ECE 480

USB-powered Micro-Reader for Self-Powered Health and Usage Monitoring

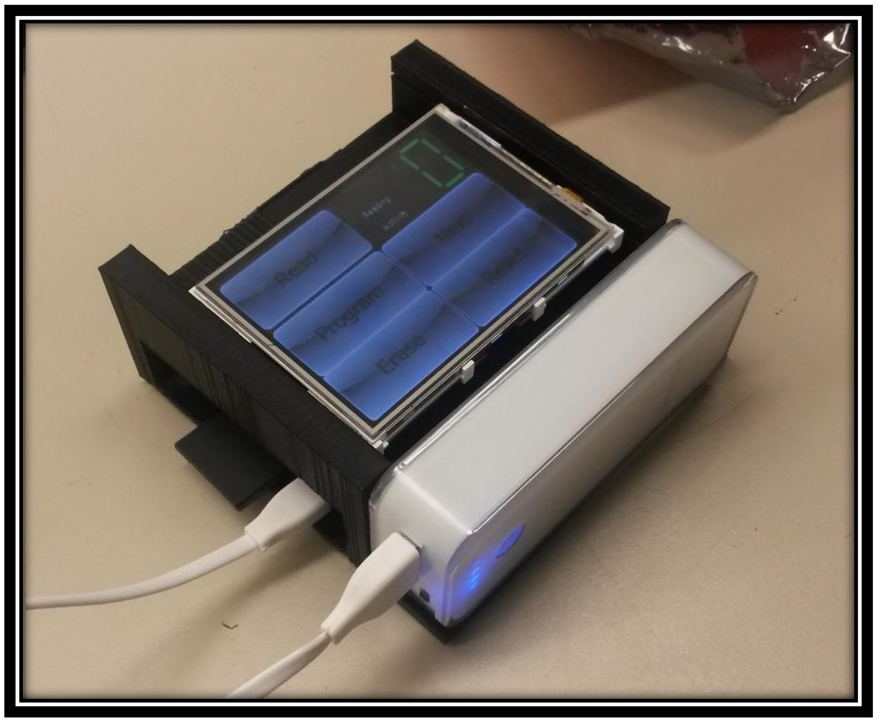
Sponsored by: MSU Technologies & National Science Foundation

Final Report

12/4/13

Design Team 1

Brett Johnson, Evan Gardetto, Ron Razalan, Thamer Alshuaibi, Yuval Levental



Executive Summary

This document describes the Design Team 1 Capstone engineering design project from inception to completion. The customer for the team is MSU Technologies and the National Science Foundation. The design team's challenge was to develop an alternative method of interfacing with a piezo floating-gate in order to extract sensor data. This meant engineering a working prototype by the completion of the semester.

Over the course of seven years, engineers have designed a piezo-powered strain sensor that measures and records impacts on football players helmets. The current method for data extraction utilizes a National Instruments data acquisition device and requires a computer running MATLAB to acquire the sensor's information. The customer requests a more compact, user friendly method to communicate with the sensor and interface with it using various commands.

The team designed a portable, cost-effective micro-reader that can successfully communicate with the stress sensors and provide reliable data during operation in the field. The chosen design is a battery-powered unit based around the Raspberry Pi platform. It is programmed to interact with a touchscreen display peripheral, with a simple to use graphical user interface that can perform all necessary functions to interact with the sensor modules per customer requirements.

The micro-reader project successfully provided the customer with a simple plug-and-play method to retrieve data off the sensors, meeting and exceeding all stated requirements within the given budget.

Acknowledgments

The project could not have been completed without the support of Michigan State University faculty members: Dr. Shantanu Chakrabartty, Dr. Fathi Salem, as well as graduate student Kenji Aono.

Dr. Chakrabartty is an associate professor at MSU within the Electrical and Computer Engineering Department and also Director at the Adaptive Integrated Microsystems Laboratory (AIMlab). He is one of the co-inventors of the piezoelectric strain sensor device and met with the team directly to outline the project and provide functional requirements. In addition, he was the team's first contact for expertise on the sensors operation and was instrumental in successful completion of the project.

Dr. Salem is a Professor at MSU in the Electrical and Computer Engineering Department. He acted as the team's facilitator, meeting on a weekly basis to stay up to date with the projects progress. Through Dr. Salem's guidance, the project and classroom assignments were completed on time and to a high standard.

MSU graduate student Kenji Aono consulted with the team and provided information on methods for interfacing with the sensor. His knowledge of the piezoelectric sensors internal operation was critical to the technical presentation and troubleshooting with the prototype.

Finally, 4D Systems technical Supporter, Danny Lamaroza, provided assistance to the team by answering questions pertaining the 4D Systems touch screen. He also supported his answers with multiple data sheets that include illustration of various examples that enhanced our understanding of the device.

Table of Contents

Chapter 1 - Introduction and Background

1.1 Introduction

In response to demand from the capital goods industry, MSU Technologies has researched methods to monitor and record stress levels applied to an infrastructure over the duration of an its lifespan. Access to this data allows engineers to determine the current health of a building, road, or other structure. With the help of piezoelectric devices, self-powered sensors can be used to determine strain on any given structure. Using this information, the condition of the infrastructure can be analyzed, and proper maintenance can take place. Applications for this type of device are being expanded further than just civil infrastructure; this type of technology can be implanted into high impact zones on sports athletes' equipment. These sensors can monitor the overall damage a player has suffered, and can predict if a player has suffered any physical injury. With the advent of micro-sensor technology, sensors can be deployed in the helmet of football players and any head injury can be monitored. Concussions in sports have become a serious area of concern over the last few years, and having the ability to read impact data in such a small device would prove to be beneficial.

The sensor data is only useful to researchers if it can be easily extracted and analyzed, so a method for retrieving the data was developed by the AIM lab at Michigan State University. The AIM lab developed a method to be able to retrieve the data from the sensor; however it is bulky, impractical, and inefficient. The scope of this project is to improve upon the current methods of retrieving the sensor data, by creating a standalone device that requires fewer components and is more practical for real world implementation. The design team developed a new device that maintains all functionality, but reduces the overall size, operation, and cost.

1.2 Background

Over the last few years, Michigan State University has developed a self-powered Piezoelectric Floating Gate sensor that can be used to monitor the health of civil and mechanical infrastructures. The sensor utilizes a piezoelectric membrane which flexes slightly whenever a force is applied to it, creating a small current that powers the sensor as well as provides data to the sensor. A higher level overview of the sensor's operation can be found in appendix 9. This sensor requires approximately 1uW of power through the Piezoelectric device in order to operate, and can therefore be used in a wide variety of applications. The MSU athletic department has recently asked for help from MSU Technologies regarding the monitoring of head injuries sustained by football players. A new application for this sensor is being implemented and tested in a football helmet, to monitor the impact of a helmet collision, which can easily exceed acceleration in excess of 100 g. Figure 1 displays a prototype of the self-powered sensor next to a penny for size relation.

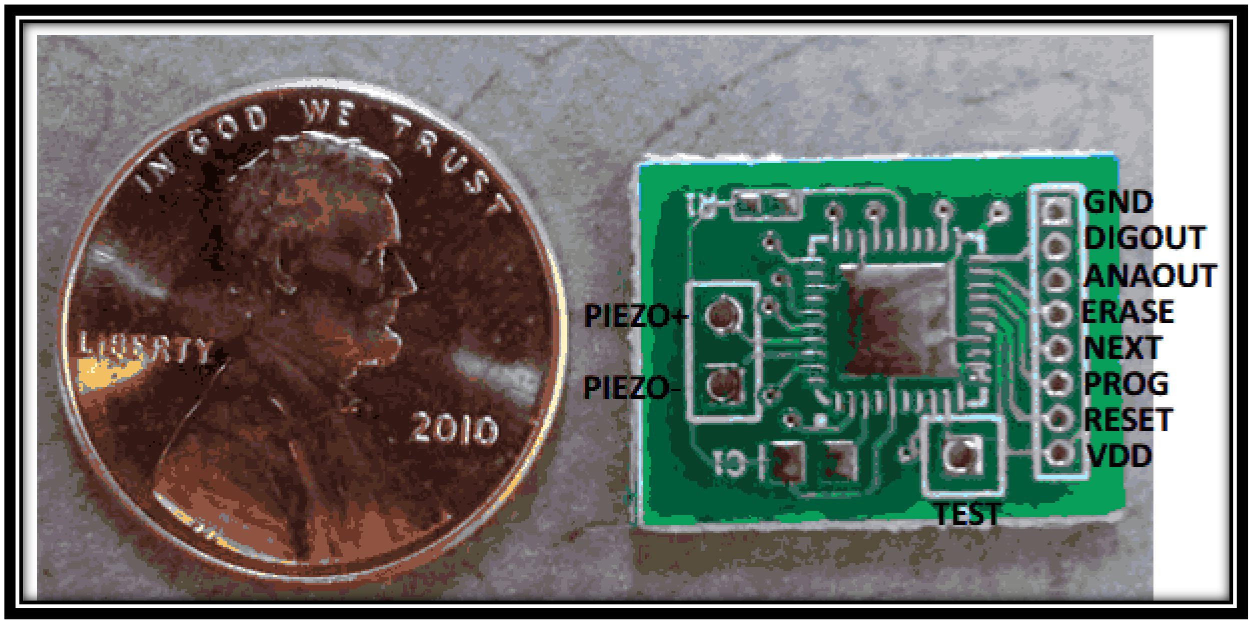


Figure 1.1: PFG Sensor prototype

Chapter 2 - Exploring Solution Space and Selecting a Scientific Approach

2.1 Problem Definition and Scope

The device was required to have the ability to read the data provided by PFG sensors. It is also required to be able to interface with the sensor and to allow users to input commands and save the data to an output file. The current method of retrieving data from the PFG sensor is impractical and cumbersome, as it requires a PC, a serial interface, as well as several long wires. In order to improve upon this design, the team has been tasked with creating a portable, more practical device that is self-sustained, and can perform all required functionality with the PFG Sensor. The device was required to be portable and durable to be utilized on various environmental conditions, and to be held in place for users' satisfaction.

2.2 Critical Customer Requirements

In order to ensure that the new design meets the needs of the customer, numerous requirements have been established and must be executed in order for the project to be considered successful. The critical requirements for the sensor reader are:

1. Communicate with the pins on the sensor to initiate sensor functions
2. Portable design that can easily be transported for usage at different locations
3. Log and organize retrieved data into universal format for future analysis
4. Voltage applied to sensor must be regulated between 1.8-2.5 V to prevent damage
5. Develop a plug-and-play connector that can connect to the sensor pins without compromising the integrity of the I/O pins on the chip

In order to determine which critical customer requirements were the most important to the design process, a House of Quality diagram was created. This House of Quality table can be located in Appendix 4 labeled at Figure. The table introduces more understanding to determine what requirements could be more important than others, and creates a path for the designing team that split primary and secondary goals.

2.3 Proposed Conceptual Designs

The team explored multiple solutions and evaluated numerous proposed designs. The core of the device needed to have an embedded computer, so the team considered two practical methods, the Arduino platform and the Raspberry Pi.

1) Arduino Uno/PC Interface

Description: Use Arduino Uno Microcontroller board as a gateway between PFG and Host PC. A simple computer program created by design team would send commands to Arduino via USB, and then the Arduino would output corresponding signals to PFG to initiate various actions. Assuming the user already has a computer available, this a lower cost solution than the Raspberry Pi method.

Required Components:

- Arduino Uno
- Laptop
- USB connection between Uno and PC, providing power and data lines
- Customized 8-pin connection between UNO and PFG Sensor.

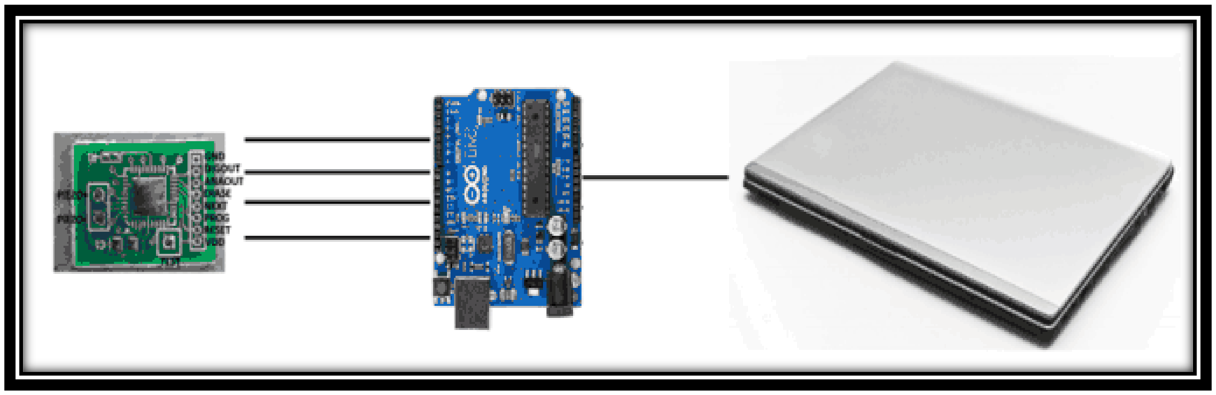


Figure 2.1: Arduino Conceptual Design

2) Raspberry Pi Linux-based Interface

Description: Construct a portable Raspberry Pi computer running a Linux OS, complete with battery power and a screen display with a simple graphical user interface, all contained within a ruggedized enclosure. The Raspberry Pi connects directly to PFG sensors via custom 8 pin connector. The Raspberry Pi is a more powerful and capable device than the Arduino Uno but costs about $20 more.

Required Components:

- Raspberry Pi
- TFT LCD Display (4-7") (Analog or HDMI)
- Battery Power
- Voltage Regulators
- Connector for PFG sensor
- Enclosure for components

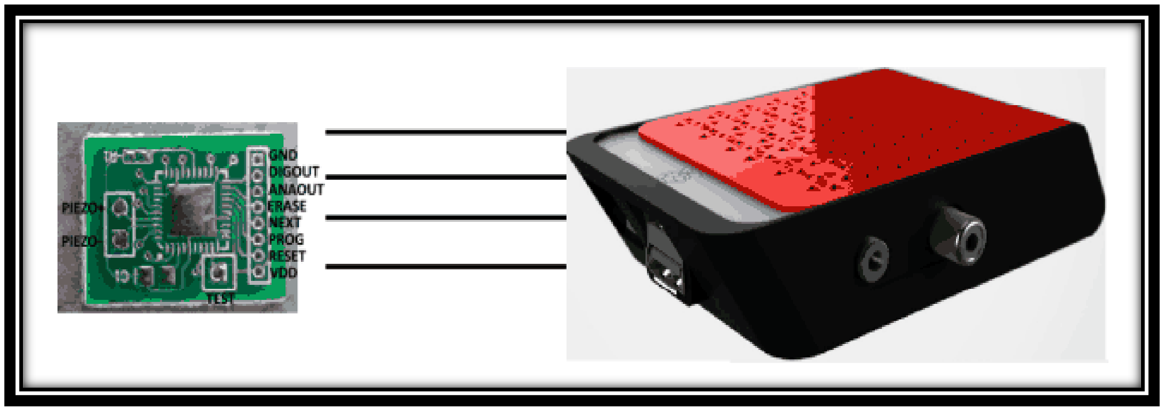


Figure 2.2: Raspberry Pi Conceptual Design

The Arduino Uno with a PC is a bulky combination and thus has an impractical implementation in the work environment. While the Arduino UNO/PC would be a simpler, cost-effective design that would accomplish all of the requirements, the team would prefer to develop a stand-alone device that provides extreme portability for the user and will be easier to use than a comparable PC program despite the cost and complexity. The easier to use design (simple touch screen interface) coupled with an all-in-one form factor, shows that a custom Raspberry Pi mini-computer is the most practical solution for reading a sensor that will be deployed out in the field.

2.4 Solution Description

The comparison above presented the reasons behind choosing the Raspberry Pi instead of other microcontrollers where the Raspberry Pi could be used as a microcomputer and that allows having a direct interaction with the chip itself rather than using another source for inputting commands. The Raspberry Pi can be programmed in C or Python, which are universal languages that can be easily translated to another platform.

The next task was picking a 4D Systems 3.2" TFT LCD touch-screen for this device. Although multiple screens were available in the market, the 4D screen was designed specifically for the Raspberry Pi with many pre-built functions to be utilized and displayed. The screen included a 4D adapter and 5-pin female-female cable that allows obtaining a simpler connection between the Raspberry Pi and the screen.

After choosing the screen, the team went to discover the options for power supplying and searched for a portable battery with a sufficient power to supply both the screen and the Raspberry Pi. A 5 V, 1 A, 4400 mAh battery designed by Adafruit Industries was the right solution since the Raspberry Pi requires only 700 mA at 5 V; the screen could be powered directly from the Raspberry Pi's 5 V pin. The large current provided from the battery was an advantage for the device because it ensured that the Raspberry Pi can have enough current draw at peak performance as well as maintaining power for the touch-screen. The team also purchased an 8GB micro-memory card to save and load the program on the Raspberry Pi and the touch-screen.

Since the Raspberry Pi does not include a built-in Wi-Fi feature, a Wi-Fi USB dongle was added to allow for interactions between a computer and the device. That provided more value to the device because it became easier to program it, as in our case, and made file interaction with the Raspberry Pi much simpler by having a Wi-Fi interface instead of an SD memory card interface.

After solving the device issues, the team faced another obstacle where the PFG sensor requires a voltage between 1.8 V-2.5 V in order to power it and the minimum output from the Raspberry Pi is 3.3 V. A voltage regulator was found that can pull-down the voltage to the required range permitting it to interface with the sensor, through the Raspberry Pi, without damaging it.

