# Final Report

**Group 8 Track Independent Train Project**
**ECE 4534 Spring 2010**
**Nina Huffman, Mark Bumiller, Struan Clark and Jimmy Cox**

## Overview

Our task was to create a system that is capable of identifying the engine and freight cars in a passing train as well as "mapping" a track. Our system consists of RFID tags that are read when the train passes the station and an accelerometer on the train that sends its data via XBee. This data is gathered and processed in the S3E and displayed on the web server as intelligible directions that describe the position of the train on a track.

## Analysis of Design

Our design had several very effective features and only a couple areas that could have used improvement. Our method of identifying individual cars of a train using a RFID reader and tags was a very effective solution to the proposed problem. The tags in the form of cards which we attached to the cars each had individual identification codes and were put to use exactly as they were intended. The use of this existing technology made it possible for us to spend time with other areas of our design rather than devising a system of identifying cars using color or size. Our wireless communication between our on train sensor and the remainder of the system was also very effective. Using the XBee transceivers was simple and effective. Attaching the modules to a PIC was straightforward because of the widely used UART standards that both are designed to implement.

The main aspect of our design that could have been improved was our choice of using an accelerometer as our onboard sensor. The accelerometer detecting bumps is probably the best choice, however the turns that the train took did not generate very large g-forces. It was difficult to distinguish when the train was going around a corner compared to when it was going straight.

# Analysis of Implementation

All major components of our projects were successfully implemented. We were able to identify train cars with complete accuracy through an RFID reader board and RFID cards. In addition to this, we were able to semi-successfully map the track using our wireless accelerometer board and XBee receive board. Our Spartan code was able to receive and analyze data from these devices, as well as a v4 board connected to Loconet.

The RFID board worked completely. It was designed to store the last card scanned and transmit this information upon an $I^2C$ request. This was verified by debug lights on the board itself, and the correct card information being transmitted via $I^2C$ to the Spartan.

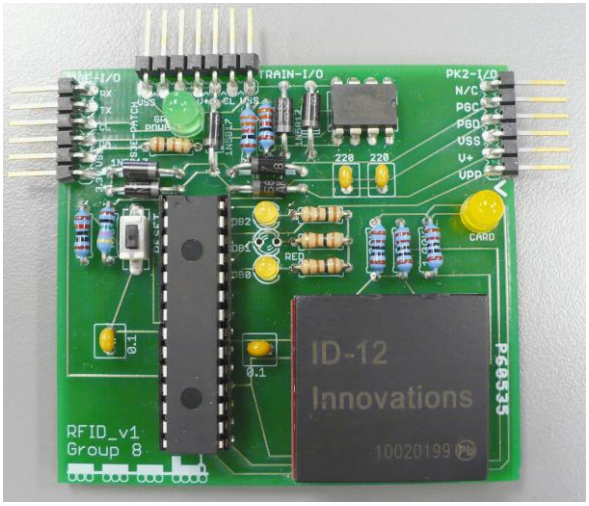


Figure 1: RFID Board

The accelerometer board implementation worked well enough for our project. It was designed to convert values from the ADC and transmit them via UART over a wireless link to the XBee receive board. Upon its construction we became aware of a lot of noise in the analog signals being read by the ADC on the PIC processor. This was due to the capacitor in the PICs ADC circuitry's inability to be driven by the low amount of current emanating from the accelerometer. We solved this problem with a compromise; for our Y channel which was being used to measure turns we implemented a low-pass filter, at the expense of a longer slew rate. This was necessary as we needed a very accurate value, however did not need it change very quickly to detect corners. For our Z axis we simply read the noisy raw accelerometer data. Since this channel was only needed for bumps this was the best solution as we needed instantaneous response to accurately detect the bumps.

The XBee receive board worked correctly. It was designed to receive UART data from a wireless link and transmit it via $I^2C$. This was verified with a UART activity debug light on the board and the correct accelerometer data being transmitted via $I^2C$ to the Spartan.

The Loconet PIC worked correctly. It was programmed to receive Loconet data and transmit it via $I^2C$. This was verified with Loconet status debug lights on the board itself, and the correct Loconet data being transmitted via $I^2C$ to the Spartan.

