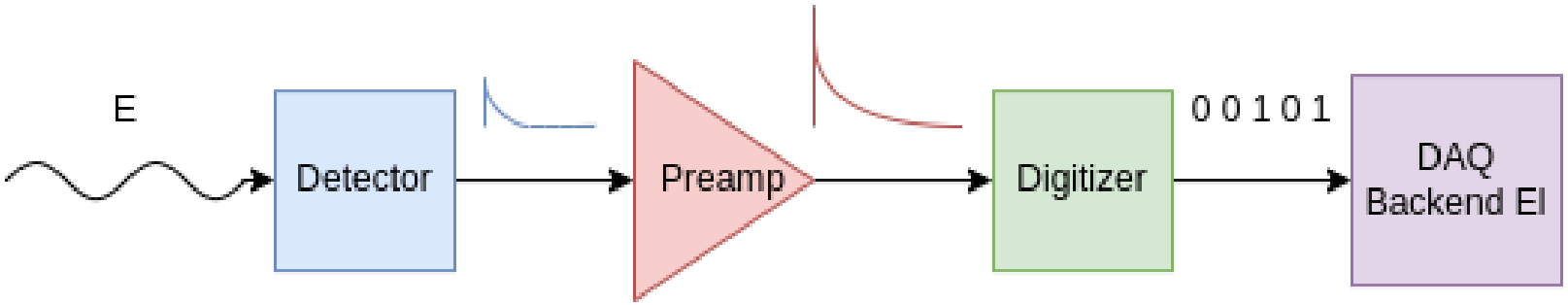
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
INFN

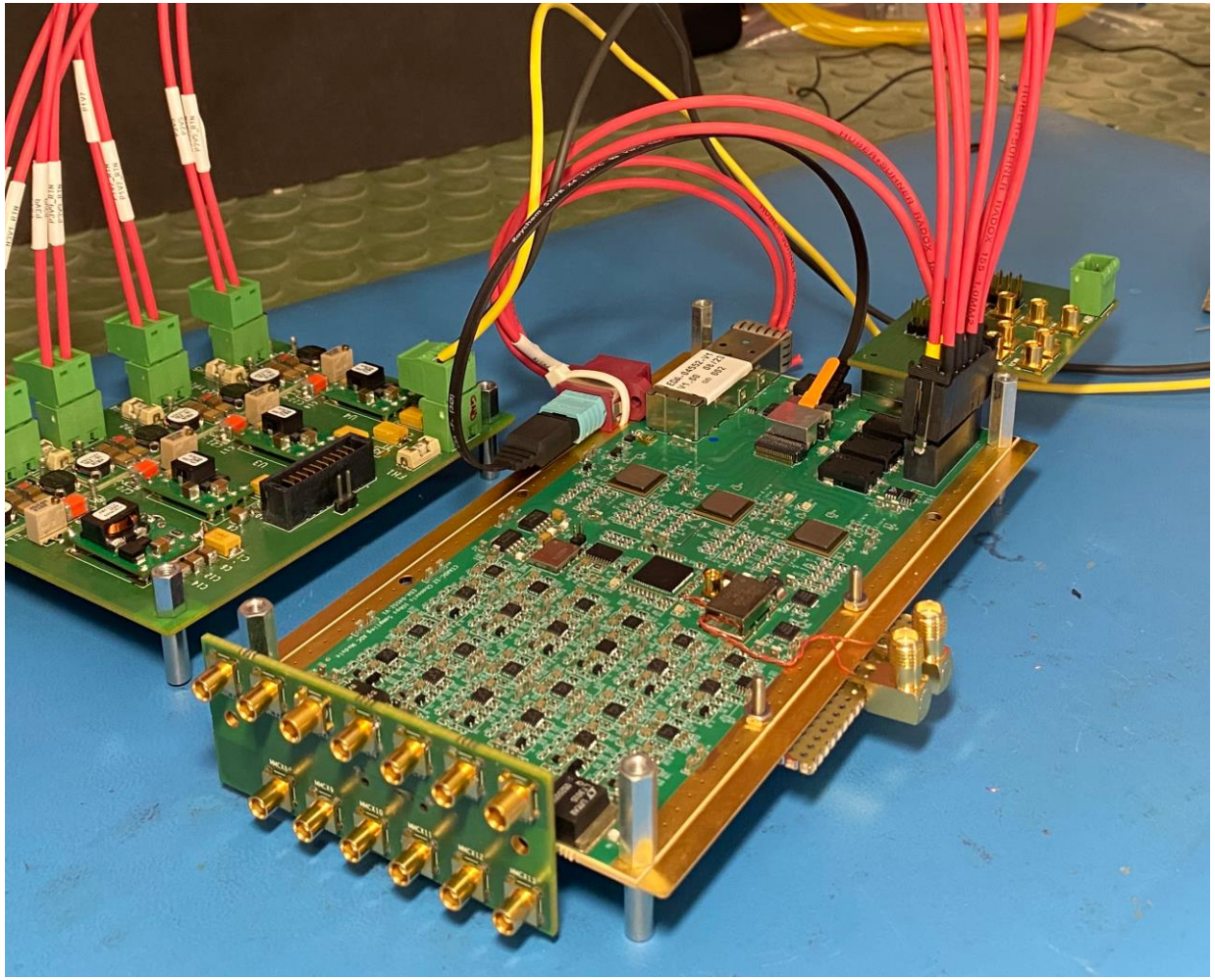
A RoCEv2 RDMA based readout prototype for next generation detectors

F. Marini[1] on behalf of the INFN Padova electronics service
[1] INFN – Sezione di Padova
filippo.marini@pd.infn.it

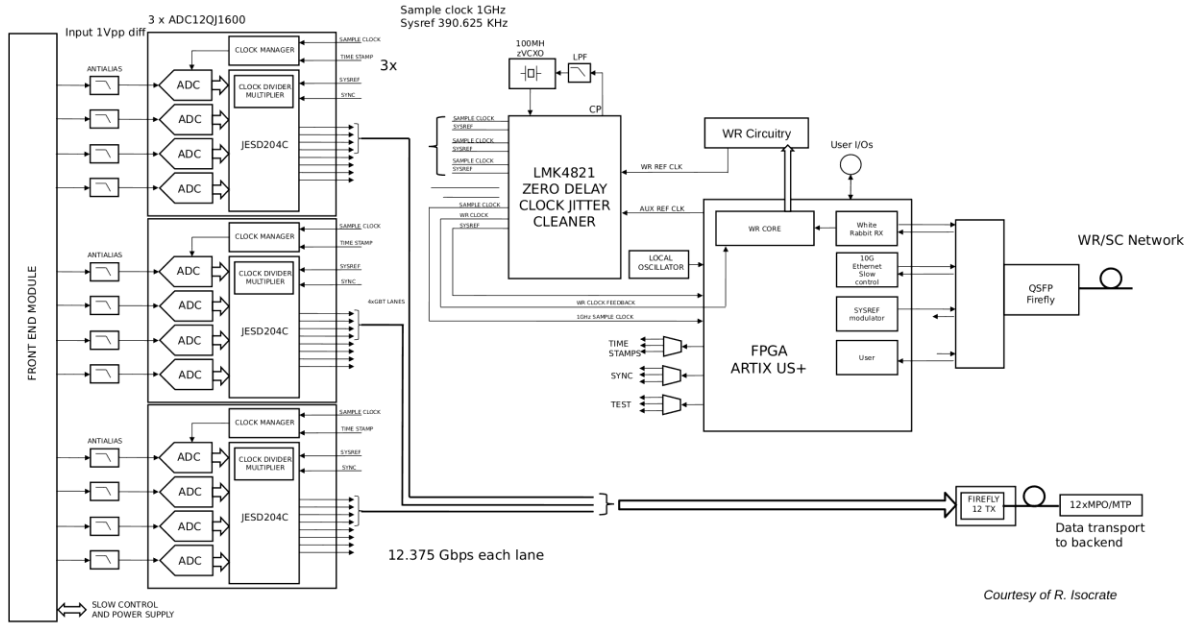


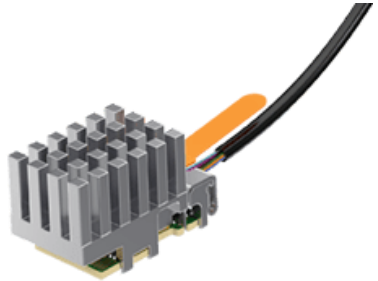
- The particle deposits its energy in a detecting medium
- Energy is converted into an electrical signal. Charge proportional to particle energy
- The charge is typically small and must be amplified to be measured and processed
- The preamplifier changes the pulse's features (typically timing and amplitude characteristics) according to the application and hardware
- The digitizer converts the analog information into sequence of bits, for immediate storage and processing, eventually sending informations to back-end electronics (trigger, waveforms, …)

