Microcontroller-based General Platform for a Reconfigurable Wireless Brain-Computer Interface

NSF Summer Undergraduate Fellowship in Sensor Technologies
Basheer Subei (BioEngineering), University of Illinois at Chicago, Cook County
Advisor: Professor Van der Spiegel, Department of Electrical and Systems Engineering

***Abstract*—Recent advances in embedded wireless technology open the door to fully portable closed loop brain-computer interfaces (BCI) that give neuroscientists the ability to run BCI experiments on primates in an unconfined natural setting. Currently developed portable BCI platforms, however, do not incorporate customizability in their designs, requiring the researcher to modify the device's firmware whenever they need to tweak the device's settings. This means that the researcher has to have designed the BCI him/herself or requires the researcher to have the BCI designer tweak the settings. Needless to say, this hinders the prospects of BCI platforms being widely deployed and used by neuroscientists for experiments. This paper presents a microcontroller-based BCI design that aims to provide a general wireless BCI platform that incorporates customizability and the ability to tweak settings over-the-air from a simple PC application interface. The limited on-air data rate (2 Mbps) of the wireless transceiver currently limits the data transfer to four 12-bit resolution ADC channels for recording at 20ksps and four DAC channels for stimulation. In-house development of an Ultra-wideband (UWB) wireless transceiver with a much higher on-air data rate has already started, which would allow for a greater number of recorder channels at a much higher sampling rate.**

***Index Terms*—Analog-to-digital converter (ADC), Brain-Computer Interface (BCI), Closed Loop System, Digital-to-analog converter (DAC), Microcontroller, Radio Frequency (RF) transceiver, Ultra-wideband (UWB) transceiver**

I. Introduction

Most brain-computer interface (BCI) experiments performed in laboratories are based on rack-mounted setups that lack portability, and therefore restrict the environment and experiment duration to several hours a day. While portable closed-loop BCI systems exist and are currently being developed [1,2], these BCI systems are designed for specific experiments in mind (they only work with specific neural signals) and lack the ability for end-user (experimenter) customization, requiring unneeded intervention of the BCI designer (computer-engineer) to customize the system for each different experiment. Therefore, there is a demonstrated need for creating a BCI system that is portable and serves as an easily customizable general-purpose platform for BCI experiments. An example of a potential experiment is to use a BCI for facial reanimation to treat patients of facial paralysis.

The BCI design discussed in this paper describes a portable general-purpose platform for a wide array of closed-loop BCI experiments. The platform doesn't rely on specific recording or stimulating methods, since the pre-amplifier and current-source circuit boards can be swapped depending on the recording and stimulating methods. As an example, the platform can be used to record ECoG signals and stimulate muscles, or it can be used to record EEG signals and stimulate the motor cortex. The changes in parameters required for recording and stimulating are set by using a MatLab GUI to communicate with the BCI. This makes the experimenting process easier for the user (researcher) and removes the need for any intervention by the BCI's designer to tweak the settings.

The closed-loop BCI system discussed in this paper consists of an analyzer and a stimulator. Both devices will have an XMEGA as the MCU performing all the calculations and processing. The analyzer communicates wirelessly with the stimulator via an RF module and triggers the stimulus response signals. Furthermore, another MCU board interfaces the PC and intercepts commands given by the user and relays them to the respective modules (analyzer or stimulator). This allows the user to set the device in different modes, including a manual mode to control the stimulator directly, and a passive mode to only receive and save brain signals from the analyzer. The block diagram illustrating the overall process can be seen in Figure 1 on the next page.

