Developing a High Speed Data Acquisition System For Nuclear Physics Experiments Using FPGAs

JT Mills

Eastern Kentucky University, Richmond KY 40475

(Dated: July 8, 2022)

I. INTRODUCTION

The study of fundamental symmetries has become a prime focus for nuclear and particle physicists, as it provides a deeper understanding of the nature of the universe and its formation. For example, the matter-antimatter asymmetry of the early universe, also known as baryon asymmetry, remains an open question in cosmology, and the study of fundamental symmetries could provide the answer. In particular, the Neutron Optics Parity and Time Reversal Experiment (NOPTREX) collaboration is studying the violation of time-reversal symmetry. This symmetry violation is required by our current model of the Big Bang in order to explain why the universe formed as matter and not antimatter.

To investigate this asymmetry, the NOPTREX collaboration is using a low energy neutron beam to measure n-gamma resonances in heavy nuclei, which will be measured by an array of 24 sodium-iodide (NaI) scintillation gamma detectors (Figure 1). Therefore, this measurement will also require a high speed data aquisition system to process the large amounts of data from the array in real time. By using field programmable gate arrays (FPGAs) to digitize and filter the analog signals from the detectors, it is possible to implement high performance trapezoidal filters that can efficiently determine the energy and timing of each detector pulse with a high level of precision.

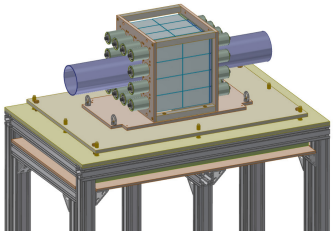

FIG. 1. The 24 NaI scintillation detector array for NOPTREX, being built at EKU.

In this research, I will use the CAEN SciCompiler software to develop a firmware for the

DT5560SE open FPGA 32 channel digitizer that uses a trapezoidal filter and multichannel analyzer (MCA) to record the energy of each detector pulse, as well as charge integration to measure the detector current over a given time period. The block-coding style of the SciCompiler application simplifies the firmware design process, and will allow for a gradual introduction to computer logic. Subsequently, I will use the Vivado coding environment to develop firmware for the KRM-4ZU27DR development board. This board uses many high performance components to process high frequency digital signals, and will be used to development new methods of data acquisition for nuclear experiments. Upon the completion of this research, the DT5560SE digitizer will be used to acquire data for the NaI array being built at Eastern Kentucky University, and I will have gained a deeper understanding of computer logic and firmware development for FPGAs.

II. FIELD PROGRAMMABLE GATE ARRAYS

Field programmable gate arrays (FPGAs) differ from conventional CPUs in their ability to have their architecture rearranged based on the demands of a specific task. The architecture of an FPGA consists of a system of configurable logic blocks (CLBs) made up of look-up tables (LUTs) to define the logic structure and flip-flops to store data. In addition to the CLBs, digital signal processor (DSP) slices are also incorporated to handle more complex tasks such as addition, multiplication, and accumulation. These components are connected by interconnects, which serve as pathways for the signals to travel through. By combining these systems with I/O ports and RAM memory, it is possible to create a fully customizable FPGA that can be reprogrammed on the fly using a hardware development language (HDL). While the use of a programming language may cause FPGA design to seem similar to computer software coding, it is actually much different as an FPGA requires the definition of the specific arrangement of its hardware. The adaptability of FPGAs make them a viable candidate for high speed data acquisition, as they are capable of efficiently processing data in real time and thereby reduce the amount of post-recording data analysis for the experiment.

A. DT5560SE

To manage the DAQ system for the NOPTREX NaI detector array, we will use the CAEN DT5560SE open FPGA digitizer featuring 32 analog input channels, 14-bit ADC, and 125 MS/s processing capabilities (Figure 2). The board also features 6 extra digital I/O ports, two optical link ports, and two sync ports which be used to synchronize the internal clock with that of another

board. The board can be connected to a PC using either a USB or Ethernet connection, allowing access to the board settings, analog front-end settings, and the FPGA flashed firmware. The high processing power of this device in conjunction with the customizability of the FPGA will allow for high speed, real-time signal processing for each of the 24 detectors.

FIG. 2. The CAEN DT5560SE 32 channel Open FPGA digitizer.

The board defaulted with a precompiled multichannel pulse height analysis firmware that included an oscilloscope and energy spectrum. This default firmware allowed for each of the 32 analog channels to collect and process data. In addition, the package included the open-source DT55xx readout software to allow for a graphical user interface with the firmware. The readout software displays the oscilloscope output, energy spectrum, and access to the signal processing and analog front-end settings. Also, it allows for statistical analysis of the energy spectrum. However, in order to customize the signal processing methods to include charge integration triggered by an external signal, it will be necessary to use the CAEN SciCompiler software to develop a new firmware for the board, and subsequently the CAEN software development kit to create a software capable of interfacing with the firmware.

III. CHARGE INTEGRATION FIRMWARE DESIGN WITH CAEN SCICOMPILER

A. SciCompiler

The SciCompiler software, developed by Nuclear Instruments in conjunction with CAEN, allows for high level block coding of the firmware for the DT5560SE digitizer and the Xilinx FPGA. Rather than starting from scratch using HDL code, this software makes the firmware development process more approachable by allowing for the use of premade digital components for a wide variety of nuclear physics applications. It also includes registers for reading or writing data, digital or analog I/O ports, and a collection of digital logic and timing components. The software's ease of access and

compatibility with the digitizer makes it the obvious choice for the development of the NOPTREX NaI array DAQ system.

