18th of January, 2023

STUDENT

Name: Doru Gabriel Chete

Group: 30433

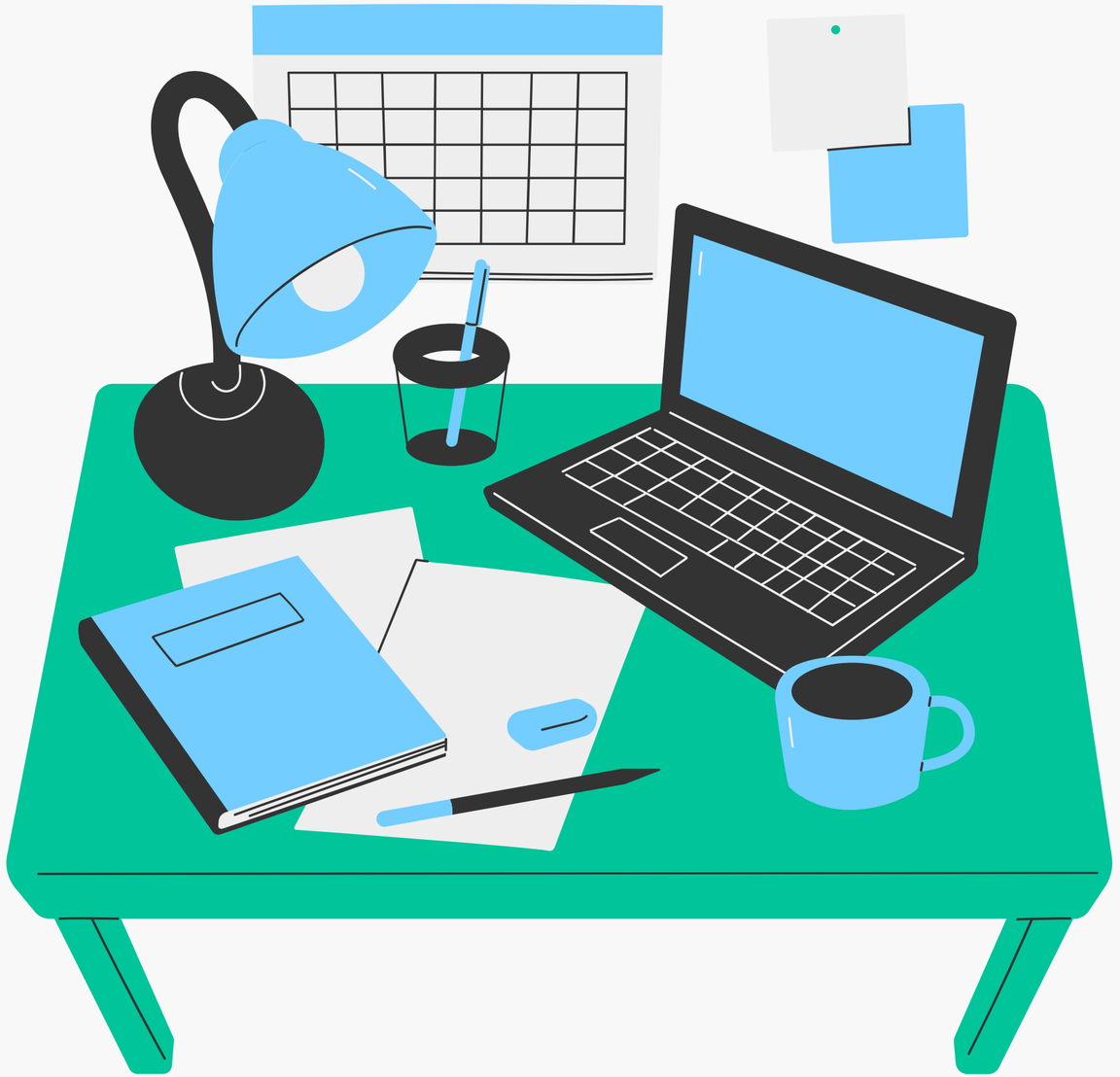
Design with Microprocessors Semester Project

