Milestone 2: Temperature Control System

Julia Konstantinos and John McAvoy
Rowan University

December 5, 2018

Contents

1 Design Overview

The goal of this lab was pick an MSP430 family processor and on a register level, design it to be an temperature controller that controls the temperature of a heat source by controlling the speed of a fan.

1.1 Design Features

The temperature controller is able to control a system to any temperature between 0 °C and 100 °C within 1 °C. Using, UART you can specify the desired temperature as well as monitor the temperature in real-time.

1.2 Featured Applications

This controller can be used for keeping circuit components from overheating.

1.3 Design Resources

The resources that were given to design this controller were: a F5529 launchpad, 5V regulator, 12V DC fan, various resistors, wire, and a breadboard.

1.4 Block Diagram

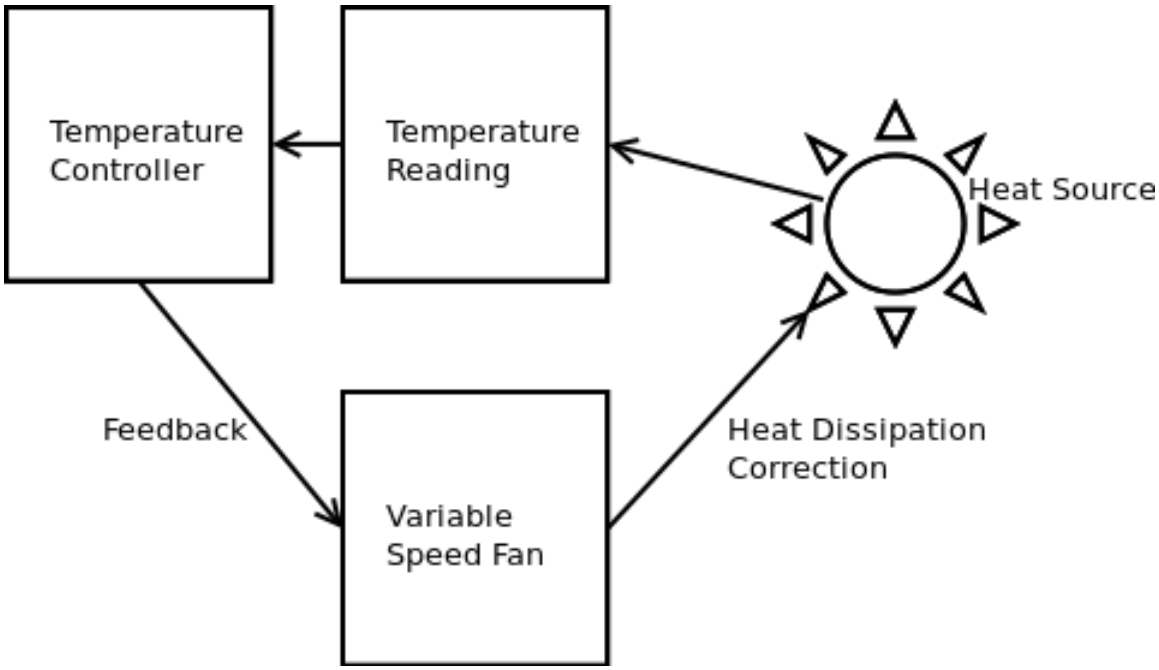

Figure 1: Detailed Block Diagram

2 Key System Specifications

3 System Description

The controller takes in a temperature value from UART to be set as the temperature goal, measures the temperature by measuring the voltage of a thermistor voltage divider using the F5529's ADC_12, and then the fan's PWM is adjusted to create a steady-state at the desired temperature.

