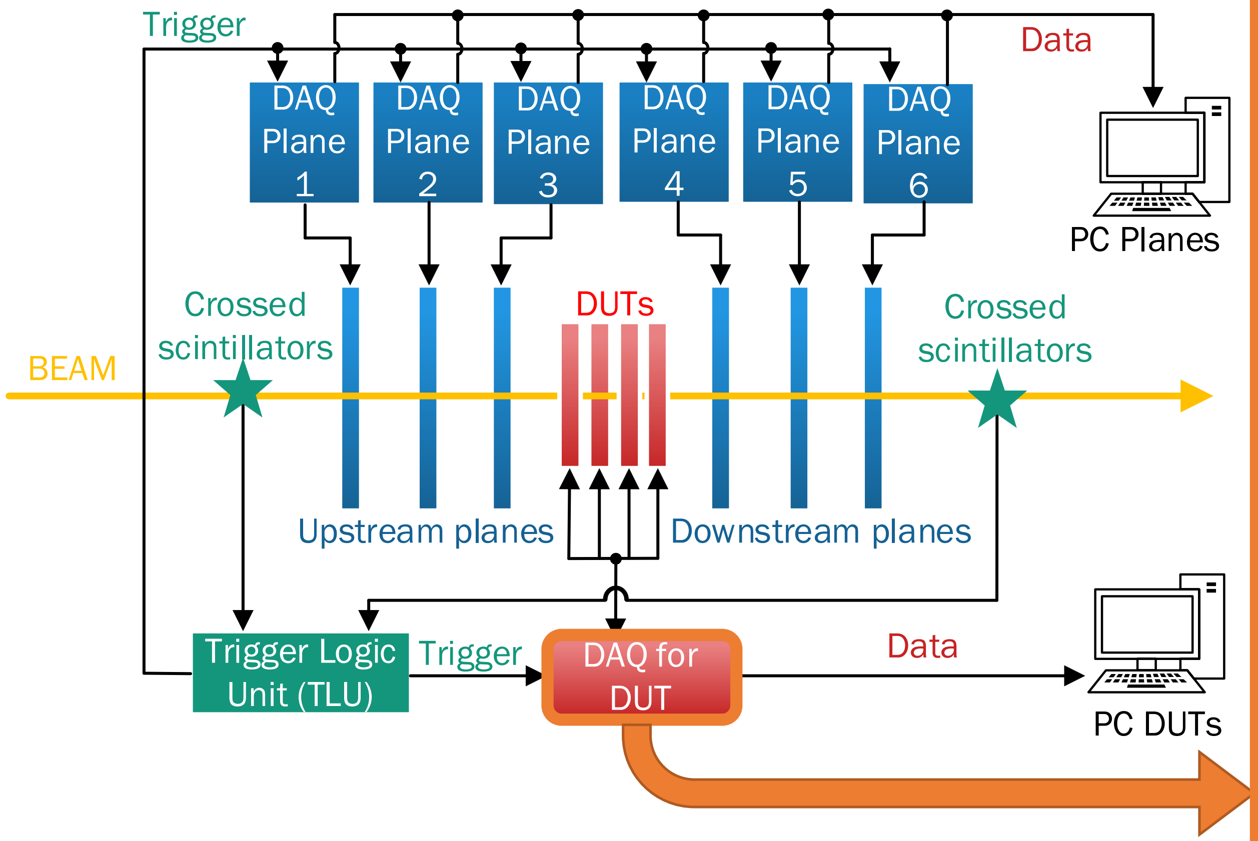
DRAD DAQ System

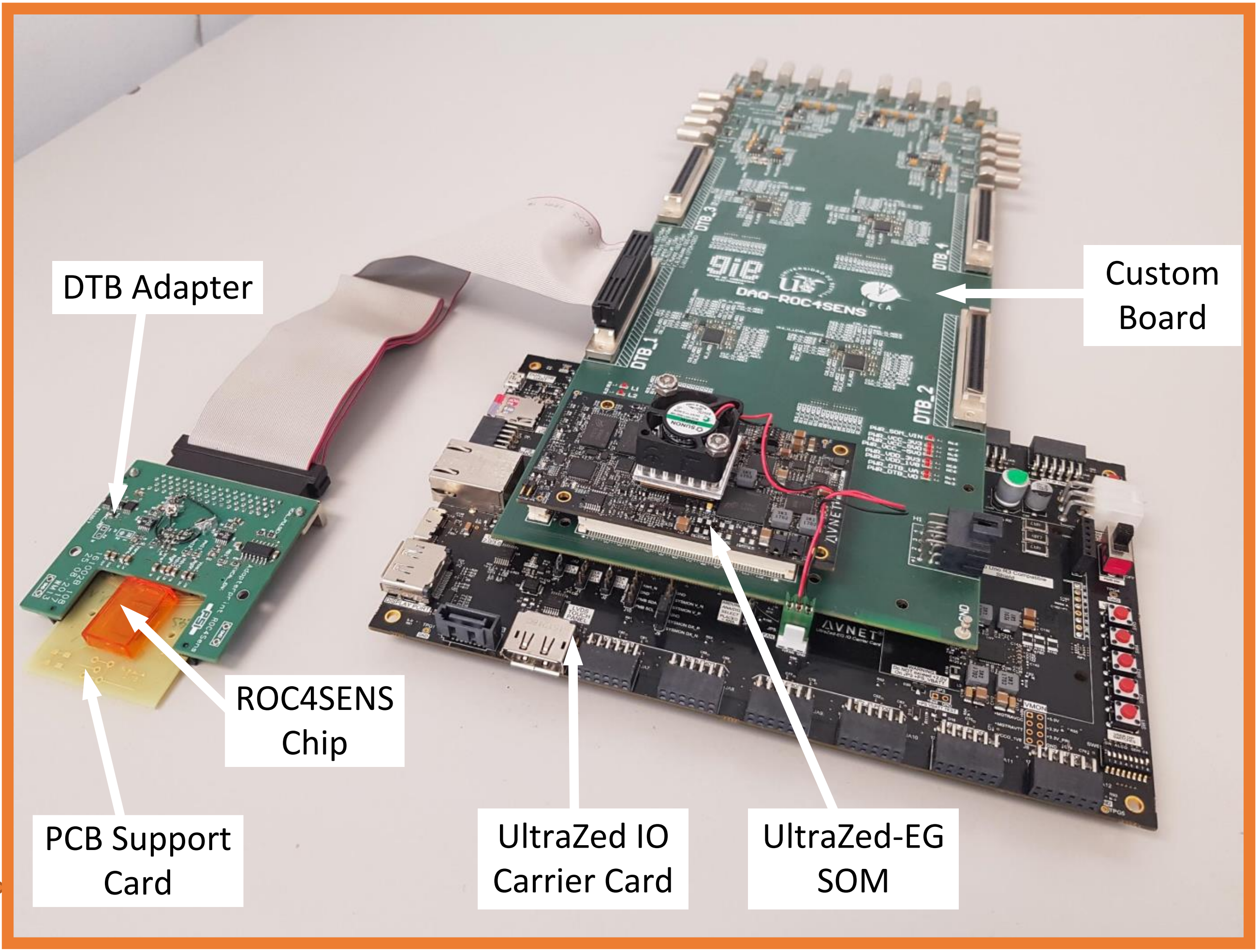
F.R. Palomo[1], J.Jimenez[1], P.Blanco[1], F.Muñoz[1], J.M.Hinojo[1], R.Millán[1], R.G. Carvajal[1]
fpalomo@us.es
[1]Departamento Ingeniería Electrónica, Escuela Superior de Ingeniería
Universidad de Sevilla, Spain

Outline

- General Description and Goals
- Hardware Description
- System Description
- Digital Design
- OS Implementation
- Software and Graphical Remote Client
- Results
- State of the Project and prospectives

