RSA Lab #8: Finite State Machines with Magnetometer & Accelerometer

# Part A: GPS Receiver…or, A Space Oddity

For Part A, I was able to construct a program using the GPS system to display the distance and angle from a starting position.

1. Draw a wiring diagram to hook up the GPS receiver to the Arduino. Submit a clearly drawn sketch. Label the appropriate pins on the Arduino and the GPS receiver.
2. Make a program that will calculate a position and display it in terms of distance (r) and angle (theta) from the starting spot, once per second. Submit your commented code.

See GitHub: <https://github.com/ykokeb1/rsa_stuff/tree/main/Arduino/lab8a>

1. Check your program by walking outside on a path that goes a hundred meters or so, coming back to your starting location. Submit a screen shot of the Serial Plotter window as you walk around a closed path.

Graphical user interface, chart, line chart

Description automatically generated

1. Use your oscilloscope to measure the TX output signal from the GPS receiver. Submit a screen shot of the signal on the oscilloscope, showing a data packet. Estimate the period of a bit from the plot, and compare that to the asynchronous serial data baud rate.

See attached scope. Each bit was measured to be approximately 100us in duration, which corresponds with the baud rate of 9600 bps, or approximately 1bit per 104us.

# Part B: Interrupting Cow

For this part of the lab, I was able to construct a simple program demonstrating the functionality of interrupts between two buttons, where a low signal in either of the buttons switches the state of the LED. I also implemented switch bouncing by adding a delay between the state changes of the LED.

1. Draw a circuit that will use two momentary switches to control an external LED with an Arduino. Submit your diagram. (Handwritten diagrams are OK.)
2. Write an Arduino sketch that will use interrupts to monitor the two momentary switches to toggle the external LED on and off. Submit your program. (Handwritten programs are OK.)

See GitHub: <https://github.com/ykokeb1/rsa_stuff/tree/main/Arduino/lab8b>

1. How does bouncing affect the handling of the interrupts? Explain briefly.

The bouncing effect eliminates the brief period of noise on the LED before it gets to a stable HIGH state. By providing a delay after the switch is pressed, the Arduino can read the signal at its stable state rather than all the time (where the noise is present).

# Part C: DC Motor with Encoder

Finally, in this section I was able to use interrupts to record the count of a DC motor by using an encoder and detecting when there is a change in the square-wave signal from the two channels.

1. Write an Arduino program that will decode the encoder pulses and keep track of the motor’s rotation using a counter variable. Submit a copy of your program.

See GitHub: <https://github.com/ykokeb1/rsa_stuff/tree/main/Arduino/lab8c>