Introduction to Actuators

Introduction

In this lab, you'll explore electromagnetism and some common actuators. You'll build a simple electromagnet and see how electromagnetism can be used both for sensing and actuation. You will then implement some basic mechanical actuators, namely a solenoid and an RC servo.

Objectives

Upon successful completion of this lab, the student will be able to:

- Describe how electromagnetism can be used to create mechanical motion,
- Describe the operation of solenoids and RC servo motors,
- Describe the operation and purpose of mechanical relays,
- Implement driver circuits for solenoids, relays, and RC servo motors,
- Describe how Hall effect sensors can be used to detect magnetic fields,
- Implement a Hall effect sensor for detecting the presence of a magnet, and
- Implement a hysteresis band for control.

There will be a quiz this coming week.

Preparation

Background materials

Review the following materials:

- **Electromagnetism and solenoids.** Read the few pages posted on collab.
- The SparkFun tutorial on RC servos is pretty good.
- Refresh your memory of hysteresis from the reading on event-driven programming.

In lab

Electromagnetism

An electromagnet can be created by winding wire into a coil and passing current through it. By wrapping the wire around a ferromagnetic material (such as an ordinary iron nail), the magnetic field becomes more concentrated and amplified. When no external magnetic field is applied to a ferromagnetic material, the magnetic domains within the material are randomly oriented and cancel themselves out so that the material produces no net magnetic field. When an external magnetic field is applied, however, the magnetic domains within the material tend to line up with the external field. These internal magnetic domains then add to the external field and create a more powerful electromagnet.

In this part of the lab, you will explore a homemade electromagnet made by wrapping a nail with insulated wire. You'll use this device not only to create a magnetic field (by passing current through the windings) but also to sense an external magnetic field passing near it.

Procedure

1. (This step has probably been done for you – ask before you go and wind your own!) Construct a homemade electromagnet by wrapping wire around a nail or screw a few hundred times – we have a drill that is good for speeding things up. Make sure to leave a few inches of wire on each end for connecting the electromagnet to jumper wires.
2. (This step has probably been done for you!) Remove an inch or so of the insulating enamel from the ends of the magnet wire by sanding them with fine grit sandpaper or scraping them with a knife until the copper beneath the enamel is fully exposed. You can also use a lighter to burn the enamel off.
3. Using alligator clips, connect your electromagnet to a bench-top power supply with the voltage set to 0V for now. The power supplies are convenient because they are easily adjustable and also show the amount of current being drawn, but **be careful not to run too much current through the coil for too long – it will get hot and burn you!**
4. Slowly turn up the voltage supply and verify that your electromagnet works by picking up nails, screws, or other ferromagnetic objects. Observe the amount of current flowing through the circuit versus the apparent strength of the electromagnet as you adjust the voltage on the power supply. How are they related?

Note that since you are picking up ferromagnetic pieces of metal, it does not matter which direction current is flowing through your electromagnet's coil – a north pole attracts a piece of iron just as well as a south pole does. If you were trying to attract (or repel) a permanent magnet with an electromagnet (which is how a typical DC motor works) the direction of current flowing through an electromagnet's coil becomes very important because it determines the polarity of the electromagnet.

Sensing using EMF

The simple device made by wrapping coils of wire around a ferromagnetic core can serve another useful purpose. Now that you have created an electromagnet by sending electrical current through your coil, you will do the exact opposite and pass a changing magnetic field near the coil to generate electricity within the coil. When an external magnetic field passes near a coil of wire, an electromotive force (EMF) induces a voltage within the coil. In this experiment, you'll only generate enough electricity to make a 'blip' on the oscilloscope, but this is the basic principle behind electric generators.

Procedure

1. Disconnect your electromagnet from the power supply and connect it directly to CH1 on your oscilloscope. You'll want to do this with breadboard and banana clips – sticking the tiny wires in the oscilloscope connectors is not going to end well.
2. Take a magnet and pass it near the end of the electromagnet such that the N-S axis of the magnet is parallel to the nail. Observe what happens on the oscilloscope. (You should see a small voltage spike on the oscilloscope each time this happens.)
3. Experiment with passing the magnet at different speeds and distances from the pickup. Why do these affect the magnitude of the voltage spike? What could you do to amplify these signals?
4. Turn the magnet around and repeat the experiment. How is the oscilloscope capture different? Why?

