**ME 615: Applications in in Mechatronics**

**Project #2**

**Instructions:**

**Complete each of the following steps. When you have completed all steps assigned, upload the files associated with the project on Canvas.   Make sure the filenames you upload as part of your exercise submission have the exact names specified in each steps of this exercise.  If the filenames do not match exactly, then your submission will not be graded.**

**Problem Statement**

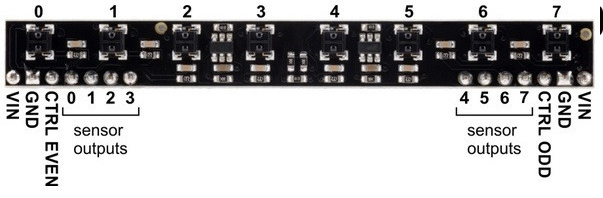
For this exercise you will using the two DRV8838 on the power distribution board to control the right and left motors of your robot. Locate the 6-pin header you soldered to the power distribution board. You will be connecting the associated DIR and PWM pins to pins on your Arduino Mega. An image of the connectors is shown below.



For simplicity I will refer to the DIR and PWM pins for the left motors as LDIR and LPWM. I will refer to the DIR and PWM for the right motor as RDIR and RPWM. As such, Connect PIN ***41*** of the Arduino to ***LDIR***, PIN ***42*** of the Arduino to ***RDIR***, PIN ***45*** to ***LPWM***, and PIN 46 to ***RPWM***.

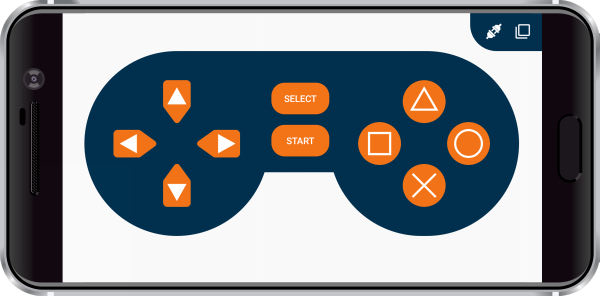
The expected operation is as follows: Applying 0V to ***LDIR*** and 5V to ***LPWM*** results in the left motor turning in forward direction at full speed. Applying 0V to ***RDIR*** and 5V to ***RPWM*** results in the right motor turning in forward direction at full speed.

Your lab kit also includes a reflectance sensor. Connect pins ***0*** through ***7*** on the sensor to pins ***A0*** though ***A7*** of the Arduino Mega. Connect the ***VIN*** pins to the ***5V*** source on the Arduino Mega, and the ***GND*** pin to the ***GND*** pin on the Arduino Mega.



You will also need to mount your color sensor to the griper of your robotic arm such that the robotic arm holds the color sensor parallel to the surface on which the robot is moving. The color sensor should positioned as close to the travelling surface as possible without touching the servo. Once mounted connect the ***SDA*** connector of the color sensor to pin ***20*** of the Arduino Mega, the ***SCL*** connector of the color sensor to pin ***21*** of the Arduino Mega, the ***VIN*** connector of the color sensor to a ***5V*** source on the Arduino Mega, and the GND connector of the color sensor to a GND source on the Arduino Mega.

You will be using the Dabble gamepad interface to instruct the robot to start navigating a line follower maze and to implement an emergency stop if the robot stays outside the bounds of the maze. A basic image of the Dabble gamepad interface is shown below:



You must implement code to use the gamepad interface to control the motion of your robot and the robotic arm. Your application must meet **the requirements listed below:**

