Software Overview

Year: 2018 **Semester:** Fall **Team:** 07 **Project:**Handi_glove
Creation Date: September 4, 2018 **Last Modified:** September 4, 2018

Member 1: Yaodong Shen **Email:** shen234@purdue.edu
Member 2: Jia En Chua **Email:** chuaj@purdue.edu
Member 3: Yao Chen **Email:** chen1748@purdue.edu
Member 4: Carol Lo **Email:** lo40@purdue.edu

Assignment Evaluation:

Item	Score (0-5)	Weight	Points	Notes
Assignment-Specific Items				
Software Overview	5	x2	10	
Description of Algorithms	4.5	x2	9	
Description of Data Structures	4	x2	8	
Program Flowcharts	5	x3	15	
State Machine Diagrams	5	x3	15	
Writing-Specific Items				
Spelling and Grammar	5	x2	10	
Formatting and Citations	5	x1	5	
Figures and Graphs	5	x2	10	
Technical Writing Style	4.5	x3	13.5	
Total Score	95.5			

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

Good job.

1.0 Software Overview

Software mainly covers the logic flow of four major parts in this project which are temperature feedback, pressure feedback, motion control, and motion restraint. There is no need for sophisticated algorithms to implement the software logic; however, we need to work out some values like (PWM duty cycle and the voltage value for Peltier Module) for feedback systems to function correctly.

The program will make sure that the robotic hand is in the default position when power turns on. Then the movement of each finger will be tracked through a potentiometer and these signals will be transmitted to the microcontroller which resides on the glove. Communication packets will be transferred to the microcontroller on the robotic arm. In the current implementation, we will have a physical connection between two microcontrollers. If time allows, we will go above and beyond to implement wi-fi features in terms of microcontroller communication. The microcontroller on the robotic arm will unpack the communication signals and generate the appropriate voltage for the servo to pull the tension cable and move each robotic finger.

To achieve temperature feedback, the temperature sensor value has to be transferred to the robotic arm microcontroller via ADC. Then the microcontroller will be responsible to calculate the amount of voltage for the Peltier Module. After the calculation is done, the value will be sent as a communication packet to the other microcontroller on the glove. Upon receiving the packet, the software program will output a certain voltage value for the Peltier Module to generate appropriate heating or cooling effects.

The linear actuator, syringe, and airbag are responsible for generating touch feedback. When pressure sensors on the robotic fingers detect changes in pressure, the RTI will trigger the program to calculate the voltage for the linear actuator to push the syringe. The airbag will be expanded and tactile feedback is achieved.

2.0 Description of Algorithms

The core software algorithm will be the system interacting between receiving the signal from sensors and outputting an appropriate action accordingly. Analog to digital converter (ADC) modules are needed to convert analog signals from different sensors(i.e. temperature and pressure). The microcontroller will utilize pulse width modulation (PWM) [1] module to control the speed of the servo and the linear actuator. Specifically, we will be using linear actuators to push the syringe and control the volume of air inside the airbag. Servos are utilized to pull the tension cable on the robotic hand and to restrain the movement of the glove through the exoskeleton.

The functionality of the PWM [1] is vital to our project. We will determine a voltage value to modulate the pulse width so that a desirable duty cycle to control the magnitude of the servo and the linear actuator is obtained. The algorithm will be a function receiving voltage value as a parameter and returning a value indicating the appropriate output to move the servo. There will be simple mathematical equations in the function to calculate proper duty cycle according to the input voltage.

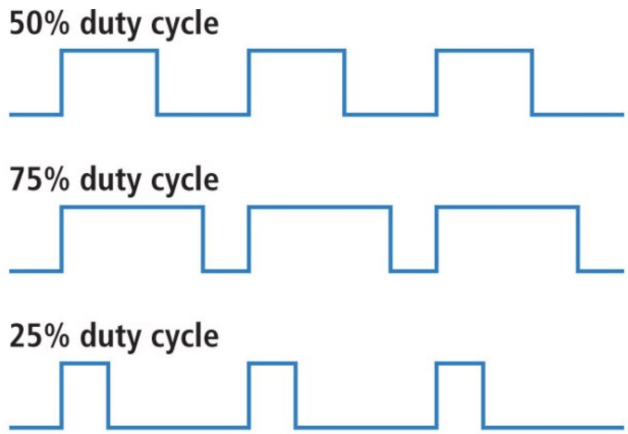


Figure 1. functionality of the PWM[1]