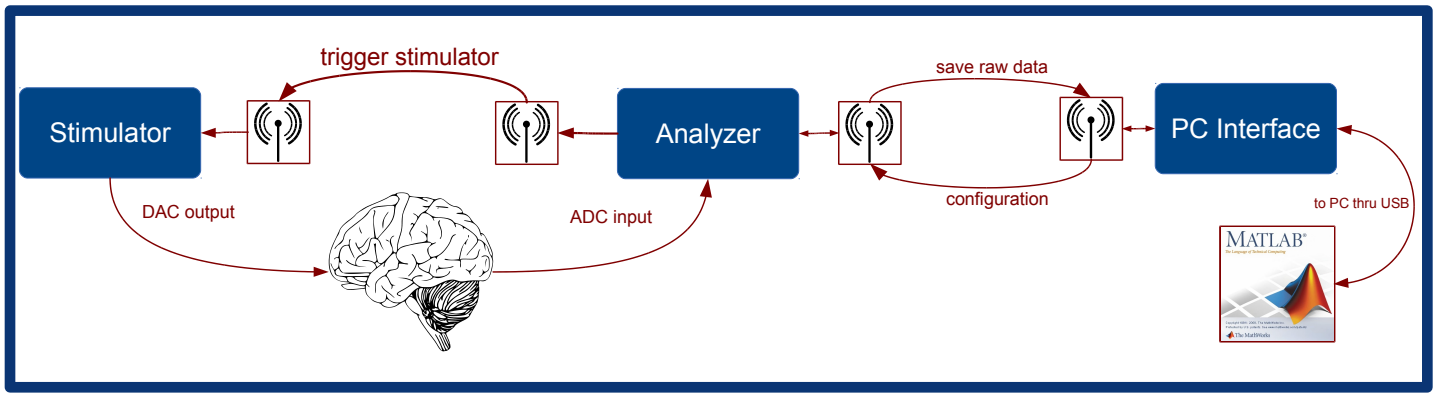

Figure 1. Flowchart illustrating how this BCI system works. The three components of the system are the stimulator, analyzer, and PC interface. The user interacts with a MatLab GUI, which sends the configurations to the PC interface. The PC interface configures the analyzer, which turns on and triggers the stimulator automatically.

II. Closed Loop BCI

A closed-loop BCI device can be described as an implantable device that sits in the skull and analyzes brain activity, and then stimulates certain brain regions in response to that brain activity (basically strengthening "brain connections"). The diagram in Figure 2 summarizes the process visually. The "closed-loop" term refers to the fact that recorded neural signals trigger the stimulating response (which, in turn, affects the recorded signals, etc.) This closed loop strengthens synapses of neurons between two different parts of the brain, as was done using the Neurochip-1 BCI [3]. This is due to a phenomenon called "neuronal plasticity [4]." Not only can a closed-loop system be used to strengthen synapses between two parts of the brain, but it can also provide artificial connections between the motor cortex and the spinal cord [5] or paralyzed muscles [6]. These approaches to neurorehabilitation would be more effective if the BCI system operated for longer periods of time in an unconstrained environment, as in a portable BCI system.

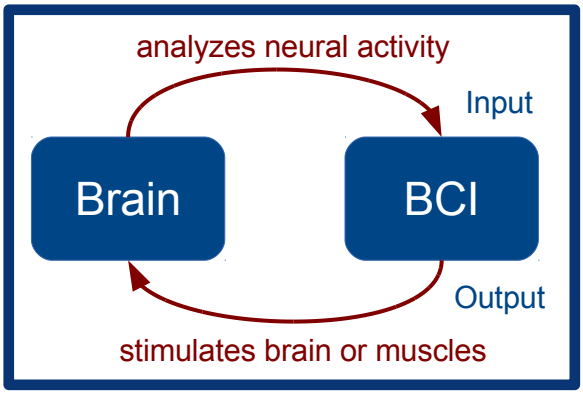

Figure 2. Diagram illustrating how a closed-loop BCI system works. In summary, the BCI uses input from the brain to drive the output to the brain, which strengthens synapses and forms connections in the brain.

III. Microcontroller

The microcontroller (MCU) used for this BCI device is the XMEGA-A1U from Atmel. It is an 8-bit MCU with enough performance and features to be suitable for a general-purpose BCI device. It features a full-speed USB module (used to connect to the GUI application on the PC) and a 12-bit ADC module (used to sample the neural signals at the recording end). The MCU also has four SPI modules to communicate with peripherals (such as the afore-mentioned RF modules for wireless connectivity) and a 12-bit DAC for generating the stimulator signals. With all the extra available processing power (because of the 32MHz processor), there is plenty of room for adding more recording or stimulating channels.

The prototyping platform we used was the XMEGA-A1 Xplained board that was provided by Atmel. It contains all the necessary components all on one board to allow for immediate testing without developing our own custom boards. Much of the testing was made on the Xplained platform, especially for testing the basic software used for the device. After that, we designed and fabricated custom boards for each of the three devices (analyzer, stimulator, and PC interface). Each of these three devices has a separate section describing it below.