General Description: A Telescope DAQ



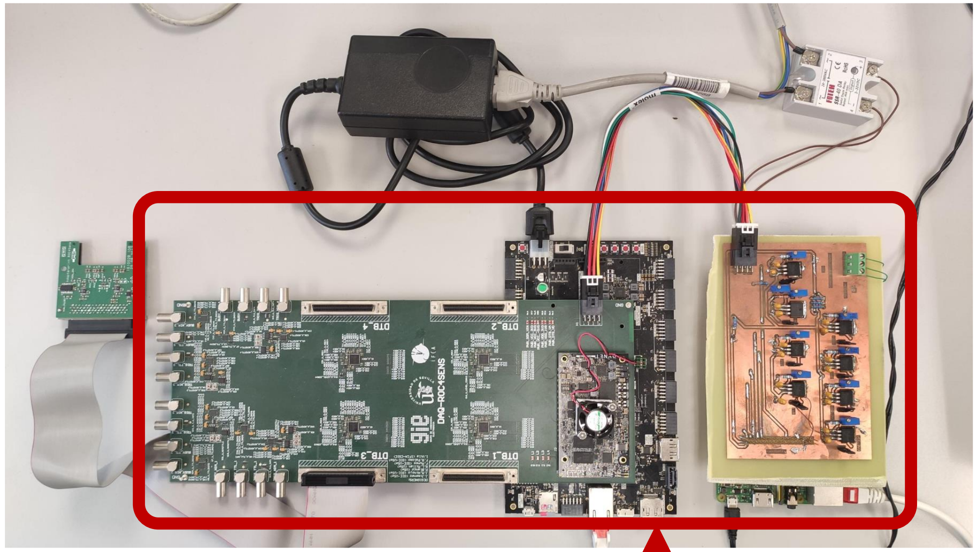


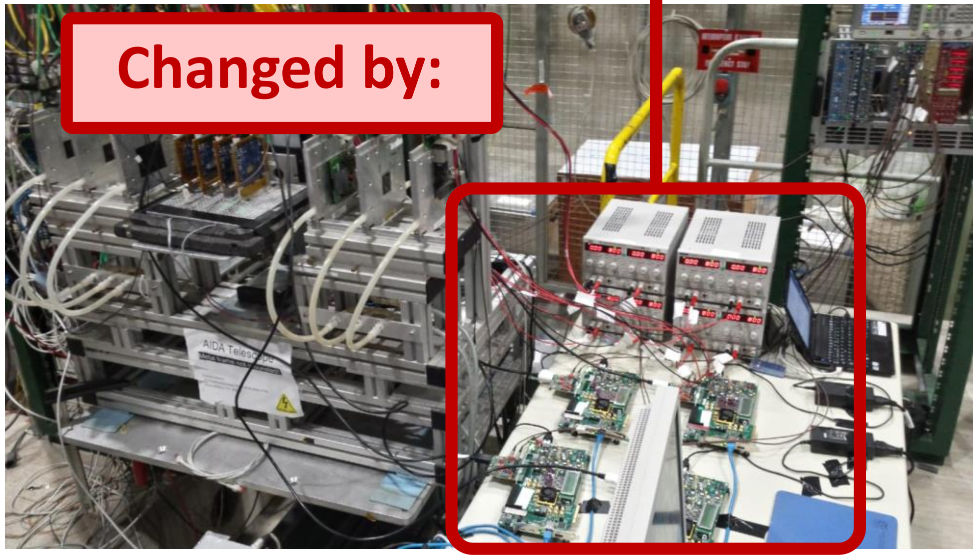
The main idea is to simplify the typical testbeam backend setup (e.g., AIDA2020 telescope) by a **standalone device** based on a powerful novel **MPSoC**, which integrates a programmable logic to implement the readout controllers and a set of multicore ARM processors to provide high-level programmability, such as application based on an architecture client-server, high-bandwidth communications, and high processing capability.

General Description: Goals

Design and implement a DAQ system for various particle tracking detectors.

1. DAQ system is based on a System-on-Chip (MPSoC)
2. Use of initial detector (ROC4SENS) but easily applicable to other detectors.
3. The system is capable of managing up to four detectors simultaneously.
4. Simplifies the experiment setup
5. Synchronize the event log (provided by the TLU) and the correlation of the data from the different detectors



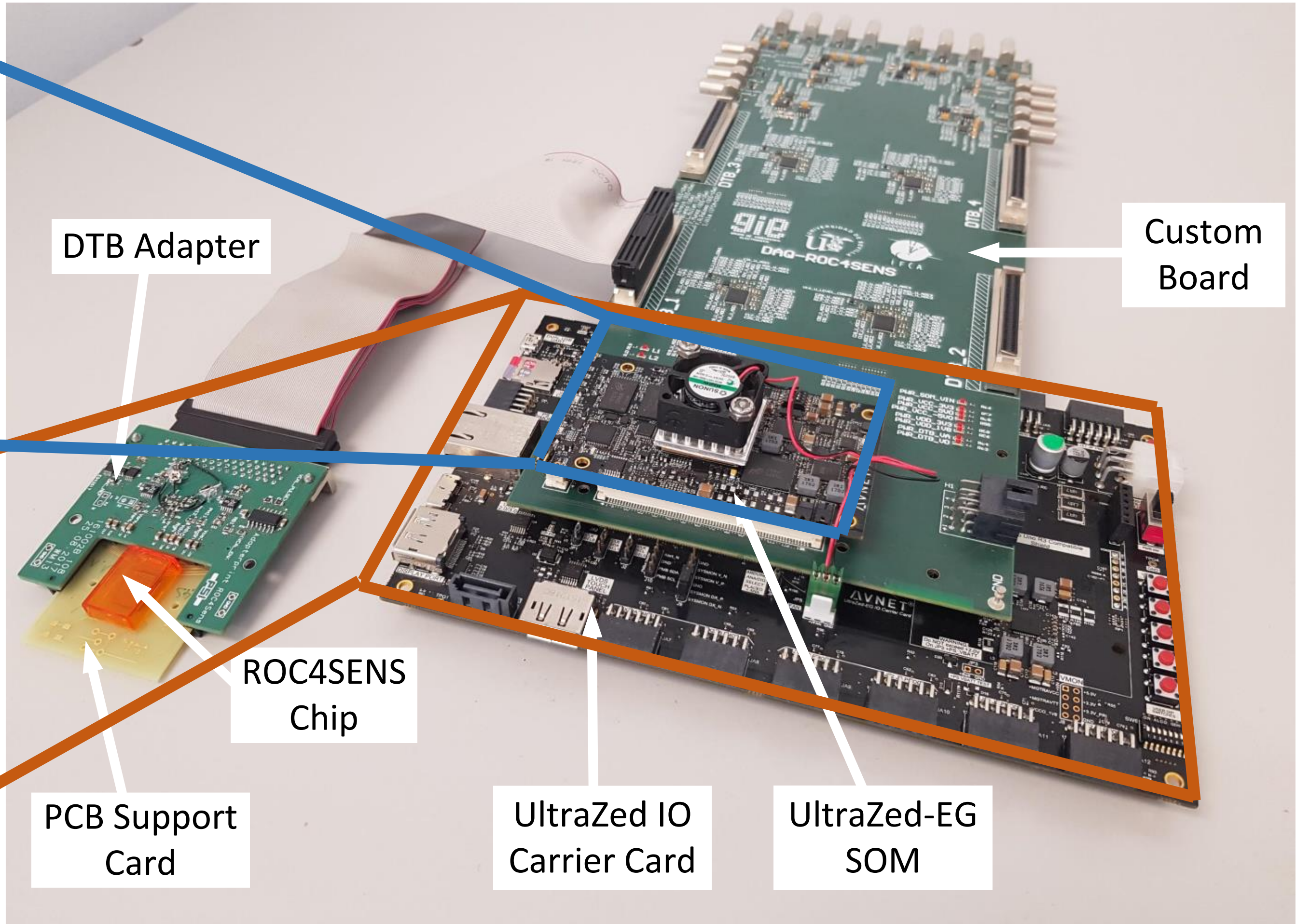


Hardware Description

- **Hybrid FPGA (Zynq Ultrascale+ MPSoC)**
  - ❑ Programmable System (Zynq)
  - ❑ Processing System
    - ❖ 1 ARM A53 quad core (1.5GHz)
    - ❖ 1 ARM R5 dual core (600 MHz)
- **ARM A53 quad core able to run a custom Petalinux**

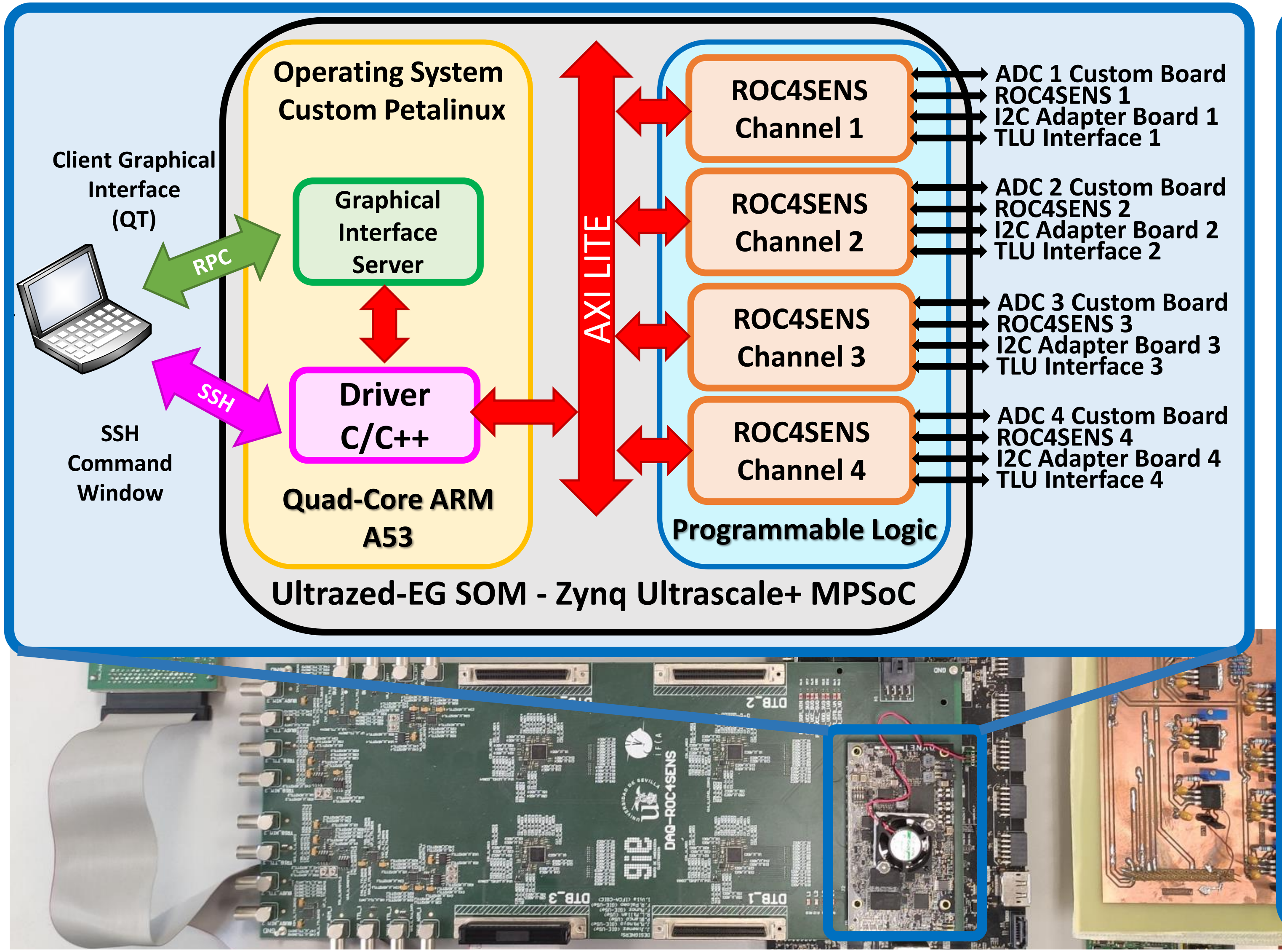
- **UltraZed-EG IO Carrier Card Board:**
  - ❑ JTAG
  - ❑ USB 2.0/3.0
  - ❑ SATA
  - ❑ SDCard
  - ❑ Ethernet Gigabit

- **DTB Adapter (Previous R4S DAQ System)**
- **PCB Support Card (Previous R4S DAQ System)**



The commercial system consisting of the UltraZed-EG SOM and UltraZed IO Carrier Card did not have enough IO pins for connections to the R4S chips, so the Custom PCB was added.

