Project-Based Learning Curriculum for the Junior Year Based on Building a Laser Tag System

Prof. Brad L. Hutchings, Brigham Young University

Brad L. Hutchings received the PhD degree in Computer Science from the University of Utah in 1992. He is currently an associate professor in the Department of Electrical and Computer Engineering at Brigham Young University. In 1993, Dr. Hutchings established the Laboratory for Reconfigurable Logic at BYU and currently serves as its head. His research interests are custom computing, embedded systems, FPGA architectures, CAD, and VLSI. He has published numerous papers on FPGA-related topics and is an inventor/co-inventor for 60+ patents.

Prof. Stephen Schultz, Brigham Young University

Stephen M. Schultz has received B.S. and M.S. degrees in electrical engineering from Brigham Young University, Provo, UT, in 1992 and 1994, respectively. He received a Ph.D. in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, in 1999. He worked at Raytheon Missile Systems from 1999-2001. He has taught at Brigham Young University since 2002 and is currently a Full Professor. He has authored or coauthored over 100 publications and holds 10 patents. His research interests are in the area of optical fiber devices with an emphasis on optical fiber based sensors.

Project Based Learning Curriculum for the Junior Year Based on Building a Laser Tag System

Brad L. Hutchings and Stephen Schultz
hutch@ee.byu.edu, schultz@ee.byu.edu
Dept. of Electrical and Computer Eng.
Brigham Young University

1 Introduction

This paper describes a Project Based Learning (PBL) curriculum[1] that spans the junior year of the Electrical and Computer Engineering Department. This curriculum consists of two, lock-step semesters. During fall semester all juniors (120+ students) enroll in three, four-credit-hour core classes: 1) analog circuit design (ECEn 340), 2) digital signal processing (ECEn 380), and 3) embedded programming (ECEn 330). During winter semester students practice the concepts learned during these earlier core courses by constructing an advanced laser-tag system (alternatively referred to as the junior project). Laser-tag is an excellent target because it provides an engaging way to integrate the concepts and practices from very different areas of electrical and computer engineering.

The goals of this PBL curriculum are to: 1) increase student confidence, 2) provide students with a fun engineering experience, 3) provide opportunities for application of concepts from prior junior courses[2], and 4) administer the PBL curriculum so that, in the long term, TA and faculty loads are reasonable. Student confidence increases as students participate in a challenging project with a high potential for success. TA and faculty loads are managed by the availability of: 1) a dedicated youTube channel that provides a series ¡of "how-to" and demonstration videos, and 2) a comprehensive set of test software and hardware fixtures that help students to incrementally test their system to ensure that each implemented module meets specifications and is bug-free. The "how-to" videos teach students how to use commercial design software based on best practices. Demonstration videos depict, in an unambiguous way, the system behavior that is expected during the pass-off of each milestone.

The resulting laser tag system runs on battery power, is portable, can detect "hits" from opponents that are up to 120 feet away (in daylight) and supports game-play between up to 10 players. The laser tag system is built using a high-performance embedded system that consists of an ARM processor, an Analog to Digital Converter (ADC), touch-screen TFT and other support components. Students build analog circuits that interface between the embedded system and a Nerf gun that has been retrofitted with an LED and a photodetector. Students implement real-time algorithms on the embedded system that transmits a modulated beam of light from an LED when

the gun-trigger is pressed, and that can detect "hits" from the modulated beam from an opponent's gun. The entire system consists of about 4,200 lines of C code; about 1/2 of that is written by the student, the other 1/2 is instructor-provided test and verification code.

2 Motivation

It has been the author's experience as an instructor that many (most) students select the electrical and computing engineering major because they ultimately want to learn how to"build things". Although EE courses often have lab sections where students learn practical applications of theoretical concepts, lab sections typically consist of focused exercises of limited breadth and depth. Students also participate in a senior-project experience in their final year of studies; however, the complexity of the senior project is usually severely limited by the student's lack of experience. Thus neither the typical lab section nor the senior project effectively integrate knowledge from multiple domains[3]. As such students often lament (in senior exit interviews) that they don't understand how the various parts of the discipline "fit together" and that they lack the confidence to attack the implementation of an entire system. Put more colloquially, students are actually asking: "what is all the stuff I learned in the last four years good for?", and "what can I do with it?" As instructors, we found this feedback disheartening and set out to enrich the student experience such that students:

- practice, in a concrete and significant way, the concepts and theories learned across a broad range of EE courses[3],
- gain confidence in their ability to implement complex systems with a broad range of requirements and specifications, and
- experience the joy and satisfaction that comes from successfully implementing, debugging and testing a complex system.

Or, to put it more colloquially, we want the students to know "what all this stuff is good for" when they graduate from our program. We genuinely want our students to be excited about their potential as they graduate and move on to their professional assignments.

3 The Laser-Tag System

Laser tag was chosen as the target system for the following reasons.

1. *It is a game*. Once operational, games are fun and engaging and students are often motivated to work harder on games than other topic-oriented labs. In addition, unlike the results from labs in other courses, students can proudly show their working system to friends and parents who will actually comprehend the system and its intended function).[1]

[1]During the latter weeks of the course, instructors will often find students in the lab with spouses, friends, etc., showing off their work.

2. *It integrates concepts from a broad swath of the EE curriculum.* As designed, our laser-tag system requires analog circuitry (transistors, photo-diodes, LEDs) to generate and to detect modulated light signals, a high-performance embedded computing system with complex programming to control the system operation, and Digital Signal Processing (DSP) algorithms to process incoming data and to detect signals at low signal-to-noise ratios.
3. *It supports self-test.* Every laser-tag hardware system contains a transmitter subsystem and a detector subsystem; a complete two-player game consists of two identical sets of hardware and software. However, testing of **all** software can be completed, on a single hardware system, independent of student-designed analog electronics, by connecting the digital output from the transmitter subsystem directly to the detector ADC on a single hardware system (using provided jumpers). This bypasses all analog electronics and makes it possible for students to completely, and individually, test their embedded software using only the embedded hardware PC board. Once software passes all tests, students attach the previously-tested analog gun circuitry to the embedded computing system and the vast majority of systems are fully operational as soon as students attach the gun to the hardware system. If problems occur, they are limited to minor problems with cables between the gun, analog electronics, and the digital hardware, and are usually fixed quickly.

3.1 Laser-Tag Design Principles

The laser-tag system was designed to achieve the following goals:

- Integrate Knowledge From Previous Courses. The primary goal is to reinforce learning and knowledge from previous courses and to thereby increase confidence. As such, laser-tag is just a vehicle to help achieve that goal. The implemented system differs considerably from commercial laser-tag systems; in these student-implemented systems players are assigned to different frequencies and filtering techniques implemented with DSP algorithms are employed to detect "hits" from opponents. In contrast, commercial laser-tag systems operate more akin to the typical IR remote used with TVs.
- High-Success Rate. Success is defined as a working system that meets all specifications. This project would fail to meet one of its primary goals if a significant number of students failed to implement a working system.
- Reasonable Faculty and TA Loads. The laser-tag junior project is a required course for all students and will ultimately need to accommodate about 120-140 students per year when it reaches steady state. Course effectiveness won't matter if the department can't commit the necessary resources to support the course in the long term.

3.2 Specification and Implementation Strategy

The laser-tag system, as completed by students, is designed such that it will support up to 10 simultaneous players, allow game play atxs 40-feet to 100-feet distances, operate for at least an hour on battery power, and be interoperable with all of the other laser-tag systems built by

students participating in the junior project. In addition, a red LED is chosen as the optical source to keep the cost down, to make the systems easier to align, and to ensure that it meets optical safety standards.

Students implement the laser-tag system by completing a series of fixed milestones that were previously defined by the instructors. Milestones break the system into digestible concepts and major subsystems. Milestones, in turn, are typically broken down into a series of tasks to help students manage workload and project timeline.

