**Arduino IDE Interfacing**

***Submitted by***

**HARSHITHA K(2022BECE07AED023)**

**in partial fulfillment of the award of the degree**

**BACHELOR OF TECHNOLOGY  
 IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

Under the supervision of

**Dr.Sridhar Thota**

**A logo of a university

AI-generated content may be incorrect.**

**ALLIANCE SCHOOL OF APPLIED ENGINEERING**

**ALLIANCE UNIVERSITY BANGLORE-562106**

**JUNE 2025**

**1.Sound Sensor with LED**

**Aim :** To design and implement LED system using a sound sensor and Arduino uno that turns on an LED when the sound level reach the certain threshold.

**Components Required:**

|  |  |
| --- | --- |
| Arduino Uno |  |
| Sound Sensor Module |  |
| LED |  |

* Breadboard
* 220Ω Resistor
* Jumper Wires
* USB Cable
* Arduino IDE software

**Theory:** A sound sensor detects vibrations or changes in air pressure (sound waves) using a microphone. In this project, the sound sensor outputs an analog signal that varies with sound intensity. The Arduino reads this signal through an analog input pin. When the sound intensity crosses a certain threshold, the Arduino sends a HIGH signal to an LED, turning it ON.

**Applications:**

* It can be used in sound-activated door systems to enhance accessibility.
* The setup is useful in voice-controlled robotics for basic commands.
* It helps build sound-reactive LED displays for decoration or entertainment.
* It can be integrated into industrial environments to detect abnormal machine noise.
* It is ideal for creating noise-sensitive streetlights that activate in busy areas.

**Circuit :**

| **Component Pin** | **Connects To on Arduino** | **Purpose** |
| --- | --- | --- |
| Sound Sensor VCC | 5V | Power supply to sensor |
| Sound Sensor GND | GND | Ground |
| Sound Sensor A0 | A0 | Analog signal output |
| LED Anode (+) | Digital Pin 13 (via 220Ω resistor) | Controls LED from Arduino |
| LED Cathode (–) | GND | Ground connection |

**Arduino Code:**

const int soundPin = A0; // Analog pin connected to sound sensor output

const int ledPin = 13; // LED connected to digital pin 13

int soundValue = 0; // Variable to store sound sensor value

void setup() {

pinMode(ledPin, OUTPUT); // Set LED pin as output

Serial.begin(9600); // Initialize serial communication for debugging

}

void loop() {

soundValue = analogRead(soundPin); // Read sound sensor value (0-1023)

Serial.println(soundValue); // Print value to Serial Monitor

// If sound exceeds threshold, turn on LED

if (soundValue > 500) {

digitalWrite(ledPin, HIGH);

} else {

digitalWrite(ledPin, LOW);

}

delay(100); // Small delay for stability

}

**Procedure:**

* Connect the VCC, GND, and A0 pins of the sound sensor to 5V, GND, and A0 on the Arduino respectively.
* Connect the LED anode to digital pin 13 through a 220Ω resistor, and the cathode to GND.
* Plug the Arduino into your computer using a USB cable.
* Open the Arduino IDE, paste the provided code, and select the correct board and COM port.
* Upload the code and open the Serial Monitor to view sound sensor readings.
* Make a sound near the sensor and observe the LED turning ON when the reading exceeds the threshold**.**

**2.Temperature Sensor with LED**

**Aim:** To read temperature using a sensor and turn an LED ON when the temperature crosses a set threshold.

**Components Required:**

* Arduino Uno
* Temperature Sensor (DHT11)



* LED
* 220Ω Resistor
* Jumper Wires
* Breadboard
* USB Cable

**Theory:** The DHT11 is a low-cost digital sensor that measures both temperature and humidity. It contains a thermistor to measure temperature and a capacitive humidity sensor. The sensor outputs data as a digital signal that can be read by microcontrollers like Arduino. The Arduino reads the temperature data from the DHT11 sensor. When the temperature rises above a specified threshold (e.g., 30°C), the Arduino switches ON an LED to indicate that the temperature is high. This simple feedback mechanism can be used in various temperature-monitoring applications.

**Applications:**

* Room temperature monitoring and alert systems.
* Part of home automation to control fans or air conditioners.
* Safety systems that warn when equipment or environment overheats.

**Circuit Connections:**

| **DHT11 Pin** | **Connects To Arduino Pin** | |
| --- | --- | --- |
| VCC | 5V | |
| GND | GND | |
| DATA | Digital Pin 2 | |
| **LED Pin** | | **Connects To Arduino Pin** | |
| Anode (long leg) | | Digital Pin 13 via 220Ω resistor | |
| Cathode (short leg) | | GND | |

**Arduino Code:**

#include "DHT.h"

#define DHTPIN 2 // Digital pin connected to the DHT11 sensor

#define DHTTYPE DHT11 // DHT 11 sensor type

DHT dht(DHTPIN, DHTTYPE);

const int ledPin = 13; // LED connected to digital pin 13

const float tempThreshold = 30.0; // Temperature threshold in Celsius

void setup() {

Serial.begin(9600);

dht.begin();

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, LOW); // Ensure LED is off initially