- The FADC board is an R&D effort
- Possible applications are
  - SWGO
  - CTA – LST
  - ENUBET
  - Others?
- Goals: Acquire the electrical signals coming from the detector, digitize it and send the data to human-accessible back-end electronics
- Optimized for
  - Energy measurements
  - Event rate
  - Power consumption
  - Mechanical constraints
  - Cost

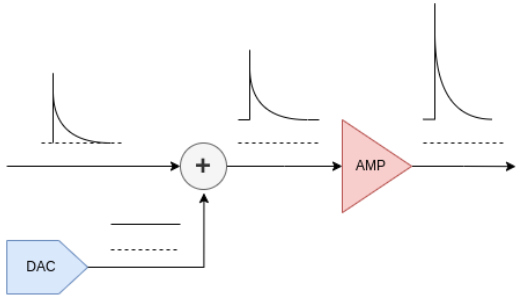


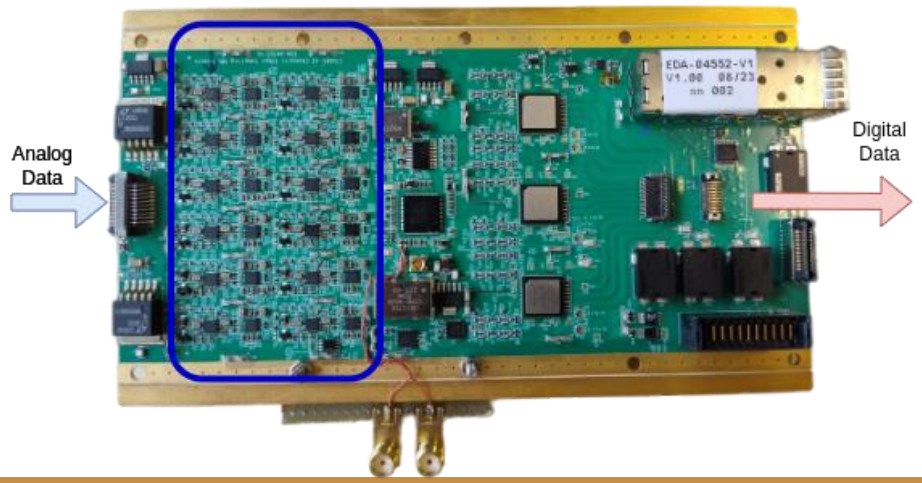
- Flexible pre-amplifier stage. Programmable gain and offset
- Digitizer: 9 bits, 1Gsample/s
- 12 analog input channels per board
- 1 GHz sampling clock generated on board
- Serial links (I2C, SPI) for ADC, PLL and preamplifier stage configuration
- 12 Gbps er channel for serial data transport to backend electronics
  - Data carried out using a FireFly connection (12 Tx)



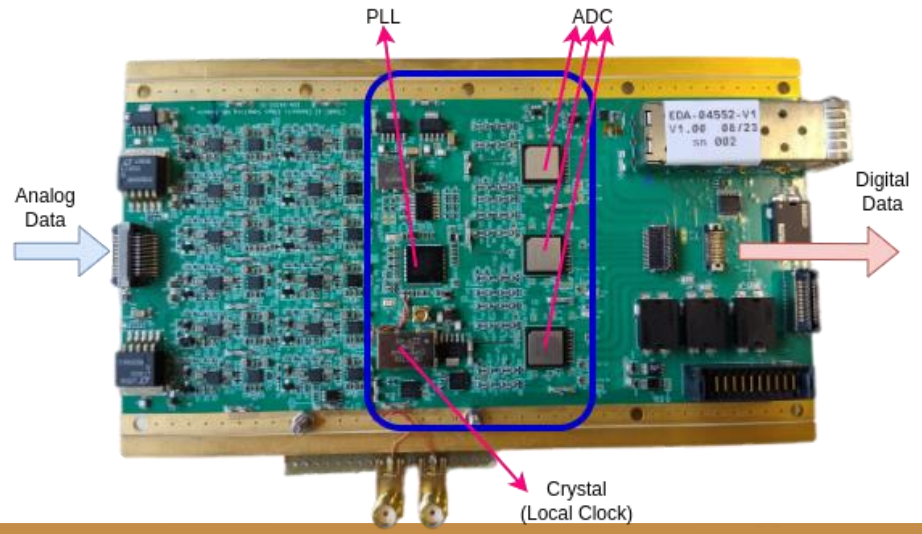


- User-controllable offset
- User-controllable amplification
- Allows for great flexibility





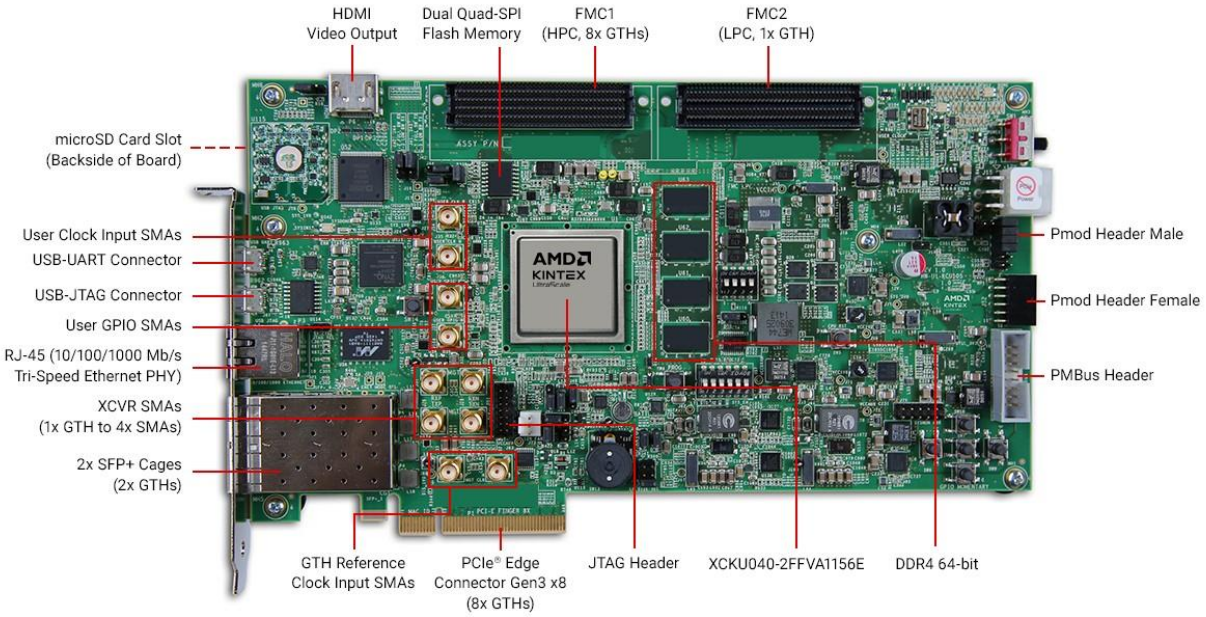
- 1 GHz digitization clock provided by the PLL
- 9 bits (footprint compatible with 12)
- Data is serialized in output (JESD204C)
- Timing synchronization between lanes and deterministic latency guaranteed by the standard



The backend is currently implemented in a Xilinx KCU105 ev. board.