The microcontroller software was written on Atmel Studio and uses Atmel's Software Framework (ASF) for the most part. After the software was written and compiled, we used the AVRISP mkII to flash the program to the microcontrollers using the PDI interface. Unfortunately, the AVRISP mkII only supports programming XMEGA microcontrollers but not debugging them. Because of that, we were unable to use breakpoints to debug the software and that slowed down the development to a significant extent. To overcome that obstacle, we resorted to debugging the software by changing the states of I/O pins and monitoring them using an oscilloscope.

The most compelling reason behind choosing the AVR XMEGA architecture as our microcontroller in lieu of other architectures such as PIC or ARM is that the entire XMEGA "A" family of microcontrollers shares a very similar schematic layout. This allowed us to swap different microcontrollers from the same "A" family, depending on the available packaging of these microcontrollers. For example, after the initial prototype board (using the XMEGA-A1) for the analyzer was finished, we designed a smaller prototype board that required a smaller packaging footprint for the microcontroller, and since the XMEGA-A1was only available in the 100-pin BGA packaging, we had to use the XMEGA-A4 in the 44-pin packaging. Since the A1 and A4 come from the same family of microcontrollers, very little of the software needed to be changed and much of the schematics remained the same after swapping the microcontroller.

IV. Wireless RF Transceiver

The RF transceiver used for wireless communication is the NRF24L01 from Nordic Semiconductor. It features a maximum on-air data rate of 2Mbps and runs in the 2.4GHz frequency range and uses GFSK modulation. It operates at about 12 mA of current when receiving or transmitting at full output power of 0dBm. The RF transceiver communicates with the microcontroller using the 4-pin SPI interface at 8 Mbps. Data is sent in the Enhanced Shockburst™ packet format, which includes a two-byte Cyclic Redundancy Check (CRC) scheme and an auto-retransmit with ACK ability. The data to be transmitted and the data received are contained in the data FIFO (first in first out) pipe registers of the RF.

In order to send data across the RF wireless link, the microcontroller has to first set up the RF transceiver with a specific configuration (including which frequency channel to use and the specific address to add to the header). Then, the microcontroller accesses the TX data FIFO and loads the 32 data bytes into it. Finally, the microcontroller has to release the latch "CE" pin to indicate that the RF should transmit whatever is in the TX data FIFO.

The RF transmitter then creates a packet in the Shockburst™ format, as shown in Figure 3. Then, the packet is transmitted on the air (modulated using GFSK) and then received by the other RF transceiver, which then demodulates the message, verifies that the address belongs to it, checks the CRC for any errors, loads the data payload unto its RX FIFO, and transmits an acknowledgment (ACK) packet to the transmitter. In this way, data error rates are minimized while sacrificing the least amount of on-air data rate possible (only 2 bytes for CRC per 32 bytes of data).

Figure 3. This figure illustrates the packet format of Enhanced Shockburst™ used by the RF transceiver. A constant payload size of 32 bytes is used in this project.

We chose this particular RF model for two reasons: the simple SPI interface and the Enhanced Shockburst™ communication protocol, which automatically handles packet retransmission and uses a two-byte CRC for error detection. Although the RF transceiver has proved to be easy to interface with, the increasing demand for more recording channels at higher sampling rates prompted us to search for an alternative wireless transceiver with a higher on-air data rate. In order to avoid frequently switching to incrementally faster and faster wireless transceivers as currently available technology develops, we opted for a long-term solution of developing an Ultra-wideband (UWB) transceiver in-house, which should increase our on-air data rate by an order of magnitude.

Each of the three components of the BCI (described in sections below) uses an RF transceiver to communicate with one another (either as receiver or transmitter). A photo of the RF transceiver on its own can be seen in Figure 4.

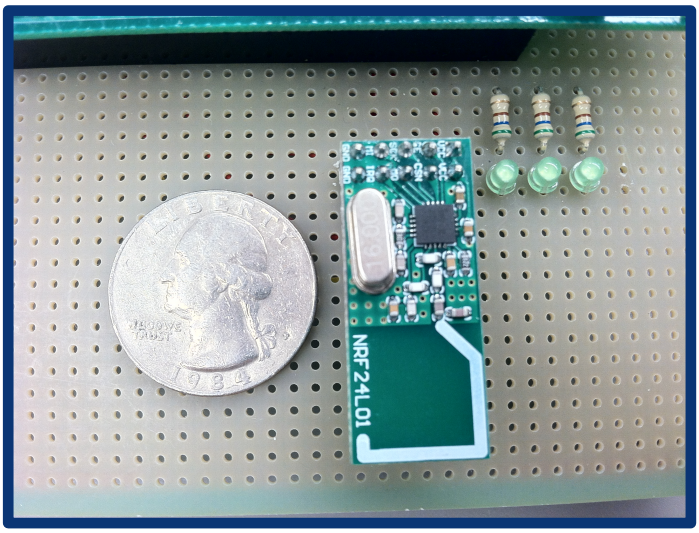

Figure 4. This photo shows the NRF24L01 RF wireless transceiver board with the built-in antenna.

