Design and Implementation of a Real-Time Programmable Multi-Channel Data Acquisition System for Sensing Applications (December 2019)

Ethan Groenow, Matthew S. Johnson, Liang-Kuan Lee and Ximena Banuelos Martinez
Professor Heydari

Abstract—**The purpose of the following project is to create a general purpose data acquisition system for use in sensing applications. As a proof of concept, the system is configured to interface with a biological subject through electromyographic surface electrodes. The signals of interest are arm and hand contractions. The overall objective established for the fall was to construct a rudimentary system prototype. The intention was to test amplification of low frequency signals and allow for single and dual channel data acquisition that is transmitted to a user interface via serial connection. The interface serves to display the acquired signals in real-time waveform plots. In order to achieve the quarter goal, a few methods of design and engineering were implemented. SPICE analysis was used to assist in the selection of a low-frequency, small signal amplifier. The rest of the components were selected based on currently available commercial circuits. The control system is centered around an Arduino microcontroller and the user interface was developed in Python. Finally, oscilloscopes and multimeters were used for laboratory testing of the system components. The work of this quarter has resulted in the finalization of part selections, testing of individual chips, and the trial of a single channel system. The code to the control system has also been finalized and the user interface can display two live waveforms on a host computer. For the following quarter, the goal is to elaborate on the current design to allow wireless transmission of live data through four channel acquisition.**

Index Terms— **Biomedical Engineering, Biomedical Signal Processing, Control Design, Data Acquisition, Signal Sampling**

I. Introduction

Electromyography, or EMG, is the study and implementation of electrical signals produced by action potentials during muscle activation [1]. Throughout the body, there are various biological processes that can be examined to determine electrical biomarkers that signify bodily functions. Other important signals can spawn from the dipole movement of the pupils or the activity of an individual's brain. The Bio-DAQ project aims to capture such signals. The emphasis is placed on EMG signal acquisition due to the vast documentation available on the subject. Methods of acquiring EMG signals can be quite beneficial as they can be developed for assistance of patients with prosthetic solutions. The field of biomedical signal acquisition is continuously expanding. Besides the immediate medical applications, processing of biological signals have been expanded for use in other environments, such as in recreational projects that include gaming [2]. With the inherent potential of biological data acquisition systems, it is desired to continue advancements in the field.

The goal of this project is to create a real-time programmable multi-channel data acquisition system. With the proper chip selection and connections, the system will amplify, sample, and display biological signals in the 1μV to 1 mV range. The capability of the system to meet these demands will be demonstrated by identifying muscle contractions via EMG electrodes. Further development in the system pushes for a final goal of recording analog myoelectric signals and using industry tested recognition techniques during data processing to determine specific movements [3].

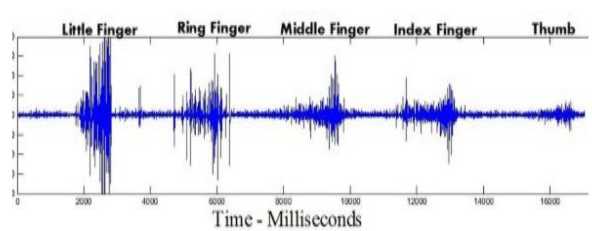

Fig. 1. Example of biomedical data collected from EMG signal acquisition. Waveforms recognize finger contractions.[4]

As previous research informs, the movements could be analyzed to make decisions on which finger contracted, as displayed in Figure 1. Developments continue to prove fruitful as measurements become more accurate due to the implementation of more complex machine learning models [5].

Due to the format of the course, the overarching goal for the system was broken down into smaller, reasonable objectives. Creation of a testable prototype was stressed for the fall quarter product. In order to produce the prototype, the first step was to be able to determine the requirements demanded by biological signals. After the initial research, the group set both software and hardware goals. The amplifiers on the product needed to be able to amplify laboratory generated signals from the millivolt range to the volt range. The ability to accept signals under 500 Hz was determined to be critical. The control system would then have to be able to accept the signals from one to two channels, digitize and then transfer the data to a host computer through a serial cable. The fall quarter GUI would then have the function of displaying the data in real-time to the user in the form of a waveform plot that keeps track of the samples.

Using the engineering knowledge gained from previous courses, the team used common methods for achieving the quarter goals. In regards to amplifier design, SPICE models were used for testing of different configurations. The models were ran through transient and frequency analysis to characterize the amplifiers. From that information, it was able to be determined if the amplifier could serve as an active filter to cut-off signals above 500 Hz and amplify the others signals in the magnitude range of 60-80 dB. An Arduino microcontroller was used to control the system and transfer data to a host for graphical display. The microcontroller was programmed to communicate to other chips via I2C and SPI protocols. The user interface was created with access to documented Python libraries. When testing was finished on all of the individual circuits and chips, a one channel prototype was created that could take in a 300 Hz millivolt signal and display the amplified signal on the GUI. Work is still needed to be done on the project. A data pipeline was found during the testing and will need to be addressed for the next quarter. The objective for the winter quarter will be to expand on the current progress, allowing for four channel multiplexing and wireless communication.