To have the device to be portable and durable, the team decided to design an enclosure to hold up the Raspberry Pi, Touch Screen, and a battery. A customized case was created using Siemens NX 8.5 computer aided design software and printed via a 3D printer.

2.5 Initial Budget

The total cost of the design project was significant for its success. It is a prime factor to determine whether or not the device can be marketed. The initial estimated costs of the parts are shown in table 2.1 below.

Part Number	Description	Quantity	Place of Purchase	Total Cost Including Shipping (USD)
04X5042	Raspberry Pi Single Board Module B	2	Newark	$80.00
LCD-11743	4D Systems 3.2” TFT LCD touch-screen	1	Karlsson Robotics	$76.27
967-ICG12006A007V03R	DC/DC Converters 4.5-14Vin 6A 0.7-5.5 Vout NegLogic	2	Mouser	$39.47
uUSB-PA5	micro-USB Programming Adapter	1	Mouser	$27.34

UUSBHAUB3RA	StarTech USB A to MicroUSB B Cable	2	Newegg	$8.98
1565-ADA	USB Battery Pack 4400mAh – 5 V @ 1A	1	ADAfruit	$29.95
1294-ADA	SD/MicroSD Memory Card (8 GB)	1	ADAfruit	$17.24
DE-SWADJ	10W Step down adjustable switching regulator	1	Dimension Engineering	$16.25
AE08A-5-ND	5 ft - 28 AWG ribbon cable	1	Digi-Key	$4.31
S9066-ND	10 pin - female header connector	1	Digi-Key	$2.76
EW-7811Un	Edimax EW-7811Un 150 Mbps Wireless 11n Nano Size USB	1	Amazon	$9.99
S8V3A	Adjustable Step-Up/Step-Down Voltage Regulator	1	Pololu Robotics and Electronics	$14.90
N/A	Enclosure 3D Printing	2	DECS Office	$22.00
	Total			$349.46/$500.00

Table 2.1: Justifications of the parts and costs

The table shows that the initial purchased components were below our maximum budget, which allowed us to have multiple parts for separate experiments and determine which component can provide better results.[1]

2.6 Gantt Chart and tasks distribution

In order to promote individual responsibility and improve the development of the product, specific roles were distributed among the team members. The Gantt chart below details the goals and milestones set by the team. Objectives are either to be completed by the group assigned to individual engineers. This ensured that the team finished objectives on time and team members are held accountable for their responsibilities.

The entirety of the project took place over a 3 month period, from September until December of 2013. The major product development segments were distributed amongst the team as detailed below:

- Ron Razalan - Software Development, Graphical User Interface
- Evan Gardetto - Enclosure Design and Verification
- Thamer Alshuaibi - Enclosure Design and Product Validation
- Brett Johnson - Connector Design and Fabricator
- Yuval Levental - Power Distribution

Important milestones in the development of the sensor are listed below:

- Distribute power appropriately across devices
- Complete graphical user interface
- Build custom interface connector
- Successfully send pulses to proper pins on sensor
- Finish enclosure
- Retrieve data and store appropriately

[1] The utilized parts were all chosen based on particular specifications to assure safety and no damages occur to the Raspberry Pi and PFG Sensor

Chapter 3 - Technical Work Performed

3.1 - Graphical User Interface

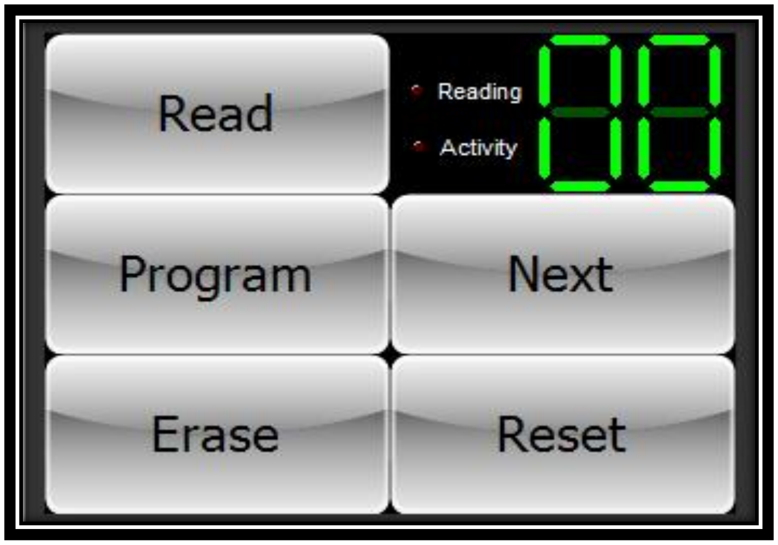


Figure 3.1: The buttoned displayed on 4D Systems

The graphical user interface provides all necessary information that a user will need to interface with the sensor. The four bottom buttons instantiate the four input commands to the sensor. Each of the four input commands (Program, Next, Erase, and Reset) have a 10 ms pulse width with a voltage level of Vdd (2V). The Program button sends a pulse to the Program pin on the sensor which initiates Program on the sensor. Program decreases the time between pulses in the digital output pin effectively increasing the frequency; this function is used for calibration of the sensor. The Next command initiates the Next function on the sensor. Next selects the next channel to output on the sensor; after the last channel is reached it, the first channel is automatically selected. The Erase command erases all channels on the sensor. Reset automatically selects channel 1 to start outputting the data. The data on each of the channels on the sensor is represented in two ways: analog and digital. Analog has linear voltage value which corresponds to the impact sustained by the sensor and has a voltage range of [0 V - Vdd (2V)]. The digital output value is represented in a series of pulses (pulse train) in which the frequency of pulses directly correlates to the magnitude of the forces sustained by the sensor. Upon connecting Vdd, the sensor constantly outputs on both the digital and analog pins. The read button above is an external program and is solely connected to the digital output pin on the sensor. Upon pressing the button, the Raspberry Pi will take a user specified sample of the digital output (hardcoded for 1 second at a 1 MHz sample rate) and write the value of the pin in terms of 1s and 0s to a file. The output file generated will have a string of 1s and 0s which can be analyzed/reproduced with Matlab. The output file represents the original signal (square wave with varying pulse widths and periods) from the digital output. The status LEDs serve as current activity on the sensor and when lit, disables interaction with the sensor to prevent sending multiple commands to the sensor. The channel indicator on the upper right hand corner displays current channel on the sensor and changes depending on input (Next, Reset, and initial power-on). The entire graphical user interface was designed in 4D Systems Workshop4 IDE and can be replicated/modified for future releases.

3.2 - Enclosure Design

The sensor reader consists of three major components, the touchscreen display, Raspberry Pi, and the battery pack. These are connected together to form the sensor reader but they need to be organized into a case. An enclosure will protect the components and make the reader more practical for real world usage. Designing an enclosure using computer aided design software and then printing the design using the available Makerbot 3D printers, provides a highly customizable, low-cost solution.

Before a case can be designed using the software, the components physical and electrical characteristics should be examined. A CEN-TECH digital caliper was used to measure the dimensions of the components. The diagrams below summarize the attributes of the major sensor reader components.

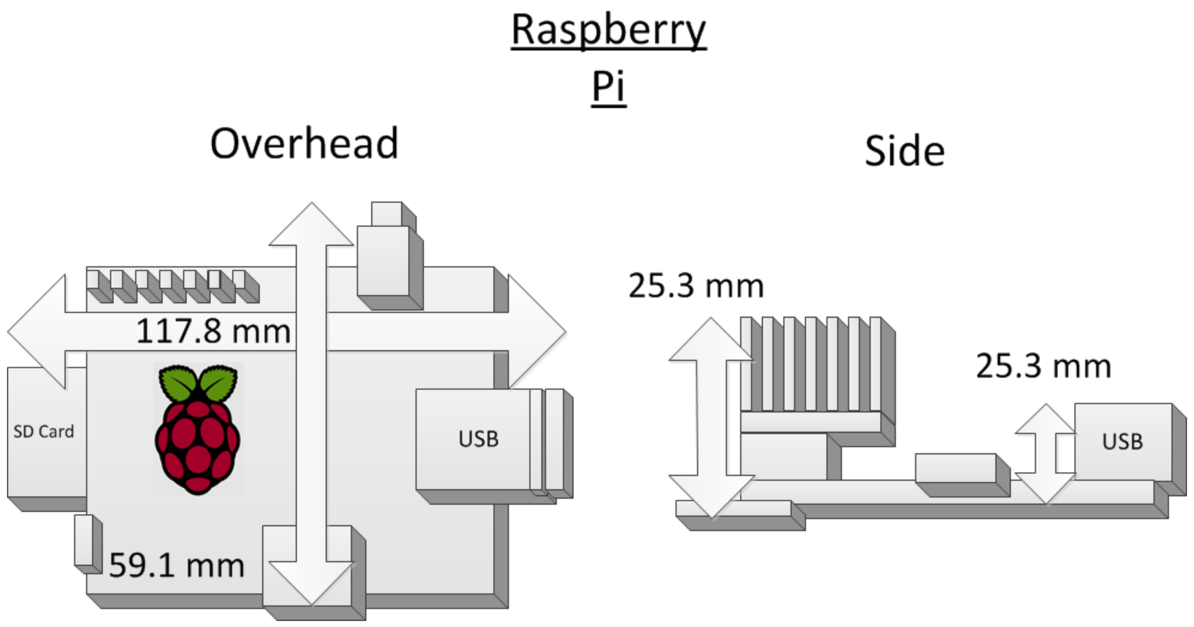


Figure 3.2: Raspberry Pi Dimensions

The Raspberry Pi only takes up about a 12cm x 6 cm area at its maximum length and width. The overhead view illustrates the general layout of the board and major components that affect design decisions for the prototype enclosure. The side view of the Raspberry Pi shows how the pins require significant vertical room. The pins connect to the proprietary LCD touchscreen display that handles user inputs.

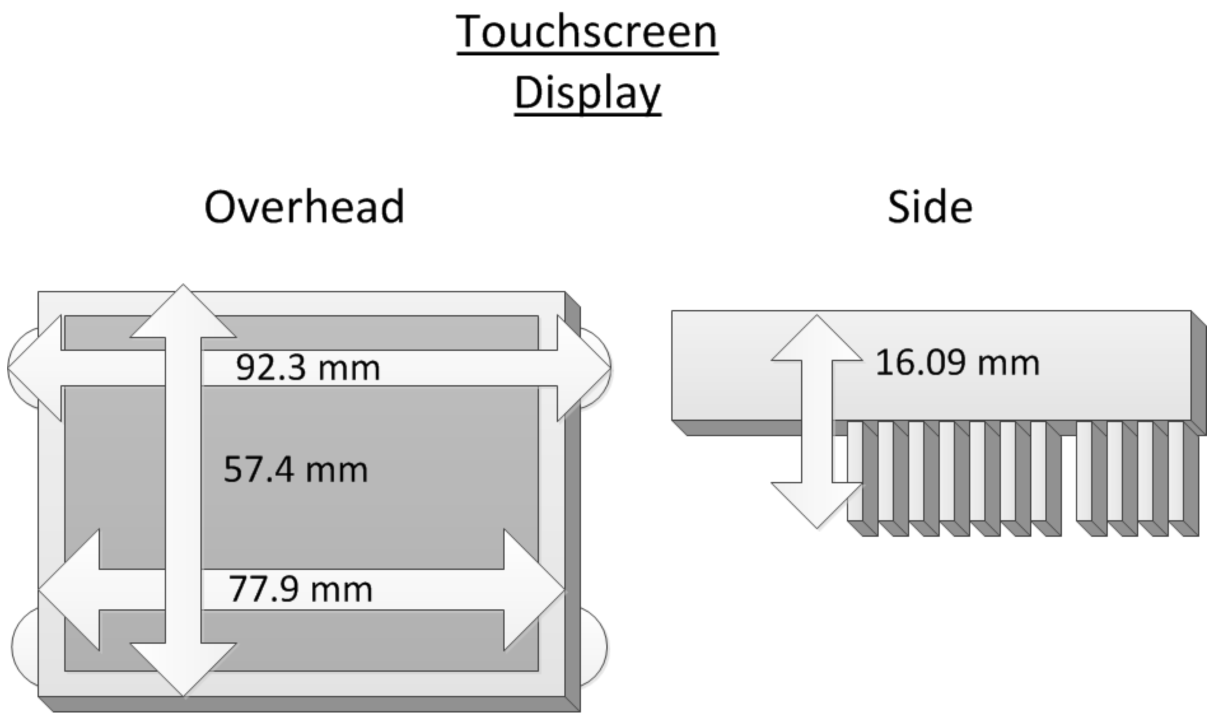


Figure 3.3: Touchscreen Dimensions

The touch screen, pictured above, is the user's interface with the reader. The display screen must be in position for the user to easily reach it. The dimensions in regard to area are very similar to that of the Raspberry Pi, which means packaging it directly below the Raspberry Pi would efficiently allocate room in the enclosure. The customer is requiring a reader that is compact, so making the enclosure as small as possible is a priority. The pins on the bottom of the screen make the screen approximately 1.6 cm thick.

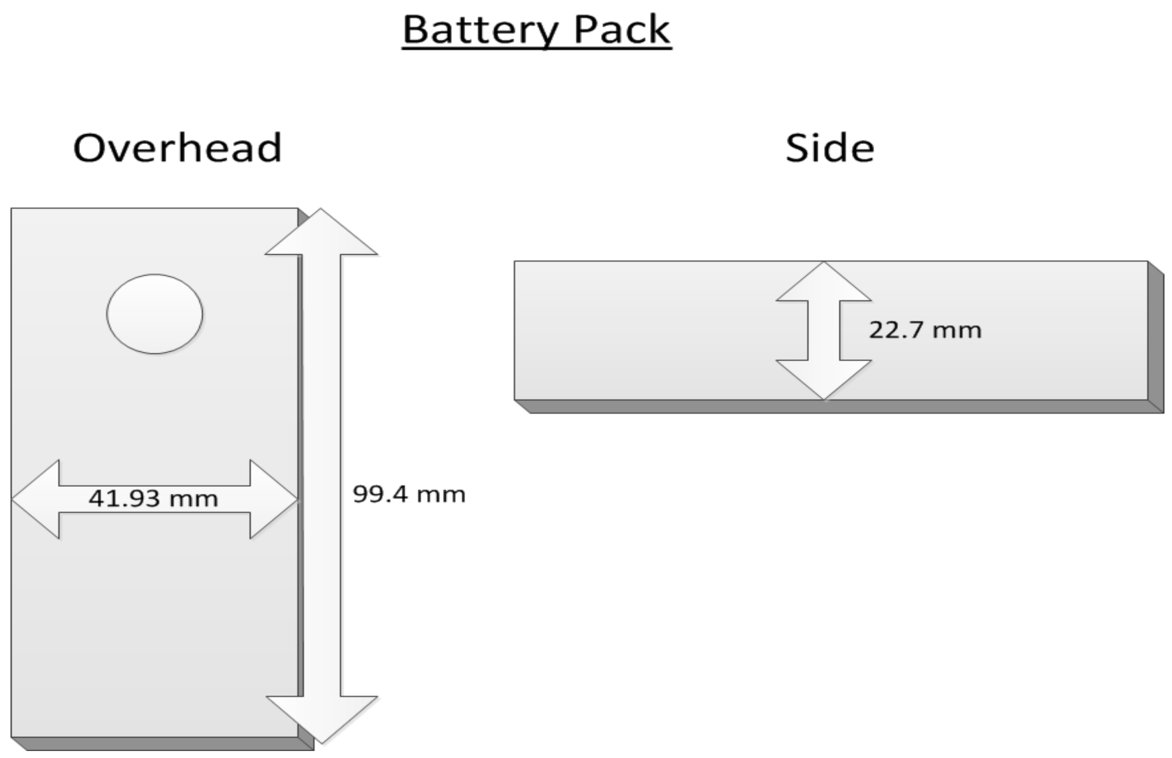


Figure 3.4: Battery Pack Dimensions

The battery pack is the heaviest component that will be included in the enclosure. The battery is already contained within a protective casing and can withstand moderate abuse. The protective casing is approximately 10 cm x 4.2 cm x 2.3 cm.

After quantifying and documenting the physical constraints for the enclosure, the designers brainstormed potential design selections. Recording and evaluating the sizing of the devices made it clear that the touchscreen display could fit on top of the Raspberry Pi. The devices operate at low power and heat is not a concern with regard to packaging the components into the enclosure. The largest factor to consider in the design is to protect the Raspberry Pi and the bottom of the touchscreen because the connections are exposed and sensitive to damage. Proposed enclosures and relative component locations are diagrammed and analyzed below.

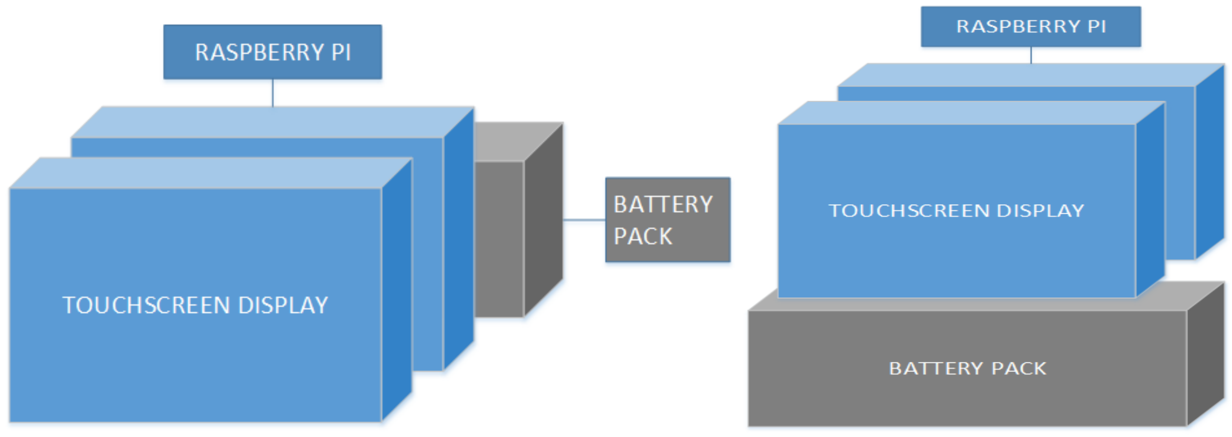


Figure 3.5: Illustration of the proposed enclosure's designs

The enclosure engineers compared the potential orientations for the major three components. Having the battery pack below the touchscreen display and Raspberry Pi was chosen as a more compact solution than having the components placed in-line with one another.

