THC

În cadrul lecției de față ne propunem să explorăm interfețe utilizator alternative. În ciuda ecranelor din ce în mai performante există o tendință clară de a afișa informația altfel, de a distruge monotonia cifrelor și a alfabetului obișnuit – putem aici să dăm exemplu moda ceasurilor binare (imagine alăturată) dar și o serie de dispozitive exotice precum dispozitive de afișare bazate exclusiv pe ”smiley faces” sau dispozitive care indică o anumită stare de funcționare prin schimbarea culorii.

Termometru/**H**igrometru **C**olor este un sistem de măsurare a temperaturii și umidității ambientale dar care spre deosebire de sistemele clasice va indica valoarea parametrilor măsurați prin intermediul unui șir de led-uri RGB.

Există mai multe proiecte ce abordează acest subiect dar într-un mod destul de simplist:

Color Thermometer, powered by Arduino

https://youtu.be/EBrK2lqup-c

Trinket LED Thermometer

https://hackaday.io/project/3440-trinket-led-thermometer

Tutorials with Arduino: Temperature by colors.

http://arduinoarts.com/2011/08/arduino-tutorial-temperature-by-colors/

Temperature-controlled RGB LED

https://www.hackster.io/ben/temperature-controlled-rgb-led-6c8cdf

Visualizing temperature as color using an RGB led, a LM35 sensor and Arduino

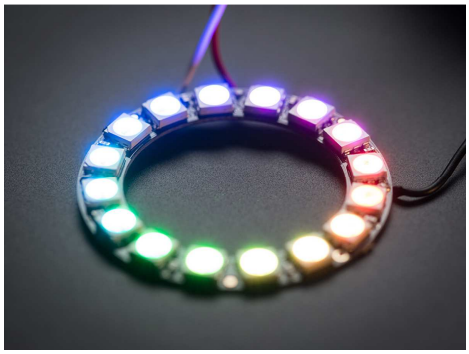
https://sjackm.wordpress.com/2012/03/26/visualizing-temperature-as-color-using-an-rgb-led-a-lm35-sensor-and-arduino/

(acest proiect a inspirat și formula de calcul utilizată în programul din această lecție)

Pentru măsurarea temperaturii și umidității vom utiliza un senzor digital I2C Si7021:

https://www.robofun.ro/senzori/vreme/enzor-umiditate-si-temperatura-si7021

Pentru afișare vom utiliza un modul Adafruit NeoPixel Ring cu 16 leduri RGB adresabile individual WS2812 5050 (bineînțeles se poate utiliza și varianta cu 24 de leduri fără a modifica schema electrică sau programul, doar numărul de leduri: constanta din program *NUMPIXELS*) – culoarea va fi dictată de temperatura măsurată iar numărul / poziția ledurilor aprinse vor fi influențate de umiditatea măsurată:



https://www.robofun.ro/electronice/led/neopixel-ring-16-x-ws2812-5050-rgb-led

În cadrul programului vom utiliza două biblioteci externe mediului Arduino IDE:

- Adafruit Si7021 – pentru a interacționa cu senzorul Si7021
  https://github.com/adafruit/Adafruit_Si7021

