PIC32 DEVELOPMENT

A Design Project Report

Presented to the School of Electrical and Computer Engineering of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering, Electrical and Computer Engineering

Submitted by

Alexander Keith Whiteway

MEng Field Advisor: Bruce Land

Degree Date: May, 2015

Abstract

Master of Engineering Program
School of Electrical and Computer Engineering
Cornell University
Design Project Report

Project Title: PIC32 Development

Author: Alex Whiteway

Abstract:

The PIC32 Development project aims to bring the new technological breakthroughs of 32-bit microcontrollers into the hands of Cornell students. With 32-bit processing (as opposed to the 8-bit microcontrollers used in courses today), learning incredible and advanced peripherals such as USB, Ethernet, and high definition graphics (among others) becomes realizable. The goal of the project is to learn how these peripherals work and how to implement them on such an advanced platform. Though Microchip (PIC32 manufacturer) provides a TCP/IP stack, USB stack, etc. as well as example code, they are incredibly complex, confusing, and error prone. Throughout this project, these resources will be combined to make simple APIs for students to use, vastly decreasing complexity and development time for laboratory exercises and projects. Once the firmware is stable, custom circuitry will be designed and manufactured with everything needed to use these peripherals so that students can focus on the bigger picture and not on details such as choosing the right magnetics for Ethernet. Upon completion of the project, future students will have: A custom circuit board built specifically for laboratory assignments, a clean and simple API for using advanced peripherals (along with example code), and thorough (but not cumbersome) user manuals on how to use all aspects of the board and firmware. These deliverables will expose Cornell students to some of the most advanced microcontrollers on the market today, replacing the decades old technology that is currently used in the classroom.

Executive Summary

The PIC32 Development project was created to provide senior level students at Cornell a more thorough and technologically advanced introduction to microcontrollers and embedded systems in general. In addition, it is possible that the products delivered for this project could be used in other courses or for hobby projects, if the students chose to use them in such a way.

The full system was constructed to deliver advanced functionality without a steep learning curve. In addition, it was built to be a modular system capable of quick reconfiguration with minimal effort. This allows it to retain the original goal advanced yet simple, while also allowing students to use it in a variety of configurations depending on the target design.

Three modular hardware pieces were developed: the main board, a communications module, and a basic input/output board. These three modules each fit in a standard solderless breadboard [1]. Since these boards are designed to fit in a breadboard, it only takes four to six wire connections to add a module (with no external components). The main board supports approximately 40 input/output pins for the student to utilize, USB OTG functionality, power regulators, and a programming circuit. The basic input and output board gives the user access to switches, LEDs, and seven segment displays. Finally, the communications board provides an Ethernet connector with all required external hardware on the circuit board.

It is difficult to say how expensive the final product actually costs. Since it comprises three printed circuit boards, costs can range from $5 to $75 per board, depending on the quantity purchased. Since this project is meant to be used in a somewhat large class, the prices should be closer to the lower end of the spectrum, making it an affordable teaching tool.

Contents

Figures

Introduction

Background

It is no secret that technology is advancing at an incredible rate. New and updated devices are coming out faster than ever, making it difficult for the average consumer to keep up to date. Unfortunately, this leaves an educational gap for students in Electrical and Computer Engineering. How are students supposed to learn everything they need to learn while course curriculum is constantly becoming outdated?

Invented in the early 1970s [2], microcontrollers quickly saw uses in calculators and other basic devices. These tended to be either 4 or 8-bit microcontrollers since the technology of the day was not able to produce anything more advanced (at least at a reasonable cost). From that point, technology began to increase rapidly, expanding the microcontroller market to over USD 16.7B [3]. Already by 2013, the market leader was 32-bit controllers. They are also forecast to lead industry growth through 2020, where the industry is expected to be worth USD 27B, or an astounding 62% growth in just seven years.

The PIC32 Development project aims to enhance ECE 4760 at Cornell University by utilizing these industry leading microcontrollers. This lab-based course introduces students to microcontrollers and their applications. Microcontrollers are at the heart of countless systems used every day. Many electronics that do not need a full CPU such as stereos, microwaves, GPS devices, etc. utilize microcontrollers as the heart of the system[4]. The problem is the microcontrollers used in ECE 4760 are no longer cutting edge. Although they have been improved over the years, these 8-bit microcontrollers have been around for many years[4].

This project will replace the current lab curriculum with exciting lab ideas based on a new line of microcontrollers, ones that are only a few years old. They will bring cutting edge technology to the lab, including 100Base-TX Ethernet, full-speed USB, full color LCD screens, and many more. Bringing these exciting technologies to the classroom will be crucial in exposing students to current technologies, creating more competitive graduates than ever before.

Issues

The biggest issue with updating the lab curriculum to utilize the PIC32 microcontrollers is complexity. TCP/IP and USB are incredibly advanced protocols, which can make implementing them an entire project in themselves. Microchip (PIC32 Manufacturer) has released full TCP/IP and USB stacks (driver source code), but they are each thousands upon thousands of lines of code and are prone to error.

At this point, getting them to work would be too much for students to handle for one or two week labs. Addressing these issues was one of the most crucial parts of the PIC32 Development project.

Another problem with using such advanced technology is the physical chips are usually so complex they cannot easily fit in prototyping circuit boards called breadboards or whiteboards. These boards make creating, debugging, and correcting simple circuits quite easy, especially in lab settings. Since some of the components cannot easily connect to these prototyping boards, it generally makes using them in class difficult. This is where a professionally made circuit board will come in. One will have to be designed, tested, debugged, and manufactured to allow students to easily use the components. This can be a months-long project, not one that is well suited for ECE 4760, and outside the scope of the class.

The last major foreseen issue is documentation. The reference manual and datasheets for the PIC32 family of microcontrollers are thousands of pages long. Much of this is absolutely crucial information, while a large portion of it is beyond what the students need to know for class or completely irrelevant. Condensing this documentation into a clear and concise reference that only contains the necessary information for the students is a key goal for the PIC32 Development project.

Existing Products

Many products are on the market that are meant to provide simple introductions to advanced microcontrollers. The products that are on the market, however, tend to be costly and specialized for one use, not general uses that are desirable for a teaching environment. Microchip offers some of these development kits for their PIC32 line of products [5], such as:

- Microstick II - $34.95
- MPLAB Starter Kit - $109.99
- Multimedia Expansion Board - $249.99
- Ethernet Starter Kit - $89.00
- USB Starter Kit - $59.99
- PIC32MX Starter Kit - $69.99

Though well constructed, each of these kits is rather expensive, especially for students. A full set of boards developed for this project will cost less than many of these individual kits.

Design Problem

This project aims to produce a range of circuit boards that students can use for a variety of lab exercises. In addition, they will be designed with flexibility in mind so students are not limited to specific microcontroller functions, but will be able to explore the entire range of microcontroller capabilities. These boards will completely replace the current teaching tools, so they will need to simple to use for both students and instructor.

Since students may enter the class without ever having seen a microcontroller, this must be kept in mind. On the other hand, some students may enter the class with existing microcontroller experience. Due to the unknown strengths of the students, the system should be such that a complete beginner or an advanced user can use the system. This limits many possible approaches, but is still a realizable constraint.

The goal of this project is, as earlier stated, to provide an easy-to-use system to introduce students to the world of microcontrollers. In addition, manuals and supplemental materials for features such those listed in Appendix A: PIC32 Features will be provided. This will allow students to actually use the feature instead of spending days or weeks attempting to get it working.

Design Requirements

Since it is being used as a teaching tool, the system will need to be easily usable to students with varied experience and skill levels. A baseline list of requirements are:

- Create custom PCBs that will be soldered by the students
- Use as few surface mount parts as possible
- Easy to use while still allowing flexibility and access to advanced features
- Ability to use the system for basic projects with little or no external components
- An interface to custom hardware must be available
- Provide device drivers for peripherals such as Ethernet and USB
- Must be more affordable than existing development systems

The project was meant to be exploratory and open ended, so the requirements evolved slightly throughout the design process. The list above represents the final requirements to the project. Due to the limited time to produce the desired results, future improvements may impact this list.

Range of Solutions

From the beginning, the PIC32 Development project was meant to be an open-ended and exploratory project. Because of this, a huge range of solutions was possible. Some of these solutions are outlines below, though more solutions may have been possible.

Software Only

One of the possibilities for this project was to provide example software that students could use to implement or enhance their laboratory exercises. This solution would have required extensive example code for a large range of advanced peripherals on the microcontroller. Examples of these features are in Appendix A: PIC32 Features.

If this route were to be selected, it would have been expected that code examples would have been provided. In addition, many of these features may have needed basic circuits to demonstrate the feature, such as a potentiometer to demonstrate the ADC capabilities. Though these circuits would have been basic, they would be necessary to effectively use the selected feature.

Basic Circuit Examples

Like the software only approach, some features would have been explored, though this time with more emphasis on hardware design. Many of the features need external hardware to be fully demonstrated, such as I2C and SPI since these features are meant to interact with external circuitry.

This option would have required both hardware and software examples as a deliverable. The software would be basic examples on using the accompanying hardware. The hardware would have been schematic designs so that students were able to recreate the design.

Custom Hardware

This solution requires both hardware and software components with the addition of custom circuitry. This approach would not require students to recreate circuits, however, but to solder components onto the board to build the circuits. Software would be developed to accompany the specific hardware.

