Push Button Counter

Embedded Systems Internship Project

Vishnu Sai Raju

June 10 – July 25

Introduction

- A push button counter is a basic embedded system project.
- Used to count the number of button presses.
- Applications include tally counters, digital access counters, etc.

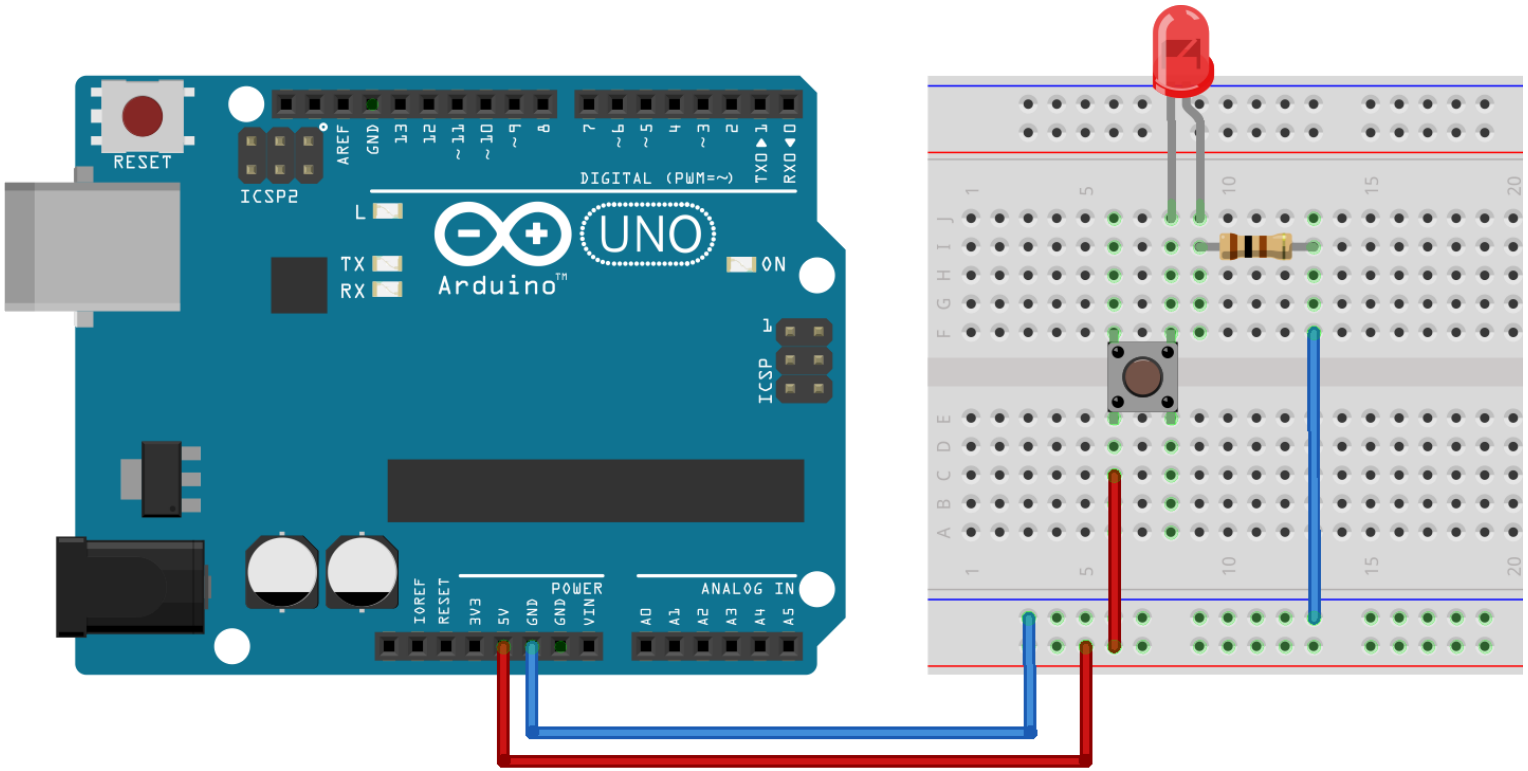
Objective

- Build a digital counter using push button input.
- Display the count using a 7-segment display or serial monitor.
- Understand microcontroller programming and interfacing.

Components Used

- Microcontroller (Arduino Uno)
- Push Button(s)
- Resistors
- Breadboard & Jumper wires
- 7-Segment Display / LCD / Serial Monitor
- Power supply or USB cable