Milestones and their constituent tasks are designed to have clear, unambiguous requirements and pass-off criteria; each task requires 1-2 weeks and the longest milestone consists of 3, 2-week tasks, for a total of 6 weeks. In cases where system behavior may be complex, youtube videos, color plots, etc., are often used to show exactly how a properly working system should appear so that students know if they have completely met the requirements for pass-off. Comprehensive testing and strict pass-off requirements are essential for the junior project; approximate or ambiguous pass-off requirements that allow interpretations by the TAs or students will ultimately lead to non-functional systems and considerable frustration by students. Student success is defined, in part, as a fully-working system that meets the original specifications and the path to success is to guarantee that each system module meets an unambiguous set of requirements so that it will work correctly when connected to other similarly-tested modules.

3.3 Fixed System Organization

The overall system organization is fixed by the instructor and students are required to implement the system as described in the milestones and tasks. For the software modules, for example, the interface for each module is described by the function prototypes contained in a provided `.h` file and students are not allowed to modify this file (students create the corresponding `.c` file). Hardware circuit laboratory exercises are similarly constrained by specifying the inputs and outputs, their voltage ranges, etc. This approach limits student creativity somewhat. Students are free to implement the internal code in a module as they see fit; however, each module (hardware and software) must meet fixed interface and performance requirements. Transmit frequencies are fixed for the project so that all student systems are interchangeable so that they can play the game in groups. There are several benefits to this fixed organization.

- TAs can help students debug their software and hardware. In a class of 120-140 students, it is impossible for TAs to help students debug each custom implementation. However, because of the fixed organization used here, previous students of the course make excellent TAs as seniors because they will be helping students to debug problems similar to those they faced the previous year.
- The project mirrors industry practice. Most engineers, especially junior engineers, typically do not specify the overall system organization of a product. This task is usually led by a more experienced engineer who functions as the product architect with other engineers implementing and testing the various subsystems (as is done in this project).
- Students can implement much more complex systems. If the students were required to

design and organize the entire system, a different, much simpler system would have to be chosen. From the view of this instructor, without a fixed organization it would be impossible to provide a meaningful engineering experience that incorporates knowledge from multiple courses.

4 System Organization

The junior project consists of two major subsystems: 1) the embedded computing hardware and its associated software, and 2) the analog circuitry that includes the LED and photo-detector contained in the Nerf gun. Students develop and test these subsystems independently and then bring them together near the end of the course. The major subsystems are discussed in the following sections.

4.1 Embedded Computing Subsystem Hardware/Board

The embedded computing subsystem consists of the following components:

- *Digilent ZYBO FPGA board*. The ZYBO board houses (listing only the components used for the junior project): a Xilinx 7010 ZYNQ FPGA along with 512 MB DDR3 memory, microSD slot, JTAG/USB port, and six PMOD ports that can be used to connect the ZYNQ device to peripherals. The ZYNQ device provides a 12-bit, 1MSPS Analog-to-Digital Converter (ADC), a programmable FPGA fabric and a dual-core 650 MHz ARM Cortex-A9 processor. This board was purchased from DigilentInc.com.
- *320x240 TFT touch display with associated touch controller*. The TFT display provides an output for verification and diagnostics as well as a way for students to add features to the junior project. Purchased from AdaFruit.com
- *Bluetooth modem*. This can be used to extend the system and pair it with a phone during the creative portion of the junior project. Source code for a sample cellphone application is provided as a starting point for students. Purchased from AdaFruit.com.
- *Baseboard.* The baseboard is a PCB that integrates the above components and also provides jumpers to connect the transmitter and detector for testing purposes.

The embedded computing subsystem is shown in Figure 1. In addition to the items described above, please note the additional labeled items in the drawing.

- *Jumper Block*. When jumpers are inserted as shown in the figure, the ADC (the input from the photodetector circuitry) is electrically connected to the transmitter output (the output to the LED circuitry) and students can completely test all software functionality without using any analog circuitry or external test equipment. In this mode, the one-board laser-tag system is essentially "shooting itself". This isolates the embedded computing system from the gun circuitry (which is tested by a separate fixture). Testing then becomes much simpler and far more deterministic than if students were manually aiming guns at detector circuitry, etc. Once correct software functionality is observed (again, using comprehensive test

software), the jumpers are removed and attached to cables connected to the appropriate circuitry in the gun.

- *Attenuator*. During verification, students twist the potentiometer to adjust the output voltage from the transmitter effectively simulating distance changes between the transmitter and the detector.

4.2 Embedded Computing System Software

The functionality of the embedded computing system software can be roughly divided into three areas: the transmitter, the detector, and system control. The software functionality is shown in block-diagram form in Figure 2. Two inputs and two outputs to the system are shown. The output of the photo-diode circuitry connects to an ADC that samples the incoming analog signal at 100 kHz and converts the analog value into a 12-bit digital value. The other input, the gun-trigger switch, enables the transmitter to run for 0.2 seconds each time the trigger is pressed. The transmitter output is connected to the gun LED and a hit-indicator LED is also provided on the embedded computing board for debugging purposes.

4.2.1 System Control

System control consists of three software state-machines[2]. Behaving similarly to their hardware-based counterparts, these C-based software state-machines use a "tick" function that is invoked at a continuous 100 kHz rate. Each time the tick function is called, the state-machine evaluates its inputs (static internal and global variables), determines the next state, and then invokes the appropriate function or sets some global variable. For example, the *trigger* state-machine (see Figure 2) is used to debounce the incoming gun trigger switch and to control the *transmitter* state-machine. Specifications require that the state-machine completely debounce the trigger switch and enable the transmitter only once for each trigger-depress and release (automatic rapid-fire is not allowed). The trigger state-machine accomplishes this task by timing the incoming switch input to ensure that it has stopped bouncing and then calling an enable function for the transmitter state machine when it detects that the trigger has been depressed and released.

The *hitLedTimer* state-machine and *lockoutTimer* state-machine (again, see Figure 2) provide timing-related functions as suggested by their names. hitLedTimer illuminates an LED for 0.5 seconds each time the detector processes a hit from an opponent. The lockoutTimer disables the detector for 0.5 seconds each time a hit is detected; this improves game play by allowing a player to escape from a group of players who are all firing their guns simultaneously at the player.

[2]Software-based state-machines are an effective way to organize and implement reactive software but an in-depth exposition of software state-machines is beyond the scope of this paper.