It is responsible for:

- Receive data from a FADC board
- Perform a simple level-1 trigger (leading-edge trigger)
- Send the triggered events to a PC via 10 GbE (Reliable UDP)
- Manage the slow-control for FADC configuration (SPI, I2C, temperature control, etc.)
- Timing synchronization with White Rabbit



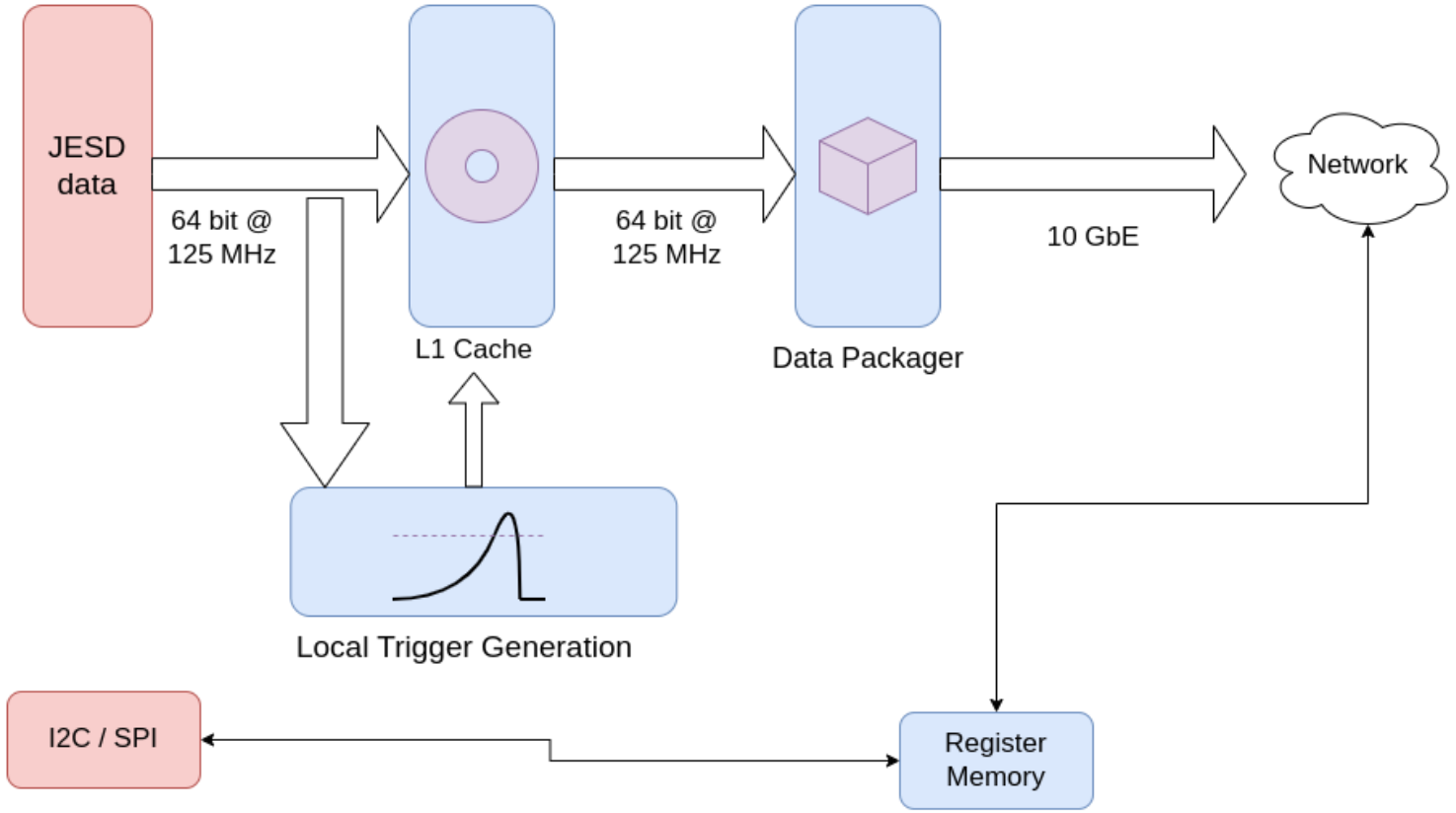
Backend electronics often need FPGAs as the 'brain' of the board

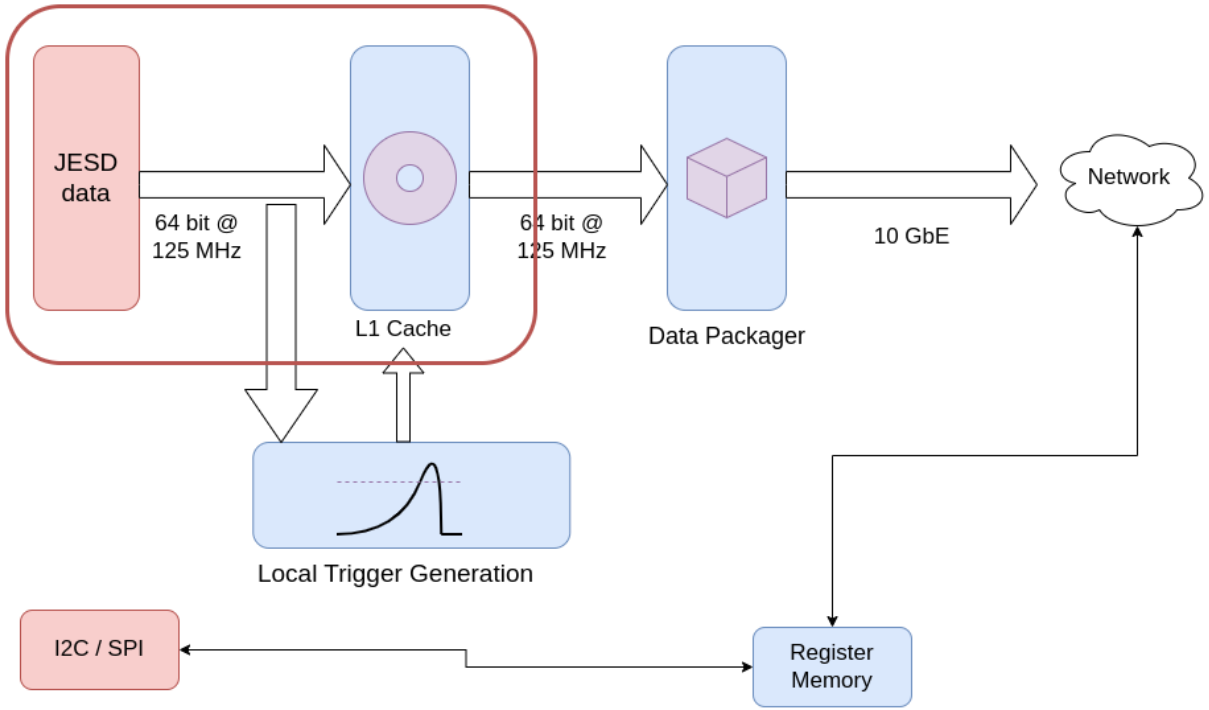
How do we communicate with the FPGA?

- A full Reliable UDP/IP stack is implemented in FPGA
  - Open source SLAC's SURF framework is used
- 10 GbE
- UDP/IP/MAC fully written in VHDL
- Framework allows to easily stream data as well as perform registry access operations for slow-control (Chip configurations, temperature, checking, etc.)

