2025 TEEP Progress Report Week-3

Used Wokwi to complete the tasks for Labs 1 through 6

(Before creating the physical project, please use **Wokwi** for online simulation to complete the initial design of the circuit and code. Then, use **Thonny** together with the **ESP32** development board to build the actual project. Please **record a video of the project** in operation, upload it to YouTube, and provide the video link(LAB9) so we can better understand your learning progress.)

Required Materials:

- ESP32 Development Board
- Pushbutton Switches (3-pin)
- Jumper Wires
- LED
- USB Cable
- Breadboard

Wiring :

Button SW1:

- **Middle pin → GPIO15**
- **Side pin → GND**

Button SW2:

- **Middle pin → GPIO4**
- **Side pin → GND**

LED:

- **Long leg → GPIO2 through 220 Ω resistor**
- **Short leg → GND**

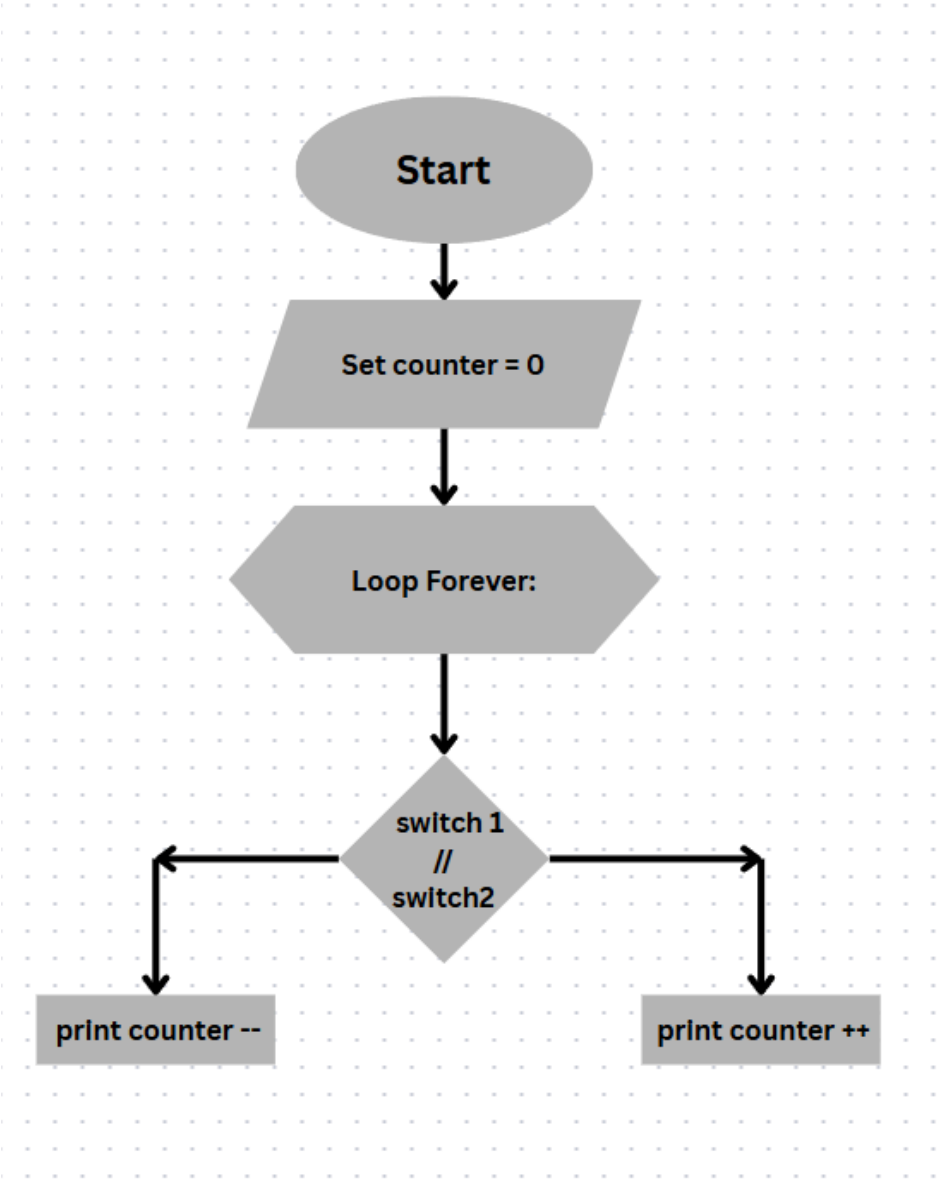
Lab1 – On Wokwi and Thonny, complete the following tasks:

Read button value and perform specified action, and a single button only executes once.

1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

- Function 1: When SW1 is pressed, **increase the counter by 1** and **display the value in the Shell**.
- Function 2: When SW2 is pressed, **decrease the counter by 1** and **display the value in the Shell**

Flow chart :

