Howard Activity Monitor (HAM)

Final Report

Spring 2015

Team Members:

Calvin Campopiano

Chris Catalino

William Schell

Michael Silvernagel

Colin Terndrup

Table of Contents

### 1 Introduction……………………………………………………………………..pg. 3

1.1 Problem Statement.........................................................................pg.3

1.2 High Level Design.........................................................................pg.3

1.3 Device Success…...........................................................................pg.3

### 2 Detailed System Requirements……………..…………………………………pg. 4

### 3 Detailed Project Description…………………………………………………...pg. 4

3.1 System Theory of Operation.........................................................pg. 4

3.2 System Block Diagram..................................................................pg. 5

3.3 Subsystem: Accelerometer............................................................pg. 6

3.4 Subsystem: Light Sensor...............................................................pg. 7

3.5 Subsystem: Memory......................................................................pg. 8

3.6 Subsystem: Battery/Charging........................................................pg. 10

3.7 Subsystem: Data Communication..................................................pg. 11

3.8 Subsystem: Command Line Software Interface............................pg. 12

3.9 Subsystem: Board Design & Packaging.........................................pg.14

### 4 System Integration Testing……………..………….……………………….…pg. 16

4.1 Description of Testing....................................................................pg. 16

4.2 Requirements..................................................................................pg. 18

### 5 User Manual……………..………………………………………………….pg. 18

5.1 Installation.....................................................................................pg. 18

5.2 Startup...........................................................................................pg. 19

5.3 Troubleshooting/Correct Functionality..........................................pg. 21

### 6 To Market Design Changes……………..…………..……………………..….pg. 21

6.1 Packaging and Board Size..............................................................pg. 21

6.2 Parts and Device Cost.....................................................................pg. 22

6.3 UV Sensor Calibration and Data Management...............................pg. 22

### 7 Conclusion………………………….……..…………..……………………….pg. 23

### 8 Appendices ……………………….……..…………..……………………...….pg. 23

8.1 Subsystem Data Sheets..................................................................pg. 23

8.2 Board Schematics...........................................................................pg. 24

8.3 MPLABX Code..............................................................................pg. 31

# **1 Introduction**

The Howard Activity Monitoring (HAM) group has worked to create a compact, wearable, cost-effective device that gathers activity data for researchers at the University of Notre Dame. The sensing components of the device consist of an accelerometer to track the overall movement of the wearer and a light sensor to detect the ambient light in the subject’s environment. The data from these sensors are time stamped and saved on flash memory, with the ability to be exported via USB in a form compatible with existing software used by Notre Dame’s Interdisciplinary Center for Network Science and Applications (iCeNSA). The design and components were chosen based on the adherence to the following design constraints: cost, size, durability, and battery life. After the creation of the initial prototype—where the sensors were configured to communicate and effectively store and export data—performance testing of the device was completed. A final working device, created on a hand designed circuit board, was produced and demonstrated to the customer. It is the group’s hope that future iterations of this design can be completed, with the final design existing in buildable kit form so that monitoring devices can be assembled by local South Bend students and sent into the field for research.

1.1 Problem Statement

Obesity is an issue which plagues the United States, and this condition is particularly prevalent among children and adolescents in the South Bend community. Looking to combat this problem, researchers at the University of Notre Dame School of Architecture hope to study the correlation between a child’s living environment and their level of physical activity. Activity monitoring devices are available commercially through companies like Actigraph (<http://www.actigraphcorp.com/>) , but these monitors cost upwards of $200, reducing the ability of these devices to be widely employed by groups with limited funding. Further, as the target subjects are children and adolescents, these devices should be easily replaceable and fairly durable. Without these devices, these researchers cannot collect the data they desire. To combat this problem, the Howard Activity Monitoring group has worked to engineer a cost-effective, wearable device that gathers and communicates activity data about the user to the iCeNSA software for analysis.

1.2 High Level Design

To accomplish our design, we have integrated two sensors—an accelerometer and a light sensor—to record activity data of the test subjects. We then store this data using flash memory and export the data through USB in a format compatible with existing software used by iCeNSA. Performance testing was completed after the integration of our subsystems, and a final working product was constructed and demonstrated.

1.3 Device Success

At the completion of this project, the HAM group was able to create a device with all subsystems working, which accurately recorded activity and light data, stored the values to memory, and could later export the values to the user in a text file format. Several refinements and improvements can be implemented, however, and these are discussed in section 6. Despite this, The HAM group is pleased with the overall functionality of the device.

# **2 Detailed System Requirements**

There are several system requirements that the proposed activity monitor must meet in order to satisfy the customer’s needs. First, the device must be able to record activity data consisting of the user’s acceleration and the ambient light in their environment, sampled at a rate of once per second when the user is active. The occurrence of this data will be tracked by a real time clock, and this information will be saved to the device’s internal flash memory, which must be large enough to store measurements for three consecutive 12 hour days. After data collection is completed, the user information will be transferred via USB in a format compatible with iCeNSA’s software. The device will be powered by a rechargeable poly lithium ion cell battery which will provide enough power to allow the device to operate for a minimum of 12 hours on a single charge; the battery will be recharged via USB. In order to achieve this battery lifetime, the device will enter a low-power “sleep” mode when data is not being collected. When the device is active, both sensors and the memory device will be configured to operate with minimal power consumption while maintaining appropriate data accuracy. A microcontroller will be utilized to communicate and coordinate the multiple subsystems located within the device.

The mechanical constraints on the monitor consist of the device’s size, weight, and durability. The device must be small and light enough to be wearable on the user’s body, ideally the arm or wrist, without inhibiting the user’s normal activity. As such, the final device outside of its package should be roughly the size of a credit card. Due to the physical nature of the data being recorded, the device must also be able to withstand significant shock and be water resistant; this will be accomplished by purchasing and modifying a case that is composed of an environmentally friendly material.

In addition to meeting the above constraints, all device components should be of minimal cost, as the customer wishes to produce several devices. Therefore, the target price for the end produce is below $50. If possible, the construction process of the device should be simple enough that it can be assembled by students in South Bend schools, further integrating the community into this project.

# **3 Detailed Project Description**

3.1 System Theory of Operation

The overall operation of this activity monitor is relatively straightforward. Two sensors—an accelerometer and a light sensor—will be used to track the user’s movement and environment, with the accelerometer calculating acceleration in the x, y, and z axes and the light sensor providing UV light readings in the test subject’s environment. This data will polled once a second when the user is active and time stamped using a real time clock. When data is not being recorded, the device components will be placed in sleep mode to conserve power. All information will be stored to flash memory via the microcontroller, and after data collection has occurred, the information stored on the device can be accessed through a USB connection. During this process, the microcontroller will transfer data to the FTDI chip, which acts as an interface between the UART transfer utilized by the microcontroller and the USB transfer used by the computer on the user end. Data can be accessed through the computer’s COM port, and this information will be formatted to the specifications of existing iCeNSA software. Powering the device is a polymer lithium ion battery, with the output voltage stepped down from 3.7 to 3.3 volts through a linear regulator. A durable, water-resistant case will cover all device components. As the primary users of these devices will be children, no external buttons will be on the device; the only alterations to the solid casing will be a light well for the light sensor and an LED to indicate when the device is charging.

