Skill-1

Title: Interfacing keypad, pushbutton, buzzer and display (LED, LCD, 8x8 matrix LED) with Arduino

Skills/Competencies Acquired:

1. Arduino

2. Use of Keypad

3. Use of Push Button

4. Use of buzzer

5. Use of LED/LCD Display.

Purpose of this Skill

The purpose of this activity is to learn and demonstrate how to interface various components like a keypad, pushbutton, buzzer, and different types of displays (LED, LCD, 8x8 matrix LED) with an Arduino. By completing this project, you will gain hands-on experience in controlling input and output devices using the Arduino platform.

Steps performed in this Skill

Step 1: Setting up the Arduino environment

Step 2: Connecting the Keypad

Step 3: Connecting the Pushbutton

Step 4: Connecting the Buzzer

Step 5: Interfacing the LED Display

Resources used in this skill

Arduino Uno

Arduino IDE

Keypad

Pushbutton

Buzzer

7-segment LED Display

LCD Display

Jumper wires and Breadboard

Skill-2

Aim: Interfacing various sensors and actuators with Arduino

Skills/Competencies Acquired:

1. Arduino

2. Arduino IDE

3. DHT 11 Sensor

4. IR Sensor

5. DC Motor

Purpose of Skill:

The purpose of this skill is to integrate and control sensors (such as DHT 11, IR) and actuators (such as a DC motor) using an Arduino. This will enable real-time sensing, processing, and control of physical systems. Understanding this skill involves working with microcontrollers to read sensor data, process it, and then interact with physical devices like motors or other actuators.

Steps to Perform the Skill:

1. Setting Up Arduino and Arduino IDE:

\* Install the Arduino IDE from ([https://www.arduino.cc/en/software](https://www.arduino.cc/en/software)).

\* Connect your Arduino board (like Arduino Uno) to your computer using a USB cable.

\* Open the Arduino IDE and select the correct board and port under the Tools menu.

2. Interfacing the DHT 11 Sensor:

\* The DHT 11 sensor is a digital sensor that outputs temperature and humidity data.

\* Connect the sensor to the Arduino:

\* VCC to 5V

\* GND to Ground

\* Data Pin to a digital pin (for example, pin 2).

\* Install the DHT sensor library in Arduino IDE: Sketch > Include Library > Manage Libraries > search for DHT sensor library and install it.

\* Use the library functions to read and display data from the sensor.

3. Interfacing the IR Sensor:

\* An IR sensor detects infrared light, and can be used for motion sensing or object detection.

\* Connect the IR sensor to Arduino:

\* VCC to 5V

\* GND to Ground

\* OUT Pin to a digital pin (for example, pin 3).

Skill-3:

Aim: Interfacing Keypad, Pushbutton, Buzzer, and Display (LED, LCD, 8x8 Matrix LED) with Raspberry Pi

Skills and Competencies Acquired:

Understanding GPIO pins

Working with libraries like RPi.GPIO

Reading input from a pushbutton

Controlling output devices

Displaying text and patterns on LED

Writing Python scripts in RPi

Purpose of this Skill:

To interface various input and output devices with a Raspberry Pi, enabling users to build interactive projects. By mastering these interfaces, one can develop applications like security systems, interactive displays, embedded control systems, and IoT applications.

Resources Used in this Skill:

Raspberry Pi, pushbutton, keypad, LED, buzzer, LCD, 8x8 LED matrix, jumper wires, breadboard, Raspbian OS, Python, Geany Python IDE

Steps Performed in this Skill:

1. Interfacing a Pushbutton with Raspberry Pi:

Connect a pushbutton to the Raspberry Pi GPIO using a pull-up or pull-down resistor.

Write a Python script using the RPi.GPIO library to detect button presses.

Print messages or trigger actions (e.g., turning on an LED) upon button press.

2. Interfacing a Keypad with Raspberry Pi:

Use a 4x4 or 3x4 matrix keypad.

Connect the keypad rows and columns to the GPIO pins.

Write a Python script to scan key presses and decode the input using the Adafruit\\_MCP230xx library or custom logic.

3. Interfacing a Buzzer with Raspberry Pi:

Connect a piezo or active buzzer to a GPIO pin.

Use PWM (Pulse Width Modulation) to control sound patterns.

Write a Python script to generate beeps or tones on specific triggers.

4. Interfacing an LED with Raspberry Pi:

Connect an LED with a current-limiting resistor to a GPIO pin.