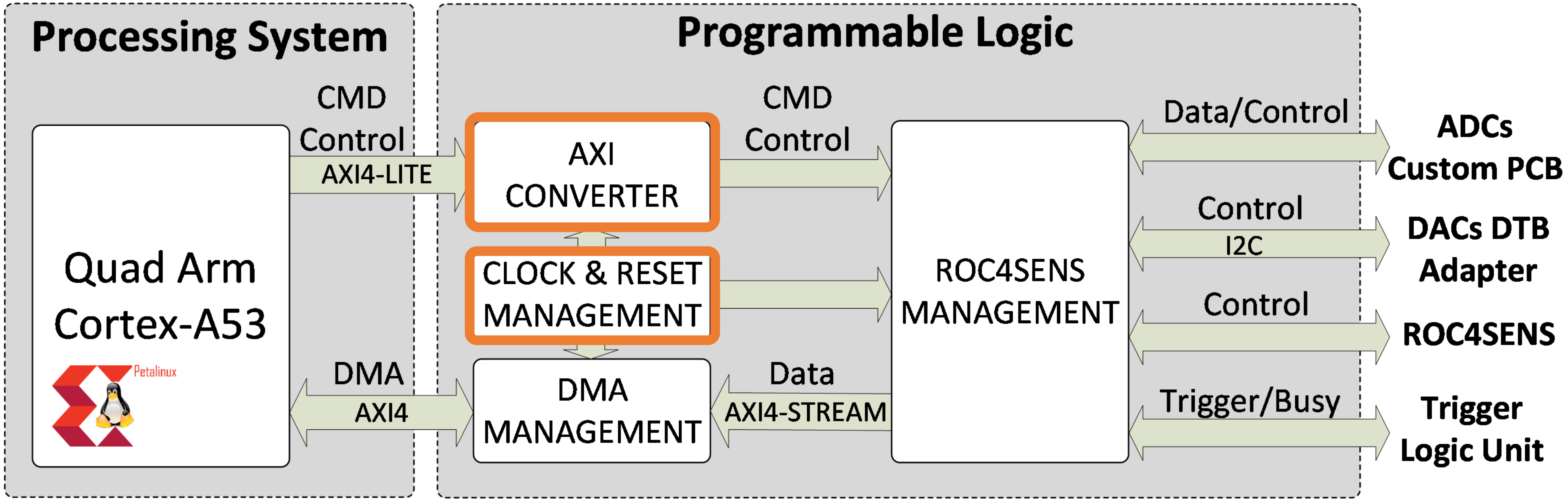
System Description



The operation of the system:

1. Starts with sending commands from the control software to a FIFO memory which is implemented in each R4S Channel.
2. Those FIFOs are processed by each R4S control unit and sent them to the corresponding block, ROC4SENS or I2C Adapter board.
3. TLU captures the event and sends a Trigger signal to corresponding Channel Block.
4. The R4S chip output is captured by the ADC.
5. The digital data is passed to the corresponding Channel Block.
6. Through the direct memory access (DMA), the data go to memory and is now available by means of driver software.
7. The data is available in the server that is accessible by RPC or SSH client.

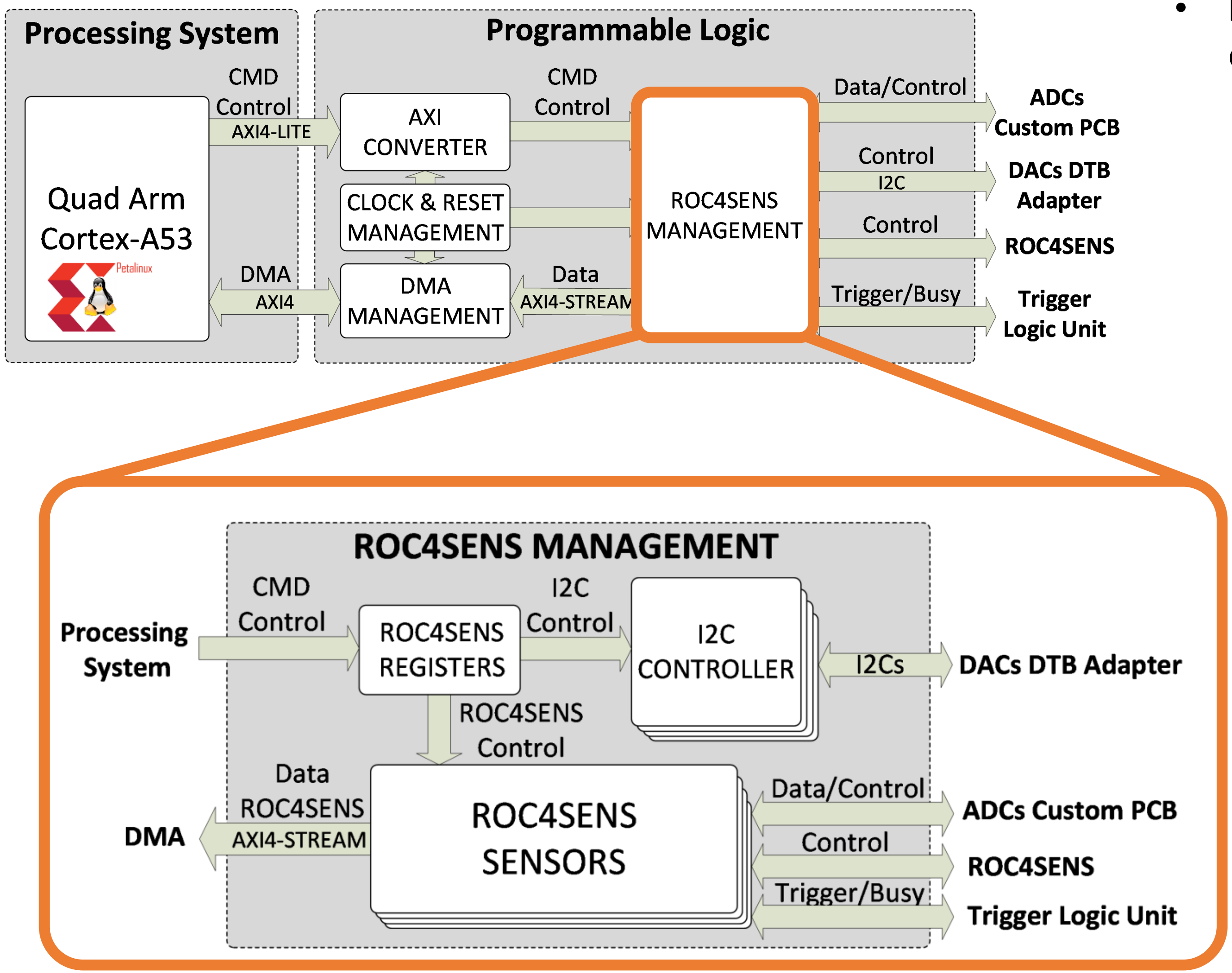
Digital Design: General Description



The implemented programmable logic is composed of four blocks:

