A Fast Programmable Trigger for Pattern Recognition

G. Gratta, L. Miller, C. Roat, D. Tracy, Y.F. Wang

Physics Department, Stanford University

Abstract
We have built a fast programmable trigger processor based on a state-of-the-art Field Programmable Gate Array (FPGA) IC for the Palo Verde Neutrino Oscillation Experiment. The trigger processor can accommodate 160 ECL input signals, 8 NIM input signals, 16 ECL output signals and 8 NIM output signals. Our two-level trigger logic is designed asynchronously to maximize speed. We have attained trigger times of 40 ns for level 1 and 100 ns for level 2 with 132 asynchronous inputs. The trigger processor can be upgraded by replacing the FPGA with more advanced versions of the chip as they appear.

Submitted to Nucl. Instr. and Meth. A
1 Introduction

Fast, complex trigger systems are central to modern particle physics experiments. While custom-made ASIC-CMOS gate arrays can reach decision times as short as 25 ns (for instance, first level trigger for LHC detectors [1]), the need for full programmability makes the Field Programmable Gate Arrays (FPGAs) a very interesting option. Processors based on FPGA's are suitable for applications in which the exact nature of signal or background is not well known a priori and the trigger algorithm cannot be fully established before the experiment begins. Such processors are flexible enough to be used as general purpose components for different applications. Although for simple experiments commercial modules based on FPGA's are available [2], in our case the number of channels, the complexity of the algorithm and the speed required led us to develop a new module taking advantage of the most recent progress in FPGA technology. The result is a general purpose programmable trigger for pattern recognition which completely fulfills our needs and may be used in other particle physics experiments or any application where fast pattern recognition is desirable.

2 Trigger Requirements for the Palo Verde Experiment

The Palo Verde Neutrino Oscillation Experiment [3] at the Palo Verde Nuclear Generating Station in Arizona, is designed to search for neutrino oscillations with a baseline of ~ 1 km. It will measure the electron anti-neutrino spectrum from the Station's reactors and search for distortions that would indicate $\nu_e \rightarrow \bar{\nu}_X$ oscillations. The detector consists of a $6 \times 11$ array of 9 m long cells filled with Gadolinium-loaded liquid scintillator. Anti-neutrinos are detected through the inverse $\beta$ decay $\bar{\nu}_e + p \rightarrow n + e^+$. Positrons deposit their energies in the scintillator and annihilate, yielding two $\gamma$'s. This process generates a triple coincidence signal in the detector. Neutrons thermalize in $10 - 100 \mu s$ and are captured in the Gd, giving a $\gamma$-ray shower of 8 MeV total energy. The 12-ton central detector is surrounded by a 1 m thick water shield and by a large cosmic ray veto counter. The detector is located in a shallow underground laboratory at a depth of 25 m.

Although the signal rate is low (~ 50 events/day), the background from
natural radioactivity and cosmic ray-induced neutrons is quite large. For this reason it is important to have a fast and efficient trigger which rejects background as quickly as possible minimizing the dead-time.

Extensive Monte Carlo studies show that we can trigger on positrons and neutrons separately and match the two sub-events off-line. With proper time stamping, this will allow us to measure the background of random coincidences which has a time correlation between the two sub-events different from that of the $\bar{\nu}_e$ signal. Although the FPGA can accommodate two independent trigger conditions, the trigger condition for the positron and neutron sub-events can be set to be similar: one high energy hit ($\geq 500$ keV) to tag the initial positron energy deposition and the core of the photon cascade from neutron capture, and two or more low energy hits ($\geq 50$ keV) to detect the annihilation photons and the remnants of the neutron capture cascade. High and low signals should appear locally, within a submatrix of cells called a "gang." Two gangs of $3 \times 5$ cells are shown in Fig. 1. Since the backgrounds from natural radioactivity and cosmic ray-generated neutrons are not well known prior to the experiment's start-up, the trigger processor has to be flexible enough to be reconfigured for different high-low patterns and gang layouts over the course of the experiment.

Figure 1: Schematic cross-section of the $11 \times 6$ array of scintillator cells in the central detector. Slashes indicate the layout of two gangs ($3 \times 5$ in this case). One gang is denoted by forward slashes, the other by back slashes. Crosses result from overlapping gangs.
Speed is a major concern since the DAQ system requires the trigger signal to begin digitization. While in accelerator experiments a beam gate can be used to tag the start of an event, in our case this function must be performed by the trigger. Analog signals can be "held" in high-quality coaxial cables but pulse shape degradation and cost considerations substantially limit the holding time. For these reasons we require a first-level trigger decision time of less than 100 ns.

