Summary

The magnetic levitation (Maglev) device is a one semester project that I completed for independent research, Fall 2007, of my sophomore year with advisor Bruce Land. All research was performed outside Cornell University, using my own equipment, resources, and funding. The magnetic levitation device is designed to levitate an object that has been implanted with a permanent magnet at a given height off the ground. The MagLev uses an electromagnet and controlling circuitry to keep the object stable at a given height. The challenge is not simple, as the system is inherently unstable, meaning that the object never wants to stay at a desired position. In particular, the system uses 2 hall-effect sensors for feedback into 1st order PID control loop. The control loop runs on a digital microprocessor (ATMEGA32) and is actuated by and H-Bridge driver (LMD18200).

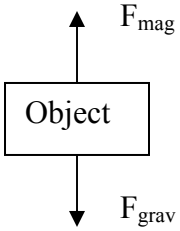
Theory

Instability

The magnetic levitation system is a classic instability problem. Instability is a special type of equilibrium in a system that is nearly impossible to reach without outside interaction. A classic example of a different type of equilibrium is the pendulum. A pendulum has two equilibrium points, based on the solution to its governing equation of motion, one at the top or apex and one at the bottom. The equilibrium at the top is unstable and the equilibrium at the bottom is stable. This can be observed by playing with a pendulum. The pendulum seems to always be attracted towards the bottom after some time, but will never stay at the top. The reason for this is that at the bottom, there are restoring forces that act in opposite directions to the direction of motion. If the pendulum, at the bottom, moves a little to the right, there is a force that moves it back to the left. If the pendulum moves to the left, there is a restoring force to the right. With the application of some small friction, the system will eventually reach the equilibrium position at the bottom. However, this is not the case for the top point. At the top, the forces push the pendulum away from the equilibrium position in either direction, such that the pendulum can never reach its equilibrium position. Instability can be shown graphically as a concave-down potential energy graph, or an unstable equilibrium in a phase plot. The goal of this project is to take the magnetic system and "balance" it. In this system (assume that the electromagnet is turned off), the magnetic object is either attracted upwards to the metallic base of the electromagnet, or is pulled down away from equilibrium by gravity.

Physics of the system

At the equilibrium point, the downward gravitational force (F_g=mg) equals the upwards magnetic force F_{mag}.



However, as the magnetic object moves a bit upward, the magnetic force increases in the upward direction and exceeds the downward gravitational force. Approximations of this force are based on an inverse square law. The magnet will move faster and faster upwards until it is collides with another object. In the opposite direction, if the magnet is displaced slightly downwards of the equilibrium point, the magnetic object will have the same gravitational force downwards, but will have a lesser upwards magnetic force because the radius has been increased, thus the $1/r^2$ magnetic force weakens, and the object falls. Intuition would lead one to believe that in order to stabilize the system, some outside object would need to apply a force downwards once the object moves upwards from the equilibrium position to counter the upward magnetic force, and an upward force once the object moves too far downward to counter the downward magnetic force.

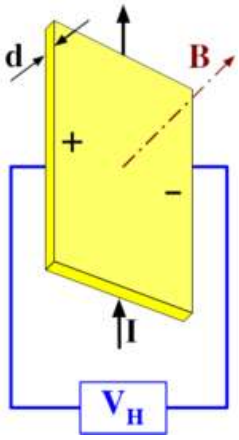
PID Controller

The PID (Proportional, Integral, Derivative) controller is the device which allows for the counter-balancing of forces. The PID controller will try and push the system towards equilibrium. The PID controller needs a way of sensing the state of the system, like how far above or below the equilibrium position the magnetic object has moved. This information is called feedback, and it is provided by two sensors that detect magnetic field strength. This will be discussed in more detail later. The PID controller needs some way to apply these "counter" forces to try and maintain equilibrium- this mechanism is commonly called the output or actuator. In this project, the actuator, or output, is an electromagnet that can vary its output field and, consequentially, the force it exerts on the permanent magnet in the levitated object.