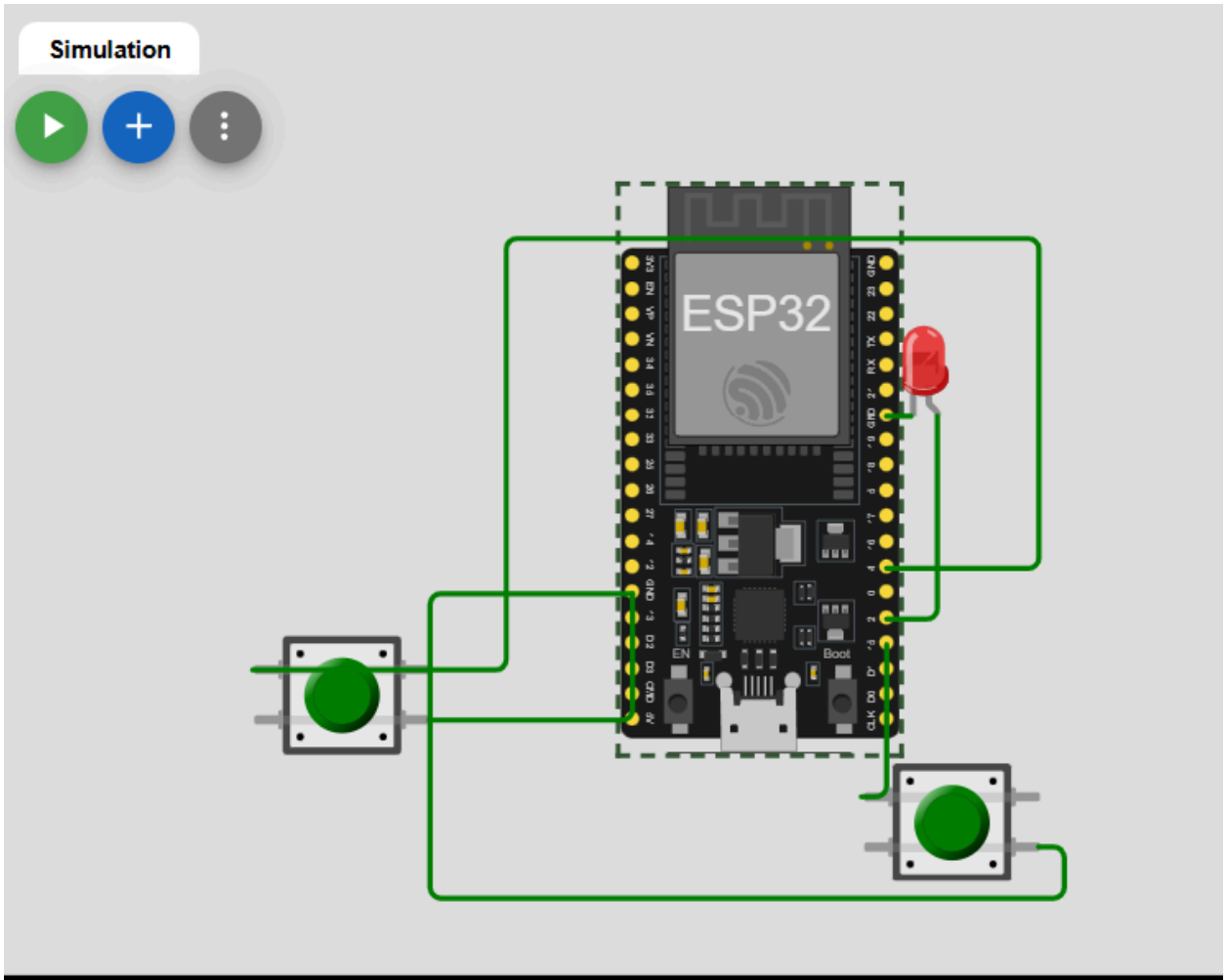


Source code :

```
from machine import Pin
import time
# Pin configuration
SW1_PIN = 15
SW2_PIN = 4
LED_PIN = 2
sw1 = Pin(SW1_PIN, Pin.IN, Pin.PULL_UP)
sw2 = Pin(SW2_PIN, Pin.IN, Pin.PULL_UP)
led = Pin(LED_PIN, Pin.OUT)
counter = 0
sw1_pressed = False
sw2_pressed = False
print("Program started. SW1=increment, SW2=decrement")
while True:
    if sw1.value() == 0 and not sw1_pressed:
        counter += 1
        print("Counter:", counter)
        led.value(1)
        time.sleep(0.2)
        led.value(0)
        sw1_pressed = True
    elif sw1.value() == 1:
        sw1_pressed = False
    if sw2.value() == 0 and not sw2_pressed:
```

```
            counter -= 1
            print("Counter:", counter)
            led.value(1)
            time.sleep(0.2)
            led.value(0)
            sw2_pressed = True
        elif sw2.value() == 1:
            sw2_pressed = False
        time.sleep(0.05)
```

Test Record:




```
ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
Blinking LED pattern...
End Program!
MicroPython v1.22.0 on 2023-12-27; Generic ESP32 module with ESP32
Type "help()" for more information.
>>>
```

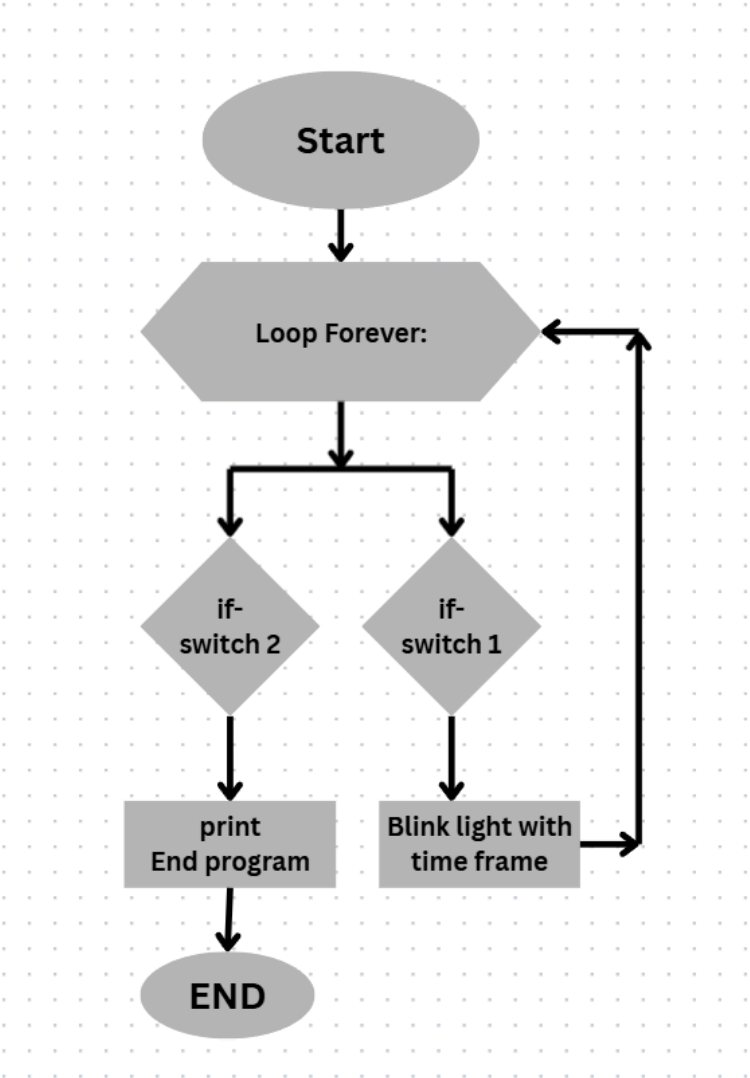
Lab2- On Wokwi and Thonny, complete the following tasks:

Read button value and perform specified action, and a single button only executes once.

1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

   - Function 1: When SW1 is pressed, stop the program and display "End Program!" on the Shell.
   - Function 2: When SW2 is pressed, make the LED blink in a pattern: ON for 3 seconds and OFF for 1.5 seconds, repeated 7 times.

Flow chart :



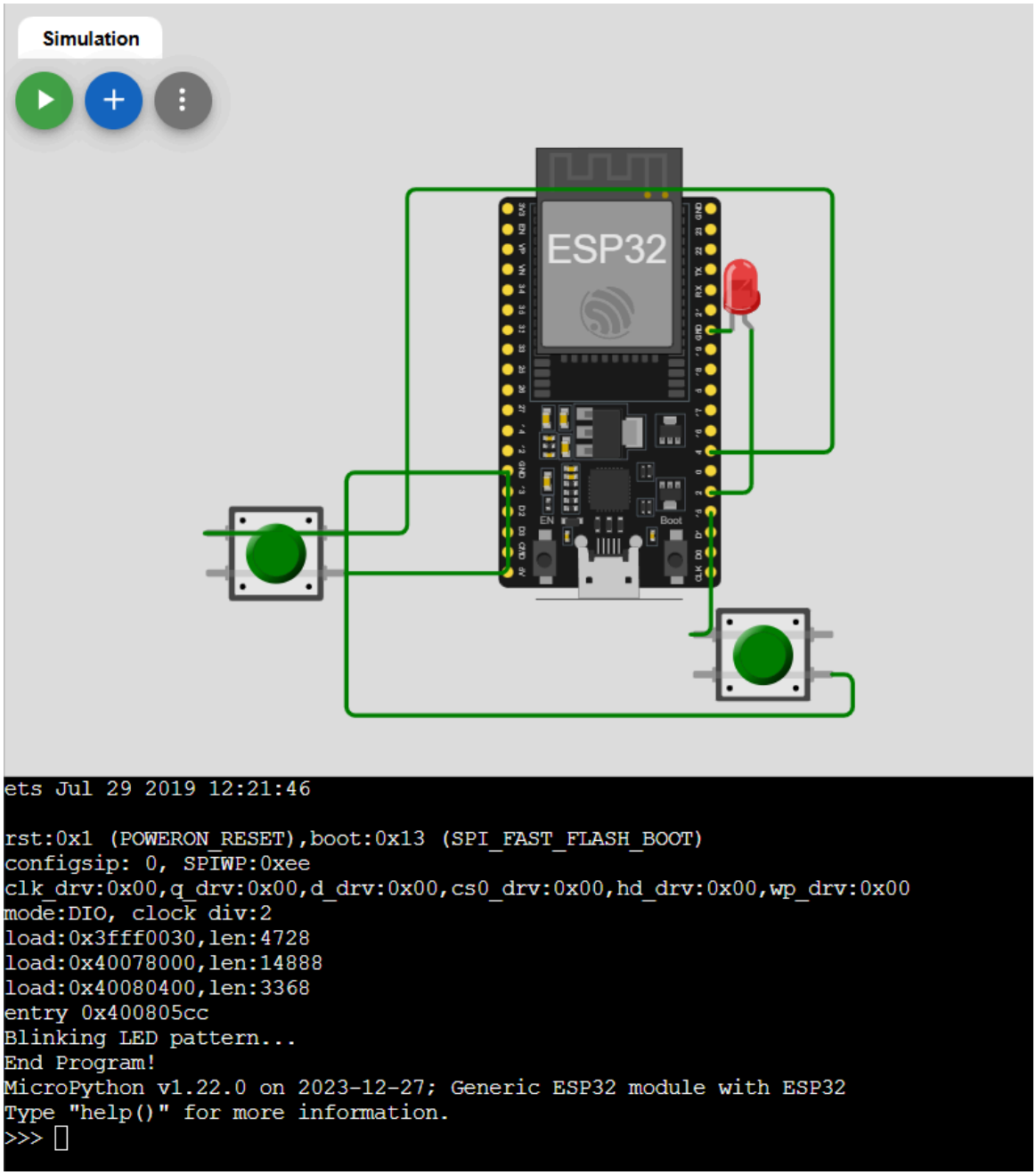
Source Code :

```
from machine import Pin
import time


# Pin setup
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)  # Button 1
SW2 = Pin(4, Pin.IN, Pin.PULL_UP)   # Button 2
LED = Pin(2, Pin.OUT)               # LED
prev_sw1 = 1
prev_sw2 = 1
running = True
def check_sw1():
    """Check if SW1 is pressed and stop program."""
    global running
    if SW1.value() == 0:  # pressed (LOW)
        print("End Program!")
        LED.off()
        running = False
def safe_sleep(seconds):
    """Sleep with SW1 check."""
    step = 0.05
    loops = int(seconds / step)
    for _ in range(loops):
        check_sw1()
        if not running:
```

```
            return
        time.sleep(step)

# Small delay so startup bounce doesn't trigger
time.sleep(0.5)
while running:
    sw1_state = SW1.value()
    sw2_state = SW2.value()
    # SW1 pressed → end program
    if sw1_state == 0 and prev_sw1 == 1:
        print("End Program!")
        LED.off()
        break
    # SW2 pressed → blink LED pattern
    if sw2_state == 0 and prev_sw2 == 1:
        print("Blinking LED pattern...")
        for _ in range(7):
            LED.on()
            safe_sleep(3)
            if not running:
                break
            LED.off()
            safe_sleep(1.5)
            if not running:
                break
```

```
prev_sw1 = sw1_state

prev_sw2 = sw2_state

time.sleep(0.05)  # debounce
```