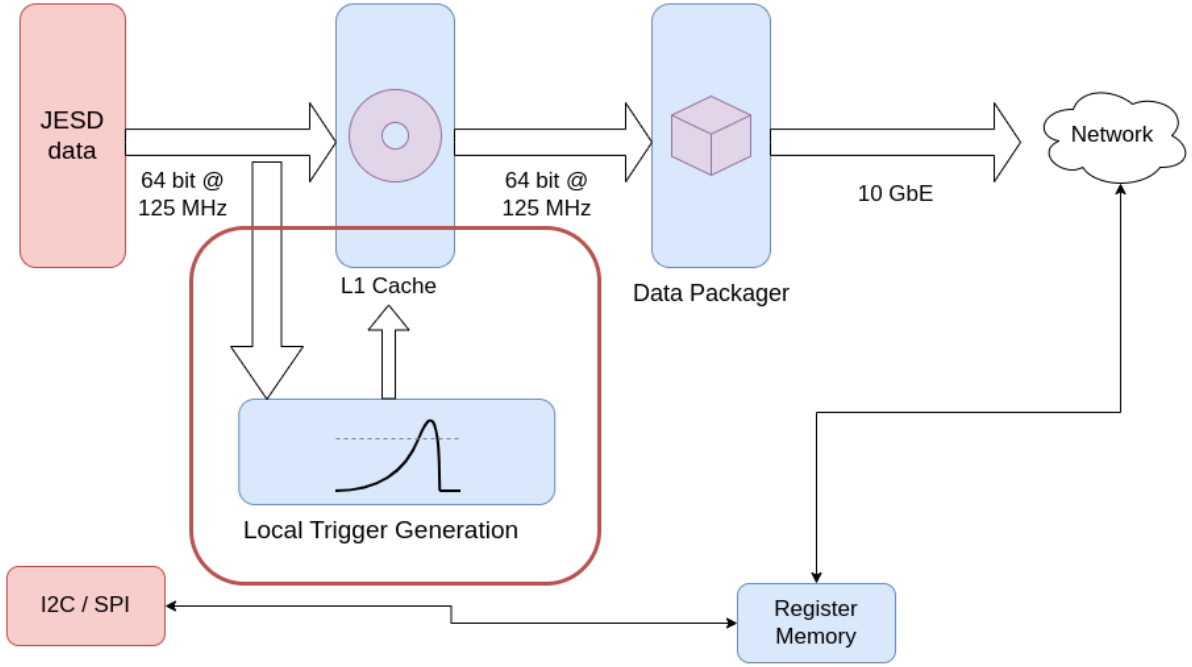
The SURF framework

- https://github.com/slaclab/surf
- HUGE VHDL library for FPGA development
- Many IPs for commonly used components
  - Ethernet
  - AXI4, AXI4-Lite, AXI4-Stream Library
  - Synchronization modules
  - …
- Managed and maintained by SLAC
  - Its possible for everyone to contribute

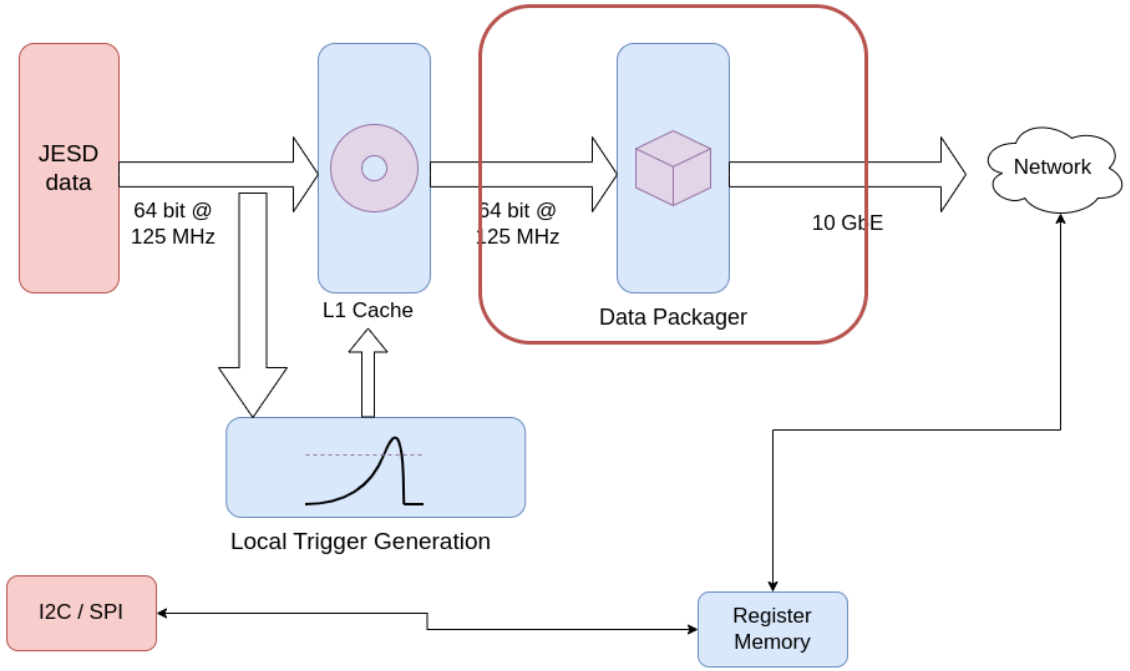

JESD data
64 bit @ 125 MHz
L1 Cache
64 bit @ 125 MHz
Data Packager
10 GbE
Network
Local Trigger Generation
I2C / SPI
Register Memory



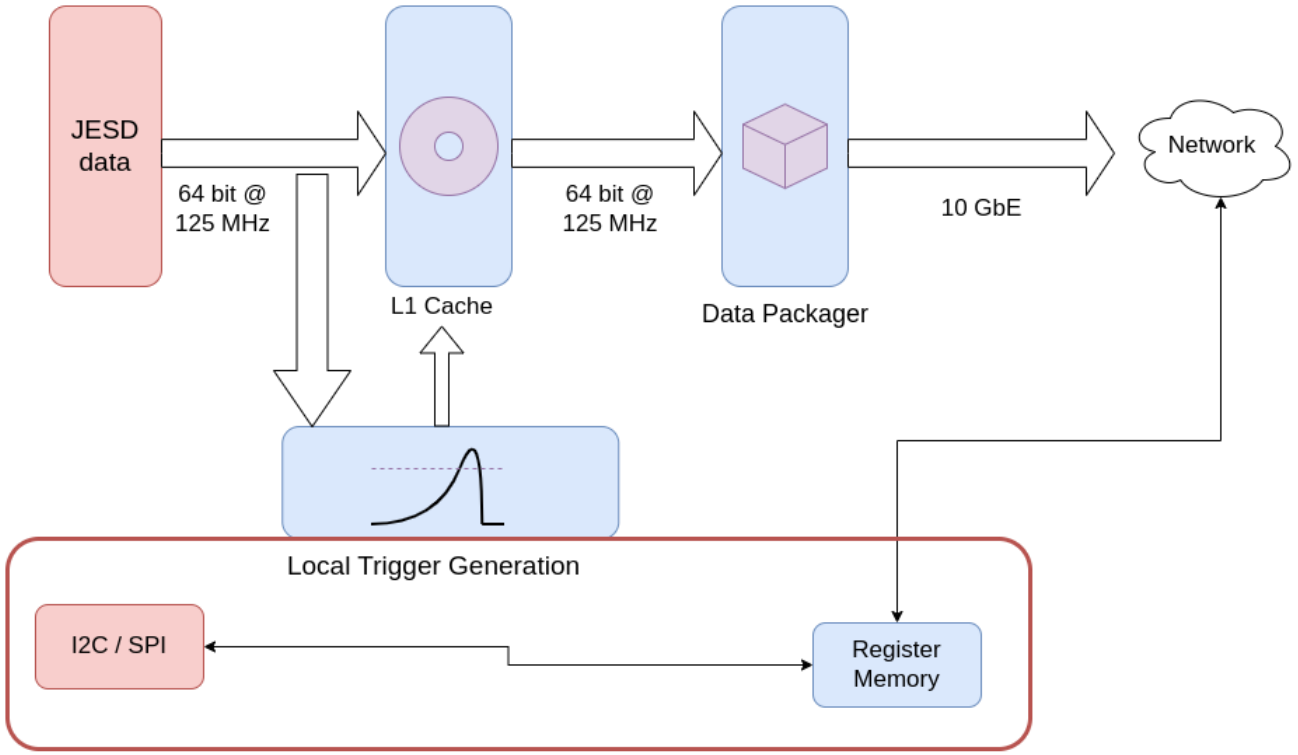
- First the serial data from the optical fibers of the FADC board is received by the FPGA. In the FPGA, data is deserialized in groups of 8 words.
- These words are saved in a circular buffer: a memory space connected end-to-end
- Once the buffer is full, the data gets overwritten



