Synchronize multiple R5560 / R5560SE to for simultaneous sampling on multiple device

Abstract

This document explains how to synchronize multiple R5560/R5560SE to extend the number of input channels sampled simultaneously.

This document covers the following task:

- How to distribute a common clock
- How to synchronize the acquisition between multiple DAQ and device
- How to distribute common trigger signal

To synchronize multiple device it is mandatory to have a DAQ and BASE operational system newer than 2022.7.0.1. Check on the display and on the web page of the DAQ and of the BASE of the R5560.

Source code available on GitHub:

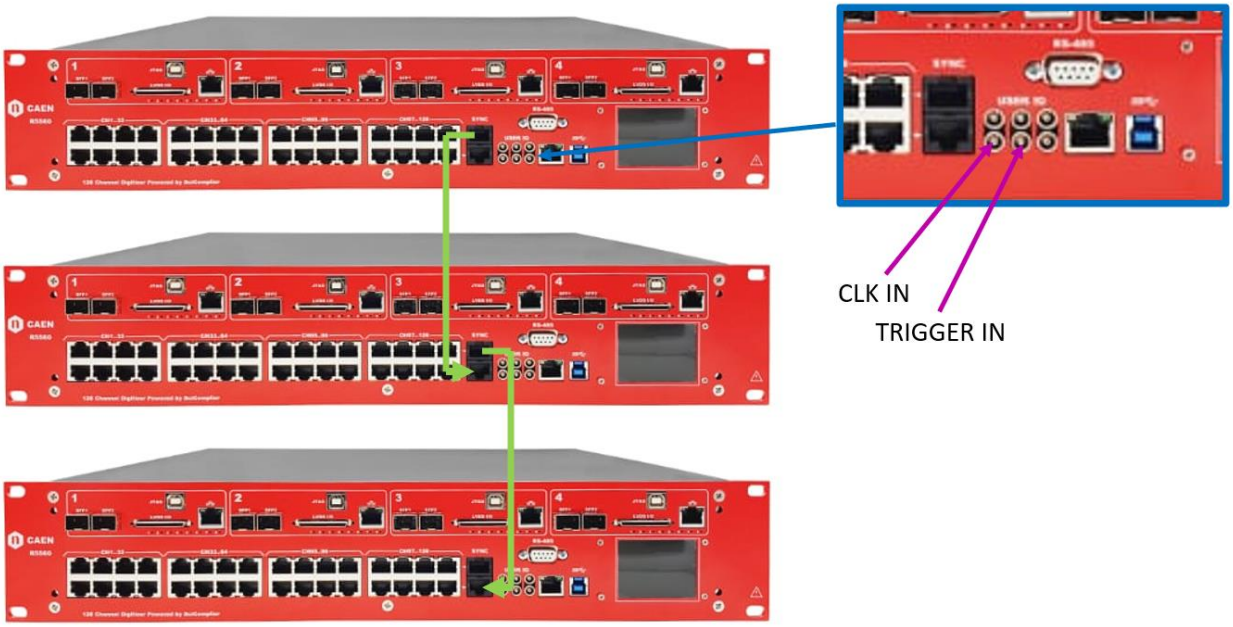
https://github.com/NuclearInstruments/r5560-multidevice-sync-wave-oscilloscope

1 Interconnection between digitizers

On R5560 family a dedicated SYNC connector is used to propagate the synchronization between several devices. The RJ45 cable carries 3 sync LVDS lines that can be configured using the display or the web interface. The function of the three lanes is not fix, just the lane 0 is special because can carry the clock signal.