Using feedback, the PID controller determines a quantity called error, which is the difference between the equilibrium position and the current position (error = $x - x_{eq}$). The controller takes this error, in real time, and multiplies it by some gain, or constant called K_p, and outputs this quantity. Output=K_p*error = K_p*($x - x_{eq}$). This term provides a restoring force proportional to the distance from the equilibrium position. The further the object moves toward the electromagnet or the ground, the more the electromagnet will try and push or pull the object. While this sort of control may seem adequate, it neglects oscillations in the system. For example, if the K_p or gain is set correctly, but the object has some inertia proportional to its mass, it will overshoot the equilibrium position when moving towards it. If the object is being pushed downwards for example, it will have some velocity *v* as it approaches the set point. The net force on it will be zero as it crosses the equilibrium position, but it will still pass the point because it has inertia p=m**v*. The object will then be pulled upwards, and the same process will occur in the opposite direction, where it will pass the equilibrium position and continue moving. This cycle will continue indefinitely, and will appear as visible oscillations. The system needs

some sort of damping, and it is implemented by adding a damping, or derivative term to the output expression. If the Output=K_p * error + K_d * d(error)/dt the derivative of the error fights the proportional term to dampen the system. For example, as the object approaches the equilibrium position from the bottom moving upward, the proportional term will continue pulling the object toward the equilibrium position, but the derivative term will know that as the error starts decreasing rapidly, it needs to put on the brakes by applying a force against the direction of motion. There is a third term called the integral term, but it is not applicable here. This term gives the system a bit of a kick if the error is too small to move the system towards equilibrium. This term is more important in systems where there is a lot of friction and high precision is needed, like CNC mills.

Sensing



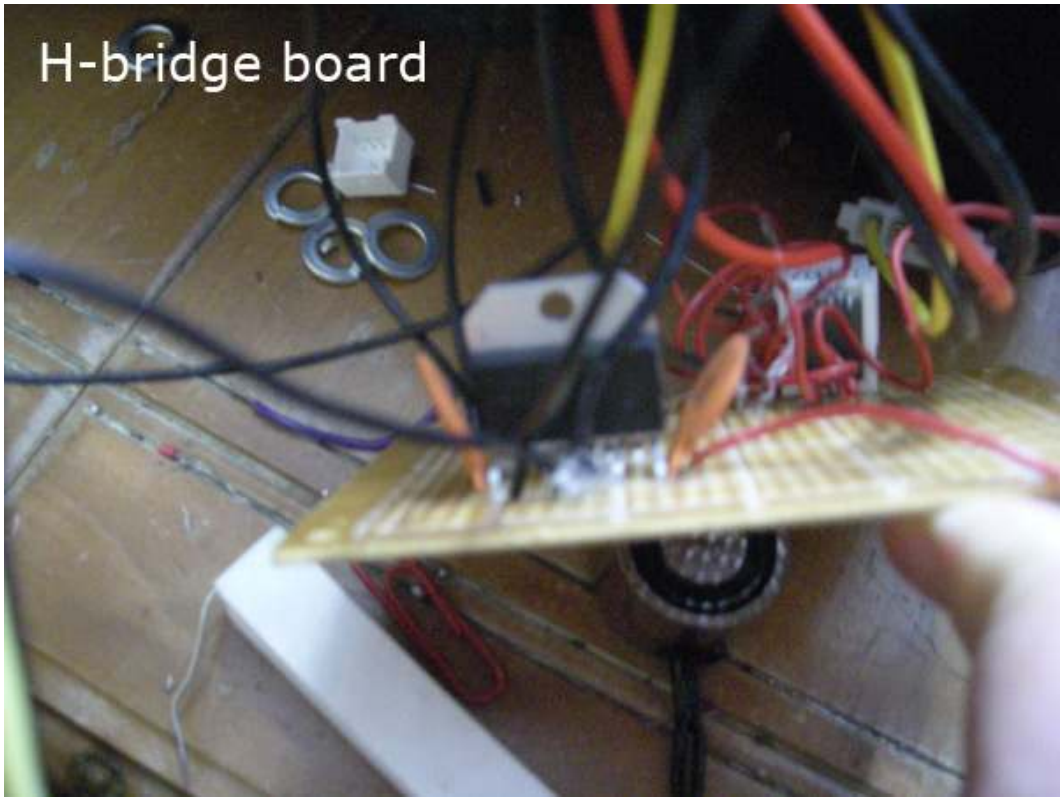
Sensing is critical to feedback for the PID controller. The sensors used in this project are two hall effect sensors. Hall-effect sensors sense magnetic field strength by applying a current through a semiconductor and measuring the voltage that is generated. The hall effect voltage is given by: $V = \frac{Eh}{B}$ Where Eh is the hall field and B is the external field.