Test Record :



Shell

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Blinking LED pattern...
  End Program!

>>>
```

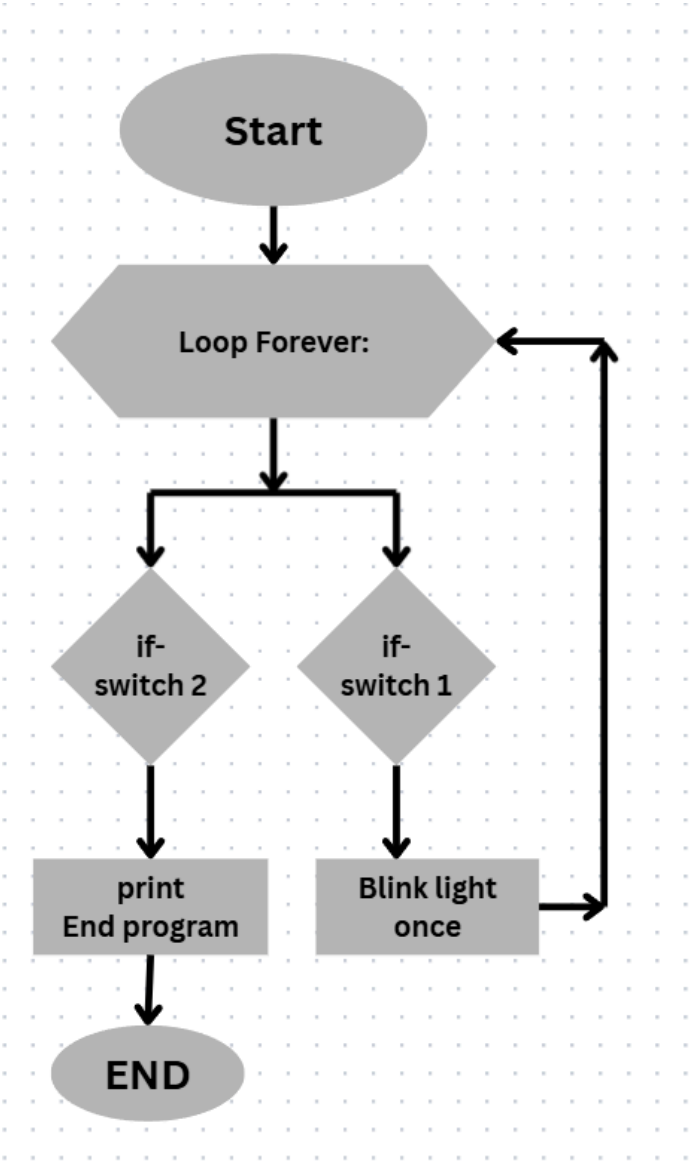
Lab3 – On Wokwi and Thonny, complete the following tasks:

Read the button value and perform the specified action.

1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

   - Function 1: When SW1 is pressed, turn on the LED; when released, turn it off.
   - Function 2: When SW2 is pressed, stop the program and display "End Program!" on the Shell.

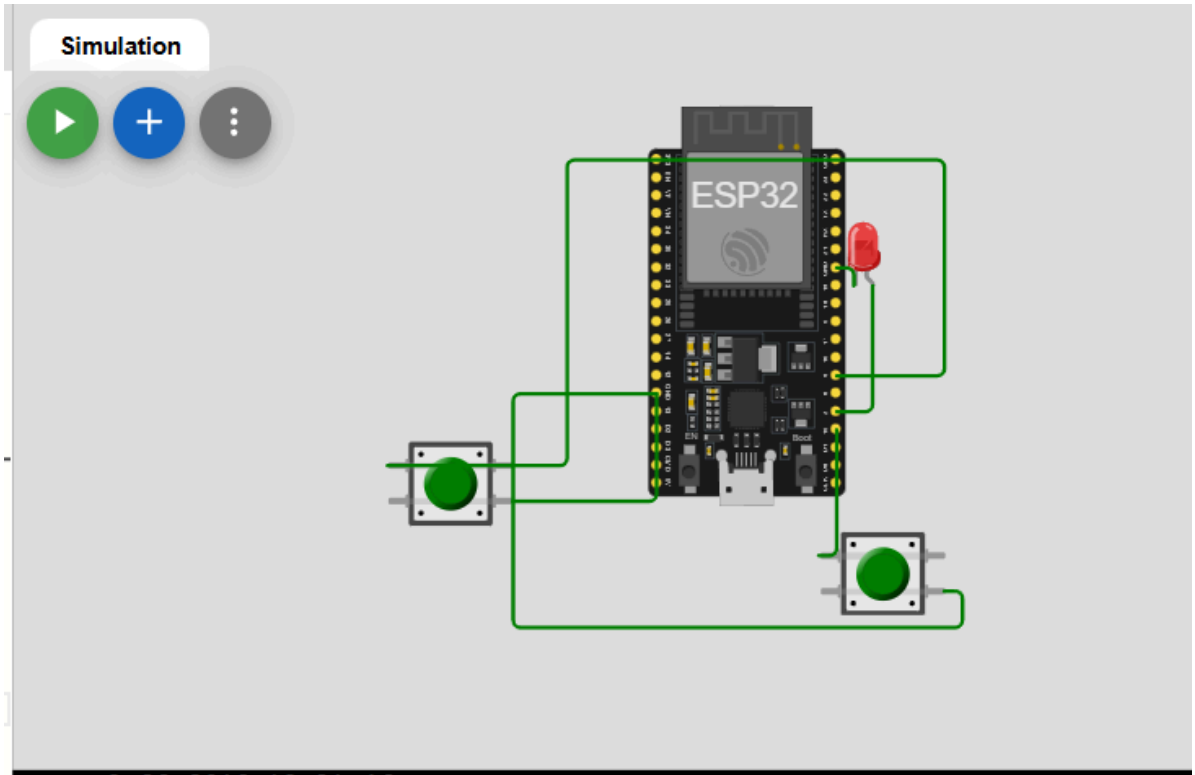
Flow Chart :



Source Code :