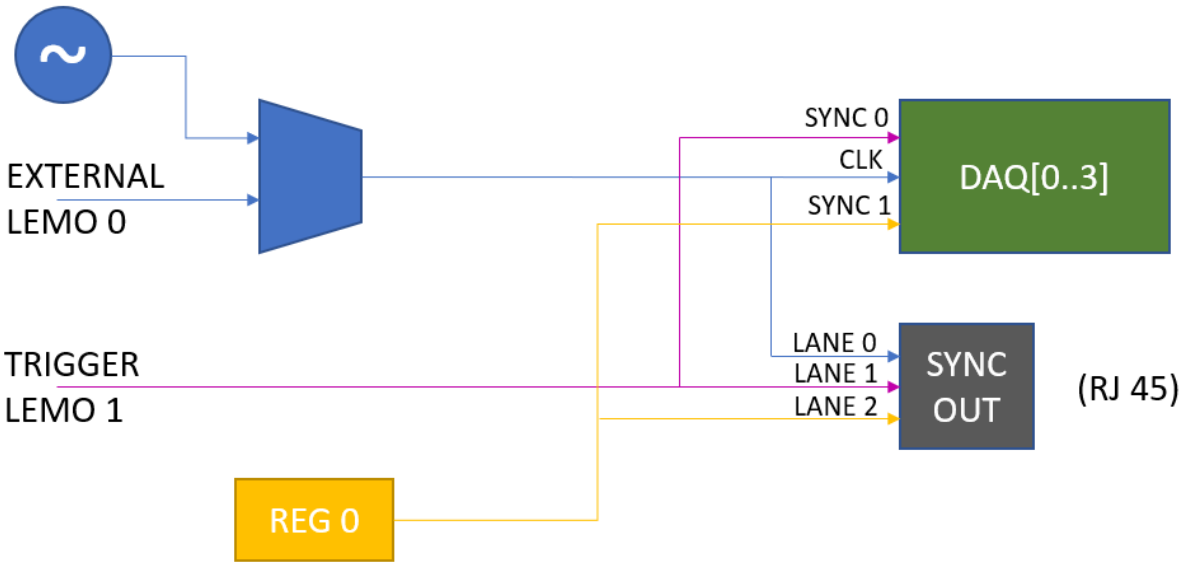
The clock signal is distributed as a 25 MHz signal. Inside the base of the R5560/SE there is a PLL to create the 125MHz clock.

All DAQ in a device will use the same clock generated by the BASE. The base can be configured to take the clock from internal 25MHz generator, LEMO 0 or the SYNC connector.



In a system with multiple devices, the master will generate the clock and control system VETO (REG)

MASTER BASE BOARD CONFIGURATION

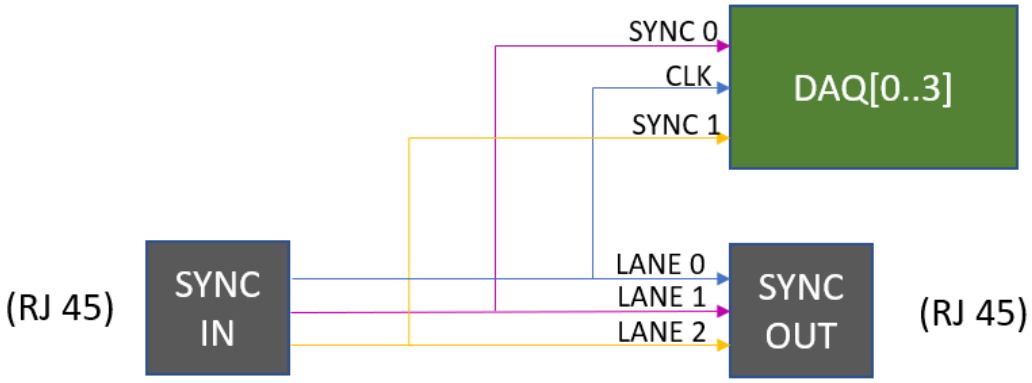


The VETO signal is a signal controlled by the software to avoid that the digitizer start acquisition before all the devices in the array are configured and ready. The REG 0 is a register in the base of the MASTER device that can be controlled by the ethernet interface of BASE the MASTER via http API or via SDK from any of the DAQ of the master device.

