### **Project: Automatic Egg Incubator System**

#### **Objective:**

Design an automated egg incubator that controls the environment (temperature, humidity, and egg tilting) to ensure proper incubation conditions for bird eggs.

### **1. Component Selection**

#### **Sensors:**

* **DHT11**: For temperature and humidity sensing at the middle of the egg shelf. Humidity is crucial for proper egg development.

#### **Actuators:**

* **Relay for Circulatory Fan**: Controls the fan to distribute heat evenly within the incubator.
* **Relay for Heater**: Controls the heater to maintain the set temperature.
* **Servo Motor**: Used for tilting the eggs to mimic the natural rotation done by birds, ensuring even heating and proper embryo development.

#### **Display:**

* **16x4 LCD**: To show the incubator name, preset and current temperatures, and humidity.

#### **Microcontroller:**

* **STM32F103C8 (blue pill)**: Chosen for its high performance, availability of multiple GPIOs for sensor/actuator interfacing, and low power consumption.

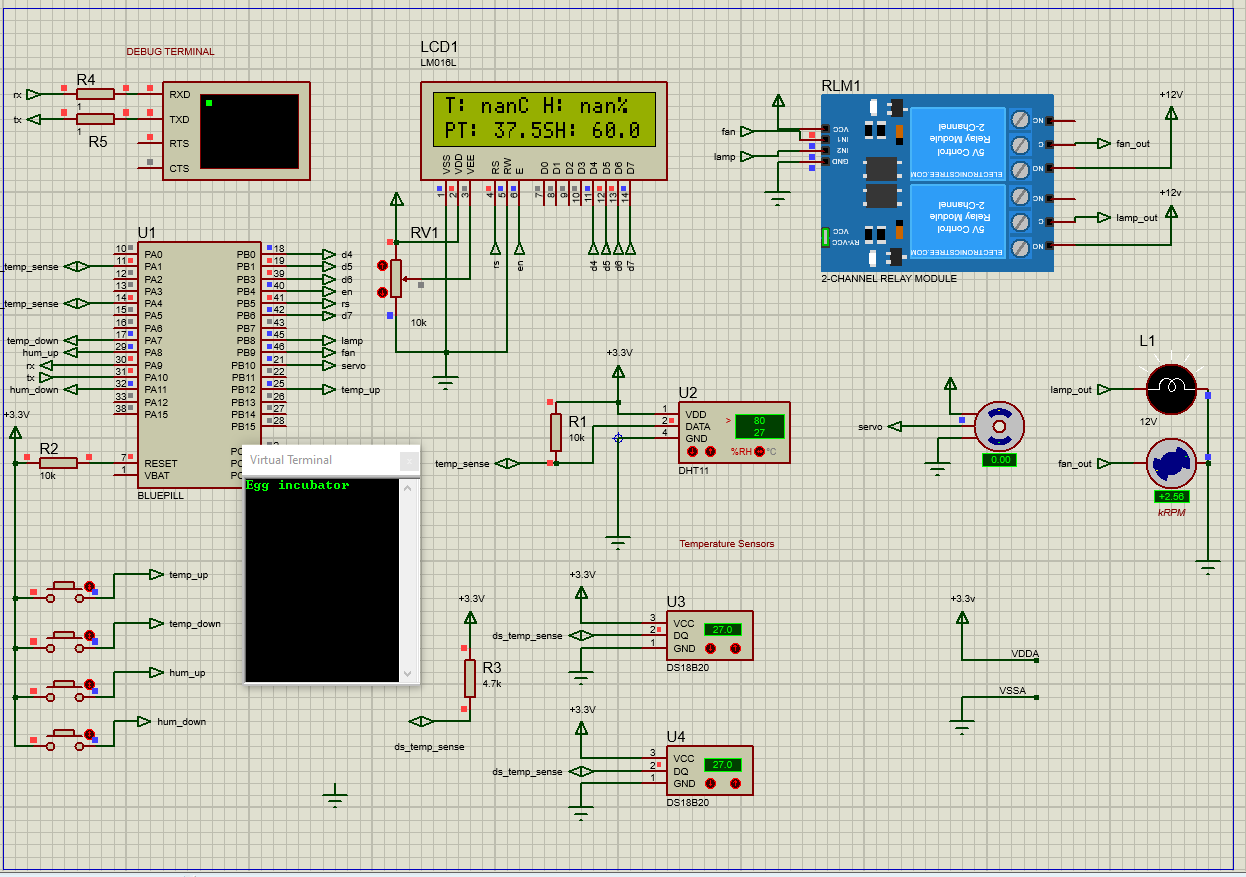
#### **Other Components:**

* **4 Button**: To adjust the preset temperature and humidity settings manually.

### **2. Schematic Diagram**

* + One **DHT11** sensor is connected to a GPIO pin for middle temperature and humidity sensing.
  + **2 DS18b20** for top and bottom temperature sensing.
  + **Relay 1** for the fan connected to a GPIO pin via a transistor for switching.
  + **Relay 2** for the heater connected to another GPIO pin via a transistor.
  + **The Servo motor** is connected to a PWM pin of STM32 for egg tilting.
  + **LCD** (16x2) connected via parallel interface to display incubation data.
  + **Buttons** connected to GPIO pins to allow the user to adjust preset values for temperature and humidity.

**Schematic Capture**

****

**Note: Proteus may not have fully support the simulation of certain components like the DS18B20 temperature sensor and the DHT11 sensor which gave me serious issues simulating it. But i know that my code would work in actual hardware with or without minimal adjustment around capturing the temperature.**

### **3. Material Selection**

* **DHT11**: Measures both temperature and humidity, providing additional information to control the incubation environment.