In order to use the full functionality of the SciCompiler software, it is necessary to install the Xilinx Vivado software and Microsoft Visual Studios. Vivado can be used to develop FPGA firmware and produce bitstreams with Vivado hardware development language, or VDHL. It allows for the entire prototyping, design, debugging, and construction process within the application. When a firmware is developed and compiled in SciCompiler, the SciCompiler software automatically connects to the Vivado application to construct the design, compile the firmware, and produce the optimal bitstream for the FPGA to execute the customized design.

In addition to producing the firmware, SciCompiler also creates a software development kit based on the firmware requirements in order to assist with the software development process. This kit includes libraries for both Python and Visual C++ that can be opened in Microsoft Visual Studios, as well as an example program to demonstrate the connection to the board. By using these libraries and examples, it is possible to design a software application capable of interfacing with the components included in the compiled firmware. In short, the SciCompiler software provides all the necessary components to make the FPGA firmware development process accessible to those at the introductory level.

B. Charge Integration

For the NOPTREX DAQ system, I have developed a charge integration firmware for processing the signals (Figure 3). The SciCompiler charge integration module takes a trigger signal and an integration time input and measures the area under the input signal over the given integration time, which can be set by a register. The module outputs a single value for the area for each integration window. It also features a pileup inhibitor setting and a baseline subtraction, which can also be set by registers. This module has been implemented into the various iterations of firmware that I have created throughout this project.

In the preliminary firmware, I used a leading edge trigger module to trigger on the analog detector signal. By setting the trigger threshold using a register, the leading edge trigger module outputs a digital signal to the charge integration module for every pulse that exceeds the threshold. However, this design lead to the integration being activated rapidly and for very short periods of time, which produced a low amount of area counts and statistically unsatisfying data (Figure 4). In order to configure the firmware to match the experimental requirements for NOPTREX, I

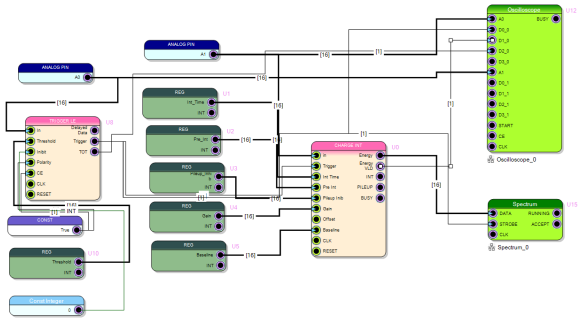

FIG. 3. The SciCompiler block diagram for the charge integration firmware.

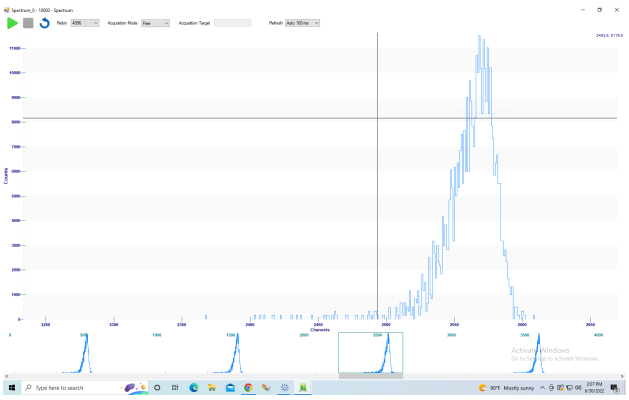

FIG. 4. The results of a Cesium 137 spectrum using the first firmware iteration.

developed more versions with different trigger settings.

A later version of the firmware featured an external trigger signal consisting of a rectangular pulse produced by a signal generator. This low frequency external pulse activated the leading edge trigger module, which then enabled the charge integration. The integration time was set to the period of the external trigger pulse. This lead to better results, since it required fewer integration windows and each window measured a greater area (Figure 5).

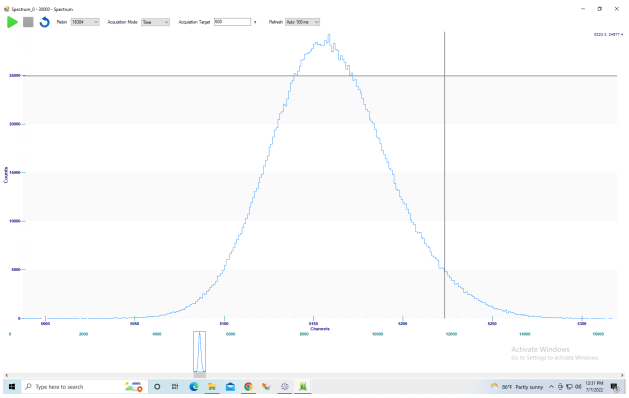

FIG. 5. The results of a Cesium 137 spectrum using the second firmware iteration.

IV. FUTURE TASKS

The next step for this project will be to update the firmware to match the specific demands of the NOPTREX experiment. In order to do this, I will design a trigger system that matches the time of flight of the neutrons from the Los Alamos National Lab neutron spallation source. This trigger will consist of a slow 20Hz external signal which will trigger a cascade of short-duration integrations of either 10μs or 100μs each. This will allow for an individual histogram to be constructed for each neutron arrival pulse with bins of the integration time. Upon the completion of the firmware, it will the be necessary to expand this firmware to all 32 channels of the digitizer and subsequently develop a readout software to interface with the completed firmware.