EECS 690

Thermal Oven Project Report

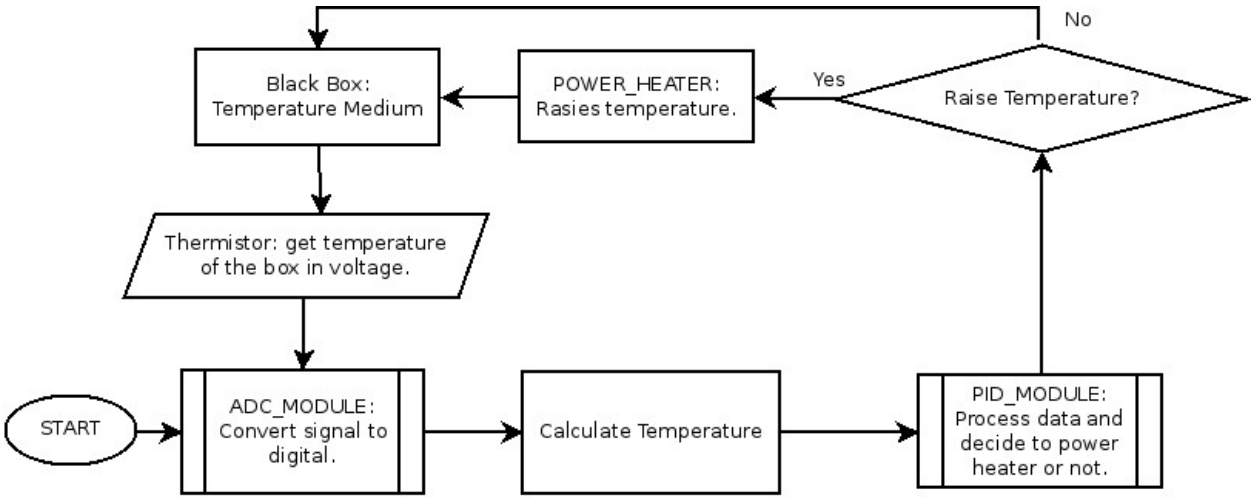
Group 11

Paul Charles
Tim Clark
Diego Soliz
Giordanno Castro
Warren Scipio

I. Module List

1. **ADC Converter**
2. **Temperature Converter**
3. **PID Controller**
4. **Power Heater**

II. Diagram



III. Pseudocode

ADC Pseudocode

```
include temperature_sensor_definitions
include board_libraries
include queues
create_queue my_queue

ADC_TASK(){
my_queue = setup_queue(parameters)
Enable_Peripheral_GPIO
Enable_GPIOS
GPIO_Configure
Enable_Peripheral_ADC
Enable_ADC
ADC_Configure

while(true){

trigger_ADC
data = ADC_data_get()
my_qeueue.send_data(data)

task_delay()
}
}
```

Temperature Converter Pseudocode

-Tm, temperature measured
-Tc, temperature converted
-Ts, desired temperature
-R, power resitance

#include all relavent TIVA libraries
#include ADC converter .h file
Set GPIO of the sensor to the TIVA board

import parameters through the ADC converter to convert signal of temperature

```
void Tempconverterask(void *pvParameters)
{
        while(true)
        {
                measures temperature of the oven and set that to Tm
                -collect signals from ADC for a time period to obtain constant temp, Tm
                measures power resistance using the ADC converter and set that to R
                Tm = measured temperature
                R = measured power resistance
                convert resistance of to C
                convert temperature using equation:
                VTemp=Rdd*(R/(R+Rbias)

                vTaskDelay();
        }
}
```

PID Pseudocode

```
import math_functions
import queues

PID_TASK()
{
while(true)
{
if queue.get_temperature() >= desired_temp
{

    #do nothing

}
else
{

    #slope will always be a positive value
    Temp_difference_slope = calculate_slope()

    supply_voltage_to_heater(5_volts * T_diff_slope)

}

}
```

Heater Pseudocode

```
#include all relevent TIVA libraries
#include thermister .h file
#include PID .h file
Set GPIO to the power resistor of the heater
powerdrive=off

void HeaterTask(void *pvParameters)
{
        while(true)
        {
                Obtain measured temperature, Tc, from thermistor .c file
                Compare Tc to Ts using the PID controller
                if(Tc != Ts)
                {

                        Turn Power Drive On or PWM until Tc=Ts
                        -uses feedback loop mechanism of PID to bring Tc to setpoint of Ts
                        -powerdrive=on;
                        -This increases or steps Temperature from Tc to Ts
                }
                else
                {
                        Keep Power Drive off
                        -powerdrive=off;
                }

                vTaskDelay();
        }
}
```

IV. List of Parameters

List of parameters used:

- ADC
  - analog temperature sample wave (input)
  - converted digital temperature value (output)
  - sample rate of temperature into the ADC (input)
- Temperature Converter
  - digital temperature from ADC (input)
  - outputted calculated temperature (output)
- PID
  - calculated temperature from temp converter (input)
  - set temperature (input)
  - boolean value which determines if input temperature equals set temperature (output)
  - integrally calculated value of how much heat needs to be output based on status of boolean value (output)
- Heater
  - boolean value form PID (input)
  - amount of heat outputted to the oven based on boolean value (output)

V. Connections between Modules

The modules exchange information in a very unique way. The process starts with temperature coming into the black box and merging with the ADC controller. The ADC controller converts the temperature from analog to digital. This digital value will be sent to the temperature converter module. This value will be recalculated to account for the resistance from the thermistor. After this, the converted temperature value will be sent to the PID module. If the PID senses that the temperature isn't at the desired state, then it will trigger the power heater module to send more heat using an integrated formula. The amount of heat will be based on an overshoot until the input temperature is equal to the desired temperature.

VI. Data Collection for Thermal Oven

There are a number of ways in which the thermal oven will collect operating data. One specific way is how the ADC converter collects analog signal sample and converts the waves into a digital signal that will be the measured temperature. That measured temperature, Tm, will be converted to a calculated temperature, which takes in temperature resistance of a thermistor. This calculated temperature, Tc, will be sent back to the computer in a variable to be sent to the next part of the process, which is the PID controller. The PID controller is a mechanism that has a feedback loop. This works by collecting the Tc value and comparing it to a setpoint, which is the desired temperature, Ts. This controls measurements by finding the difference between the setpoint, Ts, and the Tc by implementing the controller until the Tc is equal to the Ts. The way that the Tc becomes the Ts, however, is done by using a power drive of the heater. So each time the heater will turn on until the actual temp of the oven is equal to the desired temp. Therefore, the heater operates in a feedback loop, by inputting heat with the power drive until the Tc equals Ts. Therefore, this demonstrates how data is collected for the PID and heater until the oven is at the preferred temperature.