# PROCHIP – Acquisition hardware and software

The setup is composed by a detection camera (Hamamatsu Orca Flash 4.0V3), two laser sources (OBIS 561-150 LS and OBIS 488-150 LS), a DAQ (NI USB 6212) and a custom-made logic port, connected as in figure 1.

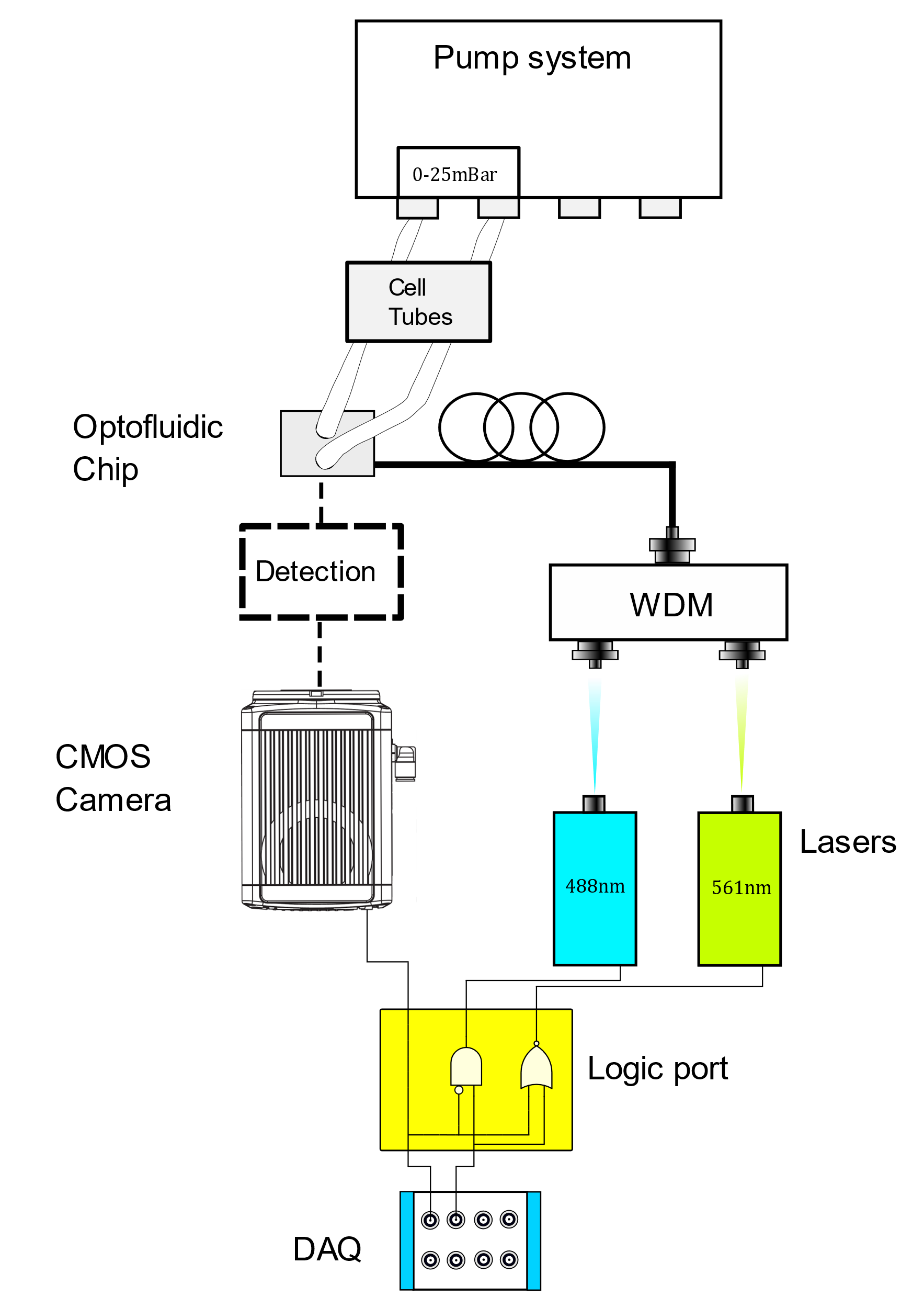


Figure 1 Setup scheme

The camera is operated in *rolling shutter mode*, i.e. the sensor lines are readout sequentially in two parallel buffers, starting from the two central lines and opening towards the edges like a curtain. Meanwhile the more external lines are read, the central ones start directly with the following acquisition.