3.2 System Block Diagram

The overall system is comprised of the following subsystems: microcontroller, memory, data collection, battery, data connectivity, and case. The data collection subsystem can be further divided into the accelerometer, light sensor, and real time clock. A block diagram and hierarchy of the overall system are shown below in Figures 1 and 2.

Memory

USB

Battery

Accelerometer

uController

Light sensor

FTDI

Real time clock

**Figure 1**: Block Diagram of Overall System

**Figure 2**: Overall System Hierarchy

3.3 Subsystem: Accelerometer

*Subsystem Requirements*

As the main measurement device in the activity monitor, the accelerometer must meet several requirements to ensure the device functions as intended. First, while the subject is active, it needs to record activity in the x, y, and z directions. The second requirement is that the device generates an interrupt when the user’s activity falls above or below a predetermined value. When this occurs, components on the activity monitor will be placed in sleep mode to conserve power. The third requirement on the accelerometer subsystem is that while active, it samples at a rate of approximately once a second. The final requirements are that the component should operate at low currents and be of minimal cost.

*Subsystem Function*

The function of the accelerometer is straightforward. While the HAM monitor is being used and the minimum activity level is being reached, the accelerometer will record measurements and communicate the data via I2C to the microprocessor, which will then store the data in the monitor’s memory. When the monitor isn’t being used, such as during periods of sleep, the accelerometer will enter a low power mode and will not awaken until sufficient activity is again reached, generating an internal interrupt and placing the accelerometer back in active mode.

*Engineering Decisions*

Several different criteria were considered when choosing which accelerometer was to be used in the monitor. Due to size constraint on the monitor, and therefore battery constraints, the accelerometer must be able to enter a low power mode when not in use. In terms of measurement, due to the comparatively low range of accelerations expected to be measured, there is little value in having an accelerometer that measures a large range of accelerations; rather a device with a range roughly around +/- 20g was decided to be ideal in terms of range and precision. In terms of either an analog or digital sensor, a digital sensor was chosen so that data could be directly stored from the sensor rather than using an analog sensor and having to perform a conversion to digital values. Size and price of the device were also considered, and after considering desired parameters, the ADXL345 Accelerometer was chosen. The ADXL345 is a digital sensor with a range from +/- 16g with the ability to enter a low power sleep mode. In comparison to other accelerometers, the ADXL345 was comparable size wise and had one of the lowest prices at $7.32.

*Subsystem Testing*

To test the basic performance of the accelerometer, the sensor was linked to the microprocessor with the measurements being displayed on the connected LED screen. Once it was determined that accurate values were being achieved, the low power mode was tested. Using a threshold value of 2g, the accelerometer was placed at rest with the logic analyzer attached to the microprocessor in order to determine when the interrupt signal was sent. Moving the sensor resulted in a sufficient reading to wake the sensor, generating an interrupt that was verified by the logic analyzer. These two tests demonstrated that the microprocessor was successfully communicating with the accelerometer and that its low power mode was able to be entered and exited.

3.4 Subsystem: Light Sensor

*Subsystem Requirements*

Before choosing a light sensor for our project, we outlined many criteria the prospective light sensor would have to fulfill in order to be a suitable candidate. These criteria included the capacity to easily interface with a PIC32mx microcontroller, ability to distinguish between indoor and outdoor light, low power consumption, low cost, and a small physical package. We ultimately chose the SI1145 Digital/UV Index/IR/Visible Light Sensor from Adafruit because it best matched the criteria listed above and because a breakout board version of the sensor was easily available online for quick prototyping.

*Subsystem Function*

The basic function of the light sensor is to measure the UV index associated with ambient light at a rate specified by the customer. When the light sensor completes is measurement, it sends an interrupt to the microcontroller, notifying it that a new measurement has been recorded. The microcontroller and the light sensor communicate through the I2C protocol.

*Engineering Decisions*

At the outset of this project, the customer clearly indicated the desire to collect data that would show whether or not a child was indoors or outdoors while measuring and recording the child’s activity levels. Therefore, arguably the most important criteria for our light sensor is its ability to distinguish between indoor and outdoor light sources. Our group decided one way of solving this problem was to choose a light sensor that can measure ambient UV levels, which would normally be low indoors and significantly higher outdoors to due to UV energy emitted by the sun. Once we narrowed down our options keeping in mind the light sensor must have the capability to measure UV and be available on a breakout board for prototyping purposes, we were left with two choices. The primary difference between the two choices was one option was a digital sensor and the other was an analog sensor. Both sensors are relatively cheap (under $10 for single unit), consume relatively little power, and were fairly easy to interface with the microcontroller – the digital sensor could communicate with the microcontroller through the I2C protocol, which we already had software developed to support, and the analog sensor through a microcontroller I/O pin. At this point it was clear the analog sensor had a simpler interface, since it essentially just outputs a current proportional to the UV index and requires fewer microcontroller I/O pins for interface than does the digital sensor. However, using the analog sensor would require the microcontroller to have to perform analog-to-digital conversion. We favored the digital light sensor because it relieved the microcontroller from performing analog-to-digital conversion and therefore allowed the microcontroller to be more responsive to sampling measurement time constraints. Also, despite having a more complex interface to the microcontroller, the digital light sensor has multiple operation modes including an idle mode, which is ideal for minimizing the sensor’s power consumption at times when the activity monitor need not be measuring light values (i.e. the child wearing the monitor is sleeping in bed). After selecting the light sensor for our project through a structured and analytical decision making process resulting in a group consensus choice, we are confident the light sensor we have chosen is the best fit for our project.

*Subsystem Testing*

To test the light sensor, we connected it to the I2C pin headers on the kit board and took the kit board outside, where there was significant UV exposure. We created a test program, which included code to complete a UV measurement twice per second and display it to the LCD screen also connected to the microcontroller. Once the LCD display clearly showed reasonable UV readings consistent with what should be expected, we verified the light sensor was functioning as intended.

3.5 Subsystem Memory

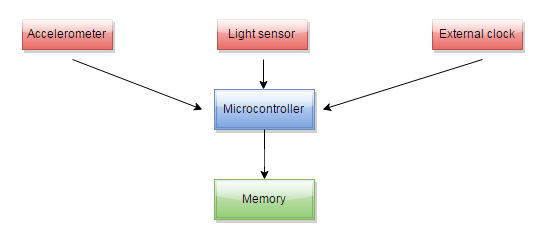
*Subsystem Requirements*

When analyzing the memory component of the HAM monitor, this device should adhere to three key constraints: (1) possess sufficient storage capacity to store activity data for three consecutive 12 hour days; (2) consume minimal power when active; and (3) be priced competitively.