* **DS18b20**: Measures temperature and provides additional information to control the incubation environment.
* **Relay Modules**: Used to control high-power components like the fan and heater, which the STM32 cannot drive directly.
* **Servo Motor**: Essential for periodic egg tilting, which helps improve the chances of successful hatching.
* **16x2 LCD**: Displays crucial information, ensuring the user can easily monitor and adjust the incubation process.

### **4. Control Process Flow**

The system can be described with the following control flow:

1. **Initialization**:
   * Initialize the DS18B20 sensors, DHT11 sensor, relays, servo motor, LCD, and buttons.
   * Display the project name on the LCD and set the initial conditions (preset temperature, humidity, etc.).
2. **Monitoring**:
   * Continuously monitor the temperature at the top, middle, and bottom of the egg shelf.
   * Read humidity data from the DHT11 sensor.
3. **Heating Control**:
   * Compare the average temperature from the sensors (top, middle, bottom) to the preset temperature.
   * If the temperature is below the preset value, turn on the heater via the relay.
   * Once the temperature reaches the preset value, turn off the heater.
   * The circulatory fan is used to distribute the heat evenly when the heater is on.
4. **Humidity Control**:
   * Monitor the humidity and display the current value on the LCD.
   * If needed, additional humidity control (e.g., by activating a water reservoir system) can be added.
5. **Egg Tilting**:
   * Every few hours, use the servo motor to tilt the eggs to a predefined angle to promote even embryo development.
6. **User Input**:
   * Using buttons, allows the user to manually adjust the preset temperature and humidity values.
   * Display the updated values on the LCD.
7. **LCD Display**:
   * Continuously update the display with the current temperature, humidity, and other relevant information.
   * Format: First row – Current Temperature and Humidity.
   * Second row – "Preset/Current" temperature values.

