Software Overview

Year: 2017 Semester: Fall Team: 15 Project: Super Susan
Creation Date: 9/8/2017 Last Modified: March 3, 2015
Author: Ziad Dannawi Email: adannawi@purdue.edu

Assignment Evaluation:

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

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

General Comments:
Relevant overall comments about the paper will be included here

1.0 Software Overview

The Super Susan will have several modules which will be communicating with our microcontroller. Specifically, those are the weight sensors, the WiFi module, and the NFC reader/writer. The microcontroller will be responsible for taking analog input from the sensors (weight in this case), and converting them to a usable format. The weight sensing is being done through a strain gauge load cell, this means that small enough resistance changes require amplification [1] to be read properly by the microcontroller. The HX711 Analog-to-Digital Converter will be used with our weight sensor. On the software side, the HX711 can only interface using SPI, so the microcontroller will have to be able to support this.

A state machine will be implemented in the microcontroller to control the Super Susan. Upon startup the Super Susan will do nothing until the user turns the dial, or uses their smartphone, to communicate with the Super Susan. At that point, depending on what option the user selects, the state machine will branch out accordingly. If the user wants to update the inventory, the Super Susan will have to go through all of its entries and get an updated weight for each entry. If the user wants to select a spice, the Super Susan will have to locate the spice then rotate it towards the user. If the user wants to add a new spice to the Super Susan, the Super Susan has to provide the user with an area to place their new spice bottle, then it has to ask the user what the spice is (this will either be entered manually, or through a barcode scanner), then it will record the slot and weight of the newly placed bottle. There will also be an option in the menu for the user to enter their WiFi settings.

2.0 Description of Algorithms

The main algorithm required would be to know where each spice is located, and how to rotate whichever selected spice towards the user without any mishaps. The algorithm will treat the Super Susan's concentric circular spice holders as an array of positions, where each position is an individual spice holder identified by its respective Hall Effect sensor, irrespective of whether or not that particular spice holder happens to house a container or not. In the array of positions, there will be a position known as the "head", which is relative to the location of our NFC reader, and will be the position on the staging area. This position should always be known, so it will be possible to calculate how many "jumps" one would have to take to position whichever selected spice towards the user. A "jump" is simply the passing of a position (or spice bottle) over the head.

Since the Hall Effect sensors are connected to the base, they will not be rotating with the assembly, hence the "position" reading will not be able to know which spice is currently on it. One solution is to have an n-element array, where n is the number of spice holders on the Super Susan, indexed by the previously mentioned "positions", with even indexes belonging to the inner circle, and odd indexes belonging to the outer circle. The array will store the NFC IDs of the bottles in the correct place in the array, and this array will be shifted n number of times left or right depending on the rotation direction, where n is the amount of “jumps” that were previously discussed. This method will allow the Super Susan to track the locations of all current spices while maintaining the position system.

3.0 Description of Data Structures

One data structure that will be be used is to communicate with the ESP8266 WiFi chip and relay commands to it. It can be sent a ‘set’ command, an ‘inquiry’ command, a ‘test’ command, and an ‘execute’ command as follows: AT+<x>=<..>, AT+<x>?, AT<x>=?, and AT+<x> respectively, where x is the desired command, according to the data sheet [2]. These commands will be sent from the microcontroller to the WiFi chip through SPI, as will the other components in the system.

COMMAND	COMMAND CODE								FUNCTION
	D7	D6	D5	D4	D3	D2	D1	D0	
Write Display Data	Data to be written into the Display Data Memory.								Write a byte of data to the Display Data Memory.
Read Display Data	Data read from the Display Data Memory.								Read a byte of data from the Display Data Memory.
Read-Modify-Write	1	1	1	0	0	0	0	0	Start Read-Modify-Write operation.
END	1	1	1	0	1	1	1	0	Stop Read-Modify-Write operation.
Software Reset	1	1	1	0	0	0	1	0	Software Reset.

Fig 1. Newhaven Display command structure [3]

Another data structure that is worth noting is related to the Newhaven Display that we are using. As indicated by figure 1 above, there are specific 8-bit codes that correspond to different commands. It is worth noting that in order to communicate with the display and get it to display whatever output we desire, the microcontroller is going to have to interface with the display and transfer these codes.

4.0 Sources Cited:

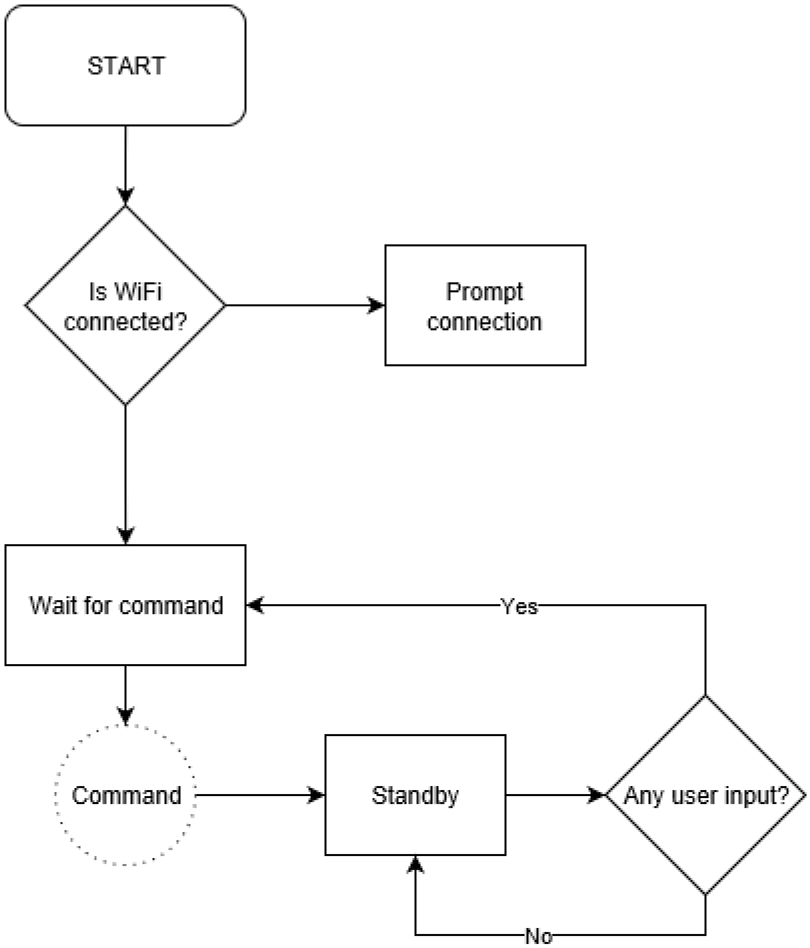
Throughout this and other papers, use of the IEEE citation style should be used. Use of embedded hyperlinks for all web-based sources is required. A reference to the IEEE citation style format is provided here.

