### **Implementation of Pushbutton Control for LED and Relay on ESP32 Using Wokwi PlatformIO in Visual Studio Code**

*Candra Wahyu Perdana*

Vocational Faculty, Brawijaya University

Email: candraega28@gmail.com

**Abstract**

This experiment aims to analyze the implementation of pushbutton control for managing an LED and a relay using the ESP32 microcontroller. The system uses a pushbutton connected to GPIO19 to turn an LED on GPIO18 and a relay on GPIO23 on or off. Development was carried out using Wokwi PlatformIO in Visual Studio Code, where the integrated simulation and debugging environment facilitates system verification. Experimental results indicate that with proper pin configuration and straightforward programming logic, external devices can be reliably controlled.

*Keywords—ESP32, Pushbutton, LED, Relay, Wokwi PlatformIO, Visual Studio Code*

**1. Introduction**

**1.1 Background**

The development of Internet of Things (IoT) technologies encourages the implementation of control solutions for electronic devices using microcontrollers. The ESP32 is a popular microcontroller due to its wireless connectivity capabilities and high performance. By integrating sensors and actuators—such as a pushbutton, LED, and relay—simple yet effective control systems can be developed for various automation applications.

**1.2 Objectives**

* Understand how the ESP32 controls external devices through pushbutton input.
* Implement a control logic that enables the LED and relay to turn on or off simultaneously based on the button's state.
* Utilize Wokwi PlatformIO in Visual Studio Code as an integrated development environment to simplify simulation and debugging.

**2. Methodology**

**2.1 Tools & Materials**

**Hardware:**

* ESP32 (simulated using Wokwi PlatformIO)
* Pushbutton
* LED
* Relay module
* Jumper wires

**Software:**

* **Visual Studio Code with the PlatformIO extension**
* **Wokwi PlatformIO as the simulation environment**
* **Arduino IDE (optional, for reference)**

**2.2 Implementation Steps**

**Project Creation:**

* Open Visual Studio Code and ensure that the PlatformIO extension is installed.
* Create a new project with the ESP32 board and integrate the simulation using Wokwi PlatformIO.

**Physical Connection:**

* Add the ESP32, pushbutton, LED, and relay module components into the simulation schematic.
* Connect the pushbutton to GPIO19 using an internal pull-up configuration, the LED to GPIO18, and the relay to GPIO23.

**Code Development:**

* Use the following code to set up the pin modes, read the pushbutton status, and control the LED and relay based on the button's state.

**Simulation & Verification:**

* Run the simulation using Wokwi PlatformIO in Visual Studio Code.
* Verify that when the pushbutton is pressed (resulting in a LOW signal), both the LED and relay activate; when the button is released, both devices turn off.