This readout mode implies a problem when we operate a sequential dual-color measure, i.e. when we shine alternatively the sample with one laser and with the other. Indeed, since the readout of the sensor lines is not simultaneous with the acquisition trigger, part of the sensor will acquire light belonging to the “next frame”, with a consequent crosstalk between the two color acquisition channel.

To circumvent this problem, we need to shape accurately the camera trigger signal and the laser switch signals. Our solution is represented in figure 2. We generate 2 different signals with the DAQ and make use of the logical port to generate two delayed laser switch signals. As it can be seen, during the ramp of the camera trigger, that represent the rolling shutter time, neither of the laser is ON, avoiding in this way the crosstalk.

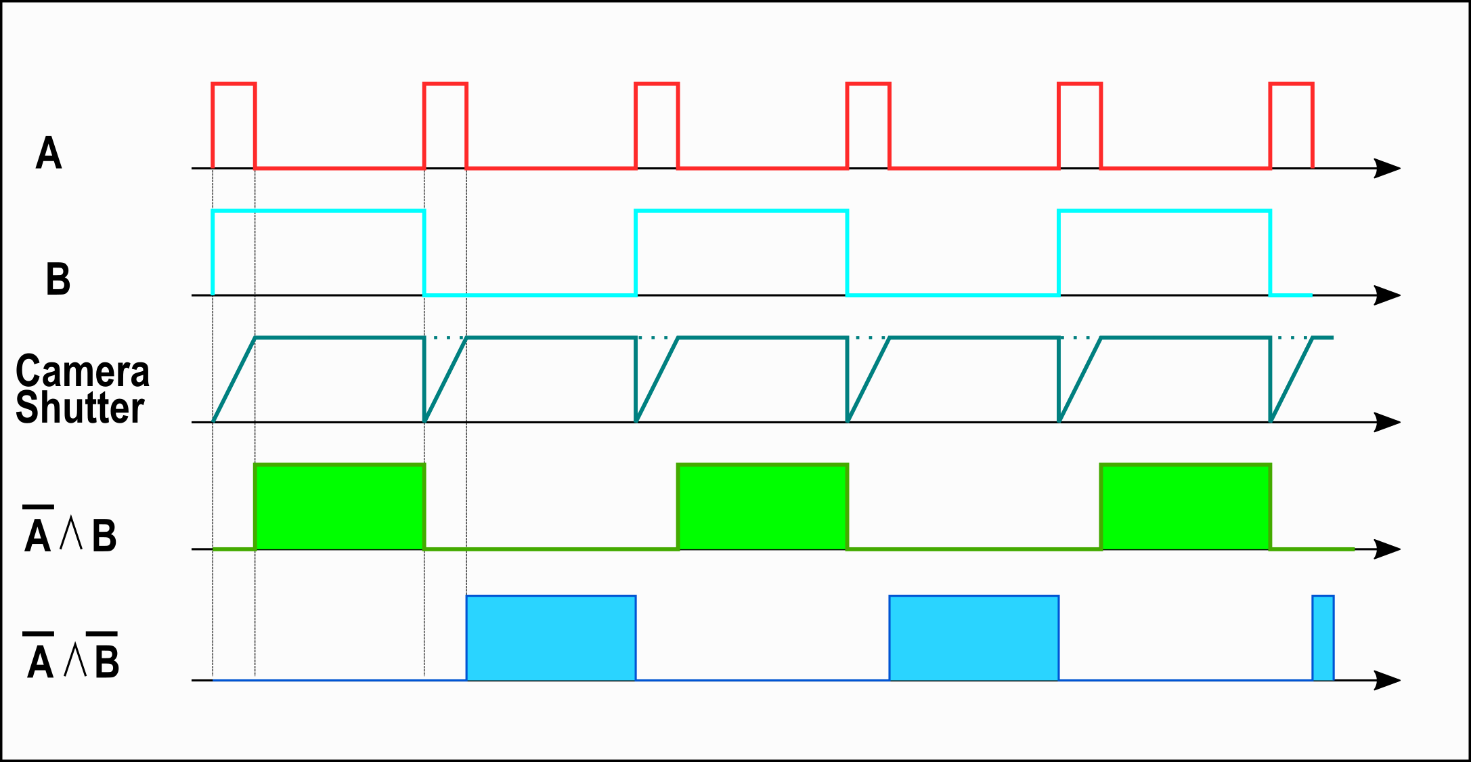


Figure Control signals scheme

## Hardware

The two lasers are set to work in DIGITAL MODE and the trigger signal is connected to their “DIGITAL INPUT” port. REMARK: to be read as “1” by the laser controller, the signal must be greater than 3 V, considering a load resistance of 50 ohm (numbers found at <https://www.coherent.com/assets/pdf/COHR_OBIS_SingleLaserRemote_DS_0220_1.pdf> , need to be verified).

The camera works in rising-edge trigger mode and is connected to the DAQ signal with a coaxial cable. The signal can go from 0V to 5V (to be verified) and the camera is triggered at 3.3V.

The DAQ control signals are 2 TTL signal generated at pins PFI 1.3 and PFI 1.2 (to be verified). The first one (signal A of figure 2) is split and sent to the camera and to the logical port, while the second (signal B) is connected directly to the latter. The signal A frequency must be two times the signal B frequency, while the signal A duty cycle depends on the readout time of the camera, in particular:

Where:

* DutyCycleA is the duty cycle of the signal A
* internalLineInterval is the time interval passing between the exposure starting of one line and the exposure starting of the next one;
* VerticalPixelLines is the number of vertical pixel lines in the frame;
* exposureTime is the exposure time for each pixel;
* The whole denominator is simply 1/frequency of the camera;

The signal B duty cycle is always 0.5.

The logical port has been realized by the Politecnico technicians and perform the operations indicated in the figures, a (NOT on the first signal +AND) and a (NXOR). It needs to be connected to power source, but is important to unplug it after the measures to avoid overheating as it does not have a “switch on” control.

## Software (to be updated)

The camera, the lasers and the DAQ board are controlled with a custom software written in Python, based on the ScopeFoundry Package.

