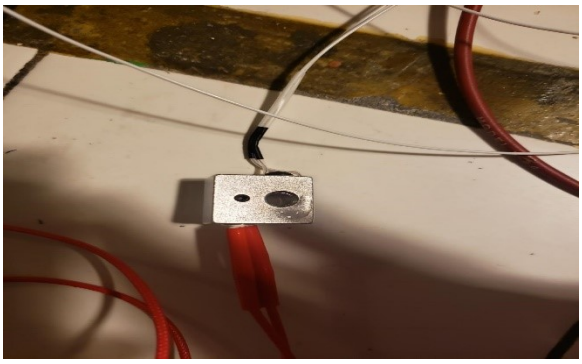
Christopher Jacobs
Electronics and Instrumentation
April 18th, 2020
Proportional Integral Differential Controllers

1 Introduction

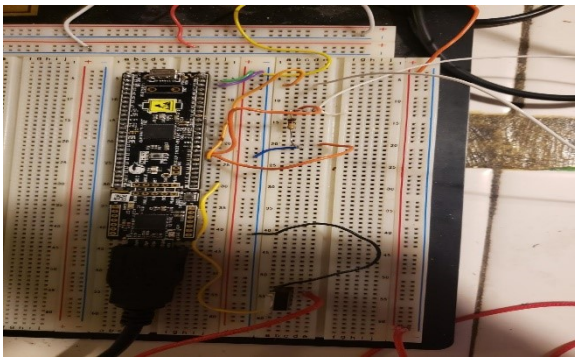
The goal of this experiment is to use a PID controller to keep the temperature of an aluminum block constant. This will be done using a PSoC, thermistor to determine the current temperature of the block, mosfet as a switch on a 3d printer head, a constant current source to power the printer head, and a coded PID controller. We experimentally determine how a PID interacts with the temperature of the block and find parameters on the system that successfully keeps the temperature of the block constant.

2 Apparatus

The system consists of 3 subsystems worth discussing: The aluminum block with a 3d printer head, a thermistor, and a mosfet and a PSoC. The aluminum block consists of a block of aluminum and a 3d printer head this can seen in the below picture. The block is electrically connected to the mosfet and physically connected to the thermistor.



The thermistor is connected to the block using a screw and used to get temperature readings to the PSoC. The mosfet is used to switch current to the 3d printer to heat up the block. The thermistor and mosfet are both controlled by the PSoC which can be seen below. It is also important to note that the mosfet has a power source that is limited to one current connected to its middle pin (drain) The gain pin is connected to the power source and the other pin is connected to ground. The PSoC, thermistor and mosfet can be seen below.



The schematic of the PSoC began with a prebuilt project titled CE210528˙PSoC3˙5˙Thermistor˙Calibration. This project was desirable because it has a prebuilt calibration code for the thermistor and can take user input for offsets. The components inside of the grey square block are used for calibration of the thermistor in combination with the code. This is done by using the VDAC and opamp to send a known voltage across the thermistor and resistor. The voltage is read across the thermistor and then the voltage is read across the resistor. This flip between

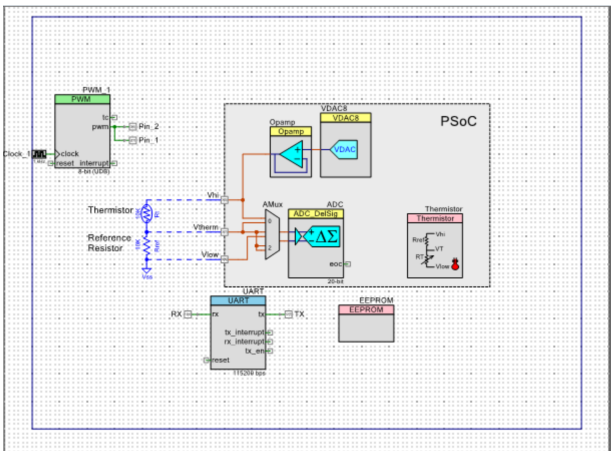
the two components is done by the multiplex and code. Then using Ohm's law and Kirchhoffs voltage laws we can find a relationship between these voltages and the known resistance to find the resistance of across the thermistor. Then we can use known temperatures on the thermistor to get the resistance and calibrate the thermistor. That relationship is the following equation.

$$R_{Thermistor} = (V_{thermistor}/V_{resistor}) * R_{resistor}$$

The PWM is for turning the mosfet on and off for various amounts of time. To increase the on time of the mosfet the duty cycle, this has the effect of warming up the block quicker if or not as quick if the duty cycle is shorter. The PWM's duty cycle is controlled by code and is determined by the following equation.

$$control = G_p error + G_i \sum_{n=0}^{number\ of\ time\ steps} error * time + G_d[(prior\ error\ -\ error)/time]$$

If the control is greater than 255 then the duty cycle has the heater on full blast and if its less than 255 then the duty cycle will become a smaller and smaller percentage until the error between the desired temperature and actual temperature is zero. The above equation will be discussed more thoroughly in the theory section. The EEPROM save inputs from the user and uses it to adjust the reference resistance and current temperature of the thermistor. Below is the schematic for the PSoC for a visual of the discussed components above.



