الجمهورية الجزائرية الديمقراطية الشعبية

République algérienne démocratique et populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique
جامعة عين تموشنت بلحاج بوشعيب
Université –Ain Temouchent- Belhadj Bouchaib
Faculté des Sciences et de Technologie
Département d'Electronique et des Télécommunications

Projet de Fin d'Etudes
Pour l'obtention du diplôme de Master en : Instrumentation
Domaine : SCIENCES ET TECHNOLOGIES
Filière : Electronique
Spécialité : Instrumentation
Thème

Dispositif de Surveillance de Température basé sur une carte Arduino UNO

Système de contrôle et de régulation thermique pour couveuses avicoles

Présenté Par :

- BOUZOUINA Boualem
- MEHIAOUI Chaimaa

Devant le jury composé de :

Dr. FEROUANI Souheyla	MCA	UAT.B. B	Présidente
Dr. BADIR Houaria	MAA	UAT.B. B	Examinatrice
Pr. AYACHE Choukria	Pr	UAT.B. B	Encadrante

Année Universitaire 2025/2026

Remerciement

*Tout d'abord je tiens à remercier **ALLAH** le tout puissant de m'avoir donné la santé, la volonté, le courage et la patience pour mener à terme ma formation et pourvoir réaliser ce travail de recherche.*

*Je tiens à remercier, très sincèrement, le Professeur **AYACHE Choukria** notre directrice de mémoire, pour son encadrement de qualité, sa motivation professionnelle, de ses précieux conseils et sa patience ainsi pour le temps qu'il a consacré à la réalisation de ce travail. Nous n'oublierons jamais votre disponibilité et votre réactivité face aux problèmes rencontrés au cœur de ce travail. Soyer assurée de notre profond respect et de notre sincère estime pour votre soutien personnel et vos conseils.*

*Nous adressons aussi toute notre gratitude et respect envers le président de jury **Dr FEROUANI Souheyla** et l'examinatrice **Dr BADIR Houaria** qui nous ont honorés en acceptant de juger notre travail.*

Nous n'oublierons pas de remercier vivement les enseignants du département qui ont assuré notre formation.

Enfin, nous remerciements s'adressent aussi à nos familles et nos amis.

De peur d'en avoir oublié, je souhaite remercier tous ceux qui ont contribué de près ou de loin à l'élaboration de ce mémoire ainsi qu'à la réussite de ce parcours universitaire.

Dédicace

D'abord je remercie le mon Dieu qui m'a donné le courage pour arriver à la fin de mes études.

Je dédie ce modeste travail à :

Mon père et ma mère que dieu les garde et les protège pour leurs soutien moral et financier, pour leurs encouragements et les sacrifices qu'ils ont endurés.

Mes chers frères.

Ma chère sœur.

MEHIAOUI Chaïmaa

Dédicace

Je dédie ce modeste travail :

À mes chers parents,Pour leur amour, leurs sacrifices et leur soutien inestimable tout au long de mon parcours.

À ma famille et mes proches,

Pour leurs encouragements permanents et leur confiance.

À tous mes enseignants et encadreurs,

Pour leurs conseils précieux et leur accompagnement.

Une dédicace spéciale à mon cher ami, ***khoualef Adam***

Qui m'a énormément aidé et soutenu durant la réalisation de ce projet.

Je le remercie sincèrement pour son aide, sa patience et ses encouragements.

Enfin, à tous ceux qui ont contribué de près ou de loin à l'élaboration de ce mémoire

BOUZOUINA Boualem

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

Avec l'évolution des systèmes embarqués et de l'électronique intelligente, la surveillance et le contrôle des paramètres physiques, notamment la température, sont devenus essentiels dans de nombreux domaines tels que l'industrie, l'agriculture, l'élevage et les systèmes domestiques. Dans ce cadre, les systèmes basés sur des microcontrôleurs offrent des solutions simples, fiables et peu coûteuses pour assurer la mesure et la régulation automatique de la température.

Parmi ces solutions, l'utilisation de la carte Arduino UNO associée à des capteurs comme le LM35 permet de concevoir des dispositifs efficaces et faciles à programmer.

Ce projet s'inscrit donc dans une démarche de conception d'un système de contrôle de température intelligent, capable d'assurer une surveillance continue et une réaction automatique en cas de dépassement des seuils.

Dans de nombreux environnements (incubateurs, systèmes de chauffage, laboratoires), une mauvaise gestion de la température peut entraîner des pertes matérielles ou biologiques importantes. La problématique posée est donc la suivante :

Comment concevoir un système simple, fiable et économique capable de surveiller la température en temps réel et d'assurer une action automatique en cas de dépassement d'un seuil critique ?

Ce projet vise à implémenter un contrôleur de température basée sur une carte Arduino-Uno et une électronique capable d'effectuer la mesure et la surveillance de la température et de l'afficher sur un écran LCD.

Ce mémoire est structuré en trois chapitres principaux :

- Ce chapitre présente le contexte général du projet ainsi que les notions de base sur les systèmes de mesure de température
- Le deuxième chapitre est consacré à l'étude et à la description des différents composants utilisés ainsi que le schéma bloc et le fonctionnement global du système.
- Le dernier chapitre à la mise en œuvre pratique du projet : programmation sous Arduino IDE, simulation sous Proteus, tandis que la seconde partie est consacrée à la réalisation pratique de notre dispositif.

Enfin, une conclusion générale qui synthétise les travaux effectués dans le cadre de ce projet en énonçant les perspectives à développer.

Chapitre I :
Capteurs de Température

I .1 Introduction :

La mesure de la température constitue un élément fondamental dans de nombreux domaines scientifiques, industriels et domestiques. Elle joue un rôle essentiel dans le contrôle des procédés industriels, la conservation des produits alimentaires, la surveillance médicale et la gestion énergétique des systèmes thermiques. En effet, une variation non contrôlée de la température peut entraîner des conséquences graves, telles que la dégradation des matériaux, la perte de qualité des produits ou encore des risques pour la sécurité des installations.

Avec l'évolution des technologies embarquées, notamment les systèmes à base de microcontrôleurs comme Arduino, il est désormais possible de concevoir des dispositifs de mesure et de contrôle de température simples, fiables et à faible coût. Ces systèmes reposent sur l'utilisation de capteurs capables de convertir une grandeur thermique en un signal exploitable par un système électronique.

Dans ce chapitre, nous présentons les notions fondamentales liées à la température, les différentes unités de mesure ainsi que les principaux types de capteurs utilisés pour sa mesure.

I.2 Notions de base sur la température :

La température est une grandeur physique qui permet de caractériser l'état thermique d'un système. Elle est directement liée à l'énergie cinétique moyenne des particules qui composent la matière (atomes ou molécules). Ainsi, plus l'agitation thermique des particules sont élevée, plus la température du corps est importante.

D'un point de vue thermodynamique, la température est une variable d'état qui détermine le sens des échanges de chaleur entre deux systèmes. La chaleur se transfère toujours spontanément du corps le plus chaud vers le corps le plus froid jusqu'à atteindre un état d'équilibre thermique. [1]

La mesure de la température repose sur l'exploitation de certaines propriétés physiques des matériaux qui varient en fonction de la température. Parmi ces propriétés, on peut citer : [2]

- La variation de la résistance électrique (RTD, thermistances)
- La génération de tension électrique (thermocouples)
- La variation de tension analogique (capteurs LM35, TMP36)

Un autre concept fondamental est celui de l'équilibre thermique. Pour obtenir une mesure précise, le capteur doit être en équilibre thermique avec le milieu à mesurer. Toute perturbation externe peut entraîner une erreur de mesure. [3]

Enfin, la qualité de la mesure dépend de plusieurs paramètres tels que la précision, la sensibilité, la résolution et le temps de réponse du capteur.

I.3 Unités et échelles de température :

Une méthode fiable et polyvalente pour identifier les différences de température corporelle ou de facteurs atmosphériques. L'échelle se compose d'unités. Cinq échelles ont été identifiées dans le monde, dont Fahrenheit, Celsius, Kelvin, Rankine et Réaumur. [4]

- **Fahrenheit (°F):** Daniel Gabriel Fahrenheit, physicien allemand qui a construit le premier thermomètre à mercure pratique. Il a utilisé un mélange de glace, d'eau et de sel marin qu'il estime être à 0°F et utilise la formule pour la température corporelle de 96°F. Après avoir divisé l'intervalle en 96 parties, il a trouvé que l'eau gelée est à 32°F et que l'eau bouillante est à 212°F.
- **Celsius (°C) :** L'astronome et physicien suédois Andres Celsius a construit son thermomètre à mercure en 1742. Il choisit de faire fondre de la glace à 0°C et de faire bouillir de l'eau à 100°C. 3. Kelvin (°K) : Sir William Thomson, Lord Kelvin a proposé une échelle absolue utilisant le zéro absolu (= -273K) comme température d'origine. Il reprend le même intervalle de tick au-dessus de Celsius (0°C = 273°K). Donc si l'eau gèle à 0°C, cela équivaut à geler à 273°K.
- **Rankine (°R ou °RA) :** En 1859 L'Écossais William Rankine a introduit pour la première fois l'échelle de Rankine. Comme l'échelle Kelvin, le point de référence de l'échelle Rankin est le zéro absolu à 0°R. Rankin a la même taille que Fahrenheit, mais zéro est très différent. Le point de congélation de l'eau est égal à 491,67° Rankine.
- **Réaumur (°Ré) :** En 1730, René de Réaumur introduit la gamme Réaumur. Ses points de référence sont le point de congélation de l'eau, 0°Re, et le point d'ébullition de l'eau,
  80°Re.

Le **tableau (I.1)** ci-dessous illustre la formule pour convertir les lectures de température d'une unité à une autre.

Tableau I.1:Formule de calcul pour la conversion entre unités de température. [5]

De/En	En °C	En °F	En °K	En °Ra	En °Ré
De °C	1	$T_{(°C)} \times 1.8 + 32$	$T_{(°C)} + 273.15$	$(T_{(°C)}+273.15)\times 1.8$	$T(°C) \times 0.8$
De °F	$(T_{(°F)} - 32)/1.8$	1	$(T_{(°F)} 459.67)/1.8$	$T_{(°F)} + 459.67$	$(T_{(°F)}-32)\times 4/9$
De °K	$T(°K) - 273.15$	$T_{(K)} \times 1.8 – 459.67$	1	$T_{(K)} \times 1.8$	$(T_{(k)}- 273.15)\times 0.8$
De °Ra	$(T_{(°Ra)} - 491.67)/1.8$	$T_{(°Ra)} – 459.67$	$T(°Ra) / 1.8$	1	$(T_{(°Ra)}- 491.67)\times 4/9$
De °Ré	$T_{(Ré)} / 0.8$	$T_{(°Ré)} \times 9/4 + 32$	$T_{(°Ré)}\times 1.25+273. 15$	$T_{(°Ré)}\times 9/4 +273.15$	1

I.4 Capteurs de température :

Les capteurs de température sont des dispositifs qui permettent de convertir la température en un signal exploitable (électrique ou numérique). Ils sont essentiels dans les systèmes de mesure et de contrôle.

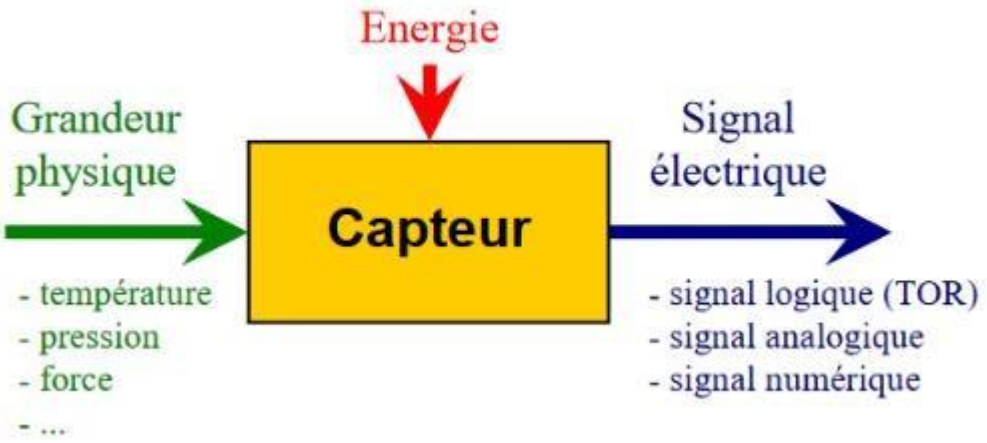

Figure I.1:Schéma d'un capteur. [6]

I.4.1 Thermistances :

Un capteur à thermistance est un corps de détection de température en matériau semiconducteur, qui se caractérise par une grande variation de résistance proportionnel à une petite variation de température. Les thermistances ont un coefficient de température négatif (également appelé thermistance CTN), ce qui signifie que la résistance de la thermistance réduit à mesure que la température augmente. En plus, les thermistances ont un coefficient de température positif (également appelé thermistance CTP). [7]

