HW6 Simon Ng

Wednesday, May 6, 2020 16:14

1.
   a. The temperature from the chip varies by 0.0625 C, it's maximum resolution. The thermistor temperature varies by around 0.08 C.
   b. I tested the resolution by holding an ice pack close to the sensor.

	10 ms	**100 ms**	**1000 ms**
Thermistor	Dropped every 2-5 reads	Dropped every read at first, but as it read cooler temperatures, it changed every few reads	Dropped every read
MCP9808, Res 0.0625C	Dropped every 11 reads	Dropped every ~2-9 reads, with the number of equivalent reads increasing as the temperature decreased	Dropped by 0.0625 almost every read, only a few times when it remained at the same reading as the previous read
MCP9808, Res 0.5C	It's hard to tell because of the same concern with the actual chip temperature not changing fast enough. However, according to the spec sheet, even the 0.5C resolution should take ~30 ms between reads, so the same temperature should be printed at least 3 times.	Appears to change every few reads, though when it takes multiple reads, I suspect it is only because the actual chip temperature isn't changing fast enough from my ice pack.	Seems to be able to change every read or two.

2. After discussion on Friday, I was able to use the bit shift operator to create my high and low bytes of the temperature limits and feed them into the upper and lower temperature limit registers. I faced prolonged problems getting the alarm pin to go off when temperature thresholds were exceeded. I read what I thought were the relevant parts of the spec sheet but still could not get the alert pin to work. I tried looping through all possible bytes for the CONFIG address in case I was misunderstanding some pin. It's possible there's a bug in that code, but no combinations worked (i.e. none gave different digitalReads for an acceptable and not acceptable temperature). I wrote a function (printDiagnostics) to print things to help me troubleshoot, including another function to easily print bytes. In this version, I don't call printDiagnostics though, and the temperature is never calculated. I added T_crit at the same value as T_up and this made the upper limit work, though the lower limit still is not detected and after scouring the specs sheet, I still could not find anything to get the lower limit to trigger the alert pin.

```
#include <Wire.h>

#define ALERTPIN 2
#define R 3
#define G 5
#define B 6

void setup() {
  pinMode(ALERTPIN, INPUT_PULLUP); // temp alert pin w/ internal pullup
  pinMode(R, OUTPUT); // pwm LED output
  pinMode(G, OUTPUT); // pwm LED output
  pinMode(B, OUTPUT); // pwm LED output
  Wire.begin(); // join i2c bus (address optional for master)
  Serial.begin(9600);

  // Get temp limits in binary
  float t_room = 19.5; // C
  float t_up = t_room + 3.0; //C . Max resolution is 0.25 deg
  float t_low = t_room - 3.0; //C . Max resolution is 0.25 deg

  // Activate Alarm
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0x01)); // set pointer to go to configuration register
  Wire.write(byte(0b00000000)); // leave high byte defaults
  Wire.write(byte(0b00001000)); // turn on Alert, leave rest of bits as defaults (nothing locked and in comparator mode)
  Wire.endTransmission();     // stop transmitting

  // Set resolution
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0b00001000)); // set pointer to go to temperature register
  Wire.write(byte(0b00000000)); // change resolution to +-0.5C
  Wire.endTransmission();     // stop transmitting

  // Set T_upper
  int intT_up = round(t_up * 4);
 byte highT_up = intT_up >> 6;
 byte lowT_up = intT_up << 2;
 Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
 Wire.write(byte(0b0000010)); // set pointer to go to upper temp limit register
 Wire.write(highT_up); // set high byte
 Wire.write(lowT_up); // set low byte
 Wire.endTransmission();     // stop transmitting

  // Set T_lower
  int intT_low = round(t_low * 4);
  byte highT_low = intT_low >> 6;
