ESP32-WROOM-32 LED Blink Guide (38-pin) - Using PlatformIO

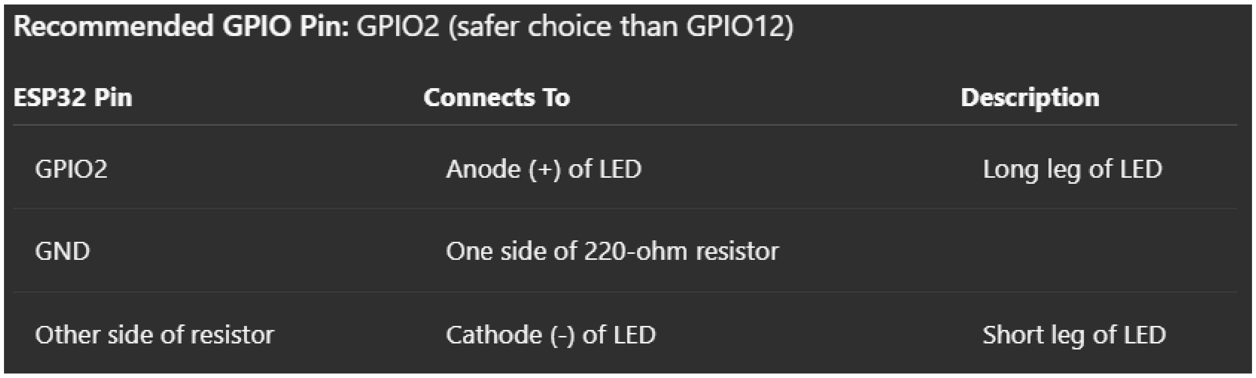
1. Objective

To blink an LED connected to an ESP32-WROOM-32 (38-pin version) using PlatformIO.

2. Hardware Required

- ESP32-WROOM-32 (38-pin) Development Board
- 1 x LED (any color)
- 1 x 220-ohm resistor
- Breadboard
- Jumper wires

3. Circuit Diagram and Pin Connections



Recommended GPIO Pin: GPIO2 (safer choice than GPIO12)

ESP32 Pin	Connects To	Description
GPIO2	Anode (+) of LED	Long leg of LED
GND	One side of 220-ohm resistor	
Other side of resistor	Cathode (-) of LED	Short leg of LED

Note: GPIO12 can cause boot issues. GPIO2, 4, 5, 13, and 14 are safer for LED control.

4. Code for LED Blink

```
#include <Arduino.h>
#define led 12
void setup()
{
  pinMode(led,OUTPUT);
}
void loop()
{
  digitalWrite(led,HIGH);
  delay(1000);
  digitalWrite(led,LOW);
  delay(1000);
}
```

5. PlatformIO Setup

1. Create a new project in PlatformIO.
2. Select the board: `esp32 WROOM 32` (depending on your model).
3. Ensure `platformio.ini` contains:

```
[env:upesy_wroom]
platform = espressif32
board = upesy_wroom
framework = arduino
```

4. Upload the code and observe the blinking LED.

6. Troubleshooting Tips

- Ensure GPIO2 is not pulled LOW during boot.
- Use a current-limiting resistor to protect the LED.
- Confirm pin connections match the code.
- Use the Serial Monitor for debugging if needed.

7. Conclusion

You've successfully created a simple LED blink project using the ESP32-WROOM-32 and PlatformIO. This is a foundational step toward more advanced embedded systems development.

Certainly! Here's a line-by-line explanation of your Arduino code that blinks an LED:

---

cpp
CopyEdit
```
#include <Arduino.h>
```

Explanation:
This includes the core Arduino functions (like `pinMode()`, `digitalWrite()`, and `delay()`). It's automatically included in most Arduino sketches by the IDE, but when writing code in certain development environments (like PlatformIO), it might be required explicitly.

CODE EXPLANATION

```
#define led 12
```

Explanation:
This defines a macro named `led` that will be replaced with the number `12` by the preprocessor. It makes the code easier to read and change. So whenever you use `led` later in the code, it means pin 12.

```
void setup()
```

Explanation:
This is a special function in Arduino that runs **once** when the board is powered on or reset. It's used to set up things like pin modes or initializing communication.

```
{
  pinMode(led, OUTPUT);
}
```

Explanation:
This sets digital pin 12 (the `led` pin) as an output pin, meaning it can send voltage (either HIGH or LOW). This is necessary before you can control the pin.

```
void loop()
```

Explanation:
This is another special function in Arduino that runs **continuously in a loop** after `setup()` finishes. It repeats forever until the board is reset or powered off.