Write a Python script to turn the LED on/off or blink it in different patterns.

5. Interfacing an LCD Display with Raspberry Pi:

Use an I2C or parallel 16x2 LCD.

Connect the display to the Raspberry Pi’s I2C pins.

Install and use the Adafruit\\_CharLCD library for controlling the display.

Write a Python script to display text messages on the LCD.

6. Interfacing an 8x8 LED Matrix with Raspberry Pi:

Use a MAX7219-based 8x8 LED matrix.

Connect it using SPI (Serial Peripheral Interface) pins.

Install and use the luma.led\\_matrix Python library to control the display.

Write a Python script to display patterns, scrolling text, or animations.

Skill-4:

Aim: Interfacing sensors and actuators with Raspberry PI

Skills/Competencies Acquired:

Raspberry Pi

Reading data from analog and digital sensors

Understanding GPIO pin configurations for sensors and actuators

Using Python libraries for interfacing with sensors and actuators

Writing Python scripts for real-time data acquisition and control

Purpose of this Skill:

This skill enables users to interface various sensors and actuators with a Raspberry Pi to develop real-world applications such as automation, robotics, and IoT-based systems. By mastering these interfaces, users can build projects involving environmental monitoring, motor control, and home automation.

Steps Performed in this Skill:

1. Interfacing a Temperature and Humidity Sensor (DHT11/DHT22) with Raspberry Pi:

Connect the sensor’s VCC to 3.3V, GND to ground, and DATA to a GPIO pin.

Install the Adafruit\\_DHT library to read temperature and humidity data.

Python Code Example:

import Adafruit\\_DHT

sensor = Adafruit\\_DHT.DHT11

pin = 4 # GPIO pin connected to the sensor

humidity, temperature = Adafruit\\_DHT.read\\_retry(sensor, pin)

print(f"Temperature: {temperature}C, Humidity: {humidity}%")

2. Interfacing a PIR Motion Sensor with Raspberry Pi:

Connect VCC to 5V, GND to ground, and OUT to a GPIO pin.

Use RPi.GPIO to detect motion and trigger an action.

Python Code Example:

import RPi.GPIO as GPIO

import time

PIR\\_PIN = 17

GPIO.setmode(GPIO.BCM)

GPIO.setup(PIR\\_PIN, GPIO.IN)

while True:

if GPIO.input(PIR\\_PIN):

print("Motion Detected!")

time.sleep(1)

3. Interfacing a Relay Module with Raspberry Pi:

Connect VCC to 5V, GND to ground, and IN to a GPIO pin.

Use Python to switch the relay ON and OFF.

Python Code Example:

import RPi.GPIO as GPIO

import time

RELAY\\_PIN = 18

GPIO.setmode(GPIO.BCM)

GPIO.setup(RELAY\\_PIN, GPIO.OUT)

GPIO.output(RELAY\\_PIN, GPIO.HIGH) # Turn relay ON

time.sleep(2)

GPIO.output(RELAY\\_PIN, GPIO.LOW) # Turn relay OFF

GPIO.cleanup()

4. Interfacing a Servo Motor with Raspberry Pi:

Connect the servo’s power and ground, and control pin to a GPIO pin.

Use PWM (Pulse Width Modulation) to control the servo position.

Python Code Example:

import RPi.GPIO as GPIO

import time

SERVO\\_PIN = 12

GPIO.setmode(GPIO.BCM)

GPIO.setup(SERVO\\_PIN, GPIO.OUT)

pwm = GPIO.PWM(SERVO\\_PIN, 50)

pwm.start(7.5)

pwm.ChangeDutyCycle(2.5) # Move servo to 0 degrees

time.sleep(1)

pwm.ChangeDutyCycle(12.5) # Move servo to 180 degrees

time.sleep(1)

pwm.stop()

GPIO.cleanup()

5. Interfacing an Ultrasonic Sensor (HC-SR04) with Raspberry Pi:

Connect VCC to 5V, GND to ground, TRIG to a GPIO pin, and ECHO to another GPIO pin.

Use Python to calculate the distance based on the time delay between the signal sent and received.