3.1 Detailed Block Diagram

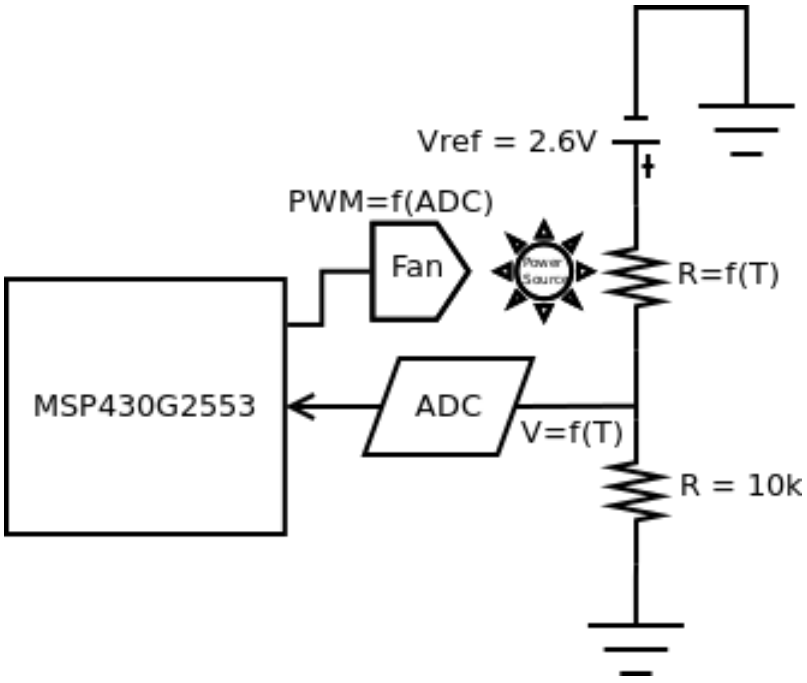

Figure 2: Detailed Block Diagram

4 System Design Theory

4.1 Thermistor Voltage Divider Setup

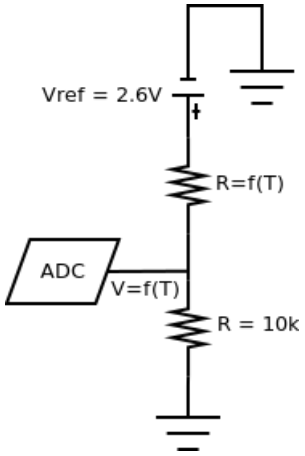

Figure 3: Voltage Divider

4.2 Optimizing Temperature Precision

Vmin	Vmax	Vref	Rload	Tmin	Tmax
V	V	V	Ohms	C	C
0	3.3	2.6	1.00E+04	0	100

Figure 4: Voltage Optimization

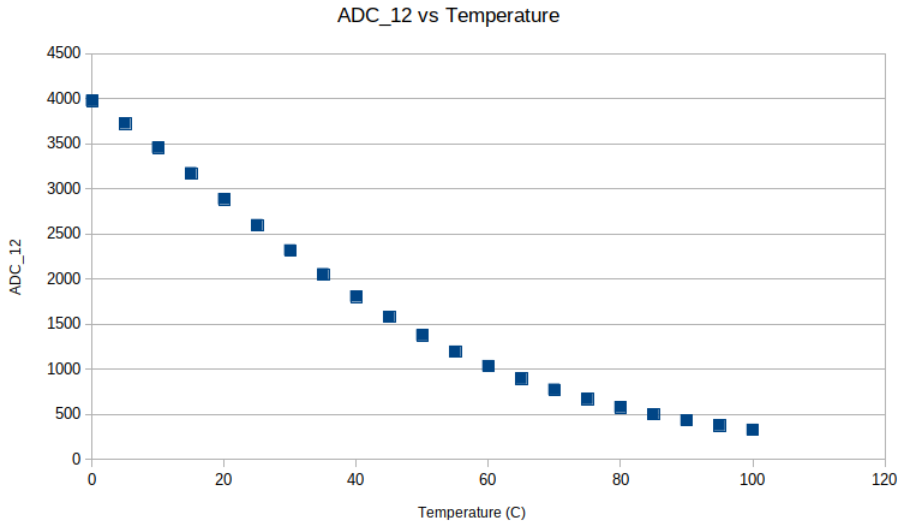

Figure 5: Voltage Optimization

4.3 Software Modified Bang-Bang

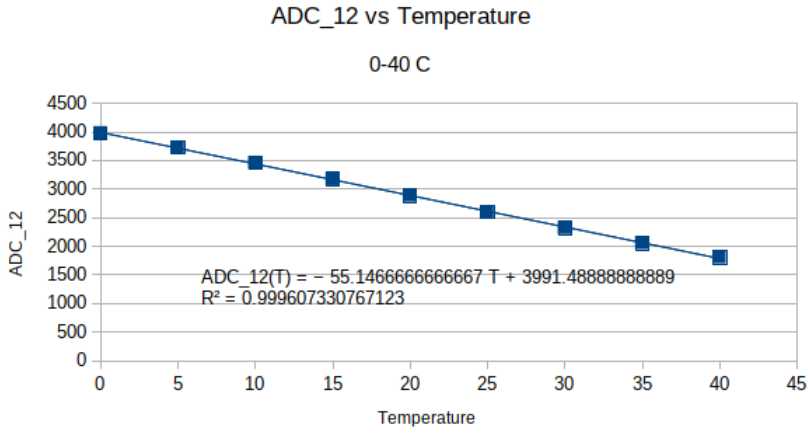

Figure 6: Linearization: 0 - 40 °C

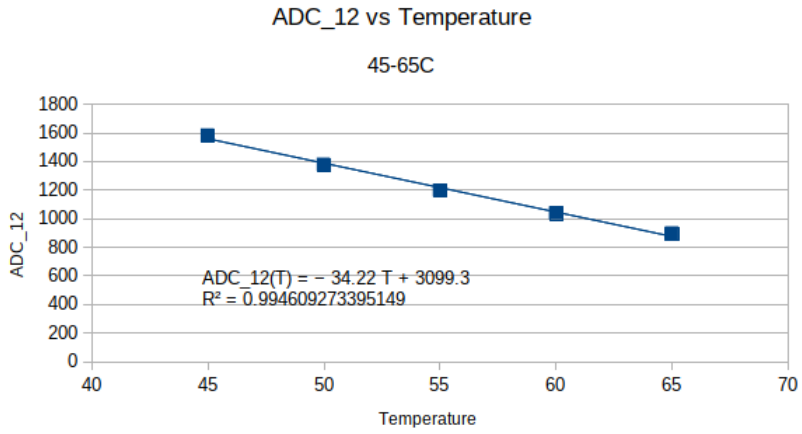

Figure 7: Linearization: 45 - 65 °C

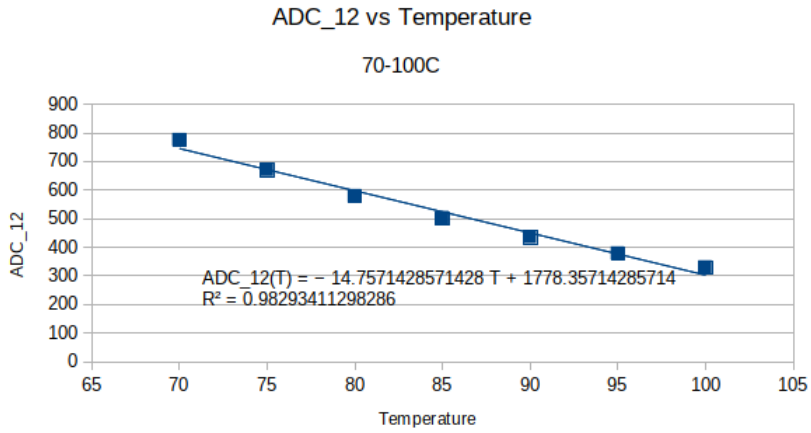

Figure 8: Linearization: 70 - 100 °C

```
  return ((float)adc − 3991.4888889) / −55.1466667;
}
else if(adc >= 896) {
  return ((float)adc − 3099.3) / −34.22;
}
else {
  return ((float)adc − 1778.357143) / −14.75714;
}
```

```
}

void updateHistory(uint16_t adc12){
  for(uint8_t i = 7; i > 0; i--){
void setTemperatureGoal(float temperature){
  if(temperature <= 40){
    adc_goal = -55.1466667 * temperature + 3991.4888889;
  }
  else if (temperature <= 65){
    adc_goal = -34.22 * temperature + 3099.3;
  }
  else if (temperature <= 100){
    adc_goal = -14.75714 * temperature + 1778.357143;
  }
  else {
    adc_goal = 0x3ff; // out of range supported range
  }

  for(uint8_t i = 0; i < 8; i++) {
    adcHistory[i] = adc_goal;
  }
}
```