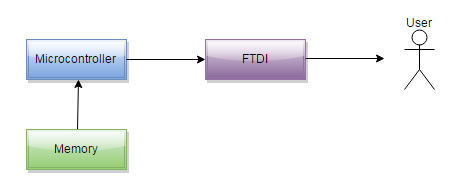
*Subsystem Function*

The function of memory in the HAM monitor is relatively straightforward. When the device is employed in the field, data from the external clock, accelerometer, and light sensor will be sent to the microcontroller, which will then store those values in the memory device. That data will remain in memory until the completion of the field study. At this time, researchers will read data from the device, a process that consists of the microcontroller accessing data from memory and sending that data to the FTDI chip, which in turn interfaces with devices on the user end. Simple block diagrams for the write and read functions of the memory are given below.

*Write*



*Read*



**Figure 3:** Memory Write & Read Functions

*Engineering Decisions*

When active, the HAM monitor will track five distinct variables: real time clock data, accelerometer readings in the x, y, and z directions, and UV light values. While extreme precision is likely not necessary for each of these values, to ensure that storage capacity is sufficient, each of these values was estimated to be stored as an eight byte double. When active, the HAM monitor will poll data once per second, and assuming the unlikely scenario where the device is continually active for 12 hours each day over three days, the minimum number of bits necessary for the device can be calculated as follows:

Therefore, a 128Mbit device would provide us sufficient memory storage with an additional buffer, giving the HAM monitor the potential be used for longer testing periods if so desired in the future. When examining various 128Mbit devices, the Spansion S25FL127S emerged as the top option for several reasons. First, it runs at low currents during operation (7mA read, 20 mA program/erase, 0.015 mA standby), which is essential for our device, as power management is a great concern. While these current values are comparable to similar devices, the Spansion S25FL127S greatly differentiated itself from other competitors through price; many 128Mbit devices range from $1.50 to $3.00 in price, with the Spansion S25FL127S falling near the bottom of this range at $1.89. Further, this device comes in an 8-SOIC package that allows for easy attachment to the board, supports dual and quad read options should faster data transmission be desired, and is accompanied by a robust datasheet that will assist in device implementation. It is the combination of these advantages that prompted our group to implement the 128Mbit Spansion S25FL127S in our design.

*Subsystem Testing*

To test the basic performance of the memory, the values from the light sensor were sent to the microcontroller, which then stored those values sequentially in memory. After each value was written to memory, the microcontroller then called a function to read back the stored value. Using a logic analyzer, the value read by the light sensor was determined, and this value was checked against the values written to and read from memory to ensure that all three were in agreement. When communication through the COM port is completely established, further testing will be completed to ensure that values in memory match data read back by the user.

3.6 Subsystem: Battery/Charging

*Subsystem Requirements*

The battery and charging subsystem of the activity monitor is crucial to the function and lifetime of the device. The battery and charging interface must be able to fulfill a series of requirements in order for the monitor to be effective. The most important requirement is the lifetime supplied by the battery. As requested by the customer, the device must be able to last a minimum of 12 hours to the last throughout the active portion of the child’s day. In order to do this, the battery must be able to supply an ideal 3.3 volts to the microcontroller, light and accelerometer sensors, and memory subsystems for the entirety of the time in order to maintain proper function. In addition, the battery must have a rechargeable option in order to allow for repeated use. In order to produce this option, the subsystem must contain a charging circuit in order to recharge the battery. Finally, the battery type must be durable enough to withstand many charges and discharges without deteriorating the battery supply voltage or capacity over time.

*Subsystem Function*

The function of the battery system is to simply supply the required voltage and current to the microcontroller and other subsystems. The function of the charging circuit is to allow for the battery to be recharged overnight while the user is not active. In addition, both the systems must be able to meet the above requirements.

*Engineering Decisions*

When beginning the decision process for the battery type, the group weighed the criteria of durability, output voltage, and size-to-capacity ratio in order to find a battery that would most effectively meet the monitor’s needs. Upon doing research, polymer lithium ion batteries provide tremendous durability and capacity figures which would help minimize the size and maximize the lifetime and capacity of the battery. In addition, the various capacities of polymer lithium ion batteries came in packages that were conducive to the size and shape of the final device. The only drawback to the lithium ion battery was the output voltage. At 3.7 volts, the output voltage was outside the operating input voltage for the microcontroller. In order to make the battery compatible with the system, a linear regulator was chosen to step the voltage down to 3.3 volts. The linear regulator was chosen do to its ability to maintain high efficiency when only stepping the voltage down a few tenths of a volt. At above 90% efficiency, this will provide adequate step down without losing too large of a portion of the energy.

Next, the group looked at the options for charging the battery. Because the group had previously determined that a USB interface for data uploading was going to be employed, USB charging while the device is connected to the computer was logical. In order to do this, the group had to implement a USB connection that could both supply 5 volts to the charging circuit while also allowing for data reading. In addition, a charging circuit to charge the battery was also needed. Because USB charging is very common, highly efficient components and circuits to provide fast, effective charging have already been developed. Finally, the group decided to use a miniUSB interface as opposed to microUSB due to increased strength and durability of the connections.

*Subsystem Testing*

Due to the nature of the battery and charging subsystems, the other subsystems of the project had to be completed before much of the necessary battery testing was done. For example, in order to determine an adequate battery capacity for the final device, the current sunk from the battery by the device is needed. However, once the final components for the device were chosen, testing of the device showed an average current draw of 5mA with the occasionally spike of 23mA that occurred once per second when the sensors were sampled. Based on our chosen battery size of 850mAh, it was found that the battery could last for at minimum 36 hours, making it more than sufficient for our device.

The other low-level functions of the subsystem were tested as well. First, the group tested the linear regulator in order to ensure that a proper output of 3.3 volts was obtained. Depending on the charge left on the battery, the linear regulator produced an output voltage of 3.29 volts to 3.24 volts, which is an allowable range for the components of the device. Next, the group tested that the USB charging circuit functioned accurately. Using a USB cable connected to a computer, the group was able to charge a test polymer lithium ion battery of the same nature as the eventual component to full capacity. In order to visually see the charging process, the group also implemented an LED that illuminated when charging and turned off when the device was at capacity; this LED was kept and is implemented on the final board design.

3.7 Subsystem: Data Communication

*Subsystem Requirements*

Ultimately the raw accelerometer, light sensor, and date and time data will need to be processed and analyzed on a computer, so the ability to communicate between the monitor and the computer is critical. This means that (1) the correct data must be transferred to the computer when requested, (2) the computer must provide a basic interface to reset and write values to the microcontroller, and (3) the data received must be formatted appropriately.