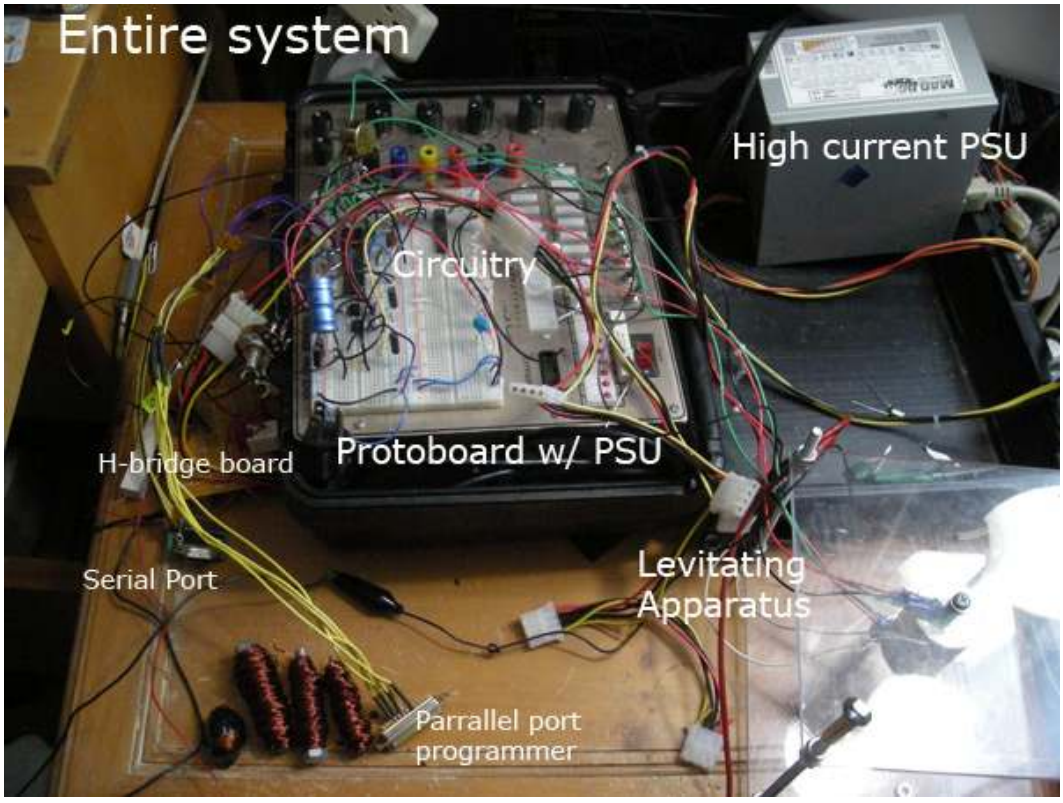
Two sensors are used- one above the electromagnet and one below in an attempt to cancel out the field produced by the electromagnet and solely measure the field of the permanent magnet in the moving object. The closer the object moves toward the base of the electromagnet, the larger the output voltage gets. By correlating field strength to distance using a linear approximation, a position measurement can be made.