Siemens NX 8.5 computer aided design software was used to develop a digital representation of the enclosure. NX is a popular prototyping and development tool used throughout industry by engineers to build designs. The software gave the team the freedom to create nearly any design imaginable. Objects in the software are built block by block, and extruded together to form sections. The component measurements were used to size the case appropriately.

A major constraint in development of the case was the limitations of the Makerbot 3D printer to build rounded edges. The designers preferred to have a case that was rounded on the edges to increase aesthetic appeal and make the case more comfortable to hold. Due to recommendations from Michigan State University computing service representatives familiar with the operation of the Makerbot, the print would be at risk for failure, so the designers chose to limit the potential for print errors by opting for a sharp edge design.

Another issue to consider in printing the final design was the density of the plastic. The Makerbot is capable of printing the designs at various plastic density levels. It is more cost effective to print a less dense enclosure. In addition, the product will also be lighter, but this sacrifices the strength of the overall case. The Makerbot creates a honeycomb pattern instead of solid plastic at lower fill levels. The team chose to print the enclosure at 75% density to ensure proper wall thickness of the enclosure.

The final enclosure model is shown below. It utilizes the stacked Raspberry Pi and display, with the battery pack attached to the bottom shelf area. The LCD touchscreen slides into the slits on either side of the upper portion. The battery pack is mounted into the shelf on the bottom and distributes the weight so that the device is balanced when the user is holding the reader. The small square hole provides access for a connector to link the touch screen display to the Raspberry Pi. The larger square hole gives the Raspberry Pi's screen adaptor vertical room, while keeping the remaining Raspberry Pi components protected within the casing.

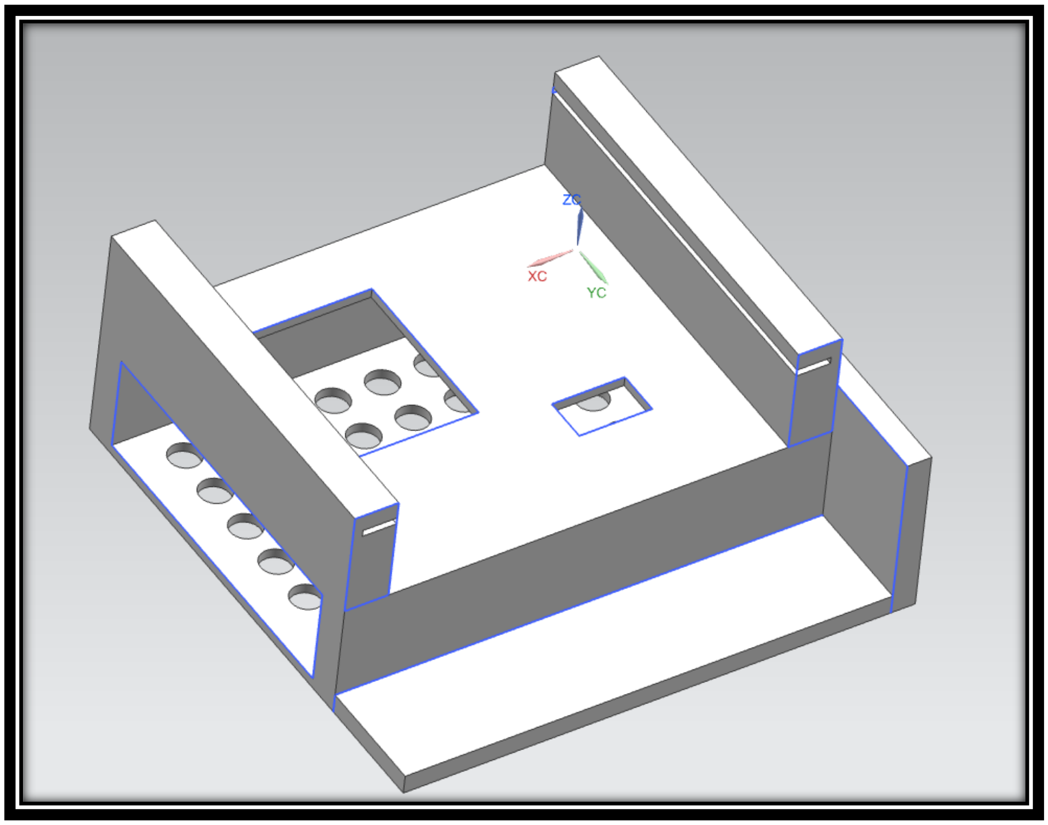


Figure 3.6: Screen capture of the final enclosure design

A side view of the enclosure is below. The Raspberry Pi can slide in through either side of the enclosure and is securely mounted. The Raspberry Pi is not to be removed and will remain stationary. The bottom section of the enclosure is large enough to hold the entire Raspberry Pi comfortably. From this view it is easier to see the slits that hold the touch screen in place.

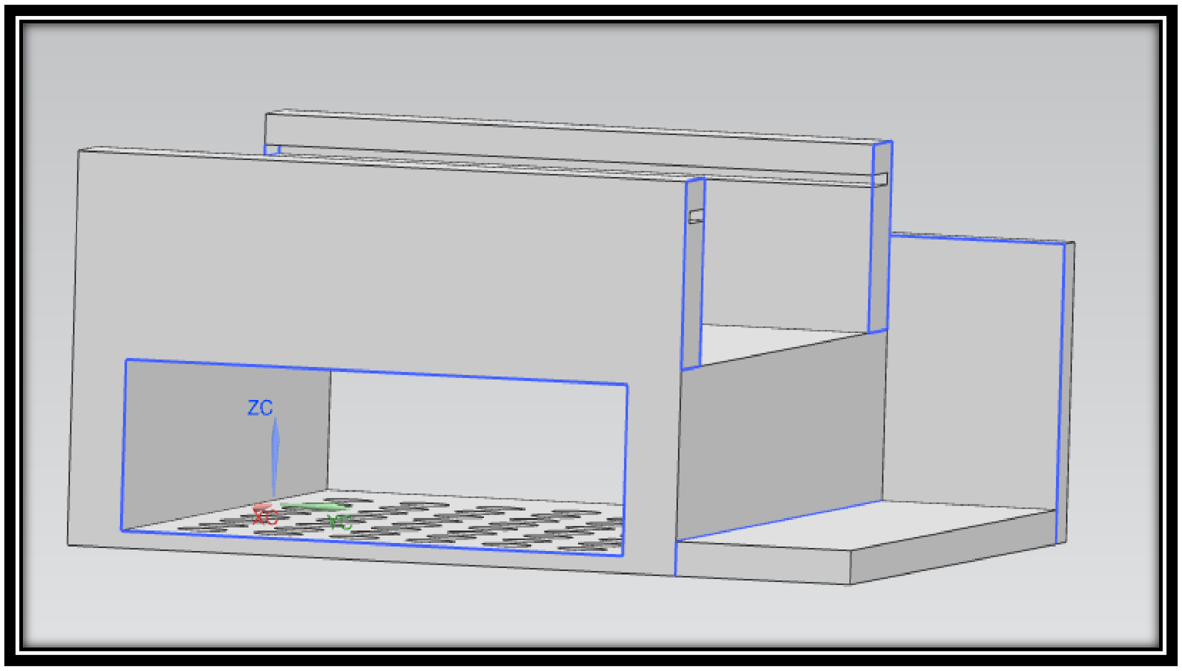


Figure 3.7: Screen capture of the side of the final design enclosure design

This final prototype design was successfully printed off the Makerbot and the components fit accordingly. Slight modifications were made to the sizing and version 1.1 was printed as the final design (same as pictured above). The enclosure feels comfortable to hold and the plastic is sturdy. The case holds up well against pressure applied to it and the surface will not chip or deteriorate. The case protects all the internal components neatly and meets the customers' expectations of a handheld device.

3.3 - Sensor Input/Output Connection

In order to be able to read data from the sensor, a serial cable was needed to connect the sensor to the microcontroller. To accomplish this, a custom ribbon cable was fabricated, which was connected to an 8-pin male header on one end that connected to the female input on the sensor. Eight individual jumper wires were soldered on other end of the ribbon cable, which were connected to the eight chosen GPIO pins on the Raspberry Pi. This prototype works well for testing purposes and is fully functional, but is not aesthetically pleasing. If this product were to be mass produced, it would not be held together with electrical tape, nor would it have jumper wires dangling off the end. Testing the cable for performance was a simple task that went a long way in guaranteeing the proper performance of the cable. Each wire in the ribbon cable was tested to ensure that the voltage into one side of the connector was the voltage coming out of the other side of the connector. A simple test consisting of a power supply and multimeter was used to make sure the connections were solid, and nothing was being shorted. Another way to prevent shorting was to measure the output voltage of each neighboring wire, making sure that the voltages were close to zero on all of the other wires except the one with the applied input voltage. This simple test ensured that the cable would perform properly, and that the ribbon cable connector pictured below would be a viable solution.

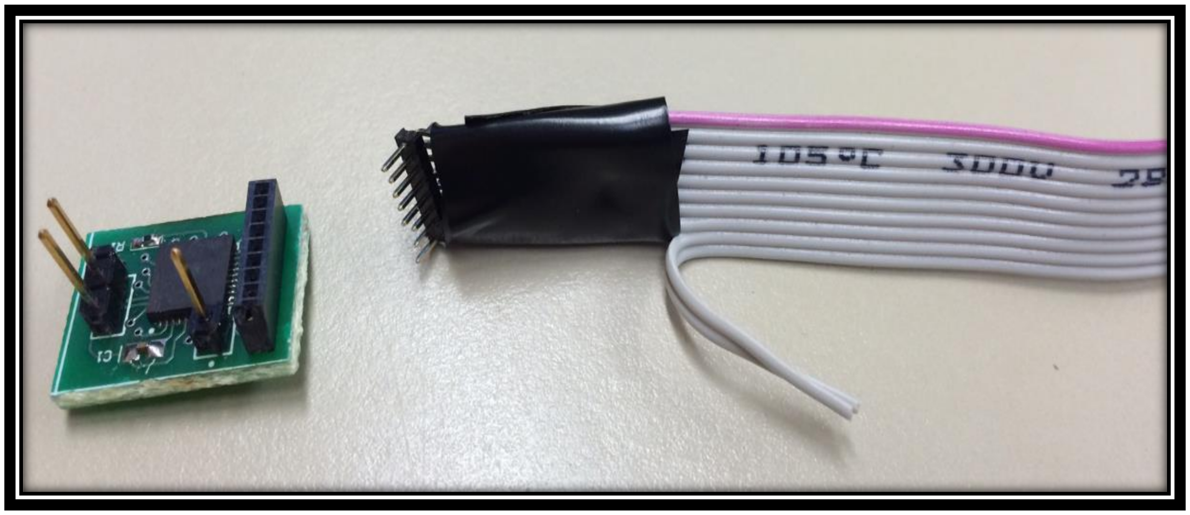


Figure 3.8: PFG Sensor (Left) and Custom 8-pin Plug-and-Play Connector (Right)

3.4 - Power Management

A key aspect of this project is power management, which entails providing constant power to the Raspberry Pi, the touch screen, and sensor. A power supply was required to support the multiple parts, keeping in mind that the device must be portable at the same time. With these requirements, the importance of finding a battery for the reader that supplies sufficient power to the components became a top priority. A USB battery pack was found that provided 5 V at a current rating that is up to 1 A with a maximum capacity of 4400 mAh. This provided enough power for the reader due to the fact that Raspberry Pi requires 5 V running at an average of 700 mA. Furthermore, the high ampere rating provided by the small battery allows the Raspberry Pi and screen to operate at any workload. For this battery, the current is enough to supply power to the Raspberry Pi, touch-screen, sensor, and the USB Wi-Fi dongle. The screen is designed to accept a 5 V power supply so that no voltage regulators were required. However, a voltage regulator was required to attach the sensor to the Raspberry Pi because the range of voltage of the sensor is between 1.8 V - 2.5 V. To solve this problem, a 3.3 V GPIO pin on the board, not used for any other purpose, was connected to a drop-down linear voltage regulator allowing the sensor to interface with the Raspberry Pi.

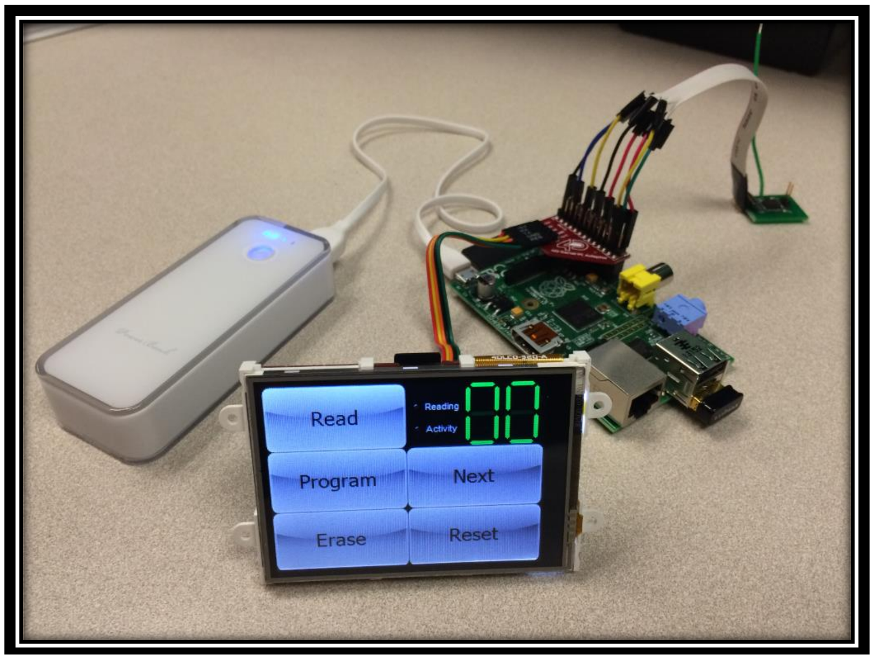
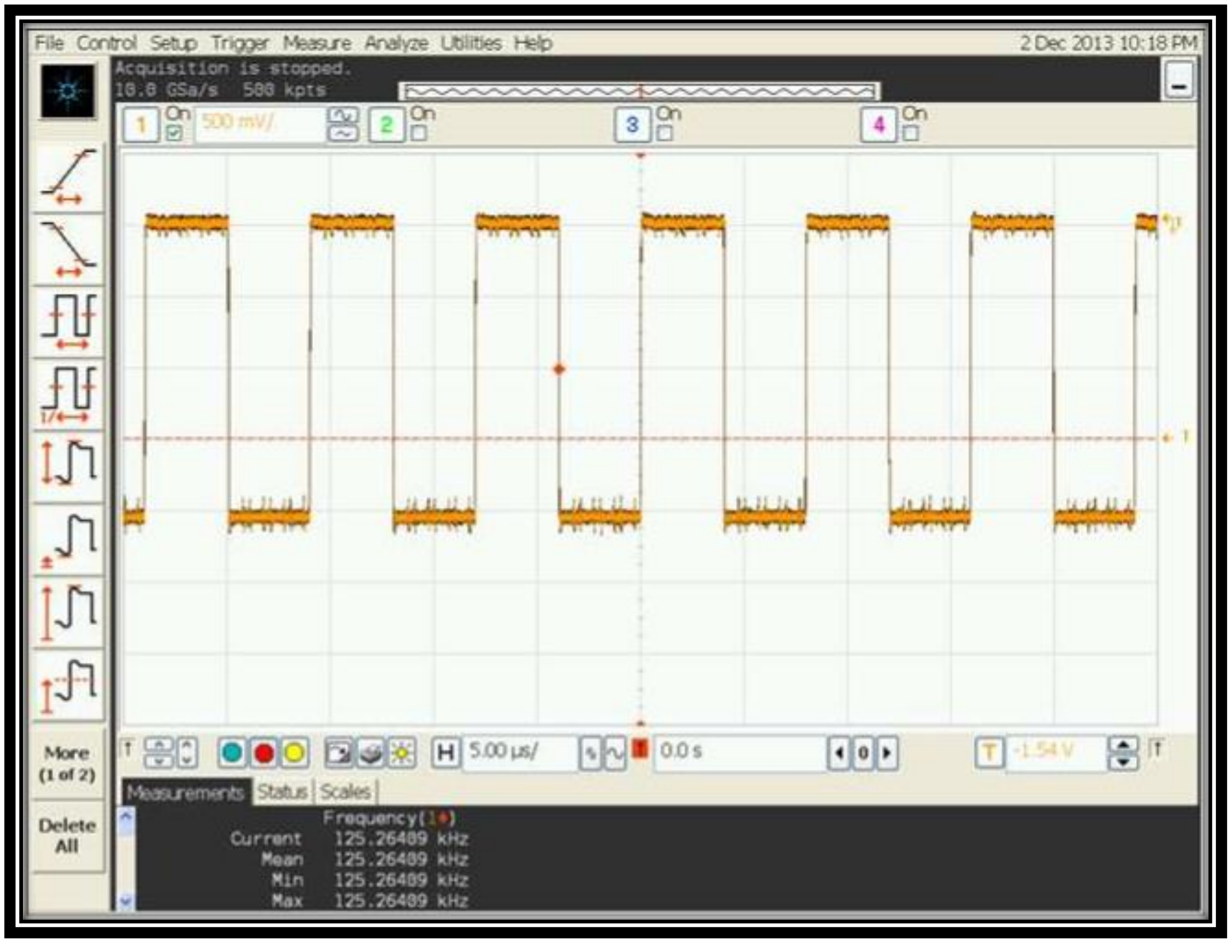


