Non-Contact Vital Signs Monitoring System

ECE4012 Senior Design Project

Section L4A, Vitals Check
Project Advisor, Dr. Zhang

Alec Adamski (alec.adamski@gmail.com)
Rohan Iyengar (rohankiyengar@gatech.edu)
Kedar Manishankar (kedarjmfl@gmail.com)
Sai Sathiesh Rajan (srajan9@gatech.edu)

Submitted

2017 May 3

Executive Summary

Most vital signs monitoring systems require patients to strap into the device so that it can measure their heart rate, respiration rate, and blood pressure. A noncontact vital signs monitor does not require the user to touch the device. Instead, a radar transmits a signal to the patient which reflects and modulates. By analyzing vital sign information with a demodulation process, specifically the overall signals and transient, the heart rate and respiration rate can be deconstructed and examined separately. This concept is not novel, but has aspects that can be improved to provide more information to the user in a more intuitive manner. Some features, such as adding a user friendly interface with the ability to vary sampling rate and acquisition time, will increase the appeal and ease of use. The radar transmitter and receiver have already been implemented, so this project focused on sampling, processing and visualizing the data that has been recorded. This project did not require any funding other than that of a microcontroller and external ADC evaluation board to sample and transmit the analog signal received from the radar. The resulting product is able to read a patient's respiration rate, and display the it on a user friendly computer interface. The incoming signal must be displayed in real time, and the processing must be done immediately after so that a patient can view his or her vital signs immediately. Although the external ADC was unable to be completed, the product still was able to showcase some of its most important aspects, such as real time display and advanced signal processing.

Table of Contents

Non-Contact Vital Signs Monitoring System

1. Introduction

The Non-Contact Vital Signs Monitoring System Team requested $175.34 to build the prototype of a device that can accurately sample and analyze a given analog signal containing vital signs such as respiration and heart beat. The analog system is sampled from the existing radar system using the prototype parts requested. The system consists of a microcontroller unit (MCU), an external analog to digital converter (ADC), and a computer user interface (UI).

1.1 <u>**Objective**</u>

The team designed and prototyped a system that can accurately sample and process vital sign signals to produce data such as heartbeat and respiration. In addition, the final product displayed information of physical conditions in real-time, without contact with the patient. Low-noise detection system to acquire vital signs exist and is used to obtain an analog signal containing information of heartbeat and respiration. An MCU samples the signal, and converts the incoming analog signal to digital for transmission to a computer. The raw digital data is processed to deconstruct the original signal into two separate signals, containing heartbeat and respiration respectively. The computer UI provides necessary functionality to display the respiration and heart beat signals, as well as store the large amounts of digital data which needs to be processed.

1.2 Motivation

Noncontact vital sign monitoring is a promising method in the field of home health care and biomedical monitoring. Real-time comparison and analysis of the obtained respiration and heartbeat signal of the patient, with that of an expected respiration and heartbeat signal is critical to search and rescue operations. Our team built upon an existing practical, low-noise detection system that detects vital signs, in order to create a commercial product that is user friendly and accurately displays respiration signals. This system is geared towards patients with disabilities, senior citizens as future improvements to our app can include automatic distress calls for abnormal physical activity.

1.3 Background

Noncontact vital sign monitoring systems utilize radar signals to determine the frequency and transient features of chest movements. There are existing algorithms to perform Doppler processing to convert seemingly random data into meaningful information [1]. The current system setup uses a Doppler radar because of its better system complexity and high distance resolution, but it cannot return the distance information. This type of radar still does have the capability to extract the frequency and transient vital sign signals.

2. Project Description and Goals

The broad goal for this project was to create an end-to-end system that samples data from the existing system in a way that is controlled by a user or team member and then analyzes this data to provide vital sign information, specifically heartbeat and respiration rate. Breaking down this desired image of the end product, the goals set for this project were:

1. Utilize a microcontroller to transfer signal data from the existing system to the computer using a rigid two-way protocol.
2. Maintain a high level of resolution on the incoming data, specifically 16 bit resolution.
3. Create an interactive computer program through which a user can control the system function and perform a single use of the noncontact system.
4. Use the incoming data to find respiration and heart beat information for the subject based on a single system execution.
5. Create a mobile application to allow a user to track a history of measurements from the noncontact system.

All goals were at least partially achieved, as demonstrated through the final product. Goal one was achieved, but with a lower sampling rate than desired. In addition, the communication was a one way stream instead of a controllable two way protocol. Goal two was not achieved in time for demonstration as the external ADC was not integrated fully with the system, and several workarounds had to be employed in the other parts of the project to compensate for this. Goals three and four were completed on the computer, through the use of the Python based GUI and MATLAB based signal processing functions. Goal five was abandoned early on in this iteration of the project as the team had one less group member than expected, and thus the project required addition responsibilities for each remaining member.

