




Régulation de Température par Arduino

Notre projet consiste à réaliser un système de refroidissement , par régulation de température en utilisant Arduino.

 Difficulté Moyen

 Durée 1 mois

 Catégories Électronique

 Coût 50 EUR (€)

Sommaire

Introduction

Video d'introduction

Étape 1 - Description du procédé

Étape 2 - Construction du plan de travail

Étape 3 - Connexion des outils utilisés à la carte Arduino et Réalisation du système

Étape 4 - Code Arduino

Commentaires

Introduction

Porteurs du projet: Mohamed-Ouassim BEHLOUL , Sofia SAHRANE , Katia MEDJBER, Saoussene KITOUNI , Katia MEHDI .

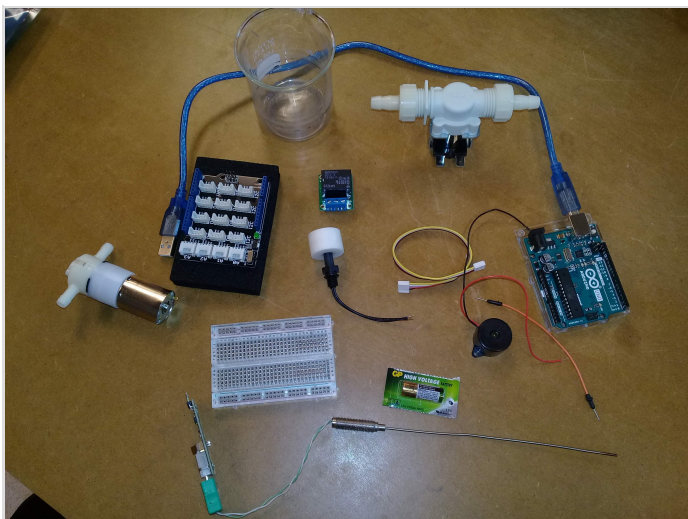
Encadré par M. Shayan TABIBIAN

Ce projet a été développé dans le cadre de relier le domaine électronique et chimique, intégré dans l'unité d'enseignement 5C803 (Optimisation et contrôle des procédés) en Ingénierie Chimique.

Responsable d'UE : M. Jerome PULPYTEL.

Il s'agit de réguler la température et le niveau d'eau par Arduino.

Ce système est souvent utilisé à l'échelle industrielle, dans les réacteurs à haute température pour éviter le risque d'emballement thermique.



Matériaux

Outils

Pour notre projet, nous avons utilisé :

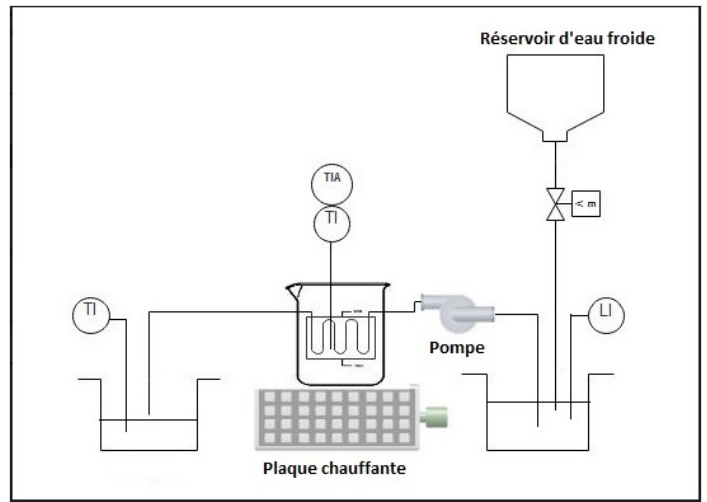
- Un bécher
- une pile 12 v
- carte arduino
- pompe miniature pour eau
- Sonde
- Buzzer
- Électrovanne
- Relay
- Capteur de niveau
- fils de connexion
- Bred Board
- Tuyaux
- Plaque chauffante
- Câble

Étape 1 - Description du procédé

On chauffe de l'eau dans un bécher jusqu'à une température de 50 °C.

