Intro

There are three hardware components. The flex sensors, the LCD display, and the Bluetooth module are separated into three different libraries.
The first library is called “AtoDC.c/AtoDC.h”. It uses the highspeed A/D converter to get voltage values from the sensors and process them as either 1s or 0s. The first function of this library is the ADC initialization (adc_init) which set all the required pins to input and links the ADC to timer3 on the pic24. Every 0.0125ms the ADC gets data from the sensors. However, the pic24 can not sample ADC data from several peripherals at the same time, therefore there is a function to select the channel (select_ch). All the buffer values are initialized to 0 (initBuffer) and then the ADC starts going through every pin one after the other so every buffer is filled with the same amount of values. That means that each ADC channel receives data every 0.0625ms. Then there is a function that computes, for each channel, the average of all the values in the buffer (getAvg(intA[])). Each channel has then a function that determines if the fingers are bent or not depending on an intermediate value determined beforehand depending on the design and the glove (digitalValShort_(float A) or digitalValLong_(float A) depending on the finger). If the flex sensor is bent enough then it returns 1, otherwise, it stays 0. Finally, there are functions that return the digital values of each flex sensor so it can be used outside of the library (finger_())

The second library, I2CDisplay.c/I2CDisplay.h, uses the pic24 inter-integrated circuit (I2C) that is documented in the pic24 family datasheet. The initialization function just sets everything up by sending a bunch of packages using a function called lcd_cmd. That function takes a single byte/char command and writes it out the I2C bus. The complete packet consists of a START bit, address with R/nW byte, control byte, command/data byte, and STOP bit. The function used to display a character, lcd_printChar() is the same with a different R/nW byte address. There is also a function called lcd_SetCursor that takes coordinates as parameters and selects the location on the display (2x8). The last function, lcd_printStr(), used to print strings is just a bunch of char printing with the cursors being changed accordingly.

The third library (UARTBluetooth.c/UARTBluetooth.h) is just the initialization (initUart() and its ISR) of the Bluetooth module through the pic24 UART functionality and the translation of the message into a digit between 0-9.

Hardware description

The main hardware elements are the flex resistors. The voltage across this type of resistor changes depending on how bent they are because the impedance changes. Two types have been used for this project. Short flex sensors with a resistance that is 25k when it is flat and is between 45k and 125k depending on the bend radius. Long flex sensors with a resistance that is 10k when it is flat and is between 60k and 110k depending on the bend radius. (short Flex sensor: https://cdn-shop.adafruit.com/datasheets/SpectraFlex2inch.pdf ;

Long Flex sensor: https://cdn-shop.adafruit.com/datasheets/SpectraFlex.pdf)

The I2C (inter-integrated circuit) provided in the pic24 to use the I2C LCD Module - AQM0802A-RN-GBW
http://akizukidenshi.com/download/ds/xiamen/AQM0802.pdf

The Bluetooth module Hc-06 is a Bluetooth serial port module with master-slave integration.
https://www.amazon.com/HiLetgo-Wireless-Bluetooth-Transceiver-Bi-Directional/dp/B07VL6ZH67/ref=sr_1_3?crid=17S7HF6S8GW1P&keywords=hc-06+bluetooth+module&qid=1651607331&sprefix=hc-06+%2Caps%2C110&sr=8-3

Documentation and usage examples

The first public function, ASL_value(int f0,int f1, int f2, int f3, int f4) just translates the binary values (fingers straight or bent) into actual ASL numerical values (0 to 9). There is a default value that happens when the binary sequence does not mean anything in ASL. That means that the function takes the binary results from the ADC library and returns a char. The second public function just displays (it is a void function) the binary sequence ASL numerical value and the binary sequence itself in which each digit corresponds to a finger.

A basic usage example to test the functionality of the hardware: all the sensors can be wired to the PIC24 without any complicated prototype just to verify that the ADC works. To do this, we wired the flex sensors to the pic24, processed their analog values through the ADC, and displayed them on the LCD screen.

An advanced usage example would be the whole prototype with all the sensors appropriately attached to the glove to test the design with the binary sequence and its corresponding numerical value that would be sent via Bluetooth to another device for more complicated purposes.