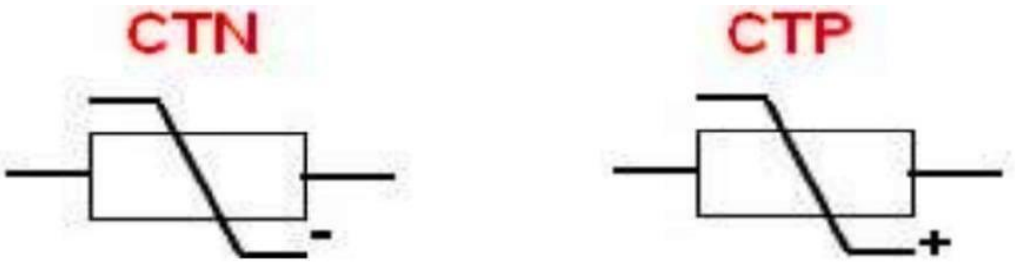

Figure I.2 : Représentation schématique. [8]

I.4.2 Capteurs analogiques :

Un capteur analogique de température est un dispositif qui mesure la température et la convertit en un signal électrique continu, généralement une tension, proportionnel à la valeur de la température mesurée. [9]

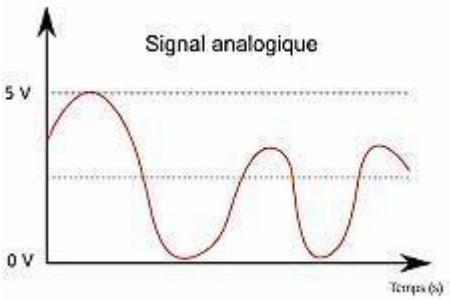

Figure I.3 : Signal analogique. [10]

I.4.3 Capteurs numériques :

Un capteur numérique de température est un dispositif qui mesure la température et la transforme directement en une donnée numérique, transmise sous forme de signaux digitaux (bits), facilitant son traitement par des systèmes électroniques tels que les microcontrôleurs. [11]

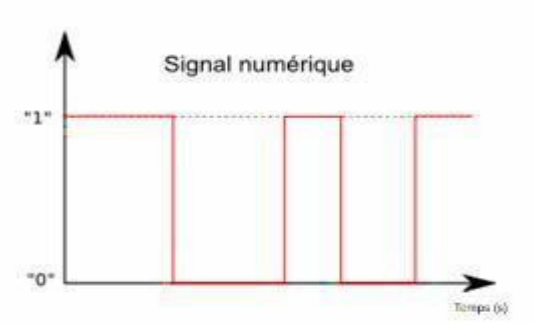

Figure I.4: Signal numérique. [12]

I.4.4 Capteur de température à résistance (RTD) :

Les capteurs RTD mesurent la température en fonction des changements de résistance à l'intérieur d'une résistance métallique. Le RTD le plus populaire, appelé capteur PT100, utilise du platine et a une résistance de 100 ohms à 0 °C. [13]

Figure I.5: Capteur de température à résistance (RTD). [14]

I.5 Critères de choix d'un capteur de température :

Le capteur de température est un composant fondamental de toute installation industrielle moderne. Il joue un rôle déterminant dans la maîtrise et la stabilité des procédés, permettant de surveiller, contrôler et réguler la température à chaque étape de la production.

Grâce à lui, les industriels peuvent garantir la qualité constante des produits finis, assurer la sécurité des équipements sensibles à la chaleur, et optimiser la performance énergétique globale des installations.

Un capteur mal choisi ou mal calibré peut engendrer des dysfonctionnements coûteux : surchauffe, gaspillage d'énergie, baisse de rendement ou dégradation des matières premières. C'est pourquoi le choix du bon capteur de température revêt une importance stratégique, notamment dans les secteurs où la précision et la fiabilité sont primordiales. [15]

I .6 Conclusion :

À la fin de ce premier chapitre, nous pouvons dire que les capteurs jouent un rôle fondamental dans le développement des systèmes intelligents. Ils permettent de surveiller l'environnement en transformant des grandeurs physiques (comme la lumière, la chaleur, la fumée, etc.) en

signaux électriques exploitables. Nous avons présenté les notions fondamentales liées à la température ainsi que les différentes méthodes de mesure utilisées en instrumentation. Nous avons également étudié les principaux types de capteurs et les critères de leur sélection. Ces connaissances constituent une base essentielle pour la conception du système de surveillance de température basé sur Arduino.

Chapitre II :
Présentation de la carte Arduino Uno

II.1 Introduction :

Dans le cadre de la réalisation d'un système de surveillance de température, le choix d'une plateforme matérielle fiable et facile à programmer est essentiel. La carte Arduino Uno représente une solution idéale grâce à sa simplicité d'utilisation, sa flexibilité et son large écosystème.

Ce chapitre est consacré à la présentation détaillée de la carte Arduino Uno ainsi que les outils associés nécessaires au développement du système proposé. Nous aborderons également son architecture interne, ses différentes entrées/sorties, ainsi que l'environnement de programmation utilisé.

II.2 Présentation générale de l'Arduino Uno :

Le module Arduino est un circuit imprimé en matériel libre (plateforme de contrôle), dont les plans de la carte elle-même sont publiés en licence libre, mais certains composants de la carte : comme le microcontrôleur et les composants complémentaires ne sont pas en licence libre. Un microcontrôleur programmé, peut analyser et produire des signaux électriques de manière à effectuer des tâches très diverses. Arduino est utilisé dans beaucoup d'applications comme l'électrotechnique industrielle et embarquée ; le modélisme, la domotique mais aussi dans des domaines différents comme l'art contemporain et le pilotage d'un robot, commande des moteurs et faire des jeux de lumières, communiquer avec l'ordinateur, commander des appareils mobiles (modélisme). Chaque module d'Arduino possède un régulateur de tension +5 V et un oscillateur à quartez 16 MHz (ou un résonateur céramique dans certains modèles). Pour programmer cette carte, on utilise l'logiciel IDEArduino. [16]

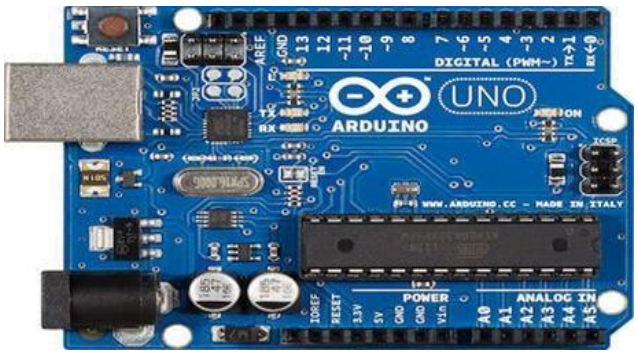

Figure II.1:La carte Arduino UNO

Tableau II.1: Caractéristiques techniques de la carte Arduino UNO. [17]

Element	Caractéristique
Microcontrôleur	ATmega328
Tension de fonctionnement	5V
Tension d'alimentation (recommandée)	7-12 V
Tension d'alimentation (limite)	6-20 V
Broches E/S numériques	14 (dont 6 disposent d'une sortie PWM)
Broches d'entrée analogique	6 (utilisables en broches E/S numériques)
Intensité Maxi disponible par broche E/S	40 mA
Intensité Maxi disponible pour la sortie 3,3V	50 mA
Mémoire programme flash	32 KB (ATmega328) dont 0,5 sont utilisés par le boot loader
Mémoire SRAM (mémoire volatile)	2 KB (ATmega 328)
Mémoire EEPROM (mémoire non volatil)	1KB (ATmega 328)
Vitesse d'horloge	16 MHz

II.3 Architecture de la carte (microcontrôleur ATmega328P) :

Le cœur de la carte Arduino Uno est le microcontrôleur ATmega328P, qui assure le traitement des données et l'exécution des programmes. [18]

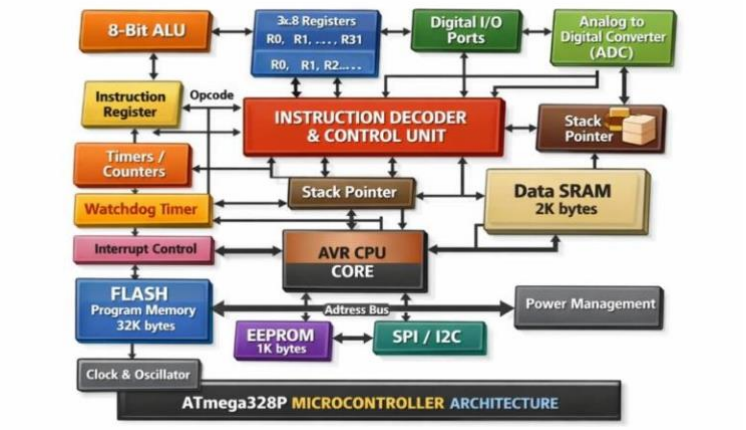

Figure II.2: Architecture de la carte (microcontrôleur ATmega328P).[19]

a) Unité de traitement (CPU) :

Le CPU exécute les instructions du programme stocké dans la mémoire Flash. Il assure le traitement logique et arithmétique. **b) Mémoire :**

Le microcontrôleur possède trois types de mémoire :

- **Flash (32 KB)** : stockage du programme
- **SRAM (2 KB)** : stockage temporaire des données
- **EEPROM (1 KB)** : stockage permanent

c) Périphériques internes:

- Convertisseur analogique-numérique (ADC)
- Timers (minuteries)
- Interfaces de communication (UART, SPI, I2C)

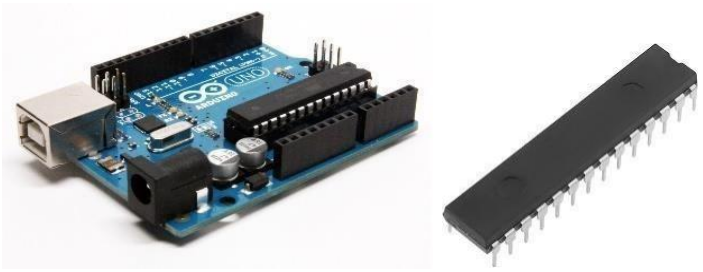

Figure II.3: Le microcontrôleur de l'Arduino. [20]

II.4 Entrées / Sorties (analogiques et numériques) :

a) Entrées/Sorties numériques

La carte Arduino Uno dispose de 14 broches numériques (0 à 13), pouvant être configurées en entrée ou en sortie.

- Mode **entrée** : lecture des capteurs (HIGH / LOW)
- Mode **sortie** : commande des actionneurs

Certaines broches supportent la modulation PWM (Pulse Width Modulation), utilisée pour contrôler la vitesse ou la luminosité. **b) Entrées analogiques**

La carte possède 6 entrées analogiques (A0 à A5) permettant de lire des tensions analogiques (0 à 5V).

Le convertisseur ADC transforme ces signaux en valeurs numériques entre 0 et 1023. [21]

II.5 Environnement de développement Arduino IDE :

L'Arduino IDE est un environnement de développement intégré dédié à la programmation des cartes Arduino telles que Arduino Uno. Il permet de créer, modifier, compiler et téléverser des programmes (sketches) vers le microcontrôleur.

Basé sur un langage dérivé du C/C++, cet environnement est conçu pour être simple d'utilisation, ce qui le rend adapté aussi bien aux débutants qu'aux utilisateurs avancés. Il fonctionne sur plusieurs systèmes d'exploitation (Windows, Linux, macOS). [22]

Fonctionnalités principales :

Tableau II.2: Fonctionnalités principales de la carte Arduino IDE. [23]

Fonctionnalité	Description
Éditeur de code	Permet d'écrire et modifier les programmes Arduino avec coloration syntaxique pour faciliter la lecture du code.
Compilation	Vérifie automatiquement les erreurs du programme avant son envoi vers la carte.
Téléversement	Permet d'envoyer le programme vers la carte Arduino via une connexion USB.
Moniteur série	Outil permettant d'afficher et de surveiller les données envoyées par la carte en temps réel.
Gestion des bibliothèques	Permet d'ajouter et d'utiliser des bibliothèques pour faciliter l'utilisation des capteurs et modules.
Gestion des cartes	Permet de sélectionner le type de carte utilisée comme par exemple Arduino Uno.
Gestion du port série	Permet de choisir le port de communication utilisé pour connecter la carte Arduino à l'ordinateur.
Débogage	Aide à détecter les erreurs dans le code grâce aux messages affichés dans la console.
Multiplateforme	Compatible avec plusieurs systèmes d'exploitation comme Windows, Linux et macOS.

II.6 Langage de programmation Arduino :

Le langage de programmation utilisé dans l'Arduino IDE est basé sur les langages C et C++, avec des simplifications qui facilitent son utilisation pour les systèmes embarqués comme Arduino Uno.

Il permet de développer des programmes appelés sketches, qui sont ensuite compilés et téléversés vers le microcontrôleur. [24]

Structure d'un programme Arduino :

```
void setup() {
  // Initialisation
}

void loop() {
  // Programme principal
}
```

- **setup()** : exécutée une seule fois au démarrage
- **loop()** : exécutée en boucle

Ce langage permet de manipuler facilement les entrées/sorties et les capteurs.

II.7 Modules et composants associés :

Ce projet présente la conception d'un système automatisé capable de surveiller l'environnement et d'agir en cas de dépassement d'un seuil de sécurité.