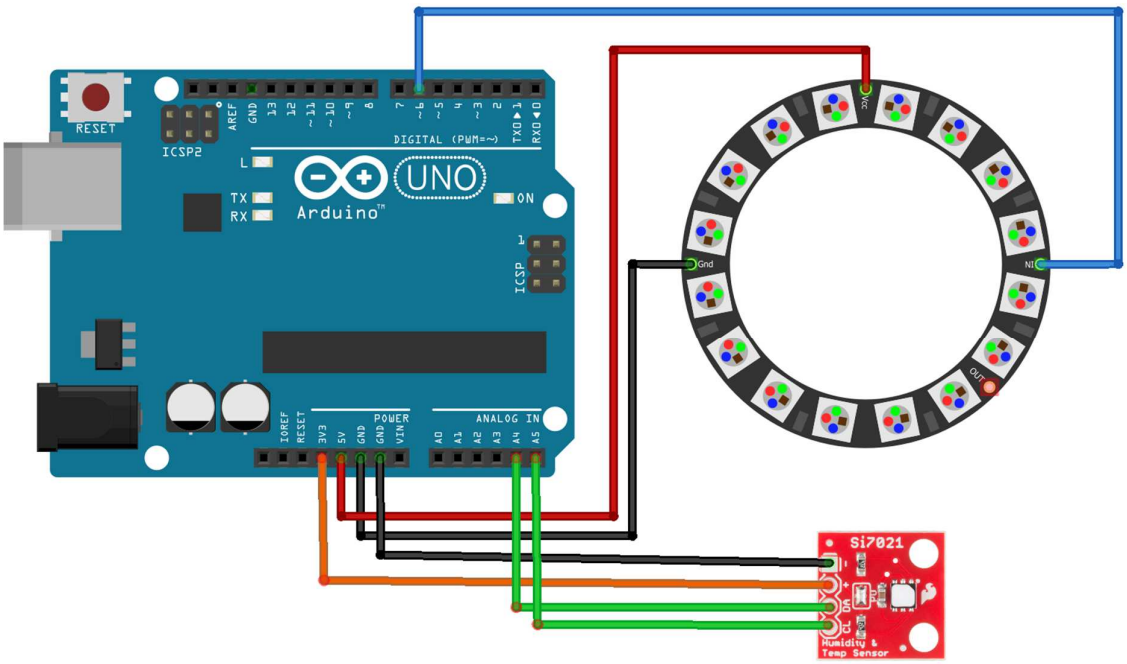
- Adafruit NeoPixel – pentru comanda ledurilor NeoPixel
  https://github.com/adafruit/Adafruit_NeoPixel

```
#include <Adafruit_Si7021.h>
#include <Adafruit_NeoPixel.h>

Adafruit_Si7021 sensor = Adafruit_Si7021();

#define PIN            6
#define NUMPIXELS      16
Adafruit_NeoPixel pixels =
Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
```

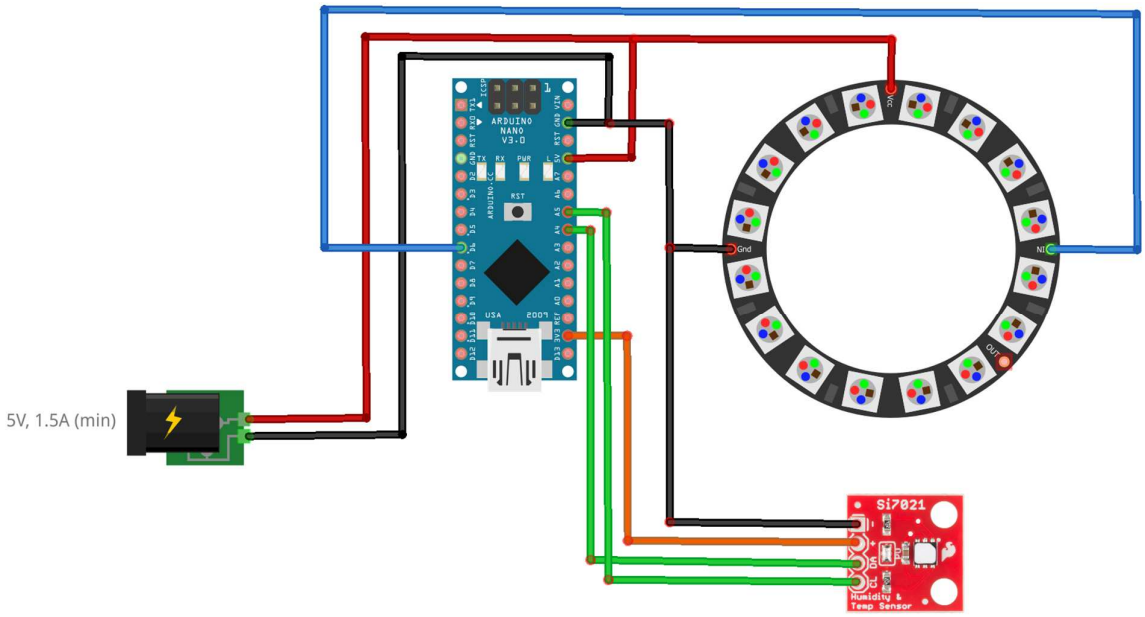
Una din problemele majore ale sistemului propus este consumul destul de mare pentru cele 16 leduri ale modulului NeoPixel – fiecare led poate ajunge la un curent maxim de 60mA adică consumul total al modulului poate ajunge la 0.06A*16=0.96A … aproape 1A. Din acest motiv vom propune două scheme electrice și două moduri de funcționare ale programului. Primul mod de funcționare (LOW_POWER) va aprinde un singur led din cele 16 – culoarea acestuia va fi dictată de temperatură iar poziția de umiditate. Acest mod de funcționare va fi semnalizat în program prin următoarea declarație:

```
#define LOW_POWER
```