The custom hardware would need to be robust enough for students to make mistakes without destroying the board. In addition, the board would need to be designed with expansion in mind so students are not pigeon-holed into using only the features on the board. Some sort of expansion header would be necessary to give full access to the microcontroller.

Multiple options existed for designing the boards. The possible options included:

- One single board with all components
- Multiple stackable boards
- Multiple modular boards

Each method had its own advantages and disadvantages, but creating multiple modular boards provided the simplicity and flexibility required for the design project.

Design Process

Initial Design

The design process began with experimenting with the PIC32 and determining if the advanced features (such as USB and Ethernet) were tractable for student usage. Microchip provides code for both USB and Ethernet, so that was a logical place to start experimenting. All of the experimenting was based around the Microstick II development board (see Figure 1 - Microstick II Development BoardFigure 1below), which included an onboard debugger and USB power supply.

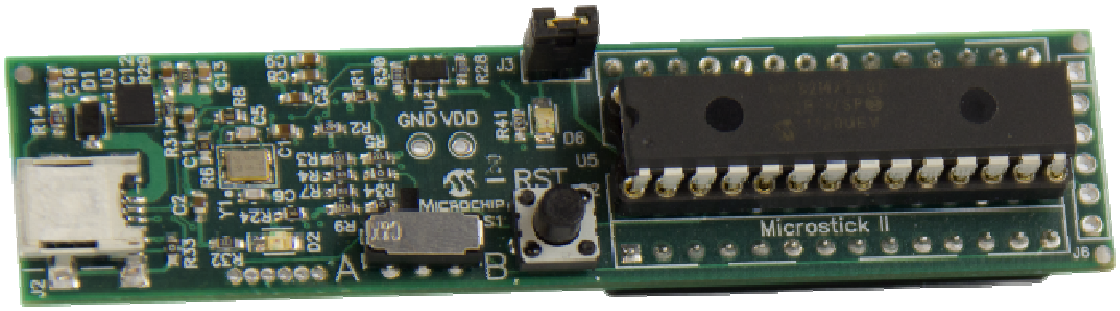

Figure 1 - Microstick II Development Board

Ethernet

Implementing the Microchip Ethernet driver was the first step in the experimentation process. The code was supposed to compile and execute without modification, but that was definitely not the case. It seemed like the code was meant to be general for hundreds of different microcontrollers, but unfortunately was not correct for the PIC32MX250F128B. File by file, function by function, the debugging process began. Hours of tedious debugging started to expose some issues with the code, including the possibility of null pointers causing the microcontroller to crash and reset. At least four or five such bugs were discovered and corrected. Though there may be more, no more have been discovered.

Once the microcontroller finally recognized the Ethernet cable was connected, the most basic functionality to test was to return a ping request from an external host. There were more issues with this code as well, most of which involved adjusting buffer sizes and some constant values that were set in the code. The microcontroller was finally able to return a ping request, but again would crash after a couple of minutes of constant requests. More debugging exposed some poor memory management issues that were soon corrected.

It was finally time to attempt more advanced functions. It took a long time to dig through the code and find the actual interfaces, but once they were located it became a simpler problem. The code seemed to produce an API similar to C Berkeley sockets and conformed to IEEE Std 1003.1 [6]. Unfortunately, the functions and other support information was sporadically spread throughout the driver. To make them usable, they were merged into a single file.

The final step was to test the code under different circumstances. During this phase, the microcontroller was able to load a webpage, host SSH and telnet sessions, and host a webpage as well. These successes provided the confidence needed to pursue Ethernet as part of the design project.

Audio

Though not a complex feature, providing a way to play audio seemed to be an important function in the project. Whether playing music or sound effects, audio can add to many different projects.

Simply playing a sound seemed too basic for this project, but combining audio with the Ethernet code seemed to be a more interesting solution. Not only did this exercise the Ethernet code, it provided confidence in using timer interrupts to create consistent and accurate time intervals down to the single clock cycle level. Combining these features, 16-bit, 44.1khz was successfully streamed over a network to the microcontroller and played over speakers in real time.

USB

USB proved to be the most difficult feature to implement. Like Ethernet, Microchip provided a USB library to use. Unfortunately, it too was filled with errors that didn't allow the code to immediately compile. A couple of errors were tracked down, but extensive debugging showed the microcontroller was not detecting when a USB device was plugged in.

The first step in solving this problem was to determine if the USB module was turned on and operating correctly. Using the debugger, it was clear that the USB module was at least being instructed to turn on. Searching through forums online narrowed the possible sources of error with the most likely situation being a bad oscillator configuration. This seemed to be a likely case since the USB module runs off a separate clock network. According to the datasheet, the USB module could run off of the internal FRC oscillator with specific PLL settings. These seemed to be set correctly, but the default clock source was set to an external oscillator.

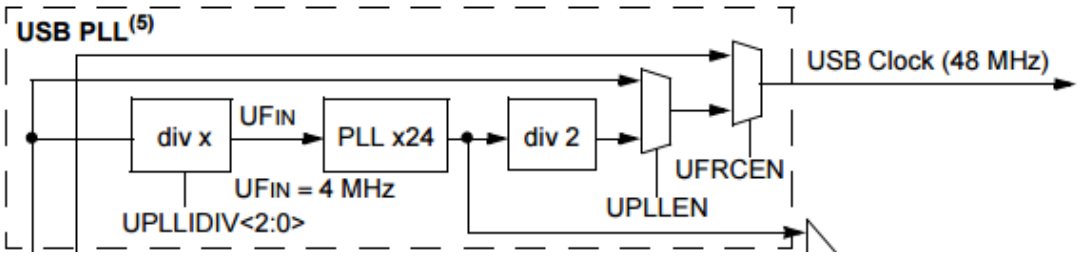

Figure 2 - USB PLL module diagram

Overwriting the default value, the clock multiplexer should have provided the USB module with the correct clock signal. The debugger showed that the module was now powering on, but still not detecting a USB device. Frustrations led to experimenting with an external oscillator instead of using the internal one. Testing showed the USB module detected a device right away using the eternal oscillator. Extensive searches exposed a small footnote in another datasheet explaining the USB module could only be turned on with the internal oscillator, but could not be used without the external oscillator.

Once the module was working, the code was modified to read a keyboard. Like Ethernet, the code to read the keyboard was buried deep in the driver. It too was moved to a new, easily accessible file so students could easily use it in their projects. Now all they must do is include the function and it will be called each time a key is detected. The depressed key is passed to this function as an argument along with any modifiers (shift, ctrl, etc). The student is then free to use the information as they desire. Similar functions exist for mice and joysticks as well.

LCD Screen

Many projects can benefit from having a full color graphical LCD screen included. The chosen LCD screen uses either SPI or a parallel interface to write the data. Though the parallel interface allows for faster data rates, using SPI allows the screen to go on the same communications bus as the Ethernet chip, saving microcontroller pins for other functions. Professor Bruce Land had some spare 2.2” 18-bit color LCD screens [7] (see Figure 3 below), so that screen was chosen for testing purposes. The screen uses a common driver, the ILI9340. Using a screen with such a common driver means the code can be used to drive a wide variety of screens, varying in both resolution and size.

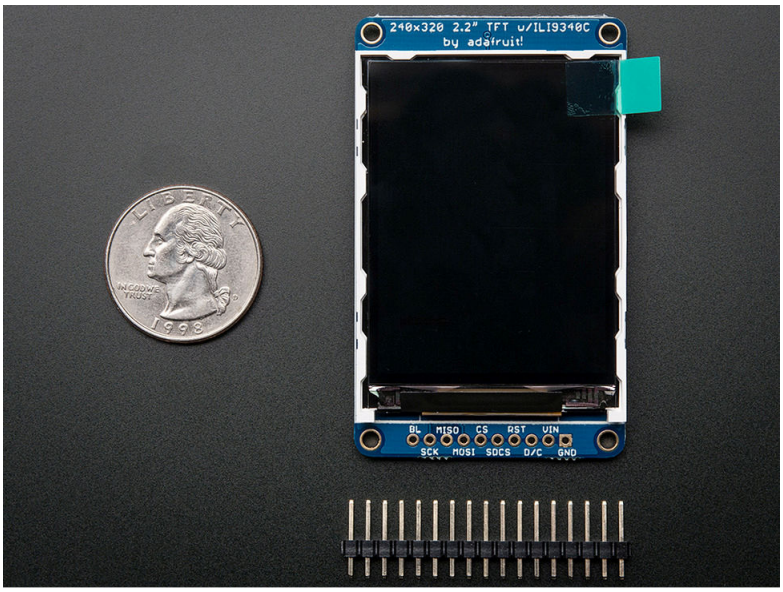

Figure 3 - 2.2" LCD screen from Adafruit

The screen comes with some basic driver, but unfortunately it was limited in scope and set for an 8-bit Arduino microcontroller. This code was used as a starting point, but heavy modifications were required to get it to run properly and efficiently on the PIC32. The code was first adjusted to use 16 or 32-bit SPI. Since the screen reads 16-bit color over the SPI interface, using either one of these data widths sufficed. Using 16-bit SPI allowed the screen to boot and show colors as desired.

The next task was to implement drawing functions for common usages such as lines, rectangles, and text. Much of the code was adapted from the Arduino library, but simple modifications made the code more efficient and usable. Functions were also added to for more advanced drawing functions, including drawing bitmaps on the screen.