```
from machine import Pin
import time
# Pin setup
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)  # Button 1 for LED control
SW2 = Pin(4, Pin.IN, Pin.PULL_UP)   # Button 2 for stopping program
LED = Pin(2, Pin.OUT)               # LED output
running = True
# Small delay to avoid false trigger at reset
time.sleep(0.5)

while running:
    # Function 1: LED ON while SW1 is pressed
    if SW1.value() == 0:  # Pressed
        LED.on()
        print("Light Blink")
    else:
        LED.off()

    # Function 2: End program if SW2 is pressed
    if SW2.value() == 0:
        print("End Program!")
        LED.off()
        break
    time.sleep(0.05)  # debounce
```

Test Record :




```
ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
Light blink
Light blink
Light blink
Light blink
Light blink
Light blink
Light blink
Light blink
Light blink
Light blink
End Program!
MicroPython v1.22.0 on 2023-12-27; Generic ESP32 module with ESP32
Type "help()" for more information.
>>>
```

Shell

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  Light Blink
  End Program!

>>>
```

Lab4 – On Wokwi and Thonny, complete the following tasks:

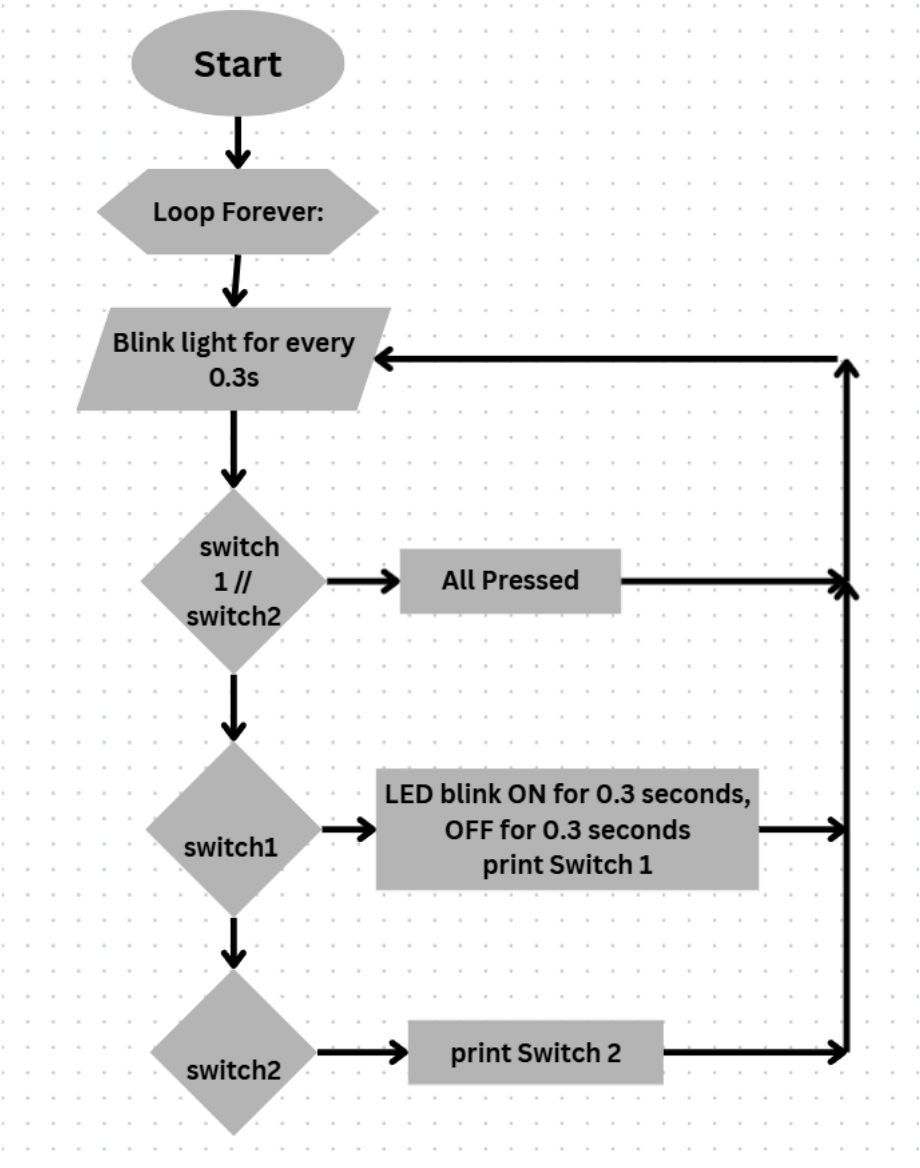
1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

- Function 1: At startup, the LED blinks slowly — ON for 1.5 seconds and OFF for 1.5 seconds repeatedly.
- Function 2: When SW1 and SW2 are pressed simultaneously, display "All Pressed!" on the Shell.
- Function 3: When only SW1 is pressed, make the LED blink ON for 0.3 seconds, OFF for 0.3 seconds, repeated 3 times, and display "SW1 pressed!" on the Shell.
- Function 4: When only SW2 is pressed, display "SW2 pressed!" on the Shell.

HINT**:**

Use an **if statement** within a **defined sub-function** to read the button value. Combined with a **while loop**, this can be used to lock the program

Flow Chart :