II. MATERIALS

The initial stage of development for the project started with research necessary to determine the desired characteristics of the components to use in the system. A data acquisition system serves the purpose of taking in analog signals, converting these signals to a digital format, and displaying and storing the data for the user. From the basic concept of a data acquisition system, the plan for the project was elaborated to meet the needs of the biomedical signals of interest. The following requirements were set as desired properties that the Bio-DAQ system was intended to have. The parts were then chosen to meet the expectations of the project as best as possible, while keeping the budget constraints in mind.

A. Requirements

The requirements used for part selection were based on the expected qualities of the Bio-DAQ. The system was intended to be a real-time programmable multi-channel system. During the initialization of the unit on power up, the user should be able to vary the gain and cut-off frequency for the amplifiers. This would allow for more diverse usage as the user can set the system depending on the application demands. Once the amplifiers are set up, the Bio-DAQ should handle input from multiple channels. The last qualitative specification is to allow for real-time data waveform display on a graphical user interface. In regard to the biomedical applications, there were some more specific requirements set for part selection. Most biological signals of interest fall below approximately 500 Hz and range between 1 μV to 10 mV [4]. Based off the research, the amplifiers would need to allow amplification of low frequency signals and have a gain between 60 dB and 80 dB. The data acquisition system should also have the ability of multiplexing between a minimum of four channels that correspond to readings from different points on the body. These specifications were used when choosing components and testing equipment.

B. Part Selection

During the purchasing process, there were six main components that were ordered that were essential to the system design. In order to speed up the development cycle manufacturer available chips were used. The most crucial chip for the system was the analog-to-digital converter, ADC. The selected ADC has 16-bit resolution and can accept 5 V supply. The supplies for the remaining chips also allow for 5 V supply which is the current demand of the prototype. The 16-bit resolution was necessary as it provides a means of measuring voltages down to 76 µV. The ADC has a max sample rate of 1 Msps and communicates over SPI. SPI was chosen to allow for full-duplex data transfer. With SPI, the ADC can transfer the samples to the microcontroller at a clock rate up to 12 MHz, significantly faster than the I2C 400 kbps limitation. On the I2C bus of the system, a 16 port I/O expander and 128 position digital potentiometer were purchased. The digital potentiometer is intended to be used with the amplifiers to select gain with a variable resistance from 125 Ω to 10 kΩ. The I/O expander serves the use of controlling additional multiplexers for amplifier control and reset pins to other chips.

The microcontroller that is implemented in the prototype is an Arduino Uno. The Uno was chosen for its familiarity and reliability. To prevent the draw of excess current from the Arduino, 50 mA linear voltage regulators were ordered to provide 3.3 V and 5 V supplies to the other circuits. The last major component is the 4-to-1 high-speed multiplexer. The multiplexer can switch on a 1 MHz clock and is used to allow the ADC to obtain samples from four different surface locations.

C. Tools & Supplementary Materials

Aside from the main components, some materials were purchased to assist in the testing and implementation of the circuits. A 16-bit digital-to-analog converter was selected for prototype testing. Online databases with previous research data can be used to recreate signals that will then be fed into the Bio-DAQ. A bidirectional level-shifting voltage translator and Arduino Nano IoT are to be used for future improvements. With the Nano as the microcontroller, the system can run at a higher clock speed and at a lower voltage level. The translator would enable the Nano to communicate to other chips that require 5 V for the digital interface. Some of the chips were configured in TSSOP-16, SOIC-8 and SC-70 packaging. PCB breakout boards were bought to allow the circuits to be used for prototyping on a breadboard. EMG electrodes were purchased for the final testing with the finished product. Most of the current testing implemented the Analog Discovery 2 and a digital multimeter for function generation, oscilloscope captures and voltage measurements.

III. High-level System Architecture

The overall architecture for the Bio-DAQ data acquisition system is based around a central ADC. Figure 2 represents the general control flow of the system.

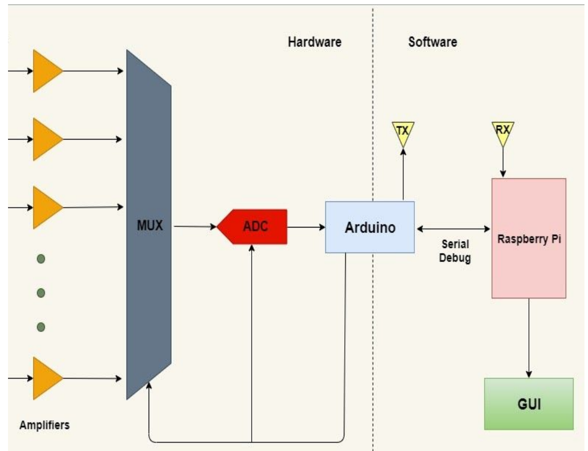

Fig. 2. The initial overall Bio-DAQ system architecture. The input to the system starts with external analog voltage from EMG electrodes. The final data destination is the GUI.

