A Closed-Loop System to Monitor and Reduce Parkinson's Tremors

Tremors Group: Anthony Calvo, Linda Gong, Jake Miller, and Mike Sander
Faculty Advisor: Dr. Gary H. Bernstein
8 March 2018
Design Review I

Table of Contents

I. Overall Requirements …… pg. 2

II. Overall Block Diagram ……..……...……..……. pg. 3

III. Subsystem Details …...…..……...……...……. pg. 3

I. Overall Requirements

Goal: A safe, lightweight, tremor-monitoring bracelet suitable for daily use that interfaces to an EMS tremor reduction system.

There are quite a few constraints and specifications that the system needs to meet in order to function properly. To meet our goal, our system will implement a wearable bracelet, cell phone application, and EMS machine. The bracelet and cell phone application will communicate via Bluetooth.

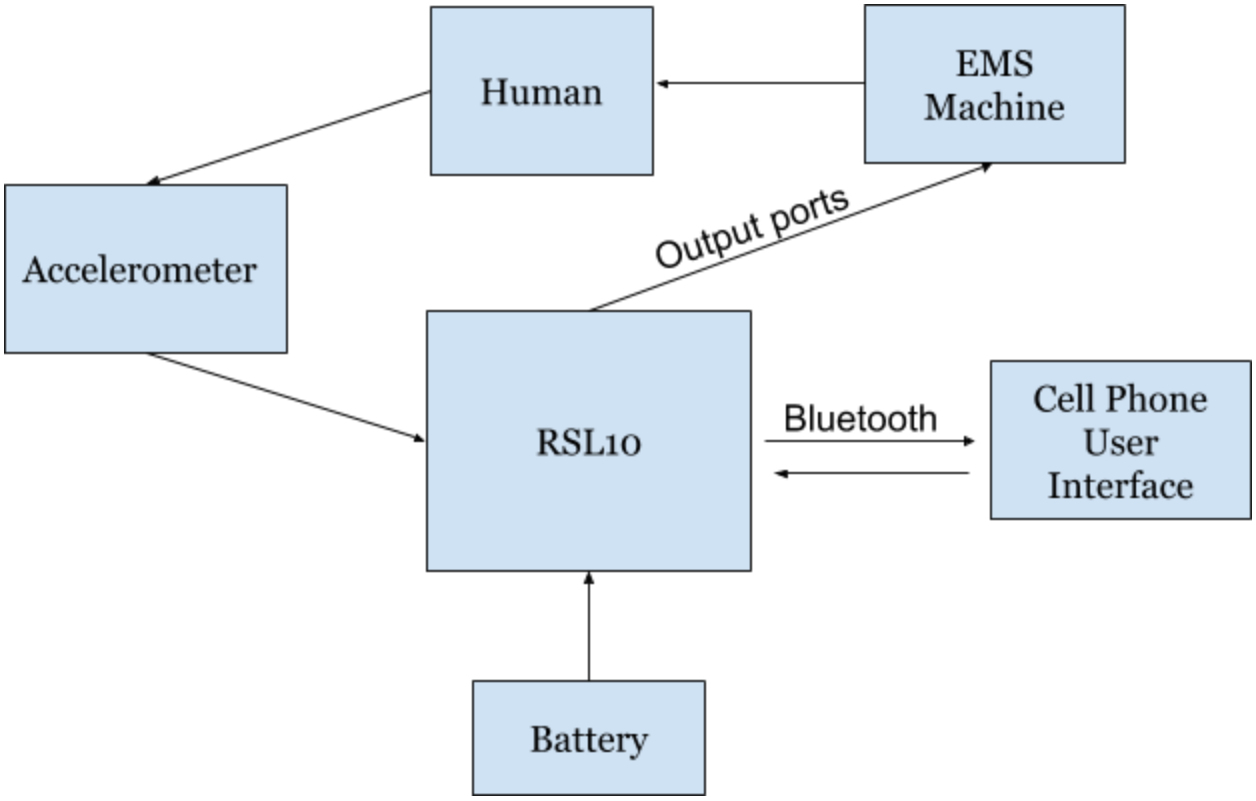
Focusing first on the wearable bracelet, the bracelet needs to be small and light enough to fit comfortably on a human wrist. The bracelet will house the RSL10 and accelerometer to allow for proper data collection. The accelerometer will measure X, Y, and Z acceleration based on the movements and tremors of the user.