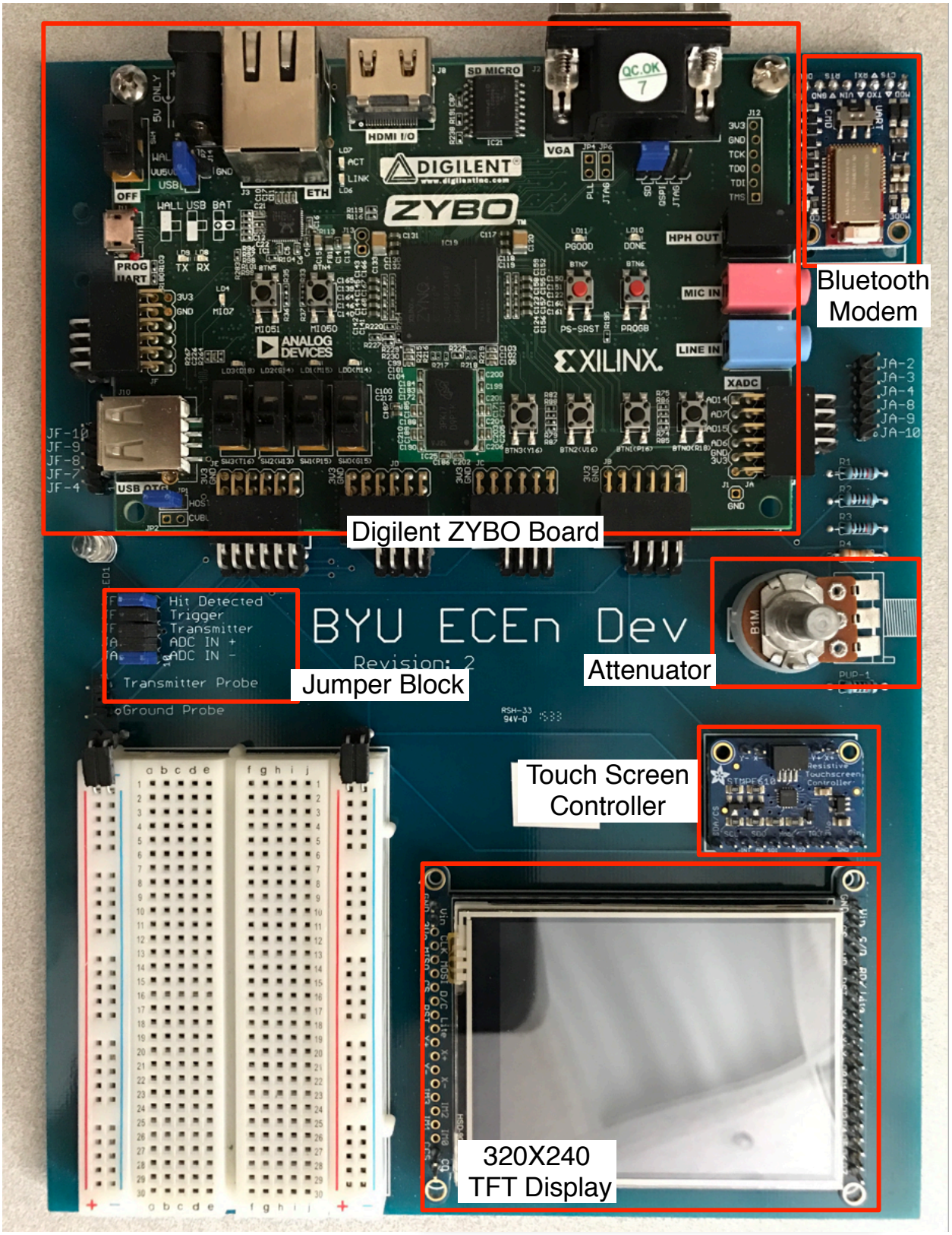


Figure 1: Embedded computing hardware

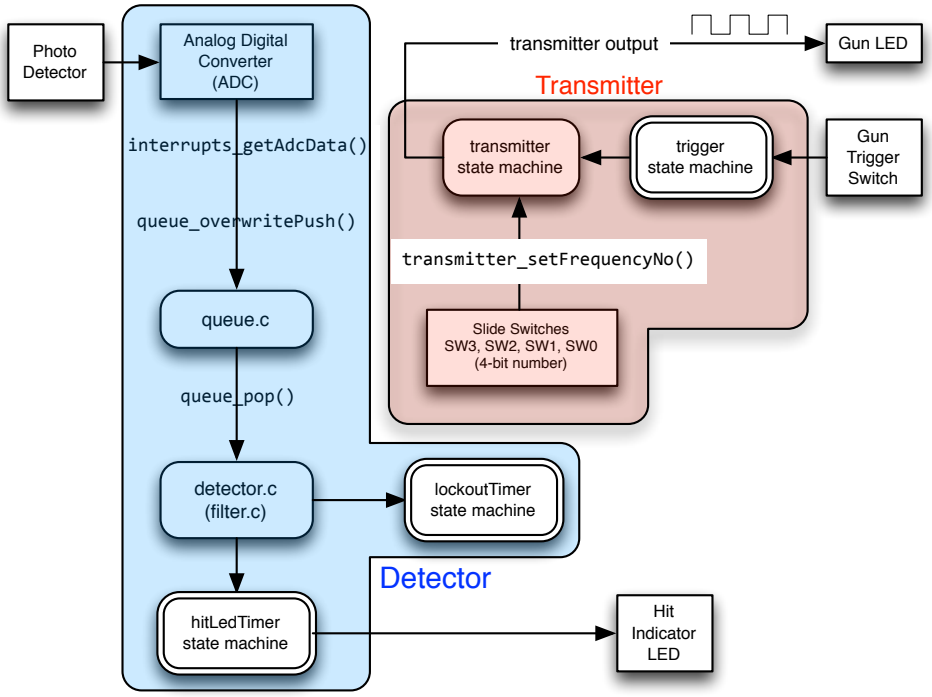


Figure 2: Embedded Computing Software

4.2.2 The Transmitter

The transmitter is a relatively simple software block that is responsible for generating a square wave at one of 10 selectable player frequencies that range from about 1 kHz to about 4 kHz (the set of 10 frequencies are fixed for the entire lab ensuring that all systems can interoperate). As with the system-control software, it is also implemented as a software state-machine with a tick function that is invoked at a frequency of 100 kHz. The transmitter output is a 50% duty-cycle square wave; as such, the transmitter state-machine simply sets the signal driving the LED to a logic '1' for one-half the user-chosen period and then sets the signal to a '0' for the other half[3].

4.2.3 The Detector

The detector module is the most complex part of the entire project and is where the students apply their knowledge of DSP algorithms. Based upon filtering techniques and algorithms taught in a previous course, students design and thoroughly model the necessary filters with MATLAB during the signal-processing milestone. A simplified block diagram of the detector is shown in Figure 3. Data arrive from the ADC at the rate of 100,000 samples per second (100 kHz). The 12-bit values are scaled and converted to a floating-point value between -1.0 and 1.0 and are then processed by a decimating FIR filter. The 81-tap decimating FIR filter removes signals with

[3] A sinusoidal output was also tested but generating a sinusoid does not improve performance commensurate with the added complexity.

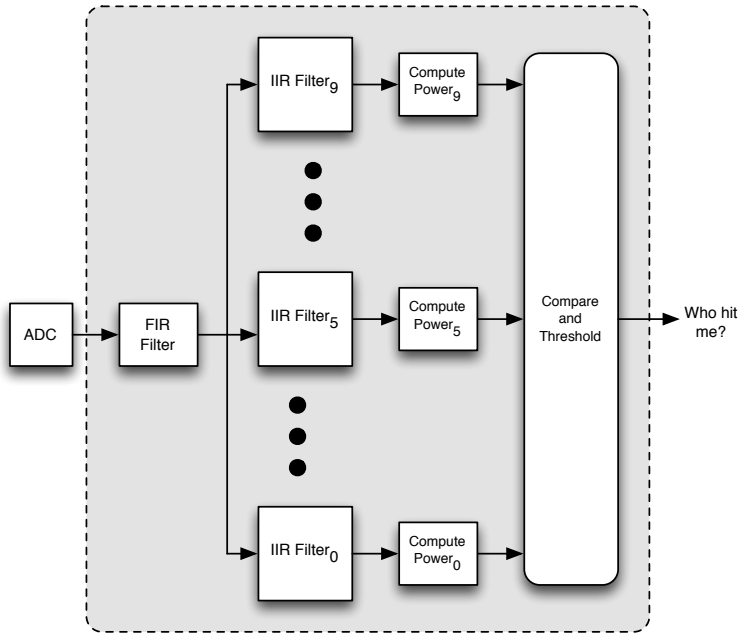


Figure 3: Detector Organization

frequency components above about 5 kHz (user frequencies range from about 1 kHz to 4 kHz). This removes noise generated by fluorescent lights, for example, and insures that the noise doesn't get aliased into the user frequencies. The FIR decimates the incoming data by a factor of 10; this dramatically lowers the computational load of the detector because the remaining bandpass filters are only invoked on 1/10 of the original ADC data.

Once processed and decimated by the FIR filter, the data are passed to 10 band-pass filters implemented as IIR filters (IIR is used because it can be implemented with many fewer coefficients than a FIR filter, for example). Data from the band-pass filters are processed by software that computes the total power contained in the signal after it has passed through each band-pass filter. Power is computed for 2,000 samples - 0.2 seconds of data - and the computed power values are passed to the compare-and-threshold unit.