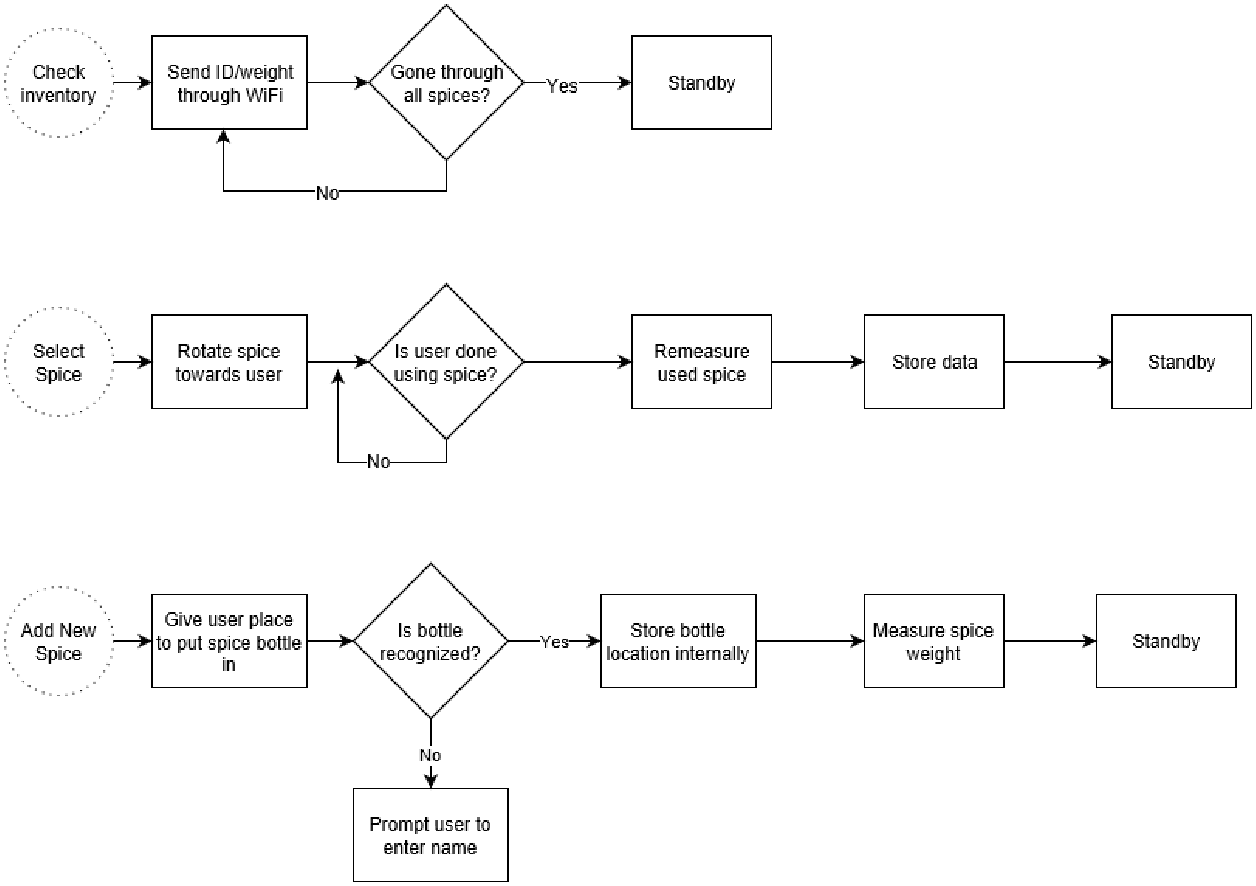
[1] M. Sarah (n.d.) *Getting Started With Load Cells.* [Online]. Available: https://learn.sparkfun.com/tutorials/getting-started-with-load-cells

[2] ESP8266 Graphical Datasheet
https://cdn.sparkfun.com/datasheets/Wireless/WiFi/ESP8266ModuleV1.pdf

[3] Newhaven Display Datasheet
http://www.newhavendisplay.com/specs/NHD-12232KZ-NSW-BBW-P.pdf

Appendix 1: Program Flowcharts


START
Is WiFi connected?
Prompt connection
Wait for command
Yes
Command
Standby
Any user input?
No


Check inventory
Send ID/weight through WiFi
Gone through all spices?
Yes
No
Standby
Select Spice
Rotate spice towards user
Is user done using spice?
No
Remeasure used spice
Store data
Standby
Add New Spice
Give user place to put spice bottle in
Is bottle recognized?
Yes
No
Store bottle location internally
Measure spice weight
Standby
Prompt user to enter name

Appendix 2: State Machine Diagrams