3. Technical Specifications & Verification

Description	Initial Proposal	Final outcomes
Sampling rate	>1000 samples/second	630 samples/second
Number of sampling channels	>2	1
Microcontroller to Board controller interface	UART	UART
Power Supply	3.3 V	4.8 V
ADC	User controlled sampling rate	N/A
Signal Display	Real-time plotting	Real-time plotting
Resolution of ADC	16 bits	12 bits

4. Design Approach and Details

4.1 Design Approach

4.1.1 Microcontroller Unit

Low-noise detection system to acquire vital signs exists in this project and is used to obtain an analog signal containing information of heartbeat and respiration. A microcontroller unit(MCU) samples the signal at a specified rate, and converts the incoming analog signal to digital for transmission to a computer. The EFM8 MCU has a UART interface as well as a successive-approximation-register(SAR) ADC with variable 12-, 10-, and 8- bit modes which will aid in user controlled sampling rate of the vital signals [2]. EFM8 Busy Bee MCU costs approximately $45.62 and combined with the adequate features incorporated in the MCU, ensures that we will incur low prototype costs and lower development costs [3].

Due to inability in configuring the external ADC, the on-board ADC of the MCU was utilized instead. Initially 16 bits of resolution were required to detect the low-voltage heartbeat signal, and two channels to obtain the incoming analog I/Q signals, however the on-board ADC has a maximum of 12 bits of resolution, and can only accept one input channel. For the UI to access the processed data directly from the MCU, each sampled point needed to send five bytes of data to the PC- four bytes for the four digit value of the voltage in millivolts, and one byte to comma separate all the values. Due to the on-chip ADC not being able to read negative values, the overall signal was increased by 1.5 V, by connecting a -1.5 V DC power supply to the reference ground.

4.1.2 **External Analog to Digital Converter**

In order to achieve the desired 16 bits of resolution, the project required an external ADC to sample both channels of the pre-existing system. The ADC chosen was the AD7770, which has 24 bits of precision and 8 channels that can be sampled [4]. This ADC was bought as a part of an integrated evaluation board, Analog Device's EVAL-AD7770FMCZ [5]. This evaluation board contained the necessary headers and connections to connect the pre-existing system directly to the board and send data to the MCU. The ADC should receive a signal from the MCU to begin sending data, which would then be sent to the PC as seen in Figure 1. Unfortunately, the ADC proved to be more complicated than anticipated and it was not completed by the time of Senior Design Expo. More work has been done on the ADC since then; the ADC sends data to the MCU, but it is unable to interpret this data or send it to the PC. Further work must be done to time data reads correctly to separate both channels of the ADC. In addition, the UI will need to be reconfigured to show two channels of data instead of just one.

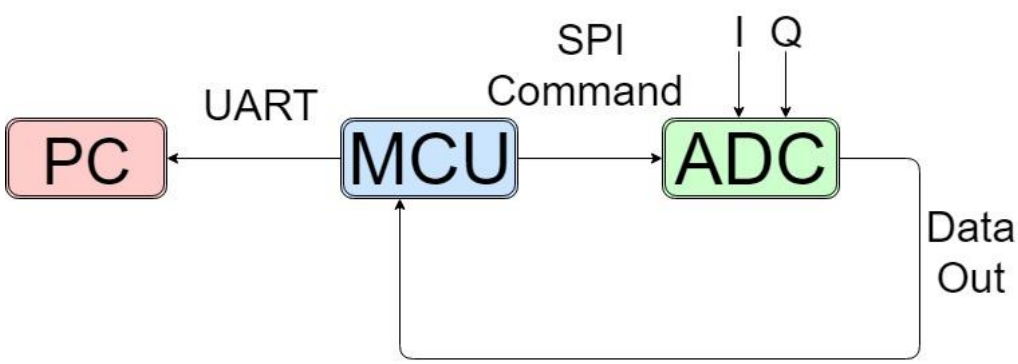


Figure 1. Block diagram depicting flow of overall system

4.1.3 **Digital Signal Processing**

The baseband signal obtained from the phase modulation of the wireless radar signal which is comprised of two components, heartbeat and respiration. Firstly, the signal from the microcontroller was passed to the computer before performing computations in MATLAB. The signal itself consisted of two channels, I channel and Q channel. The data from the two channels were combined using Matlab in order to construct a more robust signal resilient to noise. Then, in order to perform meaningful analysis on the two source signals, the frequencies of the respective signals were determined with the aid of a Fast Fourier Transform (FFT). Since the respiratory rate is the most obvious large amplitude feature in the FFT barring the DC component, the frequency of the respiration was identified with the aid of a peak finding function. Once the respiratory rate was identified, the heart rate could be determined by accounting for the harmonics of the respiratory rate. Once the frequencies were identified, two bandpass filters centered around the respiratory rate and the heart rate were constructed to isolate the transients of the respiration and the heartbeat respectively. The transients were then sent to the GUI along with the determined respiratory and heart rate.