The compare-and-threshold unit sorts the incoming 10 power values in ascending order. The middle power value is selected from the sorted values and is then multiplied with a constant to create a threshold value. The maximum power value (from the sorted list of power values) is compared against the threshold; if it is greater than the threshold, a "hit" has been detected, otherwise, no hit has occurred.[4] Sorting the power values and then using the median value to compute a threshold makes the laser-tag system immune to most noise that, in practice, typically occurs simultaneously across most/all user frequencies.

[4]The constant is chosen through experimentation; however, the detector performance is not particularly sensitive to the constant value. In practice, values from about 200 - 1600 work well.

4.3 Analog Subsystem

As has been previously discussed the high success rate is attained by (1) creating a fixed overall system organization with fixed interfaces and (2) creating specific testable modules. The fixed analog subsystem includes a laser tag gun that is a modified Nerf gun and a board that interfaces between the laser tag gun and the embedded computing hardware. The students design, build, and test two electronics boards with fixed interfaces. The first board, called the LED Driver Board, is the interface between the LEDs on the laser tag gun and the embedded computing hardware. The second board, referred to as the Receiver Board, is the amplifier between the photodetector and the ADC located on the embedded computer hardware. The fixed interfaces enable any student board to be plugged into a system. These construction and testing of the electronics boards is accomplished in ECEn 340, a course taken during the Fall semester.

4.3.1 Modified Nerf Gun

The laser-tag gun begins life as a commercially-available Nerf®Firestrike gun that is well-suited to this project. As sold, this commercially available product uses a visible LED+lens assembly for aiming purposes and ordinarily shoots foam darts when an electronic trigger is pressed. The department bought a quantity of these Nerf guns from Toys 'R Us and modified them as follows.

- The original LED is replaced with a higher power CREE C503B-RAS LED. This LED has a 15-degree emission angle and can handle currents of 100 ma. This high power LED is mounted onto the original mounting board so that it maintains alignment with the original lens.
- The foam dart shooting mechanism is removed and a BPW34 photodetector from Vishay is inserted in its place.
- An additional "hit" indicator LED is added to the top of the gun so that can students can visibly determine hits during test and actual use. The hit LED is connected to an output pin on the embedded computing system that is controlled by the detector software.

The trigger, shooting LED, and hit indicator LED are connected to a four-wire cable. The photodetector is connected to a coaxial cable to reduce noise. Figure 4 shows an external view of the gun while Figure 5 shows the internal view of the modified gun.

Figure 6 shows how the gun's external cabling is attached to the internal components. All components are powered by a single voltage provided on one common wire and each component (excluding the photo-detector) contributes one additional wire as shown. As previously mentioned, the photodetector uses a separate coax cable to preserve signal integrity. The modified Nerf gun with attached cabling, including standard connectors, is provided to students as part of the project hardware.

As part of the project, the students must design and implement the analog circuitry that interfaces between the gun cabling and the embedded computing hardware. Figure 7 shows three major subsystems (Embedded Computing System, Transmitter Board, and Receiver Board) along with

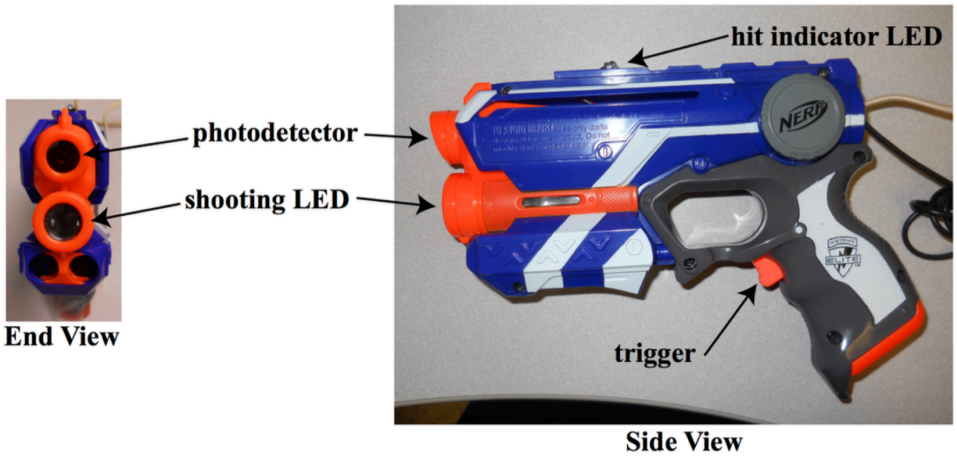


Figure 4: Modified Firestrike Nerf gun.

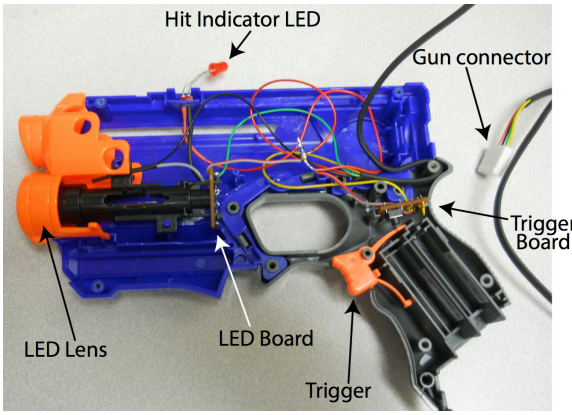


Figure 5: Internal View of Modified Nerf Gun

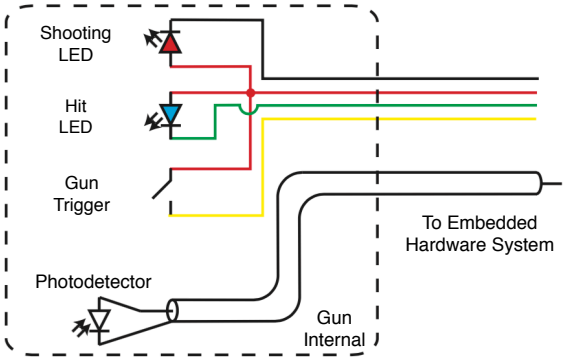


Figure 6: Modified Nerf Gun Cabling

several connections and two batteries. The receiver board contains the analog circuitry that amplifies the signal from the photodetector. The transmitter board contains circuitry that accepts a digital signal from the embedded computing system and amplifies the signal so that it can drive the previously mentioned CREE LED. For convenience, the transmitter board also contains circuitry to drive the hit-LED that is located on the top of the gun and a resistor divider circuit used in conjunction with the trigger switch. The intermodule connections show two inputs and two outputs between the embedded computing system and the receiver and transmitter boards. One digital output from the embedded computing system drives the hit LED; the other digital output is a digital output that drives the shooter LED analog circuitry. The trigger switch from the Nerf gun provides one of the inputs to the embedded computing system. The other input is an analog signal from the photodetector that has been amplified by analog circuitry designed by the students (all analog design is performed during a previous course, ECEn 340).

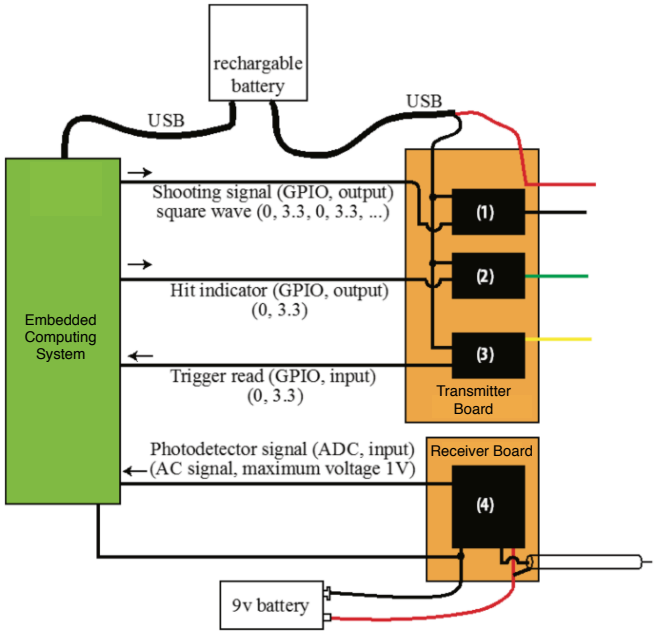


Figure 7: Connections between embedded system hardware and modified Nerf gun. The 4 components are (1) shooting LED driver, (2) hit indicator LED driver, (3) trigger signal, and (4) amplified photodetector signal.