The RSL10 will collect this data and will relay this information via Bluetooth to a cell phone application. The bracelet components (RSL10 and accelerometer) will be powered by a battery. The battery should have at a 48 hour lifetime (based on the battery life of a Fitbit), and a recharge time of 5 hours.

The cell phone application will communicate via Bluetooth with the bracelet module and perform signal processing on the data to determine if tremors are occurring. The app will have a graphical user interface (GUI) that displays whether tremors are being detected. In addition, the app will save tremor data to memory to allow the user to review his/her tremor history.

The EMS machine will be controlled with GPIO pins on the RSL10. When tremors are detected by the bracelet, the EMS machine will apply an electric signal to mitigate the tremors. The EMS can be powered by either battery or a wall power supply. Adjustments will be made to the output of the EMS based on feedback (e.g. is the tremor reduced?) to allow for variations in the magnitude of muscle stimulation signals.

II. Overall Block Diagram



III. Subsystem Details

1. Signal Processing

The signal processing subsystem is implemented within the cellphone application. Using Swift's vDSP library, a Fourier transform is performed on the data. For the first subsystem demo, an FFT is performed on dummy data to demonstrate that we can determine when tremors are occurring. One set of dummy data contains a sine wave with a frequency of 5 Hz representing movement the presence of tremors. Another set of data contains random data to simulate movement without the presence of tremors.

2. Accelerometer

The ADXL345 breakout board is used as the digital accelerometer for the first design review because it is I2C enabled. The RSL10 is configured as the I2C master and the ADXL is the I2C slave, which allows the RSL10 to read data from the ADXL. Using a multibyte read, acceleration data is read from the x/y/z directions. For our first subsystem demo, we are able to demonstrate a write/read from the ADXL and gather x/y/z data.

3. Bluetooth

The Bluetooth subsystem utilizes UART and the DMA data transfer registers to send multiple bytes of data between the RSL10 and a paired cell phone. The program is built off the existing peripheral UART code that was provided by On Semi.

The RSL is first in "pairing mode" and sends advertising packets while blinking the onboard LED. Using the nRF Connect cellphone application, we are able to connect and pair to the RSL10. This allows the board to enter into "connection mode." The data that is sent from the RSL10 is generated from the rand() function for now, but can be adjusted to receive data from an I2C buffer instead. The data is sent when the UART interrupt is triggered, which currently is done so by pressing down on the keyboard when the serial terminal is open.

A second custom connection allows data to be sent from the cellphone application to the serial terminal and the UART buffer. This gives us the option to send data back to the RSL10 in order to provide feedback to the EMS system. Since we plan on doing most of the signal processing within the cell phone application, this connection will be useful in the future.

4. EMS

The EMS subsystem is a modification of the existing MPO 8500 Combo EMS device so that it can be controlled by the RSL10. The RSL10's output ports will be used to drive circuitry (MOSFETs and resistors) that electrically short the EMS device's buttons. This will allow the system control of the EMS output's amplitude, frequency, and pulse width. We will demonstrate its functionality by showing that voltages can be used to electrically "press" (short out) EMS buttons.

5. Cell Phone App

The cellphone application was created for an iPhone application using XCode and Swift on a MacOS. The cellphone application will store and display the data collected from the accelerometer. The app is Bluetooth enabled to allow for data transfer between the RSL10 and the cellphone. The signal processing is also done within the cell phone app using Swift's vDSP libraries. For the subsystem demos, Fourier transforms were performed on dummy data and displayed. One set of dummy data was created with the built in sine function to represent tremor motion at 5Hz. The other set of dummy data was created using a random number generator to represent movement without the presence of tremors. A notification showed if a tremor is occuring (sine wave) or not occurring (random data); the Swift-FFT-Example by christopherhelf (https://github.com/christopherhelf/Swift-FFT-Example) was used to help with this portion of the code. To show Bluetooth using code from Xcode, we followed an example from Adafruit (https://learn.adafruit.com/crack-the-code/overview) and used the Adafruit Feather M0 with an Arduino IDE to show that data could be sent from the phone via UART with the app that was coded in Xcode.