4.1.4 **User Interface**

To control the entire system and coordinate communication between the microcontroller and PC, a user interface was required. This includes a graphical user interface (GUI) that contains a visual interactive tool to use a well as the communication and data processing code working in the background (backend). All code was written in the Python programming language, version 2.7.x, using the 64-bit version.

The graphical user interface uses the Tkinter framework to construct the visual elements. There are three screens that go in order. The first screen allows the user to specify a sampling rate and duration to measure data values for. The sampling rate field is not used because the function was intended to be used for down sampling. The current sampling rate is too low to warrant down sampling so this feature is not implemented. After entering these parameters, the next screen displays a graph of the signal in real time, updating every half second. The last screen takes displays the processed graphs using MATLAB to get both the heart rate and respiration rate values and graphs. There is also a filtered signal displayed on this screen due to the noise issues. Screenshots of each screen are displayed below.

The backend of the GUI is primarily comprised of communication and data structures that are continually updated with signal data. To accomplish real time plotting and communication, multithreading is used while carefully monitoring race conditions in the code. The signal data array is updated continually with new data from the channel, also updating the time values as data comes in. This use of threading to update the arrays allows for the screen to update in real-time without freezing the program as updates are done concurrently to the screen graphics. Communication is performed using the pyserial library to communicate via the COM port, only reading the COM channel due to the microcontroller code limitations. At the end, the data is written to a CSV file in the format that has time and each channel read as columns. Figures 2-4 demonstrate how each page of the GUI look.