*Subsystem Function*

Transferring the collected data from the activity monitor to a computer is a key requirement of the overall system. More specifically, the activity monitor must be able to send the correct data to the computer when requested, and reset and rewrite values specified by the user from the computer end. On the computer end, a basic interface is necessary to control both the reading of the raw data sent by the activity monitor and writing/setting of new parameters in the monitor. Additionally, the data received must be arranged and stored in a useable form.

*Engineering Decisions*

There are many possible ways of transferring data to the computer, including Bluetooth, Wi-Fi, USB, and UART methods. In terms power usage, space on the PC board, and cost, wireless communication methods were not desirable. Turning to wired communication methods, both USB data transfer and data transfer over a serial COM port (using a UART) were subsequently investigated.

Since a USB cable will be used to charge the battery on the device, USB initially looked promising. Additionally, many microcontrollers available come with a “USB On-The-Go” module, providing all of the hardware support necessary to implement the USB. However, the USB transfer protocol requires much more overhead on both the computer and microcontroller ends; implementing the UART as means of communication is much more robust. The solution to this problem of wanting the robustness of the UART while still using the hardware of the USB was to use a “virtual COM port.” Basically, by including a separate IC chip (FTDI’s FT232R) as an interface between the USB transfer utilized on the computer end and the UART transfer utilized on the microcontroller end, the data transfer still takes place over a USB cable, but without actually having to implement the specifics of the USB. On the computer end, Microsoft provides a useful API for connecting and transferring data through the COM port. Other software might need to be written for different platforms (Mac OS or Linux).

*Subsystem Testing*

In order to test the ability to read data from the microcontroller over the virtual COM port, arbitrary bytes of data were continuously written into the microcontroller’s UART transmit buffer (by the microcontroller itself) and shifted into the FTDI’s transmit buffer. The computer would then make read requests to the FTDI, resulting in the bytes of data to be sent out of the transmit buffer, to the computer. This data was then printed in a terminal window.

3.8 Subsystem: Command Line Software Interface

*Subsystem Requirements*

Due to the nature of the activity monitor as a data collection device, it needs to be accompanied by a simple and easy to use software tool to offload activity data stored on the device’s flash memory to a personal computer. After speaking with a colleague of our customer responsible for the customer’s software, we were advised to create a command line tool which was capable of communicating with the activity monitor and writing its data to a text file in a format easily consumable by the customer’s existing data analysis software. We also determined the software interface should provide other useful options to its user, which are described in the following section.

*Subsystem Function*

The command line software interface’s primary function is to write the activity monitor’s data to a text file. The software interface achieves this by requesting data from the device’s flash memory chip and reading the data from the device’s FTDI chip once the data is moved from flash through the microcontroller and to the transmit register of the FTDI chip. The software interface also allows the user to set the device’s real-time clock, enter a device ID for a particular activity monitor, clear the device’s flash memory contents, bring up a help menu, and close the interface. The text file produced by the command line tool is a csv file, in which each line of data contains a time stamp, a light sensor reading, and acceleration values in the x, y, and z directions.

*Engineering Decisions*

There are two main types of user interfaces on personal computers: command line interfaces and graphical user interfaces. Our group ultimately chose a command line interface due to our greater familiarity with its development, our customer’s needs, and the limited time we had to create the interface. Although a graphical user interface is usually preferable for ease of use and intuitiveness, the development time to create such a tool would have been a hindrance to our efforts to ensure the functionality of more important subsystems. In addition, we confirmed with the customer that a command line interface would be satisfactory. Understanding the command line interface to be potentially unfamiliar to non-technically inclined customers, we made it as straightforward to use as possible. The interface contains one main menu, which provides a numbered list of options. The user is instructed to enter the number corresponding to the option they would like to perform. Upon completion of the task associated with the user-entered option, the main menu reappears. The main menu also provides a help option to further aid users who may not initially be sure of how to operate the tool. When chosen on the main menu by the user, this help option displays instructions for what each option does and how to use it properly.

*Subsystem Testing*

To test the software interface, we entered each of the possible options and verified upon completion of each option that the function of each had been achieved. First, we allowed the device to collect a significant amount of data and then chose the offload option from the menu to offload a subset of the device’s flash memory contents to both the terminal screen and an output text file. Next, we chose the option to set the device’s real-time clock and entered a value for the new real-time clock value. Then we allowed the device to collect new data, offloaded the data, and verified from the timestamps from the output data that we had indeed successfully changed the time in the device’s real-time clock. A similar procedure was used to test the option for setting the device ID. After entering the device ID we set for our prototype activity monitor, we offloaded the device’s data and saw in the top line of the output file the device ID we had set on the command line amongst the other metadata. Finally, we chose the option to clear the memory contents, then chose the option to offload the device’s data, which when viewed in the output text file and in the terminal showed the device’s memory contents were empty.

3.9 Board Design and Packaging

*Subsystem Requirements*

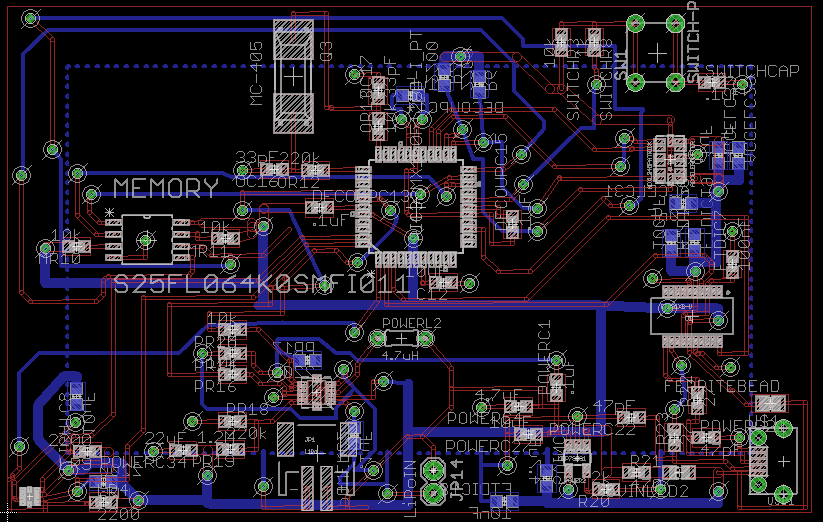
Comparable to the commercial Actigraph product available, our device had similar size and durability constraints; these constraints were met primarily through the case and board design. To begin with, the customer desired the board to be no bigger than the size of a credit card (3.370 x 2.125 in) in order for the overall product to be as small as possible. The board must also be built in such a way that there is not possibility of an electric shock to the wearer. As mentioned previously, due to the nature of the device, the case needed to be strong enough to resist moderate shock as well as being somewhat waterproof yet light in order to not be cumbersome to the user.