Since most ADC's and TDC's are too slow to digitize within the neutron capture time that can be as short as 10 µs, we employ two separate banks of ADC's and TDC's, one for positron sub-events and the other for neutron sub-events. Data arriving from the detector are digitized by the second bank if the first one is busy. The trigger processor must be able to direct its signals to the appropriate bank depending on the digitizers' busy status and trigger conditions.

As the scintillator-filled detector cells are 9 m long, signals arriving at the trigger module are spread over 50 ns, while their width is 20 ns, determined by front-end electronics. Hence, the incoming signals must be latched to perform pattern recognition.

Given the complexity discussed above and the flexibility needed, we excluded the use of hard-wired devices such as discrete components, gate arrays, EPROM look-up tables, or Complex Programmable Logic Devices (CPLD's). A sufficiently complex FPGA is the ideal solution to this problem. FPGA technology is based on SRAM, which can support very complex, high density logic designs, and has large input/output capabilities. The largest chip available at the time of the design was the Xilinx XC4025EX [4], which has the equivalent of 20K-48K logical gates in 1024 Configurable Logical Blocks (CLB's), 256 input/output channels and 2560 flip-flops. There is a 3 ns propagation delay through each CLB\(^1\), and the flip-flops can be clocked at a speed of up to 100 MHz, depending on the complexity of the logic design. The configuration is lost during a power-down, but it can be reloaded in several milliseconds from an on-board EPROM or from a computer through a dedicated Xchecker cable.

Since the trigger processing time depends on the algorithm used, in the following we describe in some detail the trigger logic design and how we achieve our performance goal.

\(^1\)Chips with 2 ns delay will give a 30% improvement on speed.
3 Trigger Design

We have used the XACT™ software provided by Xilinx to design, debug and simulate our logic. The trigger processor accepts the logic inputs from the high- and low-threshold discriminators (one for each cell). These signals are generated by front-end electronics. In order to reduce the length of delay cables for analog signals and to tolerate possibly high background rates, we have designed a two-level trigger system. A simplified logic schematic is shown in Fig. 2.

Figure 2: Simplified trigger logic schematic. The first-level trigger is simply a logical OR of the 66 incoming high signals. The second-level trigger performs pattern recognition on each of the 28 gangs in the 6 × 11 central detector array.
The first-level trigger consists of a logical OR of all high-energy hits, and as such can be performed very quickly (\(\sim 40\) ns including receiver and driver delays). Once a high hit is detected, a gate is sent to the ADC's to begin charge collection.

At second level, the trigger runs the whole algorithm using high and low signals to find patterns as explained above. If the event is accepted, the ADC's will continue digitization and a common stop will be sent to TDC's to start digitization. If the event is rejected, a clear signal will be sent to the ADC's to abort digitization. The clear will also be sent to TDC together with the common stop.

The signals from the input latches enter the logic stage grouped in gangs. The gangs are of user-defined size and shape; they determine how close in space signals must be in order to be considered correlated. Gangs overlap one another and completely fill the \(6 \times 11\) grid. A total of 28 gangs of \(3 \times 5\) size cover the whole detector. Hit patterns in each gang are evaluated in parallel to speed computation.

The algorithm proceeds by first counting the number of highs and lows which occur within the boundaries of each gang. Both the high count and the low count are compared to user-defined thresholds. The FPGA can accommodate two distinct sets of thresholds for positron and neutron signals. Currently, both positron and neutron thresholds are \(\geq 1\) high hits and \(\geq 2\) low hits. Should any of the 28 gangs meet or exceed the threshold conditions, a second-level trigger is produced and the event is accepted. This process takes a total of 100 ns.

For events rejected at the first level, the FPGA is reset and then ready to receive new signals. The reset of the whole chip takes about 80 ns. This is shown by the simulated timing diagram in fig. 3a, where the width of the reset pulse represents the time for resetting the chip. The total time needed for fast rejection is about 170 ns including the 50 ns time jitter of input signal.