Figure 3.9: A picture of the battery pack connected to the Raspberry Pi

Chapter 4 - Test Data

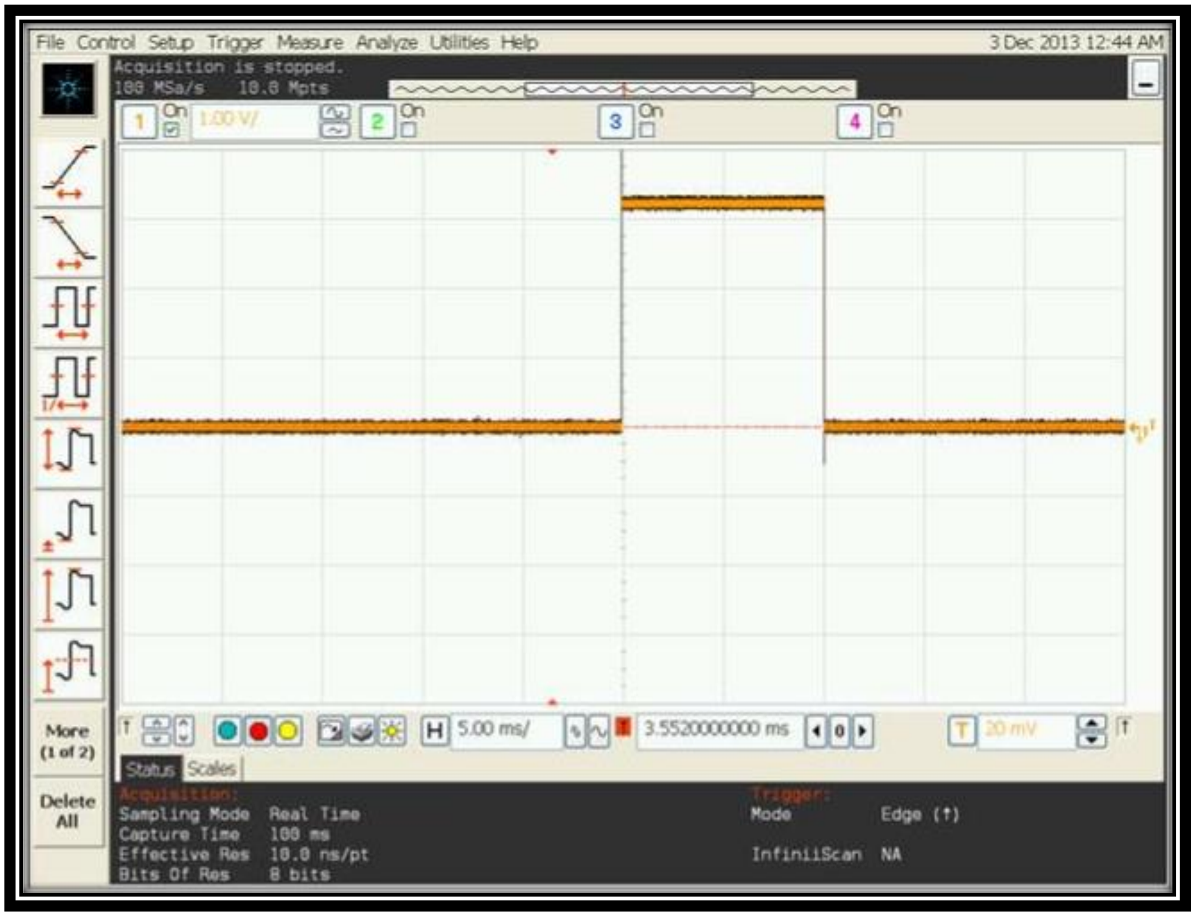

File Control Setup Trigger Measure Analyze Utilities Help
2 Dec 2013 10:18 PM
500 mV/
5.00 µs/
0.0 s
-1.54 V
More (1 of 2)
Delete All
Measurements Status Scales

```
Acquisition is stopped.
10.0 GSa/s   500 kpts

           Frequency(1*)
  Current   125.26409 kHz
     Mean   125.26409 kHz
      Min   125.26409 kHz
      Max   125.26409 kHz
```

Figure 4.1: Clock Speed of PFG Sensor

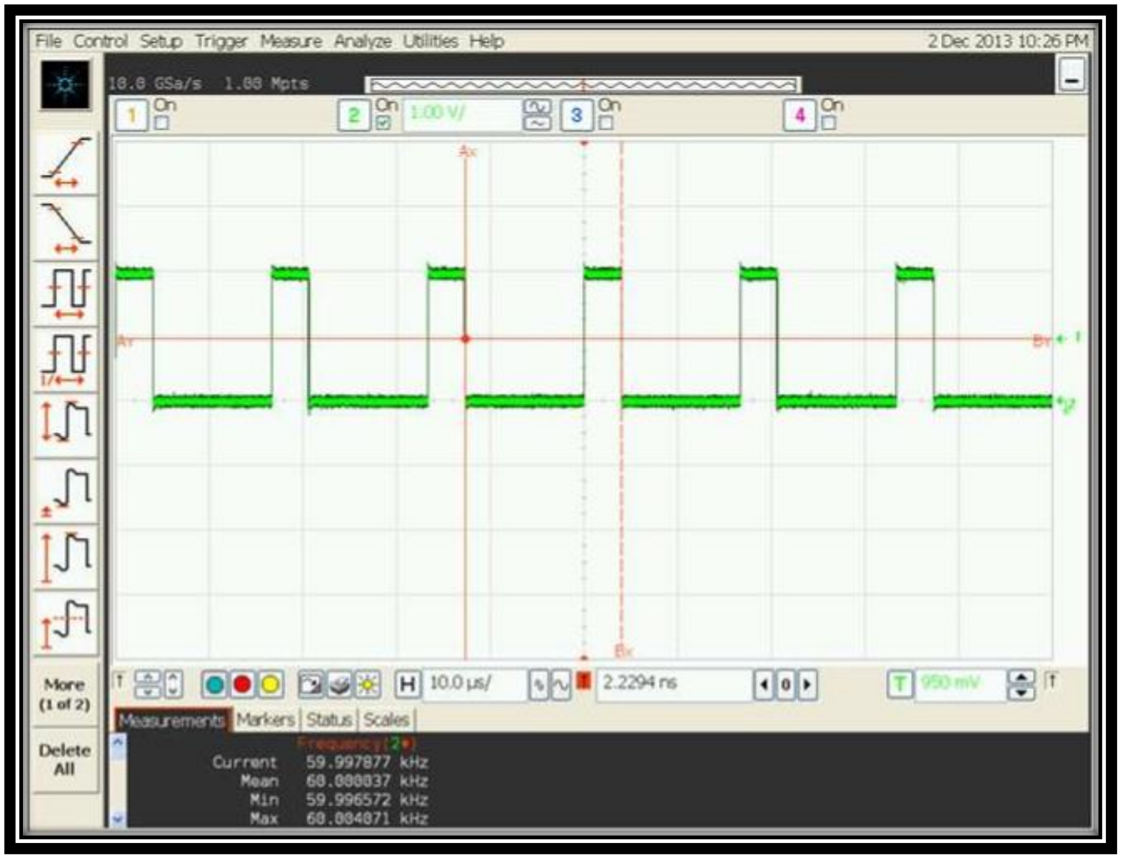
The oscilloscope capture above shows the square wave output of the PFG sensor's clock, which operates at 125 kHz. This confirms that the sensor is receiving power and is operational.

After the sensor is receiving power, sending a digital pulse to pins on the sensor can initiate their functions. When a user inputs their function selection, the screen communicates with the Raspberry Pi, and then the Raspberry outputs a 10ms pulse through its GPIO pins. The screen capture below shows how the pulse appears, as measured on an oscilloscope. The pulses from the Raspberry Pi are 3.3V because of the on chip regulator, however the sensor requires a smaller voltage of around 2V to ensure the chip is not damaged.   An external voltage regulator will be implemented in order to step down the voltage from the supply voltage of 5V to the sensor output voltage of 2V.



```
Acquisition is stopped.
100 MSa/s   10.0 Mpts

Acquisition:                        Trigger:
Sampling Mode   Real Time           Mode          Edge (↑)
Capture Time    100 ms
Effective Res   10.0 ns/pt          InfiniiScan   NA
Bits Of Res     8 bits
```

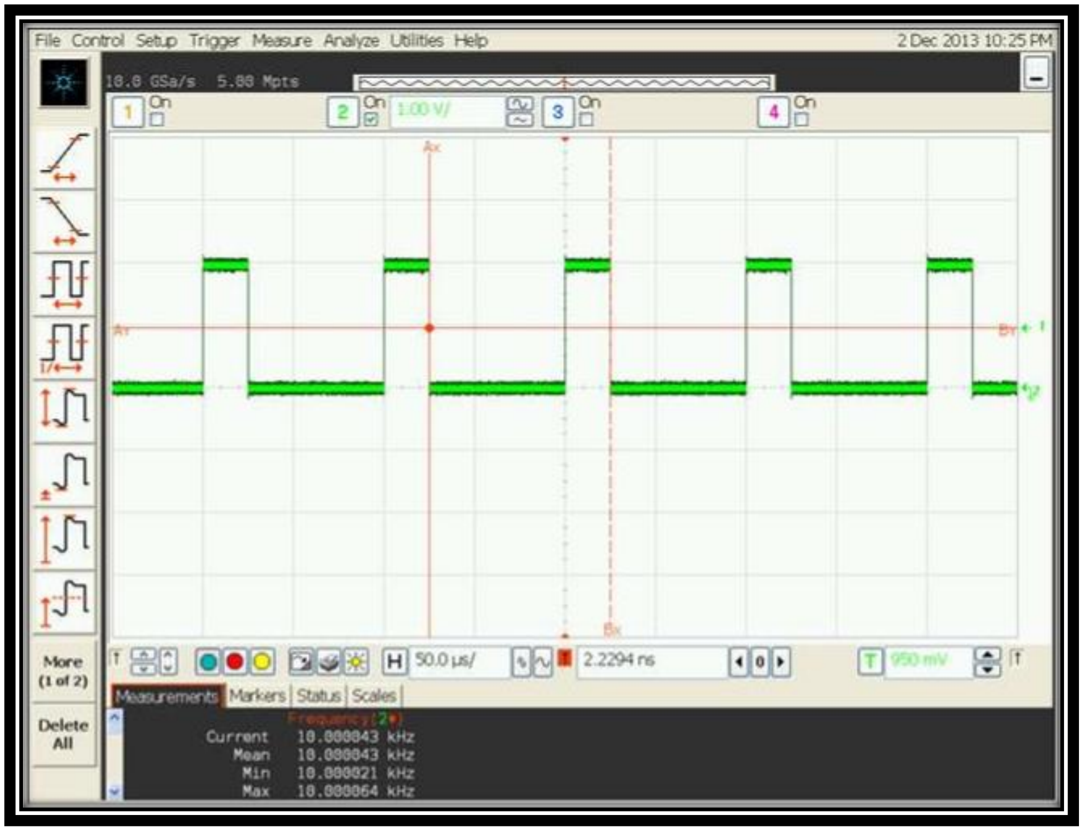
Figure 4.2: 10 ms Clock Pulse via Raspberry Pi GPIO



```
         Frequency(2)
Current  59.997877 kHz
   Mean  60.000037 kHz
    Min  59.996572 kHz
    Max  60.004071 kHz
```

Figure 4.3: Maximum Output Pulse Train of the PFG Sensor

When a pulse is received by the sensor, it outputs both analog and digital representation of the data stored on a specific channel. The capture above shows how a major impact is represented when outputted from the digital out pin on the sensor. The higher the strength of the impact the higher the frequency outputted by the sensor. The frequency of this particular impact is approximately 60 kHz. A smaller impact is shown in the figure below, with a digital frequency of about 10 kHz.



```
           Frequency(2)
Current   10.000043 kHz
   Mean   10.000043 kHz
    Min   10.000021 kHz
    Max   10.000064 kHz
```

Figure 4.4: Minimum Output Pulse Train of the PFG Sensor

Figure 4.1 demonstrates the testing functionality of the sensor. The plot shows the clock speed that the sensor is running (~100 kHz typical). This was used to verify that the sensor was working and that power was being delivered. Figure 4.3 shows the maximum digital output of the sensor (~½ clock speed). Each channel on the sensor has digital output that is a pulse train with varying pulse widths and periods. The frequency of the pulse has a direct correlation to the magnitude of the force sustained by the sensor and the average of the pulses (analyzed using MatLab) is equivalent to the analog DC output of the sensor. The lowest frequency of the sensor is (~10 kHz) as shown in figure 4.4 which correlates to a constant average DC value. The sensor outputs the data both analog and digital in linear steps (.2 V – 2 V) in which the step size can vary with each sensor. Note that each channel on the sensor will have different frequencies associated. The higher in magnitude the impact, the more channels are activated; a weak impact will activate the first few channels and a harder impact will activate more channels. Figure 4.2 verifies that the Raspberry Pi outputs a 10ms pulse width to the sensor which is the minimum time frame needed to instantiate (Next, Program, Reset, and Erase). The software accommodates for extended commands (holding down one of the function buttons) as well as multiple commands (multiple presses of the buttons). The software deals with this by simply ignoring user input to insure the sensor has time to properly decipher user input.

Chapter 5 - Final Cost, Schedule, Summary, and Conclusions

5.1 Final Cost

MSU provided the team with a maximum budget of $500 to be spent on the project. The total spent was $201.42 which is around 41% of the total amount. This is ignoring the prototyped parts that were ordered yet not used in the final design.

Part Number	Description	Quantity	Place of Purchase	Total Cost (USD)
04X5042	Raspberry Pi Single Board Module B	1	Newark	$40.00
LCD-11743	4D Systems 3.2" TFT LCD touch-screen	1	Karlsson Robotics	$76.27
1565-ADA	USB Battery Pack 4400mAh – 5 V @ 1A	1	ADAfruit	$29.95
1294-ADA	SD/MicroSD Memory Card (8 GB)	1	ADAfruit	$17.24
EW-7811Un	Edimax EW-7811Un 150 Mbps Wireless 11n Nano Size USB	1	Amazon	$9.99
S8V3A	Adjustable Step-Up/Step-Down Voltage Regulator	1	Pololu Robotics and Electronics	$14.90
AE08A-5-ND	5 ft - 28 AWG ribbon cable	1	Digi-Key	$4.31
S9066-ND	10 pin - female header connector	1	Digi-Key	$2.76
N/A	Enclosure 3D Printing	1	DECS Office	$11.00
	Total			$206.42/$500.00

Table 5.1: Final cost of the utilized components

5.2 Schedule

Task	Duration	Start	Finish	Member Responsible
Pre-Proposal Due	1 day	Wed 18,09,13	Wed 18,09,13	Group
Team Webpage	49 days	Fri 27,09,13	Wed 04,12,13	Yuval
Gantt Chart Due	1 day	Fri 27,09,13	Fri 27,09,13	Ron
Design Issues Paper Due	1 day	Fri 04,10,13	Fri 04,10,13	Group
Voice of Customer Due	1 day	Mon 30,09,13	Mon 30,09,13	Group
Page for Design Day	1 day	Fri 04,10,13	Fri 04,10,13	Group
FAST Diagram Due	1 day	Mon 07,10,13	Mon 07,10,13	Group
Oral Presentation Practice	1 day	Wed 02,10,13	Wed 02,10,13	Group
Final Proposal Due	1 day	Fri 11,10,13	Fri 11,10,13	Group
Team Oral Presentation	1 day	Fri 04,10,13	Fri 04,10,13	Group
Technical Lecture Assignment	1 day	Fri 06,09,13	Fri 06,09,13	Group
2-Page Progress Report Due	1 day	Wed 23,10,13	Wed 23,10,13	Group
Engineering Notebook Photocopy Due	1 day	Wed 23,10,13	Wed 23,10,13	Group
Business Canvas Assignment	1 day	Fri 06,09,13	Fri 06,09,13	Group
Technical Lecture	1 day	Fri 01,11,13	Fri 01,11,13	Group
Application Note Due	1 day	Wed 06,11,13	Wed 06,11,13	Group
Team Design Issues Paper Due	1 day	Fri 22,11,13	Fri 22,11,13	Group
Progress Report	1 day	Wed 20,11,13	Wed 20,11,13	Group
Project Demo to Facilitator	1 day	Wed 20,11,13	Wed 20,11,13	Group
Personal Self-Assessment Papers	1 day	Wed 27,11,13	Wed 27,11,13	Group
Demo Project Prototype	3 days	Mon 25,11,13	Wed 27,11,13	Group
Final Report Due	1 day	Wed 04,12,13	Wed 04,12,13	Group
Final Poster Due	1 day	Wed 04,12,13	Wed 04,12,13	Group
Design Day	1 day	Fri 06,12,13	Fri 06,12,13	Group
Final Notebooks	1 day	Fri 06,12,13	Fri 06,12,13	Group
Final Documentation	1 day	Fri 06,12,13	Fri 06,12,13	Group
Initial Team Meeting	1 day	Fri 06,09,13	Fri 06,09,13	Group
Weekly Team Meeting with Dr. Salem	56 days	Wed 18,09,13	Wed 04,12,13	Group
Understanding Project Statement	1 day	Wed 11,09,13	Wed 11,09,13	Group
Initial Sponsor Meeting: Dr. Chakrabartty	1 day	Wed 11,09,13	Wed 11,09,13	Group
Order Parts	1 wk	Fri 20,09,13	Thu 26,09,13	Thamer
Power Supply	49 days	Fri 27,09,13	Wed 04,12,13	Yuval
Power Supply Design	4 wks	Fri 27,09,13	Thu 24,10,13	Group
Initial Protoype	3.2 wks	Fri 25,10,13	Fri 15,11,13	Group
Power Supply Validation	13 days	Mon 18,11,13	Wed 04,12,13	Group
Raspberry Pi Programming	49 days	Fri 27,09,13	Wed 04,12,13	Ron
Raspbian OS Install	1 day	Fri 27,09,13	Fri 27,09,13	Group
Program Creation	6.7 wks	Fri 27,09,13	Thu 14,11,13	Group
Program Debugging	2.9 wks	Thu 14,11,13	Wed 04,12,13	Group
Custom I/O Port Design	49 days	Fri 27,09,13	Wed 04,12,13	Brett
3D CAD Drawing	21.5 days	Fri 27,09,13	Mon 28,10,13	Group
3D Print	14 days	Mon 28,10,13	Thu 14,11,13	Group
Port Connection Validation	2.8 wks	Fri 15,11,13	Wed 04,12,13	Group
Custom Enclosure Design	49 days	Fri 27,09,13	Wed 04,12,13	Thamer,Evan
Overall Device Measurement	3 days	Fri 27,09,13	Tue 01,10,13	Group
3D CAD Drawing	3.6 wks	Wed 02,10,13	Fri 25,10,13	Group
3D Print	10.5 days	Mon 28,10,13	Mon 11,11,13	Group
Product Assembly	17.5 days	Mon 11,11,13	Wed 04,12,13	Group