The Spartan code worked correctly for the most part. It was designed to receive $I^2C$ data on a timer interrupt, analyze and interpret this data, and display it on a web server. The card data

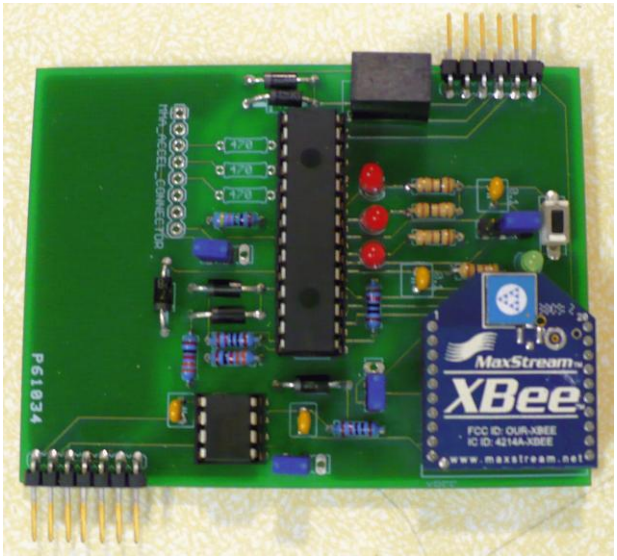


Figure 2: XBee Receive Board

and Loconet data were correctly analyzed and displayed. The accelerometer data was also analyzed, and the results were fairly accurate but not perfect. The algorithm used to analyze the turn status of the accelerometer was based on static constraints, however there was no way to ensure these constraints would be accurate for all turns as even a slight inconsistency in the accelerometer's placement on the train car, or tilt of the car itself could throw this off. It would have been better to use a calibration algorithm that would base the definitions of a right or left turn off of the initial accelerometer state, as opposed to a hardcoded value. We verified the Spartan code was working by the activity of the PIC devices that required its polling and by its output which we displayed on a web server.

# Lessons Learned

Throughout the project, the team learned when to use an oscilloscope. At first, the group was reluctant to use the oscilloscope because members were unsure of how to properly use it. The scope proved useful in reading $I^2C$ and UART messages to see exactly what was being read or sent from the PIC when LEDs were not enough. Analyzing analog signals from the accelerometer helped recognize the noise issue when connected to the PIC and design a low-pass filter to help with the issue. The oscilloscope was the most underutilized tool at the group's disposal and problems could have been averted if it was used more during the initial design and testing stages.

When working with the Spartan code, we initially started with code from a similar past project. This became quite a problem as the project progressed. The code seemed to be done in a rush and the group had trouble understanding how the code worked. As more features were implemented, it became even harder to integrate into the existing code. The group should have worked from the baseline in order to understand the code and then add on in a manner that best suited the project.

# Tangible Achievements

The example code provided by our professor helped us comprehend much of the coding aspects of the project. We ran the sample Test Project and analyzed the $I^2C$ code in order to implement $I^2C$ in our own project as we saw fit. Our PIC code successfully packages the data from the sensors into messages and passes the messages via $I^2C$ to the S3E. We developed our Spartan code to analyze the messages to determine where turns and bumps occur. Based on the Test Project, we developed code to pass turnout information from the DCS-50 to the Spartan board via Loconet. The web server was developed off of the Test Project in addition to methods of refreshing and iframe manipulation found through research. In addition to the software components, our achievements are plentiful on the hardware side, as well. All three of our boards' designs were based off of the PIC V4 board. The unique components that our project uses allowed us to develop most of the board according to the specifications from the datasheets of the individual components (RFID, XBee, and Accelerometer). We were successfully able to design and solder these boards such that they operate correctly during testing. All aspects of our project were successfully integrated together, which is generally one of the more difficult aspects of a project like this. We were able to successfully design, build and implement our system from the ground up. We researched our initial ideas on how to tackle this project, and revised our plans until we came to a reasonable design. After a lot of hard work, we were able to put together a functioning system with all components working as we indented.