For events rejected at the second level, the total dead time (decision time plus reset time) is about 250 ns. Since the old Lecroy 1885N Fastbus ADC's in our experiment only accept fast clear signals after the termination of the gate (300 ns in width), and the fast clear takes about 600 ns, the clear signal is issued after \(\sim 400\) ns and the chip is reset after \(\sim 1\) \(\mu\)s. This is shown in the simulated timing in fig. 3b. Fig. 3c shows an event accepted at both levels. Note the timing refers to the chip simulation and our board will provide pulse
Figure 3: Simulated timing for a) first-level rejection, b) second-level rejection, and c) an accepted event. Note the different time scale in b). The timing refers to the chip simulation and our board will provide pulse stretching in some cases and also introduce some small delays.
stretching in some cases and also introduce some small delays.

We have used a 20 MHZ external clock to drive timing functions of the FPGA such as flip-flops and counters. Although simpler logic designs can tolerate higher clock speeds, attaining sub-microsecond decision times with a 20 MHz clock requires the use of asynchronous logic. In this way the trigger speed depends only weakly on the clock speed. However in this mode the design is complicated by the fact that path delays within the FPGA chip are hard to characterize. Careful design is needed to overcome this problem.

The current logic uses approximately 800 CLB's, ~ 80% of the chip capacity. Since the number of CLB's used depends on the complexity of the algorithm, we select a FPGA chip with enough room to allow for future changes. For instance, we may need to tag neutron and positron sub-events with different conditions, and/or to change the size of the gangs. These changes may increase the CLB's needed and pushes the design to the limit of the current generation of FPGA's. For this reason, we have included on our modular board a socket for an auxiliary FPGA. We also expect to replace the current chip, XC4025EX, with new chips such as XC4036EX, which has the same pinout and almost twice as many CLB's.

4 Hardware Implementation

The trigger module houses the main FPGA chip, an auxiliary FPGA, I/O connectors, ECL-TTL and NIM-TTL converters, the EPROM holding the FPGA configuration, and miscellaneous test, diagnostic, and downloading circuitry. The schematic is shown in Fig. 4 and an assembled board is shown in Fig. 5.

The Printed Circuit (PC) board is a 20.3 cm × 36.1 cm four-layer board with 150 μm traces. All connectors, switches, test, and downloading circuitry are mounted on one side, while all chips that do not need to be accessed are mounted on the opposite side (shown in fig. 5).

The logic signals to and from the FPGA's are TTL CMOS, and need to be interfaced with the ECL logic signals of the rest of the trigger and data acquisition system. Motorola 10H124(10H125) chips are used to convert TTL to ECL (ECL to TTL). Also built on board are several discrete component NIM-TTL converters to interface with external NIM modules (the muon veto signal, for example, is produced by NIM discriminators). In total, we have
Figure 4: Schematic of PC board for the trigger module
160 ECL input signals, 8 NIM input signals, 16 ECL output signals, and 8 NIM output signals.

Programs for the FPGA chips can be downloaded using an EPROM chip or via an XChecker cable plugged directly to a serial port of the computer. Both protocols have been built into the PC board and the user may switch from one to the other easily.

As described above, in order to have enough flexibility to accommodate more complicated trigger programs, we have included a socket on the PC board to house a secondary FPGA chip, such as the XC4005H, XC4013E, or even the XC4025EX. The newly developed FPGA chips like the XC4036EX, which are pin to pin compatible with the XC4025EX (package HQ304), can be easily replaced on the board to improve the performance.

The trigger processor has been fully tested and the results have been compared with software simulations. It is very encouraging to see that simulated timing data (40 ns level 1, 100 ns level 2) coincide with the measurement within ± 5 ns.
5 Conclusions

We have realized a fast trigger processor based on a state-of-the-art Field Programmable Gate Array logic chip. The processor can be programmed to perform pattern recognition with up to 160 ECL input channels. Our algorithm is organized in 2 levels, with 40 ns and 100 ns decision times. We have found good agreement between the performance predicted by the simulation and the measurements on the hardware. Enough hardware and software flexibility has been built into the processor that it could be used in a variety of different experiments.

6 Acknowledgments

We would like to thank the Xilinx Inc. for providing us XC4025EX-3 chips free of charge. We would also like to thank our colleagues from Caltech, F. Boehm, B. Cook, N. Mascarenhas and A. Piepke, for many useful discussions. The assistance of E. Crabill and K. Olukotun of the Department of Electrical Engineering, Stanford University and M. Freytag of SLAC have been invaluable. We are indebted to Ben Turek for work on the drawings. This work is supported in part by DoE Grant # DE-FG03-96ER40986.

References