și va permite alimentarea modulului NeoPixel direct din placa de dezvoltare Arduino Uno:



Pentru a lucra cu (a aprinde) mai multe leduri simultan este necesară utilizarea unei surse de alimentare externă. Prin comentarea (ștergerea) declarației de *LOW_POWER* din program, sistemul va aprinde toate ledurile până la poziția dictată de umiditate (variabila *max_pixel*) generând un efect luminos mult mai puternic. Având în vedere faptul că sistemul se pretează foarte bine la integrarea într-un alt obiect (sub o vază de sticlă sau într-un obiect decorativ transparent) cea de a doua schemă propusă (cu alimentare suplimentară externă) utilizează placa de dezvoltare Arduino Nano pentru o dimensiune mai mică a sistemului:



https://www.robofun.ro/arduino-nano

Secțiunea *setup()* se va ocupa cu inițializarea celor două obiecte de lucru *sensor* și *pixels*. Secțiunea *loop()* va realiza partea de achiziție (*sensor.read…*) și partea de comandă (*pixels.setPixelColor…*) a ledurilor RGB.

```
void setup() {
  sensor.begin();
  pixels.begin();
}

void loop() {
  float temperature, humidity;
  temperature = sensor.readTemperature();
```

```
humidity = sensor.readHumidity();
```

Poziționarea ledului aprins sau a numărului de leduri aprinse până la o anumită poziție se va face printr-o corelație umiditate (0%-100%) și numărului de leduri din modulul NeoPixel (0-NUMPIXELS).

```
byte max_pixel;
max_pixel = map(humidity, 0, 100, 0, NUMPIXELS);
```

Corelarea între temperatură și culoare ledului / ledurilor aprinse se face după următoarele reguli:

- Componenta roșie (Red) va fi aprinsă la maxim peste 35 de grade, gradual între 18 și 35, deloc sub 18 grade;
- Componenta verde (Green) va fi aprinsă crescător în intervalul 25 – 28 de grade, descrescător în intervalul 28 – 35 de grade și deloc în afara celor două intervale;
- Componenta albastră (Blue) va fi aprinsă la maxim sub 18 grade, deloc peste 25 grade iar între cele două valori va descrește gradual.

Bineînțeles, plajele de corelație pot fi modificate în funcție de temperatura ambientală a mediului în care va funcționa sistemul.

```
byte r = 0, g = 0, b = 0;

if(temperature<18) r = 0;
else if (temperature>=18)
   r = map(temperature, 18, 35, 1, 254);
else if (temperature>35) r = 255;

if(temperature<25) g = 0;
else if ((temperature>25)&&(temperature<=28))
       g = map(temperature, 25, 28, 1, 254);
else if ((temperature>28)&&(temperature<=35))
       g = map(temperature, 28, 35, 255,1);
else if (temperature>35) g = 0;
```

```
 if(temperature<18) b = 255;
 else if ((temperature>=18)&&(temperature<=25))
        b = map(temperature, 18, 25, 255, 0);
 else if (temperature>25) b = 0;

 pixels.setPixelColor(0, pixels.Color(r,g,b));
 pixels.show();
 delay(500);
 for(int i=1;i<max_pixel;i++){
    #ifdef LOW_POWER
      pixels.setPixelColor(i-1, pixels.Color(0,0,0));
    #endif
    pixels.setPixelColor(i, pixels.Color(r,g,b));
    pixels.show();
    delay(500);
  }
 delay(5000);
 for (int i=0;i<max_pixel;i++) {
    pixels.setPixelColor(i, pixels.Color(0,0,0));
  }
 pixels.show();
}
```

Programul a fost realizat și testat cu Arduino IDE 1.6.12, Arduino AVR Boards 1.6.15, Adafruit NeoPixel 1.0.6 și Adafruit Si7021 1.0.0.