CSCE 336 Embedded Systems, Spring 2025
Project 1

Bitbucket Repos: Tuesday, Mar 11th (end of class)
Checkpoint 1 – "Robot Assembly/Schematic" Thursday, Mar 13th (close of business)
Checkpoint 2 – "Robot Motion", Tuesday, Mar 25th (end of class)
B Functionality – "Wall Following", Thursday, Mar 27th (in Lab)
Competition, Thursday, Mar 27th (in Lab)
Code and Written Report: Tuesday, Apr 1st (beginning of class)

Name: _____

1. Instructions

This Project is an individual assignment, collaboration is not allowed. Show your work and describe your reasoning to get partial credit if your solution is incorrect. Complete all of the sections below and make sure to get the instructor or TA to sign off on the cutsheet where required. You should keep notes on what you complete since parts of future homework will build on this project.

All code will be uploaded in your code project folder on Bitbucket! Failing to electronically turn in your code will result in up to at least a 20 point penalty on this assignment. Points may also be deducted for coding errors, poor style, or poor commenting.

2. Late Work

If problems arise with graded assignments, see your instructor in advance. Assignments portions turned in or checked off later than the due date without prior permission from the instructor will be penalized 25% per day on that portion. Plan ahead to get things completed on time.

3. "Code Repository on BitBucket"

Go to www.Bitbucket.org and create a code repository named "YourLastName_CSCE_336". Make sure you make it **private** and share with "jfalkinburg2@unl.edu". Bitbucket is an online code sharing system that supports git.

You should be committing your code early and often. Use meaningful commit messages as well. Use the commit messages to describe what changes you are committing at a given time.

4. Robot Assembly

Complete the assembly of your Robot kit. You should have already mounted your power switch, motors, Arduino Uno, servo, ultrasonic sensor, battery pack, and L298N Motor Driver onto the robot chassis. You should only have to wire up the motors and L298N to the Arduino.

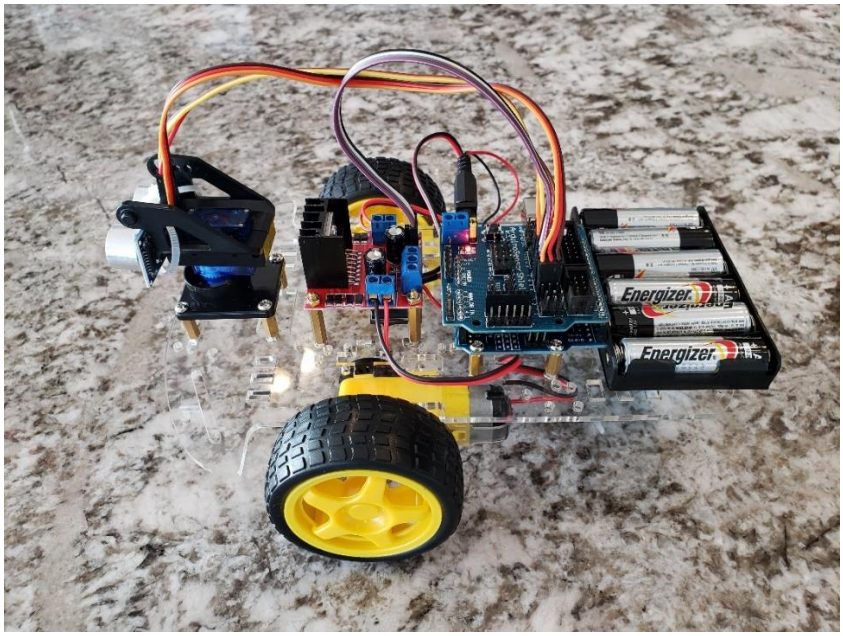

Figure 1: Assembled Robot

Checkpoint 1 - Checkoff: Create a schematic/wiring diagram (using a program like Fritzing) to s*how the instructor how you decided to wire up your robot motors to your Arduino and the L298N (Note: I uploaded the L298N part to Canvas). Attach a copy of the schematic to your report and upload a picture in your Bitbucket repo.* ***Use the Canvas quiz to upload a picture of your wired robot, the schematic diagram, and answers to the following questions.***

Discuss how will you be wiring up your motors? Do you plan on using a two or three wire interface to your motors?

What will you be doing with the motor enable pins?