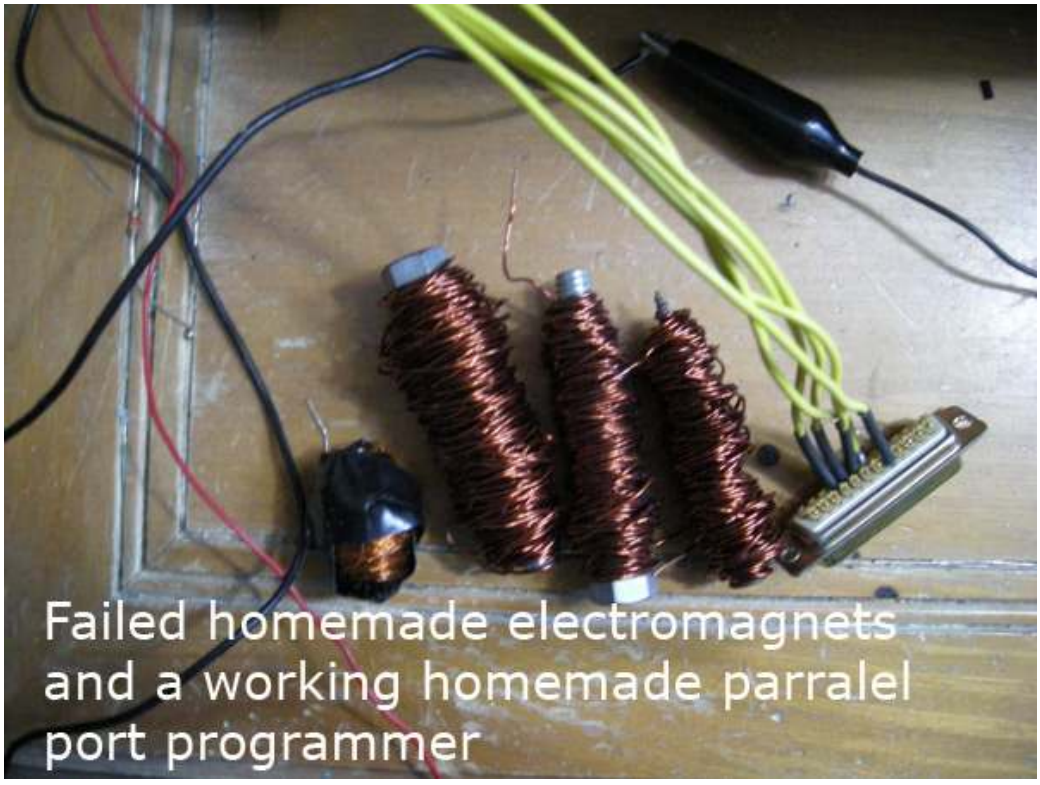
PWM

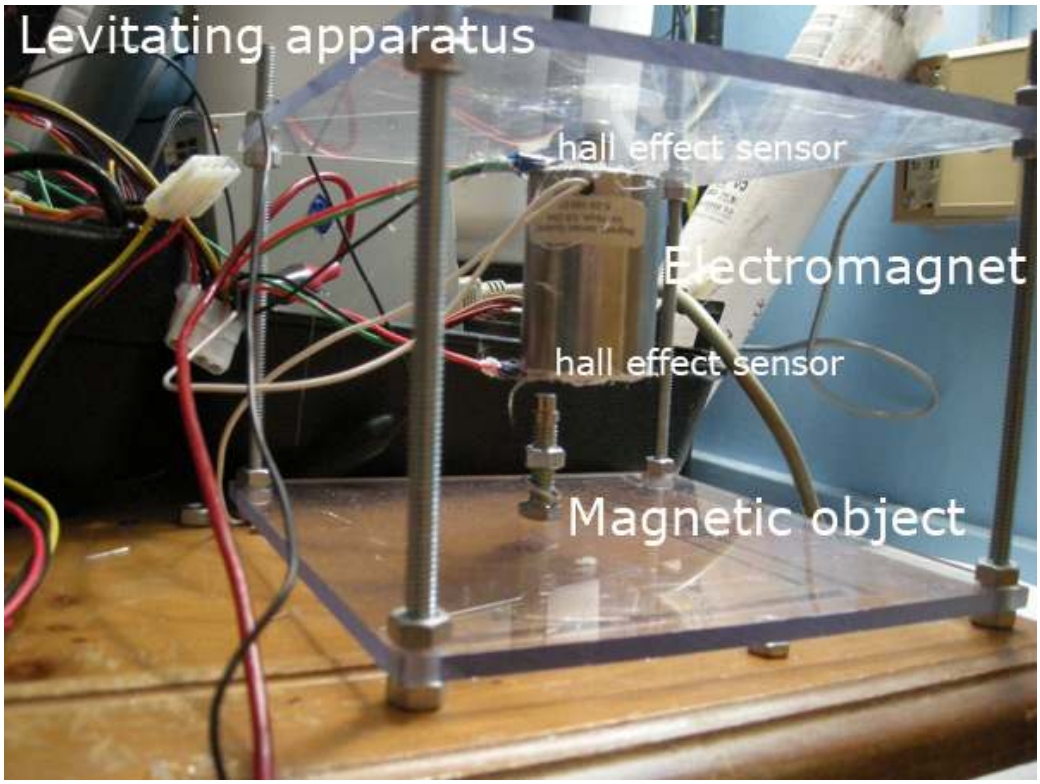
Pulse width modulation is a way to encode a signal in a digital manner. In PWM, a digital square wave has its duty cycle increased or decreased proportionally to an input signal. In a microcontroller's built-in PWM, an 8-bit unsigned integer (0-255) is used to set the duty cycle of the output signal. The microcontroller sets an input of 0 to be a 0% duty cycle (0% on-time) and an input of 255 to be a 100% duty cycle (100% on-time).

