

LF5 Projektarbeit

1. Halbjahr



24. Januar 2024

Oskar-von-miller Schule

FIc26

Daniel Tscheremnych

Inhaltsverzeichnis

[Projektbeschreibung 2](#_Toc156991840)

[Einbindung von Bibliotheken 2](#_Toc156991841)

[Pin-Definitionen 2](#_Toc156991842)

[Konstanten und Variablen 2](#_Toc156991843)

[Initialisierung von Sensor und Display 2](#_Toc156991844)

[setup() Funktion 3](#_Toc156991845)

[loop() Funktion 4](#_Toc156991846)

[Quellcode 5](#_Toc156991847)

## Projektbeschreibung

Diese Dokumentation bietet eine umfassende Beschreibung des Arduino-Codes, der zur Messung von Temperatur und Luftfeuchtigkeit in einem Warenlager der HerkulesFlight AG sowie zur Anzeige dieser Daten auf einem LCD und zur Steuerung von LEDs entsprechend der gemessenen Temperatur dient. Der Code verwendet einen DHT-Temperatur- und Feuchtigkeitssensor sowie ein I2C-LCD für die Anzeige.

## ****Einbindung von Bibliotheken****

1. **#include <DHT.h>**: Bindet die DHT-Bibliothek ein, die für die Interaktion mit dem DHT-Temperatur- und Feuchtigkeitssensor notwendig ist.
2. **#include <LiquidCrystal\_I2C.h>**: Bindet die Bibliothek für die Verwendung eines I2C-LCD-Displays ein.

## ****Pin-Definitionen****

* **#define DHTPIN 12**: Definiert den digitalen Pin 12 für die Datenverbindung zum DHT-Sensor.
* **#define DHTTYPE DHT22**: Spezifiziert den Typ des DHT-Sensors, in diesem Fall ein DHT22.
* **#define BLUE\_LED 6**, **#define RED\_LED 4**, **#define GREEN\_LED 5**: Definiert die Pins für die blauen, roten und grünen LEDs.
* **#define POTENTIOMETER A0**: Legt fest, dass das Potentiometer an den analogen Eingang A0 angeschlossen ist.
* **#define CLK 10 | #define DIO 11**: Digitale Pins für die Uhr (Clock) und Daten (Data) des TM1637-Displays (Pins 10 und 11).

## ****Konstanten und Variablen****

* **const int messIntervall = 5000**: Das Zeitintervall zwischen den Messungen in Millisekunden.
* **unsigned long letzteMessung = 0**: Speichert den Zeitpunkt der letzten Messung.
* **const float minTemp = 20.0**: Definiert die minimale Temperaturgrenze für den Überwachungsbereich.
* **float maxTemp = 24.0**: Die maximale Temperaturgrenze, initial gesetzt und später durch das Potentiometer einstellbar.

## ****Initialisierung von Sensor und Display****

* **DHT dht(DHTPIN, DHTTYPE)**: Erstellt ein DHT-Objekt.
* **LiquidCrystal\_I2C lcd(0x27, 16, 2)**: Initialisiert das LCD-Display. Die Adresse **0x27** und die Dimension **16x2** können je nach verwendetem Display variieren.

## ****setup() Funktion****

* Initialisiert die Pins für die LEDs als Ausgänge.

  pinMode(BLUE\_LED, OUTPUT);

  pinMode(RED\_LED, OUTPUT);

  pinMode(GREEN\_LED, OUTPUT);

* Startet die Kommunikation mit dem DHT-Sensor und dem LCD.

  dht.begin();

  lcd.init();

  lcd.backlight();

  display.begin();

* Beginnt die serielle Kommunikation mit einer Baudrate von 9600.

  Serial.begin(9600);

## ****loop() Funktion****

* Überprüft das Messintervall und liest die Daten vom DHT-Sensor, wenn das Intervall abgelaufen ist.

  if (millis() - letzteMessung >= messIntervall) {

    letzteMessung = millis();

* Liest die Luftfeuchtigkeit (**h**) und die Temperatur (**t**).

    float h = dht.readHumidity();

    float t = dht.readTemperature();

* Überprüft die Gültigkeit der Messwerte.

    if (isnan(h) || isnan(t)) {

      Serial.println("Fehler beim Lesen des Sensors!");

      return;

    }

* Liest den Wert des Potentiometers und setzt damit die maximale Temperaturgrenze, die dann auf dem TM1637-Display angezeigt wird.

    maxTemp = map(analogRead(POTENTIOMETER), 0, 1024, 21, 31);

    display.showNumber(maxTemp);