The external trigger can be provided from external scintillator on the LEMO 2 or can be used in self trigger as the or of the T0/T1 output of the trigger output to the BASE for each DAQ.

The slave board is configured to receive all sync signals from the master device.

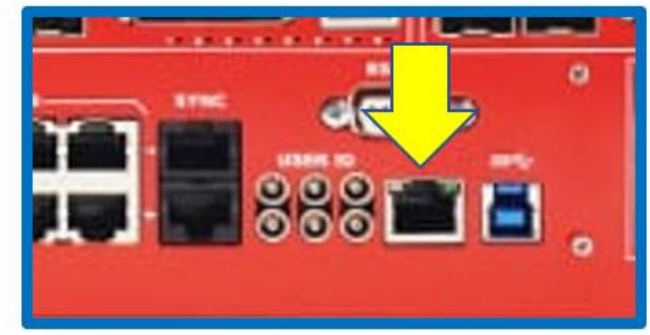
SALVE BASE BOARD CONFIGURATION



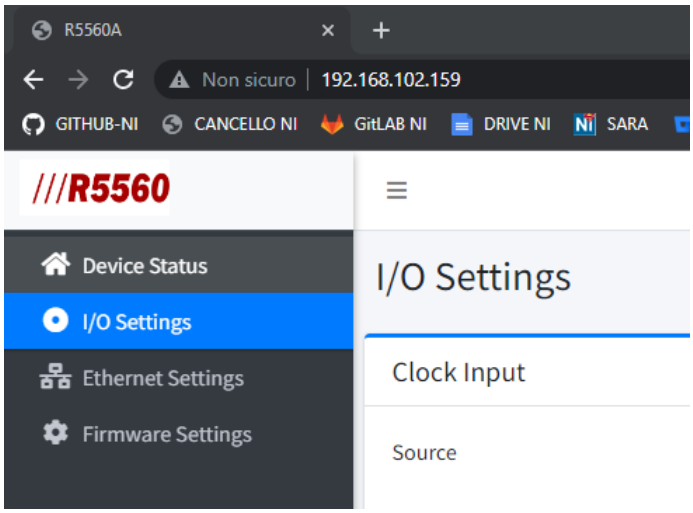
2 Configuration of the R5560 signal router

The R5560 can be configured to route the LEMO and SYNC wires in several ways. The web interface or the display can be used to configure the router

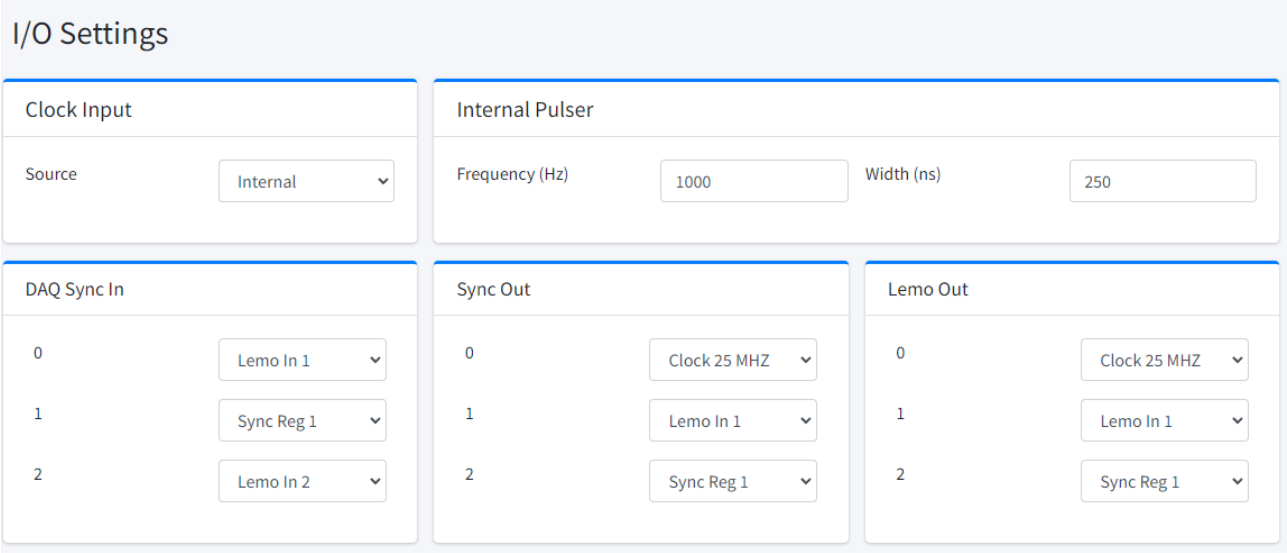
Connect the eternet cable to the base board and check on the display the IP of the base board