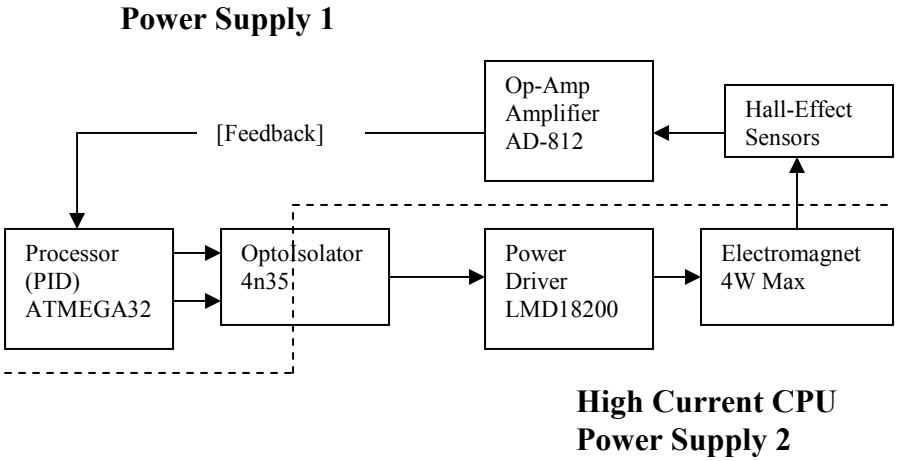
What I Built:


H-bridge board


Entire system
High current PSU
Circuitry
Protoboard w/ PSU
H-bridge board
Serial Port
Levitating Apparatus
Parrallel port programmer


Failed homemade electromagnets and a working homemade parralel port programmer


Levitating apparatus
hall effect sensor
Electromagnet
hall effect sensor
Magnetic object



This diagram shows the overall functional block design of the maglev system. The bulk of my work was spent on the microcontroller code, however I designed circuits for both the op-amp and opto-isolator and designed noise-immunity mechanisms. The PID algorithm is performed on an ATMEGA32 microcontroller, as well as other debugging information, as needed. The microcontroller gets positional information from 2 hall-effect magnetic sensors and uses a PID algorithm to derive floating point signed output. This output is then converted into a Direction signal (5V for the electromagnet to attract the moving object and 0V for the electromagnet to repel the moving object) and a PWM signal with a duty cycle proportional to the magnitude of the output. These two signals are then sent to the isolated H-Bridge (LMD18200) circuit using optocouplers. The H-Bridge is a device that allows for bi-directional magnet operation. The sensors sense the field produced by the moving permanent magnet by canceling out the field of the electromagnet. Since essentially 2 fields are produced around the sensors, one from the electromagnet and one from the moving magnet, it is difficult for a single sensor to distinguish which field belongs to the permanent magnet. Another sensor is introduced at the top of the electromagnet, a large distance away from the moving permanent magnet, and is only affected by the electromagnet. By amplifying this sensor's output and subtracting it from the bottom sensor's output, the field of the electromagnet can be cancelled out. Amplification is achieved (~6.7x) with a small signal, small operating voltage, op-amp. The analog signals are digitized using the microcontroller's internal 10 bit ADC.

