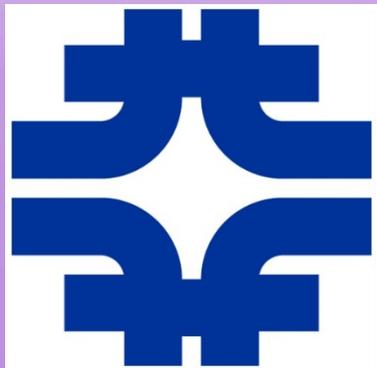


**Optimization of the Data Acquisition
Software
(PxSuite DAQ)
for the Silicon Strip Telescope at FTBF**



Clifford Denis
SIST Intern, Fermilab
Ramapo College of New Jersey
08/04/2014



Background

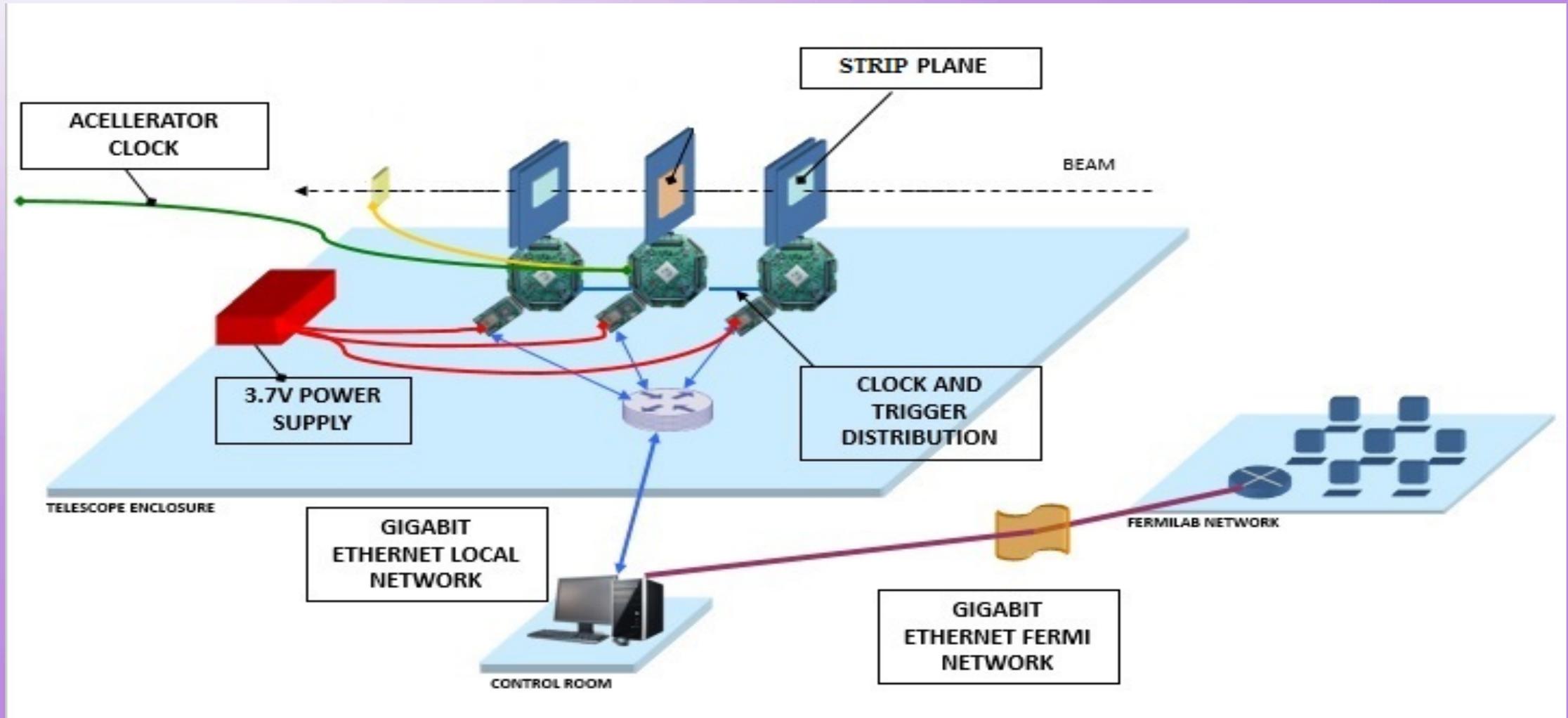
➤ What is the Silicon Strip Telescope?