✓ **Description du Système**

Le dispositif repose sur une architecture en trois étapes :

- **Acquisition** : Le capteur **LM35** convertit la chaleur en signal électrique.
- **Traitement** : La carte **Arduino** interprète ce signal et calcule la température.
- **Action & Interface** : Le système affiche les données sur un **LCD**, signale une alerte via une **LED** et commande un appareil via un **Relais**.

Pour la réalisation de ce projet, nous avons sélectionné avec soin divers composants, que nous allons présenter dans ce qui suit, afin d'assurer un fonctionnement efficace et fiable du système.

✓ **Les Composants Clés :**

- **Microcontrôleur** : Arduino Uno.
- **Capteur LM35** : Précis à ±0,5°C, linéaire (10mV/°C).
- **Écran LCD 1602** : Pour un affichage clair et réduit en câblage.
- **Module Relais** : Interrupteur électronique pour piloter une charge..
- **LED Rouge** : Indicateur visuel d'alarme.

II.7.1 Capteur de température LM35 :

Le LM35 est un capteur de température intégré qui délivre une tension de sortie proportionnelle à la température en degrés Celsius (°C). Contrairement à d'autres capteurs, il ne nécessite pas de calibration externe. [25]

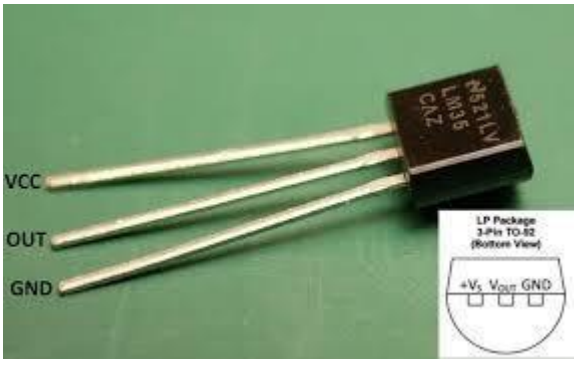

Figure II. 4: Capteur de température LM35. [26]

le capteur contient trois broches: la première est la tension d'entrée (5V), la deuxième est la tension de sortie Vout et la troisième est la masse (GND).

Une des grandes forces du capteur LM35, qui fait sa popularité, c'est sa pré calibration en sortie d'usine. Tous les capteurs LM35 sont calibrés en degré Celsius lors de la fabrication. La tension de sortie Vout est linéairement proportionnelle à la température en degrés Celsius (Vout augmente de 10 mV à chaque fois que la température augmente de 1°C) ou elle fonctionne sur une plage de température de -55°C à +155°Cavecune Très faible consommation (environ 60 µA), ce qui évite qu'il ne chauffe tout seul.

Le capteur LM35 fonctionne avec n'importe quelle tension d'alimentation comprise entre 4 volts et 30 volts, ce qui permet de l'utiliser n'importe quel montage numérique ou analogique.

Le LM35 convertit les changements de température en une différence de potentiel dans un rapport potentiométrique. En alimente le capteur par les deux pattes VCC et GND, la sortie Vout est récupérée sur l'entrée analogique de l'Arduino UNO.

La conversion est également facile car un volt correspond à 100 degrés Celsius, et la lecture analogique d'un signal de 0 à 5 V est codée de 0 à 1023 à l'aide de la formule suivante :

$$\boldsymbol{Température = (volt \times 5 \,/\, 1024) \times 100}$$

- **Expliquer le choix du LM35** : il a été choisi pour sa simplicité et sa précision par rapport à une simple thermistance.

Exemple de Code Arduino

Pour convertir la valeur lue par l'Arduino en température réelle :

```
int pinLM35 = A0; // Branchement sur la broche A0
 void setup()
{
  Serial.begin(9600);
}  void
loop() {
  int lecture = analogRead(pinLM35);
  // Conversion : (Lecture * Voltage / Résolution) * (1000mV / 10mV)
  float temperature = (lecture * 5.0 / 1024.0) * 100.0;
Serial.print("Température : ");
  Serial.print(temperature);
  Serial.println(" °C");

  delay(1000); // Attendre 1 seconde
}
```

II.7.2 LED d'alerte :

Une LED d'alerte (Light Emitting Diode) est une diode électroluminescente utilisée comme indicateur visuel pour signaler un état particulier ou une anomalie dans un système électronique. [27]

Figure II.5 : LED d'alerte

Pour notre Dispositif de Surveillance, la LED d'alerte sert d'indicateur visuel immédiat. Elle permet de savoir, sans regarder l'ordinateur, si le seuil critique est attaint et pour l'ajouter, il faut définir un seuil de température (par exemple 30°C). Si la température dépasse ce seuil, la LED s'allume.

Une LED ne se branche jamais directement sur l'Arduino sans protection, car elle risque de griller ou d'endommager la carte. Pour son branchement, la patte longue (Anode) est connectée à une broche numérique de l'Arduino (ex: Pin 13) via une résistance de 220 ou 330 ohms et la patte courte (Cathode) est connectée au GND (masse).

Code Arduino complet (LM35 + LED)

Voici le code mis à jour pour inclure l'alerte:

```
int pinLM35 = A0;       // Capteur sur A0 int pinLED
= 13;        // LED sur Pin 13 float seuil = 30.0;
// Température d'alerte en °C

   void setup() {
     Serial.begin(9600);

pinMode(pinLED, OUTPUT); // Configure la broche de la LED en
sortie } void loop() {    int lecture = analogRead(pinLM35);
  float temperature = (lecture * 5.0 / 1024.0) * 100.0;
    Serial.print("Température : ");
```

```
  Serial.print(temperature);
  Serial.println(" °C");    //
Condition d'alerte    if
(temperature >= seuil) {
    digitalWrite(pinLED, HIGH); // Allume la LED
Serial.println("ALERTE : Chaleur excessive !");
  } else {     digitalWrite(pinLED, LOW);   //
Éteint la LED
  }
delay(1000);
}
```

vous pouvez ajouter une deuxième LED (Verte) pour indiquer que le système est "Sous contrôle" quand la température est basse ? Cela permet de vérifier d'un coup d'œil que le montage est bien alimenté.

II.7.3 Le relais:

Le relais est un composant électromécanique largement utilisé dans les systèmes électroniques pour isoler la partie commande de la partie puissance. Il fonctionne comme un interrupteur pilotable, permettant d'ouvrir ou de fermer un contacteur sur un circuit de puissance en fonction d'un signal d'entrée, généralement compris entre 0 et 5V.

Pourquoi utiliser un relais ?

- **Isolation :** Il sépare le circuit basse tension (Arduino, 5V) du circuit haute tension (Secteur, 220V).
- **Puissance :** L'Arduino ne peut délivrer que 20 à 40 mA, alors qu'un relais peut supporter jusqu'à **10 Ampères**.

Branchement du module relais (Côté Arduino) : Le module possède généralement 3 broches d'entrée :

- **VCC** : À brancher sur le **5V** de l'Arduino.
- **GND** : À brancher sur le **GND** de l'Arduino.
- **IN** : À brancher sur la broche numérique 8.

Les caractéristiques principales du relais sont présentées dans le tableau suivant [28]:

Tableau II.3: Caractéristiques du relais. [29]

Caractéristique	Description
Tension de commande	5V
Courant de commande	Faible
Tension de commutation	Peut supporter des tensions élevées pour la commutation de charges électriques
Durée de vie électrique	Dépend de la charge électrique commutée et des conditions d'utilisation

Figure II.6: Relais. [30]

II.7.4 Écran LCD (16*2):

L'ajout d'un **écran LCD** est l'étape ultime pour rendre votre dispositif de surveillance de température totalement autonome. IL permet de lire la température en temps réel sans avoir besoin de laisser l'ordinateur branché.

Un écran LCD (Liquid Crystal Display) est un dispositif électronique largement utilisé pour afficher des données et des messages dans de nombreux systèmes. Le modèle couramment utilisé dans notre projet est un écran LCD 16x2, qui comporte 16 colonnes et 2 lignes. Cela lui permet d'afficher jusqu'à 32 caractères au total. Chaque caractère est composé de 5 × 8 (40) points de pixels. Ainsi, l'écran LCD 16x2 peut afficher des informations de manière claire et concise. [31]

Les caractéristiques principales de l'écran LCD sont présentées dans le tableau suivant [32]:

Tableau II.4: Caractéristiques de l'écran LCD. [32]

Caractéristique	Description
Type d'écran	LCD (Liquid Crystal Display)
Format	16x2 (16 colonnes x 2 lignes)
Nombre total de caractères affichables	32
Couleur des caractères	Généralement blanche
Lisibilité	Bonne, même dans des conditions de faible luminosité
Consommation électrique	Variable selon la luminosité et le contenu affiché, généralement basse
Durée de vie	Longue, avec une utilisation normale et un entretien adéquat

Figure II.7: Écran LCD. [33]

L'ajout d'un **écran LCD** (généralement un modèle 16x2 avec interface I2C) est l'étape ultime pour rendre votre dispositif de surveillance de température totalement **autonome**. Il permet de lire la température en temps réel sans avoir besoin de laisser l'ordinateur branché.

II.8 Conclusion:

Dans ce chapitre, nous avons présenté la carte Arduino Uno ainsi que son architecture interne et ses différentes fonctionnalités. Nous avons également étudié les outils logiciels et matériels nécessaires à la réalisation du système.

Ces éléments constituent la base du développement de notre dispositif de surveillance de température, qui sera détaillé dans le chapitre suivant.

Chapitre III :

Simulation et Réalisation

Application : “Système de contrôle et de régulation thermique pour couveuses avicoles “

III.1 Introduction :

Ce chapitre présente la simulation, l'application et la réalisation pratique du système intelligent de contrôle de température destiné à l'aviculture. Il permet de valider le fonctionnement du système proposé à travers une simulation sous Proteus, puis d'expliquer son adaptation au domaine avicole et enfin sa réalisation matérielle.

III.2 Logiciel Proteus :

Proteus est une suite logicielle destinée à l'électronique. Développé par la société Labcenter Electronics, les logiciels incluent dans Proteus permettent la CAO (conception assistée par ordinateur) dans le domaine électronique. Deux logiciels principaux composent cette suite logicielle: ISIS, ARES. Cette suite logicielle est très connue dans le domaine de l'électronique. De nombreuses entreprises et organismes de formation (incluant lycée et université) utilisent cette suite logicielle. Outre la popularité de l'outil, Proteus possède d'autres avantages: Pack contenant des logiciels facile et rapide à comprendre et utiliser. Le support technique est performant. L'outil de création de prototype virtuel permet de réduire les coûts matériel et logiciel lors de la conception d'un projet.

Le logiciel ISIS de Proteus est principalement conçu pour éditer des schémas électriques puis procéder à une animation en temps réel du montage. Par ailleurs, le logiciel permet également de simuler ces schémas ce qui permet de déceler certaines erreurs dès l'étape de conception. Indirectement, les circuits électriques conçus grâce à ce logiciel peuvent être utilisé dans des documentations car le logiciel permet de contrôler la majorité de l'aspect graphique des circuits.[34]

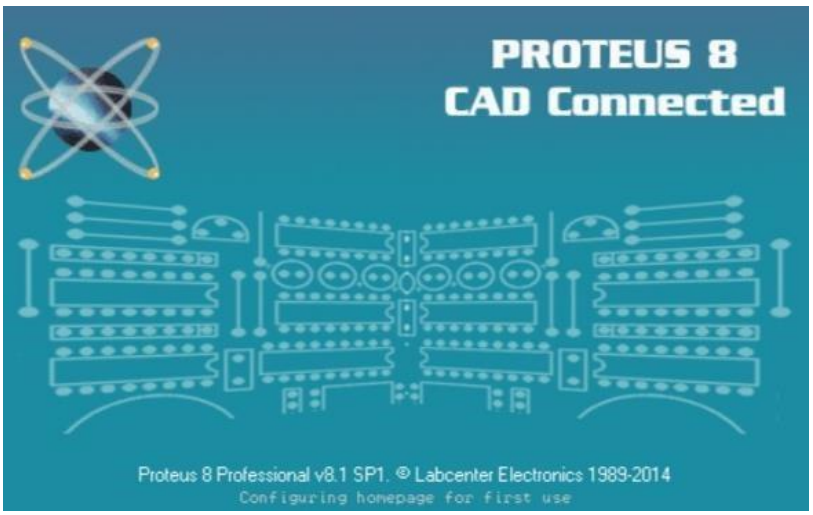

Figure III.6: Logiciel Proteus. [35]

III.3 Simulation du dispositif de surveillance de température:

III.3.1 Présentation du système proposé:

Le système proposé dans ce projet est un dispositif intelligent de surveillance et de contrôle de la température, basé sur une carte Arduino UNO. Il a été conçu afin d'assurer une mesure continue de la température et de réagir automatiquement lorsque celle-ci dépasse une valeur seuil prédéfinie.

Le système repose sur un capteur de température LM35, qui fournit une tension analogique proportionnelle à la température ambiante. Cette information est traitée par la carte Arduino UNO, qui exécute un programme de commande de type Tout ou Rien (ON/OFF).

Lorsque la température mesurée dépasse la valeur de consigne, le système active automatiquement un mécanisme de protection. Un relais coupe l'alimentation du système de chauffage, une LED rouge s'allume pour indiquer l'état d'alerte, et un écran LCD affiche la température ainsi que l'état du système en temps réel.

Afin de valider le bon fonctionnement du système avant sa réalisation matérielle, une simulation est réalisée à l'aide du logiciel Proteus Design Suite. Cette simulation permet de tester différents scénarios de fonctionnement et de vérifier la fiabilité du système.

Ce dispositif constitue une solution simple, efficace et économique pour la surveillance de la température, notamment dans les applications d'aviculture et les environnements nécessitant un contrôle thermique précis.

III.3.2 Schéma synoptique:

Le système est constitué des éléments suivants :

- ✓ Capteur de température LM35
- ✓ Carte Arduino UNO
- ✓ Écran LCD pour l'affichage
- ✓ Relais de commande
- ✓ Système de chauffage (lampe)
- ✓ LEDs de signalisation

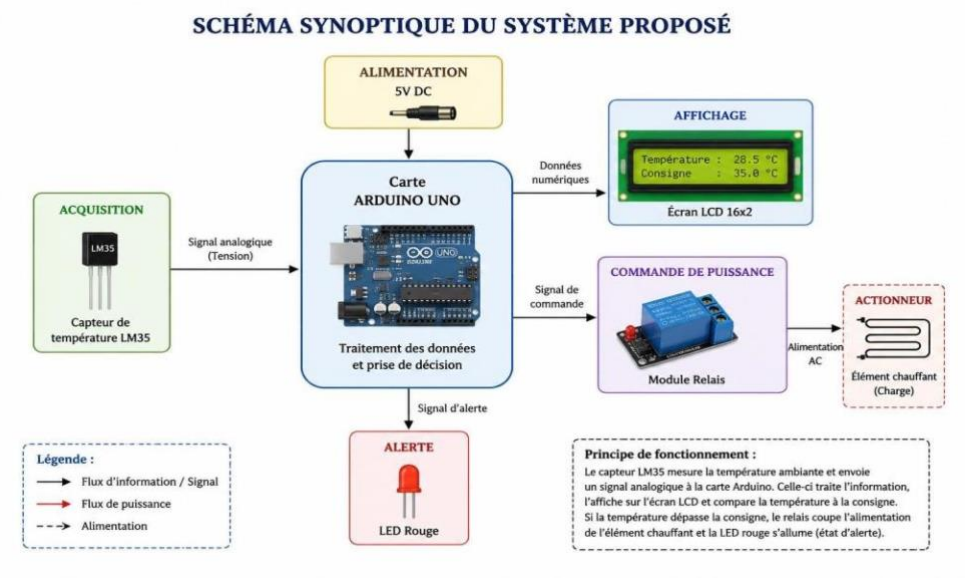

Figure III.2: Schéma synoptique du système proposé.

III.3.3 Conception logicielle en environnement Arduino :

Le programme du système a été développé sous l'environnement Arduino IDE. Ce programme permet la lecture de la température à partir du capteur LM35, le traitement des données mesurées ainsi que le contrôle automatique du système de chauffage et l'affichage des informations sur l'écran LCD.

Dans un premier temps, les bibliothèques nécessaires au fonctionnement de l'écran LCD I2C ont été intégrées. Ensuite, les différentes entrées et sorties du système, telles que le capteur de température, le relais et la LED, ont été configurées comme illustré dans la figure suivante.

code_proteus_INPUT32

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

int sensorPin = A0;
int relayPin = 8;
int ledPin = 9;

float temperature;
int value;

void setup() {

lcd.init();
lcd.backlight();

pinMode(relayPin, OUTPUT);
pinMode(ledPin, OUTPUT);
pinMode(sensorPin, INPUT);

lcd.setCursor(0,0);
lcd.print("Temperature");
delay(1000);
lcd.clear();
```

Figure III.7: Initialisation des bibliothèques et déclaration des variables dans le programme Arduino

.

Après l'étape d'initialisation, le programme effectue une lecture continue de la température puis affiche la valeur mesurée sur l'écran LCD. La température obtenue est ensuite comparée à une valeur de référence fixée à 40°C.

Lorsque la température est inférieure à 40°C, le relais est activé afin de mettre en marche le système de chauffage. En revanche, lorsque la température atteint ou dépasse cette valeur, le chauffage est arrêté et une LED d'alerte s'allume, comme présenté dans la figure suivante.

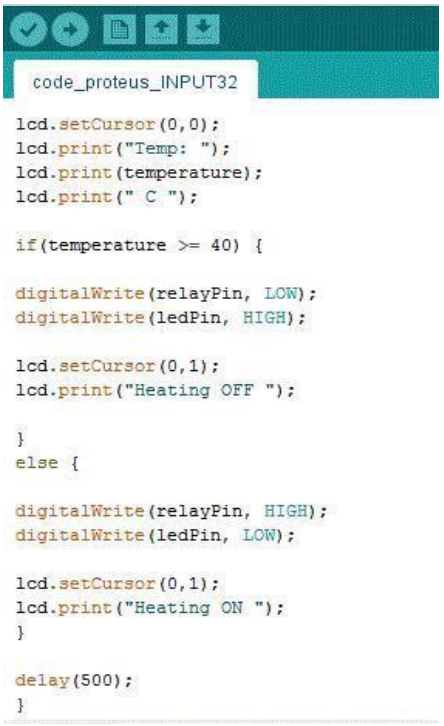

Figure III.8: Partie du programme dédiée au contrôle du système de chauffage.

Le système fonctionne selon une commande de type « Tout ou Rien », où le chauffage est activé ou désactivé en fonction de la température mesurée. Les données sont mises à jour périodiquement toutes les 500 ms afin d'assurer une surveillance continue et une réaction rapide du système.

III.3.5 Simulation du capteur de température:

La simulation du capteur de température LM35 a été réalisée à l'aide du logiciel Proteus Design Suite afin de vérifier le bon fonctionnement du système avant sa réalisation matérielle. Dans cette simulation, le capteur est relié à la carte Arduino UNO, où il mesure la température ambiante et la transforme en un signal électrique analogique proportionnel à la valeur de la température.

L'Arduino UNO lit ce signal via un" entrée analogique, puis le convertit en une valeur numérique représentant la température réelle. Cette valeur est ensuite affichée sur un écran LCD et intégrée au système d'alarme. Lors de la variation de la température dans l'environnement Proteus, le système réagit immédiatement : dans le cas normal, le fonctionnement est stable, tandis que lorsque la température dépasse la valeur seuil définie, le

relais est activé et la LED rouge s'allume pour indiquer un état d'alerte. Cette simulation confirme ainsi l'efficacité et la précision du capteur ainsi que du système dans son ensemble**.**

III.3.6 Affichage des données sur LCD:

Les données relatives à la température et à l'état du système sont affichées sur un écran LCD afin de permettre à l'utilisateur une surveillance simple et en temps réel des valeurs mesurées. Cet écran reçoit les informations provenant de la carte Arduino UNO, après leur traitement par le programme embarqué. Le système affiche en continu la température mesurée en temps réel ainsi que l'état de fonctionnement, qu'il soit normal ou en mode alarme lorsque la valeur dépasse le seuil prédéfini. Les données sont mises à jour de manière continue sur l'écran LCD, ce qui permet de suivre les variations de la température instantanément. Cet affichage améliore l'interaction de l'utilisateur avec le système et rend la surveillance plus claire et plus précise, notamment dans les applications nécessitant un suivi permanent telles que le contrôle de température et les systèmes de protection.

III.3.4 Simulation sous Proteus :

La simulation réalisée permet de vérifier le bon fonctionnement du programme chargé dans le microcontrôleur Arduino UNO. La température de consigne a été fixée à 40°C, et le système réagit selon deux états principaux : l'état de chauffage et l'état de sécurité.

- Dans l'état de chauffage (température < 40°C), comme illustré sur la figure ci-dessous:

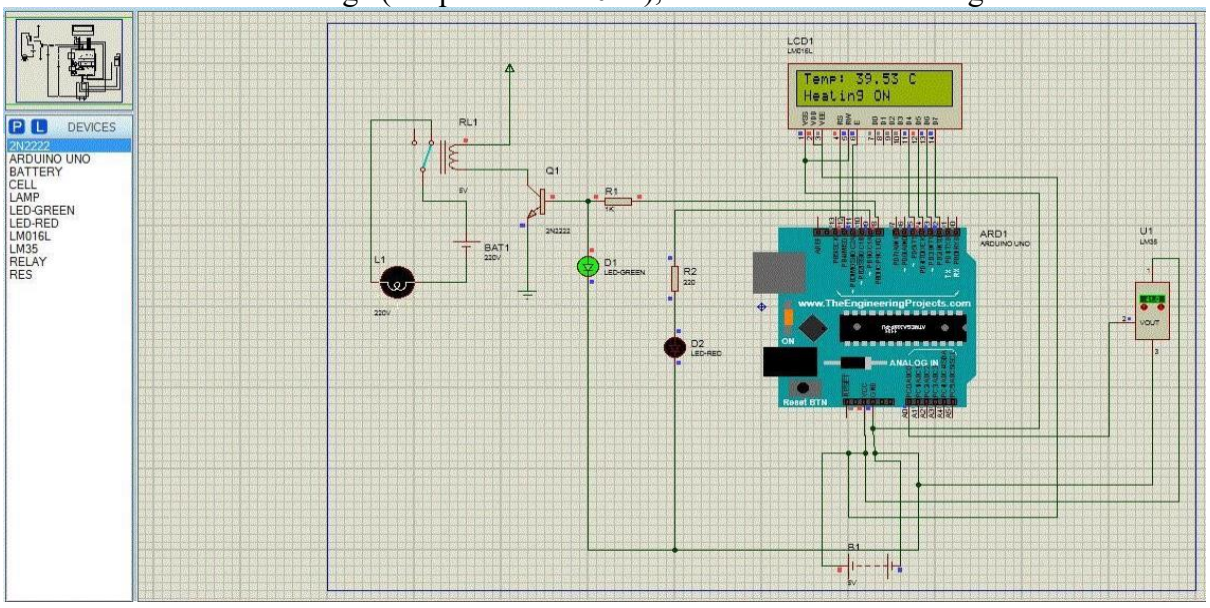

Figure III.5: Simulation de l'état actif pour T < 40°.

La température mesurée par le capteur LM35 est de 39.53°C. Étant donné que cette valeur est inférieure à la température de consigne (39.53°C < 40°C), le microcontrôleur active la sortie numérique commandant le relais. Par conséquent, l'écran LCD affiche le message "**Heating ON**", le relais est activé (contact fermé), ce qui permet l'alimentation du chauffage, et la LED verte s'allume pour indiquer le fonctionnement normal du système.

➢ Dans l'état de sécurité (température ≥ 40°C), comme montré sur la figure ci-dessous:

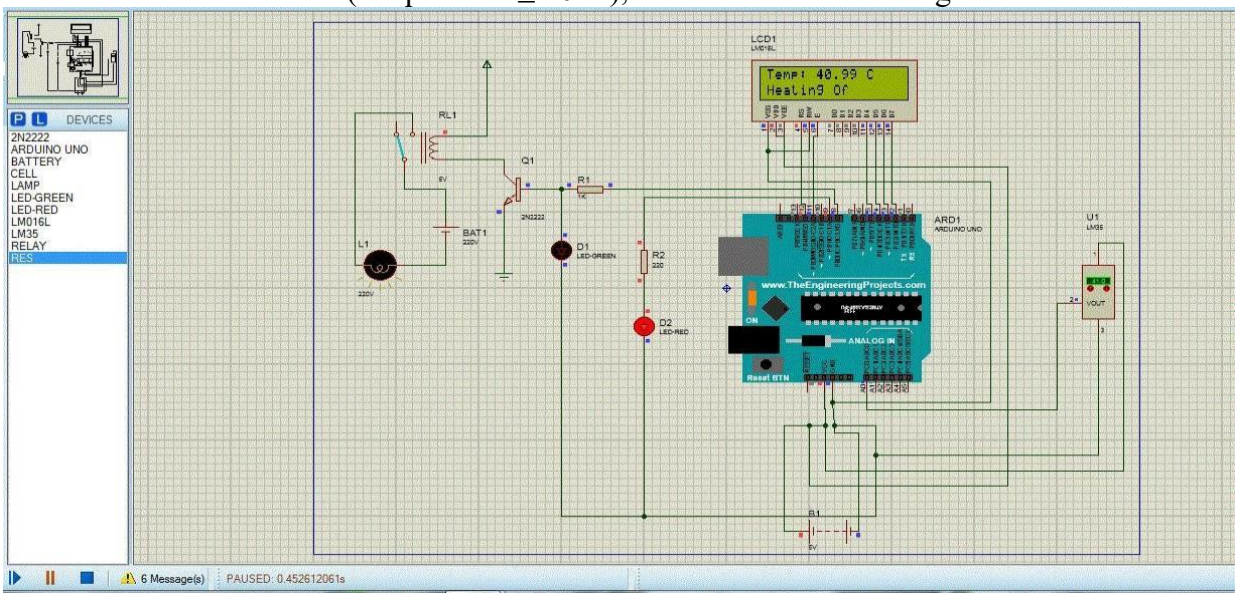

Figure III.6: Simulation de l'état de sécurité pour T ≥ 40°.

Lorsque la température atteint ou dépasse la valeur de consigne,le système passe automatiquement en mode protection, avec une température mesurée de 40.99°C. Étant donné que cette valeur est supérieure ou égale à 40°C (40.99°C ≥ 40°C), la carte Arduino désactive le relais afin de protéger le système. Ainsi, l'écran LCD affiche "**Heating Off**", le relais est désactivé (contact ouvert), ce qui coupe l'alimentation du chauffage, tandis que la LED verte s'éteint et la LED rouge s'allume pour signaler l'état d'alerte.

III.3.7 Analyse des résultats :

L'analyse des résultats obtenus lors de la simulation réalisée sous Proteus Design Suite permet de valider le bon fonctionnement du système de surveillance et de contrôle de température basé sur la carte Arduino UNO. Les différents tests effectués montrent que le dispositif répond correctement aux variations de température conformément à la loi de commande Tout ou Rien (ON/OFF) implémentée dans le programme embarqué.

Dans le mode de fonctionnement normal (température inférieure à la valeur de consigne fixée à 40°C), la température mesurée est de 39.53°C. Dans ce cas, le système maintient l'état de chauffage actif. Le relais est donc activé, assurant l'alimentation du système de chauffage. L'affichage sur l'écran LCD indique clairement l'état "Heating ON", ce qui confirme la cohérence entre la mesure, le traitement et l'action de commande. Par ailleurs, la LED verte reste allumée, indiquant un fonctionnement stable et normal du système.

Lorsque la température atteint ou dépasse la valeur de consigne, le système bascule automatiquement en mode de protection. En effet, pour une température mesurée de 40.99°C, le microcontrôleur Arduino désactive la sortie de commande du relais afin de couper

l'alimentation du chauffage. L'écran LCD affiche alors le message "Heating OFF", tandis que la LED rouge s'allume pour signaler un état d'alerte. Ce comportement traduit une réaction automatique et conforme aux conditions de sécurité définies.

Ces résultats mettent en évidence la rapidité de réaction du système ainsi que sa capacité à assurer une régulation simple mais efficace de la température. La transition entre les deux états (chauffage et protection) s'effectue de manière instantanée dès le dépassement du seuil, ce qui est essentiel dans les applications sensibles telles que l'aviculture. Enfin, la simulation confirme également la fiabilité du capteur LM35, qui fournit une tension analogique proportionnelle à la température, ainsi que la précision du traitement réalisé par la carte Arduino UNO. L'interface d'affichage LCD assure une visualisation claire et en temps réel des données, facilitant ainsi la supervision du système.

III.4 Système de contrôle et de régulation thermique pour couveuses avicoles :

III.4.1 Présentation de l'application avicole:

L'aviculture est la branche de l'élevage spécialisée dans la production de volailles (poulets, dindes, canards) destinées à la consommation ou à la production d'œufs. En Algérie, ce secteur représente une filiale stratégique de l'agriculture nationale et contribue directement à la sécurité alimentaire.

L'une des phases les plus critiques de l'aviculture est la période de couvaison et d'élevage des poussins durant leurs premières semaines de vie. Durant cette période, les poussins sont dits athermiques, c'est-à-dire incapables de réguler leur propre température corporelle. Une déviation de quelques degrés par rapport à la température optimale peut entraîner :

- Une mortalité massive des poussins par hypothermie ou hyperthermie
- Un retard de croissance et une mauvaise conversion alimentaire
- Une vulnérabilité accrue aux maladies infectieuses

Le contrôle thermique de la couveuse est donc un facteur déterminant de la rentabilité de l'exploitation avicole. Le système développé dans ce projet vise à automatiser et optimiser ce contrôle de manière fiable et économique.

Figure III.7: Exemple de couveuse avicole. [36]

III.4.2 Exigences thermiques selon l'âge :

Les besoins en chaleur des poussins diminuent progressivement au fil des semaines selon le tableau suivant, largement adopté dans les exploitations avicoles :

Tableau III.1: Exigences thermiques et état physiologique des poussins selon l'âge.

Période (jour)	Température requise (°C)	Etat physiologique
1-7	35	Poussins nouveau-nés , athermiques
8-14	32	Début de pousse du plumage
15-21	29	Thermorégulation partielle
22-28	26	Plumage presque complet
>28	23	Thermorégulation autonome

Le système développé s'inscrit dans le domaine de l'élevage avicole, plus précisément dans la gestion thermique des couveuses pour poussins. Durant les premières semaines de vie, les poussins sont incapables de réguler leur propre température corporelle (phénomène d'athérmie). Une température inadaptée peut entraîner une mortalité élevée ou un retard de croissance significatif.

Le dispositif proposé offre une solution automatisée et économique permettant de :

- Maintenir la température optimale selon l'âge des poussins
- Réduire les interventions manuelles de l'éleveur
- Signaler visuellement les états normaux et les situations d'alerte

- Réinitialiser facilement le système lors d'une nouvelle couvée

Ce type de système trouve également des applications dans d'autres domaines nécessitant un contrôle thermique automatisé : serres agricoles, incubateurs médicaux, et stockage de produits sensibles à la température.

III.4.3 Principe de fonctionnement du système de surveillance :

Le système repose sur une boucle de régulation continue exécutée toutes les secondes. Le schéma fonctionnel ci-dessous illustre le principe général :

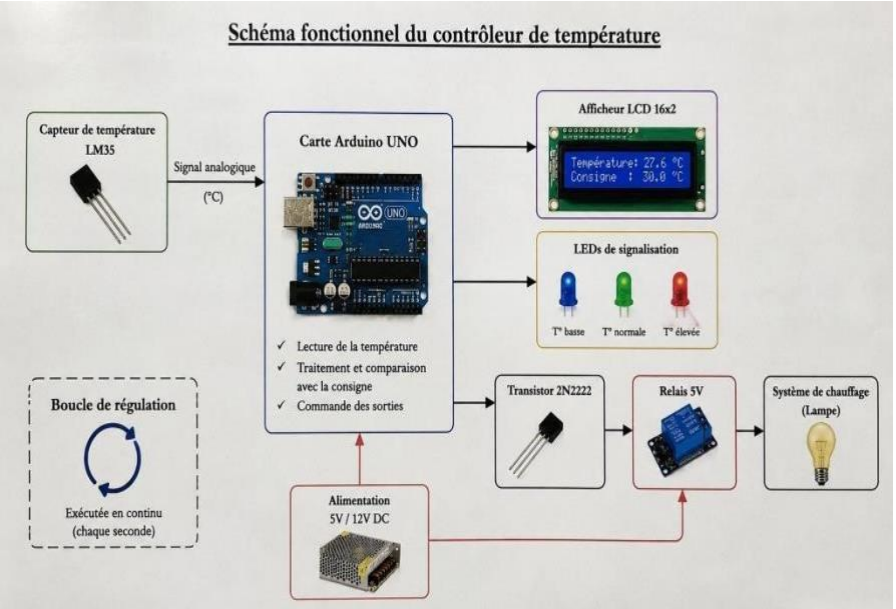

Figure III.8: Schéma fonctionnel du contrôleur de température.

III.4.4 Acquisition de la température

Le capteur LM35 délivre une tension analogique proportionnelle à la température selon la relation Vout = 10 mV × T(°C). Cette tension est lue par le convertisseur analogique-numérique

(ADC) 10 bits de l'Arduino sur l'entrée A0, puis convertie en degrés Celsius :