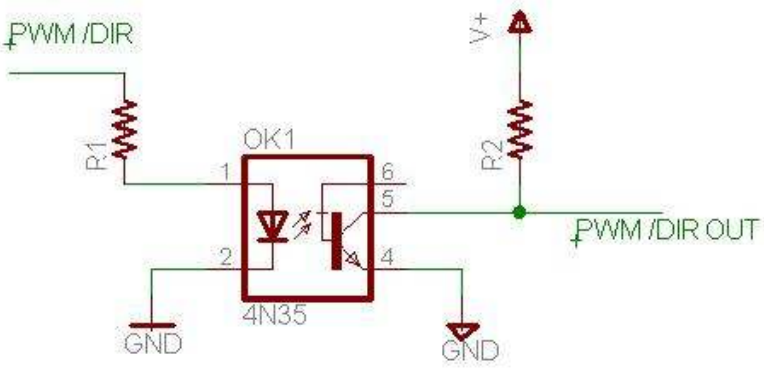
Power

The dotted line represents isolation in power supplies. All logic devices share a common power supply, power supply 1. This power supply is good for up to 3A, and has

very small non-DC components on its supply rails. All power devices share a high-current computer power supply. A computer power supply is one of the least expensive and versatile power supplies available. It is small, relatively lightweight, and commonly available. A 350W unit like the one I used can supply +33A peak on the 5V rail. The reason for using two separate, isolated, power supplies is that a significant amount of noise can be created by the high frequency, high current switching of the electromagnet. These disruptions to the power supply rails can throw off sensor readings and reset digital circuits. Isolation is achieved using optocouples that transmit a digital signal using light, rather than electrons, from LED's and phototransistors as receivers. In addition to isolated supplies, every logic component had a .047uF capacitor placed as close to the GND and +5V supply lines as possible to block any higher frequency noise. Furthermore, each power supply had large (.68 uF) electrolytic capacitors to block any 60 Hz noise that may be present.

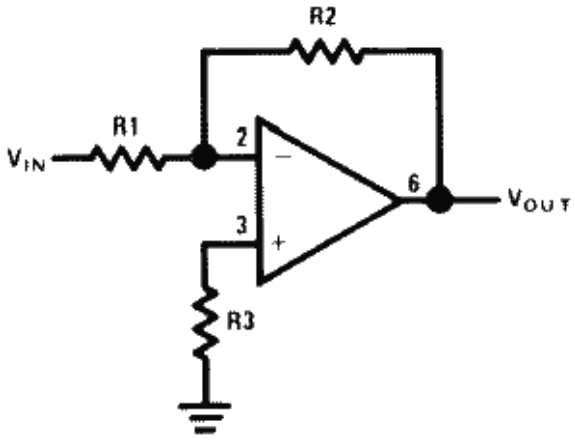
OptoCoupler

As mentioned earlier, an optocoupler is used to isolate the two supplies. The isolator uses an LED on the transmitting end and a phototransistor on the receiving end to allow two separate circuits to communicate. The 4n35 was chosen because of its availability and low cost. Also, it is effective at high frequency transmissions up to 250 Khz, which is much faster than the PWM frequency (7.8 kHz). Special considerations had to be made for the forward current through the LED vs. frequency and phototransistor saturation. Also, consideration had to be placed on the phototransistor current because less current meant slower switching speeds. R_L was chosen to be 1 K Ohm and R_B was chosen to be 1 K Ohm based on switching speed and LED saturation voltage.



Amplifier

Amplification is performed using a basic inverting op-amp circuit. The circuit is shown below.