}

void loop() {

float temperature = dht.readTemperature();

if (isnan(temperature)) {

Serial.println("Failed to read from DHT sensor!");

} else {

Serial.print("Temperature: ");

Serial.print(temperature);

Serial.println(" °C");

if (temperature > tempThreshold) {

digitalWrite(ledPin, HIGH); // Turn LED ON

} else {

digitalWrite(ledPin, LOW); // Turn LED OFF

}

}

delay(2000); // Wait 2 seconds before next reading

}

**Procedure:**

* Gather all components: Arduino Uno, DHT11 sensor, LED, resistor, jumper wires, breadboard, and USB cable.
* Connect the DHT11 sensor: VCC to 5V, GND to GND, and Data pin to Arduino digital pin 2.
* Connect the LED: Anode (long leg) to Arduino digital pin 13 via a 220Ω resistor, and cathode (short leg) to GND.
* Open the Arduino IDE and install the DHT sensor library by Adafruit if not already installed.
* Write or upload the provided Arduino code that reads temperature from the sensor and controls the LED based on the threshold.
* Upload the code to the Arduino board via USB connection.
* Open the Serial Monitor to view real-time temperature readings.
* Test the setup by increasing the ambient temperature (e.g., by warm breath or nearby heat source) and observe the LED turning ON when temperature crosses the threshold.
* Adjust threshold value in code if needed for different temperature sensitivity.

**3.LDR with LED**

**Aim:** To interface an LDR (Light Dependent Resistor) with an LED using Arduino, so that the LED automatically turns ON in darkness and OFF in light, using digital input.

**Components Required:**

* Arduino Uno
* LDR (Photoresistor)
* 1 LED

**Code:**

int ldrDigitalPin = 7;

int ledPin = 9;

int a=0;

void setup() {

  pinMode(ldrDigitalPin, INPUT);

  pinMode(ledPin, OUTPUT);

  Serial.begin(9600);

}

void loop() {

 /\* int ldrState = digitalRead(ldrDigitalPin);

  Serial.print("LDR Digital Reading: ");

  Serial.println(ldrState);

  if (ldrState == LOW) {

    digitalWrite(ledPin, HIGH);

    Serial.println("It's dark → LED ON");

  } else {

    digitalWrite(ledPin, LOW);

    Serial.println("It's bright → LED OFF");

  }\*/

int a = analogRead(A1);

Serial.println(a);

  delay(500);

}

**How it works:**

* The Arduino reads the voltage on the LDR pin. When it’s dark, resistance of LDR is high → voltage on A0 drops → analog Read returns low value → LED turns ON.
* When it’s bright, resistance is low → voltage on A0 rises → LED turns OFF.

**4.DHT11 with LED (analog to temperature convertion)**

**Aim :** To interface an analog temperature sensor with an Arduino and display the temperature in Celsius and Fahrenheit on the Serial Monitor.

**Components Required:**

* Arduino UNO
* temperature sensor
* USB cable for Arduino
* Computer with Arduino IDE

**Circuit Connections:**

| **Sensor** | **Connection** |
| --- | --- |
| VCC (Pin 1) | +5V on Arduino |
| OUT (Pin 2) | A0 (Analog Pin 0 on Arduino) |
| GND (Pin 3) | GND on Arduino |

**Code:**

const int PIN\_DHT11 = A0; // input pin

void setup() {

Serial.begin(9600); // Initialize serial communication

}

void loop() {

int adcVal = analogRead(PIN\_DHT11); // Read analog value

float voltage = adcVal \* (5.0 / 1024.0); // Convert ADC value to voltage

float tempC = (voltage - 0.5) \* 100.0;

float tempC = voltage \* 100;

float tempF = tempC \* 9.0 / 5.0 + 32.0; // Convert °C to °F

// Display all values

Serial.print("ADC Value: ");

Serial.print(adcVal);

Serial.print(" Voltage: ");

Serial.print(voltage, 3);

Serial.print(" V Temp: ");

Serial.print(tempC, 2);

Serial.print(" °C ");

Serial.print(tempF, 2);

Serial.println(" °F");

delay(1000); // Delay for 1 second

}

**Procedure:**

1. **Connect** the sensor as per the table above.
2. **Open** Arduino IDE and paste the corrected code.
3. **Select** the correct board and COM port.
4. **Upload** the code to Arduino.
5. **Open Serial Monitor** (set to 9600 baud).
6. **Observe** real-time temperature readings in °C and °F.

**Ultrasonic Sensor**

#define TRIG\_PIN 9

#define ECHO\_PIN 10

void setup() {

  Serial.begin(9600);         // Serial output for measurement

  pinMode(TRIG\_PIN, OUTPUT);

  pinMode(ECHO\_PIN, INPUT);

}

void loop() {

  long duration;

  float distance;

  // Trigger pulse

  digitalWrite(TRIG\_PIN, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN, LOW);

  // Echo response

  duration = pulseIn(ECHO\_PIN, HIGH);

  // Calculate distance in cm

  distance = duration \* 0.034 / 2;

  // Print live update

  Serial.print("Distance: ");

  Serial.print(distance);

  Serial.println(" cm");

  delay(100);  // Short delay for quick updates (adjust if needed)

}