SPI

SPI was necessary to communicate with many of the features explained above, so it was clear SPI was easy to use and functioned as desired. Though the Ethernet code did most of the SPI setup it needed, the LCD screen code did not. The SPI documentation for the PIC32 [8] is quite clear and thorough, however, so using the SPI module was a simple task.

Other Features

Other features were tested during the initial phases of the project, they were either too simple or too complex to be considered part of the project. This by no means limits the student from exploring these peripherals, it was just not the best use of time for this design project.

Hardware Design

Successes in the initial design process made it clear that using the PIC32 for teaching was a realistic expectation. This meant hardware needed to be developed to support some of the advanced features. The main question was how to divide the hardware. The obvious options were to put all of the hardware on one circuit board or to make modular units. The latter option was chosen for simplicity, flexibility, and future expandability. Multiple iterations settled on three boards: the main board, the basic input and output board, and the Ethernet board.

Designing these boards could be broken down into two distinct steps: schematics and circuit board layouts. As stated above, three circuit boards were chosen to be produced for the final design. Each one of these required both schematics and layouts. These schematics and board layouts can be seen in Appendix B: Schematics and Appendix C: Board Layouts.

Creating the schematics for the boards came in a few basic steps. The first step was to gather the hardware requirements for each board. Next, each datasheet was searched to determine what external components were required, if any. Next, the circuits were constructed on a solderless breadboard to test any complex circuits, such as the Ethernet MAC chip. A combination of datasheets and online help were used to develop these initial designs. Once the designs were confirmed, they were designed on the computer.

After the schematics were checked, the schematics were sent to circuit board layout software. The general strategy was to route the high speed data lines first. This gives them the most direct routes possible, minimizing interference, impedance, and capacitance on the lines, allowing them to run at full speed. Next, less important data lines were routed, such as chip select lines. Finally, power was supplied to all of the necessary components on the board.

Main Board

The main board was designed to be the heart of the system. This board was to feature:

- PIC32MX250F128B

- Port Expander
- Programming Circuitry
- USB
- Power Regulation

The design was made incrementally, starting with the microcontroller and programming circuitry combination. The recommended connections were made, including bypass capacitors and power/ground connections. Microcontroller operation was verified by programming it and flashing an LED. This success meant more complex features could be added.

The next circuit to be added was the port expander. This only required a couple of external components, mainly bypass capacitors and current setting resistors. Once these components were added, the necessary SPI connections were completed to finish the circuit. The port expander took longer to debug than originally thought, mainly due to some incorrect register settings. The default settings disable the inputs and outputs until the user is ready to enable them, but that was unclear at first. Once the issues were corrected, the port expander worked as corrected.

This concluded the features that would be prototyped before designing the board. The USB connector only required a direct connection to the microcontroller, so prototyping it was unnecessary. Additionally, prior experience with power regulation made it, too, unnecessary to prototype.

Each part was systematically added to the digital schematic one at a time. The external components for each part were connected to the respective part and then each was thoroughly checked before interconnecting the parts. The connections were each checked multiple times before determining the schematic was complete.

Designing the PCB started by placing the largest parts as close as possible to make the circuit board as compact as possible. This resulted in the two IC chips being placed vertically and parallel to each other. The board was sized around these parts to make it as narrow as practical. Once these parts were placed, the location-critical parts were placed next. These parts consisted of the headers to connect the board to the breadboard and the USB oscillator being as close to the microcontroller as possible. Finally, the remaining components were placed as logically as possible.

The final step was to route the traces around the board. The high speed data connections (SPI and USB data) were routed first to provide them as little impedance and capacitance as possible. The next lines that were routed were the other data lines that may be used in the future: the microcontroller and port expander I/O pins. Finally, power and any remaining pins were routed, leading to a compact circuit board.

Basic Input and Output Board

The basic input and output board was created to make using switches and LEDs easy while still providing a large number of them to the students. This board would have the following features:

- LEDs
- Switches
- Seven segment displays
- Seven segment display driver

Fortunately, designing this schematic was much simpler than the main board. Part of this was due to prior experience with seven segment displays. Additionally, the port expander that would drive the LEDs and read the switches was already prototyped for the main circuit board, so no additional work was necessary.

With this board, it was most important to orient the seven segment displays correctly. Besides this, 16 LEDs had to be arranged. Since aligning the 16 LEDs in a single row would make the board two wide, they were place in two rows. The ICs were then oriented vertically next to the arranged displays and LEDs. This made the board only slightly larger, but it was the best orientation option. Finally, the switches were placed in between the individual LEDs and the displays. This was done without making the board any larger since the vertical IC orientation made the board tall enough to place the switches.

Communications Board

The communications board contained the most complex circuitry for the project. Luckily, the datasheet for the MAC layer IC had thorough instructions on all external components. Using an Ethernet jack that had built-in magnetic hardware made the circuitry much simpler, though.

It was unclear whether the chip would work on a breadboard, which was unnerving at first. The oscillator must run at 25.000 MHz, while data between the Ethernet jack and the IC may run even faster than that. This is well beyond the 10 MHz frequency limit that many people assume [9]. Somehow the oscillator did manage to work at 25 MHz, and the circuit worked nearly immediately when following the design recommendations.

Laying out the circuit board followed many of the patterns used in the other boards. The board size was set to be just large enough for the IC and Ethernet connector, with all external components being placed underneath these two parts. The data lines from the connector to the IC were routed first, along with the oscillator lines. This guaranteed the shortest routes between these points. Once these were run, the

same procedure as the previous boards took hold. All lower frequency data lines were routed next, finally followed by the power sources. This board turned out to be relatively compact, perfect for a student project to use.

Final Design

The final design consists of three modular, custom circuit boards that are designed to be assembled by the students during the first weeks of the semester. Since they will be built by the students, they were designed to minimize small surface mount components and avoid complex assembly techniques. The three boards are a main board with the microcontroller, a basic input and output board, and an Ethernet communications board. The boards are designed to interconnect with only a few external wires and no external components, making a complete system possible.

Main Board

The main board is the heart of the design project. It is used to interface with the other boards, as well as provide I/O pins for students to connect custom hardware. By itself, it constitutes a complete system that can be used with no external components. The main board has six main sections: the PIC32, a port expander, USB connectors, programming hardware, power regulation, and expansion headers. Figure 4 below shows the final circuit board.

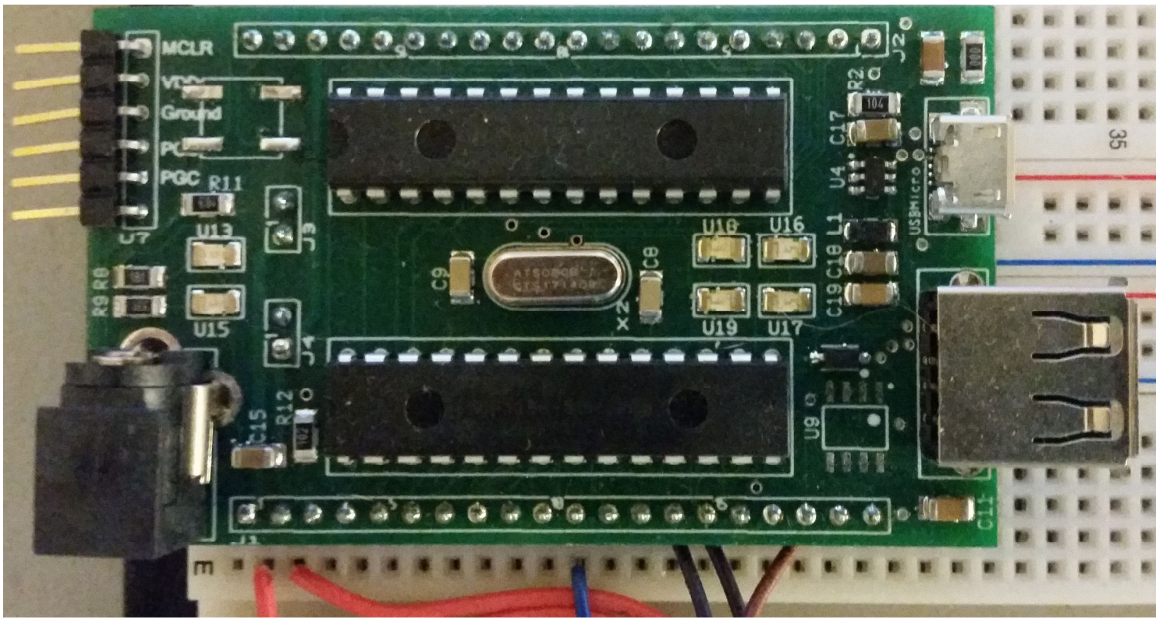

Figure 4 - Main Circuit Board

PIC32

The PIC32MX250F128B microcontroller was chosen for this system since it is the only through-hole PIC32 microcontroller available. This in no way means that it has

limited performance, however. The microcontroller is capable of high performance in a small and easy to handle package. This microcontroller is inserted into a 28-pin DIP socket on the circuit board for easy replacement if it is damaged. The microcontroller is powered by the onboard power regulators and connects to each of the other systems on board.

Port Expander

The port expander is a 20 I/O Maxim 6957 chip. Like the PIC, it sits in a 28-pin socket for ease of replacement, should that become necessary. The port expander connects to the PIC via a 4-wire SPI interface. The port expander is highly configurable, providing the students with a wide range of options. First, each pin can be set to one of the following modes:

- Output
- Input
- Input with internal pullup resistor
- Constant current LED driver

Additionally, each pin configured as an input can be configured to trigger an interrupt to the microcontroller, requesting immediate action from the microcontroller. Any pin configured as an LED driver supports 16 current settings, allowing the user to set any individual LED to a specific brightness.