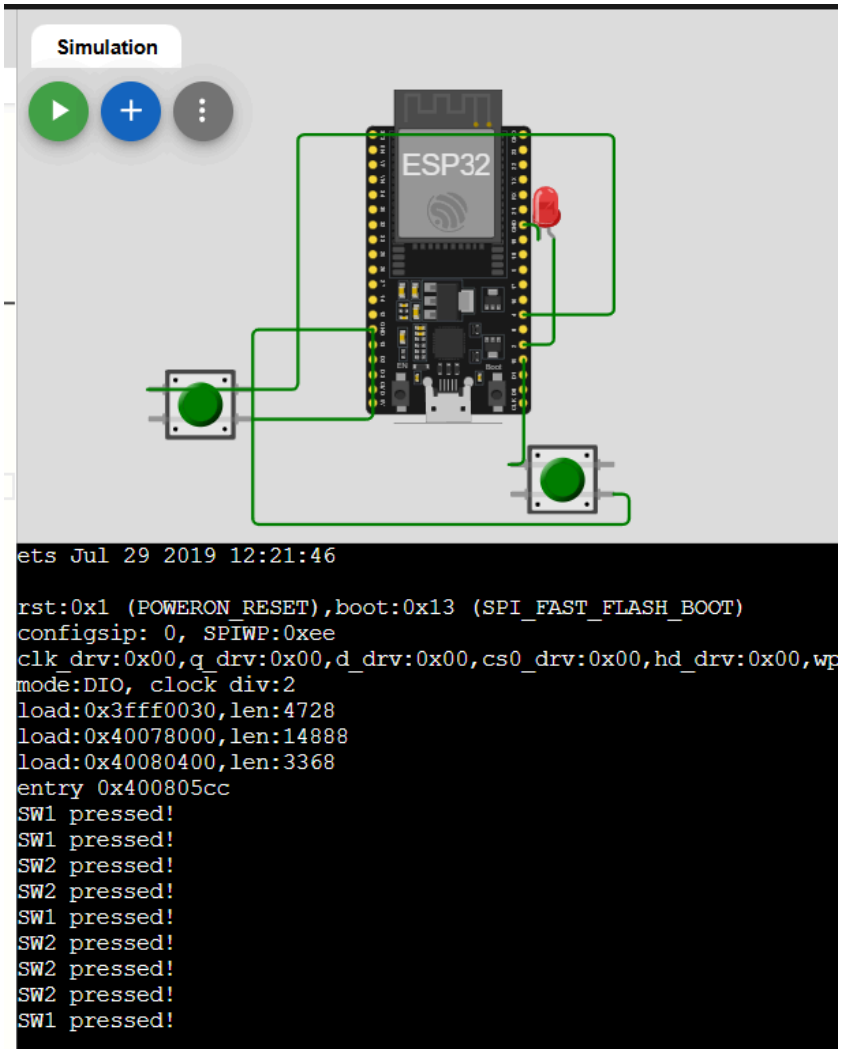
Source code :

```
from machine import Pin
import time
# Pin setup
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)
SW2 = Pin(4, Pin.IN, Pin.PULL_UP)
```

```
LED = Pin(2, Pin.OUT)
def slow_blink():
    LED.on()
    time.sleep(1.5)
    LED.off()
    time.sleep(1.5)
def sw1_blink():
    for _ in range(3):
        LED.on()
        time.sleep(0.3)
        LED.off()
        time.sleep(0.3)
# Main loop
while True:
    sw1_state = SW1.value() == 0
    sw2_state = SW2.value() == 0
    if sw1_state and sw2_state:
        print("All Pressed!")
        time.sleep(0.2)  # debounce
    elif sw1_state:
        print("SW1 pressed!")
        sw1_blink()
        time.sleep(0.2)  # debounce
    elif sw2_state:
        print("SW2 pressed!")
```

```
        time.sleep(0.2)  # debounce
    else:
        slow_blink()
```

Test Record



:

Shell

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  SW1 pressed!
  SW1 pressed!
  SW1 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW1 pressed!
  SW1 pressed!
  SW1 pressed!
```

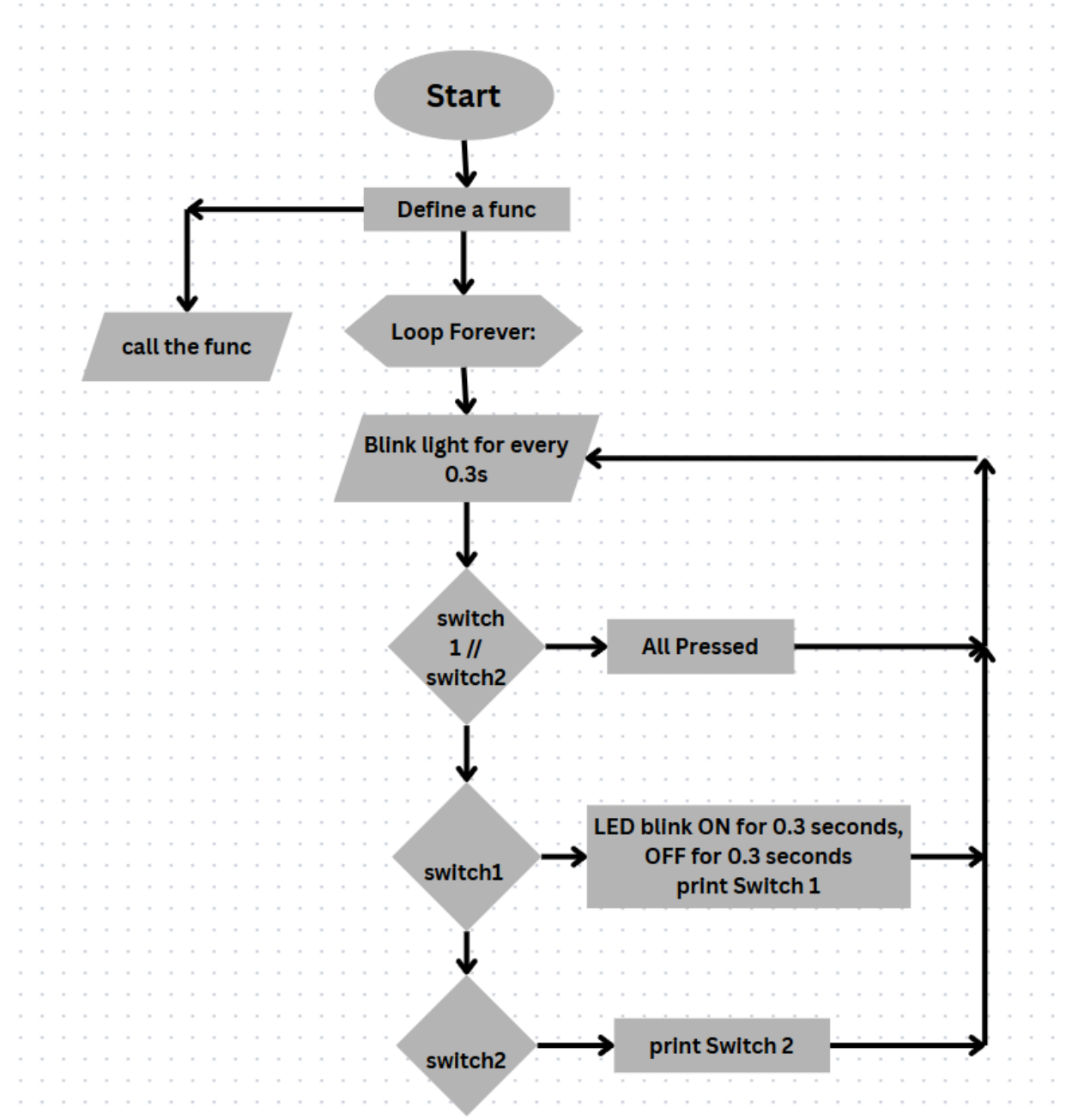
Lab5 – On Wokwi and Thonny, complete the following tasks:

1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

- Function 1: At startup, the LED blinks slowly — ON for 1.5 seconds and OFF for 1.5 seconds repeatedly.
- Function 2: When SW1 and SW2 are pressed simultaneously, display "All Pressed!" on the Shell.
- Function 3: When only SW1 is pressed, make the LED blink ON for 0.3 seconds, OFF for 0.3 seconds, repeated 3 times, and display "SW1 pressed!" on the Shell.
- Function 4: When only SW2 is pressed, display "SW2 pressed!" on the Shell.

HINT:
Use the def keyword to define a sub-function, and combine it with if statements to read the button value. With a while loop, you can lock the program execution.

Flow chart :



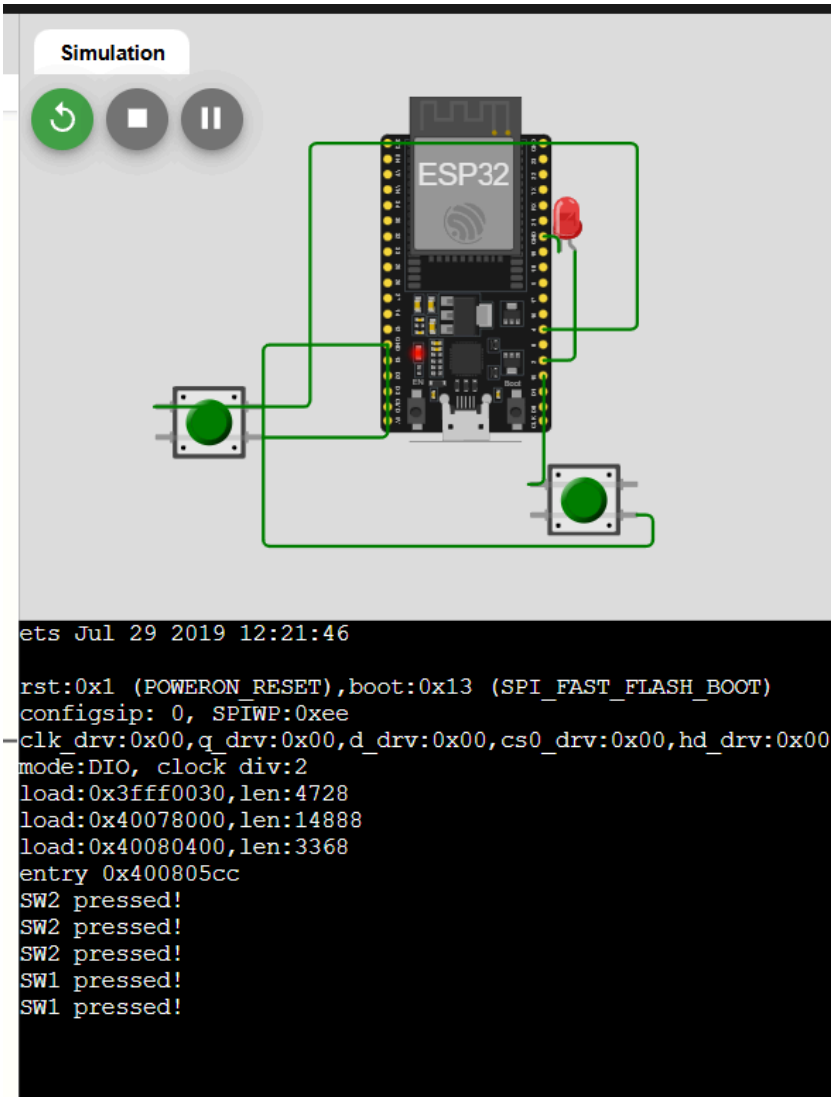
Source Code :