4.3.2 Interface

For the sake of durability and to make robust game play feasible, the embedded computing hardware is placed in a closeable plastic box (the boxed components are placed inside a backpack during game-play). As shown in Figure 8, all non-analog signals are conveyed from the gun to the embedded hardware system via a ribbon cable that has one end attached to the "Digital Interface". This digital interface is the jumper block that was discussed previously as the mechanism that enables testing and verification by attaching the transmitter subsystem to the detector subsystem. Once testing is complete, the testing jumpers are removed so that the ribbon cable can be attached. The other end of the ribbon cable is attached to a board labeled "Analog Interface" that provides connectors for the gun cabling, and to the analog circuitry that is designed and implemented by the students.

Figure 9 shows that the analog interface board has four separate connections listed below.

1. Connection to the LED Driver Board that the students construct in ECEn 340.
2. Connection to the Receiver Board that the students construct in ECEn 340.
3. BNC connection to the photodetector located in the modified laser tag gun.
4. Connection to the laser tag gun (shooting LED, hit indicator LED, and trigger).

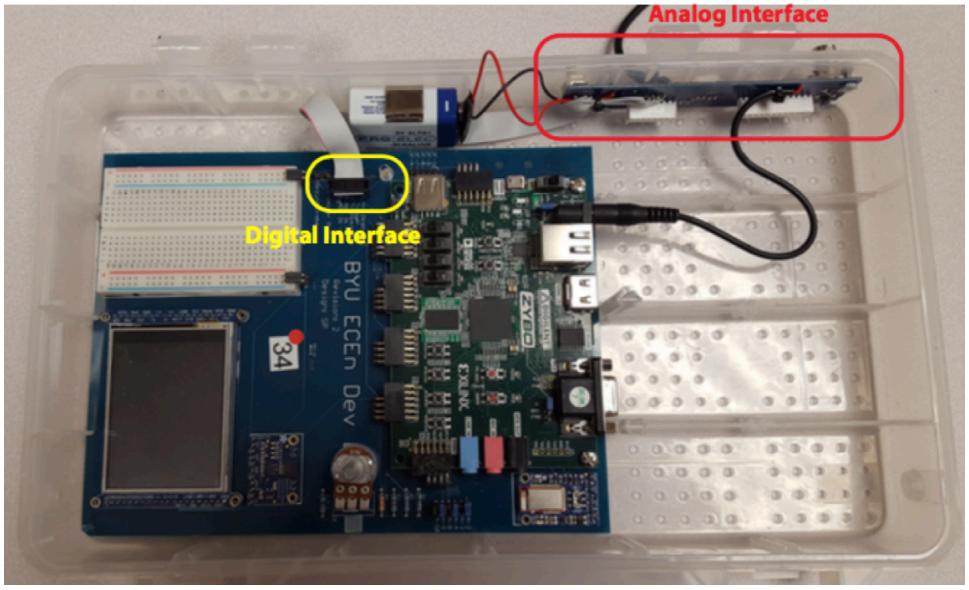
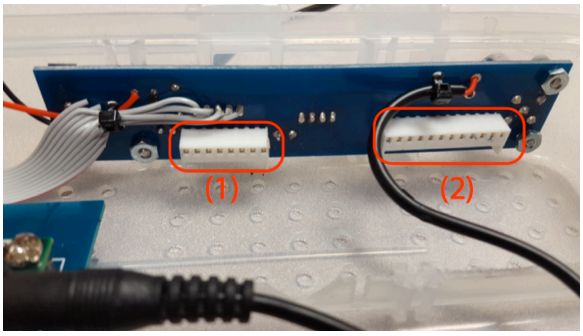
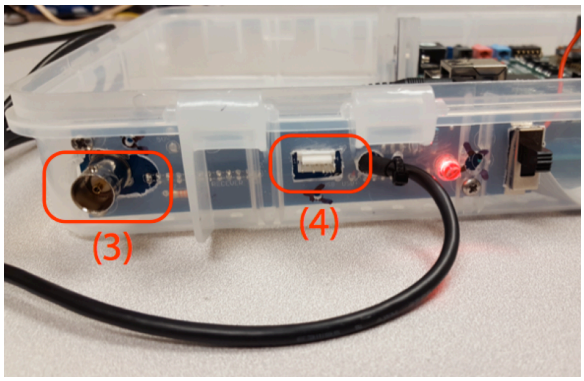


Figure 8: The laser tag system packaging.

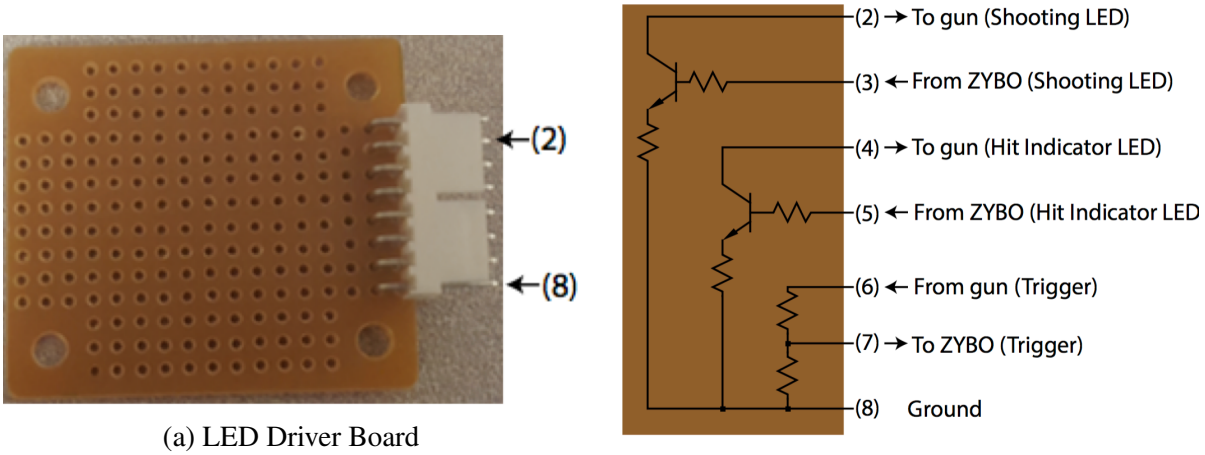


(a) Inside of Box



(b) Outside of Box

Figure 9: The Analog Interface Board



(a) LED Driver Board

(b) LED Driver Schematic

Figure 10: Student LED Board and Schematics

4.3.3 LED Driver Board

A previous figure (please see Figure 7) shows that three of the electronic circuits are fabricated onto a single circuit board called the "Transmitter Board". Referring back to this figure, these three circuits are (1) the Shooting LED Driver, (2) the Hit Indicator LED Driver, and (3) the Trigger.

Section 4.2.2 explains that the transmitter subsystem (implemented as software on the embedded computing system) generates a square wave at one of 10 selectable player frequencies. Whenever the signal is set to a logic '1' the shooting LED circuit produces a current of 100 mA. Since the students will not have working embedded computing system software during fall semester (the junior project takes place during winter semester), the LED driver is tested using a function generator that outputs a square wave with a frequency of 1 kHz, a low voltage level of 0V and a high voltage level of 3.3V. The Hit Indicator LED driver is the same as the Shooting LED driver with the only difference being that the LED current for a logic '1' is 10 mA. The trigger circuit is simply a voltage divider configured as a pull-down resistor that converts the +5V supply to a +3.3V signal when the trigger is pressed.