Second, the port expander gives the user the ability to read or write a single pin or up to eight pins at a time. The single pin option is nice when the user requires only needing to read or write one thing, whereas reading or writing eight at a time is nice when the user requires simultaneous updates between multiple pins, such as reading the states of eight switches at once.

The final feature of the port expander is the addition of four LEDs on the PCB. These are connected to four of the pins on the port expander and allow the students to blink LEDs or make simple patterns without any external hardware at all. This feature may be utilized in the first couple of weeks of lab to become familiar with the board and its operation.

USB Connectors

The main board comes with two USB connectors: one USB type A host connector and a micro USB connector. Though the two cannot be used simultaneously, this gives the students an option to use USB in either host mode or device mode, again with no external components required.

In host mode, the PIC32 can support many standard devices, such as mice, keyboards, and flash drives. The board supplies the devices with 5V power, so the user does not need to worry about powering the devices. In device mode, the PIC can emulate any standard USB device. Students can hook the board to a computer and become a custom keyboard or joystick. If necessary, the micro USB port can provide power to the PIC32, though it will not power the port expander.

Programming Hardware

Though not a complex circuit, one of the most important connections on the main board is the programming header. This connects to a standard PICKIT 3 debugger/programmer to provide USB programming and debugging to the board.

A unique feature of the PIC32 is the ability to use different sets of pins to program and debug. While this is not an interesting feature when programming, it can greatly aide in the debugging process. If a pin is being used by the debugger, it cannot be used for any other function. Unfortunately, the PIC32 is set up to where one set of programming pins overrides USB, another ADC pins, and the last SPI pins. Fortunately, the programming section on the main board provides a switch so the user can choose which set to used based on which pins they are currently using in their design.

Power Regulation

One annoying issue when using a microcontroller on a breadboard is how to power the device. This board does not have that problem, however. An onboard barrel plug is provided, allowing the user to power the circuit with a standard 5-6 volt wall wart power supply. On the board are two regulators: a 3.3v regulator and a 5v regulator. The 5v regulator provides up to 1A at 5v to the breadboard for custom circuits, as well as powering any USB devices. The 3.3v regulator provides up to 1A at 3.3v to the breadboard, as well as powering the PIC32 and the port expander. Providing both voltages from a single supply makes it easier to build custom hardware when external chips require either voltage, but not necessarily both.

Expansion Headers

The main circuit board was designed to be plugged directly into a solderless breadboard. It does this by providing a 20-pin header on either side of the board. One of the headers provides a connection to each I/O pin on the microcontroller,

while the other provides a connection to each I/O pin on the port expander. In addition, 5v, 3.3v, and ground pins are provided on each header to power external circuits. One unique feature of the board is the additional 2-pin headers that plug directly into the power rails on the breadboard. This energizes one rail with 3.3v and the other with 5v, making it even easier to distribute power throughout the system.

Basic Input and Output Board

This board provides the students with basic inputs and outputs to manipulate without having to wire any external components. Though this board may be basic, many designs can utilize LED outputs and switch inputs, whether for debugging purposes or in the final design. The board has three main sections: the LEDs, the switches, and the seven segment displays. Since the seven segment displays and LEDs come in standard packages, students are free to use any colors they desire. Figure 5 below shows the completed board.

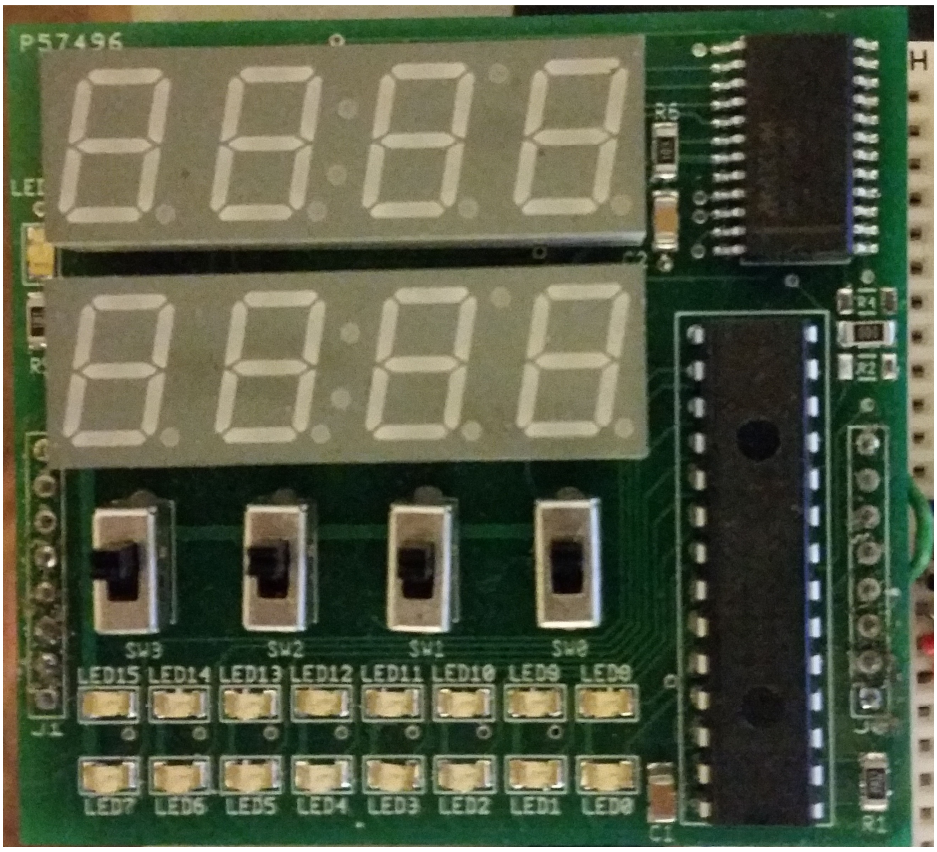

Figure 5 - Basic Input and Output Board

LED Outputs

The board provides sixteen LEDs arranged in two rows of eight. The LEDs are controlled via a port expander identical to the one on the main board. Because of this, the students can write to a single LED or eight at a time. In addition, they can control the brightness of each LED, giving them more flexibility than simply connecting an LED to the microcontroller. Since the port expander is set in constant-current mode for the LEDs, no resistors are necessary. This not only reduces the part count, but makes it easier for the students to solder.

Switch Inputs

Since only sixteen of the twenty I/O pins on the port expander are connected to LEDs, the remaining four pins are connected to SPST switches. In the on position, the switches are connected to 3.3v, and connected to ground in the off position. Again, no external resistors or anything are needed; they are simply connected to the port expander. The switches are set to normal inputs by default, but could be adjusted to create an interrupt to the main board (if desired).

Seven Segment Displays

The board also includes two rows of four seven segment displays. Since seven segment displays are normally driven with a multiplexing circuit [10], a Maxim 7319 chip is used to drive the displays. This means the displays can be updated by simply sending commands via the SPI interface; no time-sensitive refreshing code is needed to drive them. A software lookup table has been provided to display decimal and hexadecimal digits on the display. This makes it easy for the students to put a score or specific value on the displays without worrying about decoding. Like the port expander, different brightness values can be set.

Communications Board

The communications board is meant to provide all of the components needed to add Ethernet connectivity to a project without complex circuitry. This board includes both the connector and the external PHY layer embedded in an external chip. This chip handles packet filtering, auto-negotiation, and timing. It is connected to the main board via an SPI connection. Students will have to attach resistors and capacitors to this board, but no off-board connections are needed except for the SPI interface. The communications board can be seen in Figure 6 below.

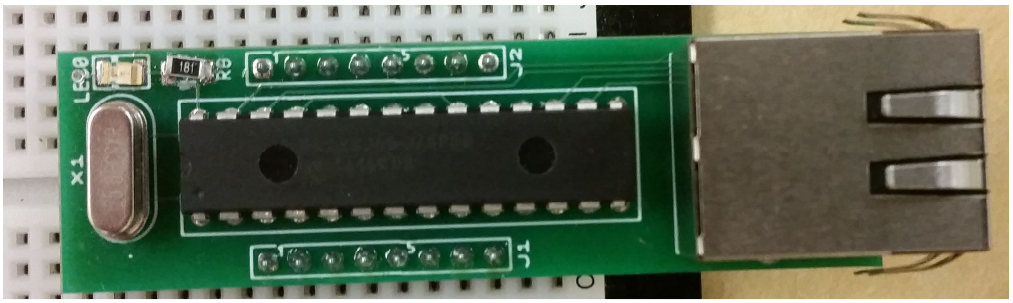

Figure 6 - Communications Board

Results

Hardware and Software

Each of the three circuit boards ended up working exactly as expected. The main board went through two iterations to fix a couple of small issues, but there were no prohibitively large issues. The second revision solved the issues and works exactly as expected. Figure 7 below shows a basic system that implements the classic Tetris game. The main board runs the game and hosts the USB keyboard. The basic I/O board is used to display the score in binary, hex, and decimal. Finally, the LCD screen is used to actually display the game. As desired, all of these systems were interconnected using only a few wires and no external components.