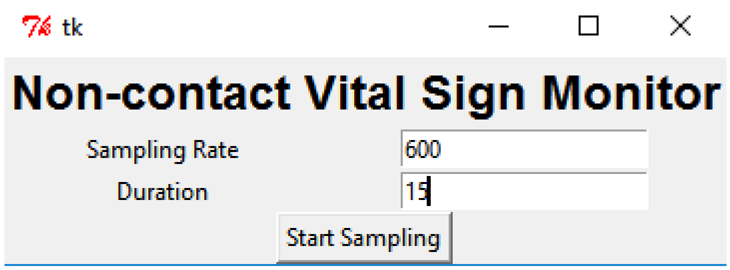


Figure 2. First Screen of GUI

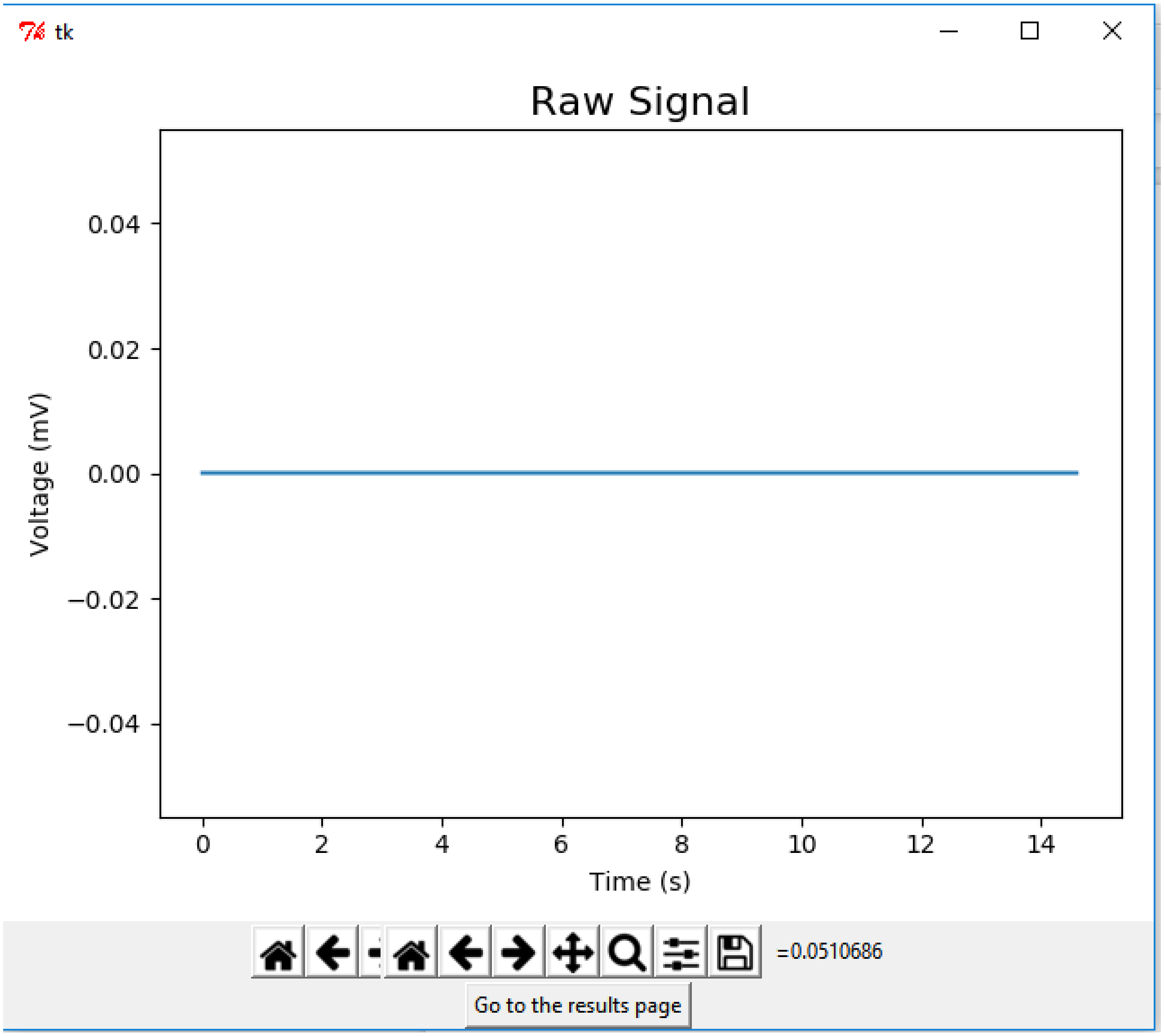


Figure 3. Second Screen of GUI

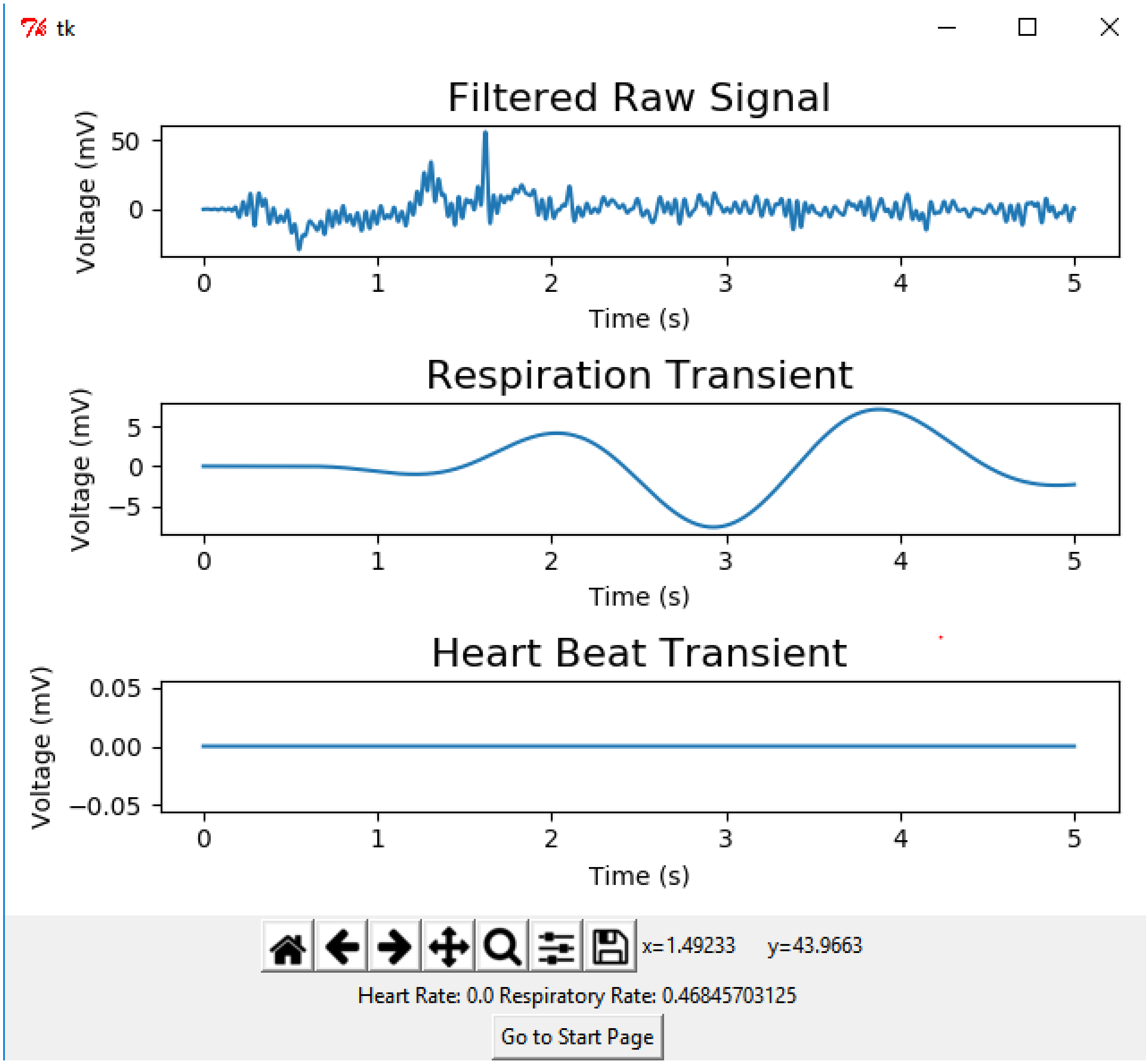


Figure 4. Third Screen of GUI

4.2 **Codes and Standards**

The Health Insurance Portability and Accountability Act (HIPPA) was introduced in 1996 and affects how the system can store sensitive data. Only medical professionals using the data to assist the patient or the individual whose data is being measured can view health data related to that individual [6]. The mobile application to enter production must use password protection for accounts to fulfill legal obligations to keep health data confidential. Information at the database level should be encrypted and signed in the final product to be protected from exposure when viewing the database.

Another standard relevant to this design is ANSI/AAMI ES60601-1:2005, which has guidelines on medical instruments. This means the error rate after sampling must be low enough to not mislead users because of the sensitive medical data shown on these devices [7]. The noise cancellation must first be implemented then filtering algorithms will take more computational power to meet the accuracy standards of this guideline.

4.3 Constraints, Alternatives, and Tradeoffs

The major design alternative was the types of communication between devices. UART was chosen to communicate between the MCU and PC, while SPI was determined to be best for communicating between the ADC and MCU. Data needed to be communicated back and forth on a shared clock between the ADC and MCU, which made it the better choice for SPI communication. Another design alternative was to alter the MCU reference ground to sample the signal instead of creating a circuit to boost it. It was determined that the op-amp would be too non-ideal and would require extra power sources that were unavailable, thus it would be simpler and just as effective to alter the MCU reference ground. The last tradeoff considered was the use of a low-level language such as C/C++ e.t.c, as opposed to using MATLAB to perform the signal analysis and give the resulting graphs. MATLAB was chosen at the cost of real-time signal processing due to its ease of use, and a separate page in the UI was created to display the result of signal processing of the real-time signal. MATLAB was also more familiar to team members. Using a low level language such as C would require difficult driver level programming that was beyond the team members' expertise and given timeline to complete.

5. Schedule, Tasks, and Milestones