Figure 10b shows the schematic for the three circuits and the connections to either the embedded computing hardware or the modified Nerf gun. Figure 10a shows a picture of the transmitter prototyping board used by the students.

4.3.4 Receiver Board

The receiver board is the analog interface between the photodetector and the ADC located on the embedded computing system. The laser tag system uses a red LED and a photodetector with a width of around $5mm$. At a distance of $60'$ the photodetector produces a current of approximately $20nA$. The photodetector current is converted into a voltage by placing a $3.3k\Omega$ resistor in series with the photodetector resulting in a voltage of $66\mu V$. The receiver circuit is a BJT transistor

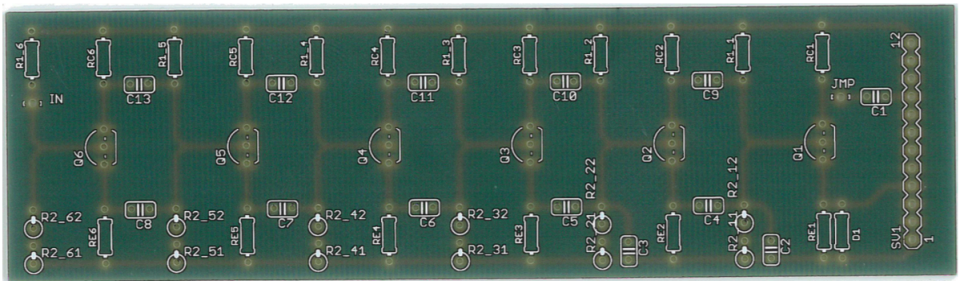


Figure 11: The printed circuit board for the receiver.

amplifier chain designed to have a gain of at least $1500V/V$. The design, fabrication, and testing of the high gain receiver chain is the main laboratory task for ECEn 340.

Figure 11 shows the printed circuit board (PCB) that is used by the students to fabricate the receiver amplifier chain. This PCB is designed to allow the students flexibility in their amplifier design. The PCB is designed to accommodate up to 6 BJT stages. Each stage is designed with voltage divider biasing. Each stage can be configured as a common-emitter amplifier by placing the coupling capacitor on the collector or as a common-collector amplifier by placing the coupling capacitor on the emitter.

In order to ensure proper operation of the receiver, it needs to be tested with a working laser tag system. Figure 12 shows how a laser tag gun is mounted to a post in the student laboratory. This test setup uses a gun that is continuously firing at the photodetector in a target gun located on the other side of the room (about 40 feet away). The "firing" gun is connected to a working embedded computing hardware system with attached LED board with the system programmed such that it fires continuously on a student-selected frequency (the photodetector is ignored for this gun). This working system is provided specifically so that students can test their analog circuitry. The target laser-tag gun is mounted at the other end of the room. This gun is connected to the student's receiver board which is then connected to an oscilloscope (only the photo-detector is used on target gun). The received signal is then displayed on the oscilloscope and verified to have a signal at the correct frequency with an amplitude of at least $20mV_{rms}$.

5 System Operation

The laser-tag system runs continuously and the various subsystems function as follows. A timer-generated interrupt retrieves data from the ADC (connected to the photo-detector circuitry) at 100 kHz and the 12-bit ADC values are placed in an ADC-buffer that maintains the last 100,000 samples, i.e., the buffer always contains all ADC data that have arrived in the last second. The timer interrupt also invokes all of the various state-machines (transmitter, trigger, lockoutTimer, hitLedTimer, see Figure 2). Thus data-sampling, detection of a trigger press, etc., are all controlled by the timer interrupt.

The detector also operates continuously, independent of the timer interrupt, essentially in a `while(1)` loop. Thus the detector and the software invoked by the timer interrupt can be seen as two major software modules effectively running in parallel; more specifically, the detector runs

Figure 12: Mounted laser tag system for testing.

continuously and is interrupted by the timer interrupt once each 10 microseconds (period of 100 kHz.) to store ADC samples in the ADC-buffer, to check for a trigger press, etc.

Each time the detector is invoked, it checks to see if there is data in the ADC buffer, if data exists, it is removed from the ADC buffer and is processed by the decimating FIR filter (the detector does nothing if no data is found in the ADC-buffer). As previously discussed, the data from the FIR filter are passed to the band-pass filters and so forth. Thus sampled photo-detector data are added to the ADC-buffer by the timer interrupt at a rate of 100 kHz; the detector, then, removes data from the ADC buffer when it is available and processes the data to detect potential hits.

To keep up with the incoming data, the detector must run somewhat faster than the rate at which data are added to the ADC-buffer. More specifically, each invocation of the detector needs to finish in less than 10 microseconds or much of the photo-detector values will simply pass through the ADC-buffer and never be processed by the detector. Using reasonable coding practice, the student-implemented detector code typically can complete a full cycle in much less than 10 microseconds thus ensuring that the overall system easily keeps up with incoming data. Note that using a *decimating* FIR filter dramatically reduces the computational requirements for the detector because it essentially reduces the detector computations by a factor of 10.

6 System Testing and Verification

Comprehensive testing and verification are essential to this project. The combination of analog circuitry that produces millivolt signals and complex DSP algorithms running continuously at 100 kHz results in a system that exhibits very counter-intuitive behavior when bugs arise. Each module of the overall system must be carefully tested to ensure that the overall system will work

properly when all modules are brought together (analog hardware and software). It is relatively straightforward for students to test and debug the laser-tag modules in isolation; it would be very difficult for even a seasoned expert to debug the complete laser-tag system without these module-level tests.

Comprehensive verification code is provided to students for each milestone and task. Though some may question whether the verification code should be instructor-provided or written by students, the instructors decided to provide comprehensive test software for the following reasons.

- The majority of the students are novice programmers. Only those students with significant out-of-class experience are capable of writing comprehensive module tests.
- Comprehensive test software often dwarfs the software-under-test in both size and complexity. For example, for any given module in the laser-tag system, the provided module-testing code typically contains 5-10 times as many lines of C code and is much more complex. Code that generates graphical output for visual verification is sometimes provided to plot the output from digital filters, etc., on the TFT display on the embedded computing system (see Figure 1). Expecting the typical inexperienced EE junior to write this kind of code is unreasonable.
- No matter how hard you try, English descriptions of software always contain ambiguity. Though the behavior of all of the modules is thoroughly described using English text in the milestone and task descriptions, the provided test-code augments the English description by demonstrating exactly how the code should behave in various corner cases, etc.

It is worth noting that students **do** write some of their own test code, specifically for simpler parts of the system. For example, students write test software for the the hitLedTimer, trigger, and lockoutTimer state machines. Moreover, when their code fails the provided tests, students usually write their own simpler test code that focuses on specific areas of the suspected software.

Specific Verification Examples

The first software module implemented by students is a specialized queue data structure that serves as a circular buffer. The queue is used throughout the project to implement the ADC-buffer, to serve as elastic buffers between stages of filtering software, to hold filter coefficients, etc. The student's implementation of the queue data structure typically consists of about 50 lines of C code; the provided test code consists of nearly 300 lines of C code and tests every function of the queue. Along with bug detection, the test code attempts to isolate bugs to specific functions where possible and to generate error and informational print messages that direct the student's attention to the function that most likely is causing a problem. Students will write their own simple test code to verify basic functionality and further locate bugs; however, once their queue code passes the provided tests, students can be confident that their code is bug free and that it meets specifications.

