**Lab Exercise 5- Controlling an External LED with Serial Input on Arduino**

In this lab exercise, you will learn how to control an external LED using serial input from the Arduino Serial Monitor. This exercise will involve connecting an LED to your Arduino board and writing a simple program that turns the LED on and off based on the commands you send from the Serial Monitor.

**Objective:**

* Control an external LED connected to the Arduino using commands sent via the Serial Monitor.

**Materials Needed:**

* Arduino board (e.g., Arduino Uno)
* USB cable to connect the Arduino to your computer
* 1x LED
* 1x 220Ω resistor
* Breadboard (optional, but recommended)
* Jumper wires

**Circuit Diagram:**

1. Connect the components as follows:
   * Connect the **anode** (longer leg) of the LED to **Digital Pin 9** on the Arduino.
   * Connect the **cathode** (shorter leg) of the LED to one end of the **220Ω resistor**.
   * Connect the other end of the resistor to **Ground (GND)** on the Arduino.

Here’s a simple circuit diagram:

Arduino Pin 9 ----->|----[220Ω]-----> GND

LED

**Steps to Perform the Lab:**

**Step 1: Set Up the Arduino IDE**

1. Open the Arduino IDE.
2. Connect your Arduino board to your computer via the USB cable.
3. In the **Tools** menu, select the correct **Board** (e.g., Arduino Uno) and **Port**.

**Step 2: Write the Arduino Code**

1. **Code to Control the External LED:**

// Define the pin for the LED

const int ledPin = 9; // Connect the LED to pin 9

void setup() {

// Initialize Serial communication at 9600 baud

Serial.begin(9600);

// Set the LED pin as an output

pinMode(ledPin, OUTPUT);

// Print instructions to the Serial Monitor

Serial.println("Type '1' to turn ON the LED, '0' to turn OFF the LED");

}

void loop() {

// Check if there is data available in the serial buffer

if (Serial.available() > 0) {

// Read the incoming byte

char command = Serial.read();

// Turn ON the LED if the command is '1'

if (command == '1') {

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

Serial.println("LED is ON");

}

// Turn OFF the LED if the command is '0'

else if (command == '0') {

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

Serial.println("LED is OFF");

}

}

}

1. **Explanation:**
   * The LED is connected to **pin 9**. We define this in the code with const int ledPin = 9;.
   * In setup(), we start serial communication and set the LED pin as an output.
   * In loop(), we check if there is data available in the serial buffer. If data is available, we read it and determine whether to turn the LED on or off based on the input ('1' for ON and '0' for OFF).

**Step 3: Upload the Code**

1. Upload the code to the Arduino by clicking the **Upload** button (right arrow icon).
2. Open the **Serial Monitor** in the Arduino IDE (Tools > Serial Monitor or press **Ctrl+Shift+M**).

**Step 4: Test the LED Control**

1. In the Serial Monitor, type '1' and press **Enter**. The LED should turn ON, and the message "LED is ON" will appear in the Serial Monitor.
2. Type '0' and press **Enter**. The LED should turn OFF, and the message "LED is OFF" will appear in the Serial Monitor.

**Step 5: Experimentation**

* **Modify the Code:**
  + Change the LED pin to a different pin (e.g., pin 10) and update the code accordingly.
  + Add a feature to toggle the LED with a single key press instead of using two commands.
* **Additional Features:**
  + Implement a feature to blink the LED at a certain interval when a specific command (e.g., 'b') is entered.

// Add this in the loop function

if (command == 'b') {

for (int i = 0; i < 5; i++) {

digitalWrite(ledPin, HIGH);

delay(500); // LED ON for 500 milliseconds

digitalWrite(ledPin, LOW);

delay(500); // LED OFF for 500 milliseconds

}

Serial.println("Blinking done.");

}

**Summary of Key Concepts:**

* **Digital Output:** Controlling an external LED through digital signals.
* **Serial Communication:** Sending and receiving commands via the Serial Monitor.