6. Battery

The power system subsystem will be responsible for powering the RSL microcontroller while it collects data from an accelerometer via I^2C and transmits it using Bluetooth to the cell phone, and receives control data for the EMS machine from the cell phone app. The RSL also must power its outputs such that it can control the transistors that control the EMS machine. The battery chosen to power the RSL must be capable of providing the right voltage to the chip while also providing enough current to power the chip for the uses listed above. The battery must also have enough capacity to power the RSL for at least 48 hours. Due to the wearable nature of the RSL device, the battery must also have a small profile such that it is not uncomfortable to wear on the wrist along with the RSL device. For this reason, we will be using a coin cell or multiple coin cells, provided that one exists that meets all other requirements.

The RSL has a DC-DC converter on chip, meaning we don't need to have one external to the chip in order to regulate battery fluctuations in Voltage. The RSL has a supply voltage range of 1.1 to 3.3V. We are choosing to run at the ~3V range so that digital output pins run at a higher voltage, making our transistors for controlling the EMS machine run as we expect.

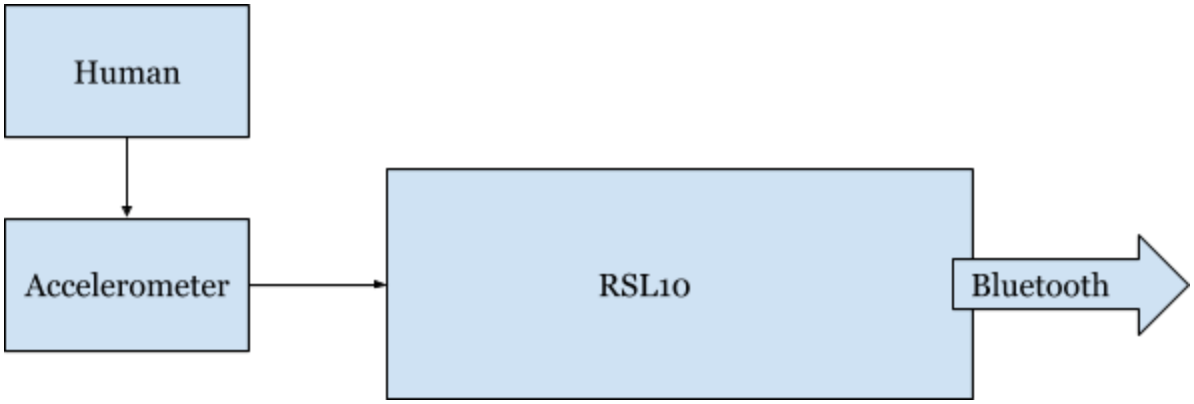
At 3V, the Bluetooth max receiving current is 3mA, and the transmitting max current is 4.6mA. The spec sheet lists audio streaming with the RSL as using 1.8 mA continuously (at 3V); we are using these estimates for our own average power requirements for the time being. We've measured the current of each button controlling transistor at 5mA-7mA; to allow room for error, we'll assume a max possible of 10 mA. Since we don't expect to push more than 1 button at a time but we want room for button-to-button press overlap, this makes the max current requirement for the transistors 20 mA. This means our battery should be able to provide a total max current of 27.6 mA. If we assume that Bluetooth receiving and transmitting occur at all times at 1.8 mA, and that one button is running 20% of the time, then our expected average current is 3.8 mA. This puts our battery capacity requirement at 182.4 mA-hr.

The 2032 coin cell from Energizer has a capacity of 240 mA with a cutoff voltage of 2V. We would like our supply voltage to be at least 2.9V; this cuts the capacity down to about half (120 mA) based on current discharge characteristics provided by Energizer. This would mean we need two in parallel to provide the needed capacity. Energizer doesn't provide a max current, but lists pulse discharge characteristics at 23 mA. Therefore, two in parallel will be able to provide pulses at 46 mA, more than enough for our max needed current.

Note: The RSL will not receive Bluetooth data if the user is not currently experiencing tremors. This could potentially mean that we can ignore the receiving current some amount of the time; however, Parkinson's tremors often occur at rest, so we want the device to be able to help someone who could hypothetically be at rest 24/7.

Note on needing multiple batteries: This system is a potential solution for Parkinson's patients when doing tasks when tremors would cause trouble. Therefore, needing multiple coin cells is justifiable.

Bracelet Subsystem Block Diagram:



EMS Subsystem Block Diagram: