Software Formalization

Year: \_2021\_ Semester: \_Spring\_\_ Team: 16\_\_\_ Project:\_Smart Conveyor Belt\_\_\_\_\_\_\_\_

Creation Date: \_\_\_3/2/2021\_\_\_\_\_\_\_\_\_\_\_\_ Last Modified: March 3, 2015

Author: \_\_\_\_Jason Cao\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Email: \_cao216@purdue.edu\_\_\_\_\_\_\_\_

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Third Party Software** | 4 | x2 |  |  |
| **Description of Components** | 4 | X3 |  |  |
| **Testing Plan** | 4 | x3 |  |  |
| **Software Component Diagram** | 5 | x4 |  |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** | 5 | x2 |  |  |
| **Formatting and Citations** | 5 | x1 |  |  |
| **Figures and Graphs** | 5 | x2 |  |  |
| **Technical Writing Style** | 5 | x3 |  |  |
| **Total Score** | 92 | | |  |

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

General Comments:

*Good work. Please see my comments – this homework is for describing your software features in detail. Please see the software homework examples on the assignment webpage for more instruction* [*https://engineering.purdue.edu/ece477/Course/Assignments/Example/SoftwareFormalizationEx1.pdf*](https://engineering.purdue.edu/ece477/Course/Assignments/Example/SoftwareFormalizationEx1.pdf)

[*https://engineering.purdue.edu/ece477/Course/Assignments/Example/SoftwareFormalizationEx2.pdf*](https://engineering.purdue.edu/ece477/Course/Assignments/Example/SoftwareFormalizationEx2.pdf)

1.0 Utilization of Third Party Software

For our project, our team does not plan on using any third party software. The main algorithm is taking in information from the Waveshare barcode scanner, which uses UART communication and an interrupt on the microcontroller. The information from the barcode scanner is likely in the form of a hex value [1]. From there, we will sort by random assignment according to the barcode information. The scanner data would be transmitted in the form of a hex value. The example code from the Waveshare barcode scanner manual would be used as a reference to parse the data from the hex value. The microcontroller would interface with a custom PCB, which consists of connections to the servo motors that control the sorting mechanism, and the distance sensor to locate parcels on the conveyor belt.

2.0 Description of Software Components

The software for the Smart Conveyor Belt system consists of sorting logic, motor controller, and the STM32F0Discovery board.

The sorting logic would mainly consist of a bitwise masking the last 2 bits from the barcode hex value. From there, these bits would determine which sorting compartment by utilizing the lookup table. This table would consist of the specific sorting compartment to certain bit-values. One simple example would be where a 2-bit value is 3 would correspond to a sorting compartment 3. From there, a signal would be sent to the motor controller to change the positions of the servo motors on the sorting mechanism.

The motor controller is responsible for taking pulse-width modulation (PWM) input from the microcontroller and changing the positions of the servo motors on the sorting mechanism. The motor controller consists of a Talon SRX electronic speed controller (ESC), a logic level shifter from Icstation, and a DC 12 volt 24 volt to 5 volt 5 amp buck converter from Bankee. The buck converter converts the 12 volts coming from the power supply to 5V to power the microcontroller, servo motors, and logic level shifters. The logic level shifters would interface with both the STM32F0Discovery microcontroller as well as the servo motors. The PWM signal would come from the microcontroller and go through the logic level shifters, which would then pass on the signal at 5V, instead of the 3.3V from the microcontroller, to the servo motors.

The STM32F0Discovery microcontroller is responsible for executing the sorting logic with information from both the distance sensor and the barcode scanner. It is also responsible for storing the information about the parcels that were already scanned by the barcode sensor. Both the sorting logic and the code that controls the servo motors would be written in C. There would be no additional C libraries included in the code. The barcode sensor would send information to the microcontroller. With the new data, the data is then parsed using bit masking and is cross-referenced with a table corresponding to specific sorting compartments. Using this information, a pulse-width signal would be routed through the logic level converters, which would then move the servo motors to the desired angles.

