

TP M2M

November 9, 2018

Thomas COUCHOUD
thomas.couchoud@etu.univ-tours.fr
Victor COLEAU
victor.coleau@etu.univ-tours.fr



Chapter 1

Prise en main

1.1 Faire clignoter une LED

Contrôler la LED ne pose pas de grand problème. Il faut juste bien penser à initialiser les différents éléments (Serial, port de la LED).

Le code utilisé est disponible en [section A.1](#).

1.2 LED RGB

Afin de pouvoir utiliser la LED RGB, nous utilisons la librairie ChainableLED. Cette dernière nous permet de créer des objets ChainableLED. Pour l'initialiser nous donnons la broche de l'horloge, la broche des données puis enfin le nombre de LEDs dans la chaîne. Par exemple si nous branchons la LED en D3, nous initialisons avec `led(3,4,1)`.

Une fois l'objet créé, il ne faut pas oublier de l'initialiser dans la méthode `init` grâce à `led.init()`.

Dans le code nous pouvons ensuite utiliser les méthodes fournies:

- `void setColorRGB(led, red, green, blue);`
- `void setColorHSB(led, hue, saturation, brightness);`

Le paramètre `led` correspond à l'indice de la LED dans la chaîne.

Concernant la boucle `loop`, cette dernière effectue les opérations suivantes:

- Récupère la valeur du potentiomètre (`analogRead`)
- Si cette valeur est plus grande qu'un seuil, on allume une des LED, sinon on l'éteint
- On map la valeur du potentiomètre entre 0 et 1 puis allumons la LED RGB avec comme intensité la valeur mappée
- On attend 20ms

Le code est disponible en [section A.2](#).

1.3 Température

De la même manière pour la température, nous utilisons la librairie DHT. Nous pouvons ensuite créer un objet DHT avec comme paramètres le port sur lequel est branché le capteur ainsi que le type du capteur utilisé.

Il faut ensuite initialiser notre objet dans la fonction `init` grâce à `dht.begin()`.

Enfin nous pouvons obtenir la température et l'humidité grâce aux fonctions `dht.readHumidity()` et `dht.readTemperature()`.

Notre code fait exactement les mêmes étapes qu'avec le potentiomètre mais map la température entre 20 et 50 degrés vers une valeur entre 0 et 255 pour contrôler la LED en RGB. Voir [section A.3](#).

1.4 LCD

Encore une fois pour utiliser l'écran LCD nous utilisons une librairie: `rgb_lcd`. L'objet à créer est un objet `rgb_lcd`. Nous l'initialisons dans le `setup` grâce à `lcd.begin(nombre de colonnes, nombre de lignes)`. Ensuite nous définissons une couleur par défaut.

Dans la fonction `loop`, nous récupérons la température et l'humidité puis l'affichons sur l'écran. Pour cela:

- On place le curseur en 0,0 grâce à `set cursor`

- On écrit "T :"
- On place le curseur en 4,0
- On écris la température
- On place le curseur en 0,1
- On écrit "H :"
- On place le curseur en 4,1
- On écris l'humidité
- On place le curseur en 15,1
- On écris "%"

De plus nous changeons la couleur de l'écran en fonction de la température et humidité grâce à `lcd.setColor(rouge, vert, bleu)`.

Code disponible en [section A.4](#).

Chapter 2

TP1

2.1 Adresse I2C

Afin de récupérer l'adresse I2C du capteur, nous utilisons le I2CScanner proposé [ici](#). Grâce à ce code, nous avons pu identifier que le baromètre a pour adresse 0x76.

2.2 Registre

- 0x0D: ID, contient l'ID du périphérique.
- 0xE0: Reset, si on écrit 0xB6, le périphérique est réinitialisé.
- 0xF2: ctrl_hum, permet de définir les options de mesure de l'humidité. Le registre devient effectif après une écriture dans ctrl_meas.
- 0xF3: status, contient 2 bits indiquant le status du périphérique.
 - Bit 3: mis à 1 quand une conversion est en cours, et 0 quand les résultats ont été transférés.
 - Bit 0: mis à 1 quand des données NVM sont copiés dans l'image du registre et 0 quand le transfert est fini.
- 0xF4 ctrl_meas: enregistre les données de capture de pression et température.
- 0xF5 config: définit des options supplémentaires.
- 0xF7...0xF9 press (_msb, _lsb, _xlsb): contient les valeurs non modifiées des mesures de pression.
- 0xFA...0xFC temp (_msb, _lsb, _xlsb): pareil mais pour la température.
- 0xFD...0xFE hum (_msb, _lsb): pareil mais pour l'humidité.

2.3 Librairie

La librairie comporte un problème, le port I2C utilisé est 0x77 et non pas 0x76. Nous avons donc du changer ce paramètre. On commence par déclarer un objet Adafruit_BME280 puis on l'initialise avec bme.begin().

Afin d'obtenir les mesures physiques, nous avons accès à:

- bme.readTemperature()
- bme.readHumidity()
- bme.readPressure()
- bme.readAltitude(SEALEVELPRESSURE_HPA) où le paramètre correspond à la pression à l'altitude 0.

Puis on affiche les données sur l'écran LCD. Le code est disponible en [section A.5](#).

