Despite minor setbacks, this week has provided significant net progress. In addition to finally connecting the MPLAB ICD2 module to the microcontroller for programming, code has been written for the motors, a binary switch sequence, and I2C communication. While all of the code shows signs of completeness, so far only the motor programming has been successfully implemented.

Having successfully connected the microcontroller to the microchip software, we immediately programmed the motors to test our code. Rather than using Pulse Width Modulation (PWM), we are “manually” producing pulses that stay “high” and “low” for the specified amount of time. Although the timing needs to be perfected to produce the optimal pulse lengths, the motors are reacting positively to the code. Additionally, the specific microcontroller we are using may not be fully functional as indicated by its erratic behavior. I am confident that once the microcontroller is replaced, the motor system will be completed and work can continue of other subunits.

The most integral set of code developed this week has been the I2C communication. Once this set of code is modified correctly it will allow for easy integration of the Real Time Clock and the SP03 Text to Speech Module. An abbreviated programming process is listed here: Before data is transferred, a series of steps must occur to initialize the transaction. First, the programming must ensure that both the SDA and SCL lines are at a constant “high” value. The voltage of a “high” varies based on the supply voltage (Vdd) to the serial BUS. In our application a “high” value will be +5V and a “low” will be 0V. The relationships between the transition timing from high to low on the SDA and SCL lines is the basis of data transferal along the I2C BUS. Once the values of each line is set to high, the “start” transaction will be initiated by switching the SDA from high to low while the SCL is kept constant. During a data transfer, the SDA can only change when the SCL is at a low value because the transitioning from high to low or vise versa while
the SCL is high is reserved for start and stop transactions. Once the start transition has been issued, all devices on the BUS will “wake up” and ready themselves for a possible data transfer. The next step is to specify the address of the device to which you would like to send data. This address will be comprised of a seven bit value and is unique to each device. If any unit on the BUS recognizes the address as its own, it will prepare itself for receiving or sending data. Because the SDA is bidirectional, the mast can then choose to read data from the slave or to write data to the slave. This specification will come as the 8\textsuperscript{th} or least significant bit in the address specification. A value of 1 will signify that the master wants to receive data from the slave while a value of 0 will establish the master as the data transmitter. It is important for monitoring the connection that the slave device acknowledges a successful communication initiation by responding to the master with an “ACK” command. This command will allow the master to then begin sending or receiving data. Once data has been sent, communication is terminated by a stop command during which the SCL is floated at a high value and the SDA transitions from low to high.

Figure 1. Motor System
Future Work:

Now that the microcontroller can be easily programmed, we must perfect and implement the remaining code sequences. Additionally, the hardware must be finalized and a PCB layout must be designed and ordered.

Hours Worked: 12 Hours