The biological signals are first introduced by contact made with EMG electrodes. Considering the small-signal characteristics of the analog signals, they are introduced to an amplification circuit used for filtering and amplifying. With the desire to incorporate multiple signals simultaneously, the operational amplifiers output to a multiplexer controlled by the Arduino. The multiplexer cycles the channels that are then being read by the ADC. Once the data has been sampled and converted, it is transferred to the Arduino. There are two methods of transferring the acquired information to the host. The current implementation is the serial cable directly to the host computer. With further testing, a wireless transmission system is to be incorporated. Finally, the data is displayed to the user with a GUI. The system architecture is essentially divided into three principal areas: analog amplification, hardware and software interfacing, and graphical display.

A. Analog Signal Overview

The purpose of the amplifier design is to handle the analog signals that are being received at the input to the system. The signals coming from the electrodes will be as small as 1μV and in order for them to be workable without losing so much of the data itself then the system needs to amplify the signals up to at least 1V. Another important aspect to consider with the overall signal path through the Bio-DAQ system is interference and noise. A common noise that occurs with biomedical signal data acquisition systems is a 60 Hz signal interference that comes from the capacitive coupling of a human subject and power/transmission lines that are around the building wherever testing is happening. The system needs to account for this noise and be able to get rid of it without compromising the important EMG data. The Bio-DAQ system also uses a great deal of multiplexers (MUXs) and these also can provide for a possibility of interference. The amplification and filtering design must take these problems into account in order to the correct data.

B. Hardware and Software Interface

The objective of the control system for the project architecture was to administer a means of converting and relaying the analog information from the hardware for use by the GUI. The controls are set by the Arduino Uno. From the Arduino, the setup of the system is managed. When initializing the Bio-DAQ, the user can select a range of gains for use through the microcontroller. That information is then sent to the amplifier circuit for proper adjustment for the application at hand. The Arduino also sets the cycle frequency and order of the multiplexer which is essential for reading from the four different channels in the most recent iteration of the project design. Another ability of the control system is to note and supervise necessary resets of the different chips. With the desired system requirements initiated, the microcontroller continuously reads the data that is converted from the ADC. From that point, the Arduino is able to send the data to a host computer for further processing through a USB cable. The final goal of the control system will be to introduce a wireless transmission functionality.

C. User Interface Objective

The goal of the graphical user interface is to provide users with an easy way to visualize captured signals. It will also be used for testing and calibration of the system. The captured data will be coming from the Arduino via serial. The software will then parse and format the incoming bytes to display the signal. The interface will also allow users to easily adjust the gain of the amplifier, making the system real-time programmable, as well as saving incoming data for future analysis.

IV. Methods

The methods used to approach the problem at hand are derived from previous coursework and online resources that explained common engineering practices. The project demands the development of a product through a full life cycle. The process is initiated with the project definition and is meant to be carried through to the testing and qualification stages. For the fall quarter, the emphasis was on design and testing of a prototype of the system. In order to accomplish these beginning processes it was evident that the assignment needed to be broken down into individualized focuses and specializations. The tasks were divided into four subcategories to represent the focus of the team members. One engineering objective was to handle the design of the amplification and filtering circuits for analog biological signals. The creation of a structured control system was another major component. The graphical user interface was separated to focus efforts on generating a real-time display of the data. The final assignment was to find a means of testing the system holistically. Through sufficient research, the following design techniques were implemented.

A. Amplifier and Filter Design

When designing the amplifier schematic and filter specifications for the Bio DAQ project, a number of different aspects had to be considered. Broadly overlooking the applications of this data acquisition system and through discussions with the team and advisors, the team was able to pick out certain details that would allow us to pick hardware and design the system. The following specifications were decided upon to use as guidelines when choosing the amplifier: a gain ranging anywhere from 40 to 80 dB, a low input noise voltage (< 10 nV/√Hz), a quiescent current under 30μA, and low power consumption. The amplifier chosen for our system is Linear Technology's LTC1050 zero- drift op amp.

After picking the amplifier, building the amplifier schematic was the next step. This design applies not only to the amplification stages of the system but also to the filtering. The Bio-DAQ system uses an active bandpass design in order to amplify and filter. Figure 3 shows the amplifier and filtering schematic. Where a pair of capacitors and resistors handles the gain, another pair handles the lowpass cutoff and the third pair handles the high pass cutoff for the filter. Test were ran to specifically test each of these aspects.

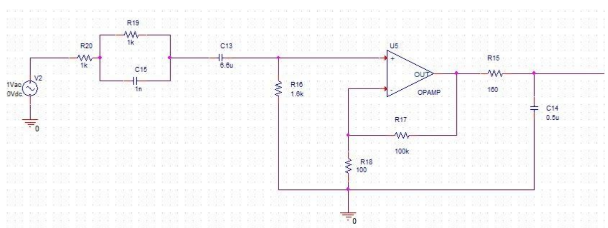

Fig. 3. The schematic accounts for matching the input impedance of the source and then also has its gain and bandwidth variability.

