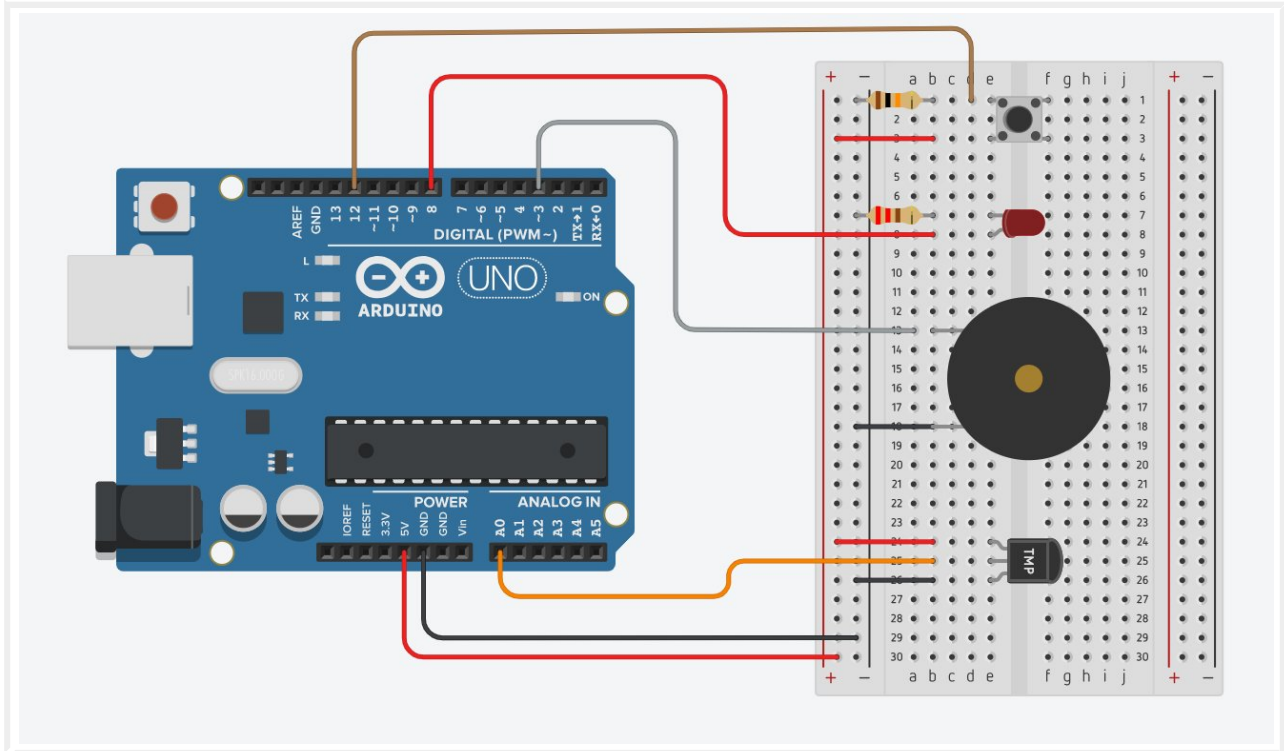


Températures – Alarme

(Alarme sonore par dépassement de température)



. Liste des composants

- . 1 capteur de température (TMP 36 ou LM 35)
- . 1 DEL rouge
- . 1 résistance de 220 Ω (résistance de protection de la DEL)
- . 1 résistance de 10 k Ω (résistance du circuit du bouton poussoir)
- . 1 bouton poussoir
- . 1 haut-parleur (ou piezo)
- . 1 plaque d'essais
- . Fils de connexion

. Objectif

L'objectif de cette activité est de réaliser une alarme sonore et visuelle qui se déclenchera lorsque la température mesurée par le capteur TMP 36 ou LM 35 du circuit d'étude est supérieure à une valeur seuil à définir, permettant, par exemple, de prévenir un utilisateur du dépassement de la température d'utilisation d'un matériel.