Comprehensively testing the students filtering code is more difficult. Test code is provided to ensure that the coefficients are properly aligned with inputs, etc., but this kind of test only verifies

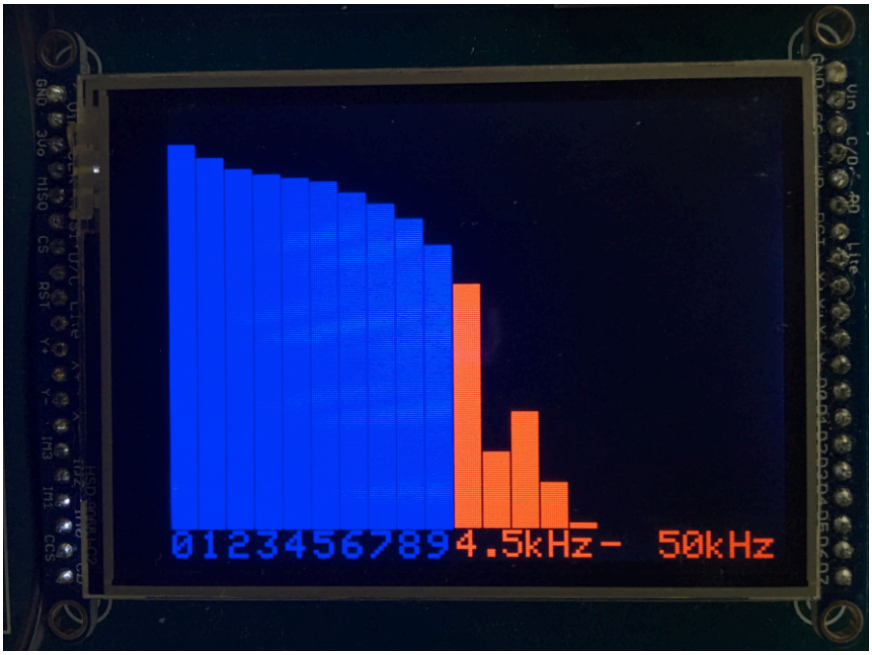


Figure 13: FIR Filter Response Plotted on the TFT Display

that the math is implemented correctly by the student's C code. Unfortunately, this code does not verify that the students actually designed the filters correctly in MATLAB in the first place, e.g., verifying that the filters actually filter out the correct parts of the incoming signal values.

To help students verify filter correctness, additional provided test code plots the actual frequency response of each filter on the provided TFT display so that students can compare the response of each of their filters against correctly-functioning filters. Figure 13 depicts the image of the plotted frequency response of a correct FIR filter as it appears on the TFT display. Note that graphical comparison works well here; the precise shape of the filter isn't that important as the filter response will vary somewhat from student to student. As long as the plotted filter meets the specifications, the filter is considered to be correct.

In the figure, the test software plots the frequency response of the student's FIR filter by generating synthetic square-wave data that sweeps through frequences for the 10 user frequencies and beyond, up through 50 kHz. Blue bars represent the 10 user frequencies, red bars represent non-user frequencies where the filter response should quickly roll off. Frequency plots like the one shown in the figure make it possible for students to verify that their filter coefficients are correct and that their filter code implements convolution correctly. The test software generates similar plots for the filter responses of all of the band-pass filters.

Final system verification proceeds in two stages. First, total system operation is performed without gun circuitry by connecting the transmitter to the detector via the jumper block as previously mentioned. Students are provided with operational code that implements *continuous-mode* and *shooter-mode*. Both displays use a histogram display; continuous-mode displays the power detected in each of the 10 user frequencies; shooter-mode displays the number of hits per user-frequency. Continuous-mode is a simpler test that verifies functionality up to but

Figure 14: Continuous Mode

Figure 15: Shooter Mode

not including power computation in the detector (see Figure 2); it also forces the transmitter subblock to continuously transmit at a selected frequency. Students test correct filter operation with continuous mode by viewing how the filters respond to each of the 10 frequencies (frequencies are selected by using the four slide switches on the Zybo board). Once the system passes this test, students move onto the shooter-mode test. Shooter-mode is a comprehensive test that verifies functionality for the entire detector. Students fire the transmitter by pressing a push-button on the Zybo board for each of the 10 frequencies and verify that the number of hits displayed on the histogram corresponds to the number of presses of the push-button for the selected frequency. Figure 14 is a screen-capture from a video demonstrating continuous-mode depicting that user-frequency 1 contains a power value of 200. Figure 15 is a screen-capture from a video demonstrating shooter-mode depicting that, thus far, user-frequency 0 has received 7 hits, user-frequency 4 has received 2 hits, and that user-frequency 9 has received 4 hits.

Knowing various run-time statistics about systems that run continuously can help debug systems with strange or substandard performance. Students can terminate operation of the system at any time by pressing one of the push-buttons on the embedded computing system (see Figure 1). As the system terminates operation, a series of statistics and, in some cases, warnings are printed on the TFT display on the embedded computing systems TFT display. These statistics note, at the time of termination, 1) how many values were remaining in the ADC buffer, 2) total run-time to this point, 3) cumulative run-time in the interrupt handler associated with the timer interrupt, and 4) total detector-invocation count to interrupt-count ratio. These values can be very helpful when helping students to debug obscure problems. For example, the number of values remaining in the ADC buffer (queue) must be very low at all times (typically 0 in a correctly-functioning system). Otherwise, this means that the detector is running too slowly to keep up with the incoming data. Figure 16 shows a photo of a statistics screen for a system that is not running properly: the element count in the ADC queue is too high and the detector-invocation-to-interrupt-ratio is too low. To ensure that the students don't miss this condition, the statistics screen also includes an error message in large red print when it detects that the detector is not running frequently enough.

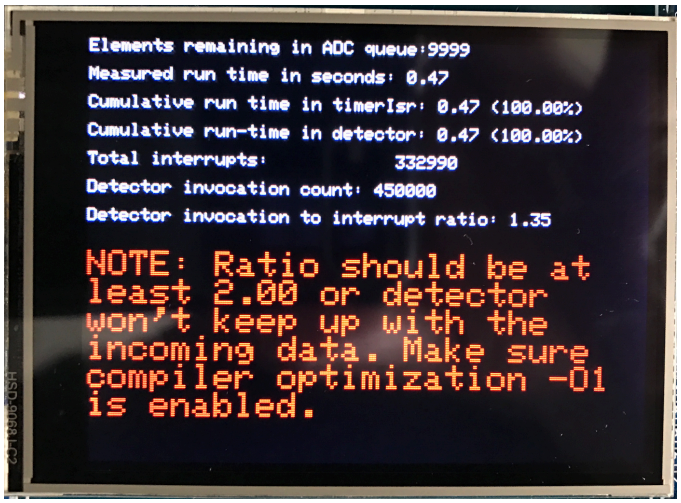


Figure 16: Run-Time Statistics Screen

7 Creative Projects to Enhance Student Engagement

Once the students have completed milestones 1-3, they have created and tested all of the necessary software and analog circuitry. Up to this point, all lab exercises are individual, i.e., each and every student must implement the entire milestone on their own. This overcomes a common complaint of students participating in group exercises: not all students in a group contribute equally[3]. This approach also ensures that all students have essentially the same learning experience. However, the last two milestones (4 and 5) are done in groups of 4-5 students. In Milestone 4, students simply replace the jumpers that connected the transmitter output to the detector input with cables that connect the gun circuitry (LED output and photo-detector with associated circuitry) and verify that everything works correctly. (Please review the earlier discussion from Section 4.3.2 regarding gun-cable connections to the jumper block.) Once students connect the guns to the embedded system board, the laser-tag system is usually completely functional. If bugs exist, they are usually caused by small problems related to the cabling that attaches the gun to the jumper block.

In Milestone 5 students are given 2-3 weeks to augment the laser-tag game with some feature of their own design. Once students complete this milestone, they create a youTube video that they share with the instructors. Examples of past creative projects include:

- a 3-D printed "grenade" with several embedded LEDs covering the surface of the grenade. The LEDs are controlled by a battery-operated Arduino processor that repeatedly fires the LEDs in all directions. Once the student rolls the grenade into the room, all players operating on a specific frequency are quickly eliminated. Figure 17 shows the CAD drawings of the grenade prior to fabrication.
- Enhanced sound effects that include "Star Wars" light-saber sound effects, etc. Though not mentioned previously, the embedded system board also contains an audio processor that students can program to generate sounds.