Initial testing for the rate of data loss at maximum on-air data rate has revealed that almost zero loss is observed when the receiving and transmitting RF boards are less than 30 cm away from each other. After a certain distance, the data loss seems to increase exponentially and it becomes necessary to increase the maximum number of auto re-transmits in order to avoid significant data loss. The results of data loss tests for different distances with different auto re-transmit values can be seen in Table 1.

# of retransmit / distance	3 auto retransmit	5 auto retransmit	10 auto retransmit
< 10 cm	0%	0%	0%
45 cm	0.671%	0.041%	0.018%
300 cm	8.78%	4.69%	0.047%

Table 1. This table shows the percentage data loss for the RF when left free running at maximum on-air data of 2Mbps. The tests were performed at different auto retransmit configurations and at different distances. The data is consistent with the fact that the data loss is proportional to the distance between the receiver and transmitter and that the data loss is inversely proportional to the number of auto retransmits. The CRC setting was 2 bytes for these results.

V. PC Interface

The PC interface is the microcontroller board that can be considered the main controller of the platform. It relays all the commands from the PC application and controls the actions of both the analyzer and stimulator. It is connected to the PC through a USB cable and uses the native USB protocol to communicate. This enables it to send and receive data at a very high rate and avoids making this connection the bottleneck in the project.

Because the PC interface uses USB to communicate, the microcontroller on that board is the USB-enabled version (XMEGA-A1U instead of XMEGA-A1). The only difference in these two microcontrollers is that the former is USB-enabled. The software used for the USB connection uses Atmel's software framework (ASF) and the UDI and CDC libraries were used.

The PC interface acts as the "master" of the other two devices, meaning that the analyzer and the stimulator wait for commands from the PC interface before doing anything. When first powered on, the PC interface listens to commands from the USB connection and then acts accordingly.

The commands the user can send are handled through a MatLab application with a simple GUI. The user selects commands to send or specifies required configurations of the devices, and then the application sends the appropriate commands through the USB. A screenshot of the MatLab GUI can be seen in Figure 5.

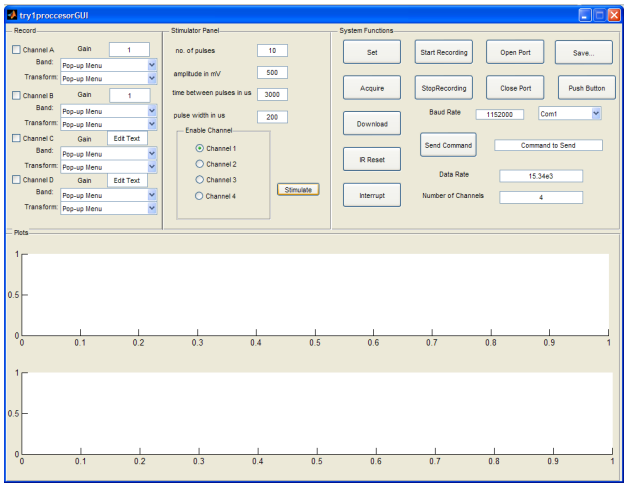

Figure 5. The MatLab GUI application that allows the user to easily wirelessly configure the BCI system depending on the current experiment.

The PC interface is the simplest board in the project, and consists only of the microcontroller on a custom breakout board and a connection to the RF transceiver. Because of this, we used stimulator boards as PC interface prototype boards, as shown in Figure 6 below.