The gain is set by R2/R1 and I chose R2=100k and R1= 15k G=6.7 and R3= 13k
R3 is a bias resistor and does not affect the gain.

Suspended Object:

I chose to suspend a ¼-20 1" bolt with a ¼-20 nut screwed onto the bolt. The magnet was placed on the tip of the bolt, opposite the head, and the nut was screwed all the way on. The nut adds mass to the system, and it is screwed down to increase the angular moment.

Software

As stated previously, the microcontroller runs the main PID loop by sensing an input and actuating some output. The code consists of four distinct blocks of code, which are conveniently located in four separate files. The code is written in gcc C, through an open source IDE called winAVR. The blocks are

- Initialization Block
- Main loop
- PID function
- RS232 / Debug functions

For the final design, the RS232 block is not used as it slows down the execution of code.

Initialization Block:

The initialization block runs once before any other code runs and it initializes all of the relevant status and control registers for the ATMEGA32. The initialization block:

- Configures Port A as all input pins (ADC pins)
- Configures Port B as all output pins (Debug LED's)

- Configures Port C as all output pins (DIR Pin)
- Configures Port D as all output pins (PWM Timer)
- Sets ADC to slowest speed and max resolution
- Sets ADC to use Analog Reference (AVCC) as a ref. voltage
- Initializes the RS232 to use 9600N81
- Turns on the PWM timer at ~7.8 kHz

Main Loop

The main loop runs infinitely and it polls the two sensors, adjusts them for offset differences, and runs the PID function, passing it the error parameter. Offset differences are calculated ahead of time by realizing that while the sensors are biased to 2.5V, there may be slight variations in this bias, especially with the amplified sensor, as there is an additional offset voltage. In the past versions, the bias points were calculated every time the system started in the initialization block, but I found that the same values were being produced, so these values were hard coded into the memory. There is a .5% error in the amplified signal and a negligible error in the non-amplified signal.

PID Function

The PID function has 3 coefficients, Kp, Ki, and Kd. Kp and Kd are read from the ADC during each function call, perhaps unnecessarily, and the previous error is stored statically so it can be called back later. The function calculates the derivative by taking the difference between the current error and the last error. The output is calculated by adding the error*Kp to the change in error * Kd. Also, the change in error is always multiplied by -1 because as the change in error grows larger, a sign of instability and oscillation, the derivative term must move in the opposite direction of the change for dampening to occur. The PWM is calculated by casting the absolute value of the output into an integer between 0-255 and moving this value into a special PWM register. The direction is calculated by taking the sign of the error. A negative error means that DIR=1. Also, at the end of the function, the current output is stored into static memory to be saved so that the next time the function is called, it can use the error when calculating the change in error for the derivative term.

RS232 / Debugging functions

I did not write the RS232 function set implemented in my code, instead I found it online years ago, posted anonymously on a web site. The code is straightforward; it uses a queue implemented as a ring buffer to hold incoming and outgoing serial data. Since C for the atmel microcontroller is a low-level language, an array is used rather than a more advanced structure like a doubly linked list. Also, pointers are used whenever possible. The RS232 serial and debugging code is not used in the actual code because RS232 slows down the control loop to a crawl (less than 10 Hz).

Source Code: [See attached code]

Results:

In my set-up, quantitative measurements were hard to make. I would need force transducers to measure forces on the object, or optical sensors to accurately measure the position of the object. However, qualitative observations were made based on different settings of the gain parameters and set points. I set up the system such that I lifted the magnetic object upwards with my hand while turning the gain potentiometers with my other hand. I set the proportional gain from 0 to 3, the derivative gain from 0-200, and the set-point from -250 to +250. These numbers may seem arbitrary, as they are scaled many different times through the control loop. However, a gain of 0 still means that there is no contribution from that specific term. Also, the derivative gain includes the discrete division by the time step Δt, so it is really $K_d / \Delta t$ where Δt is small, making the quotient larger than expected.

