Lab-01

Course Title: Internet of Things

Course Code: CSE406

Section: 01

Submitted By:

Name: Md. Samir Hossain

ID: 2022-3-60-161

Semester: Summer-2025

Submitted To:

Dr. Raihan Ul Islam

Associate Professor

Department of Computer Science & Engineering

Date: 30-06-2025

1. Arduino Blink Project

Introduction

The Blink project is a fundamental Arduino example that toggles the built-in LED on and off at regular intervals. It serves as a basic test to confirm that the board is properly set up and functioning.

Components

- Arduino Uno
- Built-in LED (pin 13)

Code

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

Explanation

The LED is configured as an output and toggled on and off every second using digitalWrite() and delay(). This loop continues indefinitely, causing the LED to blink.

Conclusion

This project validates the basic operation of the Arduino board and introduces core concepts like pin control and timing.

2. Arduino DHT11 Sensor Project

Introduction

This project demonstrates how to interface a DHT11 temperature and humidity sensor with an Arduino board. The sensor reads ambient temperature and humidity, then displays the results, including the calculated heat index, via the Serial Monitor.

Components

- Arduino Uno
- DHT11 Sensor
- Jumper wires

Circuit Diagram

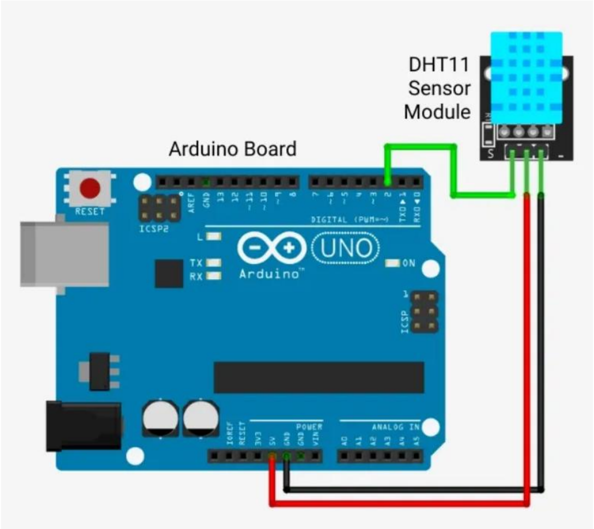


Figure 1: DHT11 Circuit Diagram

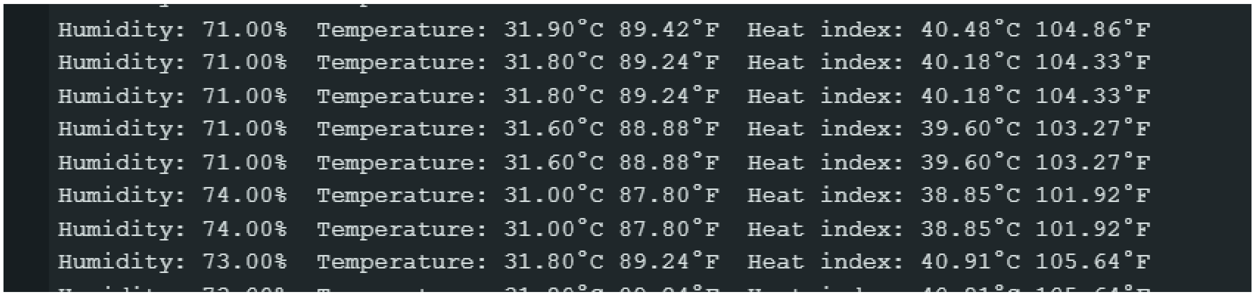
Code

```
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHT11 test"));
  dht.begin();
}

void loop() {
  delay(1000);

  float h = dht.readHumidity();
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);

  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  float hif = dht.computeHeatIndex(f, h);
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

Explanation

- The DHT11 sensor reads humidity and temperature values.
- Temperature is provided in both Celsius and Fahrenheit.
- The code also calculates the heat index, indicating the "feels-like" temperature based on humidity.
- Results are displayed every second using the Serial Monitor.

Output