The team designed and implemented the real time vital sign processing system over the course of the past semester. Appendix A consists of the all major milestones, the person(s) assigned to those tasks, and their relative difficulty levels. Appendix B consists of the Gantt chart while appendix C contains the detailed breakdown of the tasks and their associated timelines, start date, end date, and duration. The critical path for the project consisted of the microcontroller research, microcontroller purchasing, MCU/PC Communication and the real time plotting on the UI.

6. Final Project Demonstration

Because the external ADC was incomplete at the time of demonstration, many of our specifications could not be met. To demonstrate the functional modules of our project, a subject sits in front of the pre-existing system to record measurements. The following series of events demonstrates the completed aspects of the project:

- The UI gives the operator the option to select how long to record data for as well as s variable sample rate
- The MCU samples data and transfers it to the PC, which displays the incoming data stream by updating every 0.5 seconds
- A button on the UI signals the PC to run the signal processing
- Heartrate is unable to be determined using the low accuracy on board ADC, so its signal is a blank vector as a placeholder
- The raw signal is filtered to reveal the respiration signal
- The respiration and heart beat signals are plotted
- Respiration and heart rate rates are indicated to the user on the bottom of the page

7. Marketing and Cost Analysis

7.1 Marketing Analysis

The target market for this product is hospitals and other healthcare locations that need to monitor patients' vital signs. Current vital sign monitoring systems range in price from about $1350 to $3895 according to Venture Medical [8]. These monitors perform analysis that the noncontact monitoring system will not be able to do, such as blood pressure and ECG for the most expensive models. A noncontact vital signs monitoring system should cost less than these comparisons because it cannot have many of these features. The main advantage that this product will have over its competitors is that it is noncontact, so patients do not need to be strapped in or have any invasive actions taken to receive aid.

7.2 Cost Analysis

The total development cost for a prototype of Non-Contact Vital Signs Monitoring System is approximately $175.34 [9]. The costs incurred in our prototype model is that of the EFM8 MCU, AD7770 ADC, SMA jack to SMB plugs, and 50 ohm SMB terminator plugs, as the existing Doppler radar which extracts the vital signs from the patient, as well as an SMA connector cable, will be provided free of cost by the advisor. The Non-Contact Vital Signs Monitoring System consists of the MCU, an external ADC as well as a computer UI which processes and displays the incoming vital signs in real-time, as well as a mobile app. The computer UI, processing of the signals and the mobile app require man hours to achieve, however they do not contribute to the base cost of the product or the prototype.