3 Experiment

This experiment was conducted to understand the use of PIDs when keeping constant temperature. First, we began with a thermistor with the goal of determining the current temperature of the block. First we needed to calibrate the thermistor and get its resistance at various temperatures. This was done by hooking up the thermistor and resistor in series and putting a known voltage across the system. Then a voltage drop could be found across the known resistor, giving us the current using Ohm's law. With this now known current and the known voltage across the system the total resistance can be found for the system. Then assuming resistance only from the resistor and thermistor the reference resistor can be subtracted out. Room temperature (21 degrees Celsius), boiling water(100 degrees Celsius), and ice water(0 degrees Celsius) were used as known temperatures. Once the thermistor was calibrated, we could screw it onto the aluminum block and determine its temperature precisely.

Temperature	Resistance thermistor(ohms)
Ice water	256K
Room	100k
Boiling Water	8k

Second, we needed a way to control the temperature of the aluminum block. This began with just using a 3d printer head and using the PSoC to turn off current to the heater when the appropriate temperature was reached. This resulted in the temperature overshooting the desired state because when the heater would turn off the block

would keep heating up. It is important to note that heating up the block took longer than cooling the block, so we would get sinusoidal temperatures, above the desired temperature, but with elongated downward section. Cooling took about 5 to 6 times longer than heating and this was found by allowing the block to heat up to a certain temperature and then seeing how long it took to cool down. This caused me to us ice to cool down the block for quicker experimentation. This combines the mosfet subsection of the system, the PWM, and the PID controller, which gave us precise control of the temperature. The PID will be discussed in the thoery section and it effects in the results section.

4 Theory

The PID controller begins with an equation that was already referred to in the apparatus section and was discretized for coding use.

$$control = G_p error + G_i \sum_{n=0}^{number\ of\ time\ steps} error * time + G_d[(prior\ error\ -\ error)/time]$$

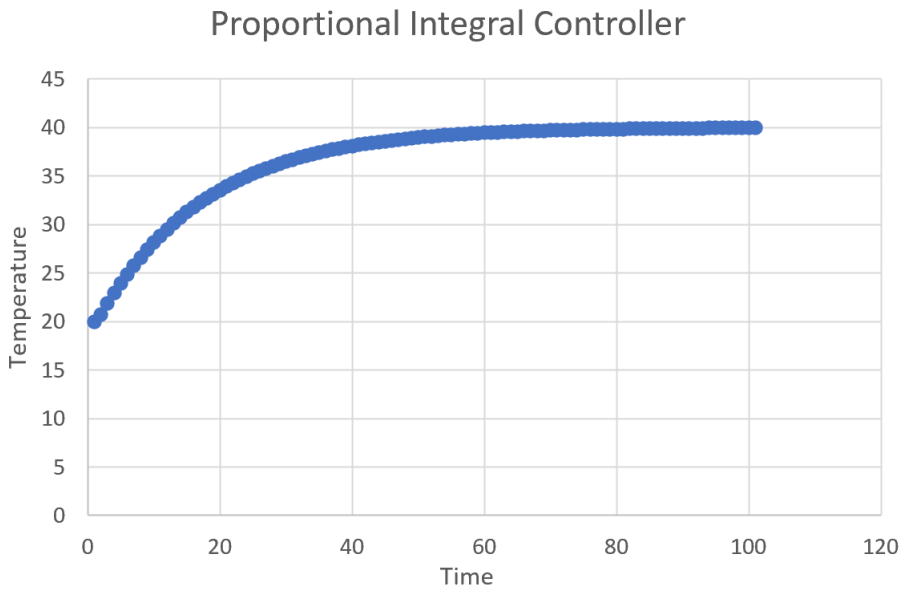
The continuous version is as follows.

$$control = G_p error + G_i \int_0^T error\ dt + G_d \frac{\partial error}{\partial t}$$

The variables are as follows.