- At the same time, the data is checked against a trigger algorithm
- Currently, a simple leading edge trigger is implemented. But much more complicated algorithm can be implemented (such as clustering-type triggers)
- If a trigger is detected, a notification is issued to the l1-cache

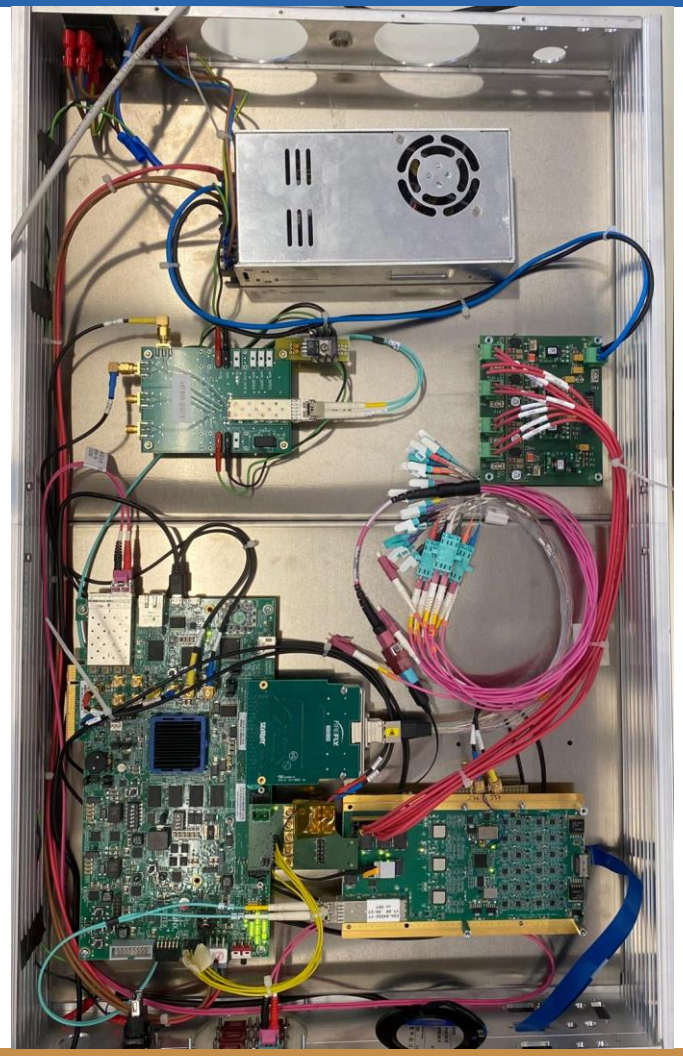


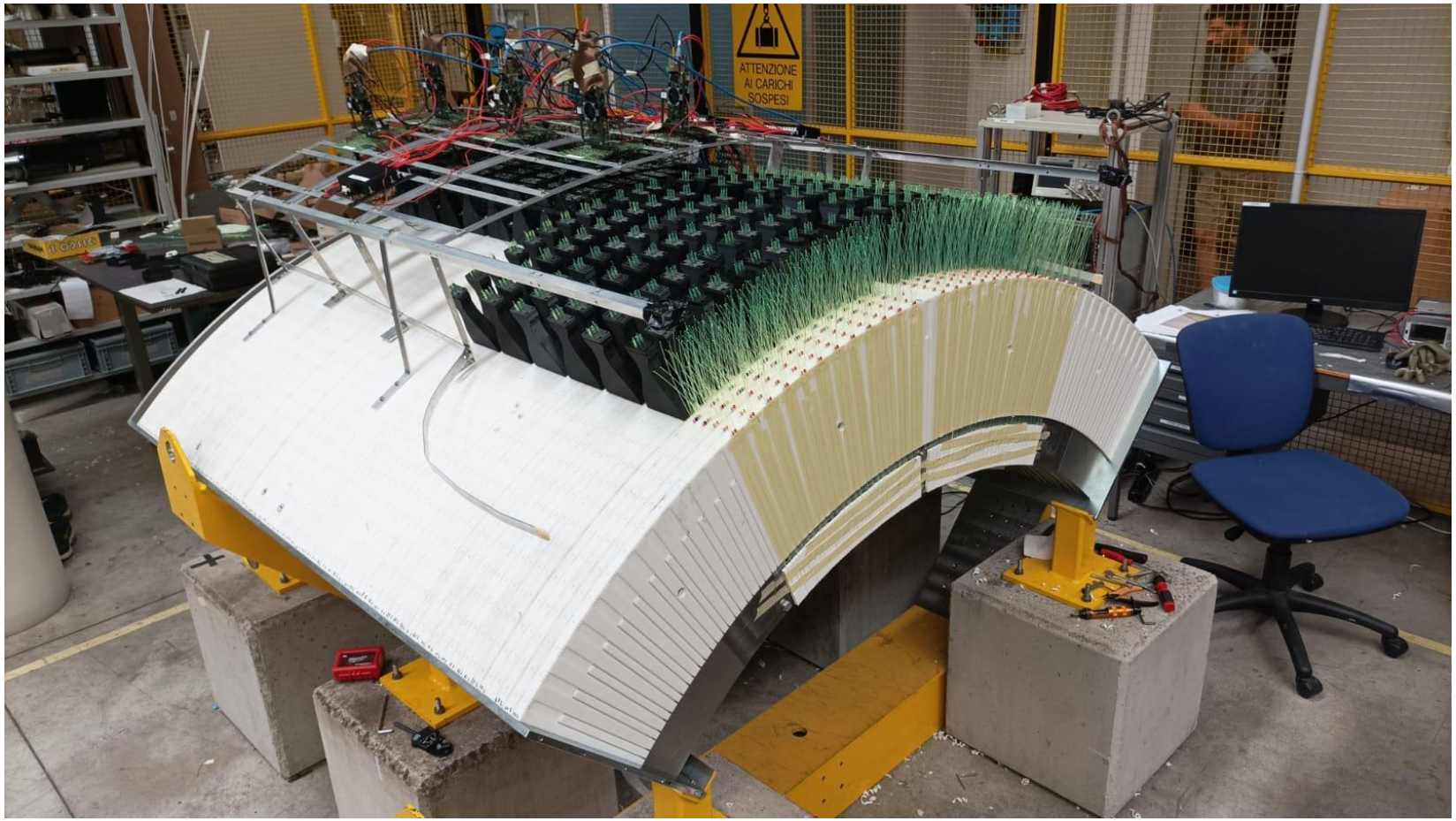
- When a trigger is detected, a chunk of data from the l1-cache relative to the time of trigger is sent to a packetizer module.
- In here, a header and trailer are inserted, with metadata (Timestamp, channel number, ..)
- This composed packet (metadata + payload data) is then sent to the UDP/IP stack (the network) in FPGA in order to reach the computer



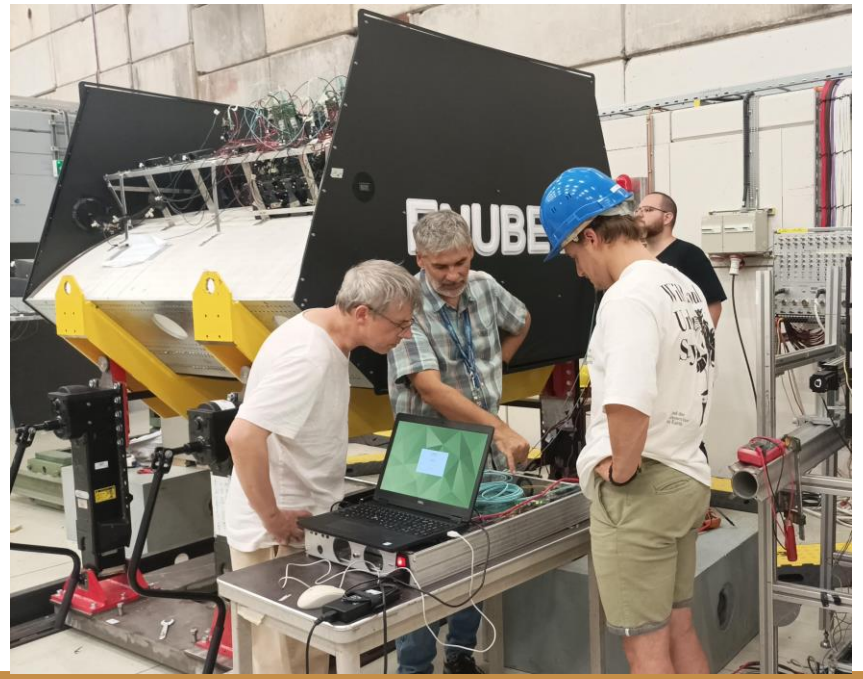
- Through the same network, the user can perform the so-called slow-control
  - Checking important information, such as temperature
  - Enable/disable channels
  - Set trigger parameters
  - Drive I2C/SPI interfaces for FADC components configuration