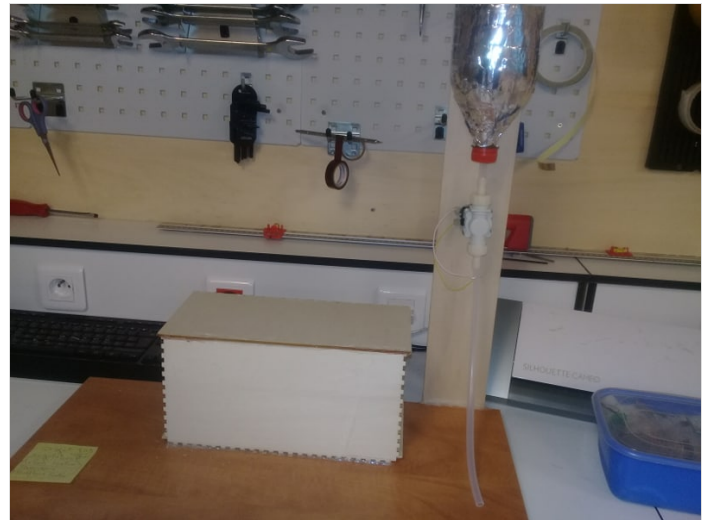
A l'instant où la température dépasse les 50 °C, le buzzer déclenche, et donne l'ordre à la pompe d'aspirer l'eau froide, afin de la faire circuler dans le tuyau qui est autour du bécher, sous forme de serpentín, qui sert à refroidir l'eau chaude.

La pompe aspire l'eau froide à partir d'un bac, qui sera constitué d'un capteur de niveau, et relié à une électrovanne. Quand le niveau d'eau dans le bac diminue, le buzzer déclenche et donne l'ordre à l'électrovanne de s'ouvrir, pour ajuster le niveau d'eau afin d'éviter la cavitation de la pompe et sa défaillance.



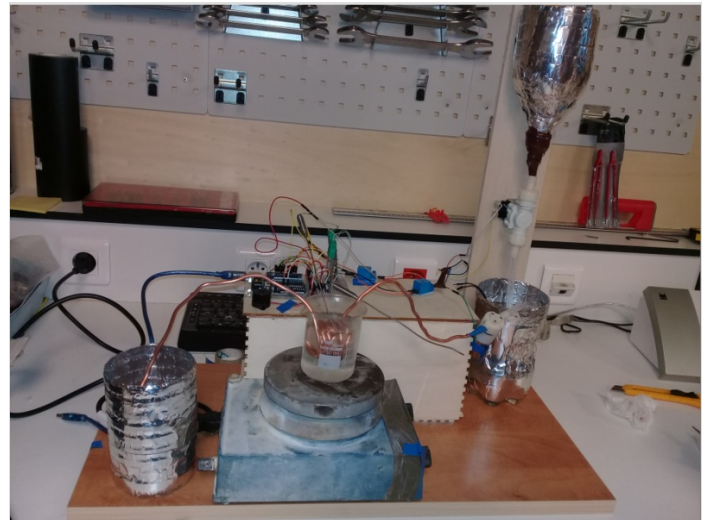
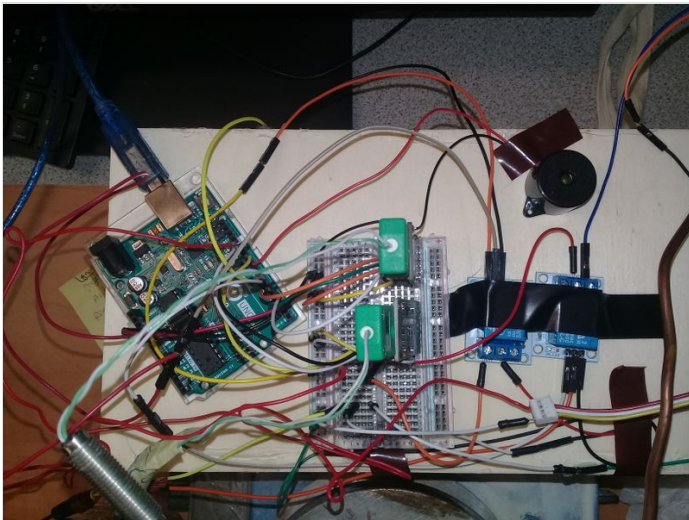
Étape 2 - Construction du plan de travail

À l'aide d'une découpeuse laser, on découpe 4 plaques de bois pour construire un boîtier qui nous sert de support pour la carte Arduino et la Bred Board, auxquels sont reliés tous les autres outils utilisés dans le projet, et cela pour être au niveau avec le bécher quand celui-là est mis sur la plaque chauffante.



Étape 3 - Connexion des outils utilisés à la carte Arduino et Réalisation du système

On relie chaque instrument (électrovanne, pompe, alarme, sondes de température, relay), à la carte Arduino, via des fils de connexion.