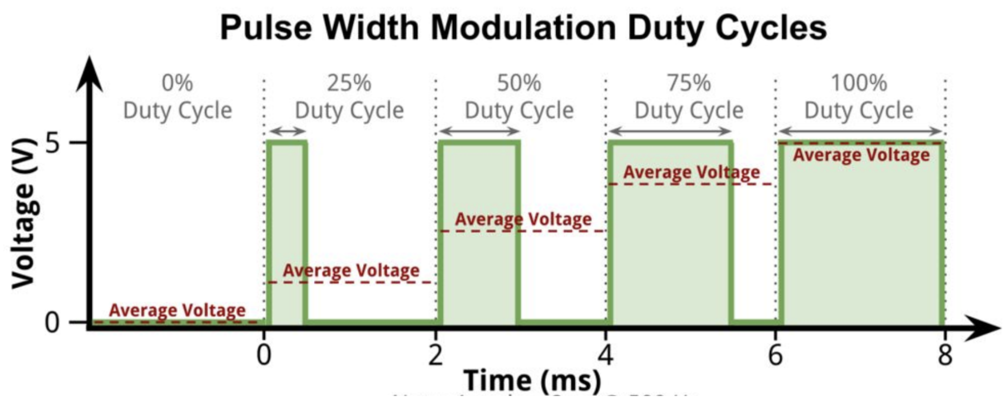


Figure 2. Pulse Width Modulation Duty cycles[2]

To achieve temperature and pressure feedback, the microcontroller will receive value from temperature and pressure sensors in real time through real time interrupt (RTI) or a similar module and calculate an appropriate value for Peltier Coolers and linear actuators to generate temperature and tactile feedback. We will also limit the maximum and minimum temperature and pressure that can be felt by the user, this feature is introduced due to the consideration of safety. Initially, we will place the pressure sensors and generators on the fingertips while heat sensors and generators on the palm. By doing so, we avoid inadvertently allowing air to expand in airbag through the heat generated from the Peltier module. Ultimately, we aim to place both temperature and pressure modules together on the fingertip. We will have several functions to adjust to this conflict: such as decreasing the air pressure when it is affected by the Peltier module heat. We will first achieve the basic functionalities before we get into advanced design implementations.

The RTI will trigger an interruption when the system detects the hand leaving the glove. When the interruption kicks in, the program is designed to call a helper function which brings the robotic fingers back to the original position. This can be achieved by changing the current set of values on the servo back to the initial set of values which is set as a constant initialized at the beginning of the program. We will also have a helper function in the case where power is unexpectedly cut off. Similar to the case when the user takes off their glove, the robotic hand will always return to its initial position upon powering on even when it was powered off in a different position.

3.0 Description of Data Structures

There will be communication packets transmitting between microcontroller on the robotic hand and the microcontroller on the glove. SPI Communication packets will consist of 8-bit instructions and each instruction will be broken down as illustrated below.

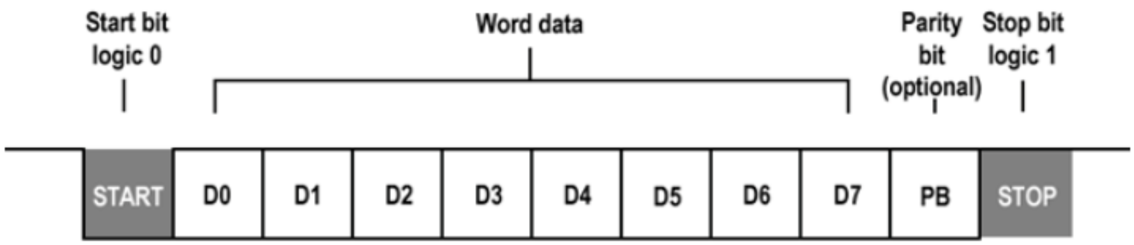


Figure 3. Instruction within communication packets[3]

We need an array or hashmap to store constant variables which represent the initial values of the robotic hand position. By utilizing hashmap, we can set a certain position as the key and set the PWM as the value of hashmap so that the code will be more intuitive and readable.