Appendix A

Prise en main

A.1 Faire clignoter une LED

Listing A.1 – led.ino

```
1 #define led 13 // Constante representant le pin de la LED
2
3 // Appelée une fois au démarrage
4 void setup() {
5     Serial.begin(9600); // Def le débit de transmission de données
6     pinMode(led, OUTPUT); // Def que le pin 13 sera une sortie
7     digitalWrite(led, LOW); // Ecrit sur un pin digital (0 ou 1)
8     Serial.println("Lancement de l'app"); // Log
9 }
10
11 // Exécuter en boucle
12 void loop() {
13     digitalWrite(led, HIGH); // Allume la LED
14     Serial.println("LED allumée"); // Log
15     delay(1000); // Attend 1s
16     digitalWrite(led, LOW); // Eteind la LED
17     Serial.println("LED éteinte"); // Log
18     delay(800); // Attend 0.8s
19 }
```

A.2 LED RGB

Listing A.2 – led_rgb_pot.ino

```
1 #include <ChainableLED.h>
2
3 #define led1 13
4 #define led2 3
5 #define pot 1
6 #define To 500
7
8 int p = 0;
9 ChainableLED led(3, 4, 1);
10
11 void setup() {
12     Serial.begin(9600);
13     pinMode(led1, OUTPUT);
14     pinMode(pot, INPUT);
15     led.init();
16 }
17
18 void loop() {
19     p = analogRead(pot);
20     if(p > To){
21         digitalWrite(led1, HIGH);
```

```
22 }
23 else{
24     digitalWrite(led1, LOW);
25 }
26 float f = p/1023.0;
27 led2.setColorHSB(0, 0.5, 1, f);
28 Serial.println(f);
29 delay(100);
30 }
```

A.3 Température

Listing A.3 – led_rgb_temp.ino

```
1 #include <ChainableLED.h>
2 #include <DHT.h>
3
4 #define led1 13
5 #define led2 3
6 #define pot A0
7 #define To 500
8
9 int p = 0;
10 ChainableLED led(3, 4, 1);
11 DHT dht(pot, DHT22);
12
13 void setup() {
14     Serial.begin(9600);
15     pinMode(led1, OUTPUT);
16     pinMode(pot, INPUT);
17     led.init();
18     dht.begin();
19 }
20
21 void loop() {
22     float humidity = dht.readHumidity();
23     float temp = dht.readTemperature();
24     p = temp;
25     Serial.print("H : ");
26     Serial.print(humidity);
27     Serial.print("    T : ");
28     Serial.println(temp);
29     if(p > To){
30         digitalWrite(led1, HIGH);
31     }
32     else{
33         digitalWrite(led1, LOW);
34     }
35     float f = 255*(p-20.0)/(50.0-20.0);
36
37     led.setColorRGB(0, f, 255-f, humidity / 100.0 * 255.0);
38     Serial.println(f);
39     delay(100);
40 }
```

A.4 LCD

Listing A.4 – lcd.ino

```
1 #include <DHT.h>
2 #include <Wire.h>
3 #include "rgb_lcd.h"
4
5 rgb_lcd lcd;
```

```
6
7 #define pot A0
8
9 int p = 0;
10 DHT dht(pot, DHT22);
11
12 void setup() {
13     Serial.begin(9600);
14     dht.begin();
15     lcd.begin(16, 2);
16     lcd.setRGB(255, 255, 255);
17 }
18
19 void loop() {
20     float humidity = dht.readHumidity();
21     float temp = dht.readTemperature();
22     p = temp;
23     Serial.print("H : ");
24     Serial.print(humidity);
25     Serial.print("    T : ");
26     Serial.println(temp);
27     float f = 255*(p-20.0)/(50.0-20.0);
28
29     lcd.setRGB(f, 255-f, humidity / 100.0 * 255.0);
30
31     lcd.setCursor(0, 0);
32     lcd.print("T : ");
33     lcd.setCursor(4, 0);
34     lcd.print(temp);
35     lcd.setCursor(0, 1);
36     lcd.print("H : ");
37     lcd.setCursor(4, 1);
38     lcd.print(humidity);
39     lcd.setCursor(10, 1);
40     lcd.print("%");
41
42     delay(100);
43 }
```

A.5 Baromètre

Listing A.5 – bme.ino

```
1 #include <Wire.h>
2 #include <Adafruit_Sensor.h>
3 #include <Adafruit_BME280.h>
4 #include "rgb_lcd.h"
5
6 #define SEALEVELPRESSURE_HPA (1013.25)
7
8 Adafruit_BME280 bme; // I2C
9 rgb_lcd lcd;
10
11 unsigned long delayTime;
12
13 void setup() {
14     Serial.begin(9600);
15     bme.begin();
16     lcd.begin(16, 2);
17     lcd.setRGB(255, 255, 255);
18     delayTime = 1000;
19 }
20
21
22 void loop() {
23
24     float t = bme.readTemperature();
```

```
25 Serial.print("Temperature = ");
26 Serial.print(t);
27 Serial.println(" *C");
28
29 float h = bme.readHumidity();
30 Serial.print("Humidity = ");
31 Serial.print(h);
32 Serial.println(" %");
33
34 float p = bme.readPressure() / 100.0F;
35 Serial.print("Pressure = ");
36 Serial.print(p);
37 Serial.println(" hPa");
38
39 float a = bme.readAltitude(SEALEVELPRESSURE_HPA);
40 Serial.print("Approx. Altitude = ");
41 Serial.print(a);
42 Serial.println(" m");
43
44 Serial.println();
45
46 float f = 255*(t-20.0)/(50.0-20.0);
47
48 lcd.setRGB(f, 255-f, h / 100.0 * 255.0);
49
50 lcd.setCursor(0, 0);
51 lcd.print("T:");
52 lcd.setCursor(2, 0);
53 lcd.print(t);
54
55 lcd.setCursor(8, 0);
56 lcd.print("P:");
57 lcd.setCursor(10, 0);
58 lcd.print(p);
59
60 lcd.setCursor(0, 1);
61 lcd.print("H:");
62 lcd.setCursor(2, 1);
63 lcd.print(h);
64
65 lcd.setCursor(8, 1);
66 lcd.print("A:");
67 lcd.setCursor(10, 1);
68 lcd.print(a);
69
70 delay(delayTime);
71 }
```