Table 5.2: Schedule of Task and Division of Labor

5.3 Summary

The Raspberry Pi minicomputer is a low cost solution that meets the customers' requirement to interface with the Piezo sensor device. The popularity of the Raspberry Pi made technical development easier for the team as a multitude of sources is available. The proprietary IDE software made specifically to program the touchscreen made development of a GUI much simpler. The touchscreen eliminates the need for physical hard switches and allows for modifications to the software in the future for future revisions of the sensor. Although the touchscreen was not the cheapest option, the economics of the products real world applications make this less of a concern as the functionality of the screen outweighed its cost. Sensors deployed into the field are expensive, in the area of $200+. In addition, the user requires only one proprietary reader to read any amount of sensors. The overall cost of the reader is just a fraction of the entire PFG Sensor system.

The PFG sensors that have stored data will need acquisition and the reader will then be attached to the sensor. The touchscreen takes the role of issuing orders allowing users to press the buttons to interact with the sensor and implement the desired functionality pre-programmed on the Raspberry Pi. After that, the Raspberry Pi will save the data provided from the PFG sensors on its SD card which can be accessed via Wi-Fi through a wireless network (using SSH) or physically taken from the SD card. The resulting data can then be analyzed using an engineering application such as MATLAB.

5.4 Conclusions

Overall, the project was a success. The design team successfully met all customer requirements for the micro sensor reader to extracted stored data on the sensor. The device was built within budget and on time. Even so, there was some unforeseen difficulties encounter in the development process.

It was challenging to modify the Raspberry Pi to meet the power requirements of the sensor. The Piezo sensor is a sensitive device with voltage tolerances that are in a slim range. The natural output of the Raspberry Pi exceeds acceptable levels, so the team had to find a solution. The Raspberry Pi has on-board voltage regulators

In addition to the challenges with the power, the actual sensor supplied by the sponsor was, at times, unreliable during tests. We experienced failures during testing with the sensor. Often, we would test the sensors clock to make sure that everything was operating properly. Sometimes the output displayed on the oscilloscope would only appear as noise, and led the team to believe the sensor was inoperational. This was confusing for the team and slowed down progress.

The actual production of the enclosure also experienced issues. The 3D Makerbot printers available in the Engineering Building were used to print the plastic enclosure. While it was anticipated that these would be readily available to develop many revisions of the design, the team had difficulties in getting on the print queue. This meant that fewer enclosures could be made and less designs tested. Also, the density of the plastic that was required made the prints significant time consumers. The enclosure took between 7-8 hours from print start to finish. While these challenges were a barrier, the team overcame and was still able to produce a case that met the needs of the customer.

Development of the sensor reader was a rewarding experience for the team. Researchers at Michigan State can use the device to quickly and efficiently extract data off of the Piezo sensors. This product contributes to the research done at the University and supports future innovation in the field of data harvesting devices.

Appendix 1 - Technical Roles

Ron Razalan - Manager, Software Lead

The primary responsibility was all software development. The software side was comprised of three major parts: GUI creation (using 4D Systems Workshop4 - ViSi Genie), data acquisition algorithm (gathering data from the GPIO pins) and control software. The company that manufactured the smart touch screen provided proprietary software to help create a functional GUI specifically for the screen and similar products. This integrated development environment allowed the creation of a screen GUI with pre-made button styles and figures. This allowed fast creation of a GUI which can be easily modified for the user to use in interacting with the PFG sensor.

The data acquisition algorithm was designed with the basis of sampling at a constant frequency (1 MHz) and a constant time (1 second). This allowed for equal and consistent measurements of data. This made the data strings represented by 1s and 0s in the output file to have similar lengths. The algorithm is allowed to run for a certain time frame to minimize inconsistent data strings within the output file. Since the algorithm runs for seconds and the data outputs at a high frequency (10 kHz - 50 kHz), there is sufficient time for the data to be sampled. The sample time of the algorithm can be easily modified with a change of a single variable. The final component of the software development is the code that implements the error checking for user input, communication between the Raspberry Pi and the touch screen, and the data-logging function. All of the coding was implemented in C, using libraries made specifically for the Raspberry Pi and the 4D Systems Touchscreen. The C libraries provided function calls that streamlined communication with the Raspberry Pi and any peripherals connected the GPIO.

The non-technical role was manager with responsibility for the group and communication with the sponsor. The customer requirements were needed to be met with precision and any group activities needed personal attention to each. Personally designated weekly meetings as a group and to the facilitator and responsibility with assigning the technical roles of each group member based on experiences/strengths.

Evan Gardetto - Documentation, Enclosure Design and Verification

Evan Gardetto was responsible for exploring and implementing methods to house project components into one handheld device. Began by creating plan for organization of electronic components into a protective casing, taking into account component size, weight, and effects on end functionality. Coordinated with Thamer Alshuaibi to sketch potential designs on paper and discuss positive and negatives of different design choices. In total, three distinctive designs were evaluated, each that met the needs of the customer, but were accomplished by orienting the components differently.

Prototyped using Siemens NX 8.5 computer aided design software. Without prior experience, extensive research and self-education was dedicated to learn how to properly take advantage of the software. After becoming confident in using NX, an application note on enclosure prototyping using NX 8.5 was writing to assist other engineers make their own cases for their projects. The note is a step-by-step tutorial to basic NX design and builds the fundamental skills required to develop a custom case.

Worked directly with Thamer Alshuaibi to build an enclosure that closely resembled the paper design that we agreed on. NX designs were examined for errors and measurements verified for accuracy. It was important that the first prototype was well designed because the printing waiting list is significant and execution of the actual print takes about 7-8 hours. After verifying design characteristics, the initial enclosure was manufactured using Makerbot 3D printer.

Once the case was printed, the components were retrofitted into the actual enclosure. Functionality and structural integrity of case was inspected to ensure that product met customer requirements. This included verifying that components were well protected against damage and that they were held tightly in place. In addition, the device was demoed to confirm that case design provided a satisfactory user experience. Evaluated enclosure results and determined modifications to design characteristics in order to improve overall product. Changes were implemented in iteration until final enclosure satisfied all objectives and housed the electronics in the best possible way.

Thamer Alshuaibi - Lab Coordinator, Enclosure Design and Product Validation

Responsible for deciding what parts to be used and designing an enclosure to protect and hold the device. Started with Structuring and establishing the device by brainstorming with teammates and then obtained the best possible choice of components for the device. Went over the specifications of the parts and read different datasheets to assure safety and to guarantee that the components were suitable for each other. Assigned for placing orders via the ECEshop website and attained components before deadlines. After that, collaborated with Evan Gardetto to design a case for protection and hold the components all together. Learned Siemens NX 8.5 computer aided design software with no previous experience, and successfully applied that learning to the project by designing multiple prototypes initially until choosing the most efficient design between all to have a house for the device. Evaluated the enclosure after using the Makerbot 3D printer, and implemented justifications to satisfy customer requirements. Moreover, re-shaped the case with an improved presentation that covered components to be held in place, and acquired more safety to users. All with maintaining the size, weight, and operation of the overall design to be convenient and handy.

Worked on an application note for going over deep explanations of the specifications of the Raspberry Pi's inputs/outputs, followed by demonstrating the connections between components – Raspberry Pi, 4D adaptor and 4D display module, and the power supply – and showing the features of the multiple parts used in the device. Furthermore, justified and clarified the value of the device and how would it be useful for users to attain else than others in the market. Helped with writing well-organized technical reports and delivered the team's poster enhanced with explanations and illustrations divided between texts, graphs, and pictures.

Created a logo to be posted on the team's poster, reports, presentations, and stickered on the enclosure as an extra work to distinguish the team's outcomes.

Brett Johnson - Presentation Coordinator, Connector Design and Fabrication

Responsible for the design and fabrication of the custom ribbon connector, as well as aided in the power conversion process, and the testing of the product. Began by exploring solutions to connect the sensor with the microcontroller, taking into account the size and pin arrangement of the sensor as well as the microcontroller. The pins on the sensor were spaced very closely together, which posed a challenge when looking for connectors. Several design options were considered, but the most practical option was to use a ribbon cable with connectors. This made the connector cable less bulky and easier to implements.

Created prototype using a 28 AWG, 8-wire ribbon cable, eight jumper wires, and an eight pin male header to connect to the sensor. Soldered each individual ribbon wire to a corresponding pin on the male header. This was a difficult task since the pins were so close together, and could be easily shorted out if solder was to bridge two neighboring pins. In order to test this connection, a voltage was applied to the free end, and was measured at the end with the connector. The output voltage of each wire was equal to the input wire, confirming that each connection was good. In order to prevent shorting, the output voltage of each neighboring wire was measured against the input voltage, and no shorts were found. To connect to the microcontroller, the other side of the ribbon cable was soldered to jumper cables. Each individual wire in the ribbon cable was soldered to a jumper cable, where the other end of the jumper cable could be connected right to the GPIO pins of the microcontroller.

Also helped design the voltage regulator to help step the microcontroller voltage of 3.3V down to 2V which was needed to power the sensor. It was a difficult task finding a regulator with a small enough voltage that could still supply enough current to the microcontroller.

Yuval Levental - Power Distribution

Collaborated with Thamer Alshuaibi in choosing the 5 V battery pack power supply for the Raspberry Pi. Selected a USB battery pack from Adafruit Industries, specifically designed for the RPi's connector. Found a linear voltage regulator to step-down the voltage from the 3.3V GPIO pin to 2V. The voltage regulator a Texas Instruments TPS7A4700, adjustable with an output current of 1A.

Selected the battery pack through conducting extensive research on supplies specially designed for the Raspberry Pi. Visited several sites for this purpose, and compared costs with benefits. Final project was chosen due to its' ability to recharge by connecting to a computer through a USB port, and its error-checking system. The original voltage regulator selected, also from Texas Instruments, only outputted 300 mA of current to power the sensor. This clearly wasn't enough.

Designed the team website, which included pages for the project overview, description, gallery, documents, and information about the team members. The project overview gave several pictures summarizing the nature of the project. The project description page explains the three parts to our project - namely the power supply design, the programming, and the enclosure. The gallery contains several pictures of the project from start to finish. The documents page contains the team members' application notes and deliverable documents. Finally, the team page gave the names, majors, and emails of the members. Used a combination of HTML, CSS and JavaScript to design the website.

Created the diagram and works cited for the pre-proposal. Lectured about the general design in the pre-proposal lecture, and the piezoelectric crystal's chemical structure in the technical lecture. The design was created in paint.net, and was two-dimensional. Researched official scientific sites to learn about how the piezoelectric crystal structure would deform.

Appendix 2 – References[2]

Lajnef, Nizar, Dr. "Self-powered Sensing in Structural Health and Usage Monitoring." Diss. Michigan State University, 2008. *Google Books*. ProQuest, 2008. Web. 17 Sept. 2013.

Michigan State University. AIM Laboratory. *PFG Self-powered Health and Usage Monitoring Systems (HUMS)*. *Spartan Innovations*. N.P., 2013. Web. 17 Sept. 2013.

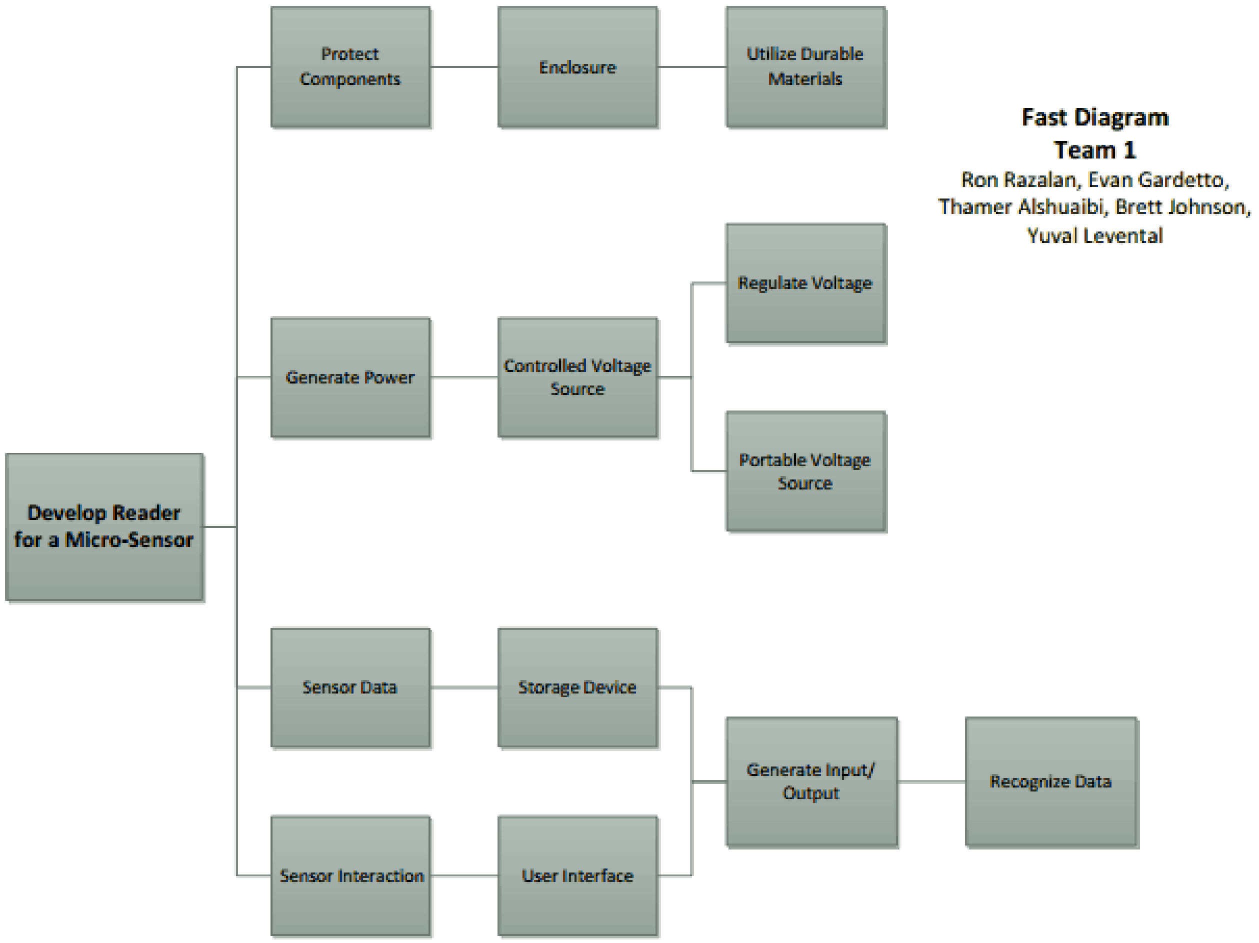
Sikora, Christopher. "Siemens NX Basics 220." Vertanux1. 2013 https://docs.google.com/file/d/0BxHoQac6LA7_dFBucEZ4ejAwSU0/edit

Zhu, Dibin, Stephen P. Beeby, Michael J. Tudor, and Nick R. Harris. "A SELF POWERED TAG FOR WIRELESS STRUCTURE HEALTH MONITORING IN AERONAUTICAL APPLICATIONS." *School of Electronics and Computer Science, University of Southampton* (n.d.): n. page. *University of Southampton*. University of Southampton. Web.

"BCM2835 ARM Peripherals." *Raspberry.org*. Broadcom Corporation, 2012. Web. <http://www.raspberrypi.org/wp-content/uploads/2012/02/BCM2835-ARM-Peripherals.pdf>.