```
Humidity: 71.00%  Temperature: 31.90°C 89.42°F  Heat index: 40.48°C 104.86°F
Humidity: 71.00%  Temperature: 31.80°C 89.24°F  Heat index: 40.18°C 104.33°F
Humidity: 71.00%  Temperature: 31.80°C 89.24°F  Heat index: 40.18°C 104.33°F
Humidity: 71.00%  Temperature: 31.60°C 88.88°F  Heat index: 39.60°C 103.27°F
Humidity: 71.00%  Temperature: 31.60°C 88.88°F  Heat index: 39.60°C 103.27°F
Humidity: 74.00%  Temperature: 31.00°C 87.80°F  Heat index: 38.85°C 101.92°F
Humidity: 74.00%  Temperature: 31.00°C 87.80°F  Heat index: 38.85°C 101.92°F
Humidity: 73.00%  Temperature: 31.80°C 89.24°F  Heat index: 40.91°C 105.64°F
```

Figure 2: Serial Monitor Output (DHT11)

Conclusion

This project effectively demonstrates reading environmental conditions using the DHT11 sensor. It introduces data acquisition, sensor integration, and basic serial communication in Arduino-based systems.

3. Arduino Water Level Sensor Project

Introduction

This project uses an analog water level sensor connected to an Arduino to monitor water levels. The sensor reads analog values representing water presence and interprets the level as high, medium or low, displaying the result through the Serial Monitor.

Components

- Arduino Uno
- Analog Water Level Sensor
- Jumper wires

Circuit Diagram

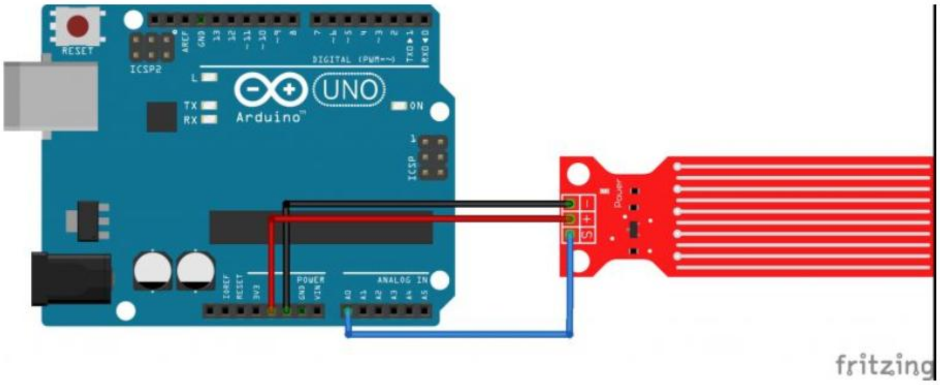


Figure 3: Water Level Sensor Circuit Diagram

Code

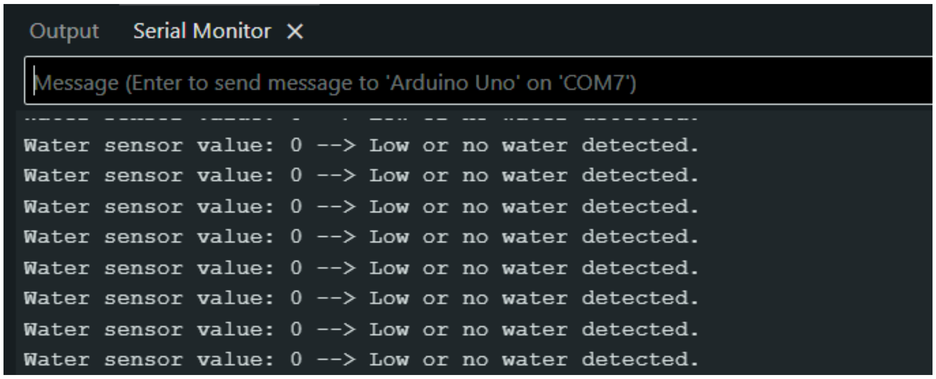
```
void setup() {
  Serial.begin(9600);
  Serial.println("Water sensor reading started...");
}
```

```
void loop() {
  int sensorValue = analogRead(A0);

  Serial.print("Water sensor value: ");
  Serial.print(sensorValue);

  if (sensorValue > 600) {
    Serial.println(" --> High water level detected.");
  } else if (sensorValue > 300) {
    Serial.println(" --> Medium water level.");
  } else {
    Serial.println(" --> Low or no water detected.");
  }

  delay(1000);
}
```

Explanation

- The sensor outputs an analog value based on the water level.
- The code reads this value from analog pin A0.
- Based on predefined thresholds, it classifies the level and prints a message to the Serial Monitor.

Output



```
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
Water sensor value: 0 --> Low or no water detected.
```

Figure 4: Serial Monitor Output (Water Level Readings)

Conclusion

This project provides a simple way to detect water levels using an analog sensor and Arduino. It is useful in monitoring applications such as tanks or irrigation systems.