Another focus of the amplifier design was to be able to build is so the gain is easily variable as to accommodate for different types of biomedical signals. To solve this, the resistor responsible for changing the gain of the system would then be replaced with a multiplexer allowing for easy gain varying.

Moving forward, in order to get the best signals across the system and insure that the team is getting the desired specifications of the design. Then a power and noise analysis of the amplification and filtering systems should be completed.

B. Data Acquisition System Control

In order to construct the control mechanism for the data acquisition system, preliminary research was performed to decide on the specific components to use in the architecture. Considering that the ADC is the most crucial part of this system, the examination of various ADC architectures was carried out. It was determined that the best converter for biological data acquisition is a successive-approximation ADC [6]. The rest of the system was then designed around controlling the inputs to the ADC. Figure 4 provides a more detailed view of the system architecture design. In terms of the control system, the most important components to control were the digital potentiometer, four channel mux, GPIO expander, ADC, and Arduino Uno. There is also a transceiver module shown to represent the future development of the project to allow for wireless communication.

After reviewing the datasheets for the components and soldering any surface mount chips onto breakout boards, there were three significant tasks to complete for controls. Schematics were drafted to provide a layout of the connections to all of the chips. The schematics included all connections made from the EMG input terminals down to the Arduino Uno. The second assignment was to program the microcontroller to be able to read and configure the I2C and SPI devices. The last task was to test the chips to ensure compatibility with the Arduino. The ADC chosen was the Texas Instruments ADS8329. The converter provides 16-bit resolution with a maximum sampling rate of 1 Msps that can be read over SPI. SPI was selected for the device so that the ADC can have its own dedicated connections to the Arduino, full-duplex could be implemented, and higher transfer rates could occur. For the prototype, the clock rate was set to 1 MHz. On startup, the device requires a two byte configuration to the control register. Communication is in MSB first and SPI mode 1. Four bits are sent for the operation code and twelve bits are used for data. The two configurations tested with the fall prototype were auto-trigger with an internal clock and manual trigger with an external clock. The commands are 0xEDFD and 0xEBFD, respectively. With auto-trigger, sampling starts as soon as conversion is finished and the conversion pin notes high. The Arduino is programmed to begin reading the previous sample once the pin goes low, during a new conversion. Timing is important to keep in mind. The sampling takes three internal clock cycles, CCLK's, and conversion takes 18 CCLK's. Using the internal clock, the clock frequency is approximately 21 MHz. When manual trigger and external clocking was used during testing, the timing was altered. When an external clock is used for the ADC reference, the internal clock frequency become half of the external clock frequency. Manual trigger also allows the start of the next conversion to be controlled by the Arduino pulling the ADC CONVST pin (pin 9) low. Reading was chosen to be performed during sample conversion to allow for the 16 external clock cycles, SCLK's, to read the 16 bits of the previous conversion. Once the data reaches the microcontroller it is then transferred serially over a USB cable to the host computer. With the ADC having been successfully configured and read from, the voltage