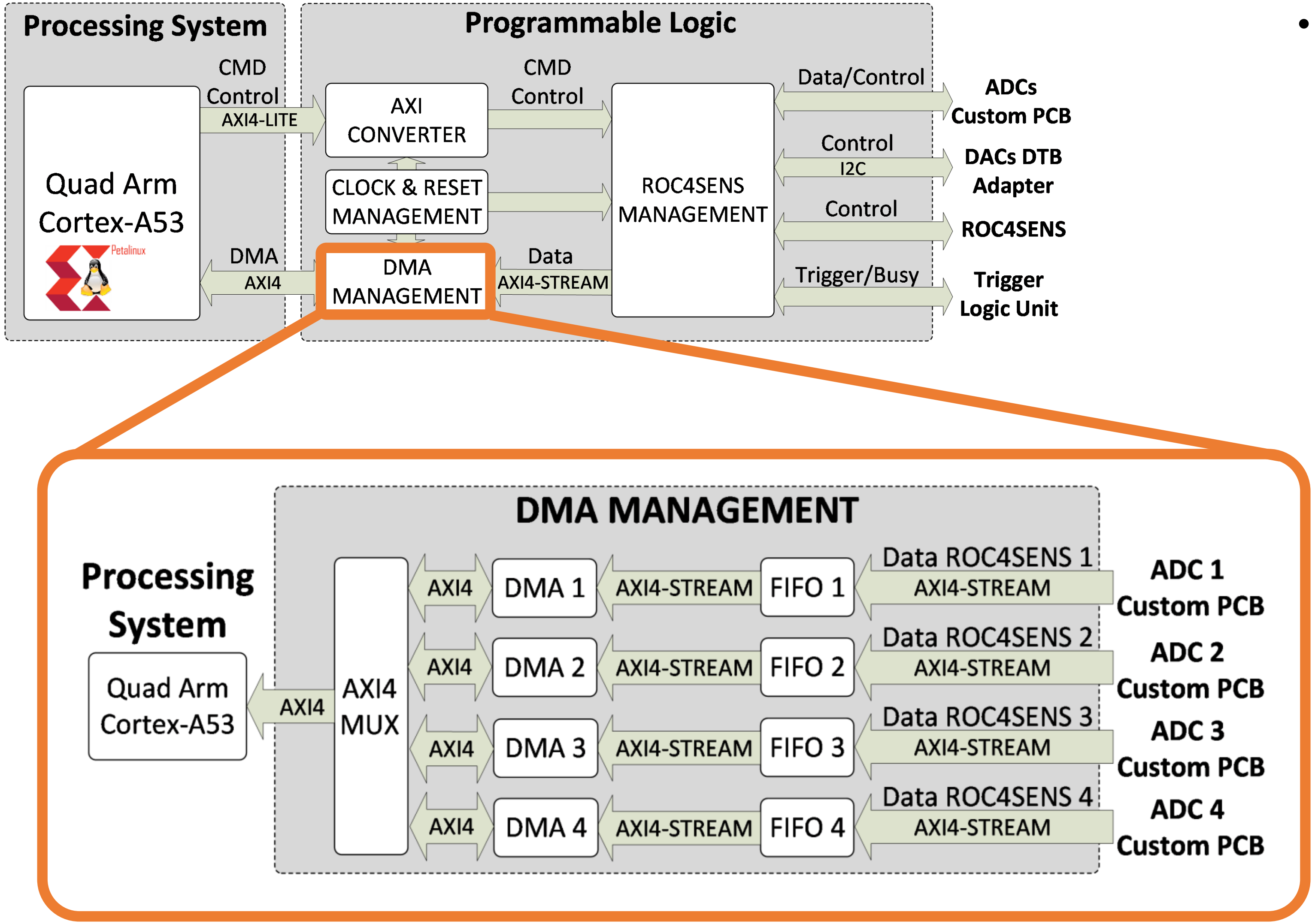
- **Clock & Reset Management block:** generates the different system clocks and the resets associated with each of these clock domains. There are 3 clock domains in the system:
  - ❑ 40 MHz as the main system clock, which corresponds with the CERN LHC frequency
  - ❑ 160 MHz for the R4S chips control signals
  - ❑ 100 MHz for the OS and DMA management in ARM Cortex-A53 (DMA Data Bus).
- **AXI Converter:** which converts the control signal provided by OS (AXI4-Lite) to the interface format of the R4S Management block.

Digital Design: R4S Management Description



- **R4S Management:** most important block in the system as it is in charge of management and calibration of the R4S chips.
  - Analyze the commands in the **R4S register block** and moves from the OS management clock domain (100 MHz) to the main system domain (40MHZ) via FIFO.
  - *Commands for I2C* **(I2C Controller):** implements an I2C master for communication with the DACs in Adapter Board (generation of reference voltages for R4S chips).
  - *Commands for R4S* **(ROC4SENS sensor block):** this block performs several tasks:
    1. Generation of the R4S chip signaling for read and/or calibration.
    2. Changing the main system clock domain (40MHz) to clock used by the R4S sensor control signals (160MHz).
    3. Trigger (TLU) control.
    4. Manages the activation and clock generation of the ADCs.
    5. After ADCs conversion, capture data output from the R4S chip.

Digital Design: DMA Management Description



- **DMA Management:** after capturing the data, the ROC4SENS Management block passes the data to the DMA Management block. This block is responsible for:
  - ❑ Storing the data obtained from the ROC4SENS and sending them through the DMA to the operating system.

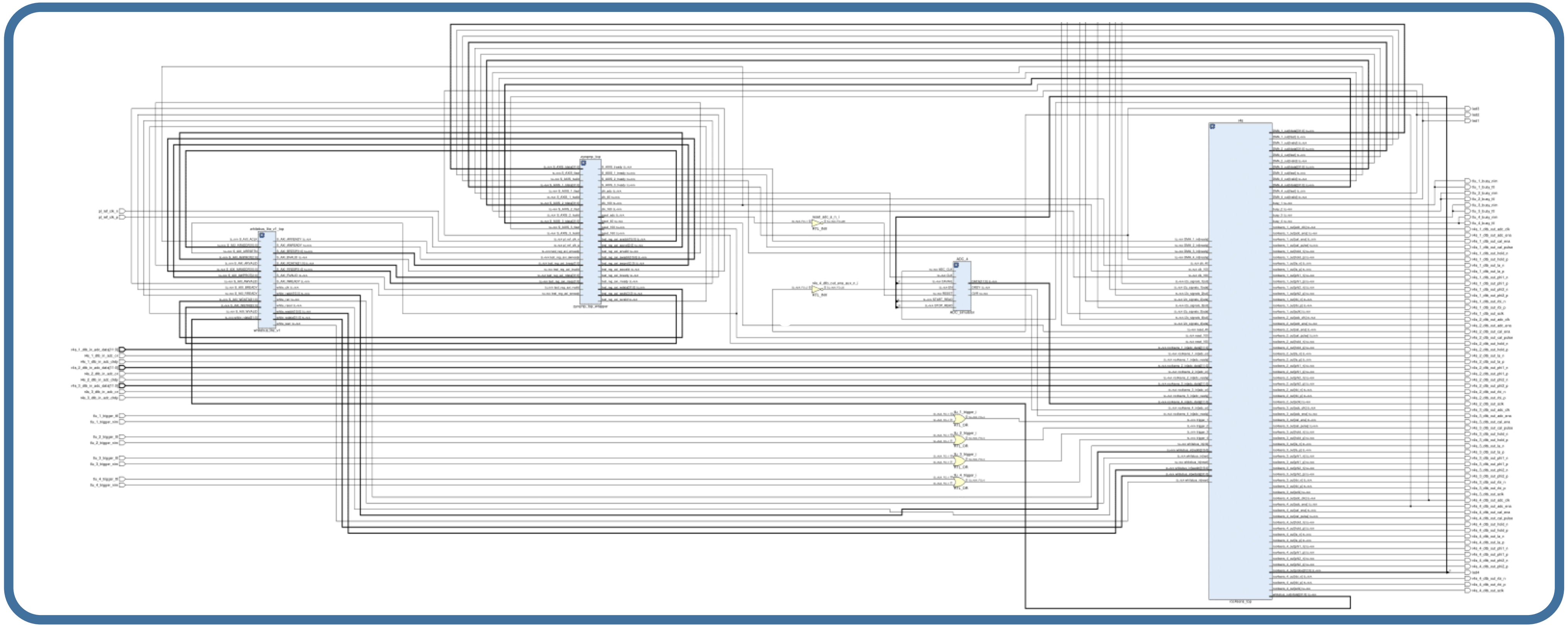
    A DMA has been used together with a FIFO for each sensor in order to have a modular and independent system for each ROC4SENS chip and to simplify the software for data handling.

  - ❑ The data obtained is stored in a FIFO memory, which is also responsible for solving the clock domain crossing. The data source operates at 160 MHz while the OS clock domain works at 100 MHz.

Digital Design: Implementation

The programmable logic was developed in VHDL and Verilog (PSI legacy code) using the *Xilinx Vivado 2020.2* tool.
The design includes:

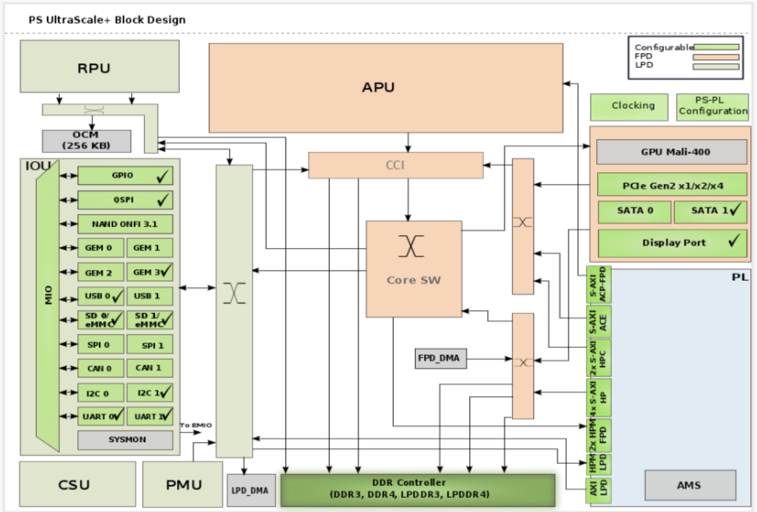
- Design in VHDL including wrapped Verilog code.
- Multiple clock domains (cross-domain interchange capability).
- Use of Xilinx IP blocks (FIFO, Clock Generator, DMA, ARM Microprocessor configuration)



