Weather Clock

*Echtzeit Datendarstellung auf dem Display

1st Imron Gamidli
Internationale Ingenieurwissenschaften
Hochschule Fulda
Fulda, Deutschland
imron.gamidli@lt.hs-fulda.de

2nd Hussain Ali
Internationale Ingenieurwissenschaften
Hochschule Fulda
Fulda, Deutschland
husain.ali@lt.hs-fulda.de

***Abstract*—**
***Index Terms*—dht22, arduino, uno,**

I. DHT22 Sensor

A. Beschreibung

Der DHT22 ist ein einfacher, preiswerter digitaler Temperatur- und Feuchtigkeitssensor. Er verwendet einen kapazitiven Feuchtigkeitssensor und einen Thermistor, um die Umgebungsluft zu messen, und spuckt ein digitales Signal auf dem Datenpin aus (keine analogen Eingangspins erforderlich). Er ist recht einfach zu bedienen, erfordert aber ein sorgfältiges Timing bei der Datenerfassung. Der einzige wirkliche Nachteil dieses Sensors ist, dass man nur alle 2 Sekunden neue Daten von ihm erhalten kann, so dass bei Verwendung unserer Bibliothek die Sensormesswerte bis zu 2 Sekunden alt sein können.

Verbinden Sie einfach den ersten Pin auf der linken Seite mit 3-5V Strom, den zweiten Pin mit Ihrem Dateneingangspin und den ganz rechten Pin mit Masse. Obwohl der Sensor ein einziges Kabel zum Senden von Daten verwendet, ist er nicht mit Dallas One Wire kompatibel! Wenn Sie mehrere Sensoren wünschen, muss jeder seinen eigenen Datenpin haben.

Im Vergleich zum DHT11 ist dieser Sensor präziser, genauer und funktioniert in einem größeren Temperatur-/Feuchtigkeitsbereich, ist aber auch größer und teurer.

Wird mit einem 4.7K - 10K Widerstand geliefert, den Sie als Pullup vom Daten-Pin zu VCC verwenden wollen.

B. Technische Daten

- Geringe Kosten
- 3 bis 5V Stromversorgung und E/A
- Maximal 2,5 mA Stromverbrauch während der Umwandlung (während der Datenabfrage)
- Gut geeignet für 0-100
- Gut geeignet für Temperaturmessungen von -40 bis 80°C mit einer Genauigkeit von ±0,5°C
- Abtastrate nicht mehr als 0,5 Hz (einmal alle 2 Sekunden)
- Gehäusegröße 27mm x 59mm x 13,5mm (1,05” x 2,32” x 0,53”)
- 4 Stifte, 0,1” Abstand
- Gewicht (nur der DHT22): 2,4g

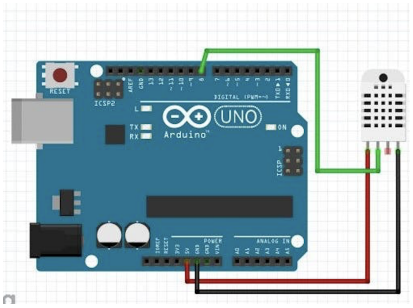

Fig. 1. DHT22 Sensor

Es ist ziemlich einfach, die DHT-Sensoren anzuschließen. Sie haben vier Stifte VCC - roter Draht Schließen Sie an 3,3 - 5V Strom an. Manchmal reicht die 3,3-V-Spannung nicht aus, in diesem Fall versuchen Sie es mit einer 5-V-Spannung. Datenausgang - weißes oder gelbes Kabel Nicht angeschlossen Masse - schwarzes Kabel Ignorieren Sie einfach Pin 3, er wird nicht verwendet. Dieses Diagramm zeigt, wie wir für die Testskizze anschließen werden. Schließen Sie Daten an Pin 2 an, Sie können dies später auf jeden beliebigen Pin ändern.

C. Implementierung im Projekt

DHT22 war nützlich für unseres Projekt. Mithilfe diesen Sensor, bekommen wir Temperatur und Feuchtigkeit Daten. Damit es möglich wird, Sensordaten zu lesen, bestimmte Bibliothek muss installiert und importiert soll. Die Bibliothek für DHT22 heißt *DHT sensor library* und installiert wurde vom https://github.com/adafruit/DHT-sensor-library . Nachdem wir die Bibliothek installiert haben, haben wir diese importiert.

```
#include <DHT.h>;
```

Diese Bibliothek ermöglicht uns eine DHT Instanz initialisieren. Hier man sieht wir haben PIN 13 für DHT Kommunikation. Danach haben wir DHT22 Instanz erstellt.

```
#define DHTPIN 13 // what pin we're connected to
#define DHTTYPE DHT22 // DHT 22
DHT dht(DHTPIN, DHTTYPE); // Initialize DHT
```