- Created by the Detector Instrumentation Group (DIG) at Fermilab
- Built to track particles' trajectories
- Used at the Fermilab Test Beam Facility (FTBF) to provide tracking information for different Detectors Under Test

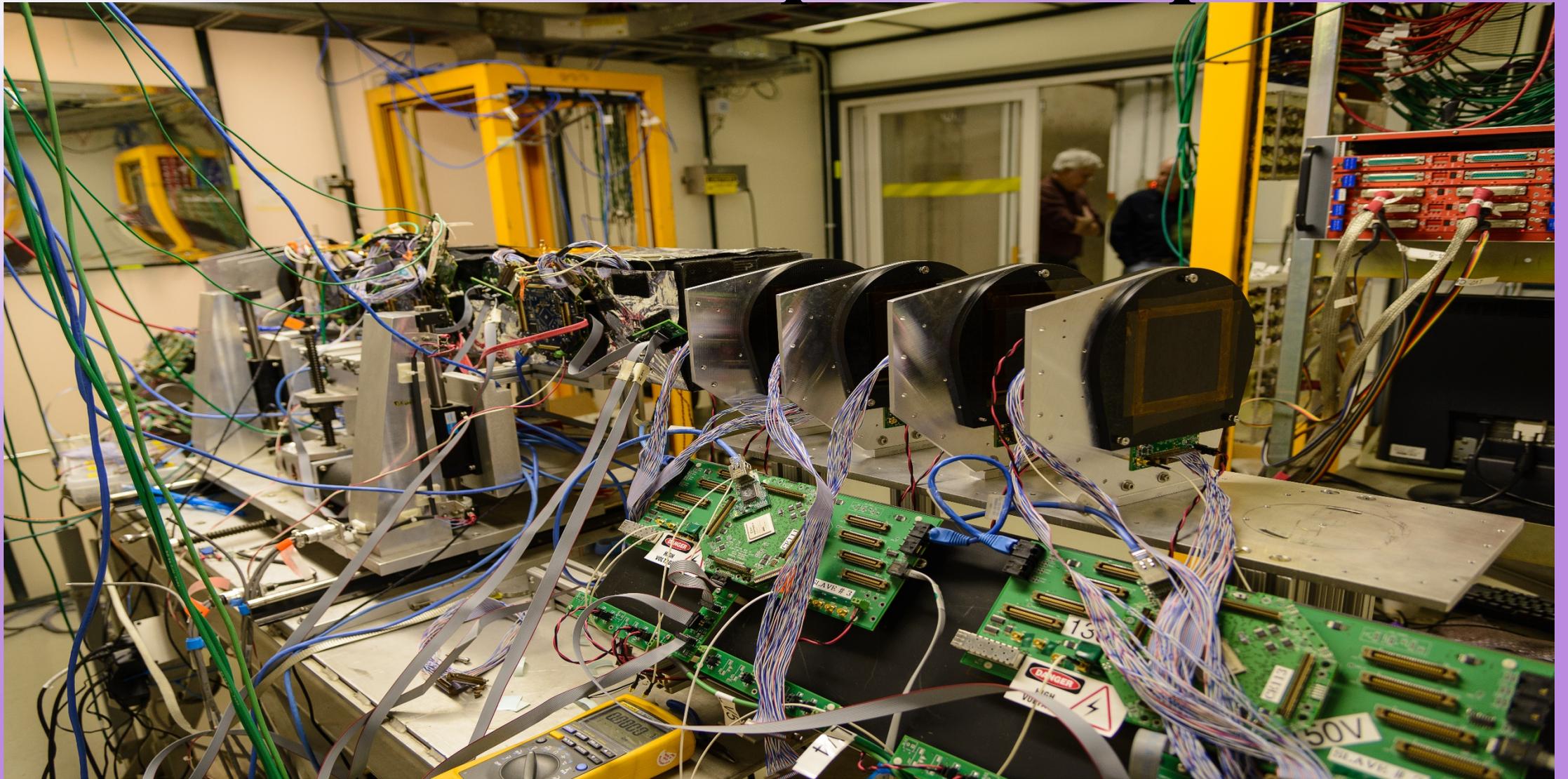
➤ What are the main components?