```
float readTemp() {  int  value  = analogRead(A0);  float
voltage = value * 5.0 / 1023.0;  // Conversion en volts
return voltage * 100.0;              // LM35 : 10 mV/°C }
```

III.4.5 Consigne adaptative selon l'âge :

La consigne de température évolue automatiquement selon l'âge de la couvée, conformément aux recommandations zootechniques. Les poussins développent progressivement leur capacité de thermorégulation au fil des semaines, ce qui permet de réduire la consigne par paliers :

Tableau III.2: Évolution de la consigne de température par paliers hebdomadaires.

Période (jours)	Consigne (°C)	Raison biologique
1 – 7	35°C	Poussins nouveau-nés, athermiques — incapables de réguler leur chaleur
8 – 14	32°C	Début de pousse du plumage
15 – 21	29°C	Plumage en développement, thermorrégulation partielle
22 – 28	26°C	Plumage presque complet
> 28	23°C	Plumage complet, thermorrégulation autonome

L'âge est calculé automatiquement sans module RTC, en utilisant la fonction millis() de l'Arduino :

```
int getAgeDays() {

  return (millis() - startMillis) / 86400000UL;
} float targetTemp(int age) {
if (age <= 7)       return 35;
else if (age <= 14) return 32;
else if (age <= 21) return 29;
else if (age <= 28) return 26;
else                return 23;

}
```

III.4.6 Régulation par hystérésis :

L'algorithme de régulation est de type « Tout ou Rien » avec une bande d'hystérésis de ±1°C. Cette bande morte évite les commutations rapides du relais (chattering) qui réduiraient sa durée de vie :

Tableau III.3:Logique de commande du système de régulation par hystérésis.

Condition	Relais	LED bleue (D8)	LED verte (D10)	LED rouge (D9)
T < Consigne − 1°C	ON	ON — chauffage actif	OFF	OFF
Consigne ± 1°C	OFF	OFF	ON — normal	OFF
T > Consigne + 1°C	OFF	OFF	OFF	ON — alerte