Il suffit pour cela de reprendre le programme de l'activité de mesure d'une température avec un capteur TMP 36 ou LM 35, auquel on ajoute le code de l'alarme sonore et visuelle.

Les mesures de température commencent après un appui sur le bouton poussoir et sont arrêtées en appuyant de nouveau sur celui-ci.

. Le programme

Le code de l'activité pourra être modifié pour voir l'influence des variables (seuil de température, fréquence de l'onde sonore, durée d'émission, durée de silence).

Temperatures_alarme

```
// Déclaration des constantes et variables
```

```
const int PinSensor=0;
const int PinButton= 12;
const int PinLed = 8;
const int PinTone = 3;
```

```
int ValSensor = 0;
float tension = 0.0;
float Temp = 0.0;
float OldTemp = 0.0;
float TempAlarme = 25.0;
```

```
int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
```

```
// Initialisation des entrées et sorties
```

```
void setup() {
  Serial.begin(9600);
  pinMode(PinButton, INPUT);
  pinMode(PinLed, OUTPUT);
  Serial.println("Appuyez sur le bouton poussoir pour commencer les mesures.");
}
```

```

// Fonction principale en boucle

void loop() {
    ValButton = digitalRead(PinButton);
    delay(10);

    if ((ValButton == HIGH) && (OldValButton == LOW))
    {
        State=1-State;
    }
    OldValButton = ValButton;

    if (State==1)
    {
        if (OldState == 0)
        {
            Serial.println("Mesure de la temperature en cours.");
            Serial.println("");
            Serial.println ("Temperature en degre Celsius:");
            OldState=1;
        }
        ValSensor = analogRead(PinSensor);
        tension = (ValSensor/1023.0)*5.0;

        // Capteur TMP 36
        Temp = (tension - 0.5) * 100;

        // Capteur LM 35
        //Temp = tension * 100;

        if (OldTemp != Temp)
        {
            Serial.println(Temp,1);
            OldTemp = Temp;
        }
        if (Temp > TempAlarme)
        {
            digitalWrite(PinLed, HIGH);
            tone(PinTone,440);
            delay(100);
            digitalWrite(PinLed, LOW);
            noTone(PinTone);
            delay(100);
        }
        else
        {
            digitalWrite(PinLed, LOW);
            noTone(PinTone);
        }
        delay(100);
    }
    else
    {
        if (OldState == 1){
            Serial.println("Fin des mesures.");
            OldState = 0;}
    }
}

```

Déroulement du programme :