Enter in a browser the IP address of the base board. Click on I/O settings

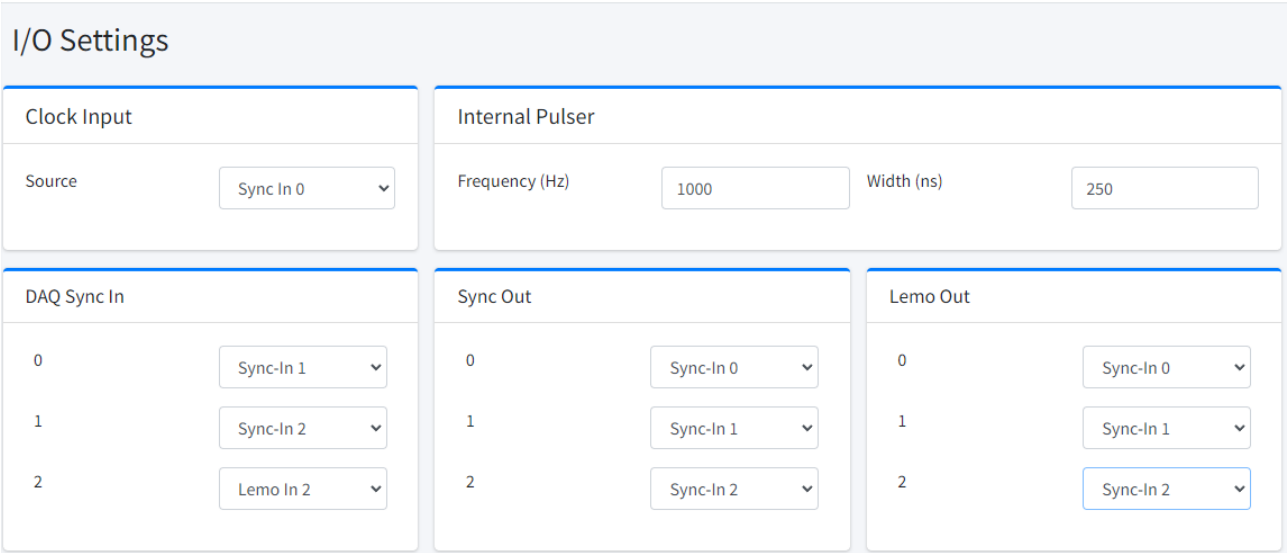


For the MASTER board configure the router as following



It is eventually possible to replace the clock source from internal to LEMO 0 if the system should be used with an external clock

For the SLAVE boards configure the router as following



For both master and slave DAQ Sync 2 is not used. We connected to Lemo 2. Can be used as flag or Veto signal. Lemo out are not used as well they are connected to the sync signals as monitor for both master and slave.