Show your working system to an instructor.

Instructor initials: ______________________

Hall effect sensor

A typical Hall effect sensor contains a small conductive plate (called a Hall plate) with a steady electrical current flowing across it. When an external magnetic field is applied perpendicularly to the plate, a Lorentz force deflects the electrons flowing through the plate. This causes more negative charges to be on one side of the plate, which creates a voltage potential across the plate. As the strength of the external magnetic field increases, the voltage potential across the plate increases proportionally. The Melexis 90127 Hall effect sensor that you will use here incorporates some extra circuitry that amplifies the signal to output either a digital `HIGH` or `LOW`. Fundamentally, what component do you think it uses to do this?

Procedure

1. Place your Hall effect sensor in a breadboard and build the circuit shown in the datasheet. Do not connect your Arduino to the breadboard at this time – use the 5V power supply from

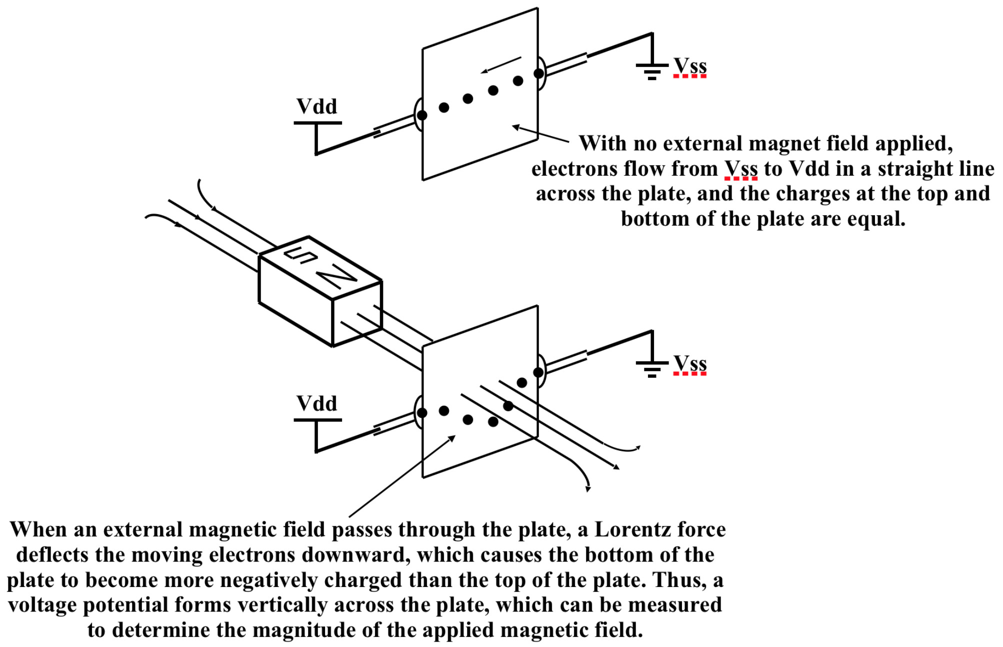
Vdd
Vss
With no external magnet field applied, electrons flow from Vss to Vdd in a straight line across the plate, and the charges at the top and bottom of the plate are equal.
S
N
Vdd
Vss
When an external magnetic field passes through the plate, a Lorentz force deflects the moving electrons downward, which causes the bottom of the plate to become more negatively charged than the top of the plate. Thus, a voltage potential forms vertically across the plate, which can be measured to determine the magnitude of the applied magnetic field.

Figure 1: Physics of the Hall effect sensor. Stolen shamelessly from Gavin Garner's Mechatronics manual.

your oscilloscope.

2. Place the magnet near the Hall effect sensor and observe Channel 1 on the oscilloscope. Try turning the magnet around and see if the effect is the same.
3. Now turn off the 5V supply and **disconnect the oscilloscope** from the circuit.
4. Connect the Hall effect sensor to your Arduino as you would a button. In fact, you should be able to run your button code from a previous lab and show that you can count the number of times the magnet passes the Hall effect sensor.

Show your working system to an instructor.