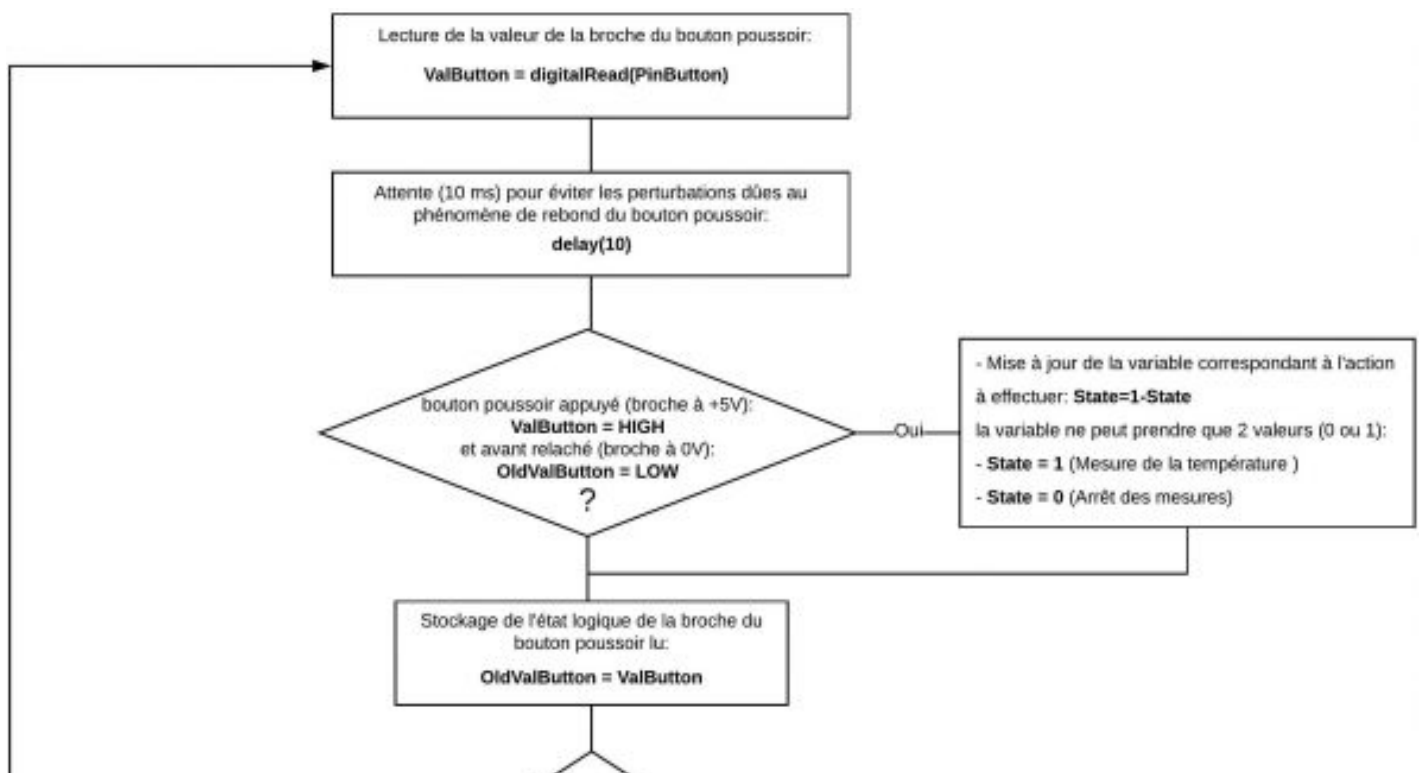
– Déclaration des constantes et variables :