### **5. Project Implementation**

#### **Codebase (STM32 - C Language)**

#### **Development Environment PlatformIO, Arduino Framework)**

#### **Simulating software (Proteus)**

Here’s the STM32 code:

#include <Arduino.h>

#include <LiquidCrystal.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include <DHT.h>

// Pin definitions

#define ONE\_WIRE\_BUS PA4 // Pin for DS18B20 sensors

#define DHT\_PIN PA1 // Pin for DHT11 sensor

#define HEATER\_RELAY PB8 // Pin for heater relay

#define FAN\_RELAY PB9 // Pin for fan relay

#define SERVO PB10 // Pin for servo motor

#define BUTTON\_TEMP\_UP PB12 // Pin for button (for adjusting temperature value, positive)

#define BUTTON\_TEMP\_DOWN PA7 // Pin for button (for adjusting temperature value, negative)

#define BUTTON\_HUM\_UP PA8 // Pin for button (for adjusting humiidity value, positive)

#define BUTTON\_HUM\_DOWN PA11 // Pin for button (for adjusting humiidity value, negative)

// Define pins for the 16x4 LCD (RS, E, D4, D5, D6, D7)

#define LCD\_RS PB5

#define LCD\_Enable PB4

#define LCD\_D4 PB0

#define LCD\_D5 PB1

#define LCD\_D6 PB3

#define LCD\_D7 PB6

#define DHTTYPE DHT11 // DHT11 sensor type

// Initialize components

OneWire oneWire(ONE\_WIRE\_BUS);

DHT dht(DHT\_PIN, DHTTYPE);

DallasTemperature ds18b20(&oneWire);

DeviceAddress tempSensor1, tempSensor2;

LiquidCrystal lcd(LCD\_RS, LCD\_Enable, LCD\_D4, LCD\_D5, LCD\_D6, LCD\_D7);

// Variables

float topTemp, bottomTemp, middleTemp;

float humidity;

int eggTiltAngle = 0;

float setTemperature = 37.5; // Default set temperature

float setHumidity = 60.0; // Default set humidity

// Function to initialize components

void setup() {

// Initialize serial communication

Serial.begin(9600);

Serial.println("Egg incubator");

// Initialize sensors

dht.begin();

ds18b20.begin();

// Assign addresses manually to each sensor

if (ds18b20.getAddress(tempSensor1, 0)) {

Serial.println("Sensor 1 found.");

}

if (ds18b20.getAddress(tempSensor2, 1)) {

Serial.println("Sensor 2 found.");

}

// Initialize LCD

lcd.begin(16, 4); // Set LCD size to 16x4

// Initialize relays and servo

pinMode(HEATER\_RELAY, OUTPUT);

pinMode(FAN\_RELAY, OUTPUT);

pinMode(SERVO, OUTPUT);

pinMode(BUTTON\_TEMP\_UP, INPUT\_PULLDOWN );

pinMode(BUTTON\_TEMP\_DOWN, INPUT\_PULLDOWN );

pinMode(BUTTON\_HUM\_UP, INPUT\_PULLDOWN );

pinMode(BUTTON\_HUM\_DOWN, INPUT\_PULLDOWN );

// Set initial relay and servo states

digitalWrite(HEATER\_RELAY, LOW);

digitalWrite(FAN\_RELAY, LOW);

// Welcome message on LCD

lcd.setCursor(0, 0);

lcd.print("Egg Incubator");

delay(1000);

lcd.clear();

}

// Function to move the servo to a specific angle (0° to 180°)

void moveServo(int angle) {

// Calculate the pulse width in microseconds

int pulseWidth = map(angle, 0, 180, 1000, 2000); // 1000us = 1ms, 2000us = 2ms

// Send the pulse to the servo

unsigned long startTime = millis(); // Record the start time

while (millis() - startTime < 500) {

digitalWrite(SERVO, HIGH);

delayMicroseconds(pulseWidth);

digitalWrite(SERVO, LOW);

delay(20); // 20ms corresponds to the 50Hz servo refresh rate

}

}

// Function to read sensor data

void readSensors() {

ds18b20.requestTemperatures();

topTemp = ds18b20.getTempC(tempSensor1); // Top sensor

bottomTemp = ds18b20.getTempC(tempSensor2); // Bottom sensor

middleTemp = dht.readTemperature(); // Middle sensor (DHT11)