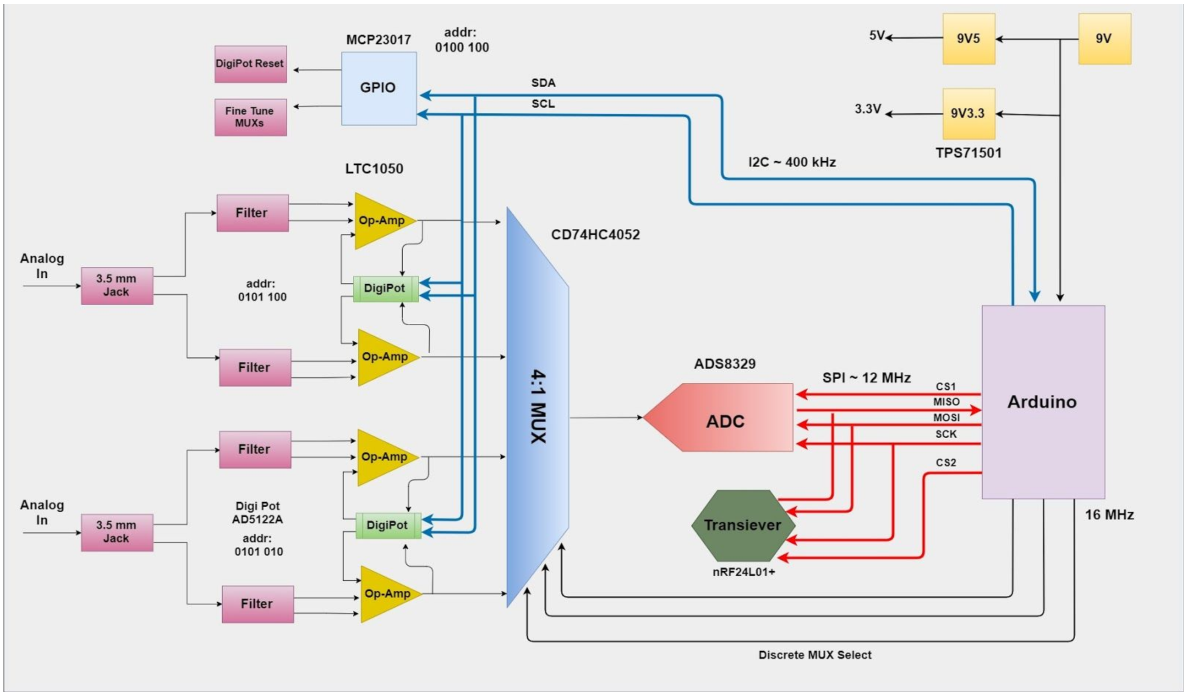

Fig. 4 Expanded view of Bio-DAQ architecture. I2C communication is shown in blue. SPI is noted in red. I2C slave addresses are also labelled along with protocol speeds.

value is converted from the transferred data, D, by the following formula:

$$V_{sampled} = V_{ref}\left(\frac{D}{65535}\right) \qquad (1)$$

The next chip to configure was the digital potentiometer. I2C is used for all peripheral chips, at 400 kbps, as communication will only normally occur during setup and transmission speed is not as demanding. The specific digital potentiometer used is the Analog Devices AD5122A which is a dual channel 128 position potentiometer with a resistance value up to 10 kΩ. The purpose of the potentiometer is to be able to vary the gain characteristic of the operational amplifiers. For the prototype, only one chip was used as the dual channel allows two op-amps to be controlled at the same time. The seven bit address of the device is 0x2A, which is equivalent to 42. This address is realized by leaving ADDR0 and ADDR1 pins floating. The only configuration needed for the digital potentiometer is to set the resistance for the individual channels. Two bytes are written to the chip during this process. RDAC1 and RDAC2 denote the two channels and are accessed with the transmission of 0x10 and 0x11, respectively. The desired resistance value is then sent over as a byte. It is important to note that the AD5122A ignores the least significant bit while writing the resistance value. Therefore, for the maximum value of 10 kΩ, 0xFE is used. The wiper pins and B terminals are connected for use as a variable resistor. To determine the transmitted data the following is used:

$$R_{wb} = 10,000\left(\frac{D}{128}\right) + 125 \qquad (2)$$

Once the decimal value is found, it must be converted to seven bits and an eighth bit is appended as the least significant bit. The current iteration of the control program allows the user to select between 1 kΩ to 10 kΩ with increments of 1 kΩ.

The last chip to be directly configured by the Arduino program is the GPIO expander. The Microchip MCP23017 was chosen to provide 14 additional discrete control pins. It is also implemented to assist in the prevention of drawing excessive current from the Arduino. The Arduino has a maximum current supply of 50 mA. Although the design aims to keep power consumption low, isolation from some of the discrete controls is desired to allow the microcontroller to focus on control of the ADC. The GPIO expander has the A2 pin tied high and A1 and A0 low to produce an I2C address of 0x24, equivalent to 36. During configuration, the IOCON register is first set with two bytes, 0x0A and then 0x20. The operation turns off sequential register access and assigns the desired banks to the registers. The IODIRA (0x00) and IODIRB (0x0A) registers are then accessed to set the GPIO pins as outputs with the command 0x00. The GPIOA (0x12) and