*Subsystem Function*

As one might expect, the board functions as the interconnection between all the previously mentioned subsystems. The case works to transform the device into an actual physical product instead of simply a piece of electronics.

*Engineering Decisions*

The process to shrink the board down to the desired size while still being spaced out enough to allow for possible hand soldering required several design iterations. The initial board design started with the thought process of routing all passive and non-microprocessor signals. The resulting design can be seen below in Figure 4:



**Figure 4**: 1st Board Design

As one can see from the figure, the result, though the board did meet the size requirement, the routing and part placement resulted in an unnecessarily complicated and ugly design. A second iteration was then designed with the opposite thought process, with the microprocessor signals and power being routed first and the passive components last. The result was a much cleaner board, both routing and layout wise, that still met the size constraints with the board measuring 2.93 x 2 in. The final board design can be seen in the appendix. As seen in the design, the sensors were placed on the board based on the side of the microcontroller their corresponding pins were on. This, in combination with the routing order, resulted in a much cleaner board.

The chosen case was based off of the need to have mounting holes to hold the board in place as well as meet the previously discussed physical requirements; the result was a case from Polycase measuring 3.29 x 2.42 x 1.00 in. In order to allow light to reach the light sensor, a light tunnel was drilled into the case and a plastic covering placed over the hole. A hole was also drilled into the side to allow USB access. The case can be seen in Figure 5:



**Figure 5.1**: USB Slot



**Figure 5.2**: Light Sensor Slot

*Subsystem Testing*

Before the parts were soldered onto the board, the routes for power and ground were tested. Using a bench power supply, it was determined that the routes between power and ground were not shorted and worked as designed. Once the components were soldered onto the board, shorts were checked for between the pins and among the components through the use of an ohmmeter. Modifications that resulted as a consequence of these tests will be discussed in Section 6: To Market Design Changes.

For the case, a test board containing an USB port and the light sensor were placed inside the package. It was verified that the USB could be successfully plugged in through the slot and that the light sensor was in line with the light tunnel. However, due to the fragility of the actual board due to the board changes implemented, the ability of the sensor to read light values through the hole was not tested. This is an aspect that would need to be tested with a future version of the board.

# **4 System Integration Testing**

4.1 Description of Testing

In order to test the various subsystems as a whole, they were first integrated using the provided prototyping board. Although the provided prototyping board ran on a PIC32MX795 microcontroller, the embedded programming required for the final microcontroller compared to this microcontroller was determined to be similar enough to provide adequate feedback. This round of testing was done by building the overall circuit from the microcontroller up. This was done by adding one subsystem at a time in order to be able to easily determine problems if they arise. The microcontroller was first implemented with the accelerometer to assure proper microcontroller function as well as the accelerometer subsystem. Then the light sensor subsystem was added to see if the microcontroller could function properly with two devices on the same set of I2C pins. Next, the memory chip was added and was tested by writing and reading actual values obtained from the light sensor and accelerometer.

Upon successful testing of these subsystems, the group then moved onto to testing the subsystems not directly involved in data retrieval and storage. The battery was then attached to the board through the power circuit. This was done in order to assure proper function of the linear regulator and to assure appropriate voltage was being supplied to the rest of the board. Despite the linear regulator producing an actual voltage of 3.28V rather than the nominal 3.3V, the group determined that the function of the regulator was functional and appropriate for proper board function. Finally, the group implemented the data communication subsystem to see if that data was able to be off-loaded to a computer in the proper manner and able to be erased in order for additional data to be stored. The COM port program was implemented and the program functioned as expected aside from a slight data formatting concern in the final text file. This was remedied with a slight change to the code.

With full functionality proven on the kit board, the group then constructed the first prototype on the designed PC board. Because the board was to be soldered with a solder-paste screen, all of the parts on the board needed to be added at the same time. When all the parts were soldered to the board, extensive board testing was then conducted in order to assure proper soldering and that that the board design was functional. Testing including checking all Vdd and ground points to assure no shorts as well as checking all the integrated circuits to assure there were shorts between the pins.

Upon completion of the board, the group then began to download code to the microcontroller subsystem by subsystem. This allowed the group to determine that every IC was added to the board correctly and that the board design was functional. In this round of testing, the group ran into several soldering and design problems. To start, there were several components that needed to be removed and resoldered. Although this was a setback, it was an error caused by inexperience ran than poor design. That being said, the group was able to determine several errors on the board design. First the ‘SDO’ pin on the accelerometer was not tied high as needed. The ‘EN’ pin on the linear regulator was also incorrectly wired. The ‘EN’ allows the user to turn the regulator on and off in order to save power when the battery is being charged. Because of this, the group connected this pin to an I/O pin on the controller in order to take advantage of this. However, because the microcontroller draws its power from the regulator, it could not set the ‘EN’ pin high to turn on the regulator until it had power first. This was corrected in the board design for the next round of prototyping. Finally, a capacitor used to ensure proper function of the clear button was placed slightly out of position.

With each individual subsystem completed and working on the board, the code required for overall functionality was implemented. At this step, the group ran into several problems switching the code from functionality with the previous microcontroller to the new one. Although the several changes were small, they were difficult to find within the many lines of code and proved to be major setbacks. The two major problems were properly setting the analog I/O pins to digital only as well as switching the real time clock output off of the ‘SDA’ pin on the inter-integrated circuit line. After finding and fixing these errors in the code, the group was able to show full functionality of the board.

4.2 Requirements

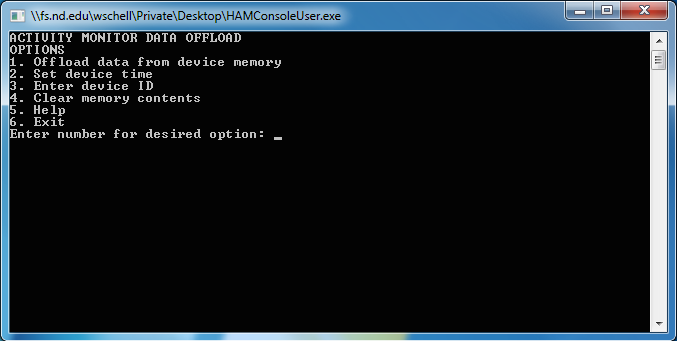
Based on the given overall system requirements, the previously discussed testing was able to prove that the prototype fulfilled almost all the requirements. The group was able to show functional accelerometer and UV sensor data being read, stored, and dumped to a text file on a computer. This fulfills the basic desired requirements. In addition, the board was designed to be smaller than a credit card as required.

That being said, the group did not test the final prototype for a few overall requirements to due to limited time constraints. These were for total battery length and amount of data that could be stored before a data dump was required. Because the group did not have several days to test the life of the battery and how much data could be stored, the group was forced to look at the results of the individual subsystem testing to help determine what the final prototype would produce. Because the individual subsystems in these cases were tested in conjunction with all the other subsystems, using this comparison is reasonable. The only difference compared to the previous tests and the final was a different microcontroller. This will not affect the total amount of data stored and will have a very small effect on the current draw compared to the final system.

