Embedded Class Project

CSE 3442 – 001 Embedded Systems I

Happy Ndikumana

ID: 1001641586

Instructor: Jason Losh

Lab Peer: Sarker Nadir Afridi Azmi

Course: CSE 34402 – 001

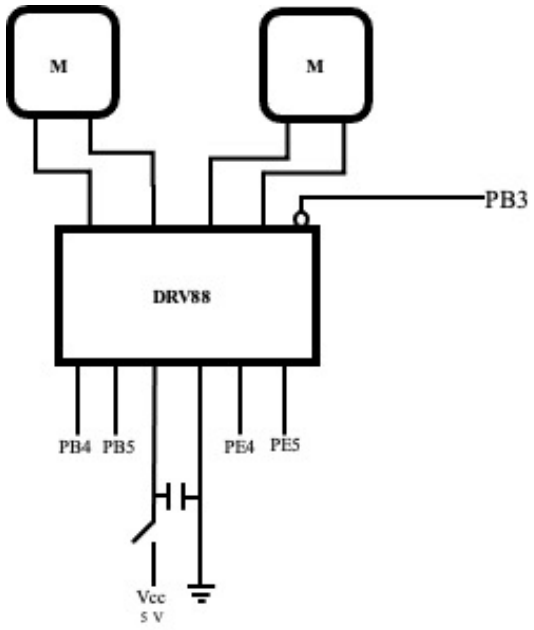
Date: 05/04/2021

Introduction

This semester we worked on a robot that is capable of being programmed from a USB port. Using sensors, motors, and microcontrollers, we were able to make the robot move in any manner. The hardware provided data and that data was manipulated in code to give us the outcome, movement, we wanted from the robot.

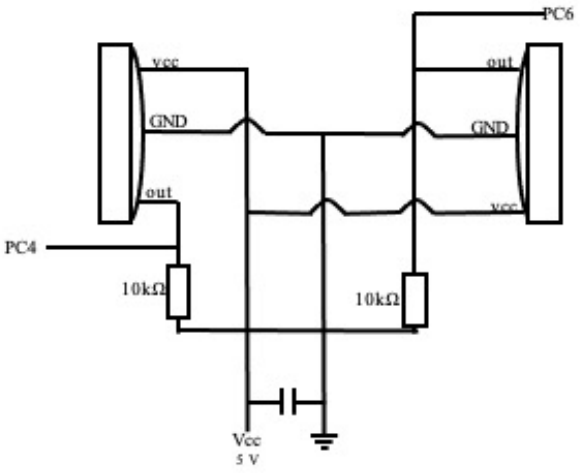
Theory of Operation

Circuit 1:



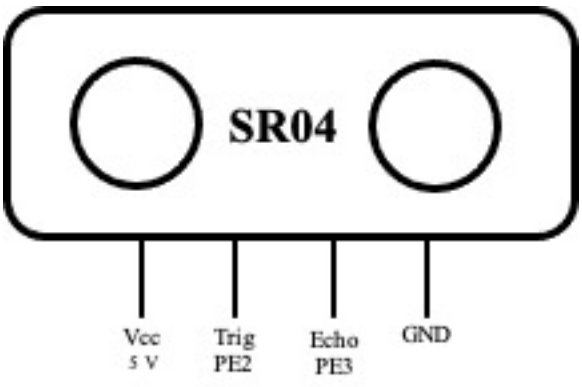
The input ports (PB4, PB5, PE4, PE5) are driven by PWMs through GPOs.

Circuit 2:



The hall effect sensors are driven by timers on PC4 and PC6 GPOs.

Circuit 3:



The SR04 is driven by GPIOs PE2 and PE3.

The motors have gears on their backs with magnets attached to them. After each rotation, the hall effect sensors detected the magnet and incremented the count

Circuit 2 provides counts for each wheel. For the motors we were working with, each motor had a different count per rotation. To calibrate the motors and to make sure the wheels would turn at the same speed and travel the same distance, I recorded the counts of each wheel per rotation and kept the difference as a constant. I then drove the motors using circuit one at a duty cycle of 90% on both wheels. If at any point the difference between the counts of wheels did not equal the constant recorded before, I made the wheel ahead wait for the other one. Using this operation, the robot was able to travel in a straight line forward and backward.

To implement the rotation functions of the robot, I took the width of the robot and divided it into two. I used that width to calculate the circumference of the circle it would form if it rotated on itself. I then took that circumference and divided it by 360 degrees. I now had the distance to travel per degree. Using this distance and counts from circuit two, I was able to make the robot rotate the appropriate degrees.

Circuit 3 made use the robot avoided collision ahead of it. Trigger was put on high though PE2 for a minimum of 10 microseconds and I waited for echo to go on high. PE3 goes on high whenever it detects the signal back. Using the speed of sound, 340 m/s, I calculated the distance between the robot and the obstacle and made the robot stop accordingly. I did this by setting the duty cycle of the wheels to 0%.

Distance Calculation code

```
double falseDistance = (double)distance_cm;
double counts = (distance(falseDistance)/ 20.344) * countsPerRotation
int idealCounts = (int)counts;
int count0 = 0
```

```
   int count1 = 0
  while(count0 < idealCounts - 1 && count1 < idealCounts - 1)
// move until counts are more than ideal counts for that distance
```

Degree Calculation code

```
   double realDegrees = (double)degrees * (1/2.2221);
   double degreeDistanceConst = distance(0.163188); // cm/degree

   double distance = realDegrees * degreeDistanceConst;
   double counts = (distance / 20.344) * countsPerRotation;
   int idealCounts = (int)counts;
   int count0 = 0;
   int count1 = 0;

   while(count0 < idealCounts - 1 && count1 < idealCounts - 1)
   // move until counts are more than ideal counts for that distance
```

Conclusion

This was the first hardware project I have ever worked on. Starting the project and understanding how to program tiva microcontroller was hard. This project served to broaden my knowledge and experience working with microcontrollers. At the end of the project, programming the microcontroller was easier.

I had a hard time with my hardware. Hardware is very hard to calibrate. I worked with motors of the same make, but each had a different speed. I calibrated them using my hall effect sensor and code, but they soon stopped working. I had to do this with approximately 5 different motors.

Getting precise results out of our hardware was also a challenge. The distance to travel and the degrees to rotate were hard to calculate. In theory, the math made sense, but it did not account for the stopping distance or deceleration. Also, given a very small distance to travel or a small amount of degree to rotate, the motors were likely not to start operating. I think with better hardware, my robot would have been more accurate.