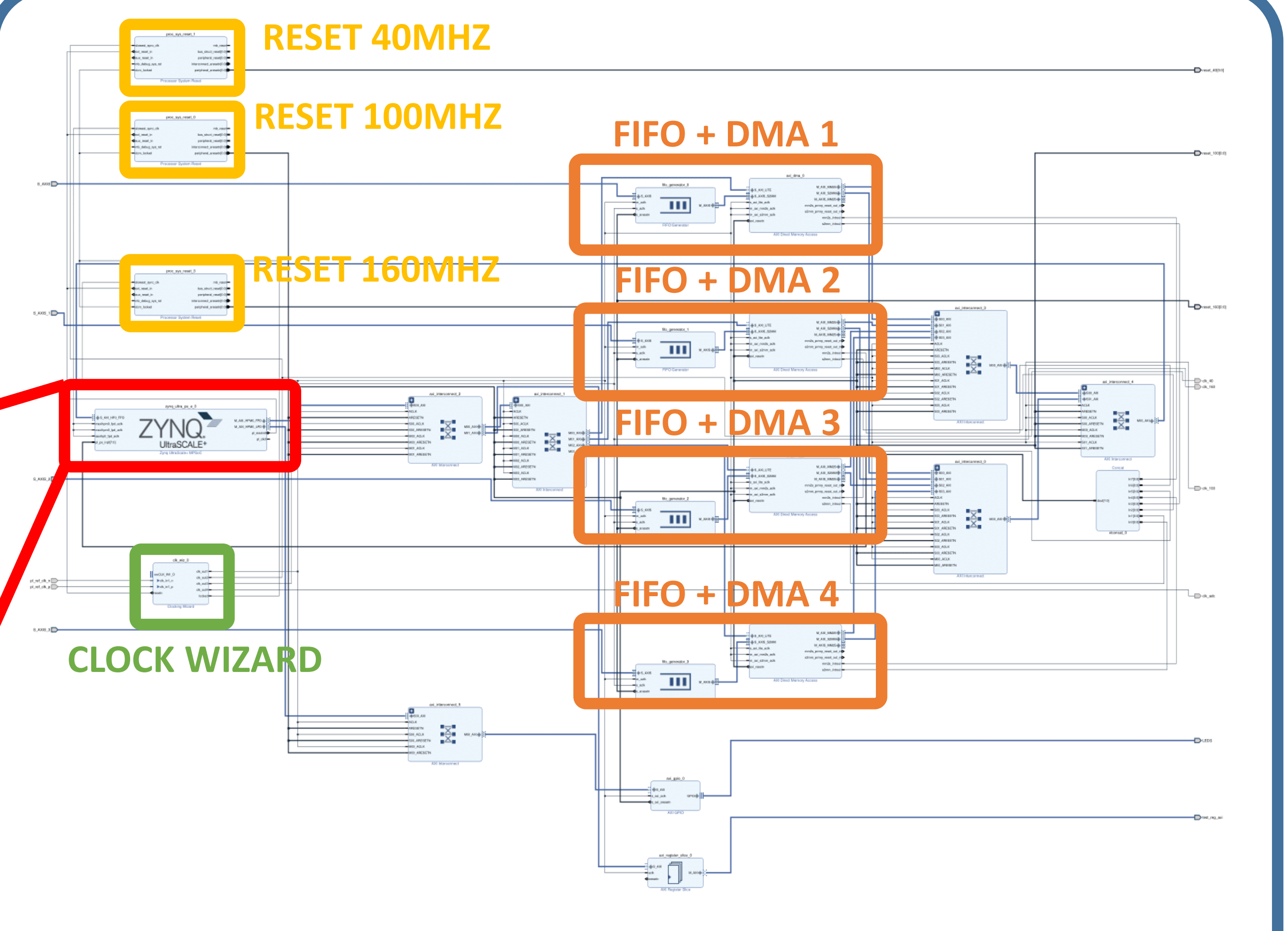
Digital Design: Block Design Tool

We used the Vivado Block Design tool to include Xilinx-specific IPs and configuring the Microprocessor. The principal IPs block are:

- Clocking Wizard (6.0)
- FIFO Generator (13.2)
- AXI Direct Memory Access (7.1)
- Processor System Reset (5.0)
- Zynq Ultrascale+ MPSoC (3.3)



Zynq Ultrascale+ MPSoC Configuration Block

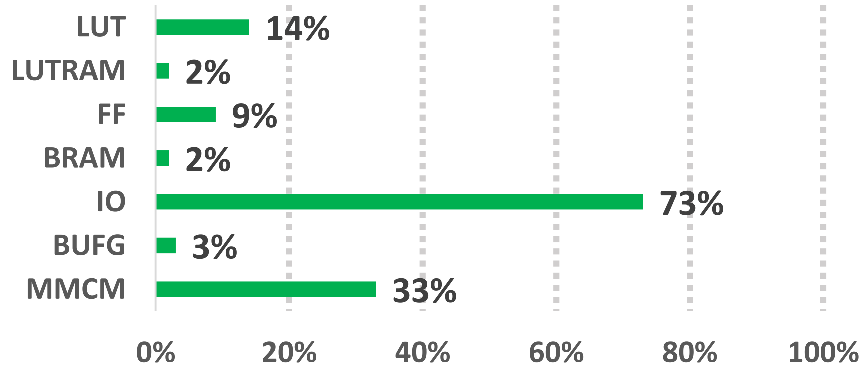


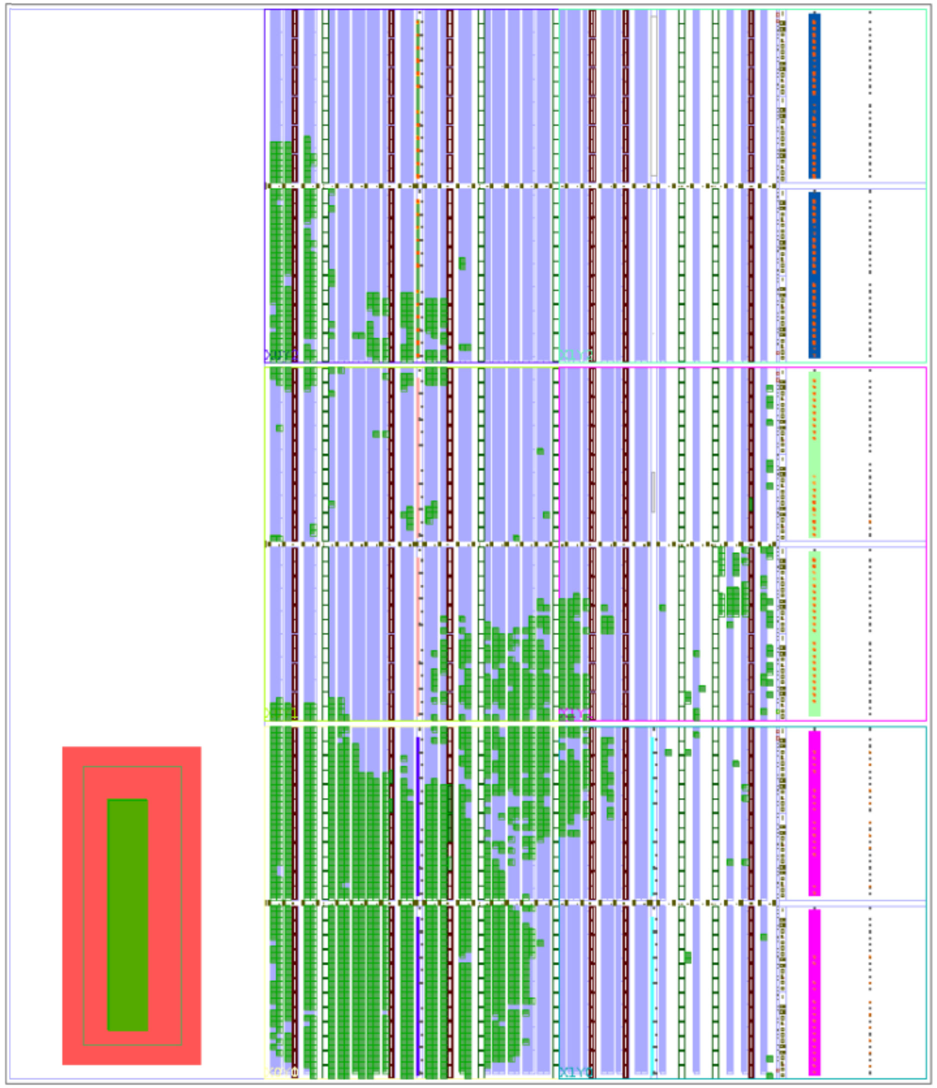
Block Design Tool

Digital Design: Resources used

FPGA Utilization resources			
Resource	**Utilization**	**Available**	**Utilization %**
LUT	14052	70560	19.91
LUTRAM	780	28800	2.71
FF	19786	141120	14.02
BRAM	18	216	8.33
IO	146	180	81.11
BUFG	4	196	2.04
MMCM	1	3	33.33

Utilization (%)