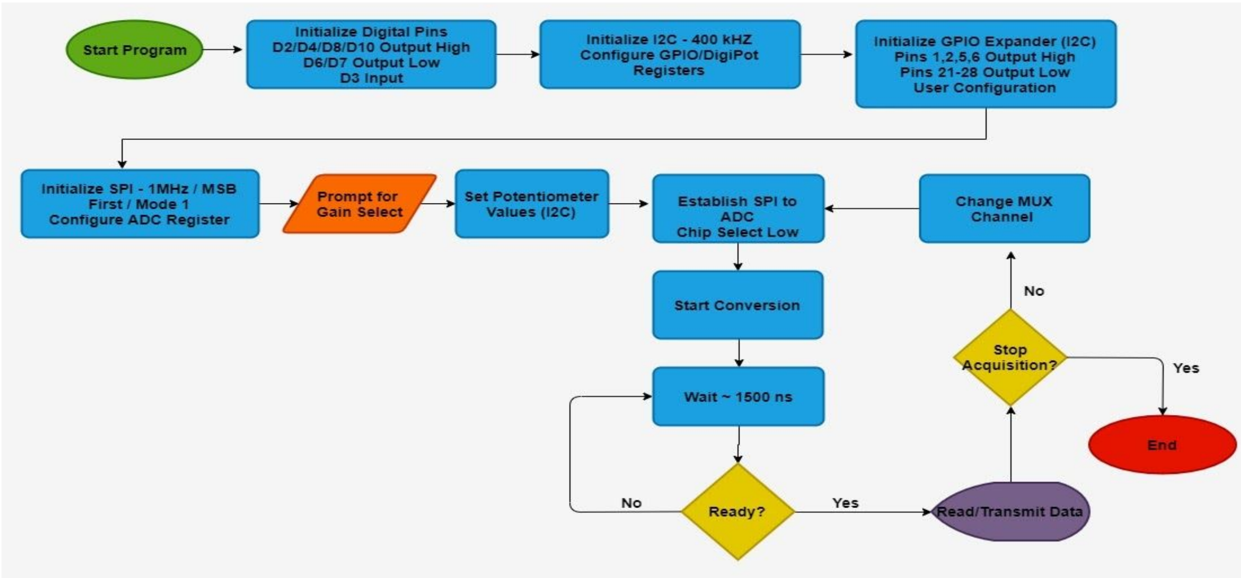

Fig. 5 Bio-DAQ control system block diagram. The diagram details the initialization process and the necessary pin configurations needed on the Arduino Uno. The loop segment of the figure emphasizes the core function of reading from the ADC.

GPIOB (0x13) registers are finally accessed to set the individual pins high or low. Once the registers are accessed, one byte is sent to choose which pins are high. Each of the eight bits represent a pin with a 1 equivalent to the pin being high. Currently, the GPIO expander is used to control the digital potentiometer reset pin. With further development, the chip will be used to control multiple digital potentiometers and multiplexers.

The last component directly controlled by the Arduino Uno is the main multiplexer that feeds into the ADC. The Texas Instruments CD74HC4052E high-speed 4-to-1 multiplexer was selected. The multiplexer is controlled directly by the Arduino due to its critical function in the architecture. To switch between the four channels the S0 and S1 pins are alternated while the enable pin is kept high. During the prototype testing only the S0 pin was toggled to switch between channel 0 and channel 1. The next quarter will work to read by cycling all four channels.

With all of the components configured, testing was performed on each of the individual chips. A function generator, oscilloscope and power supply were used to test the ADC and ensure accurate readings. Readings were found to be correct for millivolt level inputs and for sinusoidal inputs. The other components were tested by connecting loads and LEDs to visualize the logic state of different pins. A logic analyzer was also used during testing to prove that the correct frames were sent over the SPI and I2C buses.

Final control of the system is programmed according to Figure 5. After initial setup of the system, the microcontroller loops through the channels and conversions from the ADC. More testing and fine tuning of the control system will take place during the winter quarter.

C. *Graphical User Interface*

A GUI is an essential part of any program since it allows the user to easily interact without placing a large cognitive load on them. In our case, it helps the user visualize the incoming signal instead of just printing out the incoming data. By displaying the data in a human readable manner, it can also be used for calibrating and testing the system. The GUI software can be broken into three parts:establishing a connection to the Arduino, reading the serial data, and plotting the data in a readable format.

To establish a connection between the Arduino and the computer, a library called pySerial had to be used. This library allows the computer to read serial data written from the Arduino via UART communication. To ensure that the data is readable, the Arduino and serial object must have the same baud rate and using the same port. After ensuring that the connection is setup properly, the next step is to start reading the data.

PySerial has a built in function that allows the reading of incoming bytes. After storing the bytes into a buffer, the data must be parsed and formatted properly. This is done by splitting the data via the newline character to mark the end of the data points. Once split, the data must be decoded from binary to a more readable format such as ascii. Once the ascii values are found, the string can then be converted to a floating point value to be stored in an array that will then be plotted.

The final part of the software is plotting the incoming data. This is done through the use of the matplotlib library. Two subplots are created. The top subplot displays the entire signal while the bottom subplot displays the last fifty samples to provide a more real-time view of the signal. The data is plotted continuously through the use of an animation function. This function will continuously call itself after a certain interval to update the data.

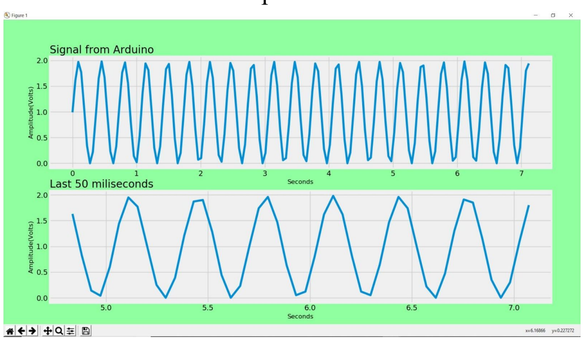

Fig. 6 Example of the GUI plotting a sine wave offset by 1V from a function generator.