```
void controlSystem(float temp, float target) {

  static bool heaterState = false;   if (temp
< target - 1) heaterState = true;   if (temp
> target + 1) heaterState = false;
  digitalWrite(heaterPin, heaterState);  // Relais + LED bleue
digitalWrite(ledGreen, (temp >= target-1 && temp <= target+1));
digitalWrite(ledRed,   (temp > target + 1));
```

}

III.4. 7 Affichage et supervision :

L'afficheur LCD 16×2 affiche en temps réel toutes les informations nécessaires à la supervision du système :

Tableau III.4: Organisation de l'affichage des données sur l'écran LCD.

Ligne	Contenu	Exemple
Ligne 1	Âge de la couvée + Température mesurée	Age:3d T:34.8C
Ligne 2	Consigne active + Temps écoulé depuis le démarrage	Set:35C 03d:06h

Le bouton poussoir permet de réinitialiser le compteur de jours lors d'une nouvelle couvée, avec anti-rebond logiciel (double lecture avec délai de 300 ms).

III.5. Analyse des résultats de la simulation :

La simulation a été réalisée sous Proteus ISIS après chargement du fichier .hex généré par Arduino IDE. Les figures suivantes présentent les résultats obtenus pour chaque scénario de test.

III.5.1.Schéma de simulation complet

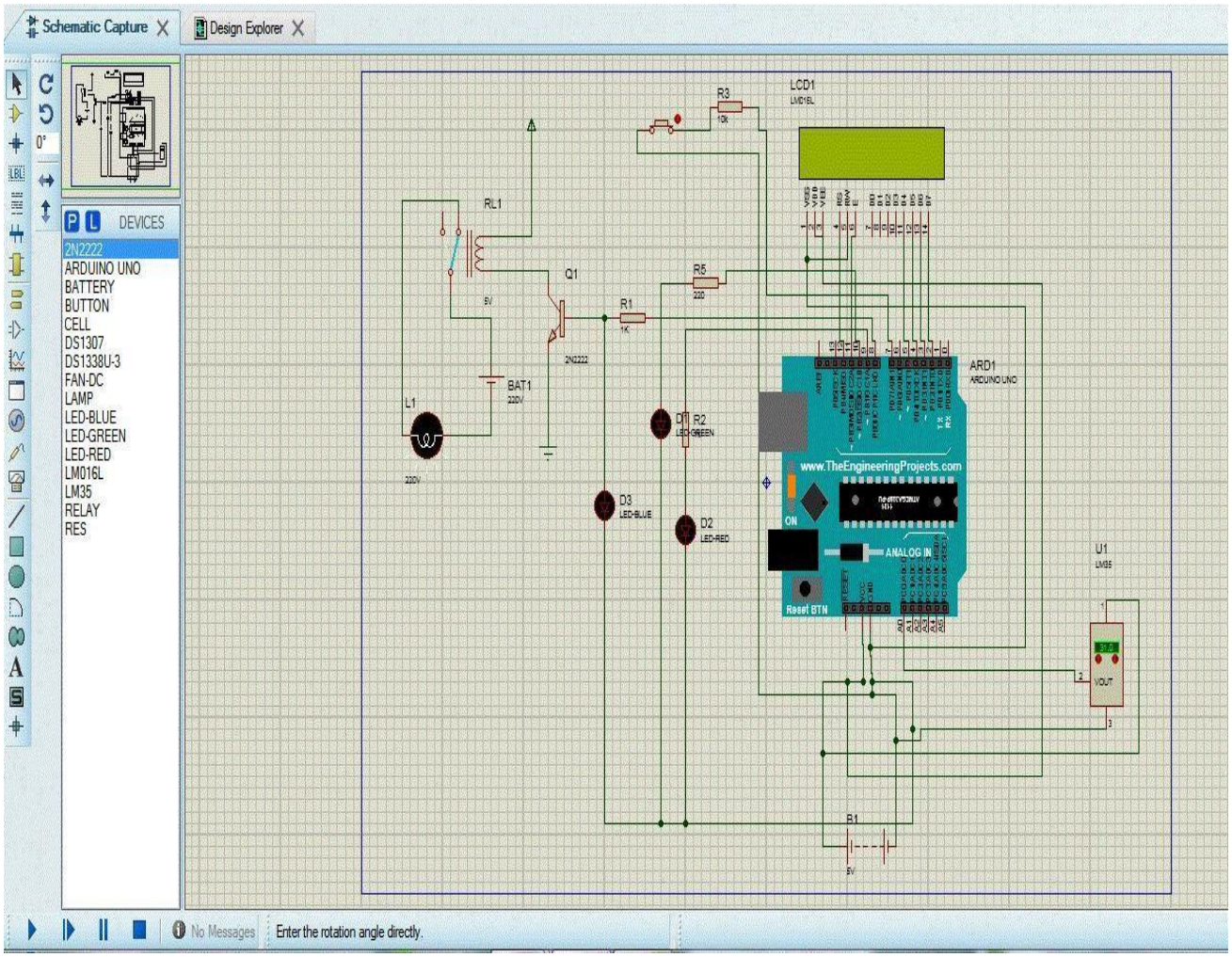

Figure III.9: Schéma de câblage complet sous Proteus ISIS.

III.5.2. Affichage LCD en fonctionnement normal

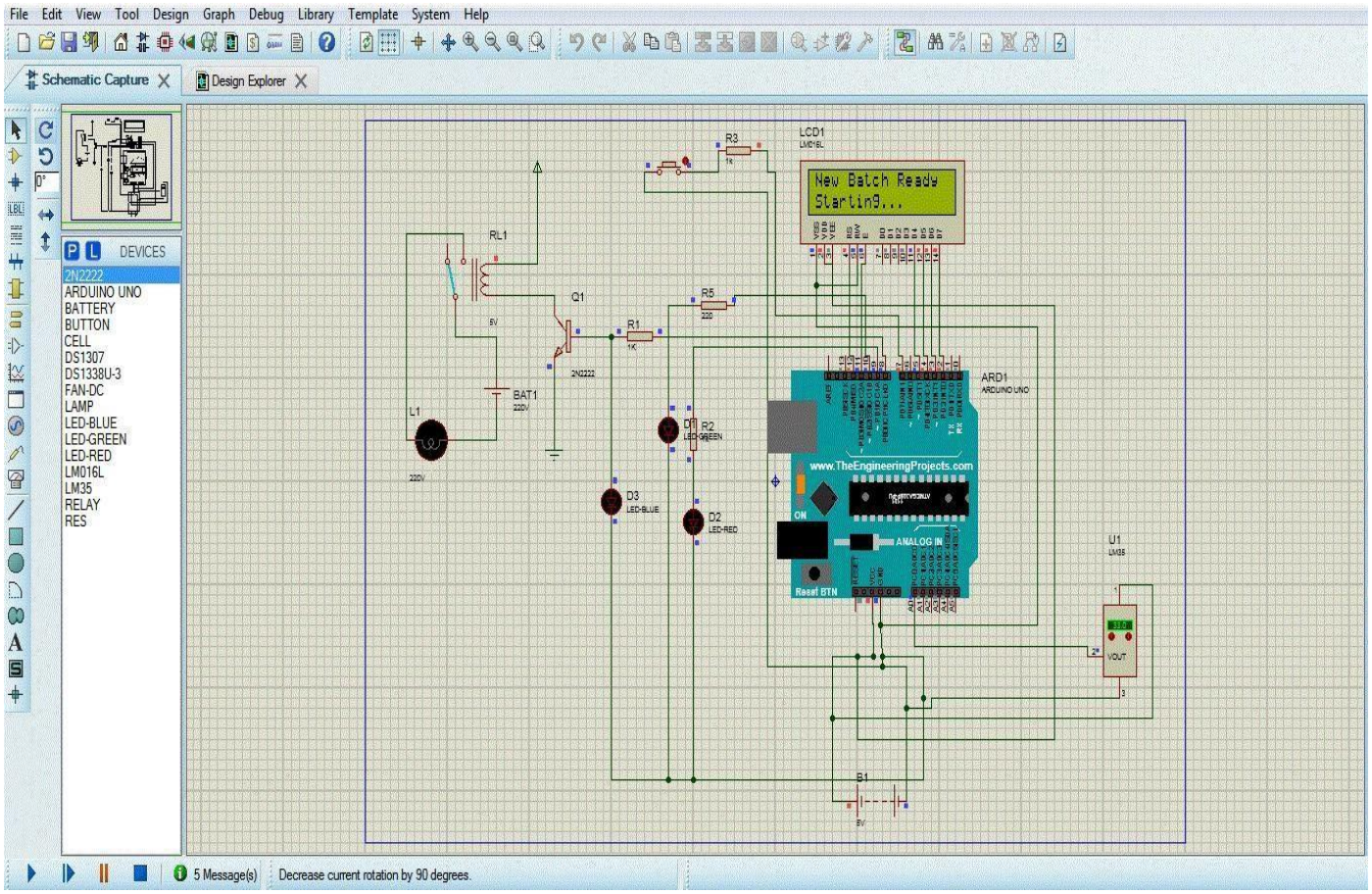

Figure III.9: Afficheur LCD - température mesurée et consigne active.

III.5.3. Activation du chauffage (LED bleue) :

Lorsque la température est inférieure à la consigne moins 1°C, le relais s'active et la LED bleue s'allume pour indiquer que le chauffage est en fonctionnement :

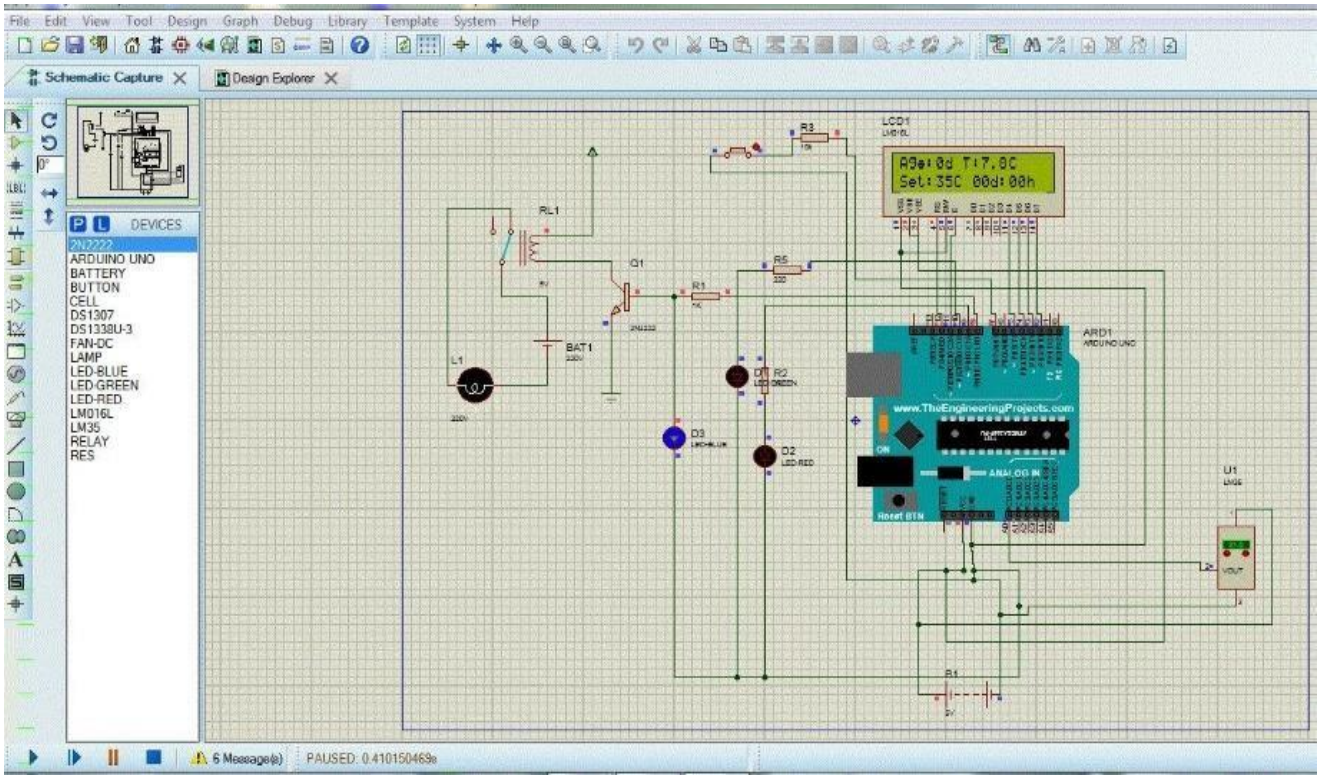

Figure III.11: LED bleue allumée - chauffage actif.

III.5.4.Température normale (LED verte) :

Lorsque la température se situe dans la plage de tolérance (±1°C autour de la consigne), la LED verte s'allume confirmant le fonctionnement normal du système :

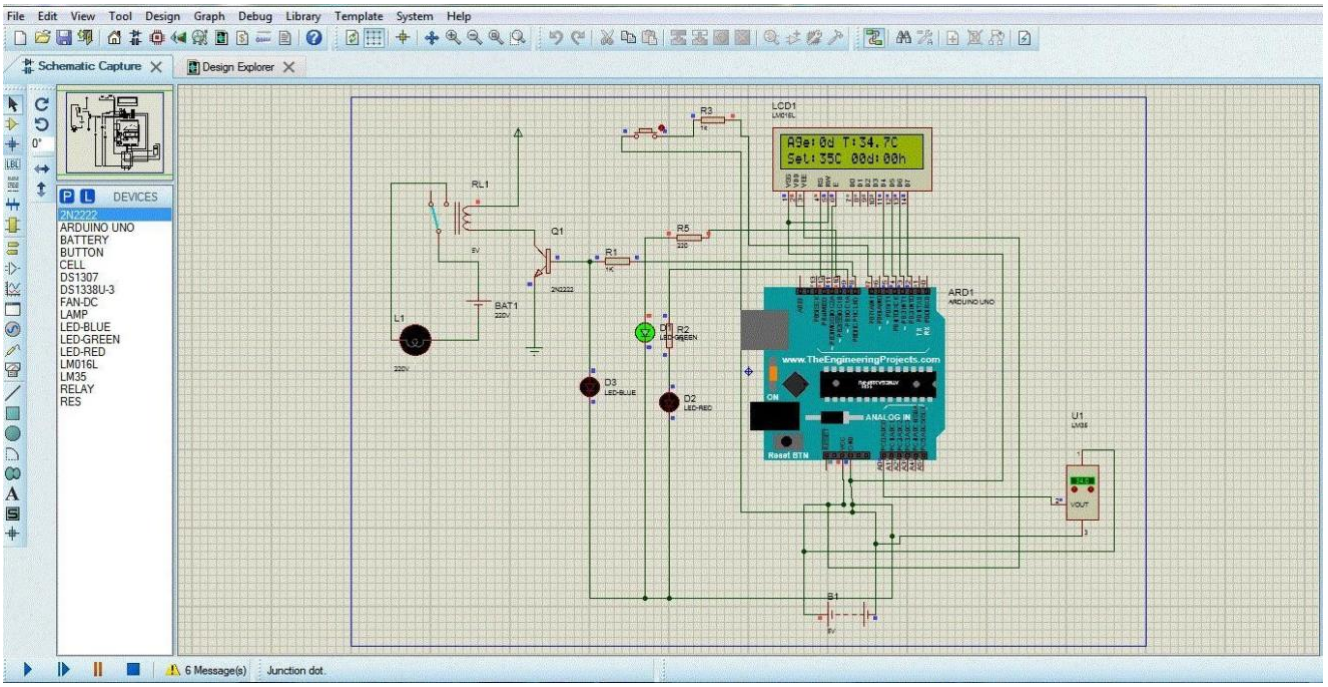

Figure III.12: LED verte allumée - température dans la plage normale.

III.5.5. Détection de surchauffe (LED rouge) :

En cas de dépassement de la consigne de plus de 1°C, le relais se désactive et la LED rouge s'allume pour signaler l'état d'alerte :

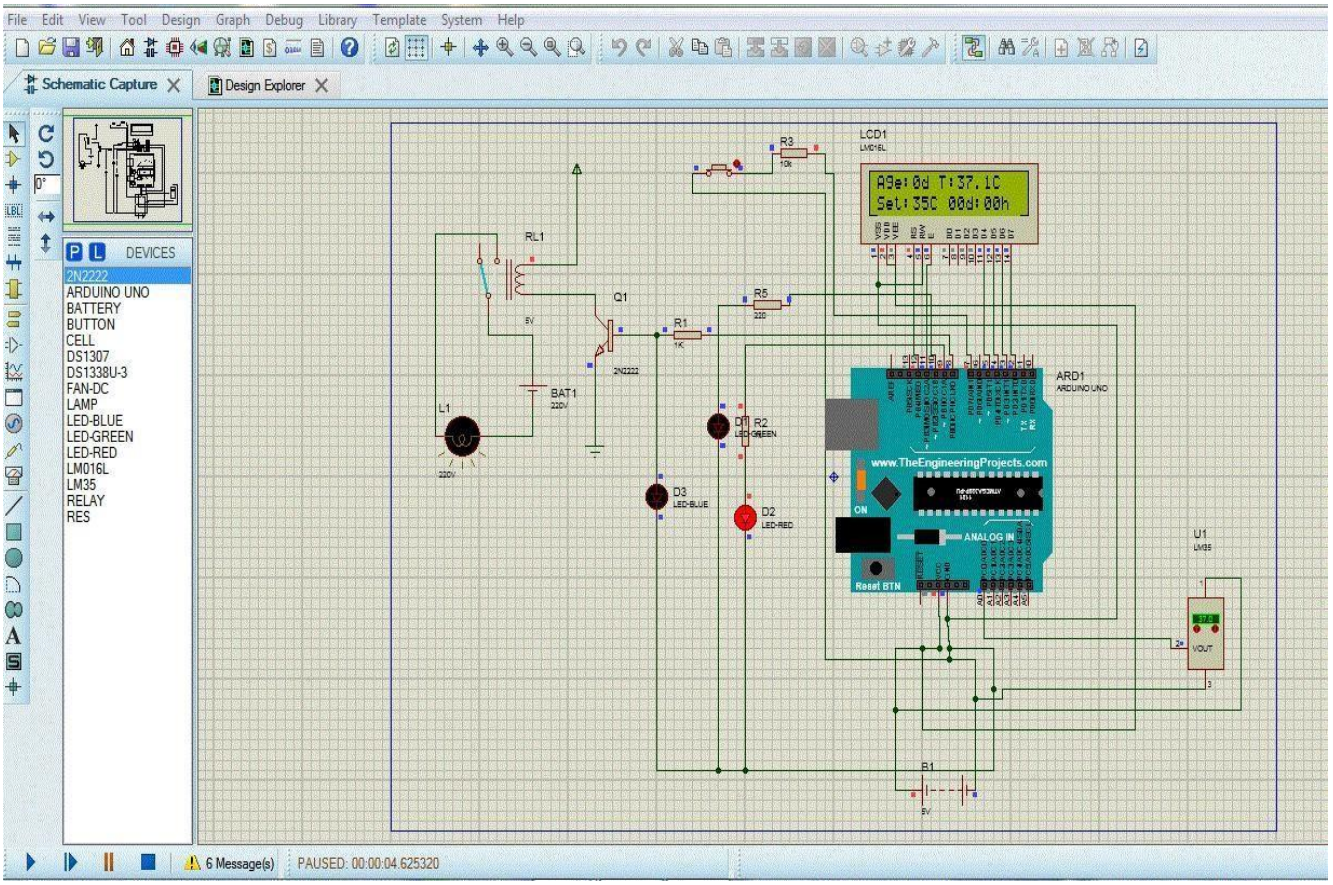

Figure III.13: LED rouge allumée - dépassement de la consigne (état d'alerte).

III.5.6.Tableau de validation des résultats :

Tableau III. 5: Tableau de validation des résultats.

Scénario testé	Comportement attendu	Résultat
T < Consigne − 1°C	Relais ON, LED bleue ON, LCD mis à jour	✓ Conforme
Consigne ± 1°C	Relais OFF, LED verte ON	✓ Conforme
T > Consigne + 1°C	Relais OFF, LED rouge ON (alerte)	✓ Conforme
Changement de consigne (J8)	Passage automatique 35°C → 32°C	✓ Conforme
Appui bouton reset	Réinit. compteur + message affiché	✓ Conforme
Affichage LCD	Mise à jour chaque seconde	✓ Conforme

III.6.Réalisation pratique du système :

Après validation par simulation et la réalisation du système. Des écarts techniques ont été constatés entre la simulation sur Proteus et la maquette réelle, principalement au niveau de l'afficheur LCD et du bloc de commutation (relais).

- **Évolution de l'interface d'affichage (LCD)**

En simulation, un écran LCD 16×2 standard à connexion parallèle a été utilisé, le module I²C étant indisponible dans la bibliothèque Proteus.

En pratique, un afficheur LCD doté d'un module I²C (pilote PCF8574) a été intégré. Cette modification a permis de réduire le nombre de fils de commande et de simplifier le câblage global.

- **Optimisation du bloc de puissance (Relais)**

En simulation, l'absence de module complet dans Proteus a imposé l'ajout d'un transistor de commande externe pour piloter le relais depuis l'Arduino. En pratique, un module relais 5V prêt à l'emploi a été adopté.

Ce module intègre nativement le transistor et la diode de protection, autorisant une connexion directe aux broches de l'Arduino.

Tableau III.6:Comparaison entre simulation et réalisation pratique.

Composant	Simulation Proteus	Réalisation réelle
Afficheur LCD	LCD 16×2 standard (LMD16L)	LCD 16×2 + module I²C PCF8574
Connexion LCD	12 fils (RS, EN, D4–D7, alim.)	4 fils (VCC, GND, SDA, SCL)
Broches Arduino	D2, D3, D4, D5, D11, D12	A4 (SDA), A5 (SCL)
Module relais	Relais commande par transistor	Module relais 5V direct avec Arduino

La figure ci-dessous montre le montage réel du système :

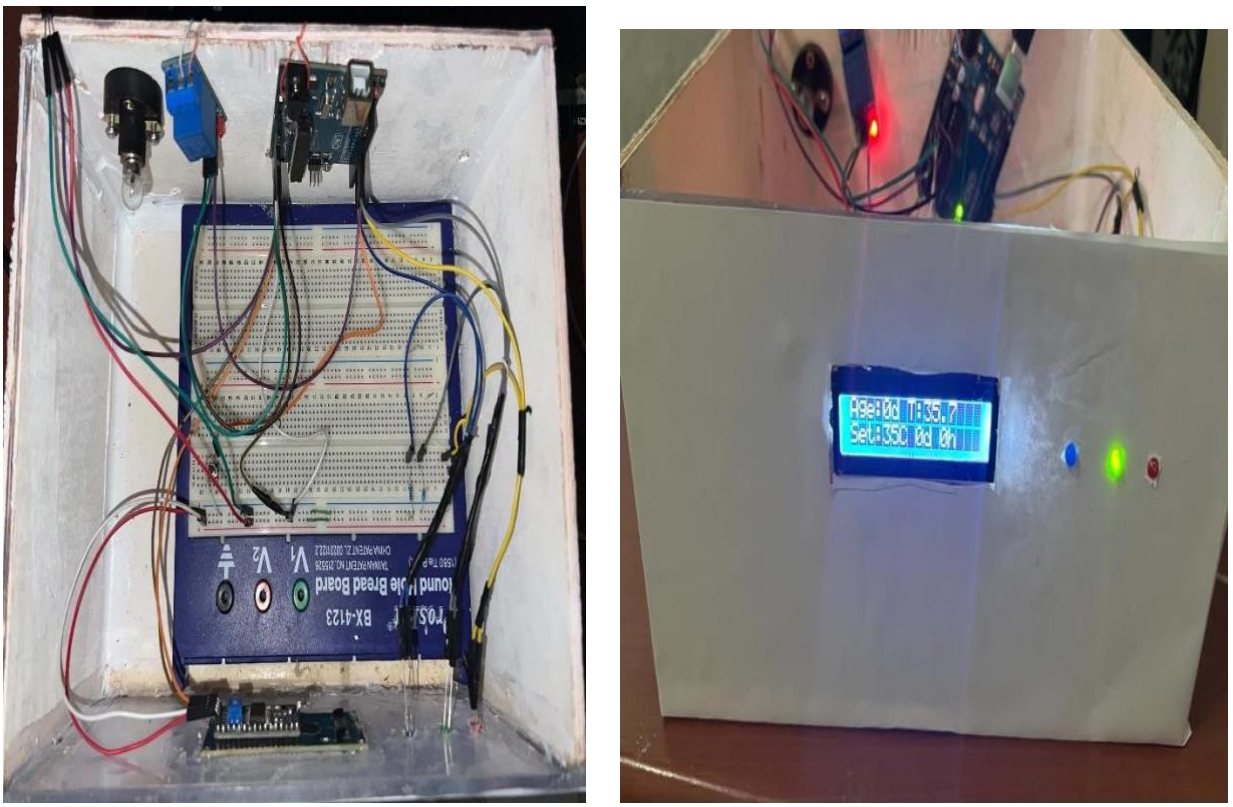

Figure III.14: Montage réel du contrôleur de température.

Cette figure illustre la maquette réelle du système développé. Elle comprend une carte Arduino UNO, un capteur de température LM35, un afficheur LCD 16×2 avec module I²C, un module relais 5V, des LEDs de signalisation ainsi qu'une lampe utilisée comme système de chauffage. L'ensemble est intégré dans une enceinte simulant une couveuse avicole. Le système mesure en permanence la température interne, affiche les valeurs sur l'écran LCD et commande automatiquement l'activation ou l'arrêt du chauffage afin de maintenir la température souhaitée pour l'élevage avicole.

Figure III.15 : Activation du chauffage dans le système de surveillance et de contrôle de température avicole.

Lorsque la température mesurée à l'intérieur de la couveuse avicole devient inférieure à la température de consigne fixée, le système active automatiquement le relais afin de mettre en marche le dispositif de chauffage. Dans cet état, la LED bleue s'allume pour indiquer que le chauffage est en fonctionnement. Cette action permet de maintenir un environnement thermique favorable au développement et au bien-être des volailles. La figure illustre un cas où la température mesurée est de 31,3 °C alors que la consigne est fixée à 35 °C, ce qui entraîne l'activation automatique du système de chauffage.

Figure III.16 : Comportement du système en état de stabilité thermique (35.2°C).

La figure montre le système en fonctionnement normal, où la température mesurée atteint 35.2 °C, une valeur proche de la température de consigne définie. Lorsque la température se situe dans la plage de tolérance autorisée, la LED verte s'allume pour indiquer la stabilité de la température et le bon fonctionnement du système. Cette situation confirme la capacité du dispositif à surveiller en permanence la température et à maintenir des conditions thermiques adéquates à l'intérieur de la couveuse, contribuant ainsi à offrir un environnement favorable à l'élevage avicole et à améliorer le taux de réussite de l'incubation.