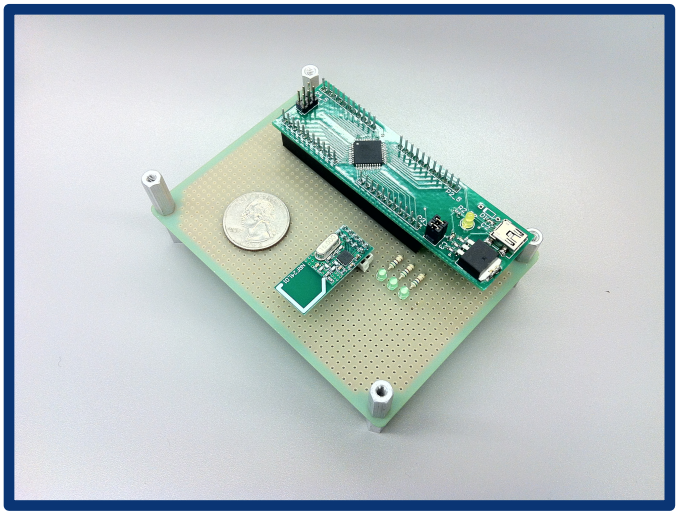

Figure 6. This figure shows **a** photo of one of the prototype boards we used as the PC interface. This was soldered unto a proto board for testing. The RF transceiver can be seen on the left side. The MCU used is the XMEGA-A4U.

VI. Stimulator and DAC

The stimulator is made of three layers connected together using header pins, as can be seen in Figure 7. The top layer is the RF board, which has been discussed in section IV. The middle layer is the MCU board that is responsible for receiving commands and generating corresponding DAC voltage pulses. These pulses are delivered in the form of square waves to the bottom layer, which holds the current-source board. The current-source circuit delivers a fixed current proportional to the pulses received from the DAC.

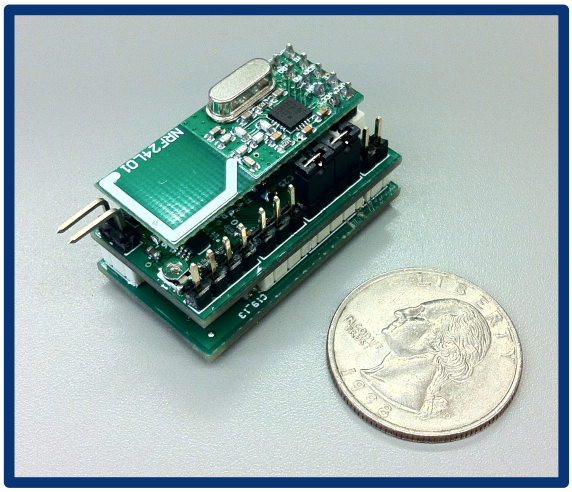

Figure 7. This figure shows a photo of the stimulator board. The top layer holds the RF transceiver, the middle layer holds the current source circuit, and the bottom layer holds the MCU board. The MCU used is the XMEGA128-A4U.

These pulses of current go through the stimulator probes and stimulate the tissue (neural tissue or muscles, depending on the experiment). The current-source circuit has to assure that the pulses are biphasic and symmetrical, and that the same amount of current that goes in comes out. Otherwise, a charge build-up accumulates on the surface of the probes and leads to damage to the tissue with chronic use [7].

The MCU receives the parameters for the required stimulation pulses from the analyzer, and these parameters are:

1. Number of pulses in pulse train
2. Pulse amplitude
3. Time interval between each pulse
4. Pulse width

An example of DAC test parameters used for stimulating facial muscles in a rat is 20 pulses with pulse amplitude of 700mV and 3 millisecond intervals between each pulse and 200 microseconds pulse width. Similar DAC output results can be seen in Figure 8 below.

The MCU uses 2 timers/counters for timing the DAC pulses, one for the pulse width, and the other for time between pulses. The overflow value for the pulse width timer is calculated by multiplying the pulse width time in microseconds and the current system clock frequency in MHz. For example, a pulse width time of 100 microseconds with a system clock frequency of 32MHz gives a timer value of: $100 * 32 = 3200$. This means that every 3200 cycles in the MCU, 100 microseconds have passed and the timer will overflow (triggering the DAC to regulate pulse width). A similar calculation is made for the other timer responsible for regulating the time between each pulse.

VII. Analyzer

The analyzer is composed of a two-sided circuit board, with the MCU on one side and the pre-amplifier circuit on the other. All the configurations for the analyzer are set in the MatLab GUI and sent from the PC interface to the analyzer using the RF transceiver. A photo of the recorder with the attached RF transceiver can be seen in Figure 9.