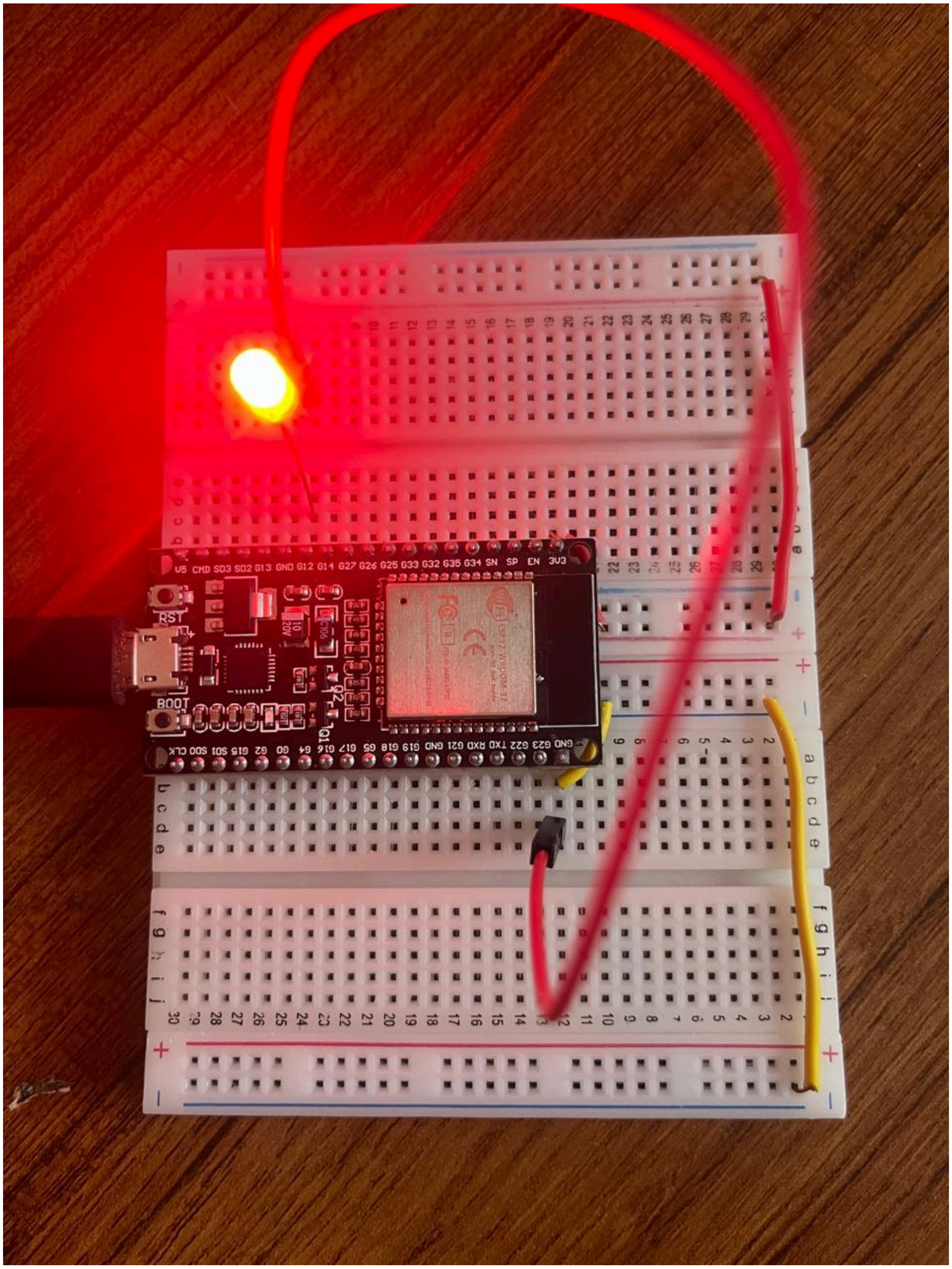
```
{
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

Explanation:

- `digitalWrite(led, HIGH);` — Turns the LED **on** by sending 5V (or 3.3V depending on the board) to pin 12.
- `delay(1000);` — Waits for 1000 milliseconds (1 second).
- `digitalWrite(led, LOW);` — Turns the LED **off** by sending 0V to pin 12.
- `delay(1000);` — Waits for another 1 second.

This cycle continues indefinitely, making the LED blink on and off every second.

So in summary, the code turns the LED on for 1 second, off for 1 second, and keeps repeating that forever.


V5 CMD SD3 SD2 G13 GND G12 G14 G27 G26 G25 G33 G32 G35 G34 SN SP EN 3V3
RST
BOOT
CLK SD0 SD1 G15 G2 G0 G4 G16 G17 G5 G18 G19 GND G21 RXD TXD G22 G23 GND