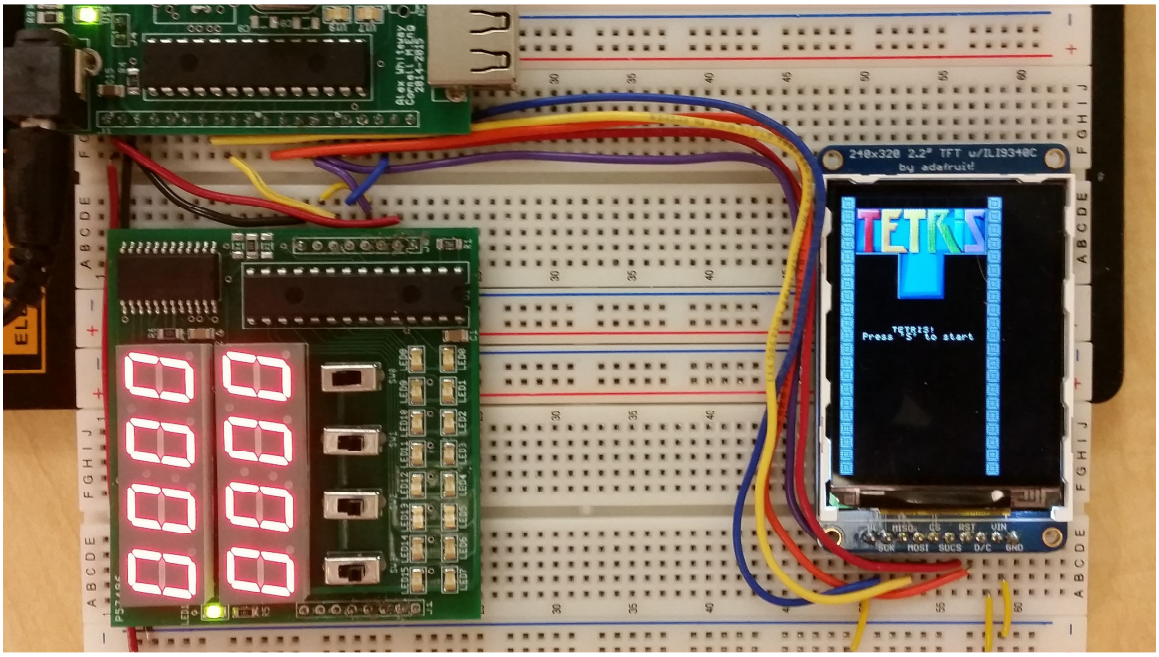

Figure 7 - Example System

The system meets all the initial project requirements for both hardware and software capability. Students are free to use any feature they choose, but some features are much easier to use thanks to the circuit boards and software examples.

Laboratory Exercises

The main purpose for this design project was to use the results for laboratory exercises. Some possible lab assignments are listed below.

Board Assembly and Introduction

This would be the first lab assignment. The students would build the main board and program it to manipulate the LEDs on the main board. The LEDs could be used as a basic counter or simply flash. Since these LEDs can be driven at different brightness levels, the students could also create sinusoidal effects on the board. These simple ideas would not require any extra components, and it would also verify the PIC and port expander are working correctly.

Basic Voltmeter

The analog pins on the microcontroller could be used to read voltages, capacitances, etc. The seven segment displays on the basic I/O board could be used to display the meter readings. Additionally, the switches on the board could set the voltmeter function or range, fully exercising the basic I/O board in addition to using the PIC32's ADC channels.

Basic Oscilloscope

As an add-on to the voltmeter idea, an LCD screen would give the students the ability to create a simple oscilloscope. Multiple channels could be displayed using different colors, making a multi-channel system possible. Though it may not be as accurate as normal oscilloscopes, it could provide a helpful debugging tool for future use.

Oscilloscope Revisited

Most oscilloscopes provide ways to manipulate what data is shown on the screen, not to mention the ability to adjust time and voltage scales. A USB keyboard could be used to add this functionality to the basic oscilloscope. Using the arrow keys, +/- keys, etc. would make this a more useful oscilloscope.

Another possibility to expand the oscilloscope would be to add the Ethernet module to the oscilloscope, allowing data to be saved to the computer or viewed on the computer monitor instead of the much smaller LCD screen. Any combination of these features would make excellent lab exercises.

Audio Player

Audio functionality comes with many of the electronics we use every day. Creating a simple audio player would provide valuable experience that could be used later, either in lab exercises, hobby projects, or industry. Many options exist, including using the LCD screen to navigate and select songs, the keyboard to control the audio, or even Ethernet to stream the media over the network.

Video Game

Similar to the Tetris example produced for the design project, each lab group could program a simple video game. One method to make this exercise more educational and interesting may be to require each team to implement the same game. Using a standard defined by the professor, the scores could be uploaded to a web server in real time using the Ethernet module. This would enable the students to enjoy a friendly competition while also learning many valuable skills.

Other Options

The ideas above are presented as preliminary ideas. The vast feature set of the PIC32 provides nearly endless possibilities for lab exercises and final projects alike. Combining the main board, communications board, basic I/O board, and LCD screen expand these possibilities. Hopefully the capabilities will entice the students to come up with spectacular projects not necessarily possible in the past.

Future Work

Though this design project was overall a success, there are many more things that could be done. More exciting features could definitely added as additional modular components. Some of these modules could be:

- A GPS Unit
- Accelerometers
- Bluetooth Module
- Wi-Fi Module
- Motor Control Module
- Variable Power Supply Module
- RAM Module

Future development may also include better documentation. Once the system has been used for a semester or two, the shortcomings will become apparent and could be addressed for future semesters. These shortcomings could include hardware, software, or documentation. These issues will need to be addressed in the future to guarantee the long term success of the project.

References

[1] W. Young. (2011) What is a "Breadboard"? [Online]. Available: http://tangentsoft.net/elec/breadboard.html

[2] Circuits Today. (2013, October) Microcontroller - Invention History and Story Behind the Scenes. [Online]. Available: http://www.circuitstoday.com/microcontroller-invention-history

[3] Grand View Research, "Microcontroller Market Analysis By Product, By Application, and Segment Forecasts to 2020," Grand View Research, Market Analysis ISBN 978-1-68038-141-2, 2014.

[4] MikroElektronika. (1998-2015) Introduction to Microcontrollers. [Online]. Available: http://www.mikroe.com/chapters/view/64/chapter-1-introduction-to-microcontrollers/

[5] Microchip. Microchip's 32-bit Microcontrollers. [Online]. Available: http://www.microchip.com/pagehandler/en-us/family/32bit/

[6] IEEE and The Open Group, "General Information," IEEE, Standard IEEE Std 1003.1, 2013.

[7] Adafruit. 2.2" 18-bit color TFT LCD display with microSD card breakout - ILI9340. [Online]. Available: http://www.adafruit.com/products/1480

[8] Microchip. (2011) Section 23. Serial Peripheral Interface (SPI). [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/61106G.pdf

[9] BusBoard Prototype Systems. Solderless Plug-In Breadboards. [Online]. Available: http://www.mouser.com/ds/2/58/BPS-MAR-%28BB170%29-001-257165.pdf

[10] Y. Tambi. Seven Segment Multiplexing. [Online]. Available: http://maxembedded.com/2013/01/seven-segment-multiplexing/

Appendix A: PIC32 Features

This appendix lists all of the available features of the PIC32 microcontroller family. Some of these features were used in the final project, while others are left for more advanced students to explore on their own. In addition, students may interface custom circuits with the microcontroller to expand functionality beyond what is listed here.

- Direct Memory Access Channels
- 50 MHz SPI
- I2C
- CODEC Interfaces
- Peripheral Pin Select
- Charge Time Measurement Unit (Capacitive Touch Sensing)
- 1 Msps 10-bit Analog to Digital Converter
- Parallel Master Port
- Input Capture
- Output Compare/PWM
- Comparators
- Real-Time Clock and Calendar
- Full Speed USB OTG
- Ethernet
- Timers
- Watchdog Timer
- Debugger
- Operating System Support