D. *Testing Infrastructure*

In order to test our system, we used an Analog Discovery 2. This console not only has a built in Oscilloscope and Arbitrary Waveform Generator (which can both run simultaneously), but can also be run and controlled with a user-friendly software interface called "WaveForms". We used the function generator of the Discovery 2 to play different waveforms through our circuit components, while running the oscilloscope function to ensure our circuit was working as intended at any segment. We were able to use the Analog Discovery 2 to play not only basic sine waves, saw waves, and square waves, but real pre-recorded electromyography (EMG) waveforms as well.

Results and Performance

In our testing procedures we were able to successfully display a pre-recorded electromyography signal by playing it through "Waveforms". This is a major step forward for the team because a crucial part of the system is integrating the hardware and software. With a solid foundation, we can start adding components such as potentiometers and start moving the communication from serial to wifi.

The prototype currently functions as it is expected to with a minor setback on the graphical interface. There is a slight problem in the parsing that causes some samples to be cut off. This causes the signal to look slightly distorted due to aliasing. The software also has not been updated to handle multiple signals.

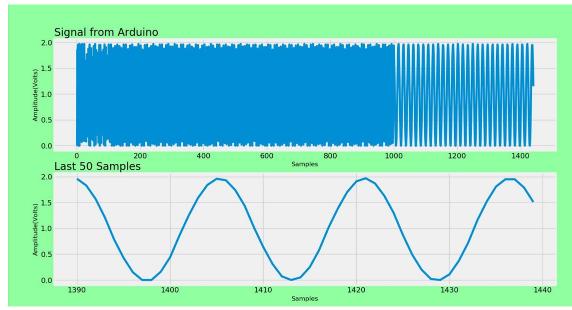

Fig 7. Testing the GUI with a 1kHz sine wave offset by 1 Volt to ensure proper functioning.

V. Conclusions

Bio-DAQ has made great progress throughout the fall quarter. The team has met many milestones such as finishing the system architecture design, completing the component testing, and starting the hardware and software integration process. Currently, our system is able to amplify a sine wave with an amplitude of around 50mV and display it on the GUI. The prototype works as expected with a minor exception in displaying the signal. The parsing of the data erroneously truncates samples which causes slight distortion of the plotted signal. Overall, our team was able to achieve a good foundation to continue building on top of.

For next quarter, the team plans on making the system multichannel and real time programmable. As of now, the GUI can only display one input signal from the Arduino. The end goal is to be able to display at least four different signals using electrodes to capture the EMG signals. The amplifiers are currently configured to provide a certain gain using simple resistors, but our end goal is to replace them with digital potentiometers in order to adjust the gain depending on the signal being measured. The team still has much to work on and is hoping to produce a PCB for the winter design review.

Appendix 1

A few technical standards were relevant to the project during the design process. The SPI, I2C and USB protocols were used during the implementation of the fall quarter prototype. The USB protocol was needed as it is built into and required by the Arduino Uno for wired communication to a host computer. USB was used to serially transfer the converted ADC data to the computer for further processing and display. The main standards that were directly chosen were SPI and I2C. When selecting the different components for the system, different chips were found to be able to communicate to the microcontroller via different communication protocol options. In this design, the ADC uses SPI and the GPIO expander and digital potentiometer uses I2C. These protocols are needed as the chips are communicating within the system. USB is used for system communication to external systems such as a laptop. I2C was selected based on its versatility and ease of attaching multiple slaves while keeping the wires on the bus limited to two. Since the chips on the I2C bus are configured on the setup of the data acquisition system, the transfer speed doesn't need to be excessively fast. The selected I2C transfer rate is compliant with fast mode at 400 kbps. On initialization the master can cycle between the GPIO and digital potentiometer slaves to configure and read the register values. SPI was selected specifically for the ADC. The drawback of SPI is that four additional lines are required for communication. The benefits compensate this in the design. SPI allows the ADC to have its own reserved bus. The four wires also allow for full-duplex communication between the Arduino and the ADC, which wouldn't be possible with I2C. The last major reason for the SPI standard is that it allows the system to take advantage of the high ADC sampling rate as the protocol isn't as limited as I2C. With the SPI connection, the clock can run at 1 MHz for the project prototype when reading ADC conversions. The Bio-DAQ design is compliant to the standards as the proper number of wires were isolated for the buses. The I2C has the two-wire bus with 4.7 kΩ pull-up resistors. The slaves then adhere to the seven bit addressing format. The SPI standard is compliant with the four wire bus and runs at 1 MHz with the most significant bit first and adherent to mode 1. The data transmission was tested for the devices with a logic analyzer and proved to receive the proper data frames and function appropriately.

APPENDIX 2

One main constraint which we are sure has affected many other students in this class was the logistic constraint of late arrival times of critical components. This uncontrollable aspect led to restrictions on what phases of our project could progress, and which phases of our originally planned timeline would have to be postponed. Our team, with the guidance of our advisors, was able to stay flexible and constantly make progress. When our immediate next step required a component which hadn't arrived yet, we made preparations (research, writing code) for steps further down our timeline. Using hard-coded inputs to test our GUI and playing pre-recorded electromyography (EMG) signals before we acquired EMG electrodes are just some of the ways our team has worked with our given conditions at any point in time.