Instructor initials: ____________________________

Solenoids

Solenoids can be used to push or pull things a short distance, and they are commonly applied in electromechanical latch mechanisms. They consist of a ferromagnetic rod that travels through a coil of wire. When the coil is energized, magnetic fields are set up that draw the rod to the center of the coil. Since a solenoid can only move in one direction, springs are often used to return it to its initial position.

Procedure

1. (This step may be done for you already – ask before you do this.) Take a piece of cardstock about 8cm x 5cm and roll it into a tube 8cm long and 2 or 3 layers thick. You want it to be stiff enough to allow you to wind wire around it without it collapsing and large enough to allow a nail to slide in and out of it easily.
2. (This step may be done for you already – ask before you do this.) Wind a couple hundred turns of electromagnet wire around the tube, leaving a few inches of wire sticking out at each end. Strip the ends like you did before.
3. Place a nail a short distance into the tube and connect the wire ends to a bench top power supply. If the current is high enough (be careful not to overdo it again), the nail will be pulled into the tube. Congratulations – you've just made a solenoid!
4. Examine the datasheet for the solenoid, found on SparkFun's product page. (If you want to watch the Product Video on that page, go ahead. However, if you ever wire a breadboard like they have in the video, I will kick you out of my class.) How much current does it draw at 5V? What is the maximum duty cycle at 5V?
5. Connect a 12V power supply to the voltage regulator on your experiment board using a barrel jack and adjust the output voltage to 5V by turning the small screw on the blue potentiometer.
6. Unplug the 12V supply and screw two wires into the output. Use red for positive and black for negative (ground). Do not strip them more than 1/4" to avoid short circuits. Wire the 5V output through a button and your solenoid, as shown in Figure 2.
7. Plug the 12V supply back in and press the button to activate the solenoid. **Be sure that you do not exceed that maximum on time for the solenoid!** Check the datasheet!

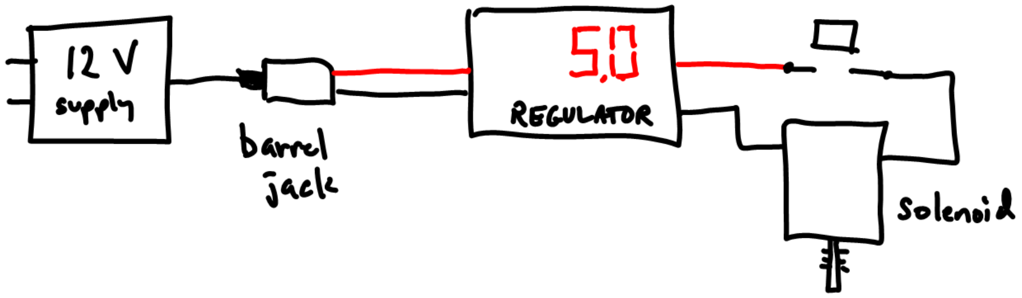

Figure 2: Circuit for testing the solenoid.

Relays

Another application of electromagnetism is found in the electromechanical relay. Here, an electromagnet is used to control a switch, which can be used to drive a much larger current. Figure 3 shows a simplified version of the inner workings of such a relay. When the coil is energized, a metal contact switches position to open or close a circuit. Relays are typically used to control very high currents, since it's easy to make the power side "beefy" enough to handle them.

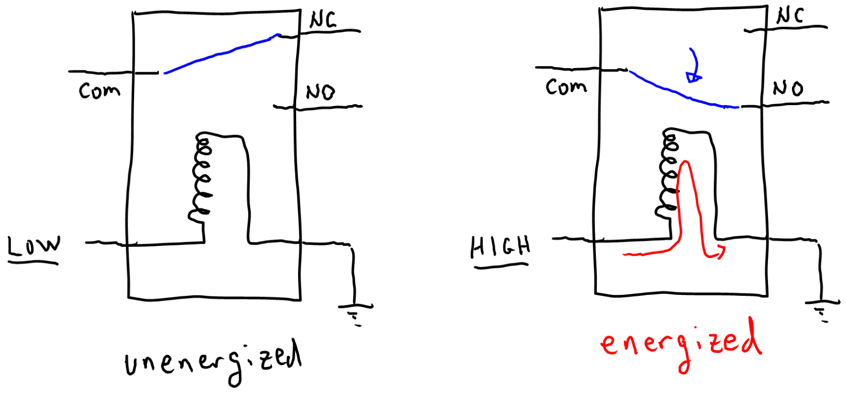

Figure 3: Inner workings of a relay. Normally (i.e., when not energized), COM is connected to the normally closed pin (labelled NC). When energized, the connection switches to the normally open (NO) pin.