Table 2. Developmental Hours Per Engineer	
Task	Hours
Class	20
Weekly Meetings	13
Reports	12
Research	20
Testing	16
Presentation	1
Total	**82**

The developmental costs shown in Table 3. were determined with an engineer's annual salary of $70,000. 82 hours of labor for 5 engineers as determined from table 2. totals $13798.07. Assuming 30% fringe benefits of labor and 120% overhead on materials/labor/fringe benefits, the total developmental costs of Non-Contact Vital Signs Monitoring Systems is shown below in Table 3.

Table 3. Total Developmental Costs	
Development Component	Cost
Prototype	$329.05
Labor	$13,798.07
Fringe Benefits, % of Labor	$4139.42
Subtotal	$17,983.12
Overhead, % of Material, Labor, & Fringe Benefits	$21,579.74
Total Development Cost	**$57,829.40**

8. Conclusion

At present, the system is able to use the 12-bit onboard ADC on the microcontroller to sample data from one channel and transmit about 630 samples /second. The external ADC cannot sample data in conjunction with the microcontroller and is not included in the setup at the moment. Hence, the current data resolution is only 12 bits. In addition, the onboard ADC cannot sample negative values, so the reference ground is connected to -1.5 volts to allow negative values to be observed. However, this makes all measurements imprecise and adds noise to the received data. The graphical user interface accomplishes the specifications it needs and communication with the microcontroller works. However, the microcontroller sends each digit as an ASCII character, which wastes many of the available communication bits. There is a comma separating each data point, but this overhead is unavoidable currently. In addition, the microcontroller streams data rather than using a start/stop protocol to control data it sends or receives from the PC. Finally, the respiration rate signal can be displayed but heart rate requires more resolution to identify and show in the GUI. The mobile application was put on hold but should be implemented once the system is completed. The setup works in a demonstration environment, but could be improved to achieve the desired project goals for daily lab or commercial use.

There are many lessons to learn from how the project was completed. First, the ADC not working was a major detriment and made all other parts of the project more complicated. A major regret was receiving the parts, i.e. the MCU and ADC, halfway through the semester. These parts were expected towards the beginning of the semester and future implementations of this project should make sure to coordinate with the ECE department to receive these parts at the beginning. Although sample code can be written without the physical devices, most development on these devices is done through repeated testing of in progress code. This process is much faster and is the standard way to expand a program's functionality in any medium.

In addition, a suggestion for future teams is to work on the ADC more sooner into the project (which requires the ADC to arrive at the beginning of the semester). This task was anticipated to take time, but eventually became a limiting factor, severely slowing down the rest of the project. Several other issues, like the reference ground introducing noise, the heart rate not being ground, and the MCU's onboard ADC not measuring negative values, were all introduced due to working around the absence of the

ADC.

The choice of MCU and ADC could have been better. Although the chosen parts minimize cost and accomplish the project goals through their specifications, they are difficult to configure and use in practice. The MCU has subpar supporting libraries to code with and a cumbersome proprietary software IDE, Silicon Labs, that anyone using this microcontroller must utilize. The ADC is not from the same company as the MCU and is difficult to configure and use. A suggestion would be for the next team that works on this project to purchase an MCU and ADC that are compatible with each other, preferably from the same company. In addition, a future team should determine the ease of use of the microcontroller by researching coding specifications before purchase.

For the UI side, the team suggests to base any future implementations on the current one, especially the communication code and threading. These problems take up a long time to figure out and are already implemented in Python. However, using a different GUI framework other than Tkinter would allow more freedom while creating the GUI and make it easier to expand. A future team should take the communication code and rebuild the visual part of the UI using a different framework.

On the MCU side, communication should be changed to transmit bits in binary/integer format using the correct number of bits. Transmitting points as an ASCII string wastes so much of the communication channel capacity and reduces sampling rate. A better MCU choice would make this easier. Future extensions should aim to use a two-way communication protocol where the MCU and PC can send-receive on a turn-based system like TCP in networking.

Overall, the lessons learned were to start prototyping earlier, find a way to receive the parts as soon as the project starts, and spend a little more money to get a compatible, easy to use MCU/ADC pair. If a future team follows the suggestions above and gets the current external ADC or a new MCU/ADC pair working quickly into the project, this team suggests adding the mobile application to the agenda and completing the entire project as specified in the project goals.

9. Leadership Roles

Because the team was so small, each member was the lead for at least one aspect of the project. Alec Adamski served as team leader, as well as ADC Lead. Rohan Iyengar served as Webmaster and Documentation coordinator; he worked on the MCU briefly, extensively on the communication aspect between MCU/PC, and led development of the GUI. Kedar Manishankar oversaw the MCU, as well as responsible for the purchase and return of all hardware utilized. Sai was the leader of the signal processing component of the project and designed a significant portion of the GUI. All documentation was coordinated through the github page (https://github.com/RohanIyengar/Noncontact-Vital-Sensor-System), where all parts of the system are separated by folder and can be downloaded by any team for future use.

