Developing a Readout Software for the NOPTREX Experiment

Aaron Moseley, Jon Mills, J. Fry, C. Crawford

University of Kentucky, Eastern Kentucky University, Indiana University

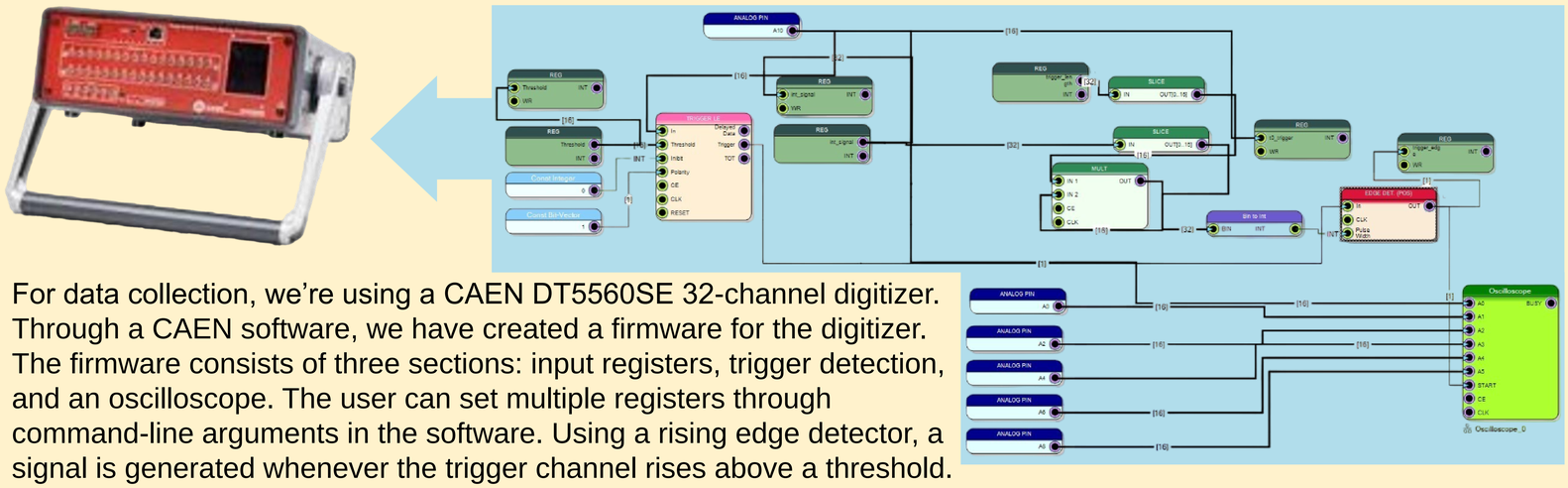
NOPTREX Experiment

The NOPTREX experiment is a collaboration investigating time-reversal symmetry violations in order to explain the Baryon Asymmetry of the early universe. Baryon Asymmetry is the imbalance between matter and anti-matter just after the Big Bang.

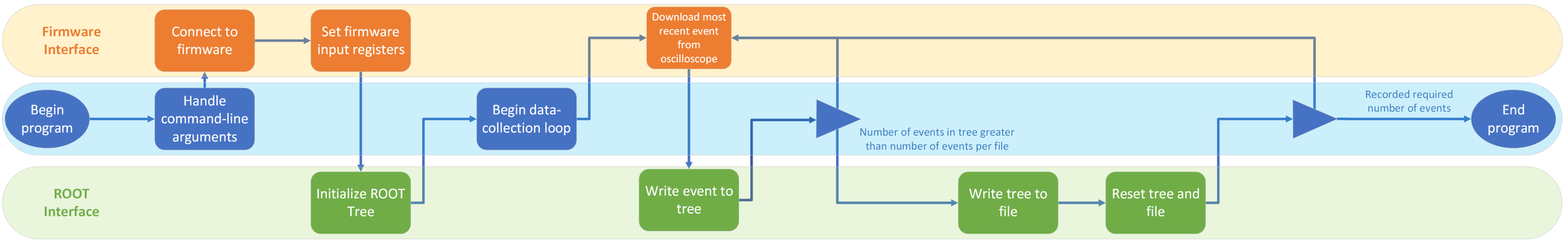
An array of 24 scintillation detectors will measure the gamma radiation emitted when an accelerated neutron impacts a heavy target after passing through an experimental apparatus simulating time reversal.