What pins/timer will you be using to create your PWM signals and how will you change directions on the motors?

The Instructor will be checking Canvas to sign off on this checkpoint!

5. "Robot Motion"

Overview:

This lab is designed to provide you with experience using the pulse-width modulation features of the Arduino Uno. You will use one of the 8-bit timers to send create two PWM signals to drive the left and right motors on your robot through your L2989N H-Bridge without using the Arduino servo libraries. You will need to manually configure your Timers to generate two output compare PWM signals to control the speed of your motors. Additionally you will use one or two digital outputs to control the direction of your robot's motors. In this lab, you will make your robot move forward, backwards, a small (< 45 degree) turn left & right, a large (> 45 degree) turn left &right.

Driving Motors:

Our mobile robots have DC motors to drive the wheels. The amount of torque provided by the motor is directly proportional to the amount of voltage provided. Therefore, there are two ways of varying the speed of the DC motors:

1. Provide an analog voltage where the magnitude is proportional to the output torque desired.
2. Provide a PWM signal where the duty cycle provides an "average" voltage proportional to the output torque desired. This is shown in Figure 1.

Figure 2: The PWM signal creates a certain duty cycle which will provide an "average" voltage to the motor. This average voltage is proportional to the motor's output torque.

The motor can move in two directions. If you ground one terminal of the motor and connect the PWM signal to the other side of the terminal, then the motor shaft moves in one direction. If you swap the terminals, the motor will move in the opposite direction.

You might want to program your robot so it turns using a tank-style turn (i.e. one motor forward and the opposite motor backward to turn on a dime) versus a pivot-style turn (i.e. stop the pivot motor and drive other motor forward/backward to make a wider turn). You will have to experiment with your robot to find out how long the PWM signal needs to be provided to turn an appropriate amount.

Required Functionality "Robot Motion" - Checkpoint 2

Demonstrate movement forward, backward, a small (< 45 degree) turn left and right, and a large (> 45 degree) turn left and right. The robot should perform these movements sequentially, completely disconnected from a computer (i.e. no USB cord).

Checkoff: *Demonstrate movement forward, backward, a small (< 45 degree) turn left and right, and a large (> 45 degree) turn left and right. The robot should perform these movements sequentially, completely disconnected from a computer (i.e. no USB cord)?*

Post a link to a video using the Canvas Quiz for checkoff.

B Functionality "Wall Following"

Using your ultrasonic sensor and servo demonstrate that your robot can follow a wall (i.e. staying within a foot of the wall without touching it). You robot must follow a wall for at least 20 feet using the ultrasonic sensor to adjust the robot distance from the wall.

For the competition we will follow the wall for speed and for Project Part 2 we will be extending our wall following algorithm to follow a wall and avoid and obstacle placed in its path (e.g. a box placed against the wall).

Checkoff: *Demonstrate that your robot can follow a wall for 20 feet (i.e. staying within a foot of the wall without touching it).*

Post a link to a video using the Canvas Quiz for checkoff.

A Functionality

Create standalone library files that includes a header and implementation file for your robot motor code and upload them to your Bitbucket repo. You can call them motors.h (header) and motors.c/motors.cpp (implementation). It should have a thoughtful interface and be capable of being reused in the robot maze project. You also need to create a README.md or a help file to explain the functions and how to use the library.

Competition Bonus Scoring

For the Competition Bonus the following scoring rules will apply:

- Must compete in competition on the day of the event.
- You get two attempts to compete in the event.
- You must use the ultrasonic sensor and servo to navigate around the obstacle.
- There are 3 marked points on each course (revealed on the day of the competition). Each point reached will result in 1 point and finishing will result in an additional point. This means you get 4 points for finishing the course.
    - You will be docked points for contact with the wall and the obstacle.
- The fastest time will get 6 additional points, second 5 points, and third 4 points. Those finishing within the top third of the class (excluding top three) will get 2 points, second third will get 1

additional point.

6. Tutorial - L298N Dual Motor Controller Module[1]

In this tutorial we'll explain how to use our L298N H-bridge Dual Motor Controller Module 2A with Arduino. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.

There is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.

So let's get started!