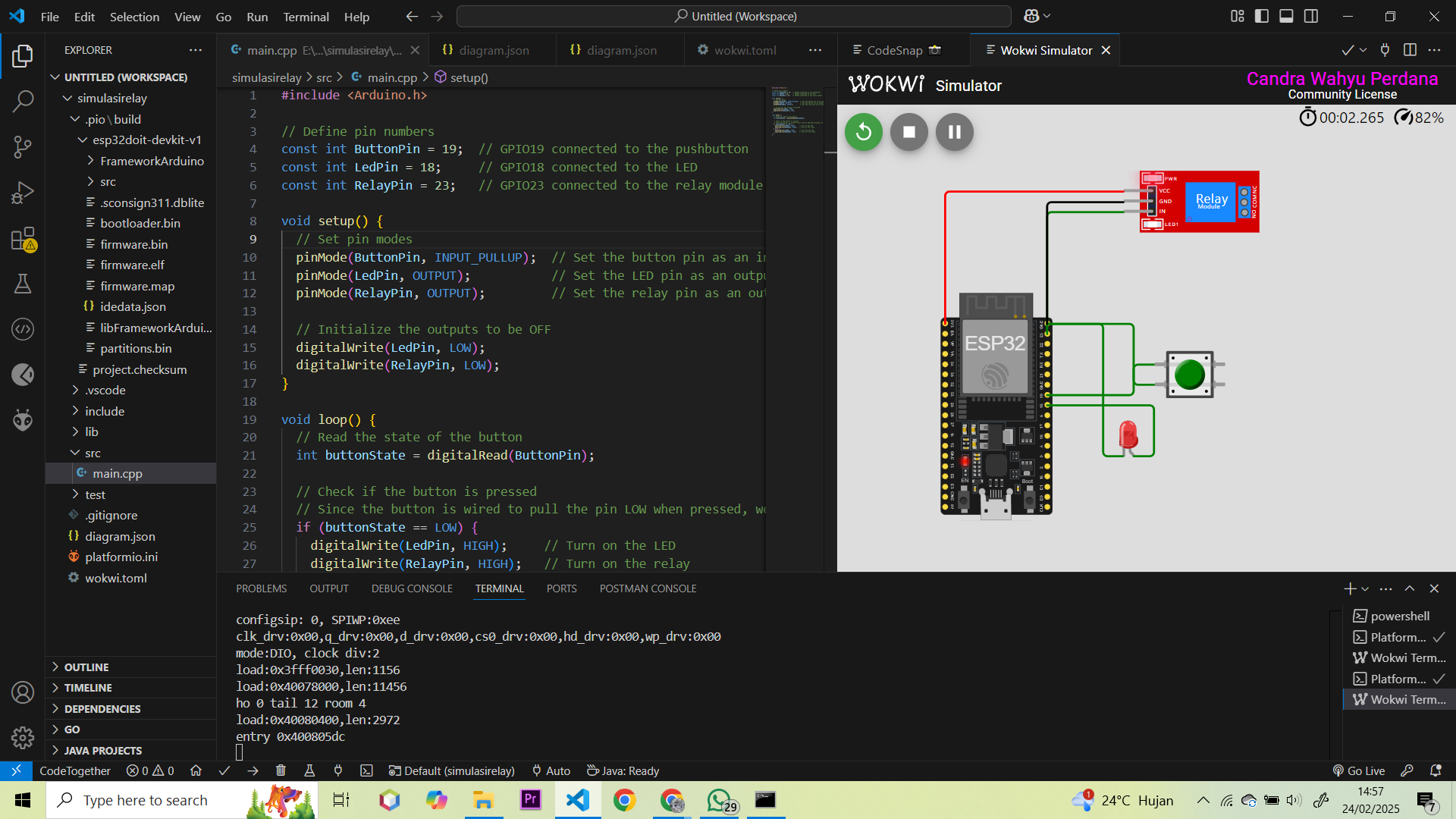
**3. Results and Discussion**

**3.1 Experimental Results**

* **Data Output:**When the pushbutton is pressed, the LED lights up and the relay activates. When the button is released, both the LED and relay turn off.
* **Error Handling:**The use of the internal pull-up configuration for the pushbutton allows reliable detection of its state, ensuring that the system responds accurately to user input.

**3.2 Discussion**

* **Successful Implementation:**The simulation demonstrates that the ESP32 successfully reads the pushbutton status and simultaneously controls both the LED and relay.
* **Potential Issues & Solutions:**
  + **Pin Configuration:** Using INPUT\_PULLUP for the pushbutton ensures that the button remains at a HIGH logic level in its default state and goes LOW when pressed, simplifying the input logic.
  + **Control Logic:** The simple conditional structure in the loop is effective for controlling outputs based on the input status.
  + **System Enhancements:** Although the system functions well, implementing software or hardware debouncing can be considered to avoid multiple readings due to mechanical noise from the button.



**4. Appendix**

#include <Arduino.h>

// Define pin numbers

const int ButtonPin = 19; // GPIO19 connected to the pushbutton

const int LedPin = 18; // GPIO18 connected to the LED

const int RelayPin = 23; // GPIO23 connected to the relay module

void setup() {

// Set pin modes

pinMode(ButtonPin, INPUT\_PULLUP); // Set the button pin as an input with an internal pull-up resistor

pinMode(LedPin, OUTPUT); // Set the LED pin as an output

pinMode(RelayPin, OUTPUT); // Set the relay pin as an output

// Initialize the outputs to be OFF

digitalWrite(LedPin, LOW);

digitalWrite(RelayPin, LOW);

}

void loop() {

// Read the state of the button

int buttonState = digitalRead(ButtonPin);

// Check if the button is pressed

// Since the button is wired to pull the pin LOW when pressed, we check for LOW

if (buttonState == LOW) {

digitalWrite(LedPin, HIGH); // Turn on the LED

digitalWrite(RelayPin, HIGH); // Turn on the relay

} else {

digitalWrite(LedPin, LOW); // Turn off the LED

digitalWrite(RelayPin, LOW); // Turn off the relay

}

}

This report comprehensively details the process of implementing and testing a pushbutton control system for an LED and relay on an ESP32 using Wokwi PlatformIO in Visual Studio Code. This approach leverages the ease of simulation and debugging provided by the integrated development environment, thus accelerating the prototyping process for IoT-based automation applications.