Appendix B: Schematics

Main Board

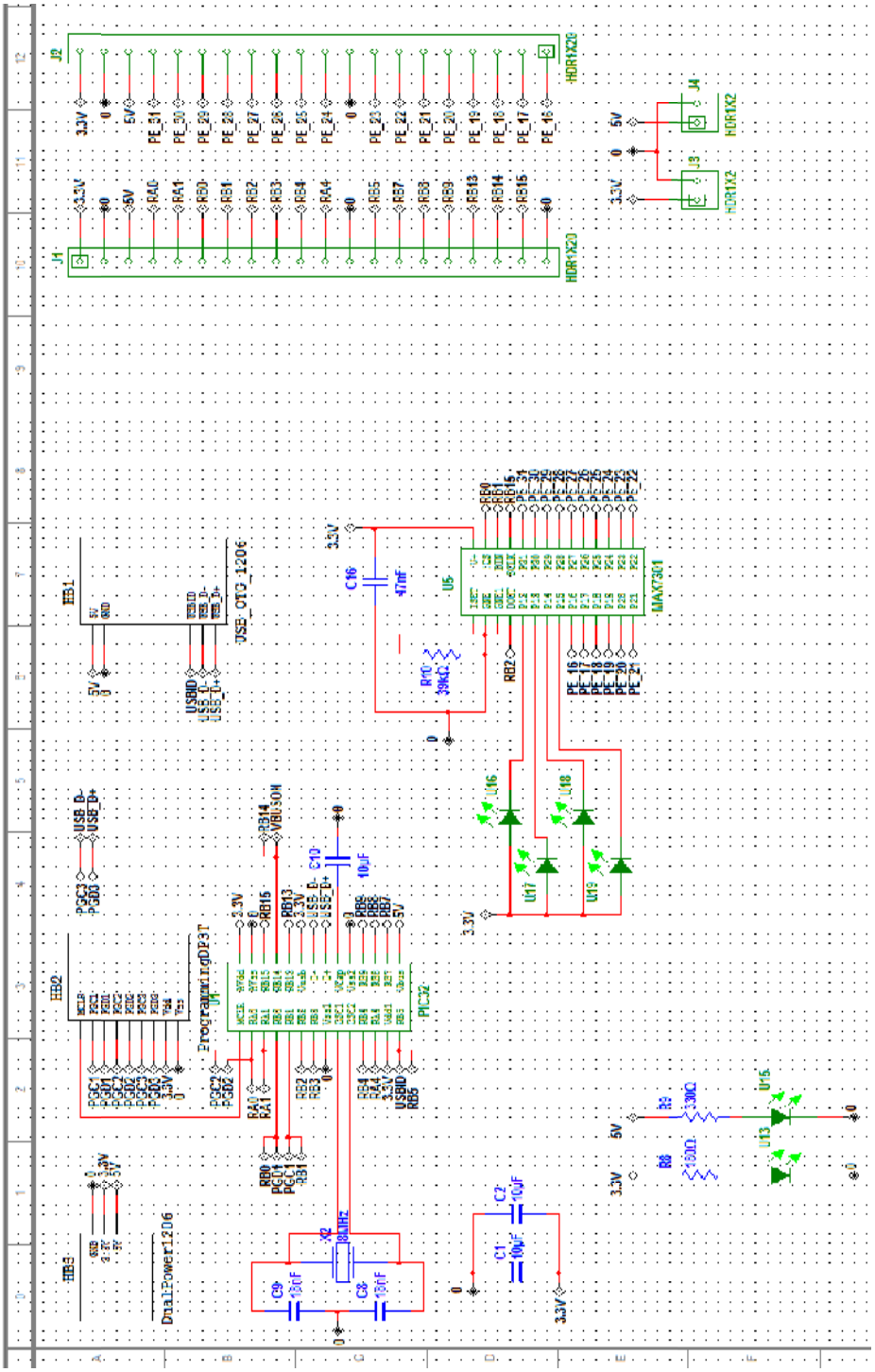

Figure 8 - Main Schematic

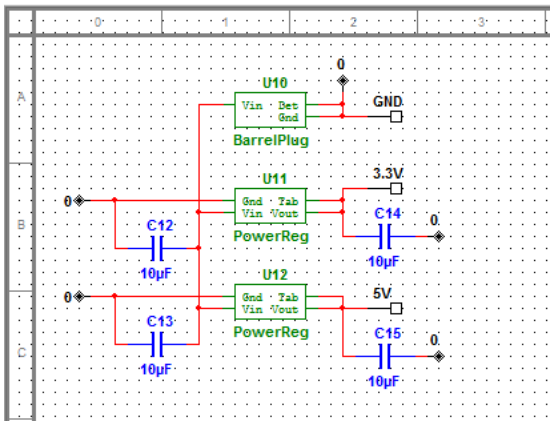

Figure 9 - Power Regulation Schematic

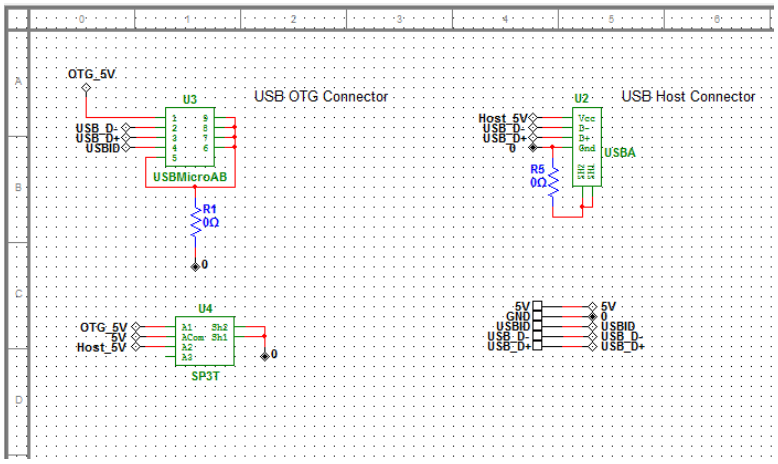

Figure 10 - USB OTG Schematic

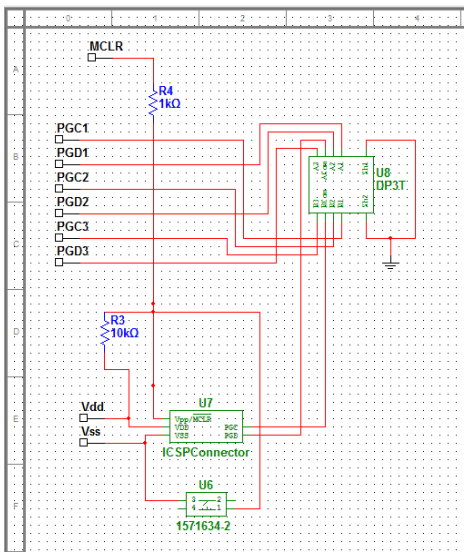

Figure 11 - Programming Schematic

Basic Input and Output Board

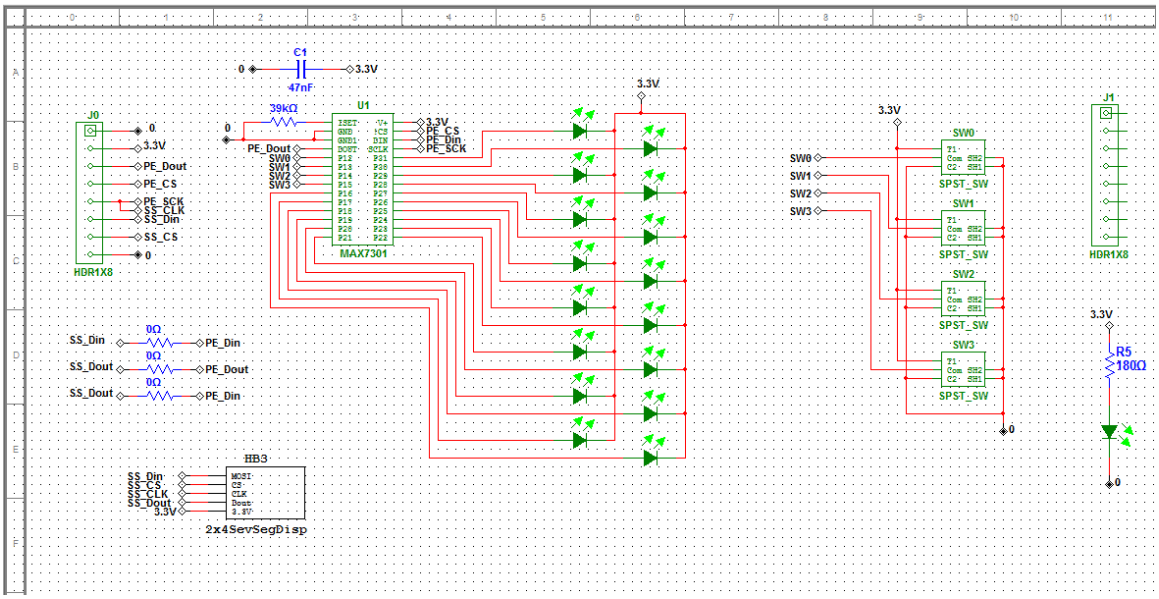

Figure 12 - Basic Input and Output Board Schematic

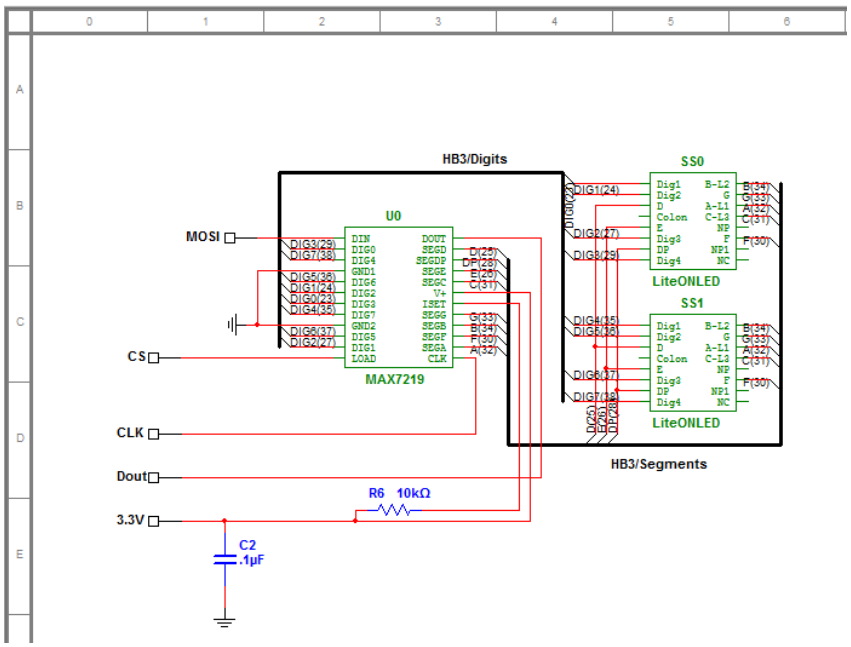

Figure 13 - Seven Segment Display and Driver Schematic

Communications Board

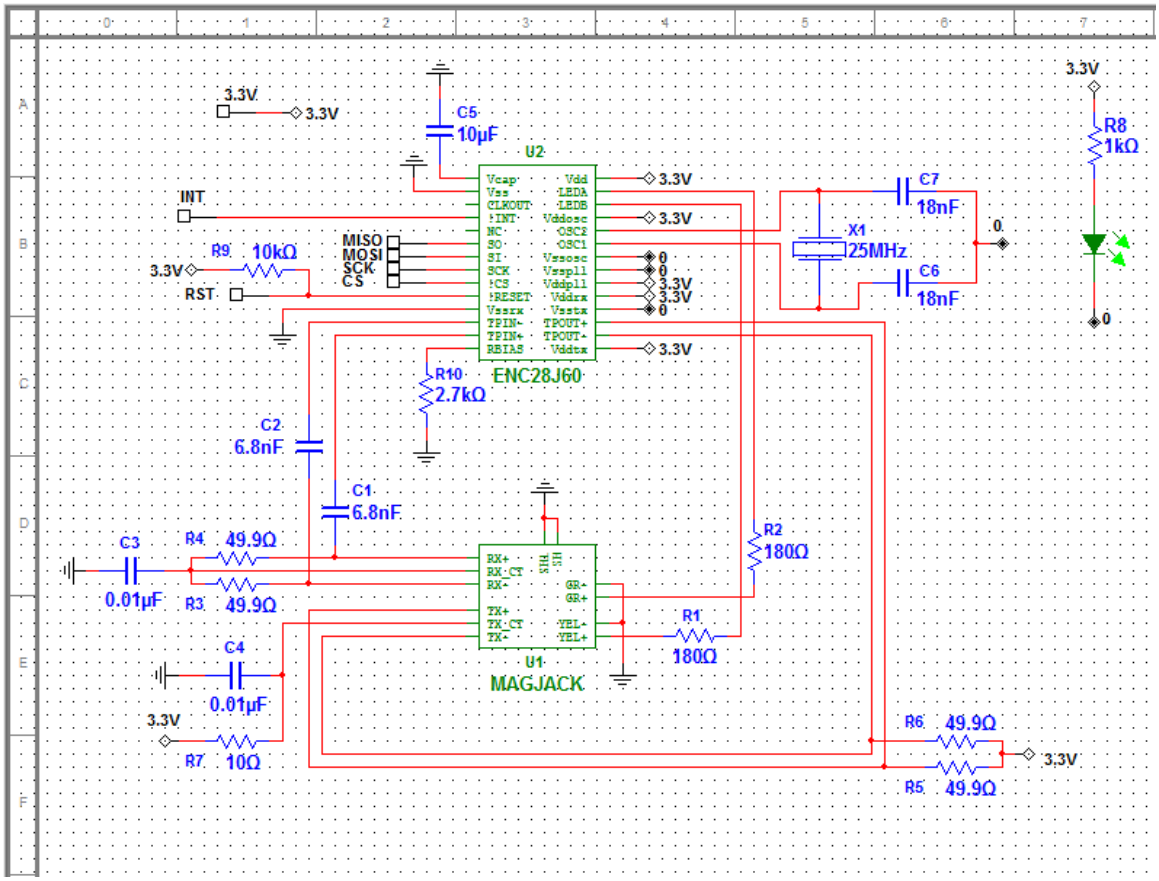

Figure 14 - Communications Board Schematic

Appendix C: Board Layouts

Main Board

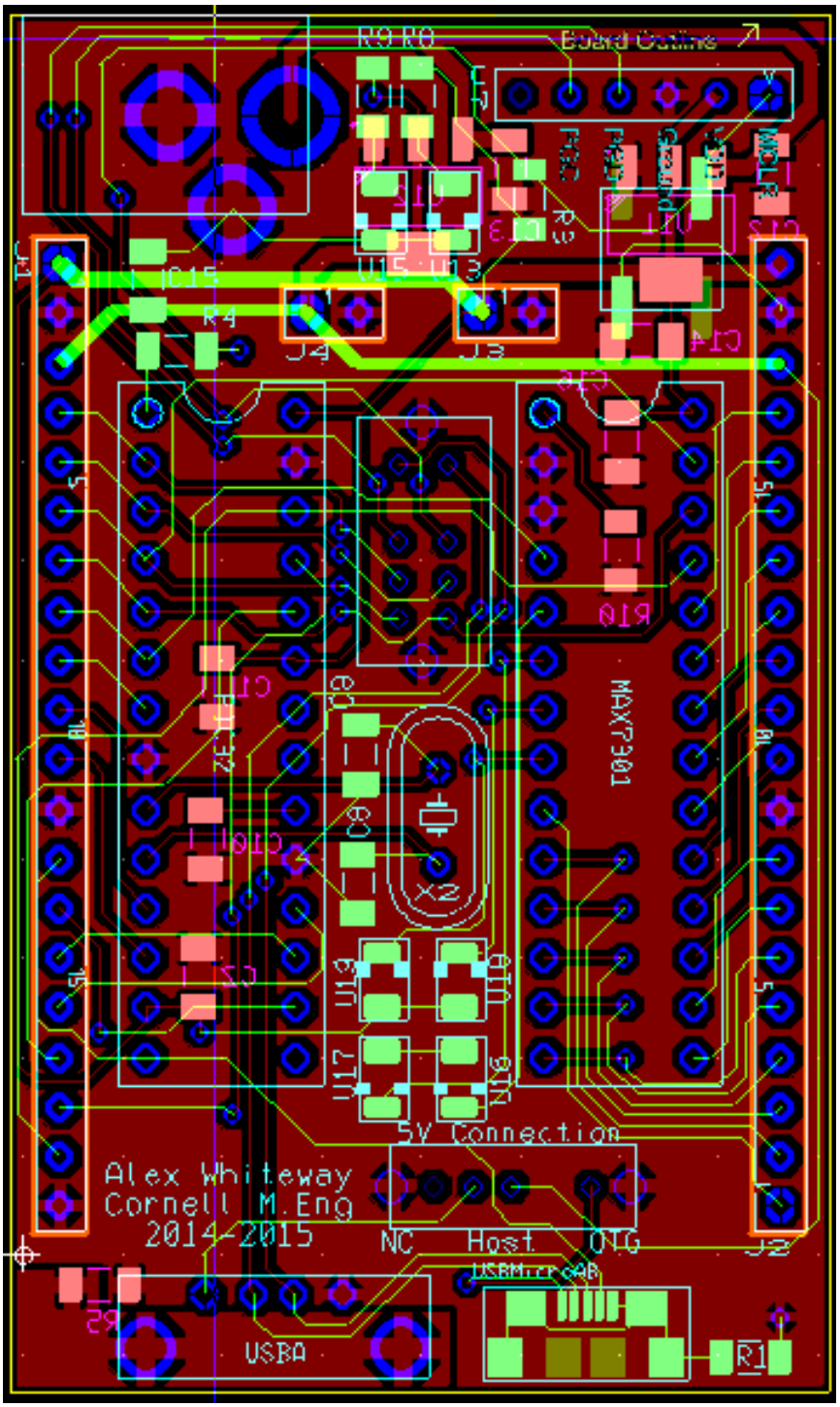

Figure 15 - Main Board Layout

Basic Input and Output Board

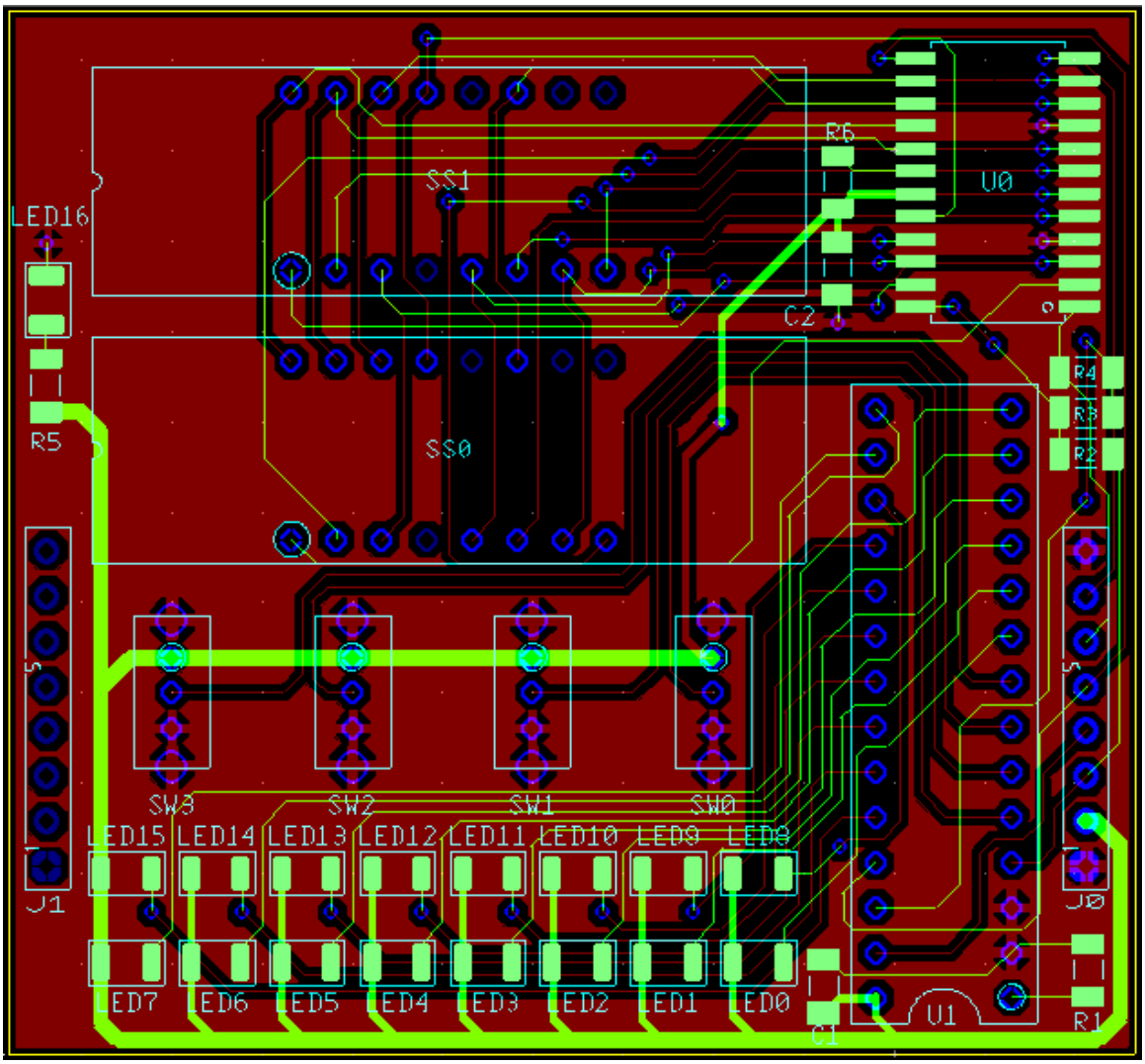

Figure 16 - Input and Output Board Layout

Communications Board

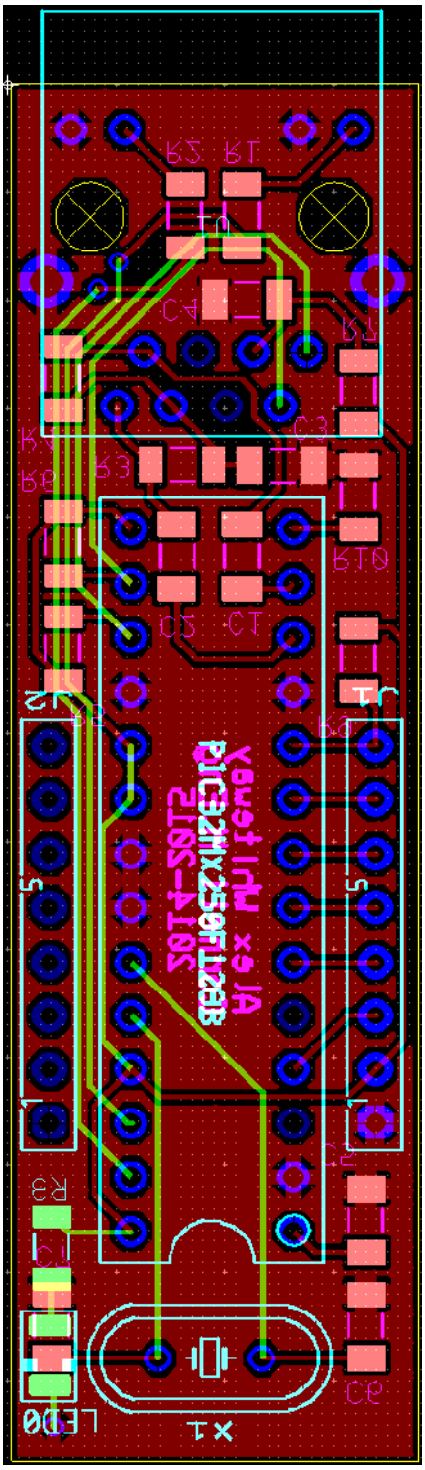

Figure 17 - Communications Board Layout

Appendix D: Parts List

Part	Value	Package	Main	LED	Comm	Total	Mfr. #	Mfr	Digikey #
7 Seg Display	Common Cath	DIP16			2	2	LTC-4727JR	Lite-On	160-1551-5-ND
7 Seg Driver		SO24W			1	1	MAX7219CWG+T	Maxim	MAX7219CWG+TTR-ND
Barrel Plug		2.1MM		1		1	RAPC722X	Switchcraft	SC1313-ND