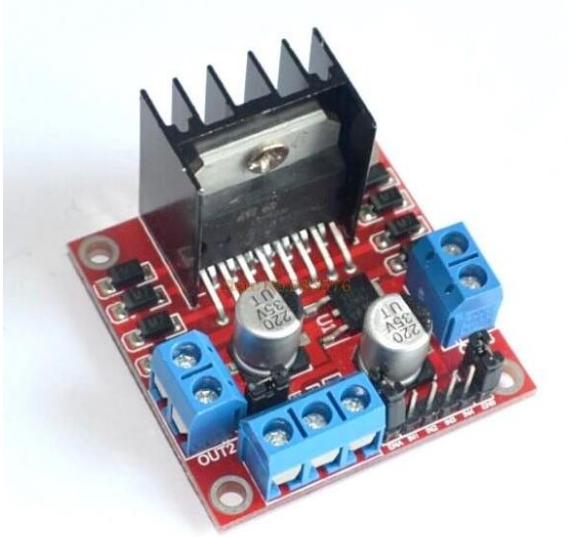

Figure 2: L298N Dual Motor Controller Module

First we'll run through the connections, then explain how to control DC motors.

[1] Abridged and adapted version of tutorial posted by Tronixlabs Australia on https://tronixlabs.com.au/news/tutorial-l298n-dual-motor-controller-module-2a-and-arduino/

Module pinouts

Consider the following image - match the numbers against the list below the image:

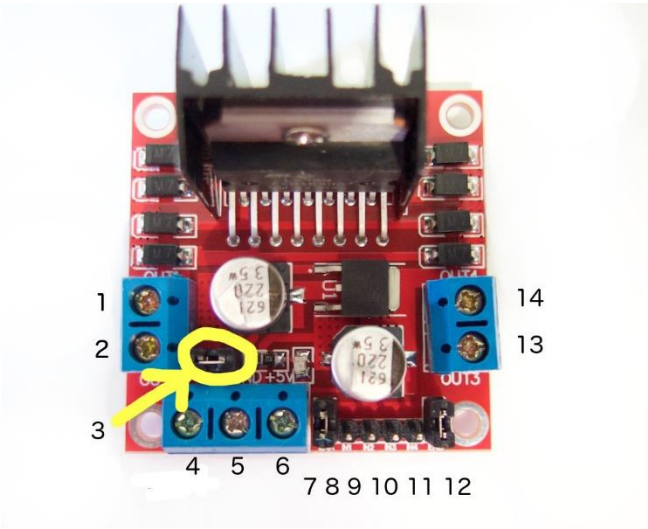

Figure 3: L298N Dual Motor Controller Module Pinout

1. DC motor 1 "+"
2. DC motor 1 "-"
3. 12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Connect to PWM output for DC motor speed control.
13. DC motor 2 "+"
14. DC motor 2 "-"

Controlling DC Motors

To control one or two DC motors is quite easy. First connect each motor to the A and B connections on the L298N module. If you're using two motors for a robot (etc) ensure that the polarity of the motors is the same on both inputs. Otherwise you may need to swap them over when you set both motors to forward and one goes backwards!

Next, connect your power supply - the positive to pin 4 on the module and negative/GND to pin 5. If you supply is up to 12V you can leave in the 12V jumper (point 3 in the image above) and 5V will be available from pin 6 on the module. This can be fed to your Arduino's 5V pin to power it from the motors' power supply. Don't forget to connect Arduino GND to pin 5 on the module as well to complete the circuit.

Now you will need six digital output pins on your Arduino, two of which need to be PWM (pulse-width modulation) pins. PWM pins are denoted by the tilde ("~") next to the pin number, for example:

Figure 4: Digital Pins on Arduino Uno

Finally, connect the Arduino digital output pins to the driver module. In our example we have two DC motors, so digital pins D9, D8, D7 and D6 will be connected to pins IN1, IN2, IN3 and IN4 respectively. Then connect D11 to module pin 7 (remove the jumper first) and D3 to module pin 12 (again, remove the jumper) to utilize OC2A and OC2B respectively.

Timer output	Arduino output	Chip pin	Pin name
OC0A	6	12	PD6
OC0B	5	11	PD5
OC1A	9	15	PB1
OC1B	10	16	PB2
OC2A	11	17	PB3
OC2B	3	5	PD3

Figure 5: PWM Output Pins Summary on Arduino Uno

The motor direction is controlled by sending a HIGH or LOW signal to the drive for each motor (or channel). For example for motor one, a HIGH to IN1 and a LOW to IN2 will cause it to turn in one direction, and a LOW and HIGH will cause it to turn in the other direction.