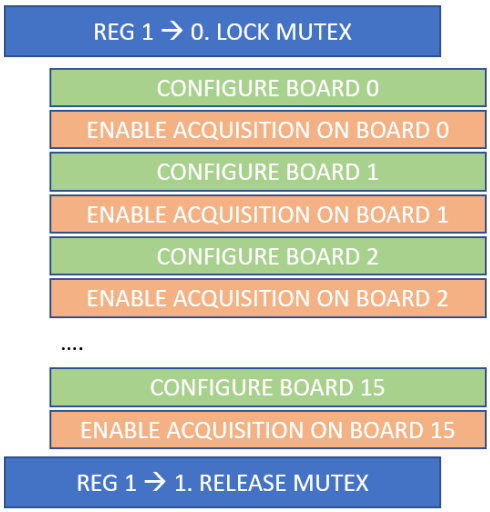
With this configuration the routing for both master and slave of diagrams in section one is achieved

3 Firmware implementation

In principle, the concept illustrated in this document can be applied to any firmware. The idea behind this synchronization is that a common trigger is provided externally from a scintillator or from a synchronous signal from a machine like a laser trigger, a synchrotron kicker signal.

The software “Sync REG 1” is a gate and prevent that the trigger can start the acquisition when not all the acquisition board are configured.

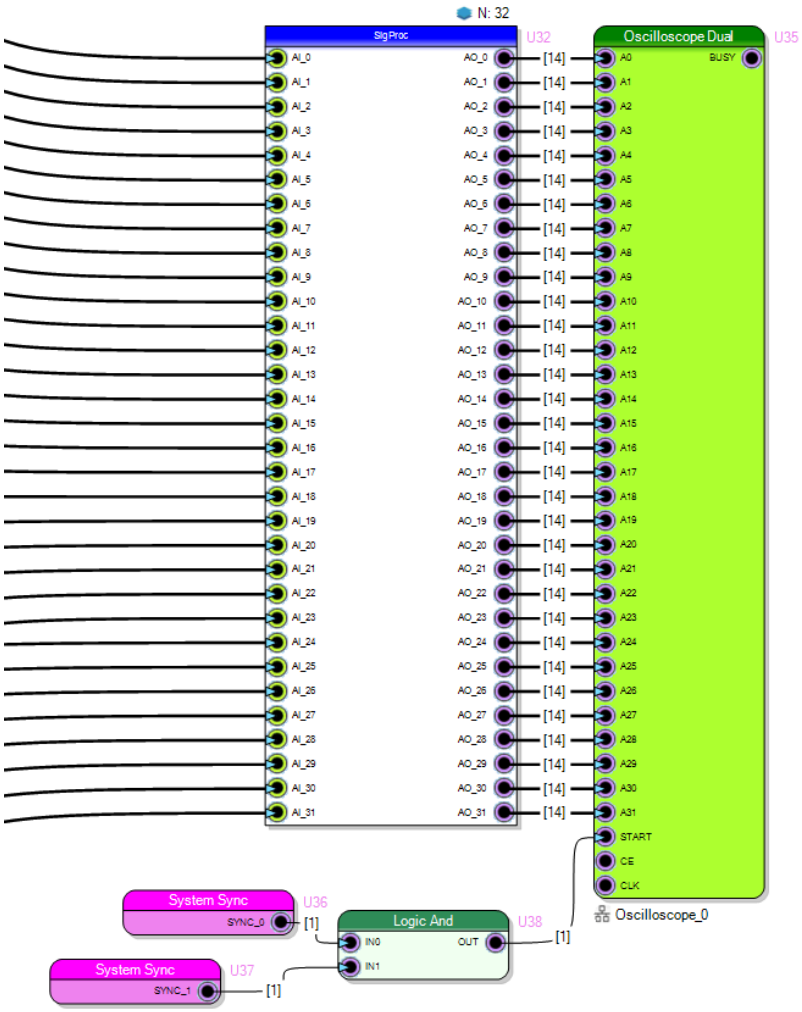
The idea is to use the Sync REG 1 of the master as a Mutex. The software will do the following operations in order