10. References

[1] J. J. M. de Wit, A. Meta and P. Hoogeboom, "Modified range-Doppler processing for FM-CW synthetic aperture radar," in *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 1, pp. 83-87, Jan. 2006.

[2] Silicon Labs, "EFM8 Busy Bee Family EFM8BB1 Data Sheet," in www.silabs.com. [Online]. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/EFM8BB1-DataSheet.pdf. Accessed: Nov. 22, 2016.

[3] Silicon Labs, "Busy Bee Family UG236: EFM8BB1-SLSTK2020A User Guide," in www.silabs.com. [Online]. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/EFM8BB1-SLSTK2020A-UserGuide.pdf. Accessed: Nov. 22, 2016.

[4] Analog Devices, "8-Channel, 24-Bit, Simultaneous Sampling ADC74HC4051," February 2017.

[5] Analog Devices, "EVAL-AD7770FMCZ/EVAL-AD7779FMCZ User Guide," September 2016.

[6] "Health Insurance Portability and Accountability Act," *California Department of Health Care Services*, 2016. [Online]. Available: http://www.dhcs.ca.gov/formsandpubs/laws/hipaa/Pages/1.00WhatisHIPAA.aspx. [Accessed: Nov 21, 2016]

[7] "ANSI/AAMI ES60601-1:2005," AAMI Standards and Recommended Practices, 2012.

[8] V. M. R. Terms, "Patient Monitors for Sale," 2012. [Online]. Available: http://www.venturemedical.com/products/patient_monitors_and_accessories/patient_monitors/?gclid=

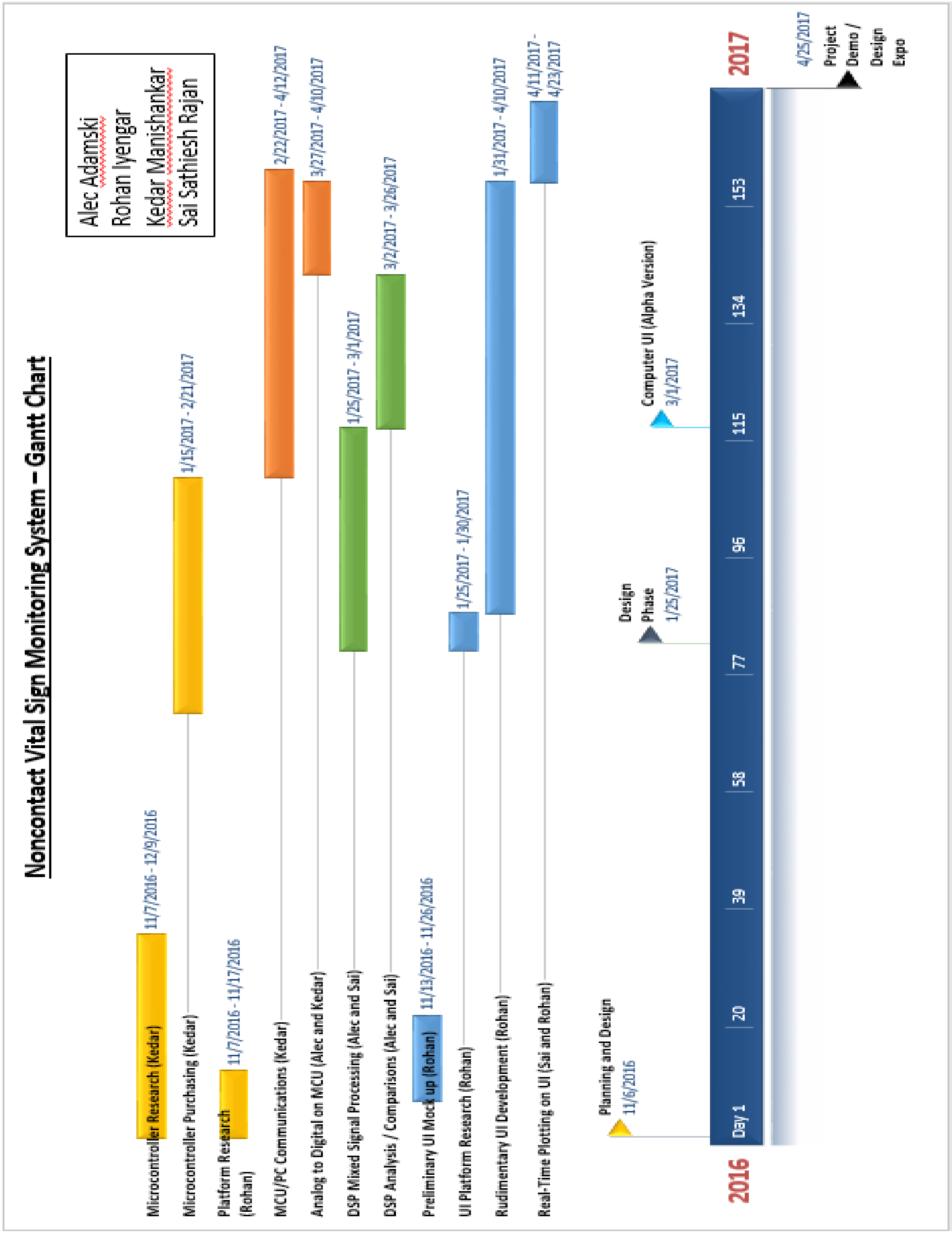
Cj0KEQiA08rBBRDUn4qproqwzYMBEiQAqpzns-VIvcJt9YWD2-uO0FOOhUpHBjR-uGYbCGCaysHW0PsaAlh_8P8HAQ. Accessed: Nov. 21, 2016.