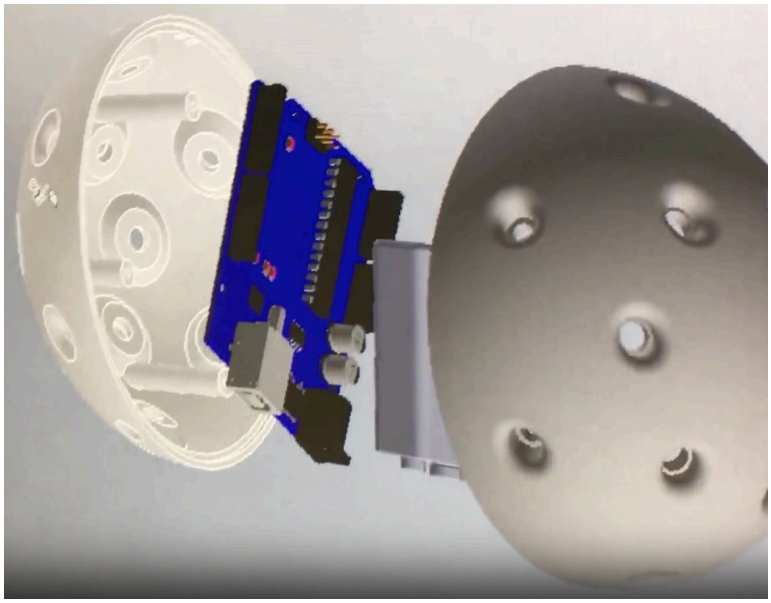


Figure 17: Creative Project: Laser Grenade

- Cell-phone-based controls and displays. Once paired with the Bluetooth modem, students controlled their game and displayed hit-counts by user on the phone screen and even had the phone vibrate in their pocket when hits were detected.

8 Assessment

Students were asked to answer the following questions at the end of the semester. Student responses are listed with the questions. There were a total of 52 students present for the survey.

1. Do you have more confidence in your abilities as a potential engineer? Do you feel more confident in your abilities to explore engineering problems and to build solutions? (78.8% responded yes).
2. Have you gained a better understanding of 380 concepts: convolution, decimation, FIR filters, bandpass filters, MatLab functions, etc? (77% responded yes).
3. Have you gained a better understanding of 340 concepts: building and debugging multistage amplifiers, BJTs, transistor biasing, etc? (34.6% responded yes).
4. Do you have better understanding of how complex embedded software can be organized and tested? (96% responded yes).
5. Did you have some fun along the way? (100% responded yes).

This rough assessment is preliminary; in the future a more rigourous survey will be designed and conducted, including individual anonymous surveys for each milestone. There were a few issues with a new instructor in the circuit design course (EC EN 340) that were the likely cause of lower

positive responses for Question 3. The high positive response rate for Question 4 is likely due to the close coupling between the project course and the previous introductory course on Embedded Systems programming (EC EN 330).

9 Conclusions

Surveys of students indicate that the laser-tag project is largely achieving its goals. Success rates are high, generally well in excess of 90%. Students state that the class has increased their confidence and also clarified their knowledge of concepts learned in prior courses. Most students feel that the class is also quite fun and they seem to enjoy playing with the laser-tag systems they have constructed. Faculty teaching and TA loads are reasonable for an immersive lab experience with close to 100 students: about 1.5 faculty loads and about 40 scheduled TA hours (this is administered as an open lab where students may work any time they like, 24-7). Experiences thus far indicate that this lab class can easily grow to 120-140 students without increasing faculty or TA loading.

One of the surprising outcomes arising from this project is the impact that it has on earlier courses. As instructors notice specific knowledge deficiencies when students struggle to complete certain milestones, they return to the course materials of the previous courses and improve these materials to improve student proficiency as necessary. This is surprising perhaps because this junior-core project is unique. In nearly all cases, once a student completes a typical course, there is no additional post-course evaluation mechanism to determine if the student truly achieved mastery during the course. However, with the junior-core project, it becomes evident very quickly if a large cohort of students have troubles with some concept from a prior class. Once the specific deficiency is determined, the corresponding course can be updated and improved for future students.

10 Acknowledgements

The instructors would like to thank the many students who helped to refine and improve the junior-project experience. In this paper, this project was presented as if it had been designed all at once and then presented to the students. In reality this has been a work in process for about three years. Although the initial course included a substantial amount of test software during its first offering, the amount and functionality of the test software were demonstrated to be insufficient as the course proceeded. The more comprehensive code that, for example, plots filter responses on the TFT screen, collects and presents run-time statistics, etc., was developed *during* the first offering when it was clear that students were having troubles with the signal-processing part of the course.

For example, the students would correctly implement convolution in their C code; however, many students had incorrectly computed or exported the filter coefficients in Matlab during the signal-processing milestone. Thus it became necessary to test the entire filter behavior on the embedded computing system. Once the filter responses were plotted on the TFT, students could quickly identify the problem, fix it, and move on.

Run-time statistics were handy in finding student detector-code that was running too slowly and that would fail to reliably detect hits. In many cases, increased compiler optimization or minor software modifications would sufficiently speed up the detector. On some occasions students would need to more carefully rewrite their code but these cases were (are) rare.

Finally, the instructors would also like to thank their department colleagues who have helped out and have supported this project.

11 Resources

11.1 Milestone Descriptions

The actual project milestones and subtasks with durations are provided below.

1. Milestone 1: Reverify analog circuitry and implement queue software (one week).
2. Milestone 2: Design decimating FIR-filter and IIR band-pass filters in Matlab (three weeks total).
   (a) Task 1: Design 10 bandpass filters.
   (b) Task 2: Design decimating system and analyze noise aliasing.
   (c) Task 3: Analyze/simulate full system (decimating FIR filter followed with 10 bandbass filters using Matlab.
3. Milestone 3: Implement all software in 'C' (six weeks total).
   (a) Task 1: Implement and test the FIR low-pass filters, IIR band-pass filters, and the power-computation routines. filters (two weeks).
   (b) Task 2: Implement software-based state machines for the transmitter, trigger, hit-LED timer, and the lockout timer (two weeks).
   (c) Task 3: Implement the rest of the detector and test the system using the jumper block to connect the transmitter and detector subsystems (two weeks).
4. Milestone 4: Connect the guns and test a complete system consisting of two working systems (one week).
5. Milestone 5: Customize the game with some new feature, final testing, and having fun with the laser-tag system (2.5 weeks).

11.2 Links to Online Content

All class content is freely available online. Links to the class materials, youTube channels, student-created videos, are provided below. (Note to reviewers: these links were blinded to maintain the anonymity of the authors during the review process). If accepted, links to all online materials will appear in the final paper.

- Junior-Core Class Web Pages:
  `http://ece390web.groups.et.byu.net/dokuwiki/doku.php?id=start`
- Embedded Programming Course Web Pages:
  `http://ecen330wiki.groups.et.byu.net/wiki/doku.php`
- Junior-Core-Project youTube Channel:
  `https://www.youtube.com/channel/UCoDVeJylUjZCQW9TZ3dNS8w`
- Embedded Programming Class youTube Channel:
  `https://www.youtube.com/channel/UC2DAqectVGE6DIT-b8N2dJg/playlists`
- Laser-Grenade Student Video:
  `https://youtu.be/pvLbELZw1Hc`
- Web-Page Containing All Student-Created Videos from 2016:
  `http://ece390web.groups.et.byu.net/dokuwiki/doku.php?id=groups_2016`

References

[1] Marilyn M Lombardi. Authentic learning for the 21st century: An overview. *Educause learning initiative*, 1 (2007):1–12, 2007.

[2] James D Lang, Susan Cruse, Francis D McVey, and John McMasters. Industry expectations of new engineers: A survey to assist curriculum designers. *Journal of Engineering Education*, 88(1):43, 1999.

[3] Julie E Mills, David F Treagust, et al. Engineering education—is problem-based or project-based learning the answer. *Australasian journal of engineering education*, 3(2):2–16, 2003.