Python Code Example:

import RPi.GPIO as GPIO

import time

TRIG = 23

ECHO = 24

GPIO.setmode(GPIO.BCM)

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

GPIO.output(TRIG, True)

time.sleep(0.00001)

GPIO.output(TRIG, False)

while GPIO.input(ECHO) == 0:

start\\_time = time.time()

while GPIO.input(ECHO) == 1:

end\\_time = time.time()

distance = ((end\\_time - start\\_time) \\* 34300) / 2

print(f"Distance: {distance} cm")

GPIO.cleanup()

Resources Used in this Skill:

Hardware: Raspberry Pi (any model with GPIO support), DHT11/DHT22 sensor, PIR motion sensor, relay module, servo motor, ultrasonic sensor, jumper wires, breadboard, resistors

Software & Libraries: Raspbian OS, Python, RPi.GPIO, Adafruit\\_DHT

Development Tools: Thonny Python IDE, Terminal for running scripts

Skill-5:

Interface ESP32 with DHT11 sensor and upload data to a webpage using Blynk

Skills/Competencies Acquired:

Interfacing of DHT11 sensor

Working with Blynk

Configuring ESP32 Wi-Fi

Writing Arduino code

Sending sensor data to cloud

Purpose of the Activity:

This skill enables learners to interface temperature, humidity, and pressure sensors with an ESP32 Wi-Fi module and transmit real-time data to the Blynk cloud platform. The data can be monitored remotely through a web dashboard or mobile app, making it useful for weather monitoring, home automation, and IoT projects.

Steps Performed in this Activity:

1. Hardware Connections

Connect the DHT11 sensor to the ESP32 GPIO pins.

Ensure proper power supply (3.3V for ESP32, 3.3V/5V for sensors).

Use pull-up resistors where necessary

2. Software Setup

Install Arduino IDE (or use MicroPython on Thonny IDE).

Install required libraries (DHT, Adafruit BMP/BME, BlynkSimpleEsp32).

Configure Blynk project and get an authentication token.

3. Writing the Code

Initialize ESP32 and connect it to Wi-Fi.

Read temperature, humidity, and pressure from the sensor.

Send the data to Blynk cloud using Blynk.virtualWrite().

4. Upload and Run

Compile and upload the code to ESP32.

Monitor sensor values on the Blynk app.

5. Real-Time Monitoring

Use Blynk widgets to display live temperature, humidity, and pressure.

Enable graph visualization for tracking trends over time.

Resources & Tools Used:

ESP32 (NodeMCU)

DHT11 (Temperature & Humidity Sensor) or

Jumper wires and Breadboard

Arduino IDE with ESP32 Board Package

Blynk IoT App (Android/iOS)

Serial Monitor (for debugging sensor readings)

Blynk Web Dashboard for online monitoring

Skills Acquired:

✅ IoT Integration

✅ Embedded Systems Programming

✅ Wireless Communication

✅ Data Visualization

✅ Sensor Interfacing

Skill-6

Title: Interfacing ESP8266 with DHT11 Sensor and Uploading Data to ThingSpeak Webpage

Purpose of the Activity:

This activity aims to demonstrate the process of collecting environmental data (temperature, humidity, pressure) using sensors like DHT11 or BMP/BME280 and transmitting that data to ThingSpeak, a cloud-based IoT analytics platform. It builds hands-on skills in sensor interfacing, Wi-Fi-based communication using ESP8266, and cloud data visualization.

Steps Performed in This Activity:

1. Hardware Setup:

Connect DHT11 or BMP/BME sensor to the ESP8266 board.

2. Install below Libraries in Arduino IDE:

DHT sensor library

ThingSpeak library

ESP8266WiFi library

3. Setup ThingSpeak:

Create an account on ThingSpeak.

Create a new channel and add fields (e.g., temperature, humidity).

Note your Write API Key.

4. Write and Upload Arduino Code:

Connect ESP8266 to Wi-Fi.

Read sensor data.

Use ThingSpeak.writeField() to send data to the cloud.

Upload code and observe sensor data being updated in real-time on ThingSpeak dashboard.

5. Monitor and Validate Output:

Open Serial Monitor to check for connectivity and sensor reading.

Login to ThingSpeak and view live graphs.

Resources and Tools Used:

1. ESP8266 NodeMCU

2. DHT11 Sensor

3. Jumper wires

4. Breadboard

5. USB Cable

6. Arduino IDE

7. ThingSpeak Platform

Skills and Competencies Acquired:

1. Interfacing ESP8266 with DHT11

2. Connecting ESP8266 to Wifi

3. Sending sensor data to ThingSpeak

4. Creating ThingSpeak channels

5. Real-time visualization of IoT data