- The Silicon Strip Detector
- The CAPTAN Boards (DAQ Hardware)
- The PxSuite (DAQ Software)

Si-Strip and DAQ layout



The Si-Strip Telescope



The Si-Strip Telescope and DAQ

➤ The Strip Station

- Two strip planes mounted orthogonally
- A strip plane is made of 639 strips, each 60 microns wide and 9cm long
- Overlap area of $\sim 4 \times 4 \text{cm}^2$

➤ The CAPTAN Board (DAQ Hardware)

- Compact And Programmable daTa Acquisition Node
- Octagon-shaped electronic board designed for acquisition, processing and control of data in High Energy Physics
- Can be customized connecting other electronic devices (including additional CAPTANs) through the main buses

The Si-Strip Telescope and DAQ

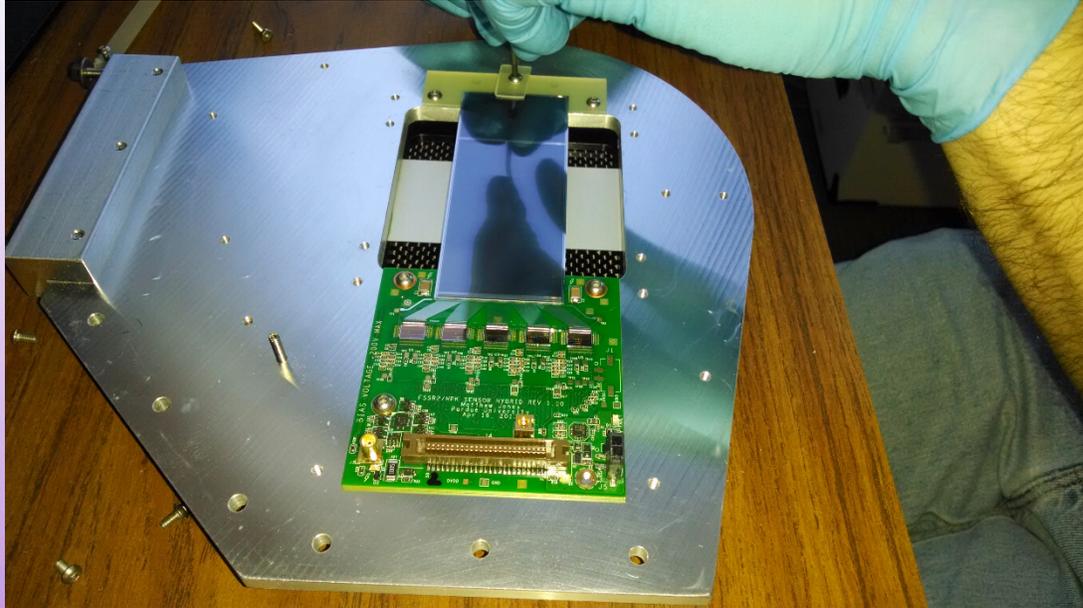
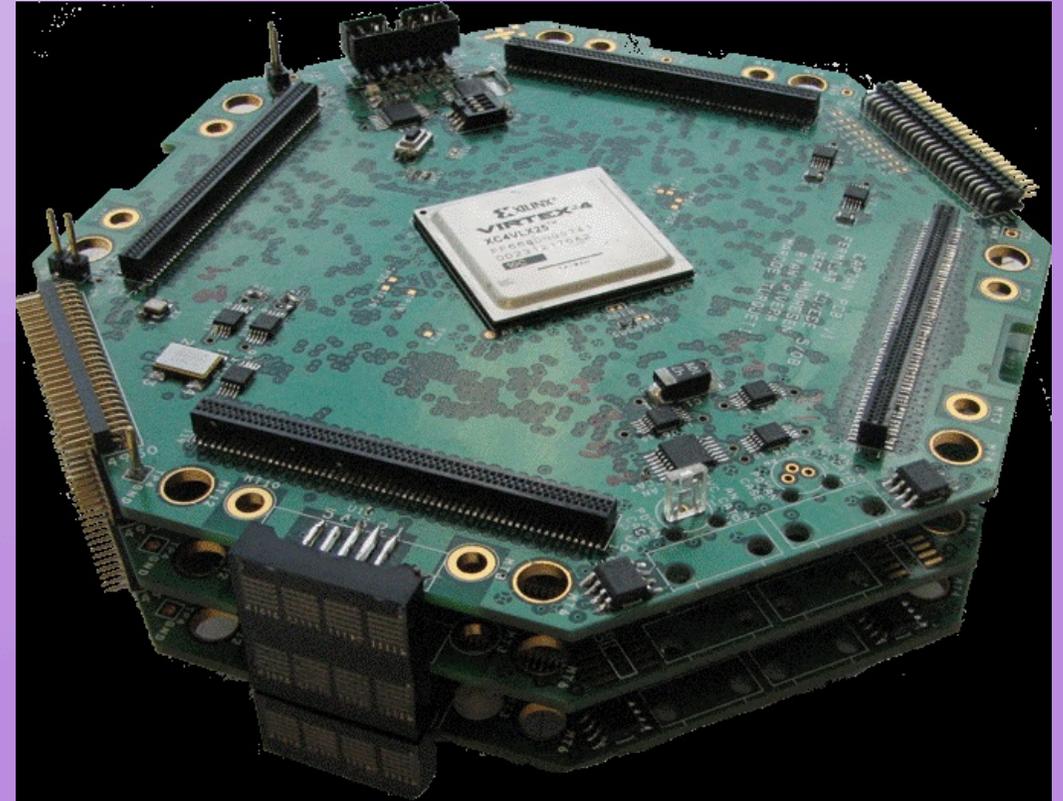