```
from machine import Pin
import time
# Pin setup
```

```
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)
SW2 = Pin(4, Pin.IN, Pin.PULL_UP)
LED = Pin(2, Pin.OUT)
def slow_blink():
    LED.on()
    time.sleep(1.5)
    LED.off()
    time.sleep(1.5)
def sw1_blink():
    for _ in range(3):
        LED.on()
        time.sleep(0.3)
        LED.off()
        time.sleep(0.3)
def check_buttons():
    while True:
        sw1_pressed = SW1.value() == 0
        sw2_pressed = SW2.value() == 0
        if sw1_pressed and sw2_pressed:
            print("All Pressed!")
            time.sleep(0.2)  # debounce
        elif sw1_pressed:
            print("SW1 pressed!")
            sw1_blink()
            time.sleep(0.2)  # debounce
```

```
        elif sw2_pressed:
            print("SW2 pressed!")
            time.sleep(0.2)  # debounce
        else:
            slow_blink()
# Start the program
check_buttons()
```

Test Records :



Shell

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  SW1 pressed!
  SW1 pressed!
  SW1 pressed!
  SW1 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW2 pressed!
  SW1 pressed!
  SW1 pressed!
  SW1 pressed!
```

Lab6 – On Wokwi and Thonny, complete the following tasks:

Determine whether SW1 is a short press (less than 1 second) or a long press (more than 1 second), and perform different actions based on the press duration.

1. Connect SW1 to GPIOXX and the LED to GPIOXX.

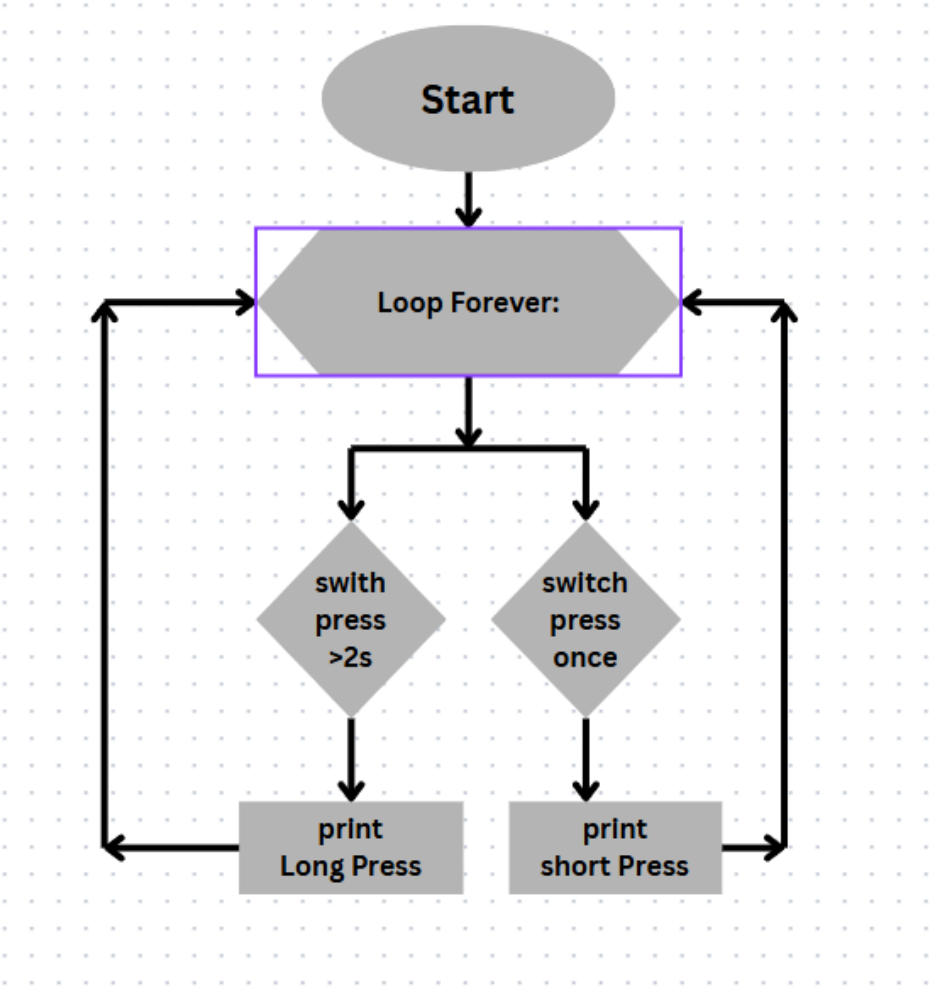
   - **Function 1**: **Short press** of **SW1**: Make the **LED** blink **once** (ON for **0.5 seconds**, OFF for **0.5 seconds**).
   - **Function 2**: **Long press** of **SW1**: Make the **LED** blink **five times**.
     At the same time, display **"Short Press"** or **"Long Press"** on the **Shell**.

Hint:
You can use the time module to record the press and release time.
Use a while loop to wait until the button is released.