Étape 4 - Code Arduino

```
// Code Echangeur thermique!  
#include "Adafruit_MAX31855.h"  
// Pin de controle de la pompe
```

```

#define POMPE 7
// Pin de l'electrovanne
#define EV 6
// Pin du capteur de niveau
#define LL 13
// Pins du capteurs de température du reacteur
int T_React_DO = 5;
int T_React_CS = 4;
int T_React_CLK = 3;
// Pins du capteurs de température du liquide de refroidissement !
int T_froid_DO = 12;
int T_froid_CS = 11;
int T_froid_CLK = 10;
const int buzzer = 9; // Buzzer vers arduino Pin 9
int cpt = 0;
const int NombreDeBip = 5;
// température de déclenchement de l'alarme !
const double T_Limit = 55;
// température de fonction du reacteur
// valeur a maintenir par régulation
const double T_stable = 50;
// Constante d'amortissement de la régulation
const double Delta = 5;
// Capteur de niveau
int liquidLevel = 0;
int cntLevel = 0;
int LimitCntLevel = 10;
Adafruit_MAX31855 TC_react(T_React_CLK, T_React_CS, T_React_DO);
Adafruit_MAX31855 TC_froid(T_froid_CLK, T_froid_CS, T_froid_DO);
void buzzerTone() {
while (cpt < NombreDeBip) {
tone(buzzer, 1000);
delay(1000);
noTone(buzzer);
delay(1000);
cpt++;
}
cpt = 0;
}
void buzzerTemp(double temp) {
if (temp > T_Limit) {
tone(buzzer, 1000);
delay(500);
noTone(buzzer);
delay(500);
}
}
// fonction de régulation de la température
void TR_control(double temp_react, double temp_froid) {
// Déclenche la pompe si la température dépasse le seuil de stabilité
if (temp_react > T_stable){
tone(buzzer, 1000);
digitalWrite(POMPE, HIGH);
digitalWrite(EV, HIGH);
delay(1000);
noTone(buzzer);
}
// Arrête la pompe si on revient à une température stable - Delta
// le Delta correspond à la valeur d'amortissement de la température
else if((temp_react - Delta) <= T_stable){
digitalWrite(POMPE, LOW);
digitalWrite(EV, LOW);
noTone(buzzer);
}
}

```

```

}
}
// command de l'electrovanne
void EV_control(int level) {
if (liquidLevel == HIGH && cntLevel >= LimitCntLevel) {
// Open ElectroVanne
cntLevel++;
if (cntLevel >= LimitCntLevel){
digitalWrite(EV, HIGH);
}
}
else {
// Close ElectroVanne
cntLevel = 0;
digitalWrite(EV, LOW);
}
}
void setup() {
pinMode(POMPE, OUTPUT); //Set Pompe as an output//
pinMode(EV, OUTPUT); //Set ElectroVanne as an output//
pinMode(LL, INPUT); //Set Liquid sensor as an input//
Serial.begin(9600);
Serial.println("MAX31855 test");
// Attendre que le circuit MAX se stabilise.
delay(500);
}
void loop() {
// Affichage de la température courante de la température du reacteur
// Serial.print("Internal Reactor Temp = ");
// Serial.println(TC_react.readInternal());
// Lecture en degrés Celcius de la température du reacteur
double c_react = TC_react.readCelsius();
if (isnan(c_react)) {
Serial.println("Quelque chose ne fonctionne pas avec le TC_react!");
} else {
Serial.print("C_react = ");
Serial.println(c_react);
}
// Affichage de la température courante de la température de l'eau de refroidissement
// Serial.print("Cooling Fluid Temp = ");
// Serial.println(TC_froid.readInternal());
double c_froid = TC_froid.readCelsius();
if (isnan(c_froid)) {
Serial.println("Quelque chose ne fonctionne pas avec le TC_froid!");
} else {
Serial.print("C_froid = ");
Serial.println(c_froid);
}
// Declanche l'alarme si la température dépasse le seuil critique!
buzzerTemp(c_react);
// déclenchement de la pompe pour la régulation de la température du reacteur
TR_control(c_react, c_froid);
// Remplissage bac du liquide de refroidissement
// en fonction du capteur de niveau
// liquidLevel = digitalRead(LL);
// Serial.print("Level : ");
// Serial.println(liquidLevel);
// EV_control(liquidLevel);
// Attendre une 0.1 seconde
delay(100);
}

```