4.0 Sources Cited:

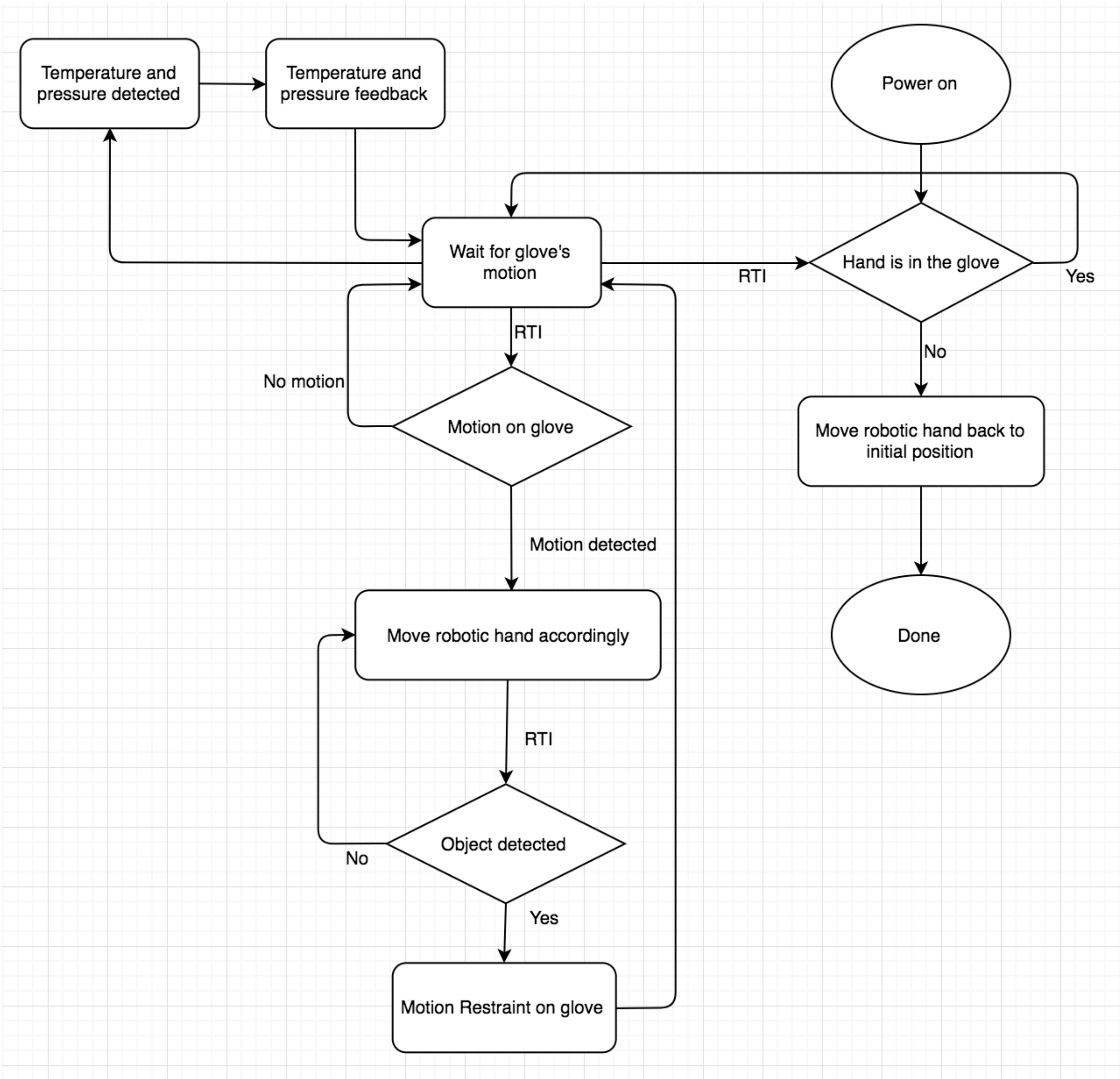
[1] Jordandee, “Pulse Width Modulation,” *learn.sparkfun.com*, n.d.. [Online]. Available: https://learn.sparkfun.com/tutorials/pulse-width-modulation. [Accessed Sept. 07, 2018].