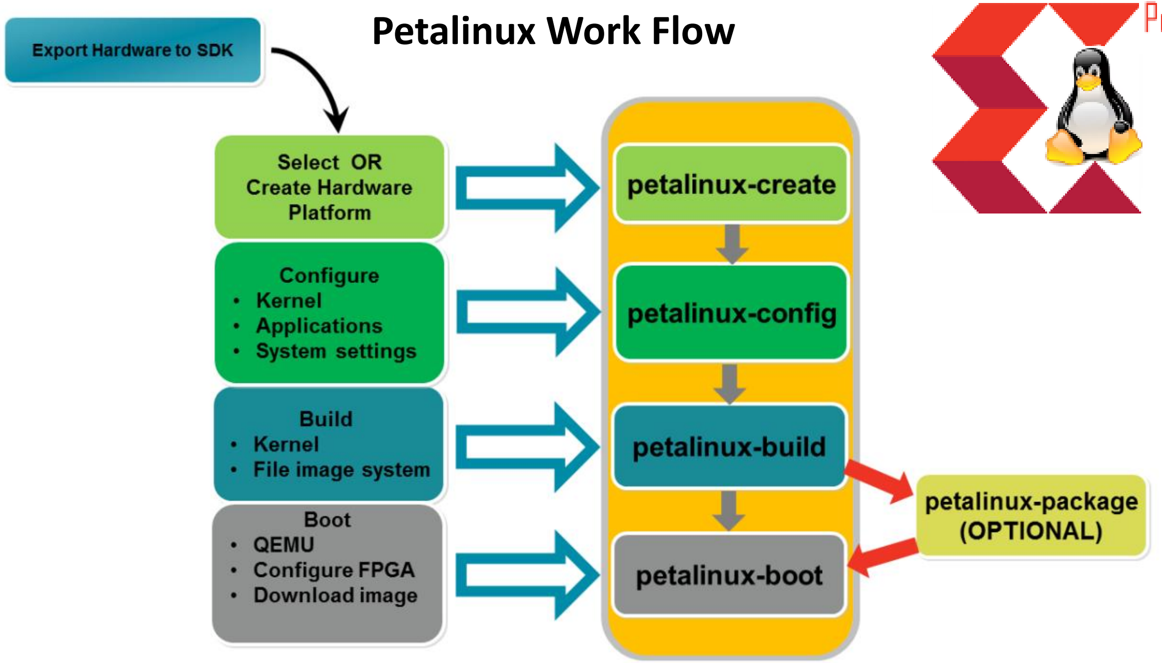
Physical Implementation

OS Implementation

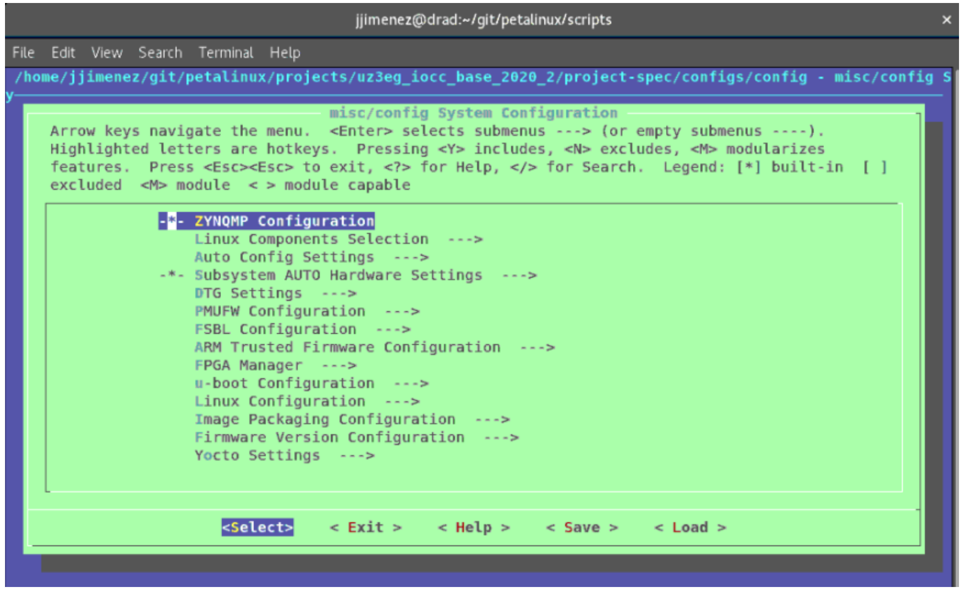
The operating system corresponds to a **Petalinux distribution**, compiled using tools provided by Xilinx.

- Petalinux tool makes use of the **bitstream** and the **hardware description** file generated in Vivado after the implementation of the programmable logic.
- To automate the process of operating system compilation and deployment of the operating system, a **script (shell)** obtained from **AVNET** was customized to perform OS compilation.
- The operating system is a full **Linux** system and integrates everything necessary for communication with the outside via **Ethernet (SSH).**

Petalinux Work Flow



Petalinux Config Tool



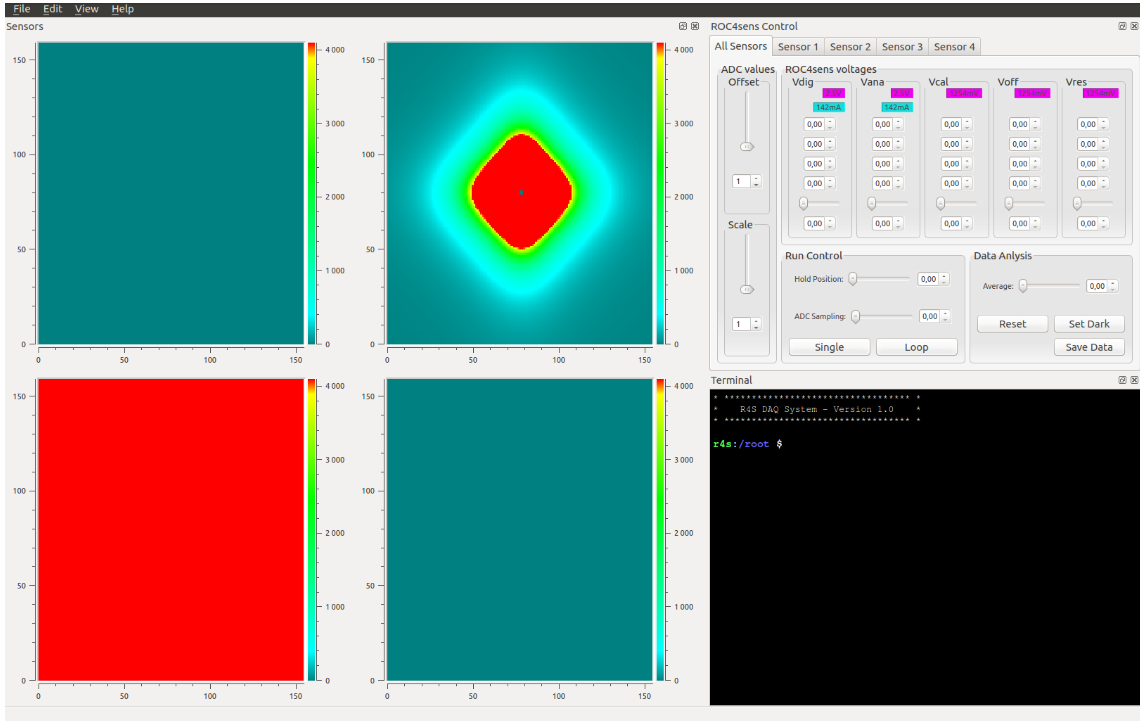
Software and Remote Graphic Client

For communication with the programmable logic, a **C/C++ software driver** was implemented. Their functions are:

- Sending commands to the R4S chips for calibration and matrix reading
- Sending I2C commands to the DACs of the Adapter Board to generate reference voltages for the R4S.
- Initialization and configuration of the DMAs
- Obtaining the memory data

Finally, a **graphical interface (still in testing)** was developed in QT (5.12.3) that allows the user to interact from any remote host with the DAQ:

- Runs on a remote laptop
- The Client Communicates with the backend by Remote Procedural Calls (RPC) and by UDP/TCP data transfer
- Four pixel displays
- Full set of configuration controls
- Shell available.
- Full Duplex (8 ports, 2 associated to each ROC for 2 simultaneous data transfer, triggered and not triggered)
- Client/Server and Server/Client if necessary (architecture duality)



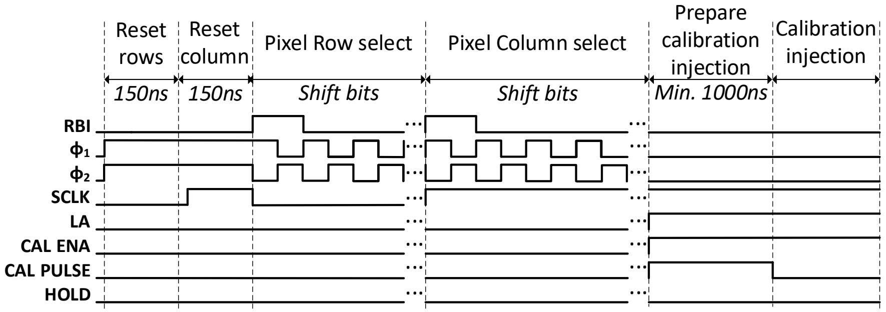
Graphical Interface (in testing)