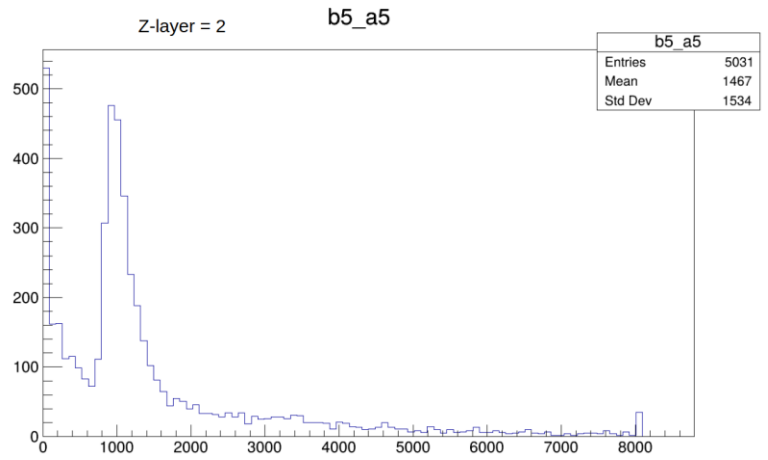
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
INFN

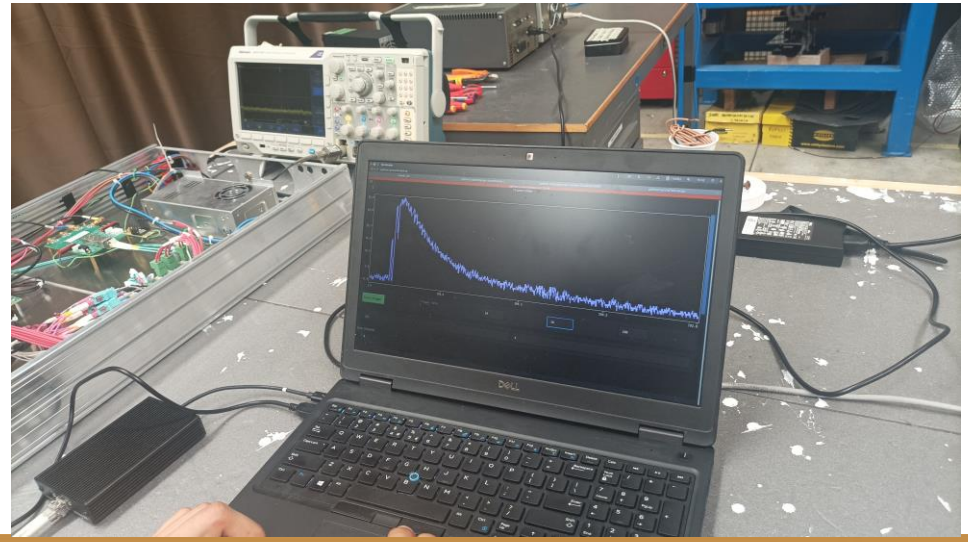




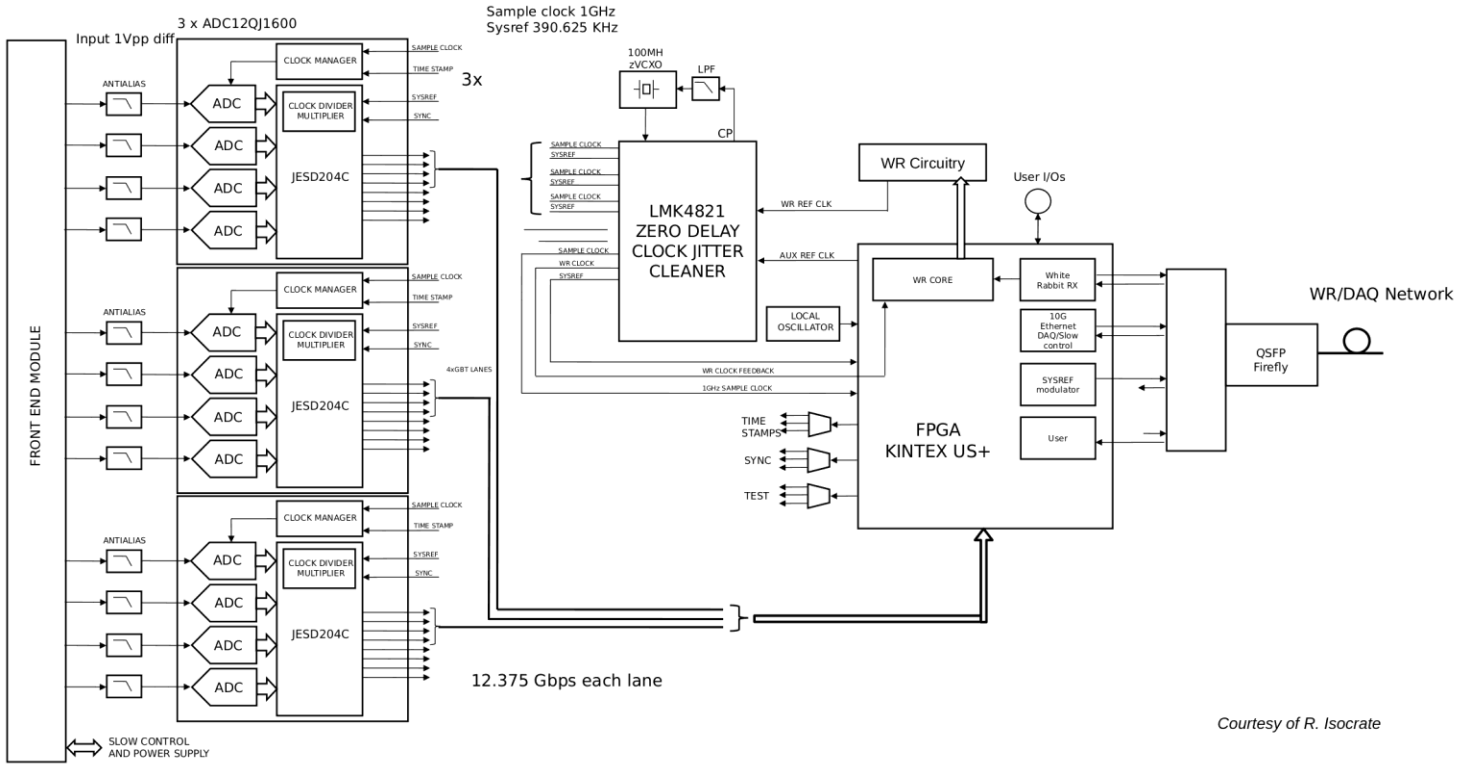
- The full chain setup (FADC + Backend electronics) has been used to acquire 8 channels during a test beam at CERN
- The acquisition has been successful, and expected spectra has been retrieved