# **5 User Manual**

5.1 Installation

The data offload program (HAMConsoleUser.exe) is a standalone executable. This program is designed mainly to extract the data stored on the activity monitor device and to write the data to a text file, along with a few ancillary features. There is no installation required, just run the application and a text interface should appear as shown in Figure 6.



**Figure 6**: The Data Offload Program on Startup

5.2 Setup

1) Plugging in the battery:

When the battery is plugged in, the activity monitor is collecting data (there is no On/Off switch). When the battery dies, the data is preserved in non-volatile Flash memory.

2) Plugging in the USB:

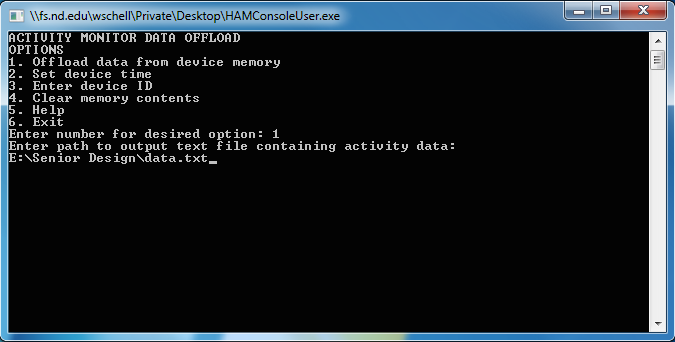
To offload the data from the device, simply plug in a micro-USB connector into the device and into the computer. Upon detecting that it is connected into the computer, the activity monitor stops recording data. Additionally, an LED on the device lights up when it is connected to power through the USB. This light indicates that the device is charging. When the battery is fully charger, the LED turns off, even if the device is connected to the computer.

3) Options:

Upon running the application, a text menu appears, listing the possible options to be performed.

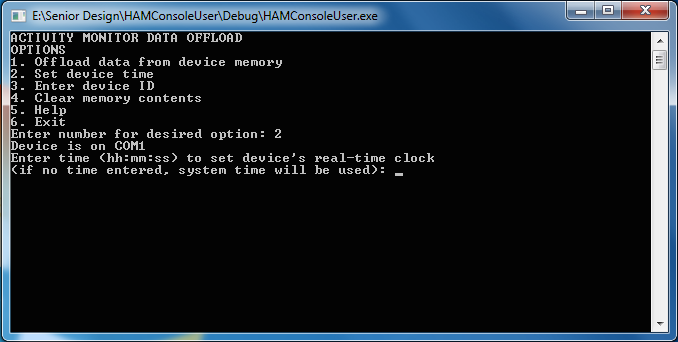
- Offload Data From Device Memory:

This option performs the main function of the software, offloading the data and storing the data in a text file. After selecting this option (1), the user is prompted for the desired path to the output text file (Figure 7). If the entered path is valid, the program will establish a connection with the activity monitor device (Device should appear as a Serial COM port). As data is transferred and subsequently stored in the output file, the data is displayed in the window.

**Figure 7**: Entering the desired output file

- Set Device Time:

On reset or loss battery life, the clock time on the device is set to 00:00:00 (hh:mm:ss). This time can be set in the program by selecting option 2 (Figure 8). If the entered value is not a legal 24-hour time value (e.g. 37:72:66), the user is prompted again for the time.



**Figure 8**: Setting the real-time clock

-Enter Device Time:

Multiple devices will most likely be used to collect data from multiple subjects. In order to distinguish the data from each participant, a device ID can be set.

-Clear memory contents:

Before the device can be sent out again to collect more data, the memory on the device needs to be erased. Option 4 in the menu performs this task. Note: erasing this memory may take upwards of 30 seconds.

5.3 Troubleshooting/Correct Functionality

The data of the output text file are as follows:

Hours, Minutes, Seconds, *blank value*, UV Index, X, Y, Z

Where X, Y, X indicate the directional components of the accelerometer data.

Blank or erased data points can be seen as:

45, 85, 85, -1, -1, -0.07800, -0.07800, -0.07800

This shows that either no data was recorded or that length of time the device was recording was not enough to reach this amount of data.

When connected, sometimes the device does not sync correctly with the computer. This can be seen when a desired option produces only a blinking cursor. One possible solution to this problem is to first unplug the device from the USB connection, then disconnect and reconnect the battery. Close and restart the program and continue normally.

# **6 To Market Design Changes**

Due to time constraints, there were several developments and improvements that were not implemented into the project. While none of these changes would have affected the overall functionality of the device, given enough time to enact, they would have resulted in a more polished, market ready product.

6.1 Packaging and Board Size

Due to the nature of the device and as required by the customer, the size of the monitor needs to be as small as possible so as to not hinder the user. While the minimum size requirement of a board smaller than a credit card was met, with the functionality of the board now developed and tested, the size of the board could be shrunk further, resulting in an overall smaller package.

The actual package of the board was only roughly prototyped as seen in Section 3.8. It was initially thought that the best approach for a package was to purchase a predesigned one from a supplier. However, after receiving the actual case and having to physical modify a board in order to fit it, it is recommended that a case be designed and printed through 3D modeling software. This will allow the sizing of the case to match the features of the board, more credit card like as opposed to blocky, thus producing a cleaner and more professional looking product. In addition, by designing the case by hand, the appropriate holes needed for the UV sensor and USB port can be added without having to physical modify an existing case, a process that was found to be rather imprecise and produced unsightly results.

6.2 Parts and Device Cost

While the price requirement of the product was met, it is believed that this price could be reduced even further. The subsystem parts selected, such as the accelerometer and UV sensor, were selected based on the need to secure functionality first with the price being considered second. As such, there are more than likely cheaper solutions that could be implemented into the device that are more “bare-bones” than the ones used. The UV sensor for example, has a proximity mode built in, a feature that for the activity monitor is unneeded and unused. So with this taken in mind, along with the idea that this product would be produced on a mass scale, the price of creating the activity monitor could be dropped significantly.

6.3 UV Sensor Calibration and Data Management

Though the board is able to both gather UV data and offload the gathered data in a text file via USB to a computer, there is calibration of both processes that needs to occur before the device would truly be market ready. The calibration of the UV sensor would depend on the medium that the sensor would be receiving light, which in turn would depend on the material covering the slit in the case, ideally clear plastic. Because the case wasn’t able to be manufactured, this calibration was never done and integrated into the final device. This tuning is necessary in order to receive informative UV values from the sensor as opposed to the unscaled numbers that are currently being received.