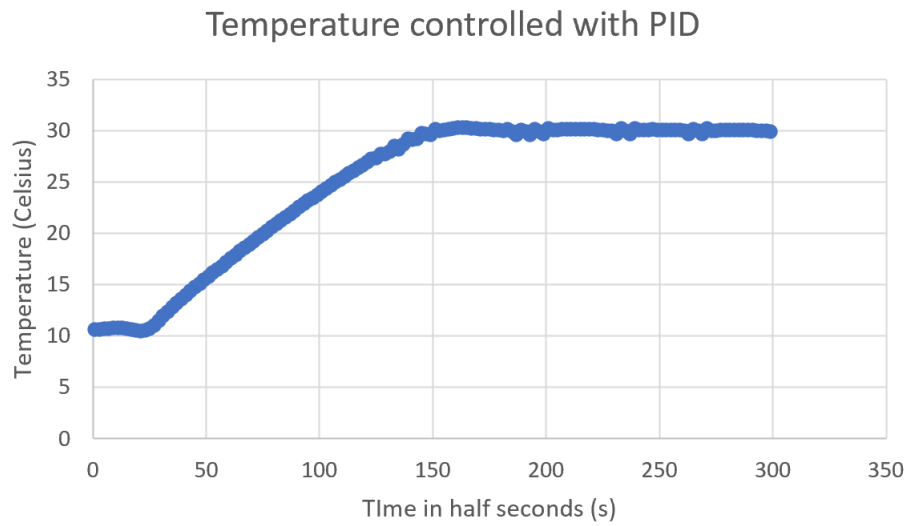
error	current temperature - desired temperature
Gp	proportional gain
Gi	integral gain
Gd	differential gain
control	control signal

Solving this equation for the control signal results in a second order differential equation with solutions of exponentials. which results in the shape of the below graph. Experimentally we found the general shape was the same as the theory, but the theory missed on the over shooting of the desired temperatures and on the small oscillations. This could be due to use losing information when taking a derivative of the control signal to find a solution. It's also important to note that there are constraints on the gain due to us wanting a control system that does not oscillate and acts like a critically damped oscillator that goes quickly to the desired temperature. Also, the proportion gain deals with large changes in error quickly. If this is too large the control will overshoot the desired temperature or if it is too small, it will take longer to reach the desired temperature. The integral gain deals with changes in error as time goes on and, if the temperature is above the desired temperature this will turn off the controller quicker and if the desired temperature is close to the actual temperature, it will slow down the heater and not have the current temperature over shoot the desired temperature. The differential gain deals will small errors between the current and prior temperature which helps dampen oscillation.



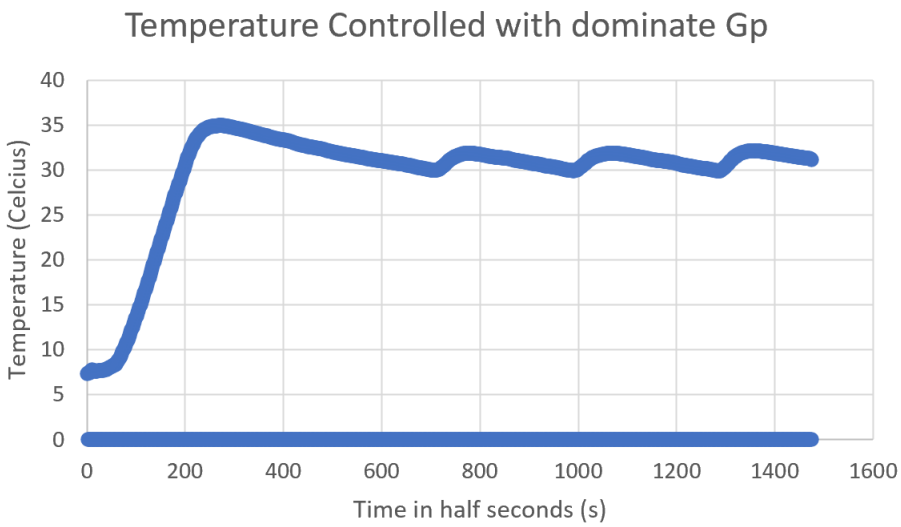
5 Results

The following graphs are results from heating the block up with various gains. The first graph shows a successful attempt at using the PID controller and the second graph shows what can happen if some of the constants are too great. Below each graph is the constants for the controller that created the graph.



Gp	60
Gi	-0.1
Gd	-6

As can be seen we successfully controlled the temperature of the block to reach a desired temperature and stay constant there.



Gp	100
Gi	-0.1
Gd	-6

As can be seen we over shot the temperature, because we know that the proportional gain causes overshooting we can lower it's magnitude and get a better result which we can see a positive result above.

6 Conclusion

The temperature of an aluminum block can successfully be controlled using a thermistor, mosfet, and a coded PID controller. If you need a system with a steady state, then attempting to control that system with a on/off controller will not be desirable. You would overshoot the desired temperature and oscillate above the temperature. A PID controller can slow down the system as you get closer to the desired temperature and dampen those oscillations.