Wiring :

- GPIO15 to Middle pin of SW1
- GND to One side pin of SW1
- GPIO2 to LED Anode (+)
- GND to LED Cathode (-)
- 3.3V or GND Leave unconnected

Flow Chart :

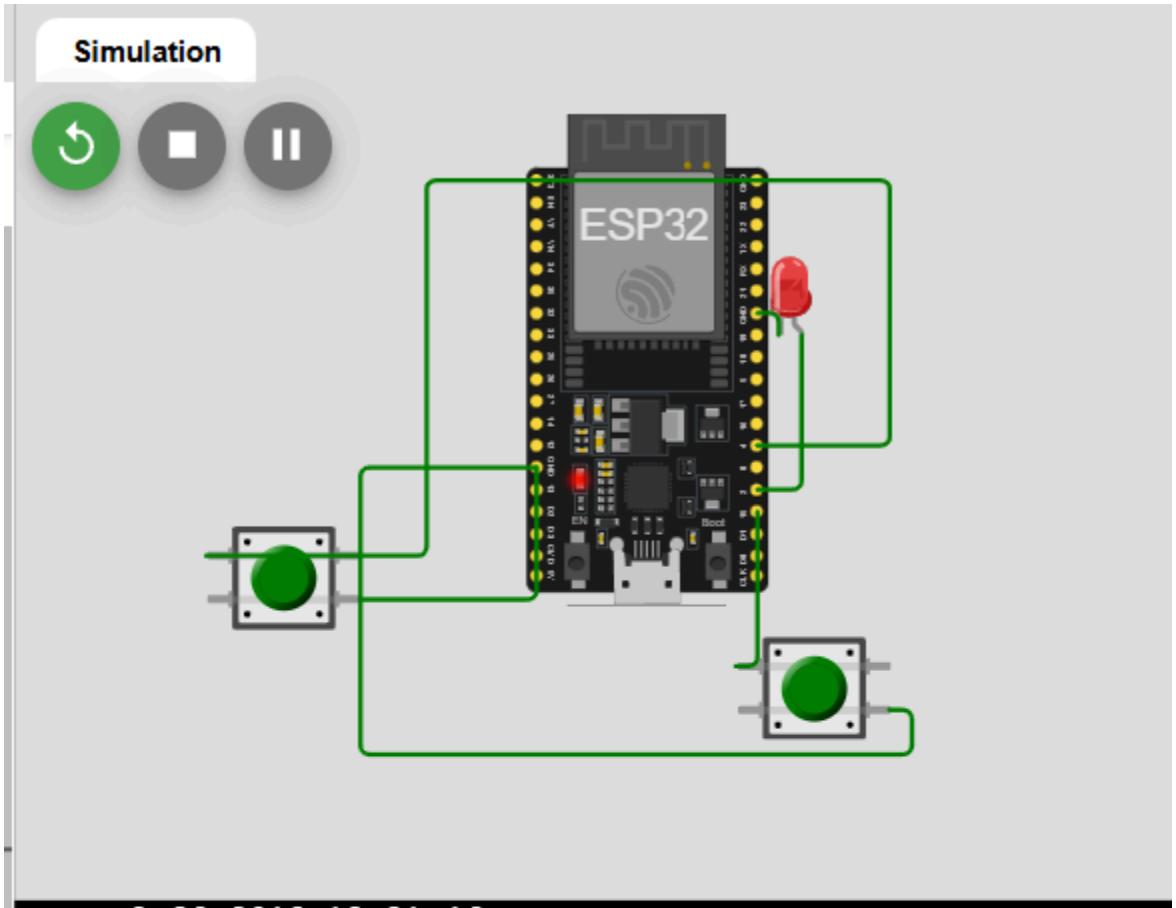


Source Code :

```
from machine import Pin
import time
# Pin setup
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)
LED = Pin(2, Pin.OUT)
def blink(times, on_time=0.5, off_time=0.5):
    for _ in range(times):
        LED.on()
```

```
        time.sleep(on_time)
        LED.off()
        time.sleep(off_time)
while True:
    # Wait for button press (active LOW)
    if SW1.value() == 0:
        press_time = time.ticks_ms()  # record press time
        # Wait until button released
        while SW1.value() == 0:
            time.sleep(0.01)
        release_time = time.ticks_ms()
        duration = time.ticks_diff(release_time, press_time) / 1000  # in seconds
        if duration < 1:
            print("Short Press")
            blink(1, 0.5, 0.5)
        else:
            print("Long Press")
            blink(5, 0.5, 0.5)
    time.sleep(0.05)  # debounce
```

Test Record :




```
ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
Short Press
Long Press
Long Press
Short Press
```