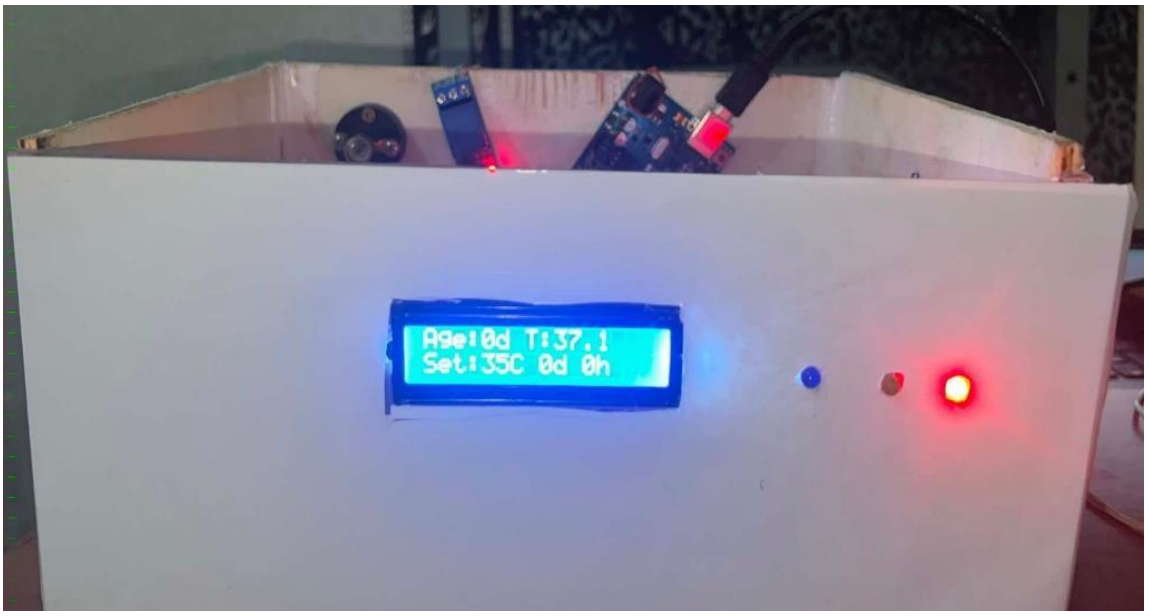

Figure III.17 : Comportement du système lors d'un dépassement de consigne (37.1°C).

La figure illustre l'état de surchauffe du système. Lorsque la température mesurée dépasse la température de consigne de plus de 1 °C, le système active le mode de protection afin d'éviter

tout risque de surchauffe à l'intérieur de la couveuse. Dans cette situation, le relais est désactivé pour couper le système de chauffage, tandis que la LED rouge s'allume pour signaler l'état d'alerte. Cette indication visuelle permet à l'utilisateur d'identifier rapidement un dépassement de température et confirme le bon fonctionnement du dispositif de sécurité intégré au système.

III.7.Solution du système :

Le système réalisé répond à l'ensemble des exigences fonctionnelles définies. Le tableau suivant synthétise les caractéristiques techniques de la solution finale :

Tableau III.7: Caractéristiques techniques de la solution finale.

Paramètre	Valeur
Plage de mesure	0°C – 100°C (LM35DZ)
Précision de mesure	±1°C
Bande d'hystérésis	±1°C
Consignes programmées	35 / 32 / 29 / 26 / 23°C (5 paliers)
Période de mise à jour	1 seconde
Interface utilisateur	LCD 16×2 + 3 LEDs (bleue, verte, rouge)
Commande chauffage	Relais 5V (isolation galvanique)
Réinitialisation	Bouton poussoir avec anti-rebond logiciel
Alimentation	5V via USB ou adaptateur Arduino
Coût estimé	Faible (composants grand public)

Les avantages du système développé sont:

- **Optimisation des coûts**: Faible coût de réalisation grâce à l'utilisation de composants disponibles dans le commerce
- **Flexibilité logicielle**: Programme modulaire facilement modifiable pour d'autres applications thermiques
- **Autonomie de supervision**: Affichage en temps réel permettant une supervision permanente sans instrumentation externe
- **Sécurité matérielle**: Isolation galvanique entre le circuit de commande (5V) et le circuit de puissance via le relais
- **Économie de composants**: Anti-rebond logiciel éliminant le besoin de composants supplémentaires pour le bouton

Les limites du système et perspectives d'amélioration sont:

- **Volatilité des données temporelles** : Le compteur de jours se réinitialise lors des coupures d'énergie. Intégrer un module RTC (DS3231) avec pile de sauvegarde résoudrait ce problème.
- **Absence de redondance**: La régulation dépend d'un point de mesure unique. Implémenter un second capteur LM35 pour moyenner la température.
- **Système isolé** : Aucune communication de données vers l'extérieur. Ajouter un module Wi-Fi (ESP8266) pour basculer vers une supervision IoT à distance

III.8. Conclusion

Ce chapitre a mis en évidence la conception, la validation et la réalisation physique d'un contrôleur de température numérique dédié à la régulation d'une couveuse avicole. Le système, architecturé autour d'un microcontrôleur Arduino Uno et d'un capteur analogique LM35, intègre avec succès une logique de régulation par hystérésis à seuils adaptatifs, programmés pour évoluer automatiquement en fonction de l'âge de la couvée.

La phase de validation par simulation sous l'environnement Proteus ISIS a rigoureusement démontré la conformité du dispositif vis-à-vis du cahier des charges fonctionnel. Les différents scénarios opérationnels testés - à savoir le pilotage du bloc de chauffe, la gestion des alarmes de surchauffe, l'affichage des états sur l'écran LCD et la réinitialisation du système - ont tous validé les choix de conception logicielle et matérielle.

Enfin, la maquette physique finale concrétise une solution à forte viabilité économique, caractérisée par un excellent rapport performance-coût. Bien que le prototype actuel réponde aux exigences immédiates, l'architecture logicielle modulaire développée laisse d'importantes perspectives d'évolution, notamment l'intégration d'un module RTC pour sécuriser la persistance temporelle et le déploiement d'une connectivité sans fil pour la supervision décentralisée (IoT).

Conclusion Générale

Conclusion Générale :

Le travail présenté dans ce mémoire a porté sur la simulation et la réalisation pratique d'un système numérique de régulation thermique automatisé, appliqué à l'optimisation des couveuses avicoles. L'objectif était de créer un système précis, fiable et économique.

La démarche s'est déroulée en deux étapes réussies :

- **La simulation sur logiciel** (Proteus) : Elle a prouvé que le programme fonctionnait correctement, notamment pour changer la température selon l'âge des œufs.
- **La réalisation pratique** : Elle a permis de valider le fonctionnement réel. Le passage au modèle physique a aussi permis de simplifier le câblage grâce à l'utilisation d'un écran I²C et d'un module relais prêt à l'emploi.

En utilisant une carte Arduino Uno et un capteur LM35, ce système montre qu'il est possible de fabriquer une couveuse efficace à un coût très faible. Ce prototype fonctionne parfaitement et sert de base solide pour de futures améliorations, comme l'ajout d'une connexion Wi-Fi ou d'une batterie de secours.

Pour améliorer ce système et le rendre plus professionnel, plusieurs évolutions simples sont envisageables :

- **Sauvegarde de l'heure (Module RTC)** : Ajouter une puce DS3231 pour que le compteur de jours ne revienne pas à zéro si l'électricité se coupe.
- **Supervision à distance (Wi-Fi)** : Remplacer l'Arduino par une carte ESP32 pour envoyer la température sur un smartphone et recevoir des alertes.
- **Contrôle de l'humidité** : Remplacer le capteur LM35 par un capteur DHT22 afin de gérer aussi le taux d'humidité, qui est très important pour les œufs.
- **Sécurité (Deuxième capteur)** : Ajouter un second capteur de température pour vérifier les mesures et éviter les pannes.
- **Circuit imprimé (PCB)** : Fabriquer une carte électronique propre pour enlever tous les fils volants et rendre le boîtier plus solide.

Résumé

Ce projet a pour objectif la conception et la réalisation d'un contrôleur de température numérique, basé sur une carte Arduino UNO. Le système mesure la température locale à l'aide d'un capteur LM35, afficheur LCD et des composants d'alertes ce qui permet la surveillance du système.
Le dispositif fonctionne comme un thermostat "Tout ou Rien" : lorsque la température mesurée dépasse une valeur de consigne prédéfinie, un mécanisme de sécurité s'active. Un relais coupe automatiquement l'alimentation de l'élément chauffant connecté, et une LED rouge s'allume pour signaler l'état d'alerte.