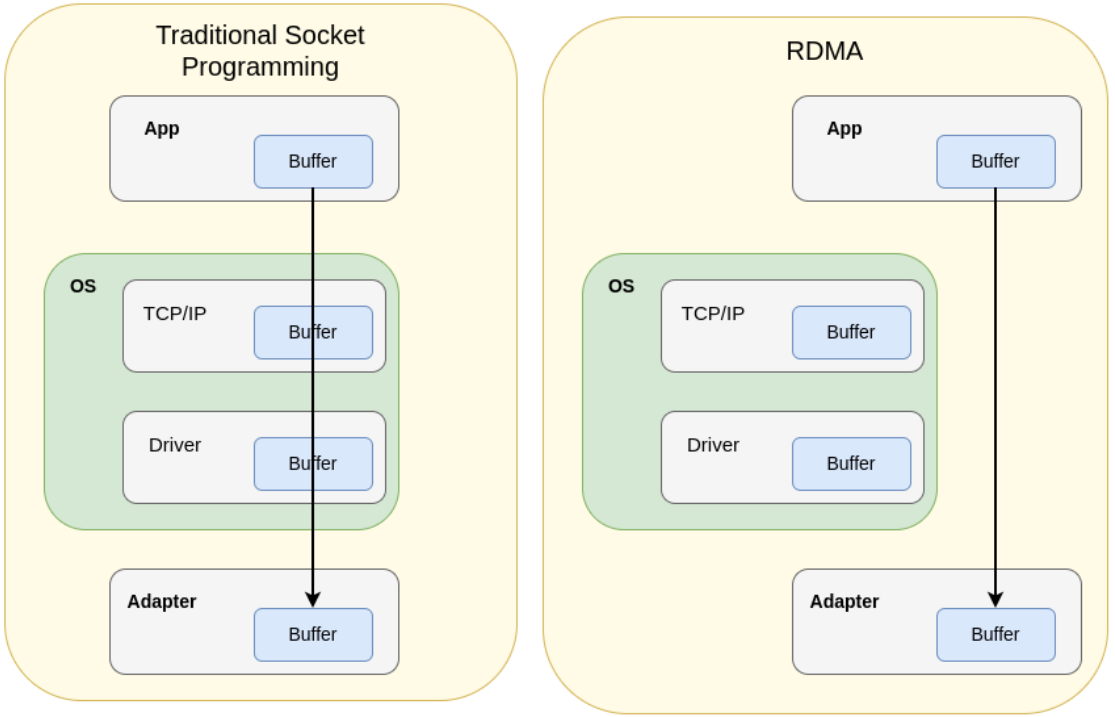
- A newer version of the FADC ver. 1 is in development
- The board will add to the current version the DAQ functionality via 10 GbE
  - Serialized data will go to FPGA. No Firefly 12 Tx for outside
  - FPGA needs to accomodate timing. Timestamps (synchronized between all FADC boards of the experiment) are sent along with each data packet
- DAQ will use the RoCE v2 protocol (RDMA over UDP)



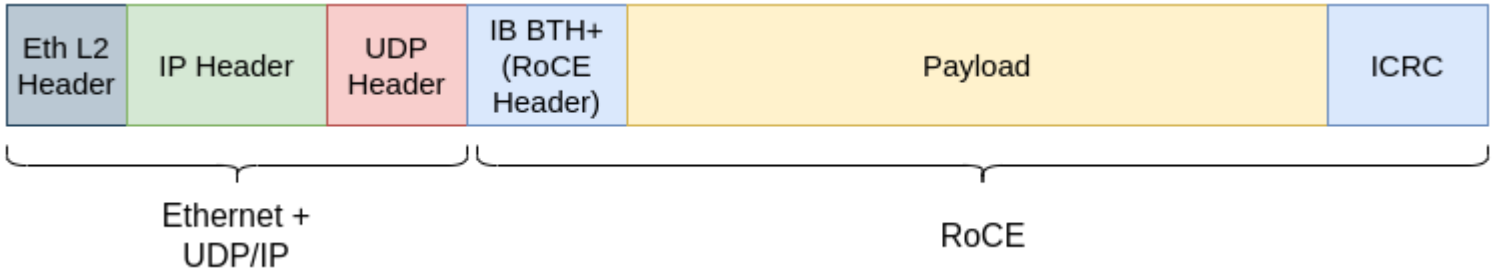
Courtesy of R. Isocrate

- RDMA is an alternative to socket programming which provides better latency and lower CPU load
- The concepts of RDMA are:
  - Transport Offload: inherit overhead usually taken care by operating system, is passed to the dedicated hardware
  - Kernel Bypass: When data is passed from the user application to the hardware, it does not have to pass through the operating system
  - Memory Zero Copy: Avoid copying data to temporary buffers

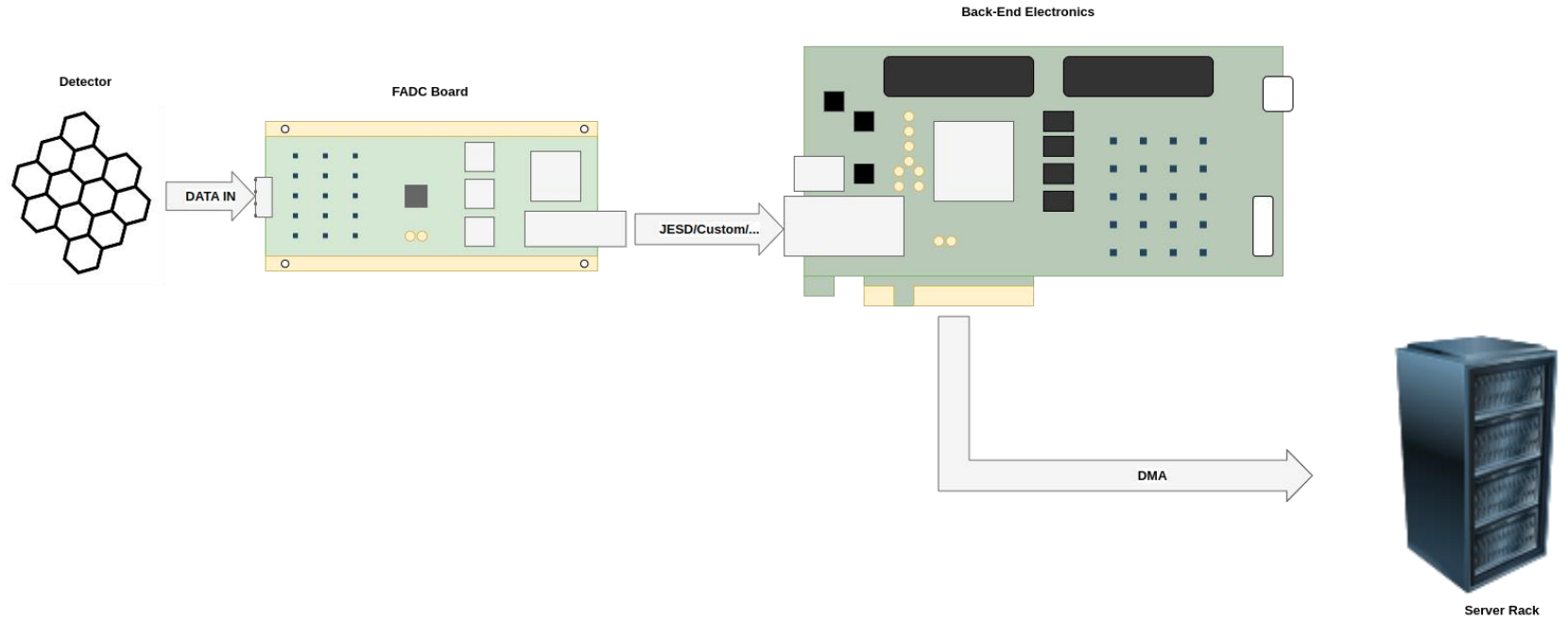
- Basically RDMA allows server-to-server data movement directly between application memory without any CPU involvement
- With RoCE, this efficient data transfer can be used with very low latencies on lossless Ethernet networks