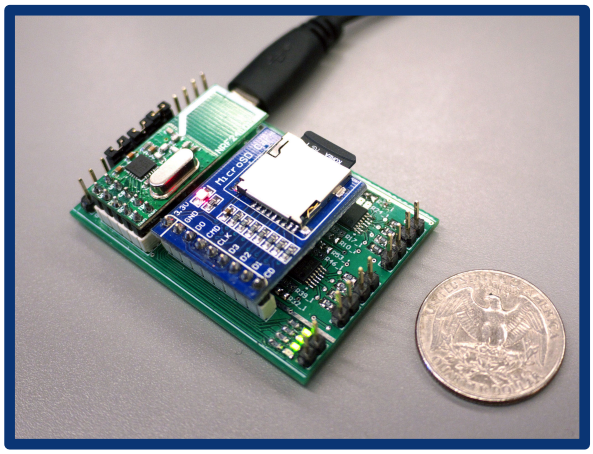

Figure 9. This figure shows a photo of one of the prototype analyzer boards. Note the RF transceiver above the USB connection on the left and the SD card for saving data on the right for later debugging. The bottom layer is the board containing the pre-amplifier circuit on the top side and the MCU on the bottom side. The MCU used is the XMEGA128-A1U

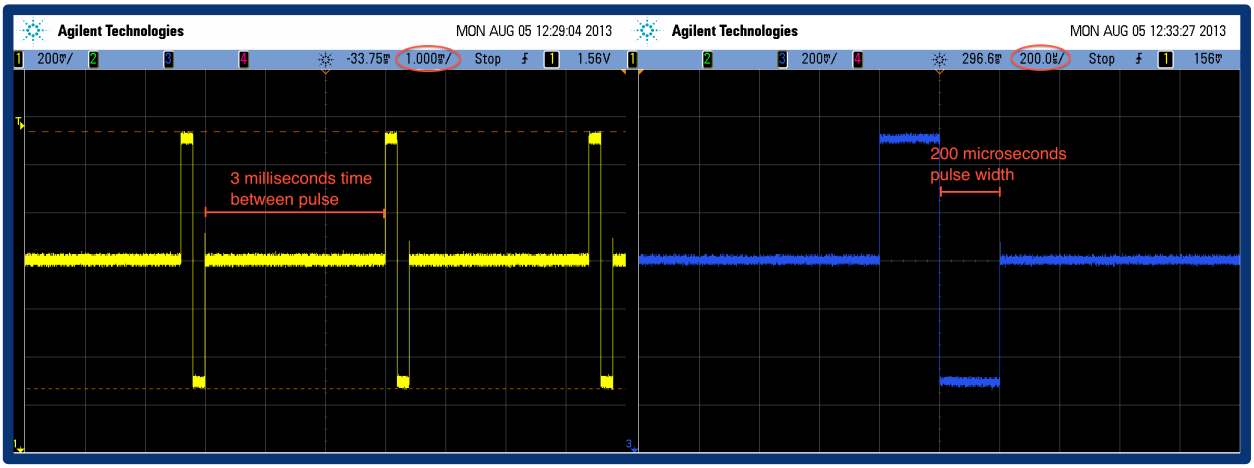

Figure 8. These two plots show the recordings of the DAC output for typical muscle-stimulating pulses (200 microsecond pulse width, 3 millisecond time between pulses). The yellow plot on the left illustrates the 3 milliseconds time between pulses while the blue plot on the right illustrates the 200 microsecond pulse width.

The pre-amplifier circuit amplifies the acquired signal to within a detectable level suitable for the ADC. Then, the ADC samples the signal and stores short segments of it in memory for analysis. The MCU then analyzes the signal's frequency content and looks for events such as when the signal amplitude exceeds the threshold. A suitable stimulus is calculated and the analyzer sends a command to trigger the stimulator with the appropriate settings. The command is sent through the same RF transceiver.

The configurations for the analyzer that are sent by the PC interface are:

1. Channel Number
2. Sampling Rate
3. Gain (for ADC amplifier)
4. ADC resolution
5. Upper and lower levels for band-pass filter
6. Amplitude threshold for spikes

The channel number specifies which channels are to be enabled for acquiring the signal. The sampling rate setting is sent as a 32-bit value and adjusts the sampling rate for the ADC on the MCU (which is actually done by changing the overflow value of a timer that triggers the ADC). Another set of amplification is performed depending on the signal, this time by the built-in gain amplifier for the ADC. Filtering the noise from the signal will be done digitally using a band-pass filter, and the settings are sent as two 32-bit values. Finally, the threshold amplitude is sent as a 16-bit value and specifies the threshold at which a stimulus should be generated.