The firmware should take care to avoid the trigger to start the acqusition whenever the mutex is locked. The software mutex in first istance works on the 4 DAQ inside the same module; the R5560 works per DAQ. It means that the SDK configure one DAQ at once indeed the 4 DAQ will receive the enable acquisition with a time difference due to the configruation software execution and ethernet delays as well as they are in separate instruments. Even inside the same DAQ, if multiple block perform acquisition, they will be configured with a small delay each one. A mechanism that prevent the acquisition to be triggered until all elements are fully configured must be always implemented. The MUTEX is propagated also to the slave because the router in the master will forwared on Sync-Out 2 the Sync Reg 1 and the slave router forward the Sync-In 2 to the DAQ Sync-In 1 those are for both MASTER and SLAVE the acquisition VETO input.

In this application note we use the oscilloscope dual as waveform digitizer. We use the dual version of the oscilloscope in order to maximizer the acquisition length (up to 16384 samples for A version of the R5560)

As you can see in the diagram below the START of acquisition of the oscilloscope (the trigger signal for the oscilloscope) comes from the SYNC_0 input of the DAQ, that is connected in the master to the LEMO 1 while on the slave on the Sync-IN-1 (the master forward the LEMO 1 to the Sync-IN-1 of the slaves).
At the same time the SYNC_1 of the DAQ is used as a gate for the trigger.



4 Control Sync Registers

Sync REG can be controlled by ANY DAQ of the master device or from the base. At the moment the base can be controlled only by HTTP api.

In this document we will consider only the control of the sync registers using the SDK via the DAQ 0.

The command to set the register is the following

```
SetSyncRegister(id, value, &handle);
```

id is the register number. At the moment valid id are 0 and 1. We are using the REG ID 0

value is the value to be configured. The register are 1 bit register so value are 0 and 1. Because we use the register as a GATE acquisition can be triggered when value is 1

handle is the pointer to the handle to the DAQ 0 returned by the `R_ConnectDevice` function.

DAQ 0 will be indeed the master DAQ in the system.

5 Software

We need to design a software capable to manage multiple DAQ ad the same time. We suggest to create a C++ class implement the single digitizer.

Lock the mutex on the REG 1: set it to 0

Configure in sequence all the oscilloscope in the DAQ

Create than a thread for each DAQ that perform the readout of the data and store on file. All thread will lock on the data available status check: no data will be available because the REG 1 is set to 0.

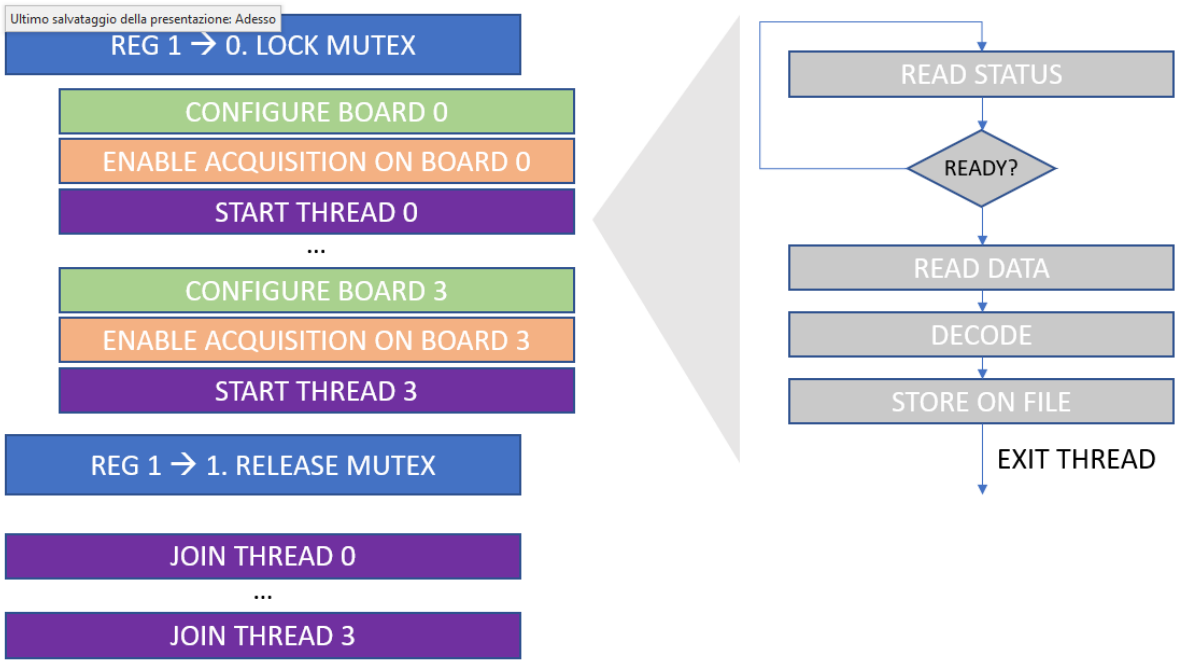
Unlock the mutex: set REG 1 to 1, all treads will start to get data and store on file.