3.0 Testing Plan

The testing would be divided up into several phases. The first phase would be to test the speed would consist of testing the maximum speed the barcode scanner is capable of scanning parcels accurately and transferring the data to the microcontroller. It is also crucial that the UART protocol receive the information as quickly as possible. This could be accomplished by outputting to a terminal screen the values that the microcontroller is receiving.

After we confirm that the barcode sensor is behaving to specification, we will move onto testing the logic that will drive the servo motors, which is the most critical component of the project. At the initial phases, we would not connect the microcontroller to any portions of the motor controller board, specifically the logic level converters. We would first measure the output of the microcontroller’s GPIO pins to see if the PWM values are being generated correctly. Once we can verify the PWM signal, the microcontroller would be connected to the logic level shifters. The output would also be tested before connecting it to the servo motors to ensure that the PWM signals are still being output correctly.

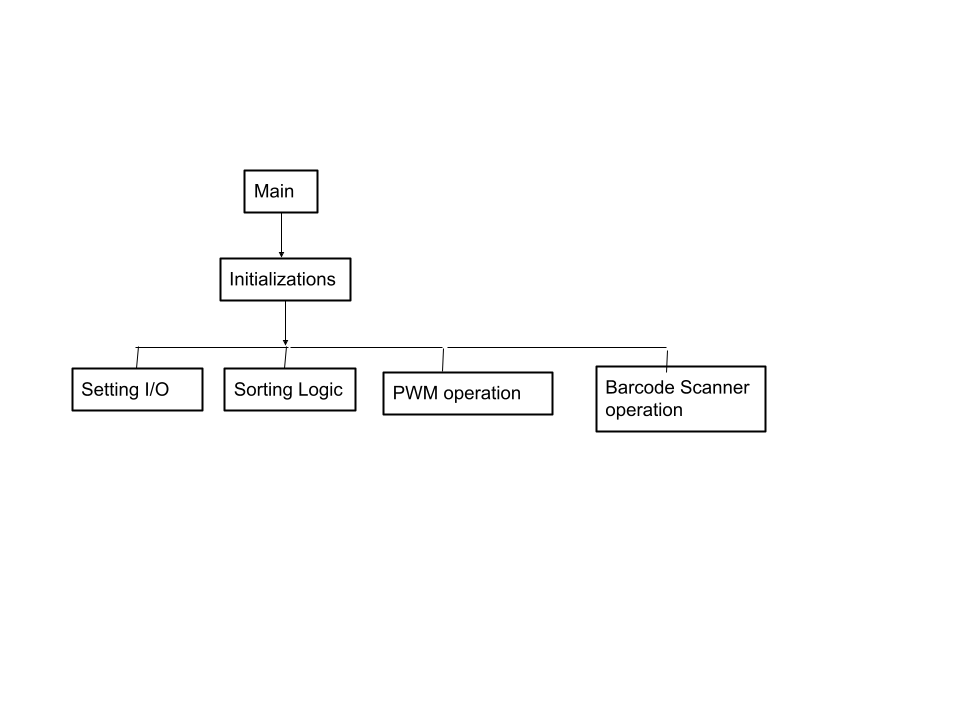
Once we have verified the operation of the servo motors, we can test the distance sensor as it controls much of the action of the servo motors. At first, we will test the distance sensor in an isolated setting (i.e. the STM32 microcontroller will not be connected to any other peripherals). We will place the sensor at set distances away from objects to measure its accuracy and to verify that the sensor is interfacing with the I2C on the microcontroller properly. This component is important to the project, but not as important as getting the fundamentals of the system operational.

From there, we would connect all external peripherals together to test the system as one unit. But, we would not use a moving conveyor for the first few tests. These first few tests would be done by moving a box through the system by hand at a reasonable speed, slightly slower than the real conveyor speed. Once we can verify that the system is able to perform to our specifications in multiple trials, we will attach all of the components and start testing the system using the conveyor belt, running at normal speeds. To ensure the quality of our project, we will put the system under several more trial runs to see it in operation and checking for accuracy and motor precision.