Temperature-controlled Lamp

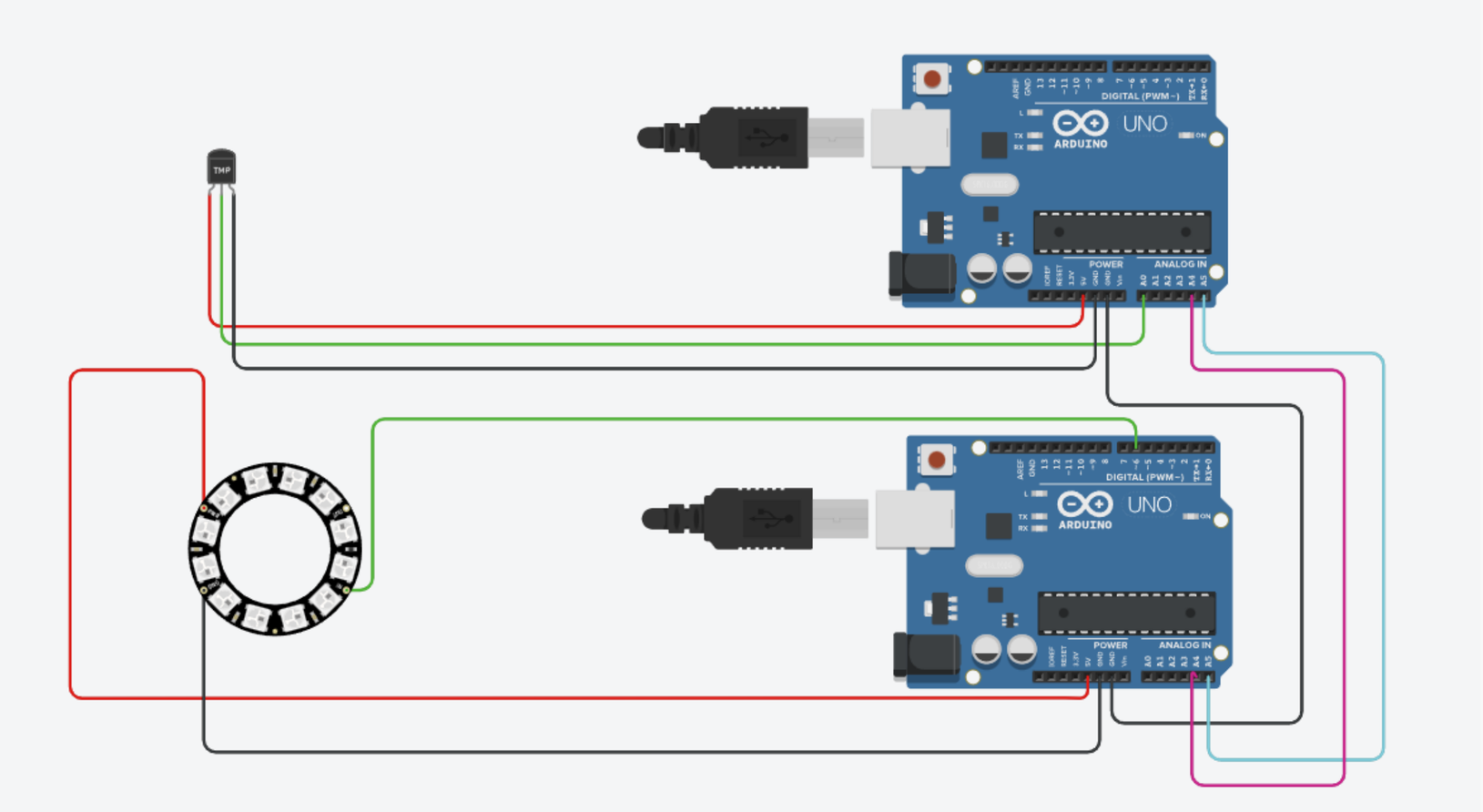
Table of Contents

I Motivation and Description

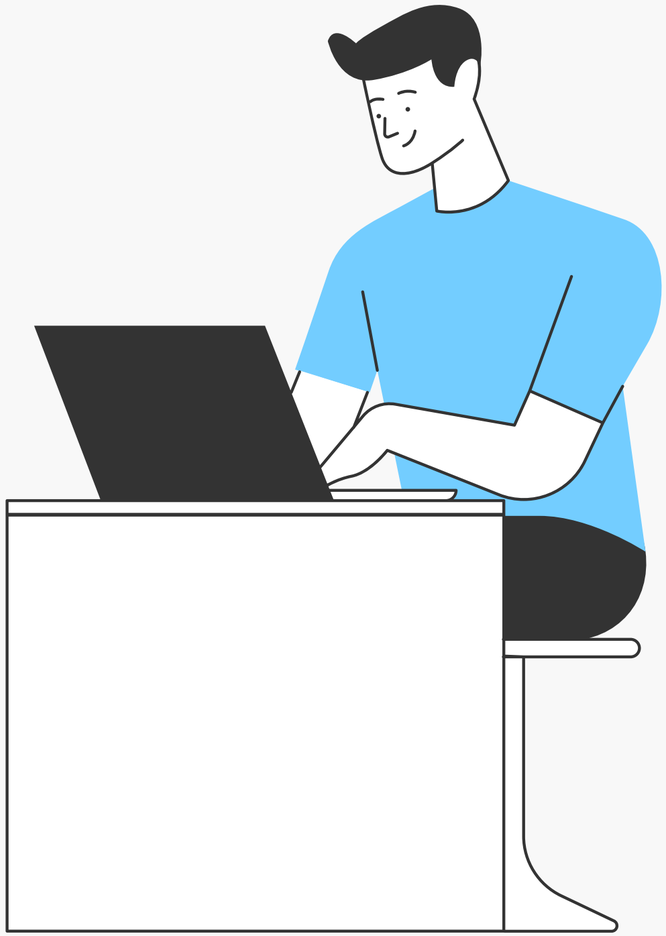
- My motivation for this project was the fact that for me personally, the light in the room I am in deeply influences the mood I am in. I always thought that an automated device which could manage the light in my home would be something helpful, so I set out to try to make such a device using Arduino.
- First thing that came to mind is translating the temperature outside to a color.



Schematic



- Two Arduino Unos are connected in order to communicate through I2C.
- The Master (below) controlls the colour of the LEDs through pin 6 while constantly requesting temperatures from the Slave (above) which in turn is reading the temperature from a sensor through A0.

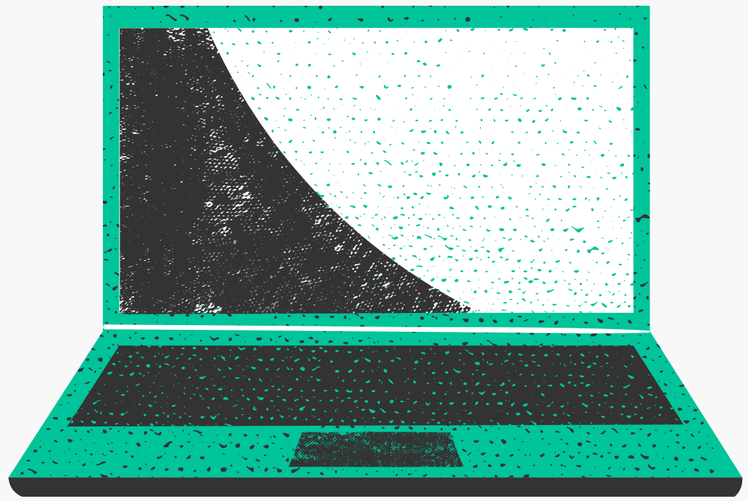


The master Arduino

- The master Arduino is the one which controls the ring of LEDs (which could be encased in a transparent matt case to enhance the effect of the light) and does so through the functions provided by the Adafruit_NeoPixel.h library. This provides some functions for easily lighting up each of the LEDs having the RGB parameters. Each of the 3 can take values from 0 to 255.
- In the setup, the pixels are initiated and the I2C communication is initiated (as master) using the Wire.h library.
- In the loop, a 7 bytes of data are requested through the Wire.requestFrom() function from the slave. Since we are dealing with floating-point numbers this is necessary. I2C cannot send these numbers at once since they are represented on multiple bytes. Finally all the read bytes are transformed from a String to a float.
- A range of values for the temperature is set: -40C to 40C. This range will be translated to values in RGB: the hottest(40C) will be pure red, while the coldest(-40C) will be pure blue. This is done in the final for loop through the pixels.setPixelColor() call.