Design the tread to make a single acquisition (do not put for inside the thread) otherwise you will lose the benefit of the mutex. If you loop inside the thread, it is possible that some thread will capture data that other threads will not acquire.

When the acquisition occurred, the thread will exit.

The main process should join (wait thread exit) for all thread. Once all thread are exited, it means that all DAQ grabbed the same data.

Now it is possible to repeat the sequence to acquire the next wave

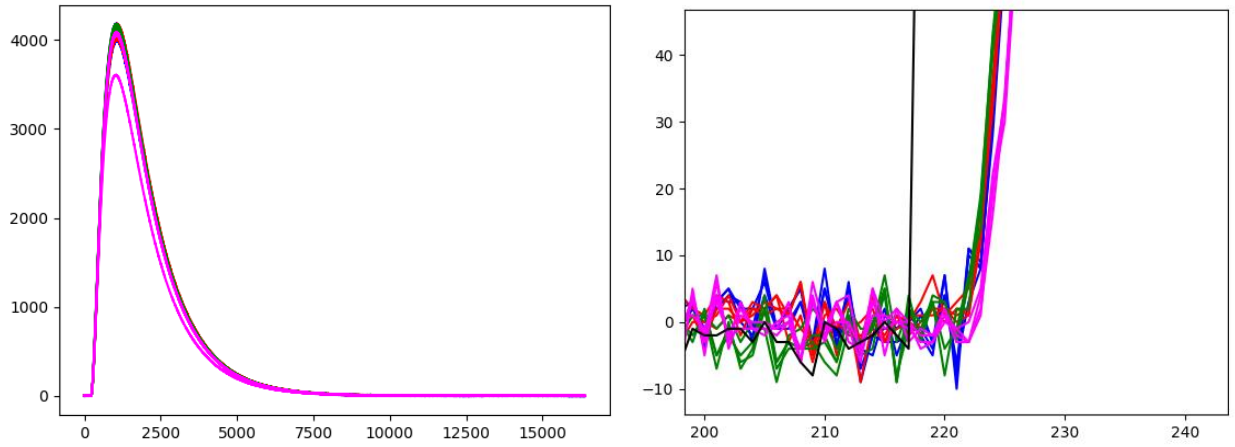


The code is available on Github in software/C folder

The program save data, one file per DAQ in text format (human readable), one column per channel, one row per sample

8158	8151	8245	8160	8155	8179	8116	8193	8214
8161	8149	8245	8162	8155	8182	8114	8194	8216
8162	8148	8245	8164	8155	8179	8117	8196	8215
8163	8151	8246	8164	8155	8180	8115	8195	8216
8162	8148	8246	8163	8153	8182	8117	8194	8215
8163	8151	8247	8163	8155	8181	8115	8194	8215

In software/python_plot it is available an example code to read the data file (in text format and plot waveform)



Zooming at the very beginning of the signal it is possible to see that all signals are in phase