Buckboost Conv		SOT-23-6		1		1	AP1603WG-7	Diodes Inc	AP1603WG-7DI-ND
Capacitor	0.01uF	1206				2	CL31B103KBCNNNC	Samsung	1276-1035-1-ND
Capacitor	0.1uF	1206		1	1	2	CL31B104KBCNNNC	Samsung	1276-1017-1-ND
Capacitor	10uF	1206		7		8	CL31A106KPHNNNE	Samsung	1276-1148-1-ND
Capacitor	18nF	1206		2		4	CC1206KRX7R9BB183	Yageo	311-1202-1-ND
Capacitor	1uF	1206		1		1	CL31B105KOFNNNE	Samsung	1276-1783-1-ND
Capacitor	47nF	1206		1	1	2	CL31B473KEHNNNE	Samsung	1276-3172-1-ND
Capacitor	47uF	1206		2		2	CL31A476MQHNNNE	Samsung	1276-1167-6-ND
Capacitor	6.8nF	1206				2	CL31B682KBCNNNC	Samsung	1276-1775-1-ND
Capacitor	6.8uF	1206		1		1	C3216X6S1A685K085AB	TDK	445-14757-1-ND
Crystal	25MHz	HC-49US				1	ABL-25.000MHZ-B2	Abracon	CTX928-ND
Crystal	8MHz	HC-49US		1		1	ABL-8.000MHZ-B2	Abracon	CTX900-ND
Diode	Schottky 50v	SOD-123		1		1	B0530W-7-F	Diodes Inc	B0530W-FDICT-ND
Eth Conn	Magjack	ZIP8				1	ARJC02-111009D	Abracon	Sparkfun PRT-08534
Header	1x2	SIP2		2		2	961102-6404-AR	3M	3M9447-ND
Header	1x20	SIP20		2		2	PRPC020SAAN-RC	Sullins	S1011EC-20-ND
Header	1x6	SIP6		1		1	961106-5604-AR	3M	3M9471-ND
Header	1x8	SIP8			2	2	PRPC008SFAN-RC	Sullins	S1211EC-08-ND
Inductor	22uH	1206		1		1	LQH31CN220K03L	Murata	490-6591-1-ND
LED	Any color	1206		6	17	1	24 APTR3216SGC	Kingbright	754-1171-1-ND
PHY/MAC		DIP28				1	1 ENC28J60-I/SP	Microchip	ENC28J60-I/SP-ND