**Requirements**

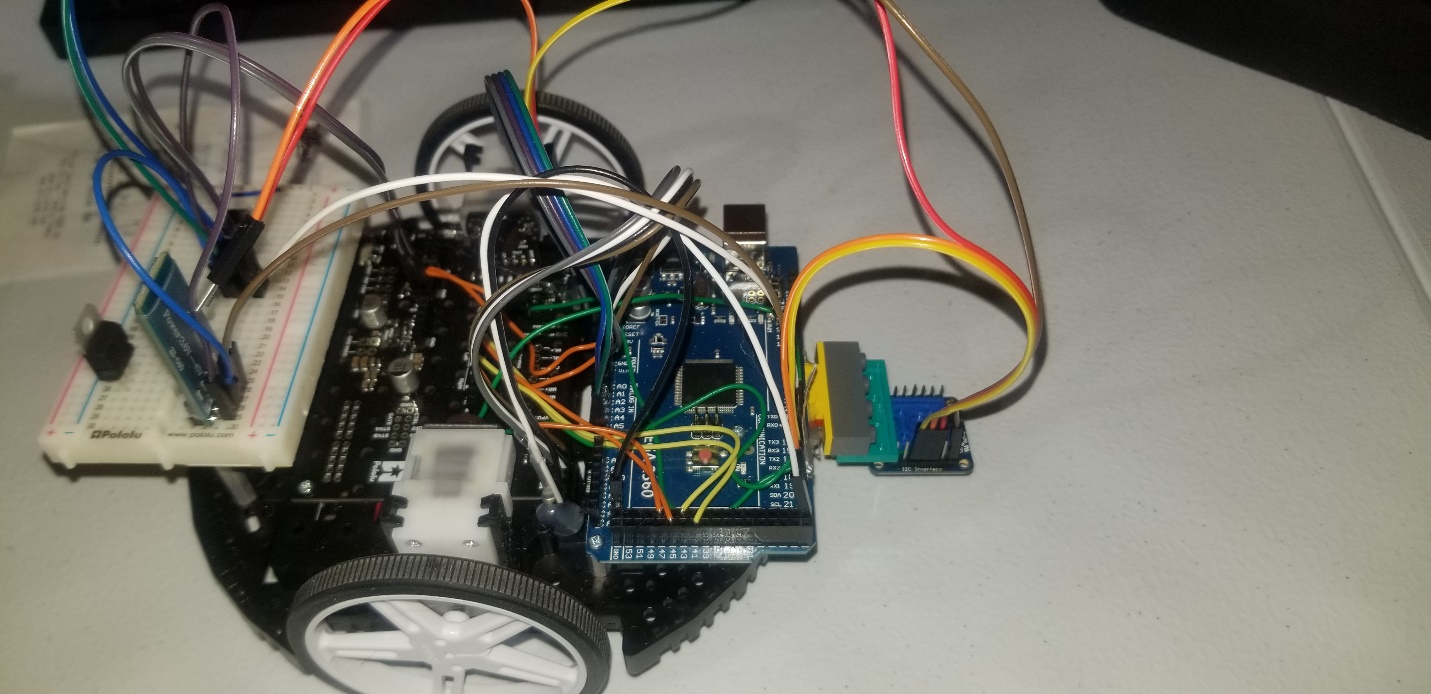
1. **Create a sub directory named “project2a”, and an associated Arduino sketch file named “project2a.ino”, in your Arduino directory.**
2. **Create a program that includes a *setup()* and *loop()* routine. Set the baud rate to 9600 bits/second for communication between the Dabble Bluetooth interface and the Bluetooth module.**
3. **Within the *loop()* routine implement the following:**
   1. **If the select button is pressed within the Dabble interface start a routine that calibrates the QTRX sensor array. This calibration should take between 10 and 20 seconds and you should move the sensor array of the robot over a strip of black tape on a white background during the calibration cycle. To help you identify the start and end of the calibration cycle, illuminate the built-in LED attached to pin 13 of the Arduino Mega during the calibration. Turn off the LED when the calibration when the calibration completes.**
   2. **If the start button is pressed within the Dabble interface set a global variable to true that indicates the robot should start navigating the maze.**
   3. **If the cross button is pressed within the Dabble interface, then the robot should immediately stop and reset the global variable to false that indicates the robot should start navigating the maze.**
   4. **If the global variable is set to true that indicates the robot should start navigating the maze, then start following the black line. Your robot should continue to follow the black line based on the following criteria:** 
      1. **Upon pressing start the robot should start navigating the maze.**
      2. **The robotic arm should be positioned such that the color sensor can detect if a red or green area exists in the line trace.**
      3. **If a red area in the line trace exists, then the robot should stop, pause for 5 seconds, and then resume following the line trace.**
      4. **If a green area is the line trace exists, then the robot should stop, reverse directions, and resume following the line trace in the opposite direction.**
      5. **A red or green segment of the line should only be evaluated once each time it is encountered. In other words, say a 2 inch red section is found in the line trace. The robot should stop, then resume without processing another stop condition until the red section is cleared. However, if a green section is encountered later, the robot will reverse course. On the return trip over the same red segment it should stop again and pause for 5 seconds.**
      6. **The line course will be constructed using black electrical tape. The red and green segments in the line will always be included in a straightaway section of the maze. Special code may need to be put in place such that the robot will traverse the colored portions of the line without losing track of the line. This problem may occur because the reflectance sensors are calibrated to track a black line, not a red or green line.**
      7. **The robot should stop line navigation if the cross button is pressed on the Dabble interface, and restart navigation if the start button is depressed.**
      8. **The code provided in lecture may not be optimal for line navigation. You are expected to consider alterations to the finite state machine to make it track more reliably and efficiently. Submissions will be timed with a stopwatch and ranked from fastest to slowest.**
4. In your project notebook provide pertinent spec sheets and detailed pictures that show your connections to the Arduino, connections to the motors, connections to the Bluetooth modules, connections to the color sensor, connections to the reflectance array, and connections to the Romi chassis. Clearly label connection points in your pictures. Supporting diagrams are also suggested.
5. You must demonstrate your code to your instructor in-person. Time slots for the demonstration will be made available after the due date for the project. You will be expected to demonstrate that the robot can navigate through a simple line maze that includes straightaways, turns, stop connections, reverse conditions, and dead-ends. The speed at which your robot completes the maze will be considered in your evaluation as well as its ability to deal with different maze conditions. Note: You will also be expected to turn in your project notebook after the demonstration for review by your instructor.