[2] D. Connolly, “Linkit One and PWM (Pulse Width Modulation),” *instructables.com*, n.d.. [Online]. Available: https://www.instructables.com/id/LinkIt-One-and-PWM-Pulse-Width-Modulation/. [Accessed Sept. 07, 2018].

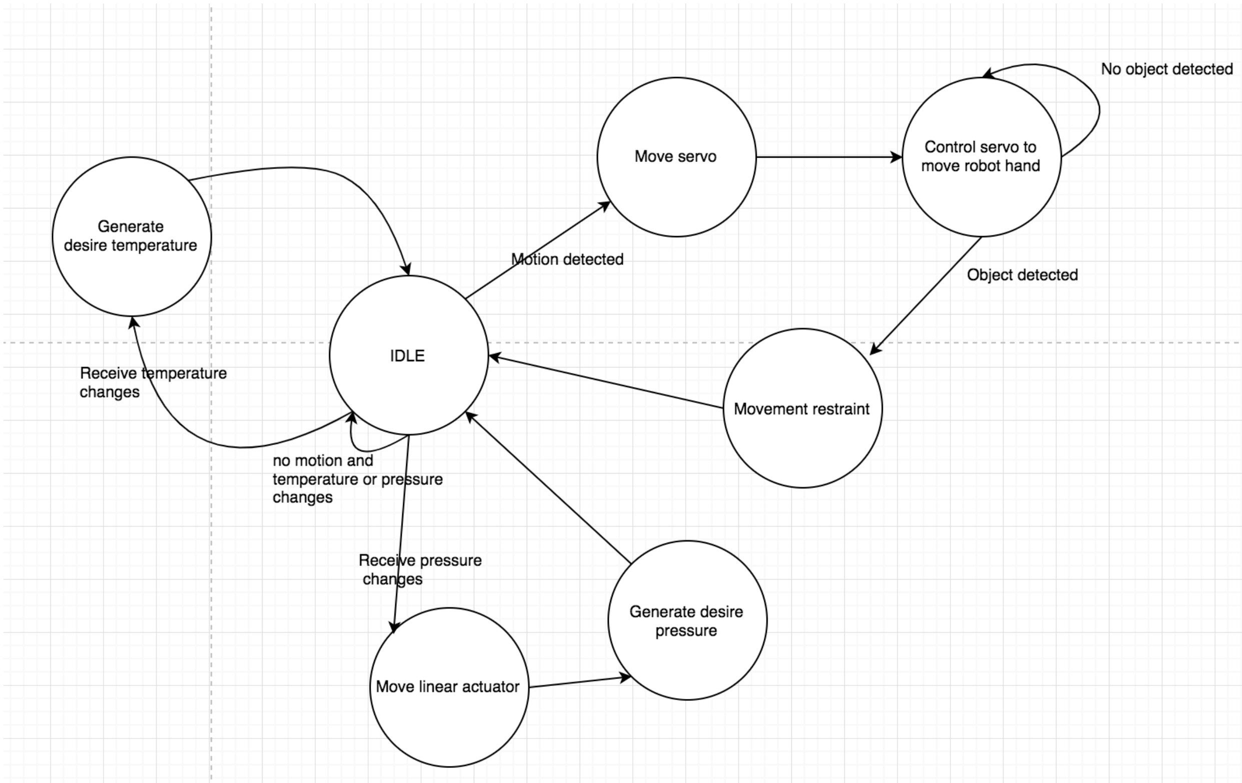
[3] U. Lokhande, “UART in LPC2148 ARM7 Microcontroller- Serial Communication,” *binary*

updates.com, 01-Jul-2018. [Online]. Available: http://binaryupdates.com/uart-in-lpc2148-arm7/. [Accessed: 07-Sep-2018].

Appendix 1: Program Flowcharts



Appendix 2: State Machine Diagrams


No object detected
Move servo
Control servo to move robot hand
Generate desire temperature
Motion detected
Object detected
IDLE
Receive temperature changes
Movement restraint
no motion and temperature or pressure changes
Receive pressure changes
Generate desire pressure
Move linear actuator