[2] These references are to be used to analyze differing types of sensor systems, including the theory behind them. They may be compared and contrasted to determine the best or most desirable method, or theoretical starting point. The sources include

Appendix 3 – FAST Diagram

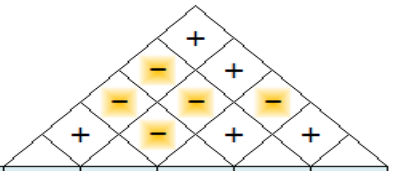


Appendix 4 -House of Quality diagram

Correlations	
Positive	+
Negative	−
Correlation	

Relationships	
Strong=9	●
Moderate=3	○
Weak=1	▽

Direction	
Maximize	▲
Target	◇
Minimize	▼



			Column #	1	2	3	4	5
			Direction of Improvement	▲	▼	◇	▼	▲
Row #	Max Relationship Value in Row	Relative Weight	Quality Characteristics (a.k.a. "Functional Requirements" or "Hows") / Demanded Quality (a.k.a. "Customer Requirements" or "whats")	Solid Assembly	Simple Physical Structure	DC Power Supply	Low Power Consumption	Ease of Sensor's Application
1	9	10%	All Inclusive	○	○	▽	▽	▽
2	9	16%	Interface with Sensor	○	▽	○	●	●
3	9	16%	Read/Write to Pins	▽	▽	○	○	▽
4	9	16%	Store Data	▽	○	●	●	●
5	9	15%	Portability	●	●	●	●	○
6	9	13%	Low Cost	○	○	○	○	○
7	9	14%	Regulate Voltage	▽	▽	●	●	●
			Target or Limit Value	9	9	9	9	9
			Difficulty (1=East to Accomplish, 10=Extremely Dificult)	3	5	8	7	6
			Max Relationship Value in Column	9	9	9	9	9
			Weight / Importance	298	298	550	646	524
			Relative Weight	15%	15%	25%	20%	25%

Appendix 5 - Data Acquisition Algorithm and Pseudo code

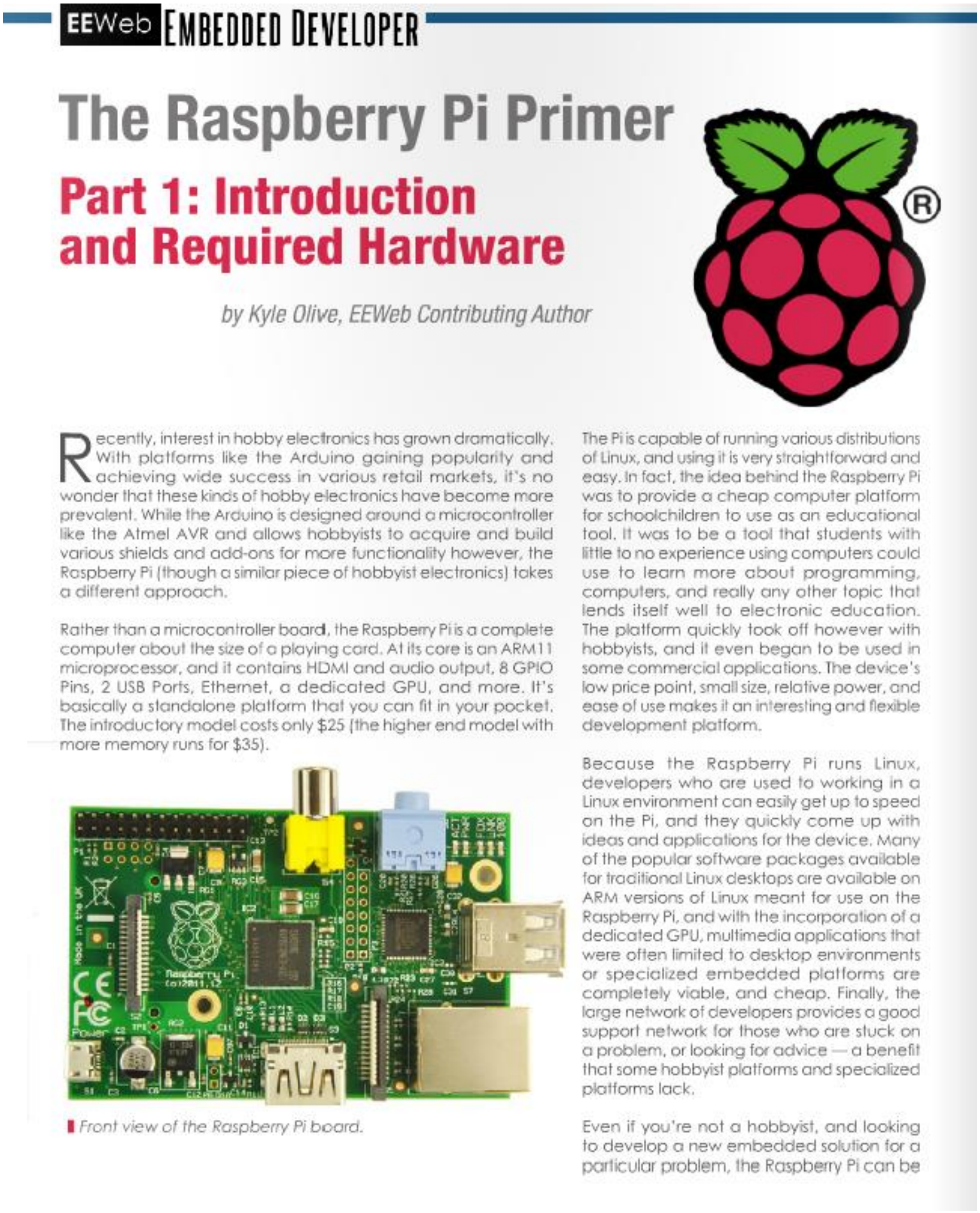
```
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <sys/time.h>
#include <signal.h>

#include <wiringPi.h>
#include <geniePi.h>

/*
for testing on Function Generator: 20KHz, 2Vpp, +1V DC Offset, 25% Duty Cycle
*/

int channel = 1; //channels 1-7 on sensor
int maxchannel = 8; //max channels on sensor + 1
volatile int running = 1; //flag for read timeout
int buttondelay = 1000; //delay in mseconds
int readtime = 1; //seconds
FILE *fp; //csv file

void SetupGenie()
{
        genieSetup("/dev/ttyAMA0",115200); //RPi device, baud rate
        genieWriteObj(GENIE_OBJ_FORM,0,0);
        genieWriteObj(GENIE_OBJ_LED,0,0);
        genieWriteObj(GENIE_OBJ_LED,1,0);
        genieWriteObj(GENIE_OBJ_LED_DIGITS, 0, channel);

        pinMode(7,OUTPUT); //Erase, Pin 7
        pullUpDnControl(7, PUD_DOWN); //Pull Down Resistor on Pin 7
        pinMode(0,OUTPUT); //Program, Pin 11
        pullUpDnControl(0, PUD_DOWN); //Pull Down Resistor on Pin 11
        pinMode(1,OUTPUT); //Reset, Pin 12
        pullUpDnControl(1, PUD_DOWN); //Pull Down Resistor on Pin 12
        pinMode(2,OUTPUT); //Next, Pin 13
        pullUpDnControl(2, PUD_DOWN); //Pull Down Resistor on Pin 13
        pinMode(8,INPUT);  //Digital Out, Pin 3
        pullUpDnControl(8, PUD_DOWN); //Pull Down Resistor on Pin 3

        //Ground, Pin 6
        //3.3V, Pin 1
        //5V, Pin 2

        //loop to monitor the touch screen
```

```
		while(1)
		{
			UpdateTheScreen();
		}
}

void UpdateTheScreen()
{
		//delay(buttondelay) disables user input while the sensor has activity
		if (genieReadObj(GENIE_OBJ_WINBUTTON,0) == 1)
		{
			genieWriteObj(GENIE_OBJ_LED,1,1); //turn on Activity LED
			Erase();
			delay(buttondelay);
			genieWriteObj(GENIE_OBJ_LED,1,0); //turn off Activity LED
		}
		if (genieReadObj(GENIE_OBJ_WINBUTTON,1) == 1)
		{
			genieWriteObj(GENIE_OBJ_LED,1,1); //turn on Activity LED
			Program();
			delay(buttondelay);
			genieWriteObj(GENIE_OBJ_LED,1,0); //turn off Activity LED
		}
		if (genieReadObj(GENIE_OBJ_WINBUTTON,2) == 1)
		{
			genieWriteObj(GENIE_OBJ_LED,1,1); //turn on Activity LED
			Reset();
			delay(buttondelay);
			genieWriteObj(GENIE_OBJ_LED,1,0); //turn off Activity LED
			genieWriteObj(GENIE_OBJ_LED_DIGITS, 0, channel); //update channel on screen
		}
		if (genieReadObj(GENIE_OBJ_WINBUTTON,3) == 1)
		{
			genieWriteObj(GENIE_OBJ_LED,1,1); //turn on Activity LED
			Next();
			inChannel();
			delay(buttondelay);
			genieWriteObj(GENIE_OBJ_LED,1,0); //turn off Activity LED
			genieWriteObj(GENIE_OBJ_LED_DIGITS, 0, channel); //update channel on screen
		}
		if (genieReadObj(GENIE_OBJ_WINBUTTON,4) == 1)
		{
			genieWriteObj(GENIE_OBJ_LED, 0, 1); //turn on Reading LED
			printf ("\nReading\n");
			Read();
			delay(buttondelay);
			genieWriteObj(GENIE_OBJ_LED, 0, 0); //turn off Reading LED
		}
```

```
}

void Erase()
{
	//Pin 7
	digitalWrite(7, HIGH);
	delay(10);
	digitalWrite(7, LOW);
}

void Program()
{
	//Pin 11
	digitalWrite(0, HIGH);
	delay(10);
	digitalWrite(0, LOW);
}

void Reset()
{
	//Pin 12
	digitalWrite(1, HIGH);
	delay(10);
	digitalWrite(1, LOW);
	channel = 1;
}

void Next()
{
	//Pin 13
	digitalWrite(2, HIGH);
	delay(10);
	digitalWrite(2, LOW);
}

void handle(int sig)
{
	//Timer handler
	running = 0;
}

void Read()
{
	//Pin 3
	signal(SIGALRM, handle); //initiate thread for timer signal
	alarm(readtime); //sample time in seconds
	fp = fopen("data.csv", "a");
	fprintf(fp, "\n\nChannel: %d,\n", channel); //write current channel
```

```
        while(running)
        {
                fprintf(fp, "%d,", digitalRead(8)); //write data to data.csv
        }
        fclose(fp);
        printf("\nDONE reading channel\n");
        running = 1;
}

void inChannel()
{
        //Increment Channel; checks if max channel count is reached
        channel = channel + 1;
        if (channel == maxchannel)
        {
                Reset();
        }
}

int main()
{
        fp = fopen("data.csv", "w");
        fprintf(fp, "START MEASUREMENT @ 1 MHz Sample Rate for %d seconds,",readtime);
        fclose(fp);
        wiringPiSetup();
        SetupGenie();
        return 0;
}
```

Appendix 6 - Data sheet for Raspberry Pi



EEWeb EMBEDDED DEVELOPER

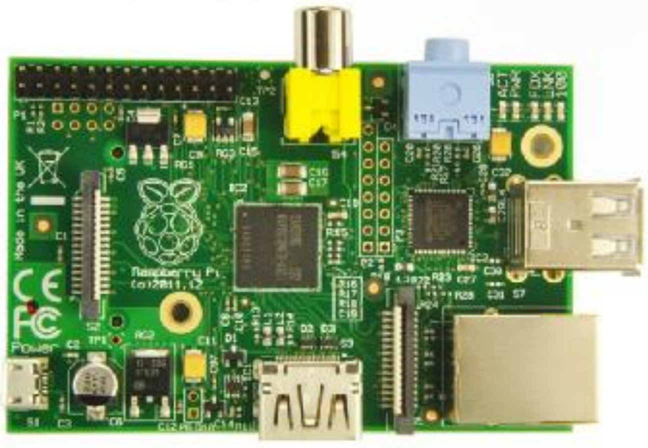
The Raspberry Pi Primer

Part 1: Introduction and Required Hardware

by Kyle Olive, EEWeb Contributing Author

Recently, interest in hobby electronics has grown dramatically. With platforms like the Arduino gaining popularity and achieving wide success in various retail markets, it's no wonder that these kinds of hobby electronics have become more prevalent. While the Arduino is designed around a microcontroller like the Atmel AVR and allows hobbyists to acquire and build various shields and add-ons for more functionality however, the Raspberry Pi (though a similar piece of hobbyist electronics) takes a different approach.

Rather than a microcontroller board, the Raspberry Pi is a complete computer about the size of a playing card. At its core is an ARM11 microprocessor, and it contains HDMI and audio output, 8 GPIO Pins, 2 USB Ports, Ethernet, a dedicated GPU, and more. It's basically a standalone platform that you can fit in your pocket. The introductory model costs only $25 (the higher end model with more memory runs for $35).



Front view of the Raspberry Pi board.

The Pi is capable of running various distributions of Linux, and using it is very straightforward and easy. In fact, the idea behind the Raspberry Pi was to provide a cheap computer platform for schoolchildren to use as an educational tool. It was to be a tool that students with little to no experience using computers could use to learn more about programming, computers, and really any other topic that lends itself well to electronic education. The platform quickly took off however with hobbyists, and it even began to be used in some commercial applications. The device's low price point, small size, relative power, and ease of use makes it an interesting and flexible development platform.

Because the Raspberry Pi runs Linux, developers who are used to working in a Linux environment can easily get up to speed on the Pi, and they quickly come up with ideas and applications for the device. Many of the popular software packages available for traditional Linux desktops are available on ARM versions of Linux meant for use on the Raspberry Pi, and with the incorporation of a dedicated GPU, multimedia applications that were often limited to desktop environments or specialized embedded platforms are completely viable, and cheap. Finally, the large network of developers provides a good support network for those who are stuck on a problem, or looking for advice — a benefit that some hobbyist platforms and specialized platforms lack.

Even if you're not a hobbyist, and looking to develop a new embedded solution for a particular problem, the Raspberry Pi can be a good way to prototype a solution without having to go through the entire design and development process. Rather than spending a large amount of money and time into developing a custom built functioning prototype, you can use a Raspberry Pi to greatly reduce initial prototyping costs and get a feel for a potential idea.

If you're going to do development with a Raspberry Pi, you're going to need some hardware. Below is a discussion of various pieces of hardware that you'll need to get started using a Raspberry Pi, and a brief note on its purpose. Most of the parts are easy to acquire at your local electronics store, but the Raspberry Pi vendors listed below will also often sell you this equipment with your Raspberry Pi.

Raspberry Pi's low price point, small size, relative power, and ease of use makes it an interesting and flexible development platform.

The Raspberry Pi – You can't have a Raspberry Pi project without a Raspberry Pi. As discussed above, the Raspberry Pi is a playing-card sized computer that starts at $25. You can get these from various locations, but some vendors that are listed on the Raspberry Pi Foundation's website (www.raspberrypi.org) are:

- Element14
- Allied Electronics

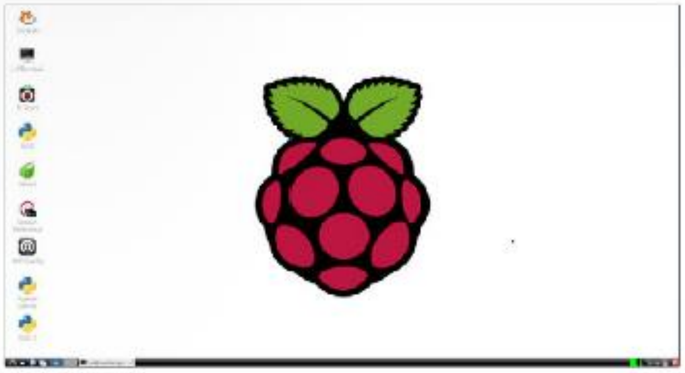
An SD Card – You'll need a regular old SD card to store the operating system and any other files you'll want to use on your Raspberry Pi. 2GB is the minimum size of SD card you'll need to run the default operating system, Rhaspbian, but if you are going to be using the Pi heavily, you'll probably want a bigger SD card to suit your needs. In that case, I would probably recommend at least an 8GB SD card. You can get one at your local electronics store, or bundled with a Raspberry Pi at the above listed vendors.

An HDMI Cable – If you plan on doing more than SSH'ing into your Raspberry Pi, you'll want an HDMI cable (or an HDMI-to-something adapter) so you can view the device video and audio output. You can get one at your local electronics store, or bundled with a Raspberry Pi.

An SD Card Reader/Writer – If you're working on a laptop, odds are you'll already have an SD Card slot to work with. If you don't, you'll have to get your hands on an SD Card Read/Write device that you can plug into your computer. You'll need it to flash the SD Card with whatever operating system you're going to be using. If you don't plan on ever needing to reflash your Raspberry Pi, you can buy a pre-formatted SD Card from one of the vendors listed above. Otherwise, you can usually pick up an SD Card Reader at your local electronics store.