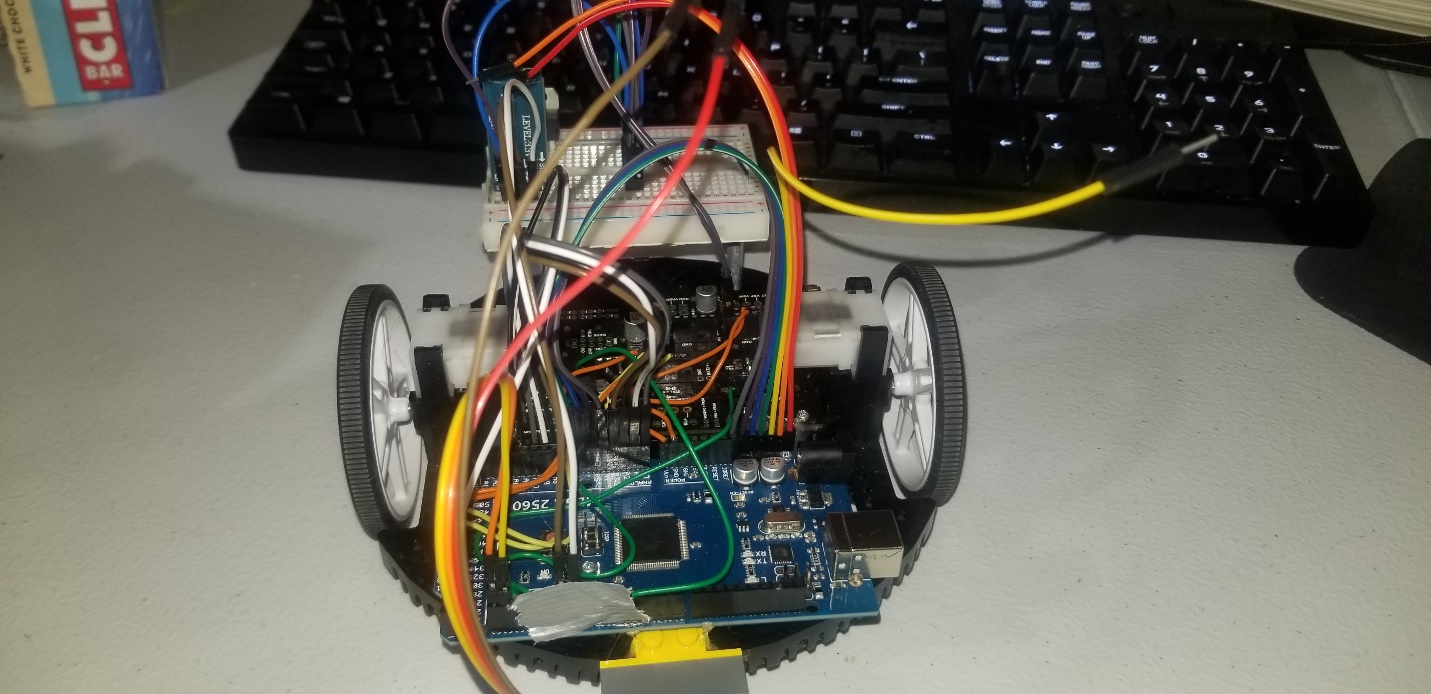
**Submission**

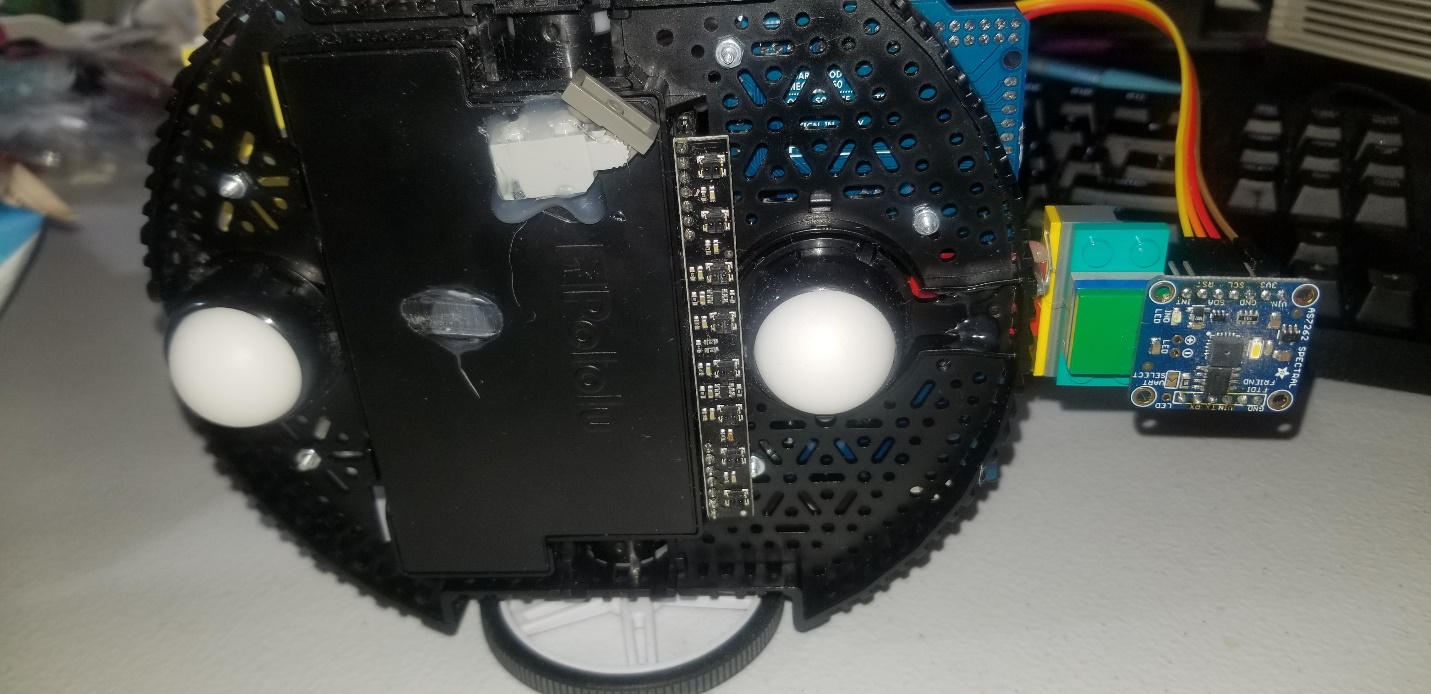
Upload the file project2a.ino to canvas once the exercise has been completed. The code will be evaluated as follows:

1. **It will be evaluated to make sure the code is implemented using the methods outlined in this exercise.**
2. **Your program will be evaluated the ensure that the program successfully compiles.**
3. **Your program will be evaluated to ensure that it includes proper comments that explain the functionality of the source code.**
4. **Your program will be evaluated to ensure that it runs correctly, collects the input as described, and outputs information as described.**

You are also expected to include proper documentation for this project in your project notebook.

****

****

****