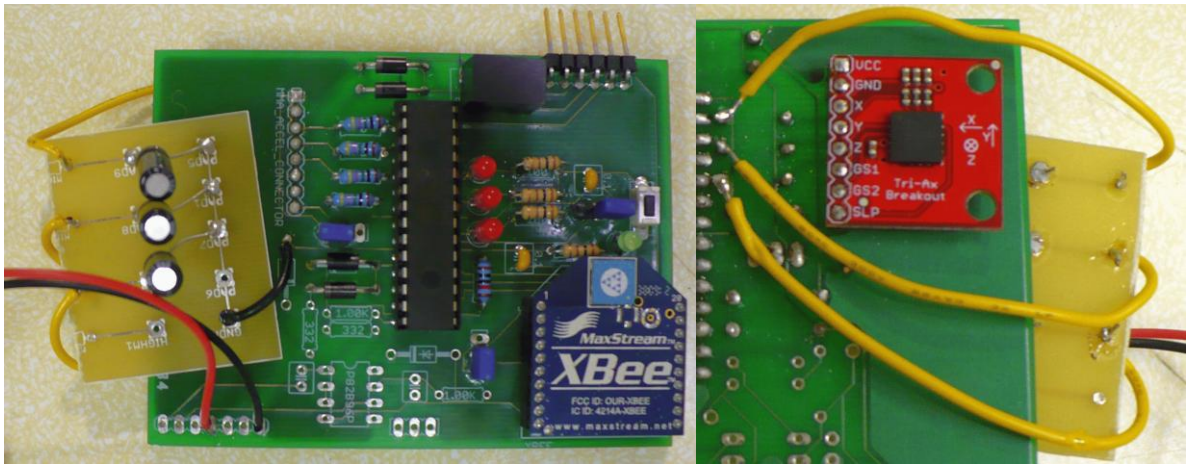


**Figure 3: Accelerometer Board (Top and Bottom Views)**

# Technical Skills Learned

We learned several important skills in developing this project. On the hardware side we learned how to select specific components to accomplish a task within a given budget. We were required to compare multiple options for completing our assignment and pick what we believed to be the best. After purchasing parts, we were then required to prototype, and finally develop a fabricated board. These tasks gave us a lot of experience with the PCB development process. All considerations such as board size, matching parts to suitable footprints on a board, and figuring out how to power everything properly was a major learning experience.

Specifically with our hardware, we learned a great deal about interfacing PIC processors via $I^2C$ and UART. Learning standards for interfacing various components is important for integration of many varied devices. We also learned some details about using XBee radios, and that they are cheap and reliable. We learned that RFID is an effective way to identify discrete objects reliably and easily, if a little slow. An important piece of hardware knowledge we learned was the variability of accelerometer data. We now know that it is necessary to have a proper calibration to get very accurate data from an accelerometer.

On the software side we learned how to write efficient C code that will run on a low power processor such as a PIC processor. We also learned the proper procedure for writing interrupt driven code on both the Spartan and PIC code. Most importantly we learned how to implement multiple threads on the Spartan's processor without overloading the resources of the real time operating system. These skills are important to reduce time consuming errors resulting from incorrect coding practice.

# An Analysis of What Didn't Work

Until the integration stage of the project, noise from the accelerometer was never noticed. Scoping the accelerometer alone produced a clean signal and data read from the ADC seemed clean. When reading the accelerometer values from the Spartan, it was realized that the values were noisy. After checking all the code, the accelerometer was scoped while hooked up to the PIC that was reading values. When the PIC was present, even not powered, the accelerometer would have a noisy signal, but a clean signal was present when the PIC was removed. As a quick fix, capacitors were tied to the input lines to create a low-pass filter. This was adequate for turning and speed changes, but not suitable for detecting bumps. The z-axis was not filtered so that bumps could still be detected.

The reason for this problem is the PICs ADC circuitry. The ADC has a 120pF capacitor that holds the voltage while being read. The accelerometer was unable to fully charge the capacitor while

being sampled, resulting in a noisy signal. This could have been solved by using op-amps to isolate the accelerometer outputs from the ADC inputs instead of low-pass filters.

To detect bumps and turns, tolerances were used. Noise from traveling on a track and from those previously mentioned gave inconsistent measurements for these occurrences. To compensate for noise, a more sophisticated algorithm is needed. One possible solution is a Kalman filter.

The Kalman algorithm is regularly used to filter out noise in a signal. The algorithm predicts the next value and updates the current value accordingly. As time progresses, measurements become more accurate. To be implemented in this project, the algorithm would assume a constant voltage reading and adjust the measurement accordingly. It would take time for turns to be detected, but false turns could be removed.

# What We Would Do Differently

We have two approaches that may have worked to correct our design limitations in regards to our accelerometer. The first involves using an algorithm that will dynamically adjust the cut off or threshold values that distinguish a turn from a straight away. A second solution would be to use a compass instead of an accelerometer. The compass would have a much easier time determining the direction the train is heading and would not need a complicated algorithm to determine when the train is turning. The complication would be the electro-magnetic field produced by the power running through the tracks. We don't foresee this being a serious problem because we do not actually need to know what direction the train travels just its relative direction. Also because this field will be constant around the track we would be able to calibrate our compass to compensate for the field.