Currently, the device only offloads a portion of the gathered data. This choice was made during the testing of the subsystem in order to analyze the functionality of the device quicker, and due to time constraints in the project, was never changed in the final code. Similarly, the device is set to erase the memory once the USB is unplugged; this would be better implemented with an erase command in the command line interface. In addition, the setting of the device ID and time for the real time clock is untested, and while not detrimental to the performance of the device, would need to be implemented in a market version in order to help keep track of when and by who the data is being collected. The identification of the com port the device is connected to also would need some improvement before heading to market; currently, if the monitor is the only device connected, it will work, but the connection of an another USB data device such as a flash device results in confusion in the software. However, it is believed that these would be relatively simple fixes and could easily be implemented by a future design team.

# **7 Conclusion**

The HAM team is very satisfied with the effort and results achieved during the course of this project. The device produced successfully met all the constraints given to the team at the beginning of the semester: gather activity (acceleration) and UV data at a specified rate, store this data on the device until available to offload, interface with developed software and be able to offload the data in a useable manner, last for upwards of 36 hours while running off a battery, and be smaller than the size of a credit card. Though there are some areas of the product that need to be more fully developed, such as the packaging and specific areas of the code, the device itself is functional and as such, can serve as a strong base for future teams who wish to continue the project. Overall, the HAM team is proud to have taken this idea through the entire engineering design process and come out the other side with a working prototype that satisfied the constraints of our client.

# **8 Appendices**

8.1 Subsystem Data Sheets

UV Sensor: <https://www.silabs.com/Support%20Documents/TechnicalDocs/Si1145-46-47.pdf>

Accelerometer: <https://www.sparkfun.com/datasheets/Sensors/Accelerometer/ADXL345.pdf>

FTDI: <http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT231X.pdf>

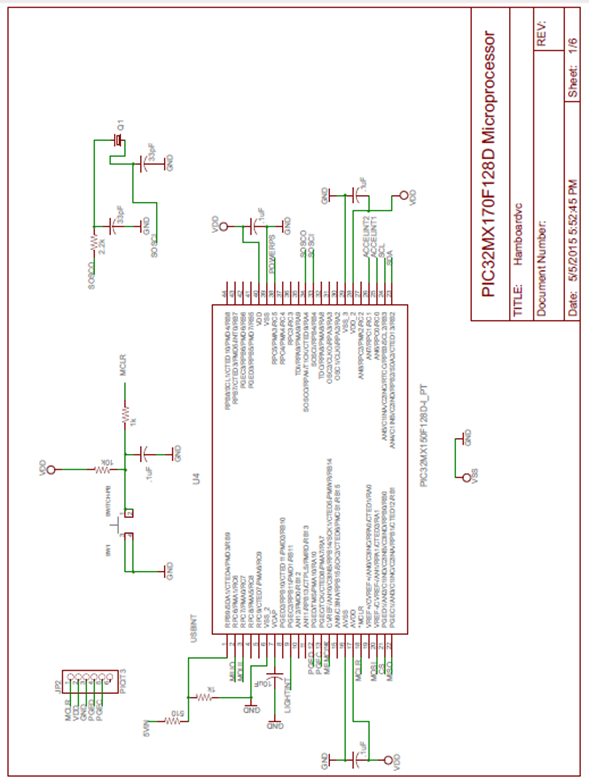
Microprocessor: <http://ww1.microchip.com/downloads/en/DeviceDoc/60001168F.pdf>

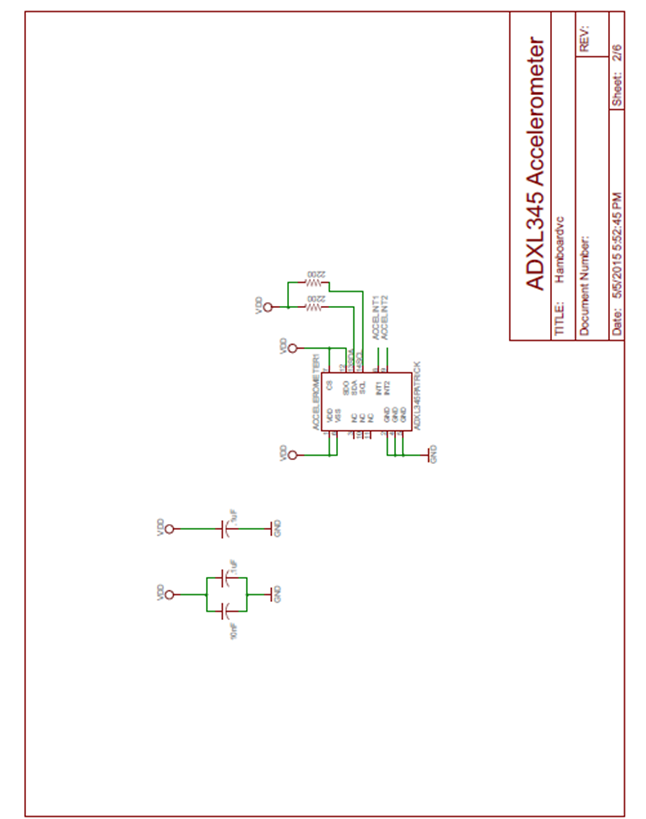
Memory: <https://www.spansion.com/Support/Datasheets/S25FL127S_00.pdf>

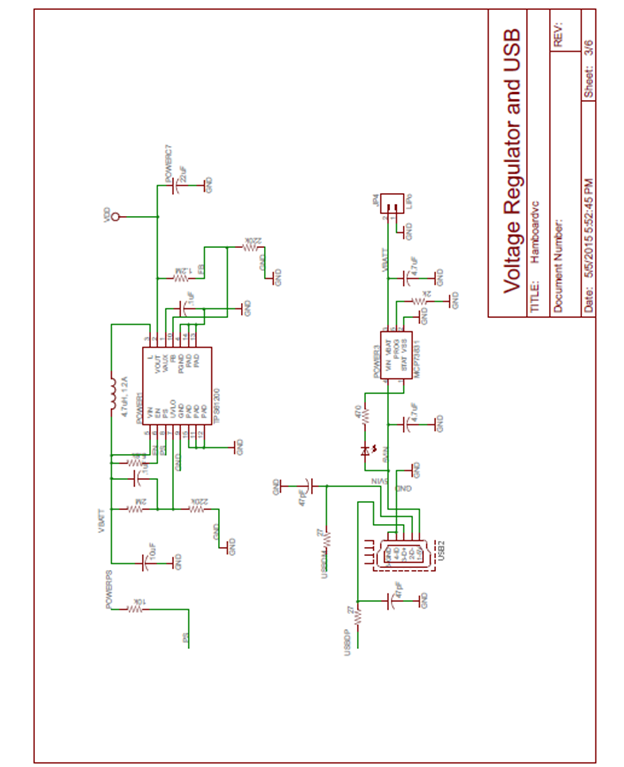
Voltage Regulator: <http://www.ti.com/lit/ds/symlink/tps61202.pdf>