Circuit Diagram



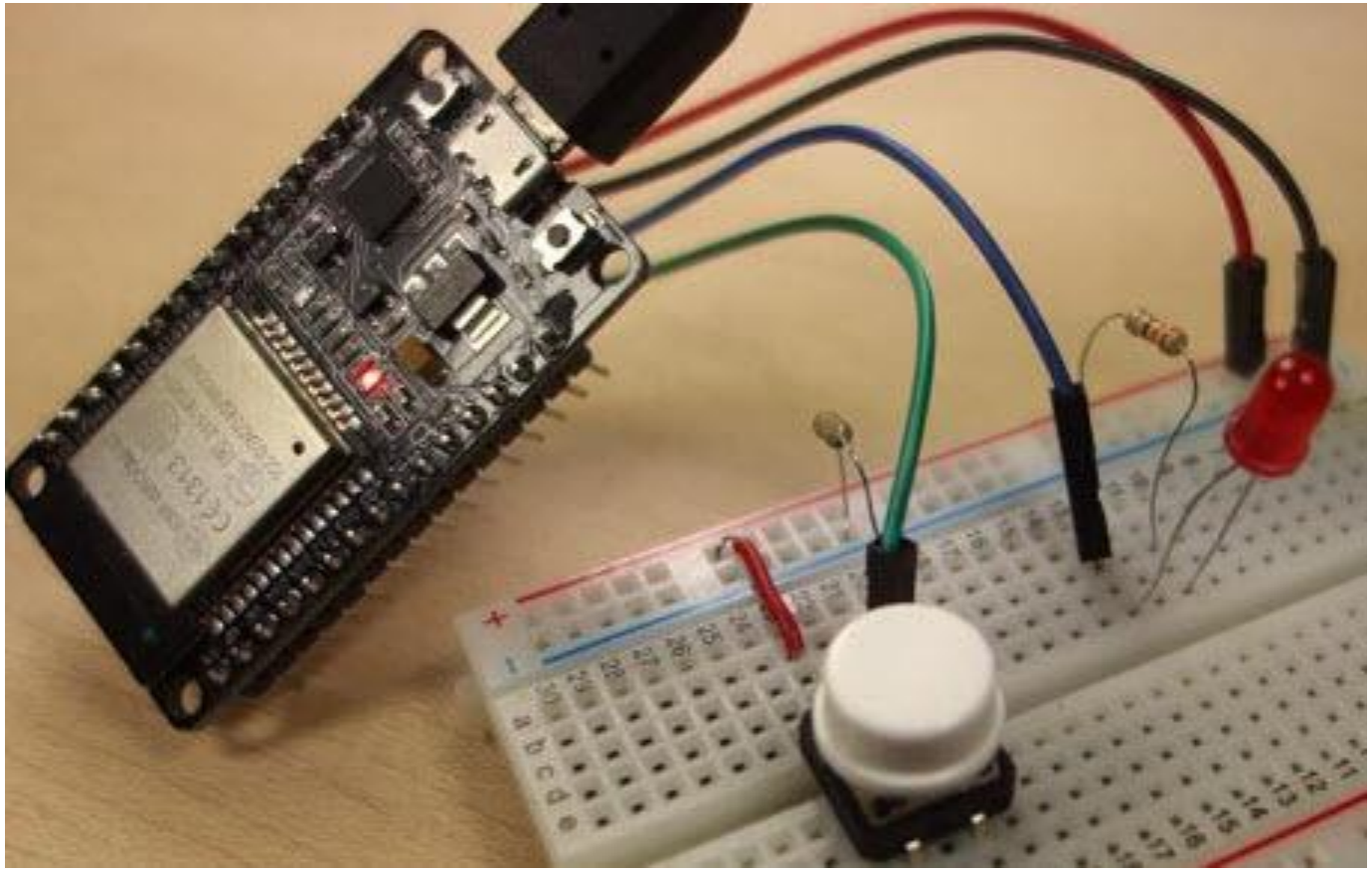
Working Principle

- Button press triggers an input signal.
- Counter variable increments on each press.
- Display updates with new count.
- Debouncing handled in software.

Code Overview

- Initialize input/output pins.
- Read button state.
- Increment counter on valid press.
- Display updated count.

Use correct Arduino code



```
int segPins[7] = {2, 3, 4, 5, 6, 7, 8};  // A, B, C, D, E, F, G
int buttonPin = 9;                  // Push button pin
int counter = 0;                    // Count variable (0 to 9)
bool lastButtonState = LOW;         // Previous state of button

// 7-segment digit patterns for 0-9
// {A, B, C, D, E, F, G}
byte digits[10][7] = {
 {1, 1, 1, 1, 1, 1, 0},  // 0
 {0, 1, 1, 0, 0, 0, 0},  // 1
 {1, 1, 0, 1, 1, 0, 1},  // 2
 {1, 1, 1, 1, 0, 0, 1},  // 3
 {0, 1, 1, 0, 0, 1, 1},  // 4
 {1, 0, 1, 1, 0, 1, 1},  // 5
 {1, 0, 1, 1, 1, 1, 1},  // 6
 {1, 1, 1, 0, 0, 0, 0},  // 7
 {1, 1, 1, 1, 1, 1, 1},  // 8
 {1, 1, 1, 1, 0, 1, 1}   // 9
};
void setup() {
 // Set segment pins as outputs
 for (int i = 0; i < 7; i++) {
  pinMode(segPins[i], OUTPUT);
 }

 // Set button pin as input
 pinMode(buttonPin, INPUT);
}

void loop() {
 bool currentButtonState = digitalRead(buttonPin);

 // Detect button press (rising edge)
 if (currentButtonState == HIGH && lastButtonState == LOW) {
  counter = (counter + 1) % 10; // Cycle from 0 to 9
  showDigit(counter);
  delay(200); // Debounce delay
 }

 lastButtonState = currentButtonState;
}

// Function to display digit on 7-segment
void showDigit(int num) {
 for (int i = 0; i < 7; i++) {
  digitalWrite(segPins[i], digits[num][i]);
 }
}
```

Challenges Faced

- Button bouncing caused false triggering.
- Voltage fluctuations.
- Limited I/O on microcontroller.

Solutions Implemented

- Used software debounce logic.
- Added pull-down resistor.
- Used delay for stable reading.

Learnings

- Hands-on experience with microcontrollers.
- Understanding of hardware interfacing.
- Improved debugging skills in embedded systems.

Future Scope

- Add increment/decrement buttons.
- Store count in EEPROM.
- Display on OLED or LCD.
- Send data wirelessly via Bluetooth/Wi-Fi.

Conclusion

- Successfully built and tested the push button counter.
- Gained deeper insights into embedded systems design.
- Prepared for more complex projects.