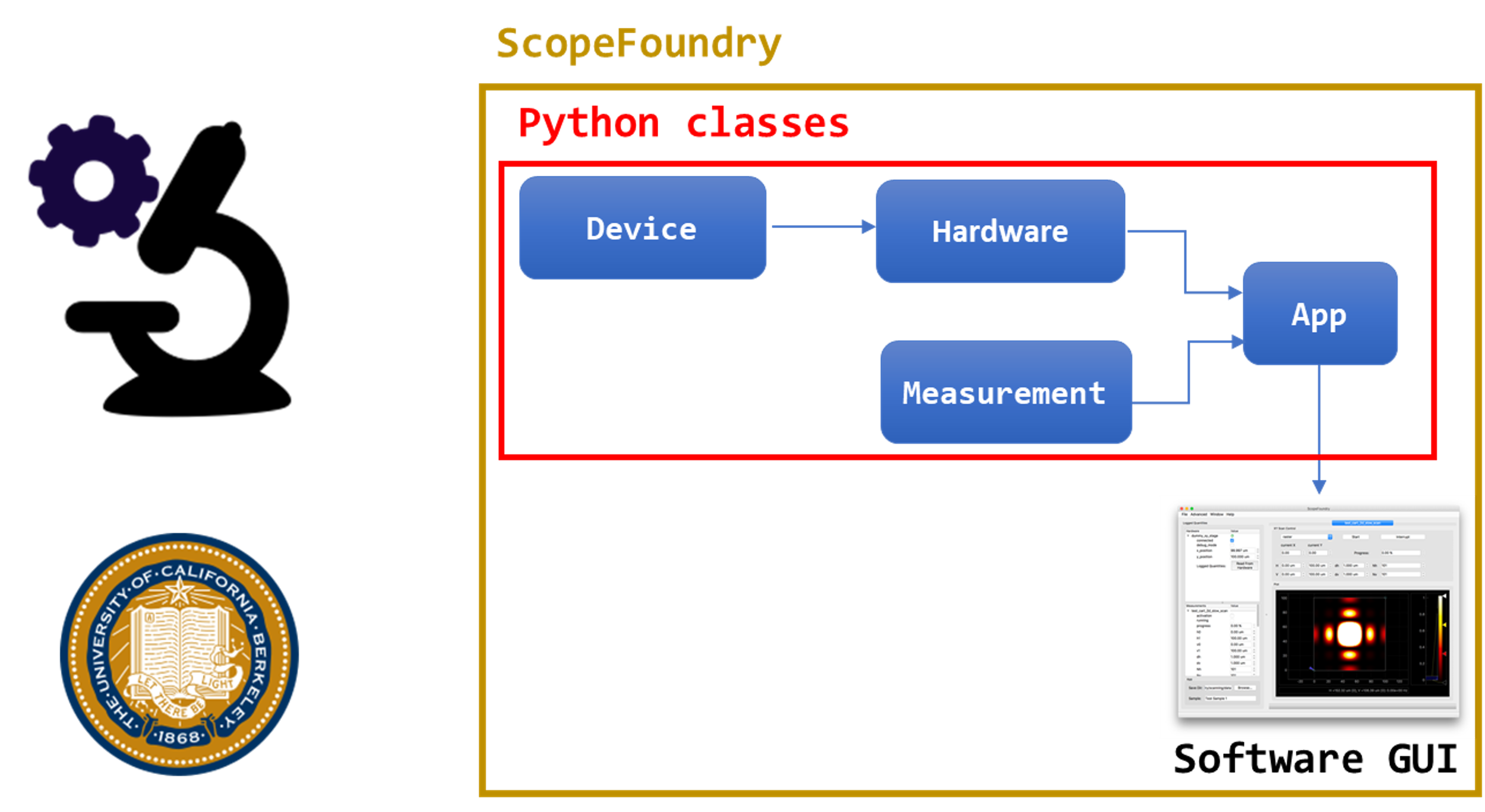


Figure 3 ScopeFoundry scheme

ScopeFoundry defines four classes (device, hardware, measurement and app) in order to run the final software GUI (figure 3).

The Python scripts can be found at the repositories of the ProChip project ( <https://github.com/micropolimi?tab=repositories>), and all the defined classes are listed in the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Device | Hardware | Measurement | App |
| Camera | HamamatsuDevice (<https://github.com/micropolimi/Hamamatsu_ScopeFoundry/blob/master/CameraDevice.py>) | HamamatsuHardware (<https://github.com/micropolimi/Hamamatsu_ScopeFoundry/blob/master/CameraHardware.py>) |  |  |
| Lasers | LaserDevice (<https://github.com/micropolimi/Coherent_laser/blob/master/laser_device.py>) | LaserHardware (<https://github.com/micropolimi/Coherent_laser/blob/master/laser_hardware.py>) |  |  |
| DAQ | NI\_DO\_Device (<https://github.com/micropolimi/nidaqmx_test/blob/master/ni_do_device.py>) | NI\_DO\_hw (<https://github.com/micropolimi/nidaqmx_test/blob/master/ni_do_hardware.py>) |  |  |
| NI\_CO\_Device (<https://github.com/micropolimi/nidaqmx_test/blob/master/ni_co_device.py>) | NI\_CO\_hw (<https://github.com/micropolimi/nidaqmx_test/blob/master/ni_co_hardware.py>) |  |  |
| Measurement |  |  | SyncreadoutTriggerCounterMeasurement  (<https://github.com/micropolimi/DAQ-Laser-Camera/blob/master/SyncreadoutTriggerCounterMeasurement.py>) |  |
| App |  |  |  | Laser\_DAQ\_Camera\_App  (<https://github.co1m/micropolimi/DAQ-Laser-Camera/blob/master/Laser_DAQ_Camera_App.py>) |