1. Do the tutorial in CIRC-11 of the SparkFun tutorials, with modifications:
   - Use the circuit in Figure 4. The one in the tutorial is crap.

   Some notes:

   - As you have seen in previous labs, the reason we need the transistor is that the relay requires more current than pin 2 can provide. Instead of controlling the relay directly, pin 2 controls the transistor, which then controls the relay, since the transistor can handle a much larger current (this one has a max of 700 mA).
   - The diode – a 'flyback' diode – is used to "snub" a voltage spike whenever the relay turns off. Diodes only pass current in one direction, so when voltage is applied to the relay, all of the current goes through the relay. When the relay opens, energy built up in the coils needs somewhere to go. The diode allows the current drain safely.
   - The relay is an "SPDT" relay. The SP stands for "single pole," meaning it has one control input. The DT stands for "double throw'," meaning that it controls two separate circuits: "normally open" and "normally closed." When no voltage applied, the normally closed pin, labelled "NC", is connected to the common pin. That is, "normal" for the relay is no voltage applied, so the normally closed pin is...well...closed. When the relay is energized (voltage applied), the normally open pin, labelled "NO", is connected to the common pin.

RC Servos

As explained in class, servo motors have circuitry in them that allow you to change the position of the motor by sending a pulse width of a given length. The length of the pulse is kind of a "code" that is interpreted by the internal circuitry of the servo, and the length of the low portion can vary somewhat. While controlling the servo is pretty straightforward, it would be a real hassle to generate the pulses yourself. Luckily, the `Servo` library that comes with Arduino does all the heavy

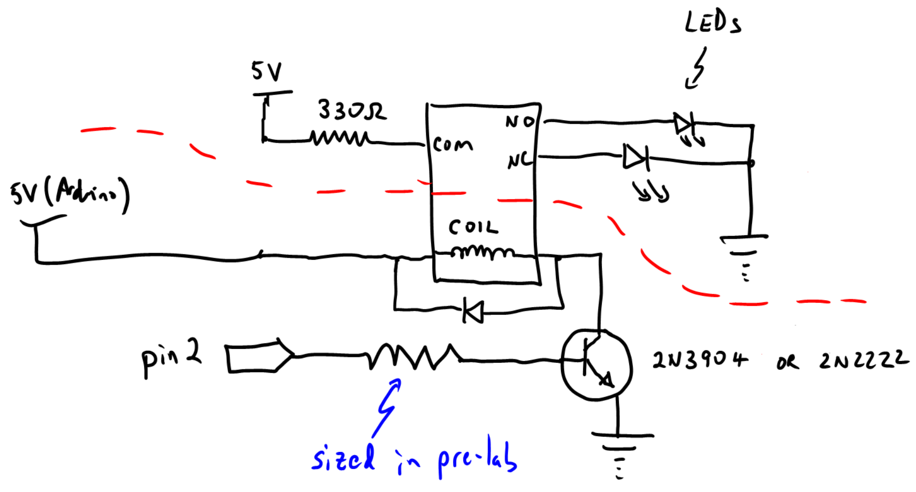

Figure 4: Relay schematic. The dashed line shows the separation between the two circuits, the control circuit on the bottom and the power circuit above.

lifting for you. For one, it manages the timings using the ATmega's built in timers and interrupts, and it also has code to ensure that you don't drive the servo too far in either direction.

Servo motors are usually rated by *holding torque*. As the term implies, holding torque is the maximum torque that the servo can put out when it's held stationary. To turn, the *load* on the servo must be less than the holding torque, otherwise the motor won't be able overcome the resisting torque. For small, hobbyist motors, typically called *RC servos* because of their popularity in remote control vehicles, holding torque is typically measured in oz.-in.: a motor rated for 1 oz.-in. can provide 1 ounce of force with a moment arm of 1 inch. Doubling the moment arm will halve the force, and vice versa, as I'm sure you recall from your physics courses.