The ability to configure all these settings wirelessly and from a user-friendly GUI gives flexibility and allows for recording signals from different regions of the brain without any intervention from the BCI designer.

VIII. Discussion and Conclusion

The most deciding factor for the general layout of the BCI system was the fact that saving the raw data and streaming it live to a PC was the highest priority. This effectively meant that the analyzer board needed to wirelessly send the data to the PC somehow. This was the motivation behind creating the PC interface as a means to achieve that. Furthermore, we needed the user to enter the desired configurations on a PC and have them sent to the rest of the BCI system. Having a PC interface board to relay those configurations to the other boards was therefore deemed absolutely necessary. The fact that the PC interface had to communicate to two different boards and sometimes in a bi-directional fashion added significant complexity to the project.

The reason behind splitting the functions of the analyzer and stimulator into two separate boards was to enable our BCI system to stimulate in distant locations from where the recording is being done. If both analyzer and stimulator were merged in one compartment, the stimulation would have to be physically located close to the recording. This would make muscle stimulation while recording brain signals more complex and difficult to perform on unrestrained primates. Additionally, since the communication between the analyzer and stimulator is very sparse and consumes very little of the available 2Mbps bandwidth, there was no issue with the commands being sent wirelessly.

With these considerations, we decided to split the BCI system into the three parts mentioned in earlier sections: the PC interface, analyzer, and stimulator. In order to keep the three devices synchronized in the same operating modes and avoid one device stalling in the incorrect mode, we implemented a simple timeout mechanism so that the devices switch to the default mode if the other devices are unresponsive after a timeout period of 500 ms. This avoids situations where one device is trying to transmit a command to another that is not in the receiving mode.

As it stands, the current BCI system lacks in the ability to send the raw data to the PC at a high enough rate. As discussed earlier, this is due to the maximum on-air data rate for the RF transceiver of only 2Mbps. A custom UWB transceiver is being designed and is planned to be incorporated into the current BCI system. This should remove the wireless bandwidth limitation, and allow data transfer at a much higher rate.

The next limitation comes from the fact that MCUs are rather lacking in their ADC specifications and in their efficiency in performing digital signal processing. We foresee that, after the entire process is properly optimized, we will replace the current MCUs with digital signal processors (DSPs) designed particularly for our BCI system. Additionally, custom ADCs with much higher specifications than the current built-in ones are being designed and will be implemented, along with the previously mentioned upgrades, to the next iteration of our BCI platform.

Acknowledgment

The authors would like to acknowledge the support of the National Science Foundation, through NSF REU grant no. 1062672.

References and Footnotes

[1] Miranda, H., Gilja, V., Chestek, C. A., Shenoy, K. V, & Meng, T. H. (2010). HermesD: A High-Rate Long-Range Wireless Transmission System for Simultaneous Multichannel Neural Recording Applications. *IEEE Transactions on Biomedical Circuits and Systems*, *4*(3), 181–191. doi:10.1109/TBCAS.2010.2044573

[2] Mavoori, J., Jackson, A., Diorio, C., & Fetz, E. (2005). An autonomous implantable computer for neural recording and stimulation in unrestrained primates. *Journal of Neuroscience Methods*, *148*(1), 71–77. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/16102841

[3] Jackson, A., Mavoori, J., & Fetz, E. E. (2006). Long-term motor cortex plasticity induced by an electronic neural implant. *Nature*, *444*(7115), 56–60. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/17057705

[4] Arcangelis, L. De. (2008). Activity-Dependent Model for Neuronal Avalanches. (G. Franzese & M. Rubi, Eds.)*Notes*, *752*, 215–230. doi:10.1007/978-3-540-78765-5

[5] Jackson, A., Moritz, C. T., Mavoori, J., Lucas, T. H., & Fetz, E. E. (2006). The Neurochip BCI: towards a neural prosthesis for upper limb function. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *14*(2), 187–190. doi:10.1109/TNSRE.2006.875547

[6] Moritz, C. T., Perlmutter, S. I., & Fetz, E. E. (2008). Direct control of paralysed muscles by cortical neurons. *Nature*, *456*(7222), 639–642. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18923392

[7] A. Scheiner, J. T. Mortimer, and U. Roessmann, "Imbalanced Biphasic Electrical Stimulation: Muscle Tissue Damage," *1990 Proceedings of the Twelfth Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 18, no. 4, pp. 407–425, 1990.