In the device classes, the functions for controlling the single instruments are defined. For the camera, the HamamatsuDevice class is mainly a Python wrapper of the library provided by Hamamatsu (hdcam), written in C language. The LaserDevice class was based on the serial Python package, since lasers communicate with the PC with an USB interface. For the DAQ board, different device classes were defined for the different channels used: namely, two counter output and one digital output channels. So, the NI\_DO\_Device and the NI\_CO\_Device classes were written with the nidaqmx Python library provided by the National Instrument.

The hardware classes represent the link between the user and the device classes, so that via a simple GUI the user can select the proper values for the most important parameters of the instruments (e.g. the power of the lasers, the exposure time of the camera, the frequency of the DAQ TTLs, etc.).

The SyncreadoutTriggerCounterMeasurement defines how the instruments operate during the measurement and how the images are visualized and saved. The steps to follow in order to successfully run the measurement are the following ones:

1. Connect all the instruments;
   1. For the lasers, it may need to choose the correct port value (it depends on which USB port on the PC they are connected).
2. Set the right parameters in the hardware panel for the instruments:
   1. For the camera, set acquisition\_mode equal to “run\_til\_abort”. Then, choose proper values for the exposure\_time, for the frame positions and sizes (subarrayh\_pos, subarrayv\_pos, subarray\_hsize , subarray\_vsize) and for the number\_frames to save in the h5\_file (in order to avoid inconsistency between a certain wavelength and the corresponding frame, this number must be an even number, since the camera buffer was organized in such a way to store a certain wavelength at even indices while the other at odd indices).
   2. For the lasers, choose the proper laser\_power.
   3. For the the counter output connected to the camera (should be Counter\_Output\_1) set the channel value equal to Dev2/ctr1, choose the proper frequency value for the TTL, and also the correct value for the duty\_cycle (for the moment, choose 0.5, while in future there will be no need for choosing it, since it will be set automatically equal to the above formula)
   4. For the other counter output (Counter\_Output\_2) set the channel value equal to Dev2/ctr1 ,choose a frequency value half of the frequency of Counter\_Output\_1, and a duty\_cycle equal to 0.5 (in future this will be all automatically set).
   5. For the digital output channel, set the channel value equal to Dev2/port1/line0.
3. Set the save\_dir equal to the saving directory.
4. Push start on the measurement panel.
5. Enjoy the images, and when you see something interesting, push save\_h5 on the measurement panel, in this way the previous frames will be saved.
6. For stopping the measurement, press the stop button in the measurement panel.

Some notes:

* A brief description of what happens after pushing start: the run function of SyncreadoutTriggerCounterMeasurement starts. In this way, some parameters of the instruments are automatically set (the ones that are not decided by the user since are automatically defined by other parameters / are always the same). Then, the camera starts the waiting for the triggering signal, as well as the laser. Then, the counter output TTLs are started by a rising edge of the digital output channel. In this way, the lasers and the camera starts the synchronized acquisitions. Then, the two channels are continuously visualized on two different panels. When the save\_h5 button is pressed, the previous number\_frames images are saved into an h5 file. When the stop button is pressed, the camera stops the acquisition, the lasers are switched off, and the TTL are stopped. (pay attention: the instruments are not disconnected!).
* During the run of the measurement, the number of detected cells is counted. This counting is implemented by using the counter\_threshold value in the measurement panel. This value is compared to the mean value of one image: if the mean is greater of the threshold, then the cell counter is increased of one unit, in the other case it remains the same. When a new cell is detected, the detection attribute of the SyncreadoutTriggerCounterMeasurement class is set to True, and return to False only when the mean frame value is smaller than the threshold, i.e. when the detected cell is no more in the frame; in this way, the counter is increased only once per cell. Pay attention: this counter is not sensitive to multiple cells in one frame, they will be counted as one!