However the motors will not turn until a HIGH is set to the enable pin (7 for motor one, 12 for motor two). And they can be turned off with a LOW to the same pin(s). However if you need to control the speed of the motors, the PWM signal from the digital pin connected to the enable pin can take care of it.

7. "The Report"

Title Page:

This should at a minimum include the following information?

CSCE 336

Embedded Systems

Robot Design Project 1 Report

Name of Project/Robot

Student's Name

School of Computing

University of Nebraska – Lincoln

Due Date: 3/31/20

Objectives or Purpose:

This should include a paragraph on what is the goal of this assignment?

Remember, the purpose of the lab notebook is to communicate EVERYTHING you have done in pursuit of a particular project. If you are sitting on the bus scribbling on the back of an envelope about your lab design, take a picture of that and include it with your work. Without adequate documentation, your instructor won't know what your thought process was and will not be able to grade you properly. Once you leave the school environment, fellow engineers may need to pick up your project where you left off if you move jobs, get hospitalized, or otherwise find yourself no longer working on something. You will save your employer/co-workers/replacement a lot of time and money if you have left a detailed record for them to easily understand what you were doing, the approach you took, the tests you performed, and what you learned.

Preliminary design:

How will you start attacking the problem? This should include detailed instructions of what you are about to do. It may include PreLab material and also information from the Lab Handout. Use pictures and data from Lab Handout. You may also use snippets of code in here as well:

You should only include important key code snippets in your repot. All code files should be included in code folder on Bitbucket.

Code:

You should only include important key code snippets in your report with discussions of how you implemented functionalities. All code files should be included in code folder on Bitbucket.

Well-formatted code

All of your code should be written with:

1. headers
2. comments
3. good coding practices.

Software flow chart or algorithms:

All coding include a pseudocode flow charts and algorithms defined your code and the algorithms used. Visio, PowerPoint, or Google Drawings works well for this!

Insert pseudocode or flowchart here. No hand drawn drawings will be accepted.

Hardware schematic:

If you are wiring things up you will need to create a schematic for your design. Fritzing (http://fritzing.org/download/) is a nice tool with all the appropriate parts used in this project. You can use whatever tool you want just as long as it shows a neat repeatable design.

Debugging:

You should be keeping track of issues as you go along. I didn't have any problems is not a good answer. Describe the problems you had and what you did to fix it. Again this is where I would say commit early and often and start your notebook when you start your code.

Testing Methodology or Results:

Detail the steps in getting the results you system is designed to achieve. Have enough detail that someone can come behind and reproduce your results.

Display your results and describe them in detail so that anyone can understand. For example Figure 1 below shows a screenshot of a memory dump for RAM from 0x0200 to 0x024E. You will also describe to the reader what they are looking at.

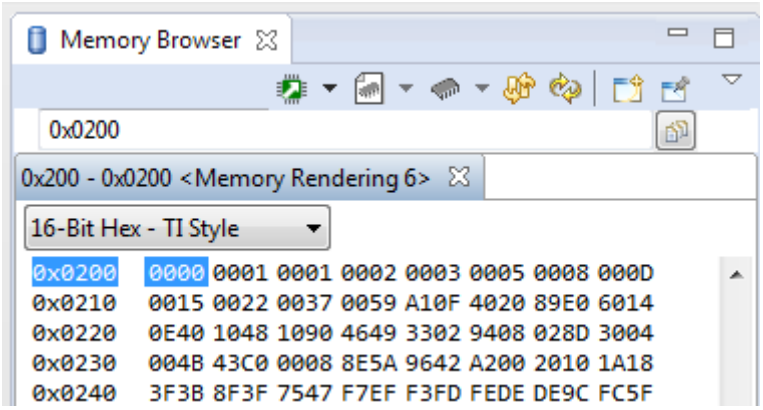

Figure 1: Memory Dump Label (Always include figure AND table labels!)

Answers to Lab Questions:

Here is where you would answer any lab questions given in the lab writeup.

Observations and Conclusions:

During this whole assignment, what did you learn? What did you notice that was noteworthy? This should be a paragraph starting with the purpose, whether or not you achieved that purpose, what you learned, and how you can use this for future labs.

Documentation:
– You always include this! Document any help received on any portion of the assignment, even from an instructor, TA, or the internet should be specifically mentioned.