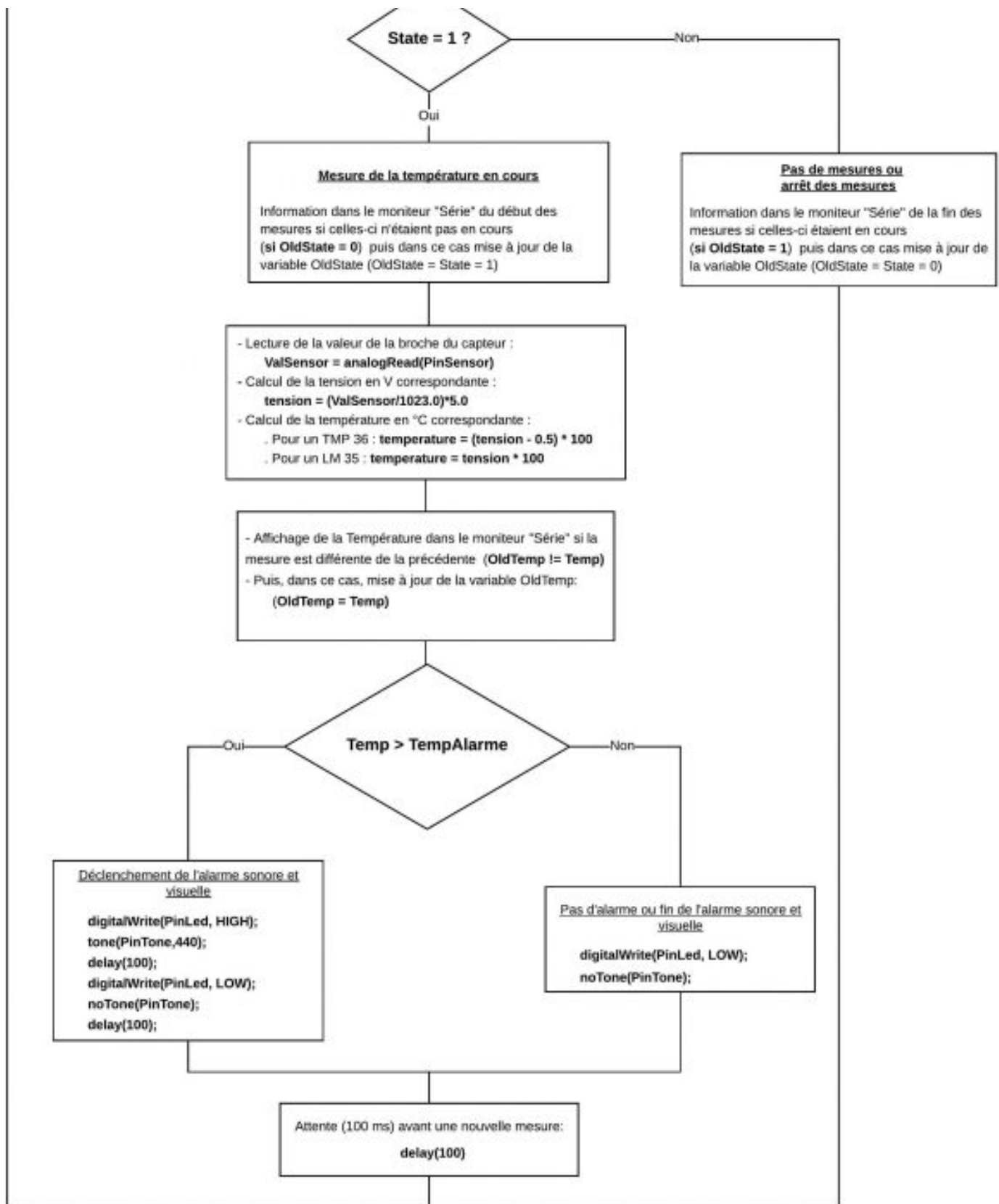
- N° de la broche correspondant au bouton poussoir: **const int PinButton = 12**
- N° de la broche correspondant à la DEL : **const int PinLed = 8**
- N° de la broche correspondant au piezo : **const int PinTone = 3**
- N° de la broche correspondant au capteur de température : **const int PinSensor = 0**
- Variable pour stocker la valeur de la broche du bouton poussoir: **int ValButton = 0**
- Variable pour stocker l'ancienne valeur de la broche du bouton poussoir: **int OldValButton = 0**
- Variable correspondant à l'action à effectuer: **int State = 0**
- Variable pour stocker l'ancienne valeur de la variable correspondant à l'action à effectuer: **int OldState = 0**
- Variable pour stocker la valeur de la broche du capteur de température: **int ValSensor = 0**
- Variable pour stocker la valeur en V de la tension correspondante à la valeur de la broche du capteur : **float tension = 0.0**
- Variable correspondant à la température calculée à partir de la valeur de la broche du capteur: **float Temp = 0.0**
- Variable correspondant à la température mesurée précédemment: **float OldTemp = 0.0**
- Variable correspondant à la température du seuil de déclenchement de l'alarme: **float TempAlarme = 25.0**

– Initialisation des entrées et sorties :

- le débit de communication en nombre de caractères par seconde pour la communication série est fixé à 9600 bauds: **Serial.begin(9600)**
- La broche du bouton poussoir est initialisée comme une entrée digitale. Des données seront donc envoyées depuis cette broche vers le microcontrôleur: **pinMode (PinButton, INPUT)**
- La broche de la DEL rouge est initialisée comme une sortie digitale. Des données seront donc envoyées depuis le microcontrôleur vers cette broche : **pinMode (PinLed, OUTPUT)**

– Fonction principale en boucle :





Résultats dans le moniteur série :