* Steuert die LEDs basierend auf der gemessenen Temperatur (blau für unterhalb der Mindesttemperatur, rot für oberhalb der Höchsttemperatur, grün für innerhalb des Bereichs).

    if (t < minTemp) {

      digitalWrite(BLUE\_LED, HIGH);

      digitalWrite(RED\_LED, LOW);

      digitalWrite(GREEN\_LED, LOW);

    } else if (t > maxTemp) {

      digitalWrite(BLUE\_LED, LOW);

      digitalWrite(RED\_LED, HIGH);

      digitalWrite(GREEN\_LED, LOW);

    } else {

      digitalWrite(BLUE\_LED, LOW);

      digitalWrite(RED\_LED, LOW);

      digitalWrite(GREEN\_LED, HIGH);

    }

* Zeigt die gemessenen Werte auf dem LCD an und sendet sie über die serielle Schnittstelle.

## Quellcode

/\*

 \* Beispiel:  Temperatur und Luftfeuchtigkeitsmesser mit LEDs und LCD für Klimabox

 \* Autor:     Daniel Tscheremnych

 \* Datum:     05/01/2024

 \*/

#include <DHT.h>

#include <LiquidCrystal\_I2C.h>

#include <TM1637TinyDisplay.h>

// Pin-Definitionen

#define DHTPIN 12          // Pin für den DHT-Sensor

#define DHTTYPE DHT22     // DHT 22 (oder DHT11)

#define BLUE\_LED 6

#define RED\_LED 4

#define GREEN\_LED 5

#define POTENTIOMETER A0  // Analogpin für das Potentiometer

#define CLK 10            // Clock Digitalpin für TM1637 Display

#define DIO 11            // Data Digitalpin für TM1637 Display

// Konstanten und Variablen

const int messIntervall = 5000; // Messintervall in Millisekunden

unsigned long letzteMessung = 0;

const float minTemp = 20.0;     // Minimale Temperatur für den Messbereich

float maxTemp = 24.0;           // Maximale Temperatur, wird durch das Potentiometer eingestellt

// Initialisierung von Sensor und Displays

DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal\_I2C lcd(0x27, 16, 2);

TM1637TinyDisplay display(CLK, DIO);

void setup() {

  pinMode(BLUE\_LED, OUTPUT);

  pinMode(RED\_LED, OUTPUT);

  pinMode(GREEN\_LED, OUTPUT);

  dht.begin();

  lcd.init();

  lcd.backlight();

  display.begin();

  Serial.begin(9600);

}

void loop() {

  if (millis() - letzteMessung >= messIntervall) {

    letzteMessung = millis();

    // Temperatur und Luftfeuchtigkeit lesen

    float h = dht.readHumidity();

    float t = dht.readTemperature();

    // Überprüfen, ob Messwerte gültig sind

    if (isnan(h) || isnan(t)) {

      Serial.println("Fehler beim Lesen des Sensors!");

      return;

    }

    // Maximalen Temperaturwert vom Potentiometer einlesen und auf TM1637 Display ausgeben

    maxTemp = map(analogRead(POTENTIOMETER), 0, 1024, 21, 31);

    display.showNumber(maxTemp);

    // LEDs steuern

    if (t < minTemp) {

      digitalWrite(BLUE\_LED, HIGH);

      digitalWrite(RED\_LED, LOW);

      digitalWrite(GREEN\_LED, LOW);

    } else if (t > maxTemp) {

      digitalWrite(BLUE\_LED, LOW);

      digitalWrite(RED\_LED, HIGH);

      digitalWrite(GREEN\_LED, LOW);

    } else {

      digitalWrite(BLUE\_LED, LOW);

      digitalWrite(RED\_LED, LOW);

      digitalWrite(GREEN\_LED, HIGH);

    }

    // Daten auf dem LCD anzeigen

    lcd.clear();

    lcd.setCursor(0, 0);

    lcd.print("r.F.: ");

    lcd.print(h);

    lcd.print(" %");

    lcd.setCursor(0, 1);

    lcd.print("Temp: ");

    lcd.print(t);

    lcd.print(" C");

    Serial.print("Luftfeuchtigkeit: ");

    Serial.print(h);

    Serial.print(" %\t");

    Serial.print("Temperatur: ");

    Serial.print(t);

    Serial.print(" \*C\t");

    Serial.print("Max. Temp:");

    Serial.print(maxTemp);

    Serial.println(" \*C");

  }

}