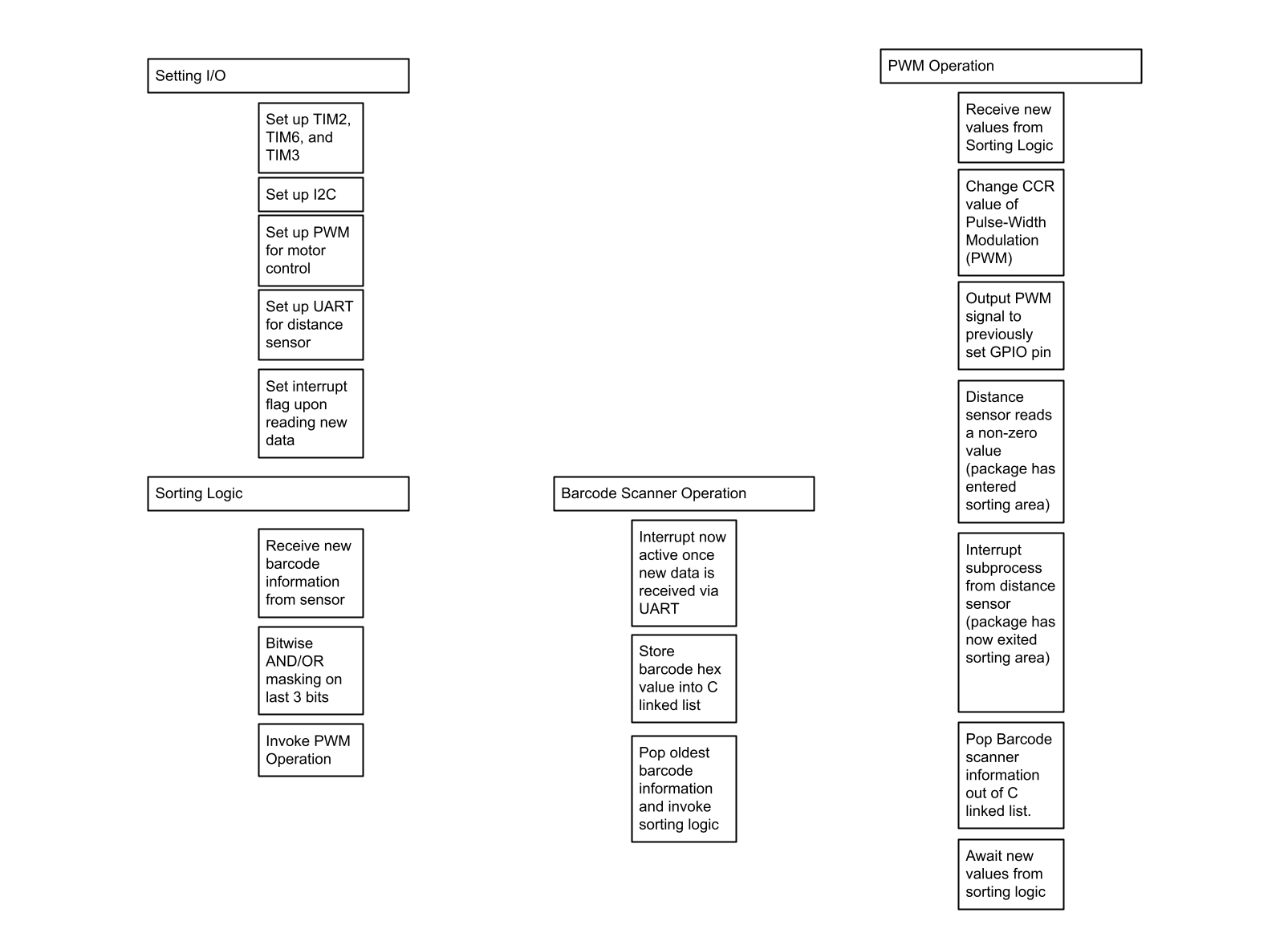
4.0 Sources Cited:

[1] Waveshare. *Barcode Scanner Module User Manual v1.2 (2019).* Accessed: Feb. 17, 2021. [Online]. Available: <https://www.waveshare.com/w/upload/d/dd/Barcode_Scanner_Module_Setting_Manual_EN.pdf>

Appendix 1: Software Component Diagram

**

*Figure 1: The main function for the project.*

**

*Figure 2: Code structure of individual functions*