PIC32	MX250F128B	DIP28		1			1 PIC32MX250F128B-50I/SP	Microchip	PIC32MX250F128B-50I/SP-ND
Port Exp.	MAX6957	DIP28		1	1		2 MAX6957ANI+	Maxim	MAX6957ANI+-ND

Power Reg	3.3V	SOT-223	1			1	AZ1117EH-3.3TRG1	Diodes Inc	AZ1117EH-3.3TRG1DICT-ND
Power Reg	5V	SOT-223	1			1	AZ1117EH-5.0TRG1	Diodes Inc	AZ1117EH-5.0TRG1DICT-ND
Resistor	0Ω	1206	2	3		5	RC3216J000CS	Samsung	1276-3514-1-ND
Resistor	100kΩ	1206	1			1	RC3216F104CS	Samsung	1276-5781-1-ND
Resistor	10kΩ	1206		1	1	2	RC3216F103CS	Samsung	1276-5742-1-ND
Resistor	10Ω	1206			1	1	RC3216F100CS	Samsung	1276-3513-2-ND
Resistor	180Ω	1206			1	1	RC3216F181CS	Samsung	1276-5679-1-ND
Resistor	1kΩ	1206	2	1	1	4	RC3216F102CS	Samsung	1276-5711-1-ND
Resistor	2.7kΩ	1206			1	1	RC3216F272CS	Samsung	1276-5725-2-ND
Resistor	330Ω	1206	2		1	3	RC3216F331CS	Samsung	1276-5690-1-ND
Resistor	39kΩ	1206	1	1		2	RC1206FR-0739KL	Yageo	311-39.0KFRCT-ND
Resistor	49.9Ω	1206			4	4	RC1206FR-0749R9L	Yageo	311-49.9FRCT-ND
Switches	Off-Mom SPST	6.2mmx6.2mm	1			1	KSC741G LFS	C&K	CKN10305CT-ND
Switches	On-Off Slide			4		4	OS102011MS2QN1	C&K	CKN9565-ND
USB Pwr Sw.		SOIC-8	1			1	MCP1253-33X50I/MS	Microchip	MCP1253-33X50I/MS
USB-A		USB	1			1	UE27AC5410H	Amphenol	UE27AC5410H-ND
USB-AB		Micro	1			1	798-ZX62-AB-5PA11	Hirose	H11635CT-ND
WallWart	5V		1			1	WSU060-1250	Triad	237-1421-ND