Wir werden später Daten aus dem Sensor in Variablen speichern. Dafür haben wir zwei Variablen, hum für Feuchtigkeit und temp für Temperatur deklariert.

```
float hum;  //Stores humidity value
float temp; //Stores temperature value
```

In setup() Funktion haben wir die DHT Instanz gestartet. Das heißt, die Instanz läuft und wir können jederzeit Feuchtigkeit und Temperaturdaten anfragen.

```
void setup()
{
  dht.begin();
}
```

In der Schleife loop() das Programm liest Feuchtigkeitswert mit readHumidity() und speichert in hum variable. Auf die gleiche Weise Temperatur wird mit readTemperature() gelesen und in temp gespeichert.

```
void loop()
{
  //Read data and store it to variables hum and temp
  hum = dht.readHumidity();
  temp = dht.readTemperature();

}
```

Letzte Aufgabe ist auf dem Display darzustellen, das vorher mit entsprechender Adresse initialisiert wurde. Wir haben zwei Displays, für DHT Daten haben wir erste Reihe des erstens Display benutzt.

```
  void loop()
  {
    //Read data and store it to variables hum and
    temp
    hum = dht.readHumidity();
    temp = dht.readTemperature();

    lcd1.setCursor(0, 0);
    lcd1.print("tmp:");
    lcd1.print(temp);
    lcd1.setCursor(6, 0);
    lcd1.print((char)223);
    lcd1.print("C hum:");
    lcd1.print(hum);
    lcd1.setCursor(15, 0);
    lcd1.print("%");

  }
```

II. PIR Motion Sensor

A. Überblick

PIR-Sensoren ermöglichen die Erfassung von Bewegungen und werden fast immer verwendet, um festzustellen, ob sich ein Mensch in den Sensorbereich hinein- oder herausbewegt hat. Sie sind klein, preiswert, stromsparend, einfach zu bedienen und verschleißen nicht. Aus diesem Grund werden sie häufig in Geräten und Vorrichtungen in Haushalten und Unternehmen eingesetzt.

PIR-Sensoren bestehen im Wesentlichen aus einem pyroelektrischen Sensor (den Sie unten als runde Metalldose mit einem rechteckigen Kristall in der Mitte sehen können), der die Stärke der Infrarotstrahlung erkennen kann. Alles gibt eine schwache Strahlung ab, und je heißer etwas ist, desto mehr Strahlung wird freigesetzt. Der Sensor in einem Bewegungsmelder ist eigentlich in zwei Hälften geteilt. Der Grund dafür ist, dass wir eine Bewegung (Veränderung) und

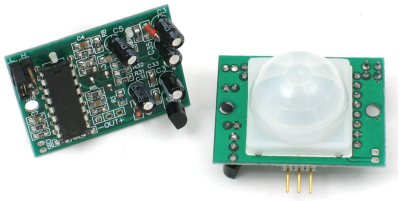

Fig. 2. PIR Sensor

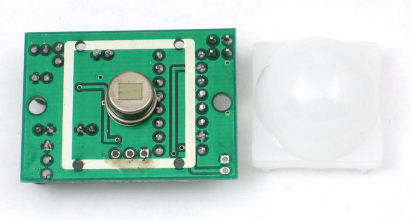

Fig. 3. PIR Sensor 1

nicht die durchschnittliche IR-Stärke erkennen wollen. Die beiden Hälften sind so verdrahtet, dass sie sich gegenseitig aufheben. Wenn eine Hälfte mehr oder weniger IR-Strahlung sieht als die andere, schwankt der Ausgang nach oben oder unten.

Zusammen mit dem pyroelektrischen Sensor gibt es eine Reihe von unterstützenden Schaltungen, Widerständen und Kondensatoren. Es scheint, dass die meisten kleinen Bastler-Sensoren den BISS0001 verwenden, zweifellos ein sehr preiswerter Chip. Dieser Chip nimmt das Ausgangssignal des Sensors auf und verarbeitet es ein wenig, um einen digitalen Ausgangsimpuls aus dem analogen Sensor zu erzeugen.

Im Vergleich zu älteren PIRs haben die neuen PIRs mehr einstellbare Einstellungen und sind mit einer Stiftleiste ausgestattet, die in den 3-poligen Masse-/Ausgangs-/Stromanschlüssen installiert ist.