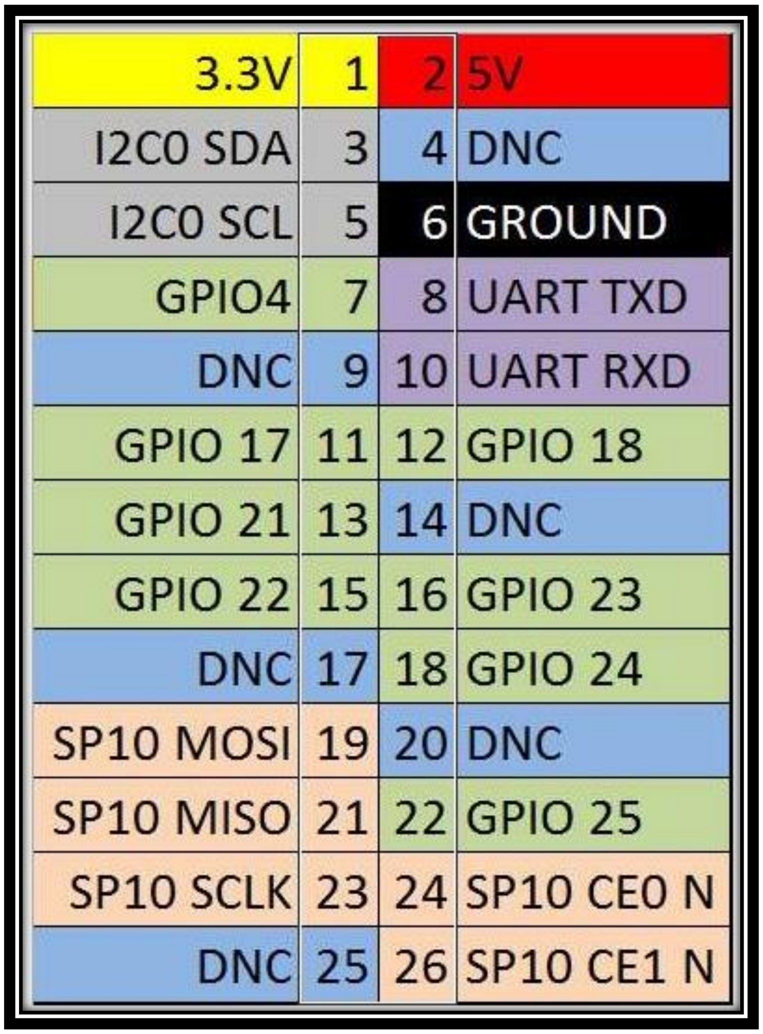
A Micro-usb Power Supply – The Raspberry Pi is powered by 5V Micro-usb, so you'll need to get a power supply. The manual recommends against powering your Raspberry Pi directly from the USB port of your computer, but unless you're working with a lot of high power peripherals or USB hubs you'll probably be OK doing so. In either case, the safest bet is to buy a power supply either at one of the retailers above or get a micro-usb power supply (the one used for your cellphone or tablet might be able to power the Raspberry Pi).

Ethernet Cable – You'll want an Ethernet cable for your Pi so you can give it an Internet connection; you'll at least need to set it up on a local network for remote access. The cables are pretty cheap, and you probably have an extra one lying around somewhere. If you don't you can pick one up online with your Raspberry Pi or at a local retail store. USB Keyboard/Mouse – This is fairly straightforward. You'll need a keyboard and mouse to develop on the Pi.


The default desktop of the Raspbian Wheezy Operating System running on a Raspberry Pi

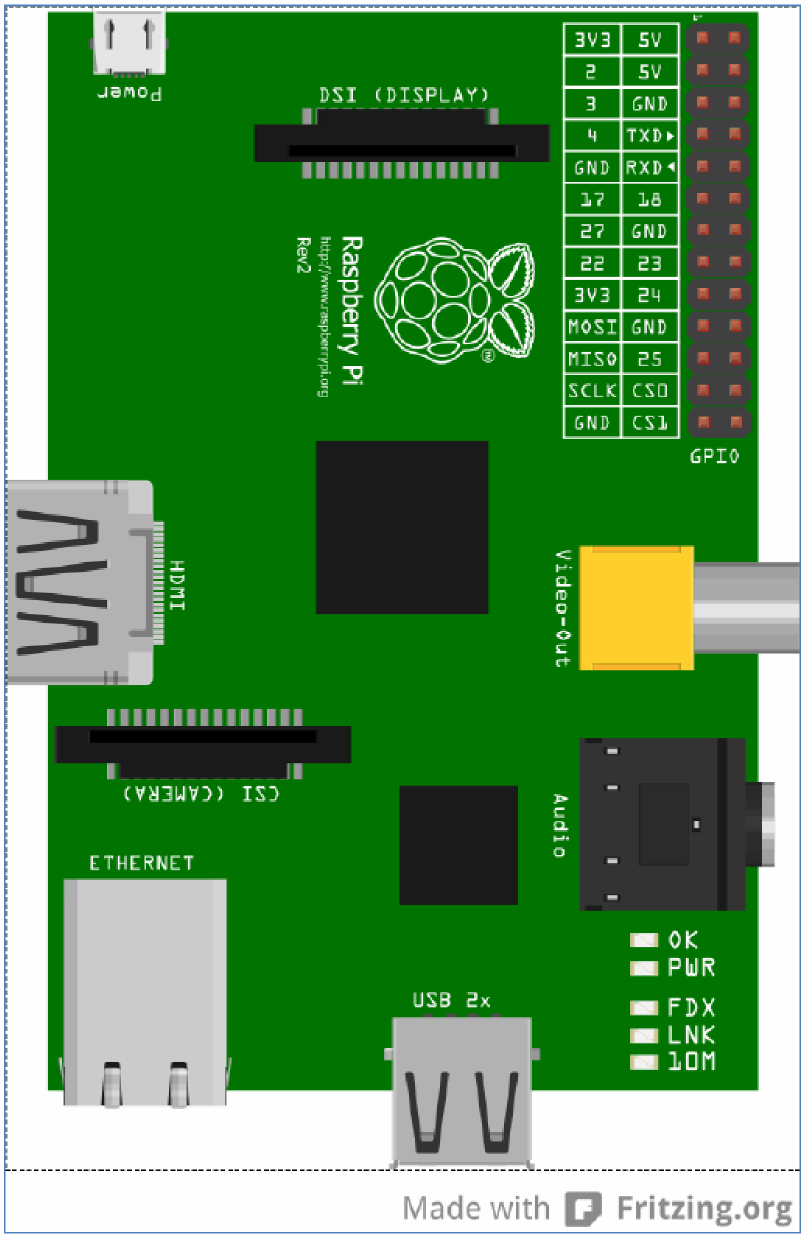
The parts listed are what you'll need to get started doing anything with your Raspberry Pi. Once you've gotten a hold of these items, you can start looking into developing the next cool Pi Project. Intrepid developers have created everything from robots to low-powered FM transmitters and webservers; they've even creating mini-supercomputers by parallelizing many Raspberry Pis. ■

Appendix 7- Raspberry Pi Physical Characteristics

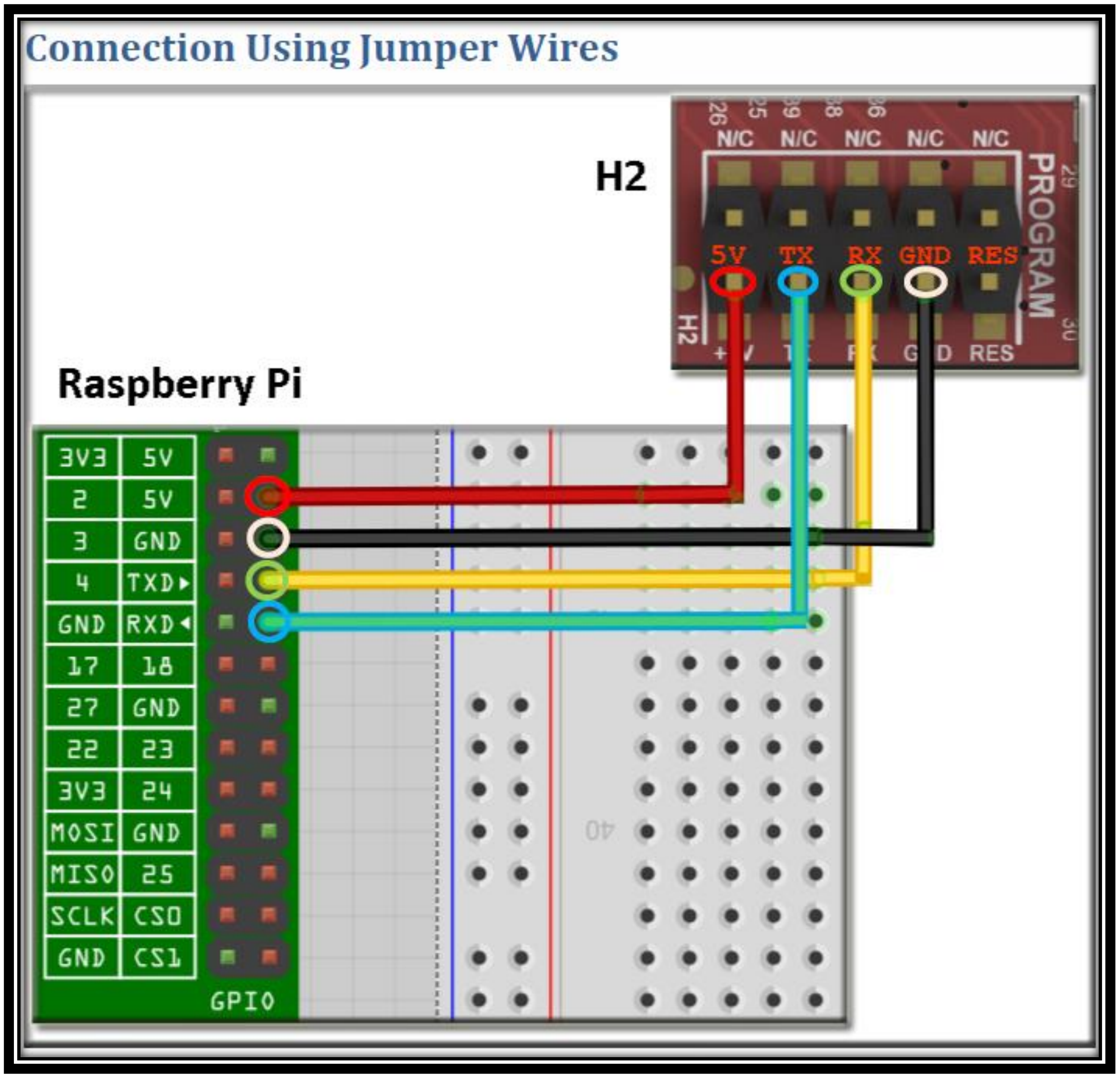


3.3V	1	2	5V
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SP10 MOSI	19	20	DNC
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
DNC	25	26	SP10 CE1 N

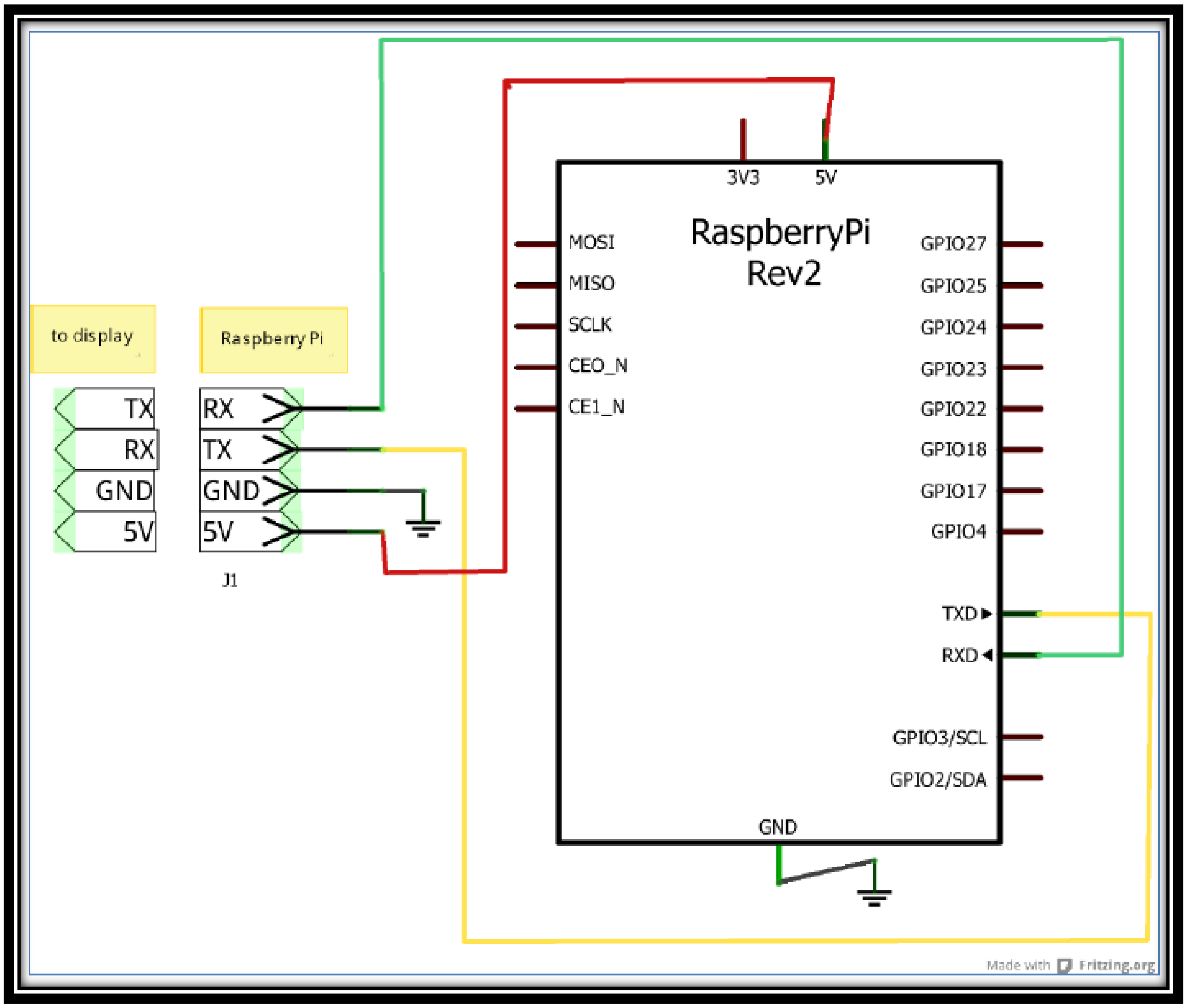
GPIO Pinout


Power
DSI (DISPLAY)
Raspberry Pi
http://www.raspberrypi.org
Rev2
3V3
5V
2
5V
3
GND
4
TXD
GND
RXD
17
18
27
GND
22
23
3V3
24
MOSI
GND
MISO
25
SCLK
CS0
GND
CS1
GPIO
HDMI
Video-Out
CSI (CAMERA)
Audio
ETHERNET
OK
PWR
FDX
LNK
10M
USB 2x
Made with Fritzing.org

PCB Layout



4D Systems Screen Pinout



Touchscreen and Raspberry Pi Connection

Appendix 9 - High Level Overview of Sensor Operation

Monitoring of Repeated Head Impacts using Time-dilation based Self-powered Sensing

Kenji Aono, Tracey Covassin and Shantanu Chakrabartty
Department of Electrical and Computer Engineering
Department of Kinesology
Michigan State University
East Lansing, U.S.A.
{aonokenj,covassin,shantanu}@msu.edu

***Abstract*—Measuring head impacts in helmeted sports is important for prognosticating onset of mild traumatic brain injuries (MTBIs) or concussions. In this paper we present a miniature battery-less, self-powered sensor that can be embedded inside sport helmets and can continuously monitor and log the statistics of different levels of helmet impacts. At the core of the proposed sensor is a novel time-dilation circuit which allows measurement of the high-levels of impact energy. An array of linear floating-gate injector is used for storing the location of the sensor on the helmet and for logging the statistics of helmet impacts which can be retrieved using an external plug-and-play reader. Measured results from prototypes fabricated in a 0.5 μm CMOS process validate the functionality of the sensor when subjected to controlled drop tests.**

I. Introduction

Understanding the relationship between head impacts in helmeted sports like American football and the risk of concussive and long-term brain injury is an active area of research [1]. While there are significant disagreements in literature about the relevance of different impact parameters (linear acceleration, rotational acceleration, location and time of impact, history of impacts, etc.) on prognosticating concussions, a common agreement has been the need for measuring and recording head impact data during the normal course of play [1]. The most popular approach in this regard has been to embed battery-powered accelerometers inside helmets and estimate the inertial response of the human brain during the impact [2]. While this is an attractive solution for controlled studies, the use of batteries increases the risk of leakage (due to high-acceleration impacts) and increases the overhead of routine helmet maintenance and recharging. In this paper, we explore a self-powering approach for monitoring helmet impacts by harvest operational energy from the impact itself.

The challenge for designing self-powered sensors for helmets is that the level of acceleration (and hence the energy) during impacts could easily exceed 100g [2]. This is depicted in Fig. 1, which illustrates the impulsive nature of the signal subjected on a helmet sensor. To prevent damage to the sensor, a typical architecture will dissipate most of the impulse energy through over-voltage protection circuits; sensors would be rendered unable to accurately measure the magnitude of energy, and hence the acceleration of the impact [3]. Therefore, we propose a time-dilation approach where the impulsive signal is stretched out in time (as shown in Fig. 1) while retaining its energy content (area under the curve). In this manner, the sensor can be operated within the levels of electronic compliance and safety. However, this mode of self-powering introduces additional challenges which will be discussed in section II, where we present the principle of operation and schematic of a time-dilation circuit.