humidity = dht.readHumidity(); // Humidity from DHT11

}

// Function to control heating and fan

void controlHeating() {

float avgTemp = (topTemp + bottomTemp + middleTemp) / 3;

if (avgTemp < setTemperature) {

digitalWrite(HEATER\_RELAY, HIGH); // Turn on heater

digitalWrite(FAN\_RELAY, HIGH); // Turn on fan with heater

} else {

digitalWrite(HEATER\_RELAY, LOW); // Turn off heater

digitalWrite(FAN\_RELAY, LOW); // Turn off fan

}

}

// Function to control egg tilting

void controlEggTilting() {

// Tilt eggs every 4 hours

if (millis() % (4 \* 60 \* 60 \* 1000) < 1000) {

eggTiltAngle = (eggTiltAngle == 0) ? 45 : 0; // Alternate between 0 and 45 degrees

moveServo(eggTiltAngle);

}

}

// Function to update the LCD to display current and set temperature

void updateDisplay() {

lcd.setCursor(0, 0);

lcd.print("T:"); // current temperature

lcd.setCursor( 3 , 0);

lcd.print((topTemp + bottomTemp + middleTemp) / 3, 1);

lcd.print("C");

lcd.setCursor(8, 0);

lcd.print("H:"); // current Humidity

lcd.setCursor(11, 0);

lcd.print(humidity, 1);

lcd.print("%");

lcd.setCursor(0, 1);

lcd.print("PT:"); // set temperature

lcd.setCursor(4, 1);

lcd.print(setTemperature, 1);

lcd.setCursor(8, 1);

lcd.print("SH:"); // set Humidity

lcd.setCursor(12, 1);

lcd.print(setHumidity, 1);

}

// Function to handle button press for adjustments

void handleButtonPress() {

if (digitalRead(BUTTON\_TEMP\_UP) == HIGH) {

Serial.println("Temperature Incrementing");

setTemperature +=0.5;

delay(300); // Debounce button press

}

if (digitalRead(BUTTON\_TEMP\_DOWN) == HIGH) {

Serial.println("Temperature decrementing");

setTemperature -=0.5;

delay(300); // Debounce button press

}

if (digitalRead(BUTTON\_HUM\_UP) == HIGH) {

Serial.println("Humidity Incrementing");

setHumidity +=0.5;

delay(300); // Debounce button press

}

if (digitalRead(BUTTON\_HUM\_DOWN) == HIGH) {

Serial.println("Humidity decrementing");

setHumidity -=0.5;

delay(300); // Debounce button press

}

}

void loop() {

// Read sensors

readSensors();

// Control heating and fan

controlHeating();

// Control egg tilting

controlEggTilting();

// Update display

updateDisplay();

// Handle button press

handleButtonPress();

// delay(1000); // Wait 1 second between loops

}

### **6. Project Explanation**

* **Goal**: Automate an egg incubator to maintain optimal temperature, humidity, and egg rotation for successful hatching.
* **Sensor-Actuator Interaction**:
  + The temperature and humidity sensors provide real-time data to the STM32.
  + The microcontroller controls the heater and circulatory fan through relays based on temperature readings.
  + The servo motor periodically tilts the eggs for healthy development.
* **Code Block Purpose**:
  + **System Initialization**: Initializes all the necessary components (sensors, actuators, LCD).
  + **Read Sensors**: Collects temperature and humidity data for controlling the incubator.
  + **Control Heating**: Maintains the incubator’s temperature by toggling the heater and fan relays.
  + **Egg Tilting**: Periodically tilts the eggs by adjusting the servo motor’s position.
  + **Update Display**: Continuously updates the LCD with temperature and humidity information.
* **Overall System Functionality**:
  + The incubator automatically maintains optimal incubation conditions, allowing for real-time adjustments via the display and buttons.

**Challenges**

**Simulation**:

* + I couldn't find fritzing simulation evironment
  + Proteus 8.15 below was difficult to simulate stm32