4.4 Software Noise Reduction

In order to prevent the temperature control logic from making drastic changes if a noisy signal is received in the ADC, a history of ADC values is recorded and the control logic takes the average ADC value instead of the instantaneous value.

```
float adc2Temperature(uint16_t adc) {
  for(uint8_t i = 7; i > 0; i--){
    adcHistory[i] = adcHistory[i-1];
  }
  // 0 1 2 3 4 5 6 7
  // * 0 1 2 3 4 5 6
  adcHistory[0] = adc12;
}

void handleTemperatureControl(uint16_t adc12){
  return adc_goal - calcAvg();
}

uint16_t calcAvg() {
  uint16_t sum = 0;
  for(uint8_t i = 0; i < 8; i++){
    sum += adcHistory[i];
```

```
    }
    return sum >> 3;
}
```

5 Getting Started/How to use the device

5.1 Wiring Diagram

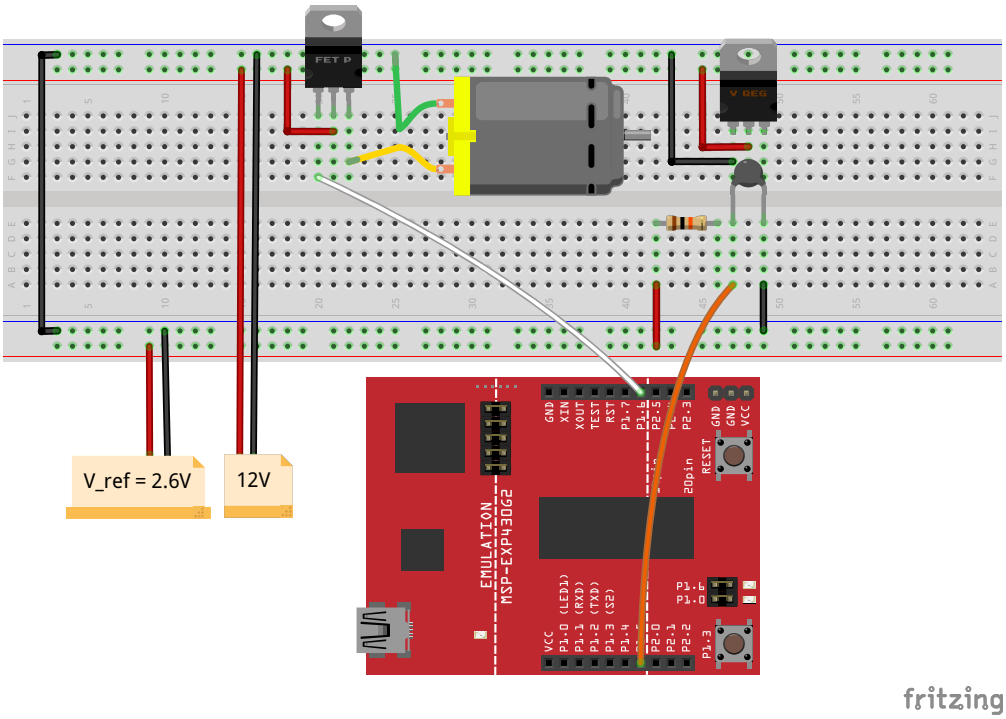

Figure 9: Circuit Wiring Diagram

5.2 Communicating with the Device

The instruction bytes so set the desired temperature in degrees Celcius can be sent via the F5529 launchpad's USB port or by connecting P1.1 (RX) to a UART signal. The controller's output can also be viewed by connecting the launchpad's USB to a computer and running a serial terminal (such as cutecom) that is able to view the hexadecimal bytes transmitted by the controller, or by connecting P1.2 (TX) to a UART decoder.

5.3 Bill of Materials

- 1 x MSP430F5529 Launchpad
- 1 x Breadboard
- 1 x 10kΩ Resistor
- 1 x 10kΩ thermistor
- 1 x 5V regulator
- 1 x 12 V DC fan
- 1 x NMOS
- Wire