Fig. 3 - The Strip Station

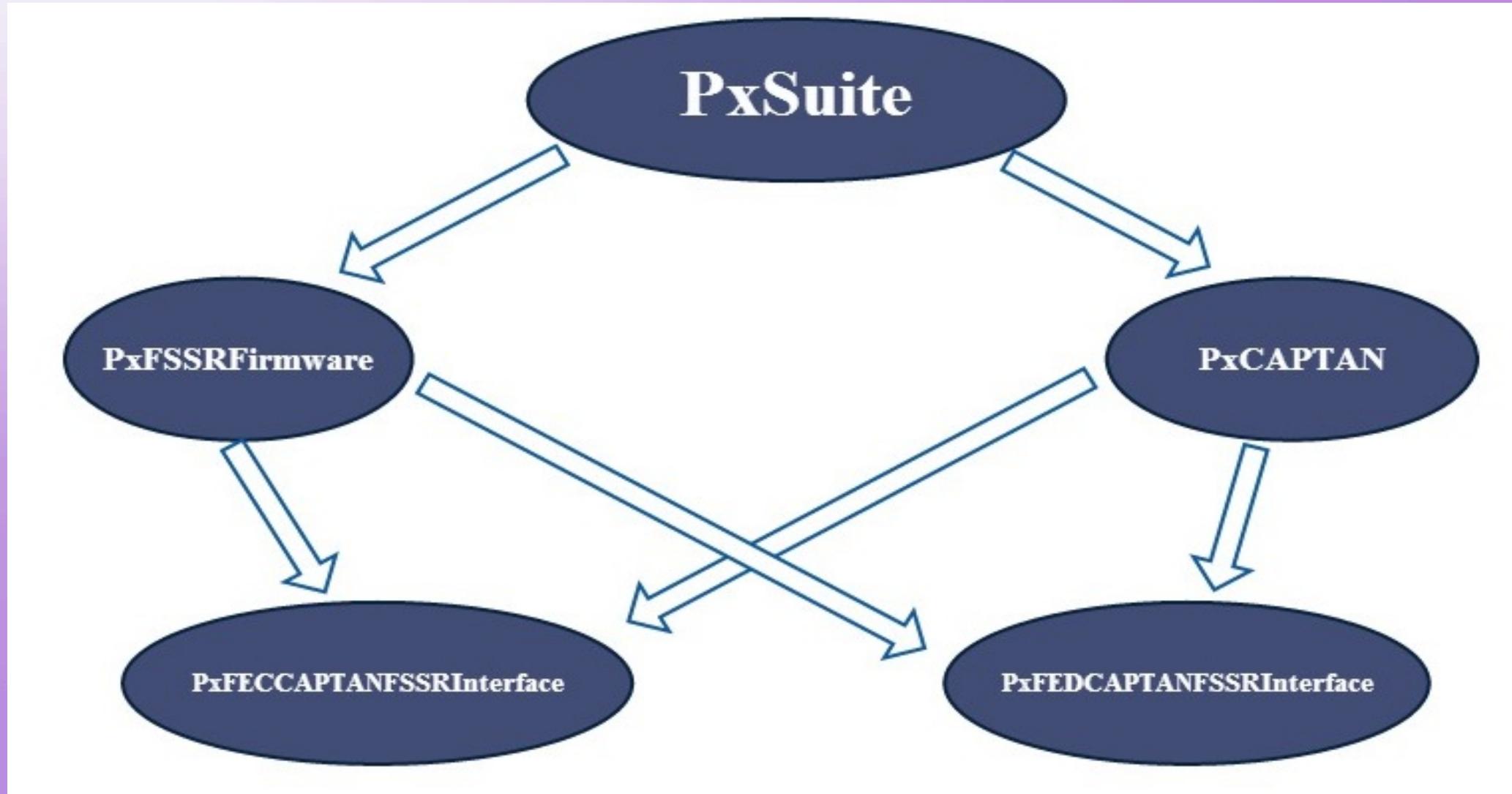
Fig. 4 - The CAPTAN Board



The PxSuite

- The PxSuite (DAQ Software)
 - Written in C++, HTML and JavaScript
 - Interface that lets the user program and interact with the hardware through a web-based graphical user interface
- The Configuration Files
 - Used to configure the telescope and the DAQ hardware
 - Sample actions include: setting the clock source, the clock speed, enabling and disabling strips, setting the threshold sensitivity of the strips

Diagram of the PxSuite DAQ Software



The Issues

➤ Accessibility

- Some parts of PxBsuite code not so easy to read
- Lacking methods to perform direct operations on registers

➤ Performance

- Taking a lot of time to communicate to the hardware

Accessibility

- Created methods that perform specific operations.
 - Directly modify individual bits of the register STRIP_CSR (setting the BCO clock state, enabling the stream, etc.)
 - Directly set the frequency according to the clock source (External or Internal).
 - Communicate directly with the configuration files, therefore avoiding the need to modify the source code itself (getting the state of the clock from configuration file, etc).

Performance

- Implemented methods that minimize the response time.