  byte lowT_low = intT_low << 2;
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0b0000011)); // set pointer to go to lower temp limit register
  Wire.write(highT_low); // set high byte
  Wire.write(lowT_low); // set low byte
  Wire.endTransmission();     // stop transmitting

  // Set T_crit
  float t_crit = t_up; //C . Max resolution is 0.25 deg
  int intT_crit = round(t_crit * 4);
  byte highT_crit = intT_crit >> 6;
  byte lowT_crit = intT_crit << 2;
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0b0000100)); // set pointer to go to critical temp limit register
  Wire.write(highT_crit); // set high byte
  Wire.write(lowT_crit); // set low byte
  Wire.endTransmission();     // stop transmitting
  //*/
}

void loop() {
  if (!digitalRead(ALERTPIN)) { // temp alarm tripped
    Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
    Wire.write(byte(0x05));             // set pointer to read which limit was exceeded
```

```
    Wire.endTransmission();     // stop transmitting
    Wire.requestFrom(0b0011000, 1); // request 1 byte from slave device
    if (Wire.available()) { // is there data to be read? If more than 0 bytes, read it
      byte highByt = Wire.read(); // read first byte off buffer, high byte, largest number
      Serial.print("TEMP Byte: "); printByte(highByt); // print high byte of temperature register to troubleshoot
      if (bitRead(highByt, 6)) { // temp is too hot. LED red
        analogWrite(R, 0);
        analogWrite(G, 255);
        analogWrite(B, 255);
      }
      else if (bitRead(highByt, 5)) {// temp is too cold. LED blue
        analogWrite(R, 255);
        analogWrite(G, 255);
        analogWrite(B, 0);
      }
    }
  }
  else { // alert pin not high. temp is okay, LED green
    analogWrite(R, 255);
    analogWrite(G, 0);
    analogWrite(B, 255);
  }
}


void printDiagnostics() { // print digital input and CONFIG register for reference and troubleshooting
  Serial.println();
  Serial.print("DigitalRead: ");
  Serial.println(digitalRead(ALERTPIN));
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0x01));             // set pointer to read CONFIG pin
  Wire.endTransmission();     // stop transmitting
  Wire.requestFrom(0b0011000, 2); // request 2 byte from slave device
  if (Wire.available()) { // is there data to be read? If more than 0 bytes, read it
    byte highBytCONFIG = Wire.read(); // read first byte off buffer, high byte, largest number
    byte lowBytCONFIG = Wire.read(); // read second byte off buffer
    Serial.print("CONFIG: "); printByte(lowBytCONFIG);
  }

  // Calculate Temp for troubleshooting/reference
  Wire.beginTransmission(0b0011000);  // this is our 7bit device address 0011000 default mcp9808 . Write address byte
  Wire.write(byte(0x05));             // set pointer to read temp
  Wire.endTransmission();     // stop transmitting
  Wire.requestFrom(0b0011000, 2); // request 2 bits from slave device
  if (Wire.available()) { // is there data to be read? If more than 0 bytes, read it
    byte highByt = Wire.read(); // read first byte off buffer, high byte, largest number
    byte lowByt = Wire.read(); // read first byte off buffer, high byte, largest number
    boolean negative = bitRead(highByt, 4); // we want the sign of the number
    boolean exceedsTupper = bitRead(highByt, 6); // is T_A>T_Upper tripped
    boolean exceedsTlower = bitRead(highByt, 5); // we want the sign of the number
    highByt = highByt & 0b00001111; // bitwise And
    int intTemperature = (highByt << 8) + lowByt;// int is two bytes. bitshift operator has order of operations
    float MCP9808Temp = intTemperature / 16.0;
    if (negative) MCP9808Temp = -1 * MCP9808Temp;
    Serial.println(MCP9808Temp, 6);
    Serial.print(exceedsTupper);
    Serial.print(" ");
    Serial.println(exceedsTlower);
  }
}

void printByte(byte myByte) {
  for (int i = 7; i > -1; i--) {
    Serial.print(bitRead(myByte, i));
  }
  Serial.println();
}
```