Tuning Method:

I tuned the system coefficients by finding the set-point, or the point where the electromagnet exerted no force on the permanent magnet, and allowed the magnet to slightly deviate from the set-point. I adjusted the set point until it was about 1 cm below the electromagnet. I chose this as my set-point because it was close enough that the electromagnet would be able to exert strong forces in either direction, while still being far enough away that the permanent magnet would not simply jump up and be attracted to the metallic surface surrounding the electromagnet. I began to increase K_p from 0 upwards until the system would oscillate violently. I stopped increasing K_p when there was an equal probability of the magnet flying upwards or falling downward. I then started increasing K_d from 0 upwards until the amplitudes of the oscillations began to dampen. Additionally, I found that as the oscillations began to dampen, the frequency of the oscillations increased. I found the longest hovering time, less than half a second, when K_p was approximately equal to 2.5, K_d was approximately equal to 200, and the set-point was close to -125. With more sophisticated equipment and sensors, a step response of the system would be calculated, and from that K_p, K_d and the set point could be better approximated.

Non-Linearities

While a linear controller like a PID controller is the simplest and should work well in this system, it is not ideal and there are other non-linear controllers that should theoretically

be able to produce a better balancing result. Firstly, the system is not linear, and can only be approximated as linear in very small regions. The response of the system is asymmetric because in the downward direction, gravity helps pull down the magnet, but in the upward direction, gravity works against the controller. Additionally, even when the electromagnet is turned off, there is still magnetic attraction between the permanent magnet and the metal casing surrounding the electromagnet. The magnetic field from the electromagnet is also not linear- it depends inversely on the distance from the electromagnet. A good approximation of this field is an inverse square law, but again, the field outside of a solenoid, especially at 1 cm away, is very hard to predict. Ideally, the field would be constant, and would only depend on the current supplied to the electromagnet. Another asymmetry occurs because the upward force produced by the field of the electromagnet gets weaker as the permanent magnet moves downward and gets stronger as the magnetic object moves closer. A linearization of this phenomenon assumes that the response is not asymmetric, which is a good approximation as long as the amplitude of any oscillation is sufficiently small ($\Delta r << r$). The asymmetry caused by the force of gravity is difficult to ignore when the magnetic forces are small in comparison to gravity, but the force of gravity can be ignored as long as the magnetic forces are sufficiently large ($F_{mag} >> F_{grav}$). This can be achieved if the set-point is moved closer to the electromagnet. Another set of assumptions that is made, and reasonably so, is that the magnetic field produced at a given point outside the solenoid is proportional to the current flowing through the electromagnet and that the H-Bridge driver provides a current through the electromagnet proportional to the duty cycle of the input PWM signal.

Balancing the Mass:

Careful considerations had to be made when choosing the size and shape of the moving object. The object needed a high angular moment of inertia because the object tended to physically flip and oscillate along a horizontal axis perpendicular to the main vertical axis that passes through the electromagnet. Also, by increasing the mass of the object, the object acquired a higher inertia and allowed for a much slower control loop frequency. A higher inertia means that the object resists changes in velocities more, so the output of the electromagnet needs to be updated less frequently. This is desirable because the control loop can only run at a maximum speed determined by the number of operations to be completed in each loop, and is also desirable because the inductance of the electromagnet prevents it from allowing the current to be changed too quickly. However, a higher mass also leads to a stronger gravitational force, which is not desirable for the system to be linearly approximated. To find a compromise, I added washers to the bolt until I found the longest balancing time (about .5 seconds).

Improvements:

To improve the magnetic levitation device, I would increase the control loop speed and perhaps add a revision in the PID controller for asymmetries in the response of the system. To increase the loop speed, I would change all floating point arithmetic to integer arithmetic and scale any quantities, like the output variable, at the end of the control loop. I would also modify the code to update the K_d, K_p, and equilibrium position quantities once every 20 loop cycles, rather than every time the loop is executed. These parameters don't have to be updated 1000's of times per second, but rather around 10 times per second. I observed that the controller seemed unable to activate changes in the magnetic field quickly enough to adequately affect a change in the motion of the magnetic object. Additionally, I would experiment with changes in PID response, adding 2 sets of K_p and K_d so that the system has a different response when moving in the upward, versus in the downward, direction.