Le projet utilise l'environnement de programmation Arduino IDE et simulé via le logiciel Proteus, permettant ainsi la validation fonctionnelle du circuit avant sa mise en œuvre matérielle.

Mots Clés : Système intelligent, Arduino UNO, capteur LM35, écran LCD, température, LED.

Abstract

This project aims to design and build a digital temperature controller based on an Arduino UNO board. The system measures the local temperature using an LM35 sensor, an LCD display, and alerting components, enabling system monitoring.
The device functions as an on/off thermostat: when the measured temperature exceeds a predefined setpoint, a safety mechanism is activated. A relay automatically cuts power to the connected heating element, and a red LED illuminates to indicate the alert status.
The project uses the Arduino IDE programming environment and is simulated using Proteus software, allowing for functional validation of the circuit before its hardware implementation.

Keywords: Smart system, Arduino UNO, LM35 sensor, LCD display, temperature, LED.

الملخص

يهدف هذا المشروع إلى تصميم وبناء وحدة تحكم رقمية في درجة الحرارة باستخدام لوحة أردوينو أونو. يقيس النظام درجة الحرارة المحلية باستخدام مستشعر LM35وشاشة LCD ومكونات اخرى للتنبيه مما يتيح مراقبة النظام.
يعمل الجهاز كمنظم حرارة ثنائي الوضع (تشغيل/إيقاف): فعندما تتجاوز درجة الحرارة المقاسة قيمة محددة مسبقا، يتم تفعيل آلية الأمان .يقوم المرحل بفصل الطاقة تلقائيا عن عنصر التسخين المتصل، ويضيء مؤشر LED احمر للاشارة الى حالة التنبيه.يستخدم المشروع بيئة برمجة أردوينو يتم محاكاته باستخدام برنامج بروتيوس مما يسمح بالتحقق بالتحقق من الدائرة قبل تنفيذها على ارض الواقع.

الكلمات المفتاحية نظام ذكي ، درجة الحرارة، مؤشر LED ، شاشة LCD أردوينو أونو، مستشعر LM35.

Réferences

References:

[1] Y. A. Çengel and M. A. Boles, Thermodynamics: An Engineering Approach, 8th ed. New York, NY, USA: McGraw-Hill Education, 2015.

[2] J. P. Holman, Heat Transfer, 10th ed. New York, NY, USA: McGraw-Hill, 2010.

[3] D. Patranabis, Sensors and Transducers, 2nd ed. New Delhi, India: PHI Learning, 2013. [4] AMMARKHODJA Nassim, « Etude et réalisation d'une alarme de température à base d'une carte arduino » Mémoire de fin d'étude, Université Mouloud Mammeri de Tizi-Ouzou, 2018. [5] Y. A. Çengel and M. A. Boles, Thermodynamics: An Engineering Approach, 8th ed. New York, NY, USA: McGraw-Hill, 2015.

[6] https://maferme.ma/capteur-pour-serre-intelligente/

[7] Texas Instruments, "Thermistors in Temperature Sensing Applications," Application Report, 2019

[8] Vishay Intertechnology, "NTC Thermistors – Theory and Application," Technical Document, 2018

[9] Jacob Fraden, Handbook of Modern Sensors: Physics, Designs, and Applications, 5th ed., Springer, 2016.

[En ligne]. https://www.ti.com/lit/ds/symlink/lm35.p

[10] V. Vanneste, « Signal analogique », Signal Analogique / Numérique [Arduino].
[En ligne]. Disponible sur :
http://www.vincentvanneste.fr/views/arduino/co/AnalogiqueNumerique.html.

[11] Maxim Integrated, "DS18B20 Programmable Resolution 1-Wire Digital Thermometer Datasheet," Maxim Integrated, 2015.

[12] B. Hanin, "Qu'est ce qu'un signal numérique ?," Conversion Numérique <-> Analogique, [En ligne].
http://bernard.hanin.free.free.fr/Secours/co/Communication/Conversion_numerique/co/02_signal_numerique.html.

[13] Omega Engineering, "RTD Temperature Sensors – Principles and Applications," Technical Guide, 2018.

[14] Madison Company, "Capteur de température à résistance (RTD)," DirectIndustry, 2026.. [En ligne]. https://www.omega.com/en-us/resources/rtd

[15] IEC, IEC 60751: Industrial platinum resistance thermometers and platinum temperature sensors, International Standard, 2008.

[16] Arduino, "Arduino Documentation – Getting Started," 2020.

[17]Arduino, "Arduino Uno Rev3 Datasheet," Arduino, 2023.
[En ligne]. https://store.arduino.cc/products/arduino-uno-rev3
[En ligne]. https://docs.arduino.cc/

[18] Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi, The AVR Microcontroller and Embedded Systems, Pearson, 2011.

[19]Microchip Technology, "Atmega328P: 8-bit AVR Microcontroller with 32K Bytes InSystem Programmable Flash," Datasheet, DS40001906C, 2018. [En ligne]. https://microchip.com

[20]Arduino, "Arduino Uno Rev3," Arduino Docs, 2024. [En ligne]. arduino.cc. [Consulté le : 23-mai-2024].

[21] Arduino, Arduino Uno Rev3 – Technical Specifications, [En ligne]. https://store.arduino.cc/products/arduino-uno-rev3

[22] Bjarne Stroustrup, The C++ Programming Language, 4th ed., Addison-Wesley, 2013.

[23] M. Banzi and M. Shiloh, Getting Started with Arduino, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2014.

[24] Brian W. Kernighan and Dennis M. Ritchie, The C Programming Language, 2nd ed., Prentice Hall, 1988.

[25] Texas Instruments, “LM35 Precision Centigrade Temperature Sensors,” Datasheet, 2017. [En ligne]. https://www.ti.com/lit/ds/symlink/lm35.pdf

[26]Texas Instruments (2016). LM35 Precision Centigrade Temperature Sensors (Fiche technique n° SNIS159G). Récupéré de https://www.ti.com/lit/ds/symlink/lm35.pdf

[27] Teas Instruments, “Light Emitting Diodes (LEDs) Overview,” Application Report, 2016.

[28] https://docs.rs-online.com/d839/0900766b8009921d.pdf

[29] Arduino, "Arduino Uno Rev3," Arduino Docs, 2024. [En ligne]. arduino.cc. [Consulté le : 23-mai-2024].

[30] SunFounder, “Module Relais 5V,” SunFounder’s Documentations, 2023. [En ligne]. https://docs.sunfounder.com/projects/umsk/fr/latest/01_components_basic/30component_relay.html.

[31] Djehaiche Rania Et Benziouche Nihad «Etude Et Application D’un Système De Communication M2m », Mémoire De Master, Université De Mohamed El-Bachir El-Ibrahimi - Bordj Bou Arreridj, 2019

[32] Hitachi, HD44780U (LCD-II) Dot Matrix Liquid Crystal Display Controller/Driver Datasheet, Hitachi Ltd., 1998.

[33]LCD Module User Manual, HD44780, Hitachi, Tokyo, Japon, 1999

[34] K. Smatel and K. Belacel, “Étude et simulation d’un capteur de température par une carte Arduino,” Mémoire de fin d’études de Master en Instrumentation, Département des Sciences et de la Technologie, Université de Tissemsilt, Tissemsilt, Algérie, 2023.

[35] A. M. Z. Sadiq, “Design and Implementation of Temperature Monitoring System Using Arduino,” International Journal of Engineering Research and Applications, vol. 10, no. 5, pp. 45–50, 2020.

[36] “Élevage de poussins en couveuse avicole,” photographie numérique.