The slave Arduino

- In the slave, first the resolution of the ADC and the resolution of the sensor are set.
- In the setup, the I2C is started as slave at address 9 and the function that handles the request made by the master is registered as requestEvent(). In this function, dtostrf() is used to convert the temperature which is is of float type to an array of characters(bytes) that can be send over I2C with Wire.write().
- In the loop, the temperature is constantly read with the help of the readTempInCelsius() function.



IV Source code

Master:

```
#include <Adafruit_NeoPixel.h>
// include I2C library
#include <Wire.h>

#define PIN 6
#define NUMPIXELS 12

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB +
NEO_KHZ800);

float readTemp = 0;
int redVal = 0;
int blueVal = 0;
float minTemp = -40.0;
float maxTemp = 40.0;
String dataString = "";

void setup() {
  pixels.begin();
  // Start i2C as master
  Wire.begin();
  Serial.begin(9600);
}

void loop(){

  String dataString = "";
  float readTemp = 0.0F;
  Wire.requestFrom(8, 7);

  while (Wire.available()) {
    char c = Wire.read();
    dataString = dataString + c;
  }

  readTemp = dataString.toFloat();
  Serial.write("I am receiving this temperature: ");
  Serial.println(readTemp);

  // set the color
  redVal = 255 / (maxTemp - minTemp) * (readTemp - minTemp);
  blueVal = 255 / (maxTemp - minTemp) * (maxTemp - readTemp);

  for(int i = 0; i < NUMPIXELS; i ++) { // For each pixel...
    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    pixels.setPixelColor(i, pixels.Color(redVal, 0, blueVal));
    pixels.show(); // Send the updated pixel colors to the hardware.
  }
  delay(500);
}
```

IV Source code

Slave:

```
// include I2C library
#include <Wire.h>

float resolutionADC = .0049 ; // default ADC resolution for the 5V
//reference = 0.049 [V] / unit
float resolutionSensor = .01 ; // sensor resolution = 0.01V/ºC

char buff[7]; // empty array where to put the numbers going to the master
float temp = 0;

void setup() {
   // put your setup code here, to run once:
 Serial.begin(9600);
 Wire.begin(8);
 Wire.onRequest(requestEvent); //Interrupt for handling incoming requests
}

void loop() {
 temp = readTempInCelsius(10, A0); // read temp. 10 times
 delay(200);
}

float readTempInCelsius(int count, int pin) {
 // read temp. count times from the analog pin
 float sumTemp = 0;
 for (int i =0; i < count; i++) {
  int reading = analogRead(pin);
  float voltage = reading * resolutionADC;
  // subtract the DC offset and converts the value in
  //degrees (C)
  float tempCelsius = (voltage - 0.5) / resolutionSensor ;
  sumTemp = sumTemp + tempCelsius; // accumulates the
 //readings
 }
 return sumTemp / (float)count; // return the average value
}


void requestEvent() {
 dtostrf(temp, 7, 2, buff);
 Serial.write("Sending temperature: ");
 Serial.println(buff);
 Wire.write(buff);
}
```

V Resources

Some of the resources that were used:

- The Lab guide, especially Lab. work no. 5.
- Documentation for various functions:

1. https://docs.particle.io/reference/device-os/api/wire-i2c/requestfrom/
2. https://docs.particle.io/reference/device-os/api/wire-i2c/onrequest/
3. https://makecode.adafruit.com/reference/light/set-pixel-color
4. https://medium.com/@sandhan.sarma/sending-floats-over-i2c-between-arduinos-part-1-4e333d8ca578

- Link to Tinkercad project:

https://www.tinkercad.com/things/f3wz1SNijEH?sharecode=4pFnPPdtTc7bqmFmse8misYlqciGCj1AbO8-EogmtFA