**Objectives**  
  
By the end of this lab, students will be able to:

* Understand the basic architecture of a microcontroller.
* Set up a development environment (IDE, compiler, programmer).
* Configure GPIO pins as input and output.
* Write, upload, and debug basic microcontroller programs.
* Implement a simple input–output control system (button → LED).

**Required Equipment**

* Microcontroller board (Arduino Uno / STM32 Nucleo / ESP32 / PIC board)
* USB cable & programmer/debugger (if needed)
* Breadboard & jumper wires
* LED (1–2 pcs), push button (1 pc)
* Resistors: 220 Ω (for LED), 10 kΩ (for button pull-down)
* Computer with IDE installed (Arduino IDE / STM32CubeIDE / MPLAB X depending on MCU)

**Pre-lab Preparation**  
Read about the differences between microcontrollers and microprocessors.  
Review microcontroller architecture: CPU, Flash memory, RAM, I/O ports.'  
Install the required IDE and drivers for your platform.  
  
  
**Lab Activities**  
  
**Part A: Toolchain Setup (30 min)**  
Connect the microcontroller board to your computer.  
Open the IDE, select the correct board and COM port.  
Upload a sample “Hello World” program (LED blink).  
  
**Example (Arduino C code):**  
  
void setup() {  
  pinMode(13, OUTPUT);   // Set pin 13 as output  
}  
void loop() {  
  digitalWrite(13, HIGH); // LED ON  
delay(1000);            // Wait 1 sec  
digitalWrite(13, LOW);  // LED OFF  
delay(1000);            // Wait 1 sec  
}  
  
**Part B: Digital Input (45 min)**  
Wire a push button to pin 2 with a pull-down resistor.  
Modify the code to read the button state.  
Print button status to the Serial Monitor.  
  
**Example snippet:**  
int buttonPin = 2;  
void setup() {  
  pinMode(buttonPin, INPUT);  
  Serial.begin(9600);  
}  
void loop() {  
  int state = digitalRead(buttonPin);  
  Serial.println(state);  
  delay(200);  
}  
  
**Part C: Input → Output Control (1 hr)**  
Connect LED to pin 8.  
Write a program where the LED turns ON only when the button is pressed.  
Extend: Press → LED toggles state (like a switch).  
  
**Part D: Debugging Practice (30 min)**  
Use **Serial.print()** (or debugger breakpoints) to observe variable values.  
Introduce deliberate error (wrong pin, missing resistor) → troubleshoot.  
  
**Assessment Questions**  
What is the main difference between microcontrollers and microprocessors?  
Why do we need a pull-down resistor on the button input?  
If you press the button quickly, why might the LED not always respond? (hint: bouncing)  
Modify your program so that the LED stays ON for 5 seconds after a button press.