In order to collect the data generated by this experiment, a custom firmware and readout software have been developed. These have already been successfully tested using 5 gamma detectors on a particle accelerator in the J-PARC lab.

Firmware and Hardware



For data collection, we're using a CAEN DT5560SE 32-channel digitizer. Through a CAEN software, we have created a firmware for the digitizer. The firmware consists of three sections: input registers, trigger detection, and an oscilloscope. The user can set multiple registers through command-line arguments in the software. Using a rising edge detector, a signal is generated whenever the trigger channel rises above a threshold. After a trigger, the oscilloscope collects data for the next 16,384 clock cycles and makes it accessible by the readout software. The oscilloscope can collect data on multiple channels, making it ideal for this experiment.



Readout Software

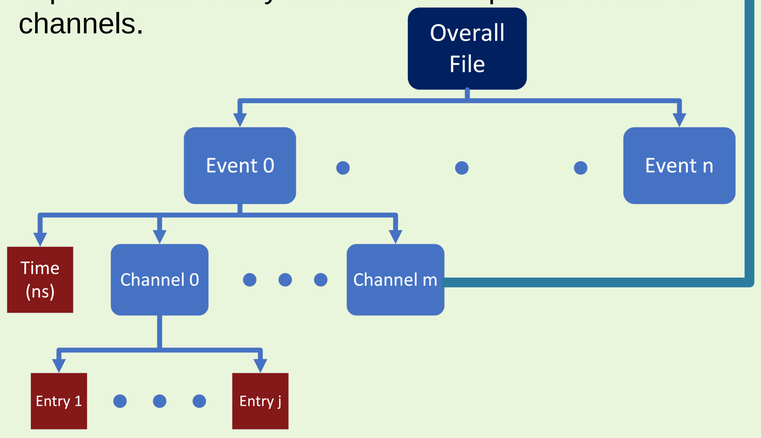
This readout software was programmed in C++ using the CERN ROOT framework and a library created by the CAEN SciCompiler software. We chose to use C++ because of the high performance when compared to other languages as well as to ensure compatibility between the libraries.

The software consists of 3 sections, a primary script used for running everything, a class for interfacing with the firmware, and a class for accessing ROOT specific functions (like file creation and modification). This abstraction was done to make the code more understandable and make updates easier to perform.

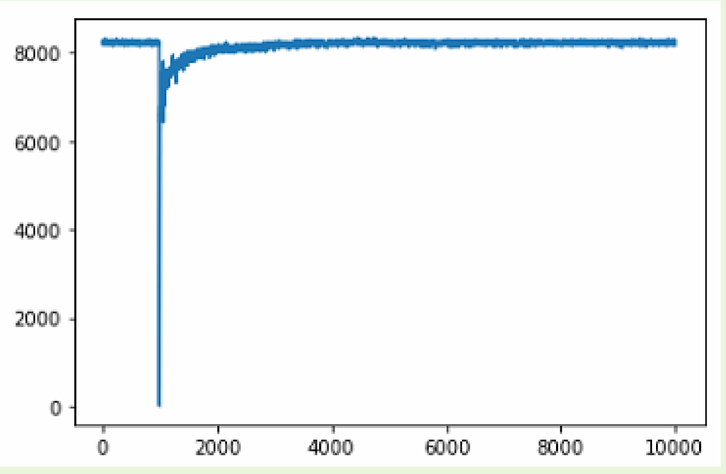
After setting the firmware inputs based on command-line arguments, the software enters into a primary data collection loop. It downloads the first 16,384 entries past a trigger pulse (with a user-defined pre-trigger) for each channel, and stores that in a branch in the data tree. Once enough of these events have been collected, the software saves the tree to a file and opens a new file, repeating the process.

File Output

The output files consist of a tree containing the data for each channel and event on multiple layers. The tree contains branches for each event. Each branch contains a timestamp in nanoseconds and a list of vectors containing the data from the oscilloscope. Organizing it in this way allows members of the experiment to easily find data from specific events or channels.



Collected Data



This shows the data on one channel from one event, collected in a test run on a J-PARC particle accelerator. The data was visualized using the PyRoot library.