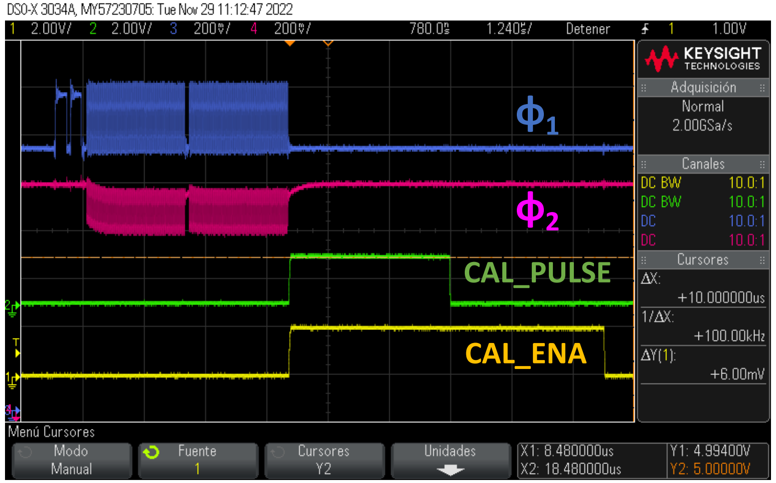
Results: Calibration

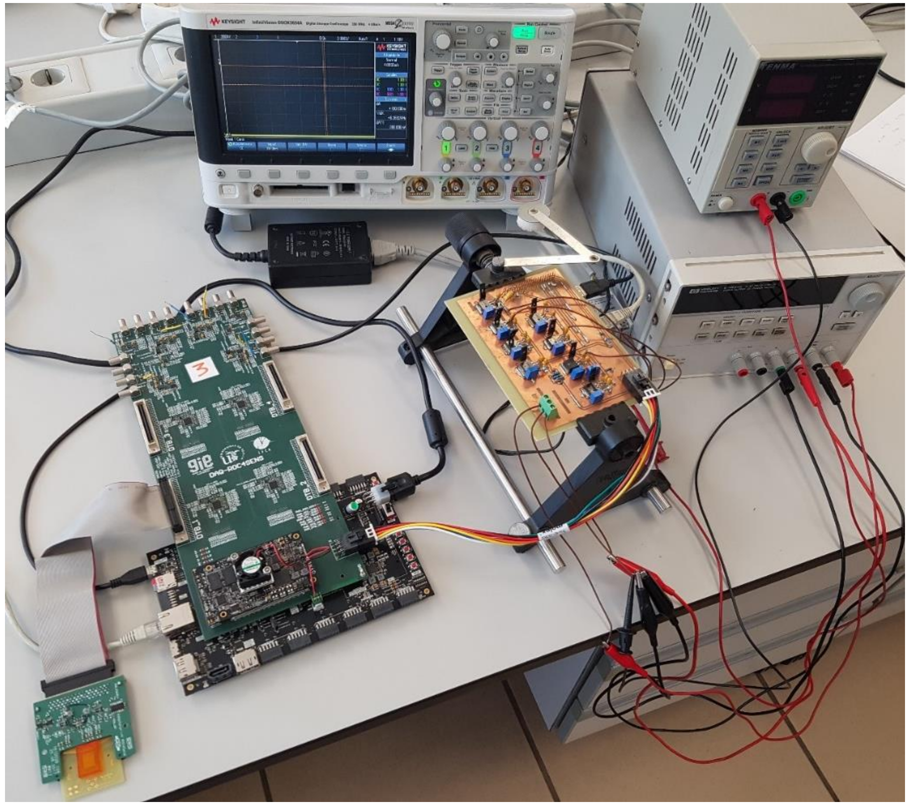
In order to validate the operation of the proposed DAQ, a **calibration test** and **readout matrix test** (with calibration mode) has been carried out with the ROC4SENS chip.

These measurements have been made with only the ROC4SENS read-out chip without connecting any detector. Calibration mode is used.

- **Calibration TEST**





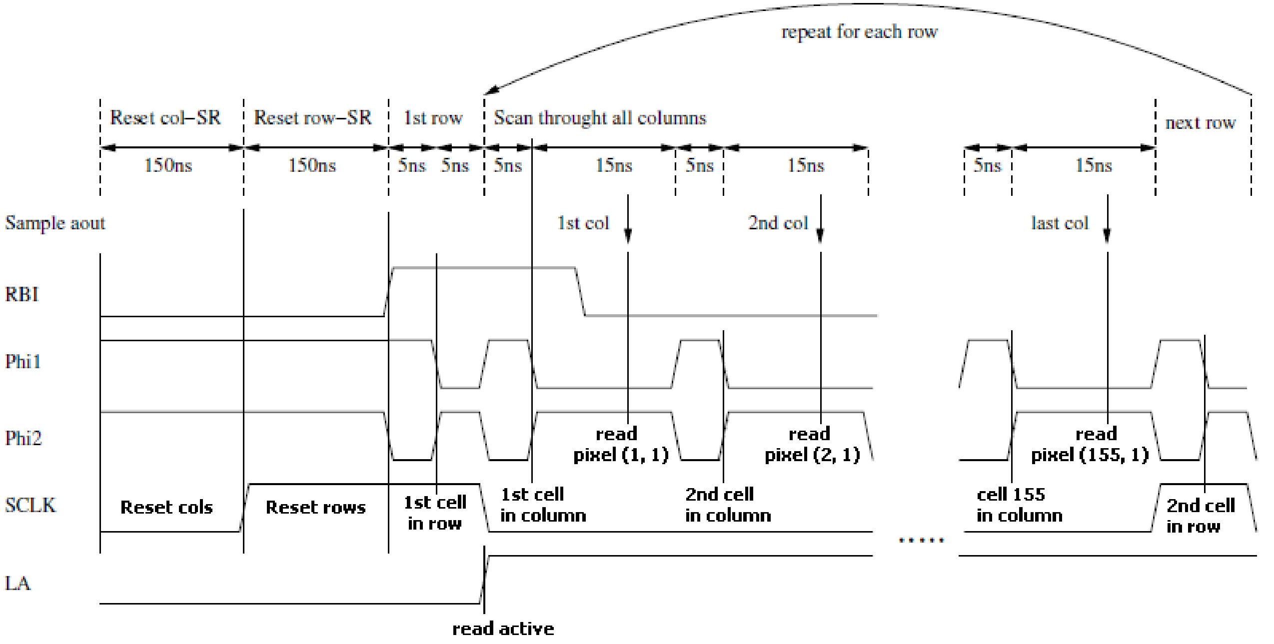


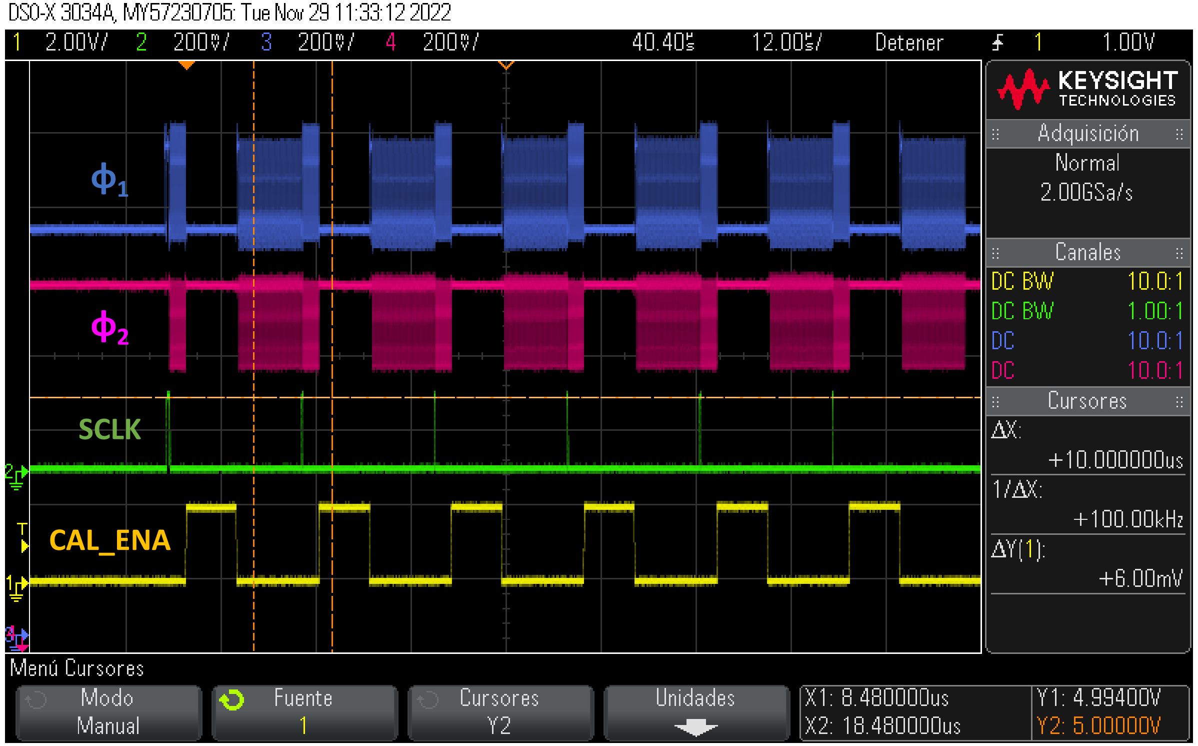
Results: Readout Matrix

In order to validate the operation of the proposed DAQ, a **calibration test** and **readout matrix test** (with calibration mode) has been carried out with the ROC4SENS chip.

These measurements have been made with only the ROC4SENS read-out chip without connecting any detector. Calibration mode is used.

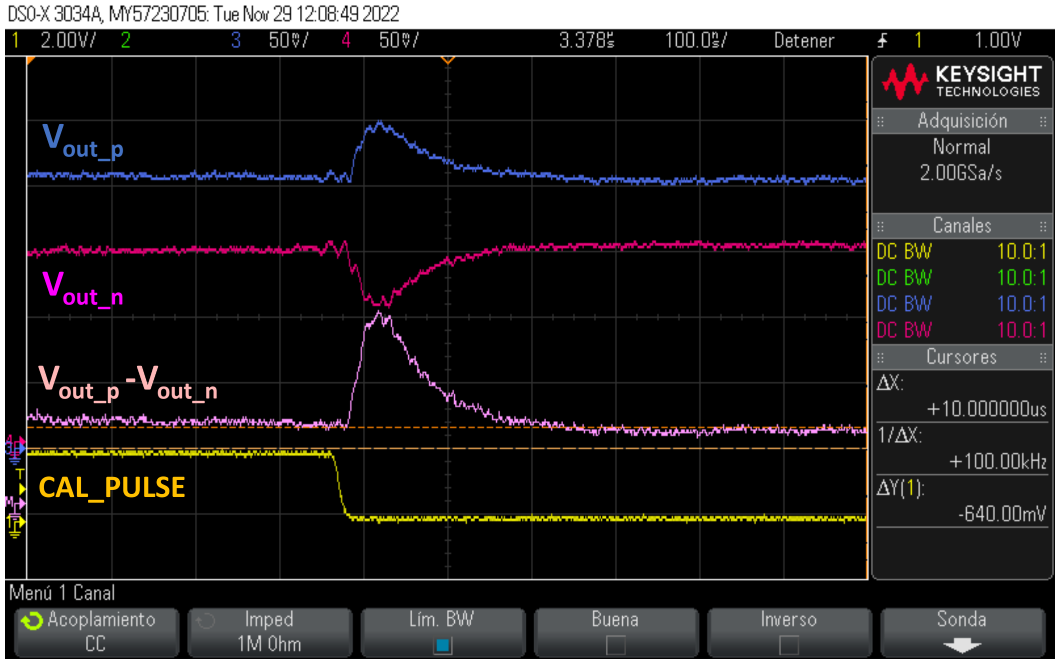
- **Readout Matrix (with Calibration mode)**