 - Send commands concurrently, therefore reducing the time the program has to wait for an acknowledgment from the hardware.
 - Compare sent and received buffers, making sure they are correctly interpreted.
 - Create a register from the hardware's acknowledgment while starting data acquisition.

Example of Code

```
void PxFEDCAPTANFSSRInterface::start(void)
{
    string value;

    PxCAPTAN::sendAndReceive(PxFSSRFirmware::readCSRRegister(), value);
    string CSRRegister = PxFSSRFirmware::readCSRRegister();
    uint32_t registerValue = PxFSSRFirmware::createRegisterFromValue(CSRRegister, value);

    if (registerValue == 0)
        cout << "\tERROR - Problem with register! See above" << endl;

    PxFSSRFirmware::setCSRRegister(registerValue);
    PxCAPTAN::send(PxFSSRFirmware::enableTrigger());
    PxCAPTAN::send(PxFSSRFirmware::resetBCO());
    PxCAPTAN::send(PxFSSRFirmware::startStream());
}
```

Conclusions and Future Work

➤ Conclusions

- Pxsuite code is now easier to read
- Pxsuite performs faster than before

➤ Future Work

- There are more registers to work on, a lot more!
- Pxsuite can probably be made even faster

THANKS!!!

- Lorenzo Uplegger (Supervisor)
- Ryan Rivera (Second Supervisor)
- Kelly Stifter (co-worker)
- Daniel Parella (co-worker)
- Fermilab and the SIST committee

Any questions?