Choosing a motor

Before we see how the various motors work, we should first wonder why we want to use an RC servo in the first place. The short answer is that RC servos are fairly easy to implement. Really, they are just DC motors with simple feedback circuit all in one convenient package. One could develop similar functionality with a DC motor, some encoders, and a clever control algorithm.[1] For many applications, its ease-of-use makes the RC servo the obvious choice.

On the other hand, for other applications, one often uses a brushed (or brushless) DC motor or a *stepper* motor – we'll investigate those later in the term. DC motors are typically the cheapest for high-power and high-speed applications, and stepper motors are often used for high-precision applications, though none of these are hard and fast rules. The 3d printers use stepper motors because precision is critical.

[1]Your professor did exactly that for Mr. Football's arms during the Tailgate Design Challenge.

Procedure

1. Do *Tutorial 8: A Single Servo* in your SIK Guide.
2. Using a potentiometer and the Arduino's ADC, construct a circuit and program that allows the user to control the position of the servo by turning the potentiometer, such that there is a 1:1 correspondence (e.g., turning the potentiometer clockwise by 45^o will make the servo do the same).
3. Attach the oscilloscope to the servo's control pin and observe what happens when you turn the potentiometer. How does the Arduino command the servo position?

Show your working system to an instructor.

Instructor initials: ____________________

Heating control

(This section is not complete – it'll serve as the challenge for the coming week, but I need to get the lab out sooner, rather than later.)

Your challenge is to implement a heating and cooling system for your greenhouse. Your heating system will compare the actual temperature to a set-point, and turn on the heat if it gets too cold. If it gets too hot, a servo motor will open the top lid to promote passive cooling.

1. Draw a state diagram that captures the following behavior:
    - Periodically (upon expiration of a timer), the temperature and setpoint are read and compared.
    - When the temperature drops below a minimum (heating) setpoint, the heater turns on.
    - When the temperature rises above the setpoint, the heater turns off.
    - When the temperature rises above a maximum (cooling) setpoint, the lid opens.
    - When the temperature falls below the setpoint, the lid closes.
2. Use the AD22100 temperature sensor for measuring both the inside and outside temperatures.
3. Draw a block diagram of the system. Show how components connect to each other and the "nature" of that connection (e.g., "5V logic", "24V power", etc.
4. Instead of using the Arduino's 5V supply, use your 24V power supply to *independently* supply the *power* (heating) circuit. You will be given 33Ω power resistors to arrange as you best can to produce maximum heat without exceeding 2A. **Do not make any connections between the 24V circuit and the Arduino system!** Not even ground.
5. Draw a complete schematic of your proposed system using the components you have used up to this point. **Label the connections between components with the voltage and current limits.** For example, The connection from the power supply to the Arduino would

have something like, “$V < 12V$” (the Arduino’s limits), and, “$I < 2A$” (the power supply’s limits).Show the schematic to an instructor.

6. Implement the system with the power resistors and your IKEA greenhouse. *Use event-driven programming.*
7. Add code to print both the setpoint and the current temperature to the serial monitor. Set the timer so that the sensors are read every 5 seconds. You may *not* use `delay()`.
8. Demonstrate your working system to an instructor. The instructor will also inspect your code.

Hysteresis

The system you developed in the previous section still has a minor flaw, namely that noise in the temperature sensing can lead to the relay switching on and off rapidly. In a full-sized heating system (e.g., a residential furnace), such *short-cycling* can lead to unnecessary wear and tear on the system; in ours, it could theoretically lead to premature failure of the relay. It’s only a $1 relay, but still, we’d like to avoid the behavior.

As described in the reading, one way to combat this is to use *hysteresis*, or memory, which allows us to change our behavior based on past events. For the system at hand, you will add a variable that raises or lowers the nominal set-point, depending on whether we’ve received a call for heat or cooling.

1. Implement a 1C hysteresis band. Make the band “below” the set-points. That is, the heater should come on when the temperature drops to 1C below set-point, and turn off when set-point is reached. Be sure to update your state transition diagram, which will make the programming task almost trivial.
2. Demonstrate your working system to an instructor.