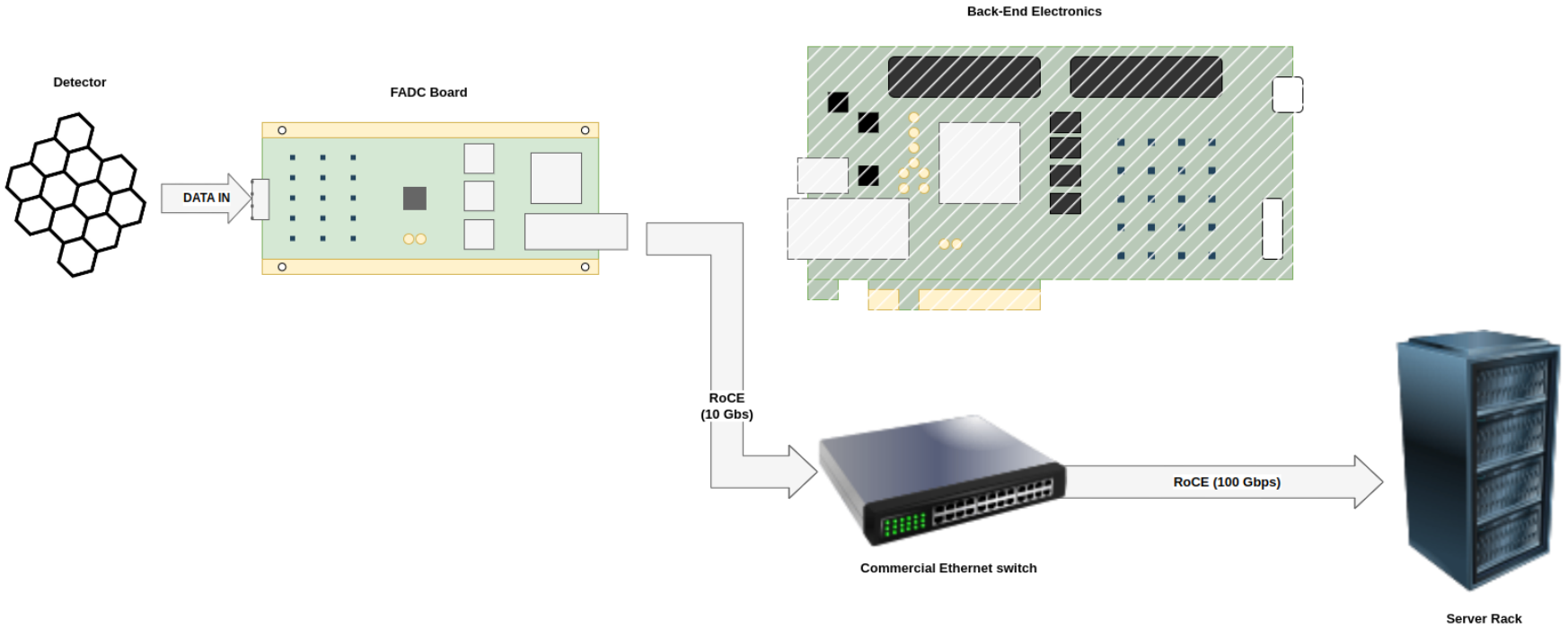
RDMA over Converged Ethernet (RoCE) is a network protocol which allows RDMA over the UDP/IP protocol.



Overall, the packet is a normal UDP packet, so regular Ethernet infrastructure can be used, like regular (but high-performance) Ethernet switches.



- First level trigger implemented in custom hardware backend's FPGAs
  - Massive FPGAs with lots and lots of high-speed links
  - Huge development effort
  - $$$
- High level trigger implemented in software running on farms of commercial computers



- Local trigger on FADC board (trigger architecture to be defined. Possibilty to talk with neighbour boards)
- Front-end data can be issued directly to GPU (GPUDirect RDMA w/ CUDA) for NN/ML trigger for example
  - No development (Off-The-Shelf hardware)
  - Cheap

- We adopted an open-source RoCEv2 RDMA core
- Full RoCE stack in hardware
- Reliability connection supported
- The core exploits RDMA data send and RDMA data receive
- To target small FPGAs, we customized it
  - Removing RDMA receiving data path
  - Squeezing memory usage to fit in a FPGA RAM (few megabits)
  - Adding congestion control to implement backpressure from network switch (ongoing work)
  - Targeting one of the smallest Xilinx FPGA, KU5P, the core uses up about 10/15% of its resources

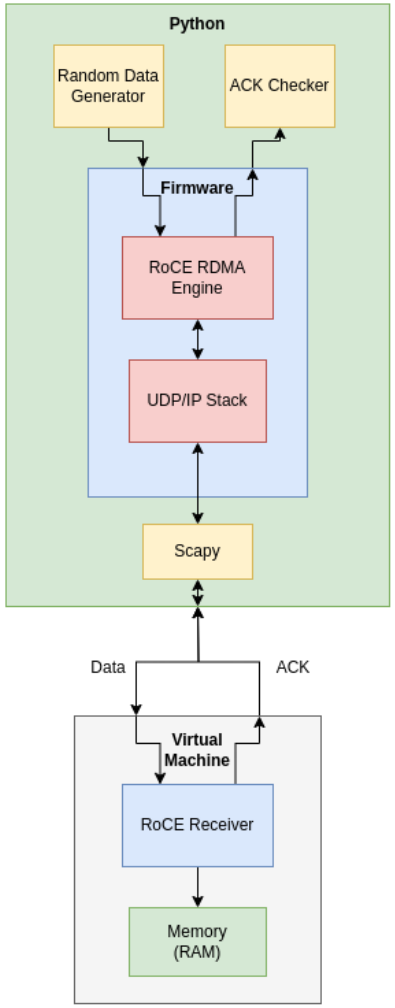
KU5P	Used	Avail.	%
LUTs	33292	216960	15.3
FFs	49712	433920	11.5

- Preliminary results on occupancy look very promising!

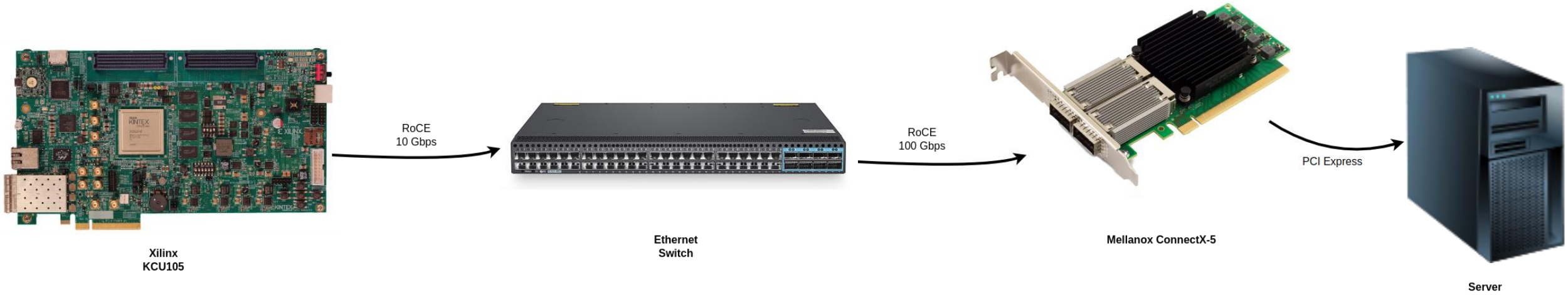
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
INFN

- We designed a full firmware simulation environment
- Random data is feeded to the RDMA core in simulation
- Using external tools- Cocotb- simulation is managed using Python
- With Python libraries- Scapy- network packets can be sent (and received) over to the actual network
- Virtual machine with a Soft-RoCE (software implementation of the RDMA transport) receiver
- Ack responses sent back to the hardware simulation from the virtual machine are received and checked by the testbench
- Check that the payload data sent is actually received and written to the memory of the receiver

- **Simulation has been successful!**



- The custom RDMA core has been implemented on a Xilinx KCU105, which features a Xilinx FPGA
- Target is a server mounting a Mellanox ConnectX-5 Card
- 2x 100 Gbps ports
- **Successfully transferred RoCE packets from the FPGA to the ConnectX-5 card on a server at 10 Gbps!**



- A RoCE-compliant Ethernet switch has been bought and put in-between the FPGA board and the Mellanox card. No significative changes have been detected
  - FS-N8550– 48 x 10Gb/s + 6 x 100Gb/s (price < 5KE)

- Add switch congestion management to RDMA core
- Test with 2 sources and 2 destinations
- Add timing synchronization
- Add FADC stream