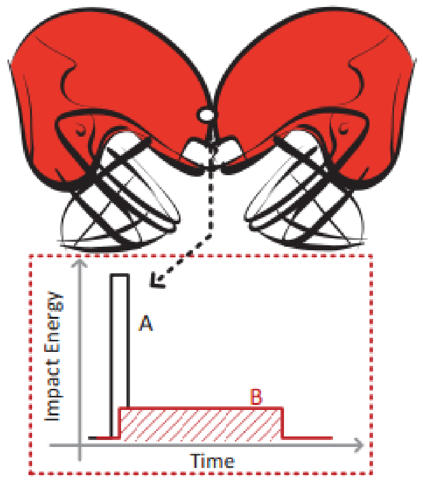


Fig. 1. Impulsive nature of the head impacts and the proposed time-dilation method to spread the energy over time while operating the sensor electronics within the compliant voltage levels.

II. Time-dilation Circuit

Fig. 2 shows the equivalent circuits of a typical piezoelectric self-powered sensor with and without the proposed time-dilation circuit. In both the circuits, the piezoelectric transducer is modeled by a simple current source I_p to model the impulse current generated due to the mechanical impact, and a capacitance C_p which models the mechanical stiffness of the transducer. I_L models the load current of the sensor, C_L the load capacitance, and D_L represent a zener diode which models the over-voltage protection circuits. In practice, the current I_L is triggered only when the voltage V_{out} exceeds a minimum threshold level V_{min}. The resistors R_1 and R_2

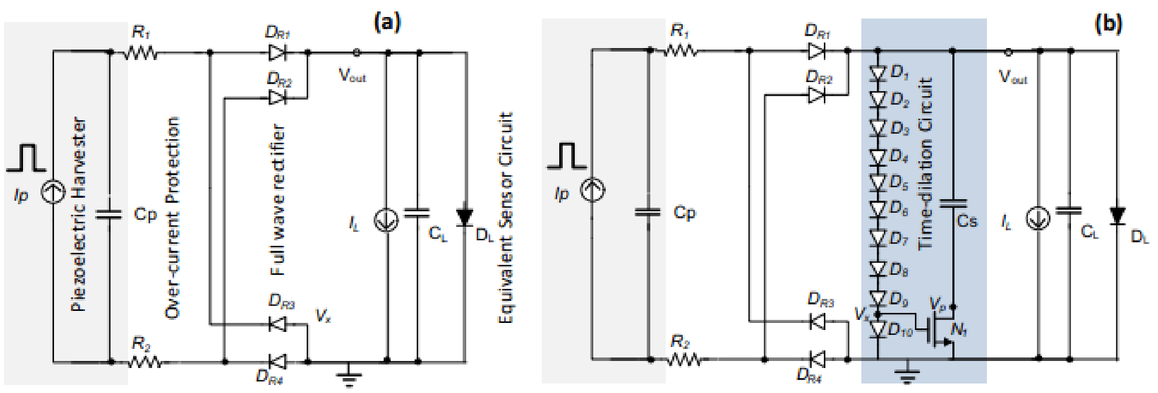


Fig. 2. Equivalent circuit of a self-powered sensor: (a) without, and (b) with the time-dilation circuit.

implement an over-current protection circuit and the full-wave rectification is achieved using the diodes $D_{R1} - D_{R4}$ formed using a bulk-driven cross-coupled pMOS circuit which was reported in [3].

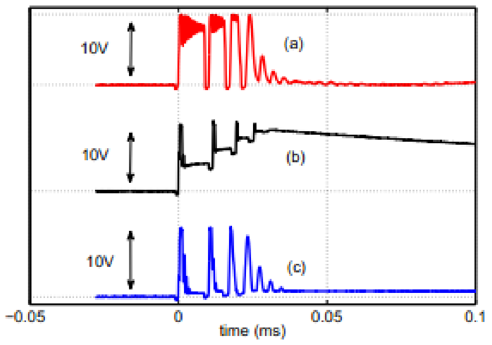


Fig. 3. Measured results showing $V_{\rm out}$ for the sensor with: (a) no time-dilation circuit; (b) time-dilation circuit with $C_S = 50$ nF; and (c) time-dilation circuit with $C_S = 1$ μF.

For the circuit without the time-dilation (shown in Fig. 2(a)), the rectified current of I_p increases the voltage $V_{\rm out}$ past the minimum threshold $V_{\rm min}$ which then activates the sensor and hence the current I_L. If the current I_p exceeds the sensor current I_L, the voltage $V_{\rm out}$ will increase until the zener, $D_{\rm L}$, starts drawing the extra current. Thus, for large impulse currents most of the impact energy is dissipated through the zener. The time-dilation circuit shown in Fig. 2(b) mitigates this problem in the following manner. Initially the sensor voltage $V_{\rm out} = 0$ implies that the nMOS transistor N_1 is OFF and hence the storage capacitor C_S is disconnected. Therefore, during start-up the behavior of circuit Fig. 2(b) is identical to the start-up for the circuit in Fig. 2(a). But when the voltage $V_{\rm out}$ exceeds $KV_{\rm th}$, with K being the number of diodes in the chain D_1-D_K and $V_{\rm th}$ being the threshold voltage of the transistor N_1, the storage capacitor C_S is connected in parallel with the sensor. Thus, any extra piezoelectric current now charges up the storage capacitor without dissipating through the zener. When the impulse is over (after the impact) the charge stored on the capacitor C_S drives the sensor current until $V_{\rm out}$ falls below the activation voltage, $V_{\rm min}$. Thus, using time-dilation, the sensor can more accurately measure the level of impact energy.

Fig. 3 shows measured results from a fabricated prototype (described in later sections) comparing the voltage $V_{\rm out}$ under three different conditions: (a) without any time-dilation, based on the schematic shown in Fig. 2(a); (b) time-dilation using $C_S = 50$ nF; and (c) time-dilation using $C_S = 1$ μF. As shown in Fig. 3(a), the zener clips the output voltage at $V_{\rm max} = 10$ V and the sensor shuts down after the impulse decays. Where as for the time-dilation circuit in Fig. 3(b), the storage capacitor C_S holds the extra charge. Note that in Fig. 3(b), the start-up response remains unaffected which is important for logging different levels of impact. The result in Fig. 3(c) shows that choosing the right range of values for C_S is important as a large storage capacitor (even though it can store more charge) will take a longer time (hence duration of impulse event) to push the voltage $V_{\rm out}$ beyond $V_{\rm min}$ and activate the sensor.

III. Self-Powered Energy Measurement and Data-Logging

The magnitude of impact can be determined by estimating the amount of charge that is deposited on the storage capacitor. This in turn can be determined by the time it takes to discharge the storage capacitor. Also, for self-powered operation the measured data has to be stored on a non-volatile memory for subsequent retrieval. Linear time-measurement on a non-volatile memory can be readily implemented using our previously reported linear floating-gate injector topology [4]. The circuit level schematic of the linear floating-gate injector is shown in Fig. 4. The circuit consists of a floating-gate pMOS transistor $M_{\rm fg}$ whose source is driven by a constant current source $I_{\rm ref}$ which is powered by either a piezoelectric transducer (during battery-less operation) or by an energy

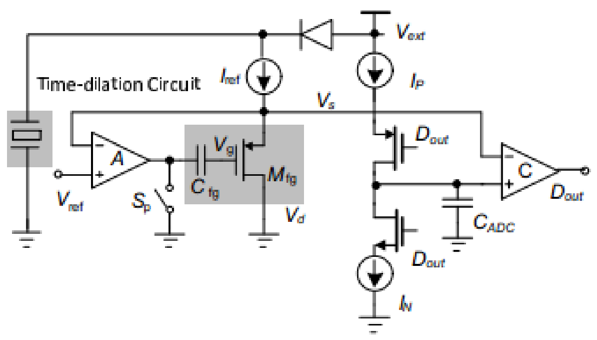


Fig. 4. Schematic of the data-logging circuit with a floating-gate based linear injector and a spiking analog-to-time converter

source V_{ext} (when the data is being retrieved by a reader). Note that the energy sources are isolated by a diode which allows V_{ext} to supersede the signal generated by the piezoelectric transducer. The opamp A (connected to V_{ref}), the constant current source I_{ref}, and the floating-gate transistor M_{fg} form a negative feedback configuration when the switch S_{P} is open, thus ensuring that the source V_s and gate V_g voltages remain constant. Since the drain voltage of M_{fg} is tied to ground, if the reference voltage V_{ref} exceeds 4.2V, the negative feedback will cause the source-to-drain voltage of M_{fg} to also exceed 4.2V. In such a case, hot-electrons are generated in the channel of M_{fg} due to impact ionization; when the electron energy exceeds the gate-oxide potential barrier ($\approx$3.2eV) they can get injected onto the floating-gate. Note that the polysilicon gate of the pMOS transistor is electrically insulated by silicon-dioxide (hence the name "floating-gate"), therefore, any electron injected onto the gate is retained for a long period of time. Because all terminal parameters of the floating-gate transistor are held constant during the injection process, the injection current I_{inj} remains constant. Thus, the amount of charge injected onto the floating-gate, or the decrease in floating-gate voltage V_{fg}, is proportional to the duration for which the source current I_{ref} is activated and S_{P} is open. This can be expressed as

$$\Delta V_{\text{fg}} = \frac{1}{C_T}\int_0^T I_{\text{inj}} dt = \frac{I_{\text{inj}}}{C_T}\tau(T) \qquad (1)$$

where τ is the duration of injection and C_T is the total floating-gate capacitance which includes C_{fg}, tunneling capacitance, and other parasitic capacitances associated with the floating node. The change in floating-gate voltage ΔV_{fg} is measured by closing the switch S_{P} which breaks the feedback loop by shorting the other terminal of C_{fg} to ground. Because the source current I_{ref} is constant, $\Delta V_s = \Delta V_{\text{fg}}$ which is read-out using a spiking analog-to-time converter as shown in Fig. 4. The voltage V_s is used as a voltage reference to a comparator whose output D_{out} periodically toggles the pMOS and the nMOS switches, one at a time. Thus the current I_P charges the capacitor C_{ADC} to the voltage V_s after which the current I_L is turned ON which discharges C_{ADC} and generates a spike, as shown in the measured response. The time-difference between the two spikes is proportional to the voltage V_s and is used to determine the floating-gate voltage V_{fg}.

IV. Measurement Results

A prototype sensor IC has been fabricated in a 0.5μm CMOS process and its micrograph is shown in Fig. 5(a). It consists of 21 linear floating-gate injector channels where 14 of the channels are programmed to trigger at different levels of impacts. The 7 remaining channels are used for storing: (a) an identification code for the sensor; and (b) placement location on the helmet. The sensor IC also integrates functional modules that are required for programming the linear injectors and retrieving data using an external reader. The circuit level implementation for many of the modules has been reported [5] and is omitted here for the sake of brevity. Fig. 5(b) shows a football helmet (manufactured by Riddell Inc.) which was used for testing the fabricated sensor. A fully integrated sensor board is shown in Fig. 5(c) which hosts the sensor IC, the time-dilation capacitor, the piezoelectric interface and the programming interface. The size of the sensor is 2 cm $\times$ 1.5 cm and easily fits in between the helmet's cushion pads.

The measurement setup emulated at a smaller scale, the drop-test procedure reported in [6]. The helmet with the integrated sensor was dropped from two height levels: Height A (1 foot) and Height B (2 feet). Fig. 6 shows the output of the piezoelectric transducer (without the sensor attached) when the helmet is dropped from heights A and B. The response clearly shows the impulsive nature of the piezoelectric signal generation and the voltage level clearly shows the need for the time-dilation approach. Fig. 7 shows the data recorded from the sensor when the helmet is dropped repeatedly from 1 feet. Note that even at this height the first three channels record the level of impact indicated by the change in their output voltage. The linearity of the response shows that the voltage measurements could be calibrated to different impact energy levels with corresponding acceleration levels. Fig. 8 shows data recorded from the sensor when multiple helmet drops from 3 different heights. The Control signifies when the helmet was just tapped while at rest. The result shows the sensor's gain changing with at various heights, validating the sensor's ability to log diverse levels of impact.

V. Conclusions

In this paper we presented a miniature sensor system that can be used to monitor the level and frequency of head impacts in helmeted sports. A time-dilation circuit enables the sensor to measure high energy impulses and a linear floating-gate injector enables the sensor to record data on non-volatile memory. The sensor is self-powered and operates by harvesting the energy from the head-impact with no need for batteries. The small form-factor and low-cost of the sensor enables it to be embedded at multiple places inside the helmet, providing MTBI researchers mapping data to effectively prognosticate concussions during the course of normal play.

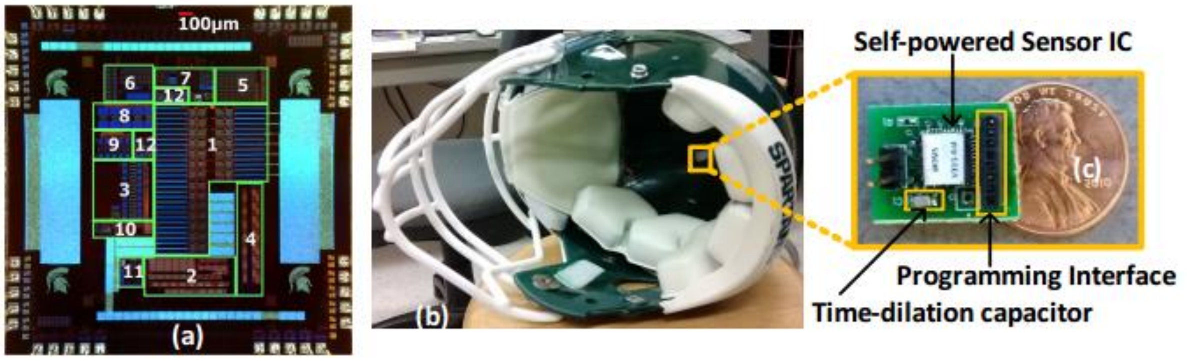


Fig. 5. (a) Micrograph of the sensor IC integrating different modules: 1. Floating Gate Array; 2. Digital Decoder; 3. Tunneling Voltage Charge-pump; 4. Level Shifter; 5. Injection Control; 6. Diode Protection & Rectifier; 7. Voltage References; 8. Tunneling Charge-pump; 9. Injection Charge-pump; 10. Ring Oscillator; 11. Analog-to-time Converter; 12. Supporting circuitry, power-on reset, buffers, etc.; (b) test helmet used for the impact measurement study; and (c) the sensor board hosting the sensor IC and the time-dilation capacitor.

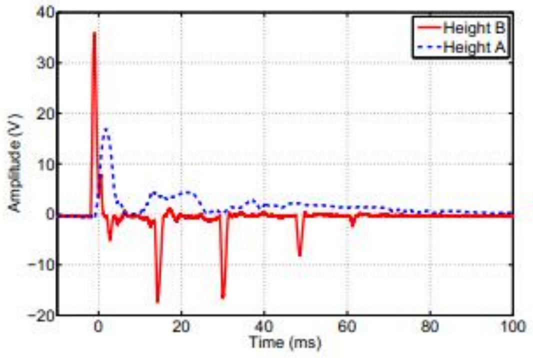


Fig. 6. Signal recorded at the output of a PZT-5H piezoelectric transducer (10MΩ load) when the helmet is dropped from 1 foot and 2 feet respectively.

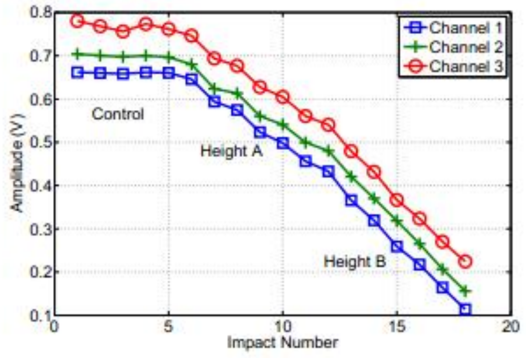


Fig. 8. Measured output from three of the sensor channels when the helmet is repeatedly dropped from different heights.

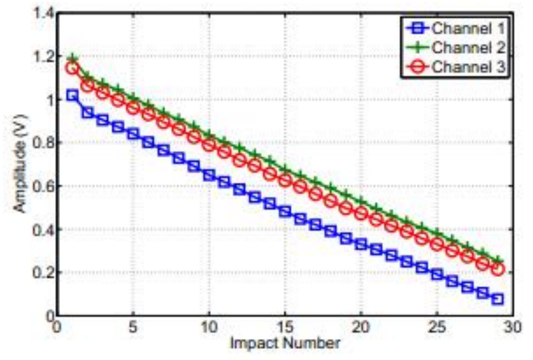


Fig. 7. Measured output from three of the sensor channels when the helmet is repeatedly dropped from 1 foot.

References

[1] Past, present and future of head injury research, *Excercise Sport Sci. Rev.*, vol.39, no:1, 2011.

[2] S. M. Duma et al., "Analysis of Real-time Head Accelerations in Collegiate Football Players," *Clin. Journal Sport Medicine*, vol. 15, no. 1, 2005.

[3] P. Sarkar and S. Chakrabartty, "Compressive Self-powering of Piezo-Floating-Gate Mechanical Impact Detectors", *IEEE Transactions of Circuits and Systems-I, (TCAS)*, vol. 60, no. 9, 2013.

[4] C. Huang, P. Sarkar and S. Chakrabartty, "Rail-to-Rail Hot-electron Injection Programming of Floating-gate Voltage Bias Generators at a Resolution of 13bits", *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, Nov. 2011.

[5] C. Huang and S. Chakrabartty, "An Asynchronous Analog Self-powered Sensor-Data-Logger with a 13.56MHz RF Programming Interface," *IEEE Journal of Solid-State Circuits*, Feb 2012.

[6] S. Rowson and S. M. Duma, "Development of the STAR Evaluation System for Football Helmets: Integrating Player Head Impact Exposure and Risk of Concussion," *Annals of Biomedical Engineering*, vol. 39, no. 8, 2011.