[9] S. Labs, "SLSTK2020A," in www.digikey.com. [Online]. Available: http://www.digikey.com/product-detail/en/SLSTK2020A/336-3160-ND/5115717?WT.mc_id=IQ_7595_G_pla5115717&wt.srch=1&wt.medium=cpc&WT.srch=1&gclid=Cj0KEQiA08rBBRDUn4qproqwzYMBEiQAqpzns67mfAczk-HkE9bZEddtHsK0Ug4W-rrUMBXEhX7r2e8aAg-L8P8HAQ. Accessed: Nov. 22, 2016.

Appendix A – Gantt Table

Task Name	Task Lead	Risk Level
Planning and Design	All	Low
Microcontroller Research	Kedar	Low
Microcontroller Purchase	Kedar	Low
Platform Research	Rohan	Low
Microcontroller	Kedar	Low
MCU/PC Communication	Kedar	Medium
A/D Conversion on MCU	Alec/Kedar	Medium
Signal Processing	Alec/Sai	Medium
Mixed Signal Processing	Alec/Sai	Medium
Analysis/Comparison	Alec/Sai	Medium
UI	Rohan	Low
Preliminary UI Mock up	Rohan	Low
UI Platform Research	Rohan	Low
Rudimentary UI Development	Rohan	Low
Real time Plotting	Sai/Rohan	Medium

Appendix B – Gantt Chart



Noncontact Vital Sign Monitoring System – Gantt Chart
Alec Adamski
Rohan Iyengar
Kedar Manishankar
Sai Sathiesh Rajan
Microcontroller Research (Kedar)
11/7/2016 - 12/9/2016
Microcontroller Purchasing (Kedar)
1/15/2017 - 2/21/2017
Platform Research (Rohan)
11/7/2016 - 11/17/2016
MCU/PC Communications (Kedar)
2/22/2017 - 4/12/2017
Analog to Digital on MCU (Alec and Kedar)
3/27/2017 - 4/10/2017
DSP Mixed Signal Processing (Alec and Sai)
1/25/2017 - 3/1/2017
DSP Analysis / Comparisons (Alec and Sai)
3/2/2017 - 3/26/2017
Preliminary UI Mock up (Rohan)
11/13/2016 - 11/26/2016
UI Platform Research (Rohan)
1/25/2017 - 1/30/2017
Rudimentary UI Development (Rohan)
1/31/2017 - 4/10/2017
Real-Time Plotting on UI (Sai and Rohan)
4/11/2017 - 4/23/2017
Planning and Design
11/6/2016
Design Phase
1/25/2017
Computer UI (Alpha Version)
3/1/2017
Project Demo / Design Expo
4/25/2017
2016
Day 1
20
39
58
77
96
115
134
153
2017

Appendix C – Detailed Breakdown

Title	Start date	End date	Duration	%	Shape
Microcontroller Research (Kedar)	11/07/2016	12/09/2016	33 days	-	
Microcontroller Purchasing (Kedar)	01/15/2017	02/21/2017	38 days	-	
Platform Research (Rohan)	11/07/2016	11/17/2016	11 days	-	
MCU/PC Communications (Kedar)	02/22/2017	04/12/2017	50 days	-	
Analog to Digital on MCU (Alec and Kedar)	03/27/2017	04/10/2017	15 days	-	
DSP Mixed Signal Processing (Alec and Sai)	01/25/2017	03/01/2017	36 days	-	
DSP Analysis / Comparisons (Alec and Sai)	03/02/2017	03/26/2017	25 days	-	
Preliminary UI Mock up (Rohan)	11/13/2016	11/26/2016	14 days	-	
UI Platform Research (Rohan)	01/25/2017	01/30/2017	6 days	-	
Rudimentary UI Development (Rohan)	01/31/2017	04/10/2017	70 days	-	
Real-Time Plotting on UI (Sai and Rohan)	04/11/2017	04/23/2017	13 days	-	