Shell

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Long Press
  Short Press
  Long Press
  Short Press
  Short Press
  Short Press
  Short Press
  Long Press
  Long Press
  Long Press
```

Lab7 – On Wokwi and Thonny, complete the following tasks:

Design a program using two buttons (SW1 and SW2) and one LED to achieve the following functions:

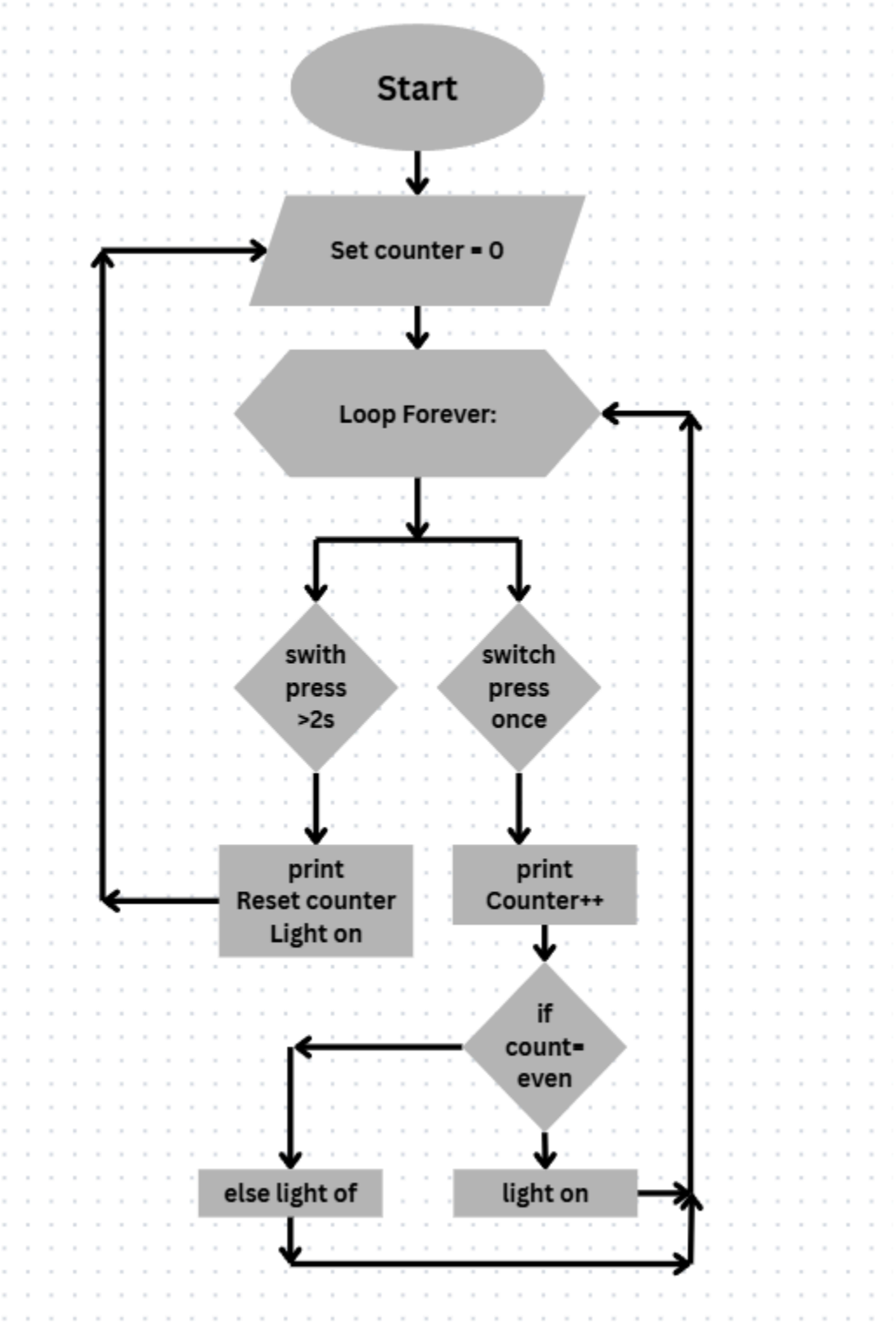
1. Connect SW1 to GPIOXX, SW2 to GPIOXX, and the LED to GPIOXX.

- Function 1: Each time SW1 is pressed, increment a counter by 1, and display the current count on the Shell (e.g., "Count: 3").
- Function 2: When SW2 is long-pressed for more than 2 seconds, reset the counter to 0, display "Counter Reset!" on the Shell, and make the LED blink rapidly 3 times as a visual cue.
- Function 3: When the counter is even, turn the LED ON; when the counter is odd, turn the LED OFF.

Hint:

- Use a variable like counter to store the current count.
- Use the time module to measure the press duration of SW2 to detect long press.

Flow Chart :

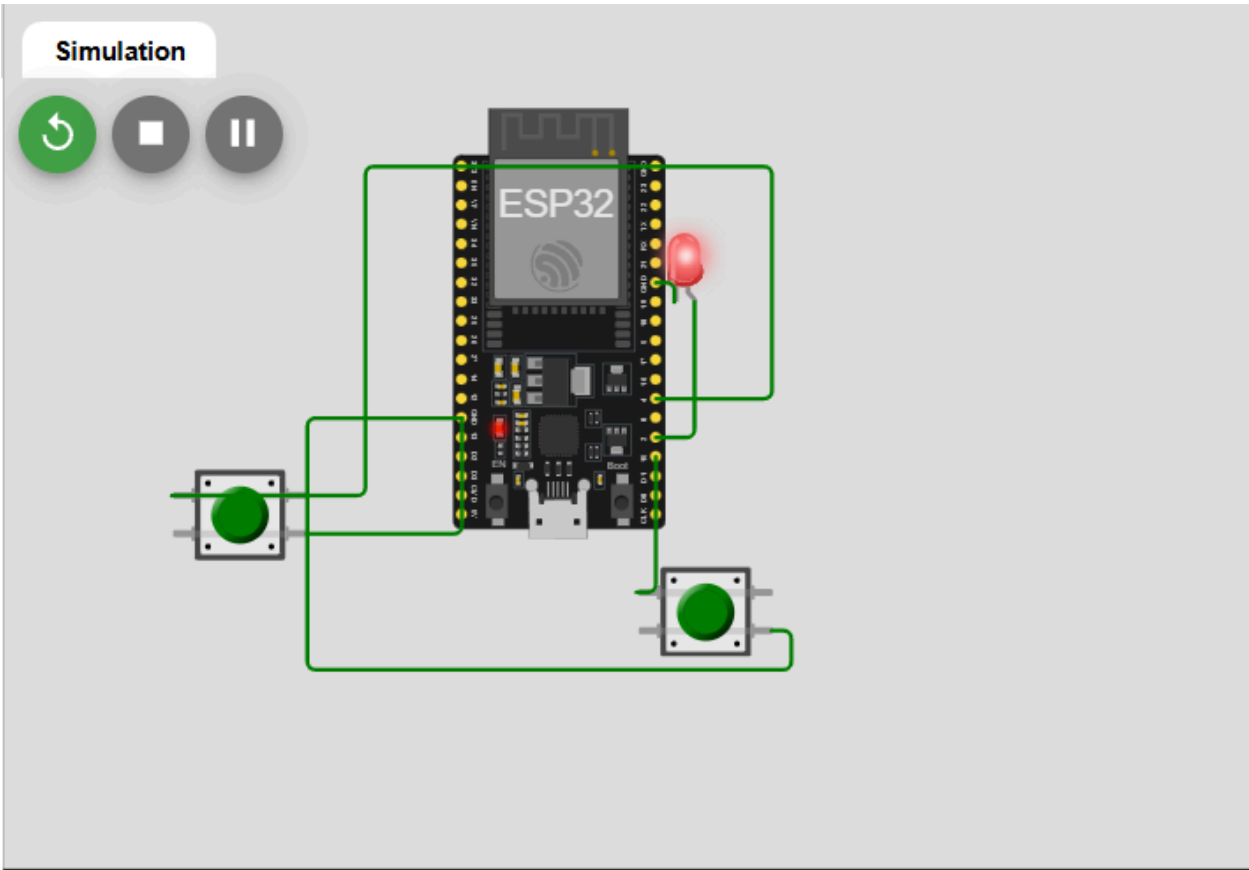

Start
Set counter = 0
Loop Forever:
swith press >2s
switch press once
print Reset counter Light on
print Counter++
if count= even
else light of
light on

Source code :

```
from machine import Pin
import time
# Pin setup
SW1 = Pin(15, Pin.IN, Pin.PULL_UP)
SW2 = Pin(4, Pin.IN, Pin.PULL_UP)
LED = Pin(2, Pin.OUT)
counter = 0
prev_sw1_state = 1
prev_sw2_state = 1
def blink(times, on_time=0.2, off_time=0.2):
    for _ in range(times):
        LED.on()
        time.sleep(on_time)
        LED.off()
        time.sleep(off_time)
while True:
    sw1_state = SW1.value()
    sw2_state = SW2.value()
    # Detect rising edge for SW1 press (button press event)
    if sw1_state == 0 and prev_sw1_state == 1:
        counter += 1
        print(f"Count: {counter}")
        # Turn LED ON if even, OFF if odd
```

```
            if counter % 2 == 0:
                LED.on()
            else:
                LED.off()
        # Detect long press on SW2 (>2 seconds)
        if sw2_state == 0 and prev_sw2_state == 1:
            press_start = time.ticks_ms()
            # Wait for release or timeout
            while SW2.value() == 0:
                time.sleep(0.01)
                    press_duration = time.ticks_diff(time.ticks_ms(), press_start) /
1000
                if press_duration > 2:
                    # Reset counter and blink LED rapidly 3 times
                    counter = 0
                    print("Counter Reset!")
                    blink(3, 0.1, 0.1)
                    LED.on()  # LED ON for even (0 is even)
                    break
        prev_sw1_state = sw1_state
        prev_sw2_state = sw2_state
        time.sleep(0.05)  # debounce delay
```

Test Record :




```
ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14888
load:0x40080400,len:3368
entry 0x400805cc
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5
Count: 6
Count: 7
Counter Reset!
```

Shell ×

```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  Count: 1
  Count: 2
  Count: 3
  Count: 4
  Count: 5
  Count: 6
  Count: 7
  Count: 8
  Count: 9
  Count: 10
  Count: 11
  Counter Reset!
  Count: 1
  Count: 2
  Count: 3
  Count: 4
  Count: 5
  Count: 6
  Counter Reset!
```

Lab8 learning reflection or comments(100 words)

The content of Learning reflection or comments can include your thoughts on the course, such as whether it was too difficult or too easy. You may also share something memorable that happened during the week. Overall, it is similar to a weekly journal, focusing on your learning experiences and personal feelings.

Learning reflection or Comments:

This week's lab tasks were straightforward and enjoyable. I learned how to integrate multiple buttons and an LED to create interactive functions using MicroPython on the ESP32. At first, handling button presses and debouncing was challenging, but through practice, I gained confidence in using conditional statements, loops, and timing functions.

I was able to implement button control, LED blinking patterns, and timing functions with confidence. The use of Wokwi and Thonny made testing smooth, and I appreciated how quickly I could see results after making changes.The tasks were neither too easy nor too hard; they required patience and logical thinking.The memorable moment was when my long-press detection finally worked after several trials—it felt rewarding to see the LED respond exactly as expected. The most memorable part was watching my code work exactly as expected on the first try.

Overall, I improved my problem-solving skills and enjoyed seeing hardware respond to my code and this week boosted my confidence in programming microcontrollers and strengthened my understanding of GPIO input and output control.

Lab9 Video Link

Record a video of approximately 3 minutes to explain your research process. The video should include the following:

- **Link: https://youtu.be/cWzoOr7IzyA**
- **Note : I use ai for Voice over to add some more details to my video**

Thank You