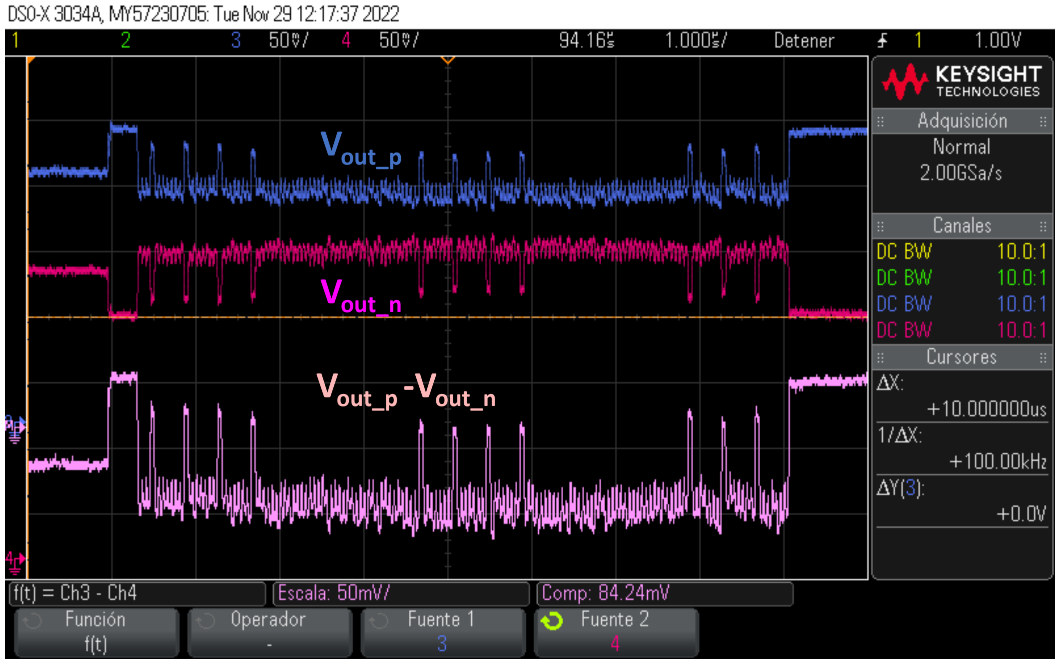


Results: Output Signals

- **Calibration TEST (one pixel)**



- **Readout Matrix (one row, with Calibration mode)**



DRAD is working and we got signals from the test pulse circuit in every R4S pixel

Next Steps

- ❑ DAQ-Roc4Sens at 85% of finalization
  - ❖ Complete the ADCs integration phase
  - ❖ Complete integration QT Interface
  - ❖ Test a fully assembled hybrid pixel detectors from PSI (RoC4Sens+Sensor) in a lab (with pulsed laser or particle beams)
- ❑ Assess EUDAQ software integration

Conclusions

- ✓ Significant simplification of the measurement setup.
- ✓ Developed system is easily adaptable to any pixel detector arrangement.
- ✓ High Data Transfer Rate that will allow data processing in real time
- ✓ Driver fully developed. Third-party software can make use to acquire data from the system

Acknowledgment

- Grant US-1266227 funded by

Junta de Andalucía

Unión Europea
Fondo Europeo
de Desarrollo Regional

Andalucía se mueve con Europa

- Grant “Contrato Predoctoral PIF VI Plan Propio de Investigación y Transferencia Convocatoria 2020” funded by

Universidad de Sevilla

Thanks for your attention

fpalomo@us.es

Chip Read-out: DAQ Previous System

The actual DRAD implementation works with the ROC4SENS readout chip from PSI. It is **versatile enough to be adapted to other ROCs**.

In addition, the DRAD system represents an **update of the initial DAQ** that was developed for ROC4SENS by PSI.

- The ROC4SENS is wire bonded directly to a carrier board, with this PCB support card.
- The adapter card board provides analog and digital supply voltages and also reference signals, generated by means of two DACs MAX584.
- The DAQ system (called DTB-DAQ) commands the adapter card board by a ribbon cable.

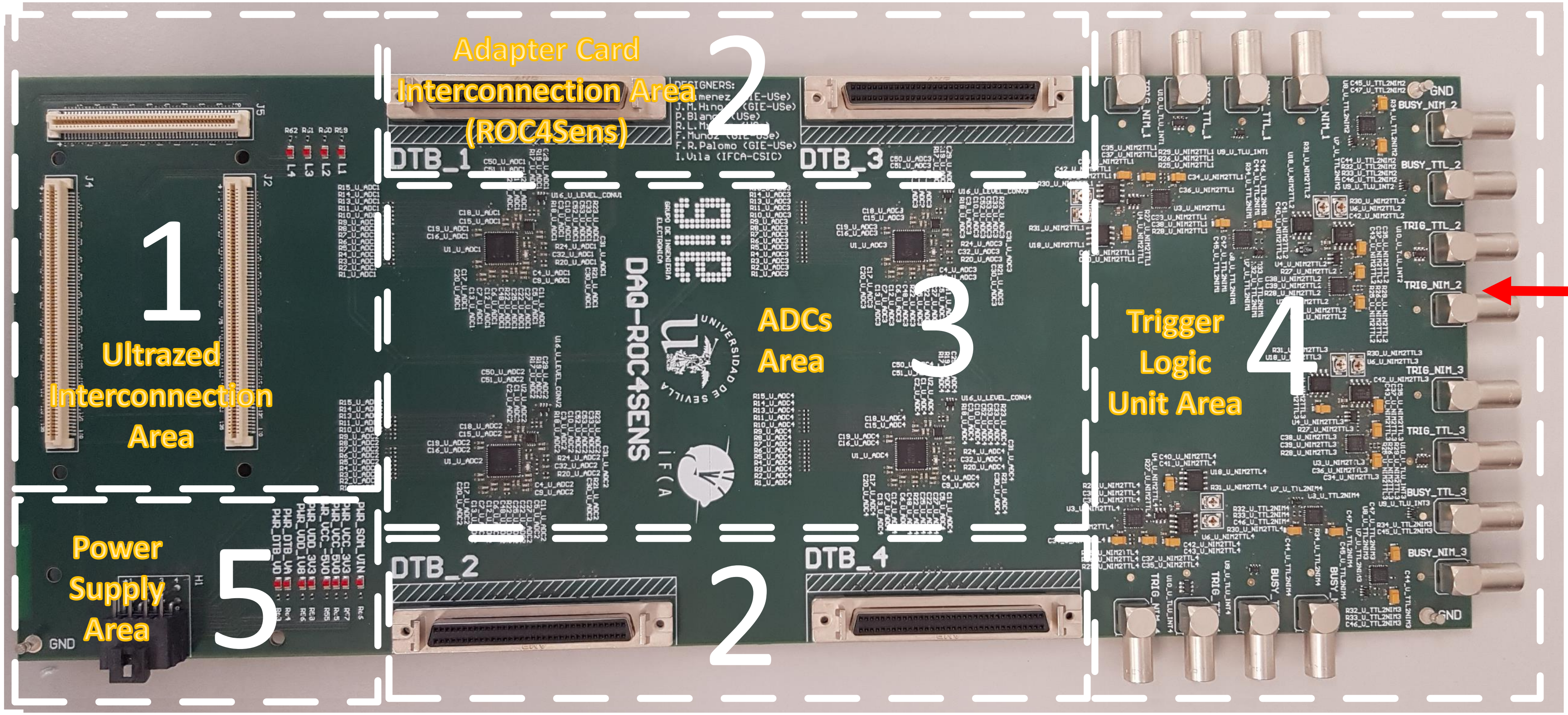
Implemented DAQ vs Previous DAQ

PAUL SCHERRER INSTITUT

- Difficulty in synchronizing data between the different sensors to reconstruct the trace and simplification in the assembly.
  - Previous DTB-DAQ: only allows control of a single ROC4SENS
  - Developed DAQ: control 4 sensors simultaneously
- ADC limitation
  - Previous DTB-DAQ: Maximum Frequency 100 MSamples/s
  - Developed DAQ: ROC4SENS chip supports 160MSamples/s
- Limited FPGA resources
  - Previous DTB-DAQ: Altera Cyclone III, simple FPGA
  - Developed DAQ: SoC including a microprocessor (adapt to the state of the art)

Hardware Description: Custom Board

- Printed Circuit Board, responsable of ADC’s (x4), Trigger ports and NIM & TTL conversión (x4)
- HV bias pixel sensor goes in parallel by a dedicated cable (not in the PCB, it is safer)
- High frecuency design (differential and single ended lines fully isochronous)
- Ten layers stacked



Interface
CUSTOM BOARD - TLU

Hardware Description: Remote Power Supply Subsystem

- **The RP2S allows to turn on the proposed DAQ system remotely and safely**
  - The start up sequence is programmed into a Python app.
- **It comprises:**
  - Raspberry PI: responsable for managing the start-up sequence and providing remote access
  - Master switch relay that turn on/off the FPGA
  - A Raspberry PI HAT that implements the switches to turn on/off each supply voltaje that must be provided to the CB
  - The HV bias will be managed by SCPI or LXI commands. A second switch relay can be included to turn on/off