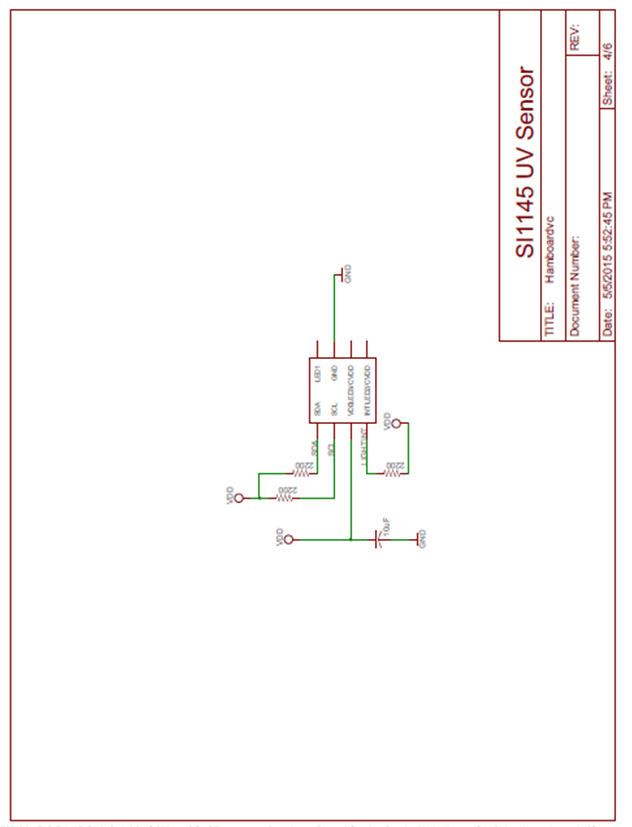
Battery Charger: <http://ww1.microchip.com/downloads/en/DeviceDoc/20001984g.pdf>

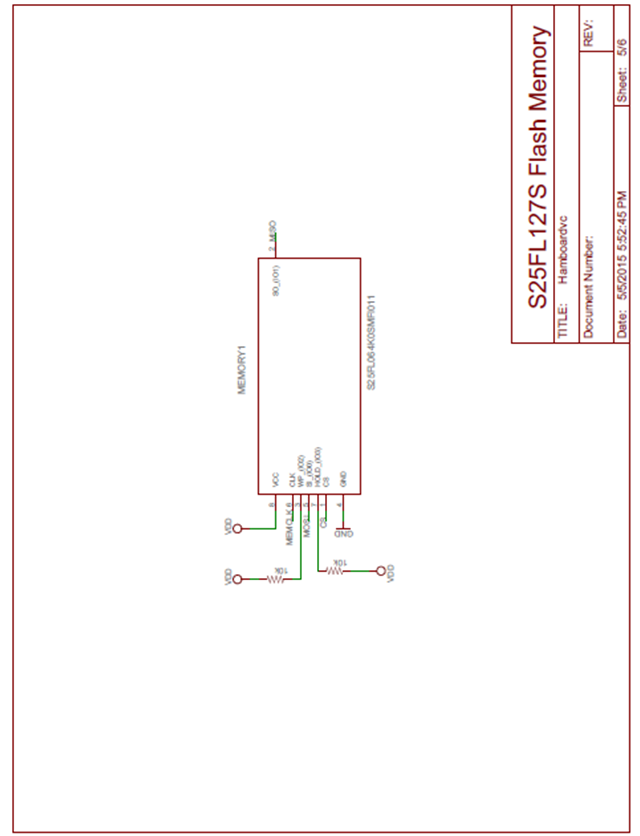
8.2 Board Schematics

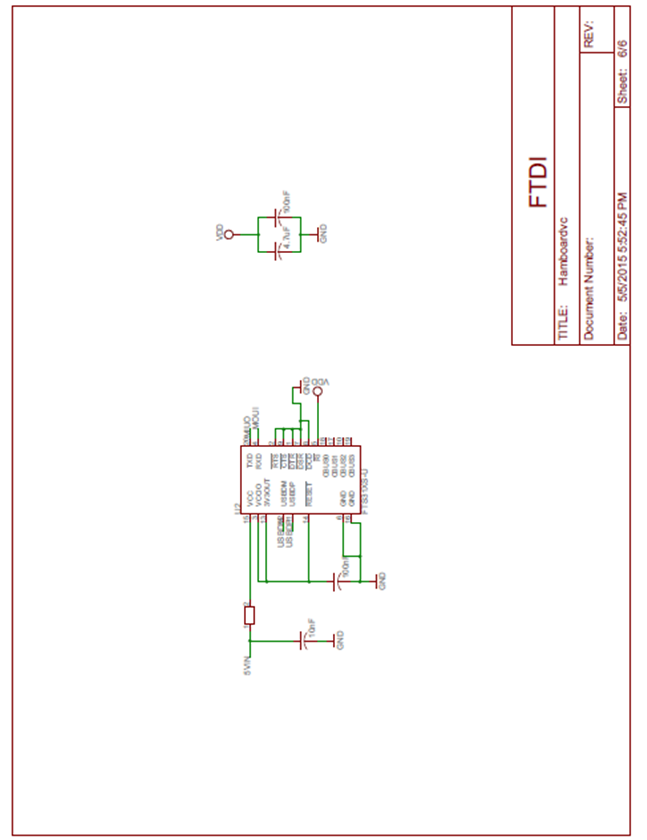


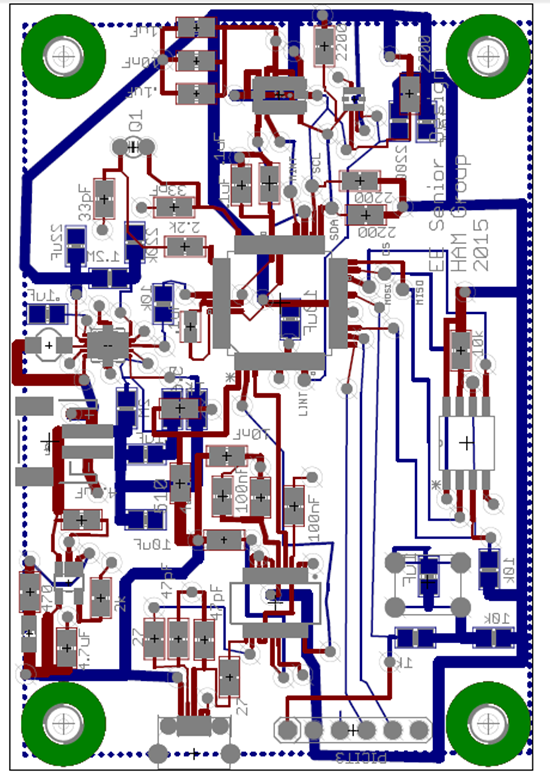












8.3 MPLABX Code

As the code listing for this project is extensive, all code can be accessed on the HAM webpage.