Für viele einfache Projekte oder Produkte, bei denen erkannt werden muss, ob eine Person den Bereich verlassen oder betreten hat oder sich ihm genähert hat, sind PIR-Sensoren hervorragend geeignet. Sie sind stromsparend und kostengünstig,

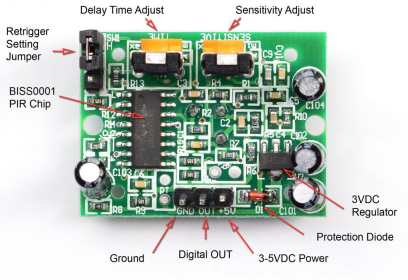

Fig. 4. PIR Sensor Einstellungen

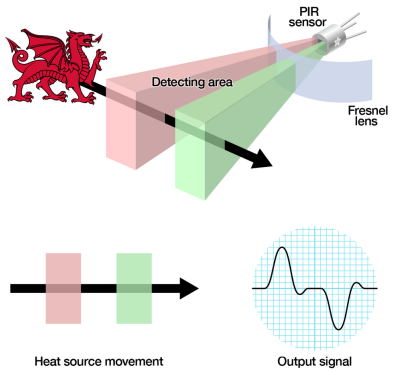

Fig. 5. PIR Sensor Funktionsweise

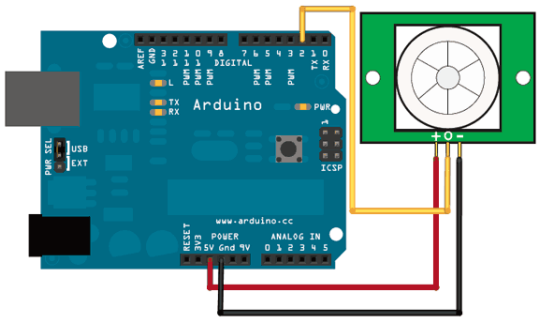

Fig. 6. PIR Sensor Funktionsweise

ziemlich robust, haben einen großen Linsenbereich und sind einfach zu bedienen. PIR-Sensoren zeigen nicht an, wie viele Personen sich in der Nähe aufhalten oder wie nah sie am Sensor sind. Die Linse ist oft auf einen bestimmten Bereich und eine bestimmte Entfernung festgelegt (obwohl sie irgendwo gehackt werden kann), und manchmal werden sie auch von Haustieren ausgelöst.

B. Wie PIRs funktionieren

PIR-Sensoren sind komplizierter als viele der anderen Sensoren, die in diesen Tutorials erklärt werden (wie Fotozellen, FSRs und Kippschalter), weil es mehrere Variablen gibt, die den Eingang und Ausgang des Sensors beeinflussen. Um zu erklären, wie ein grundlegender Sensor funktioniert, verwenden wir dieses schöne Diagramm

Der PIR-Sensor selbst hat zwei Schlitze. Jeder Schlitz besteht aus einem speziellen Material, das für IR empfindlich ist. Die Linse, die hier verwendet wird, macht nicht wirklich viel, und so sehen wir, dass die beiden Schlitze über eine gewisse Entfernung hinaus ”sehen” können (im Grunde die Empfindlichkeit des Sensors). Wenn sich der Sensor im Leerlauf befindet, erkennen beide Schlitze die gleiche Menge an IR, nämlich die vom Raum, den Wänden oder der Außenumgebung abgestrahlte Menge. Wenn ein warmer Körper, z. B. ein Mensch oder ein Tier, vorbeikommt, fängt er zunächst eine Hälfte des PIR-Sensors ab, was eine positive Differenzänderung zwischen den beiden Hälften bewirkt. Wenn der warme Körper den Erfassungsbereich verlässt, geschieht das Gegenteil, wobei der Sensor eine negative Differenzänderung erzeugt. Diese Änderungsimpulse sind das, was erkannt wird.

C. Implementierung im Projekt

TODO

```
int inputPin = 2; // choose the input pin (for PIR sensor)
int pirState = LOW; // we start, assuming no motion detected
int val = 0;  // variable for reading the pin status

void setup()
{
  pinMode(inputPin, INPUT); // declare sensor as input

}

void loop()
{
  val = digitalRead(inputPin);  // read input value
  if (val == HIGH)
  {
    lcd.backlight();
    lcd1.backlight();
    if (pirState == LOW)
    {
      // we have just turned on
      Serial.println("Motion detected!");
      // We only want to print on the output change, not state
      pirState = HIGH;
    }
  }
  else
  {
    lcd.noBacklight();
    lcd1.noBacklight();
    if (pirState == HIGH)
    {
      // we have just turned of
      Serial.println("Motion ended!");
      // We only want to print on the output change, not state
      pirState = LOW;
    }
  }
}
```

REFERENCES

[1] https://github.com/adafruit/DHT-sensor-library
[2] https://github.com/semestrinis/Arduino/wiki/DHT22-temperature-humidity-sensor
[3] https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview
[4] https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work
[5] https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/connecting-to-a-pir
[6]
[7]