With limited funding, another problem we had to face was finding a method of purchasing the components. The original iteration of the data acquisition system required more components. Due to this specific constraint, ordering of parts was limited to what was critical to deliver a functioning prototype. The task required going through the part sheet multiple times and determining what was truly needed. The most obvious example of this was that the Bio-DAQ was originally intended to have eight input channels. The amplifiers that were selected to meet the design requirements ended up being relatively expensive, so the number of channels was decreased to four. In the end, some funding was obtained from the course for ordering chips and what wasn't covered from the class was purchased out-of-pocket.

Another small constraint involved the manufacturability of the prototype. After researching the current chips that are in the market, many of the design requirements of the project were only able to be met by chips that were fabricated in surface mount packages. A simple, yet difficult, approach to overcoming this was to use breakout boards. The surface mount chips were soldered onto the breakout boards so that they could be used on breadboards for the prototype.

Lastly, a constraint we all have to learn to accept is that we are all human, and while perfect productivity and perfectly met deadlines are ideal, team members have their own lives and outside struggles which can affect a team's schedule and progress. Every member of the Bio-DAQ team had other obligations, such as other classes and work. To overcome the scheduling constraint the team created an online drive that allowed members to contribute when was most convenient for them. The drive was kept up-to-date throughout the quarter to let the members know where progress was at on the project. Besides the weekly lab check, the team was able to agree on a specific meeting hour once a week to update on progress and meet with our advisor. In order to perform testing during times when everyone could meet, the team used a personal computer-based function generator and oscilloscope so the team could test outside of lab hours. With proper collaboration, the team was able to overcome the constraints that were placed on the project throughout the quarter.

Appendix 3

As designers of a product which may very likely end up in medical environments handling sensitive information, it is our responsibility to build security measures to protect personal data. In our initial planning steps we took this into account with our choice of localized wireless communication. As an additional measure of protection, we implemented encryption as well, so that should data fall into the wrong hands, encryption could serve as a deterrent or obstacle to unethical users. If our product were to grow in scale and usage, we would have to implement additional safety measures, and tread the waters of cloud storage very carefully. It is important to understand the vulnerabilities at each link of our system's communication chain, which is why our networking team has done research on the vulnerabilities involved with any of the software or hardware we use (Raspberry Pi is actually immune to both Spectre and Meltdown!). In our second quarter we will be able to further test software and hardware vulnerabilities, and implement measures to prevent exploitation.

Acknowledgment

Designing the data acquisition system was not as simple as it seemed. The team spent many weeks at the drawing board revising the base design again and again. Without our advisors, the team would still be in the design phase of the project. The Bio-DAQ team would like to say thanks to Professor Heydari and Omid Arasteh for providing guidance and pushing us in the right direction. Without them, we would not have been able to have our vision become a reality. The team would also like to acknowledge the EECS department for providing funding to assist in the build of the project and providing an open lab to conduct testing. Finally, thanks to the Bio-DAQ team, Ethan Groenow, Matthew S. Johnson, Liang-Kuan Lee and Ximena Banuelos Martinez for putting in time to work on the project despite having such busy schedules.

References

[1] M. Jamal, "Signal Acquisition Using Surface EMG and Circuit Design Considerations for Robotic Prosthesis," in *Computational Intelligence in Electromyography Analysis – A Perspective on Current Applications and Future Challenges*, G. R. Naik, 2012. [E-book] Available: IntechOpen e-book.

[2] L. van Dijk, C. K van der Sluis, H. W. van Dijk, and R. M. Bongers, "Learning an EMG Controlled Game: Task-Specific Adaptations and Transfer," *PloS one*, vol. 11, no. 8, August 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4996424/. [Accessed Oct. 22, 2019].

[3] D. A. Reyes López, H. L. Correa, M. A. López, and J. E. Duarte Sánchez, "Expert committee classifier for hand motions recognition from EMG signals," *Revista chilena de ingeniería*, vol. 26, no. 1, pp. 62-71, 2018.

[4] P. Kumari, and A. Raj, "ORIGIN of Biomedical Signals," *Ashpreetpro*, August 12, 2018. [Online]. Available: https://www.ashpreetpro.com/biomedical-signals/. [Accessed Oct. 10, 2019].

[5] E. Donati, M. Payvand, N. Risi, R. Krause and G. Indiveri, "Discrimination of EMG Signals Using a Nneuromorphic Implementation of a Spiking Neural Network," in *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 5, pp. 795-803, Oct. 2019.

[6] W. Kester, “Which ADC Architecture is Right for Your Application?” *Analog Dialogue*, vol. 39, no. 1, June 2005. [Online]. Available: https://www.analog.com/en/analog-dialogue/articles/the-right-adc-architecture.html. [Accessed Oct. 12, 2019].