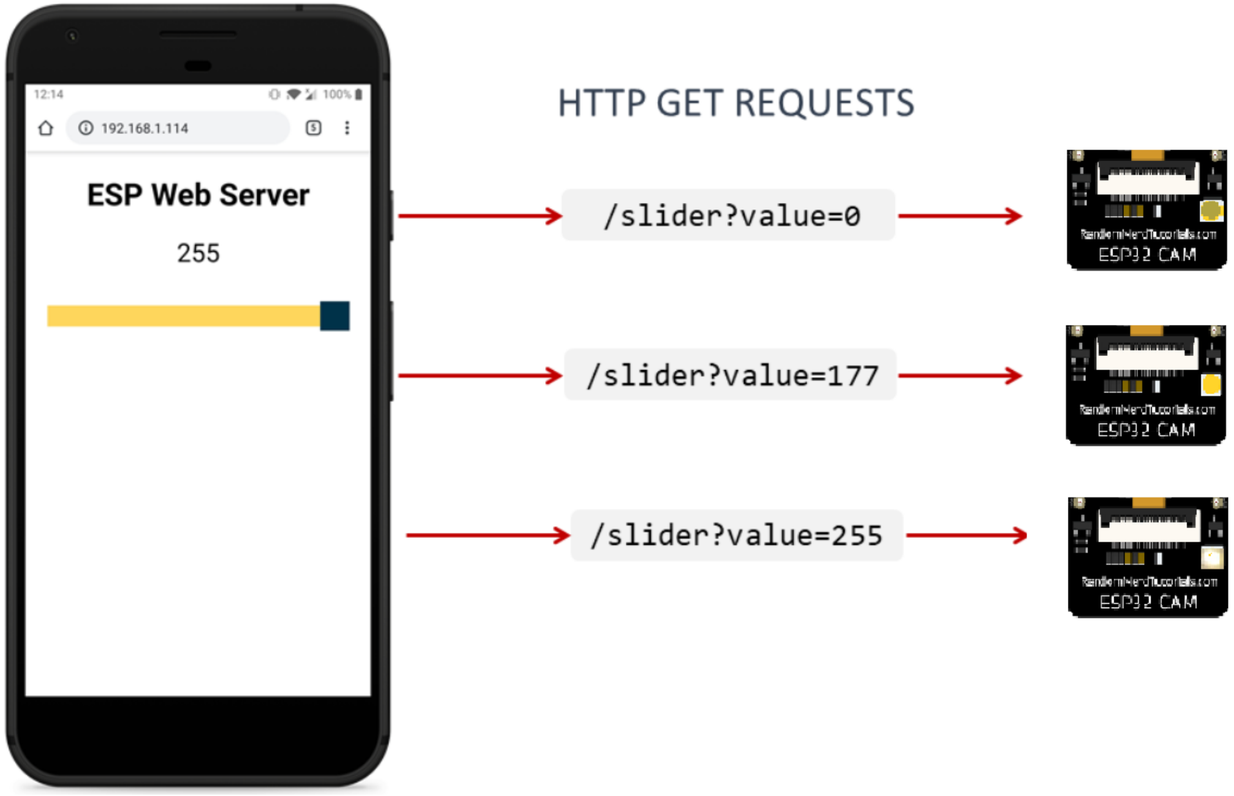
ESP32 ledcwrite PWM control



7/29/2023

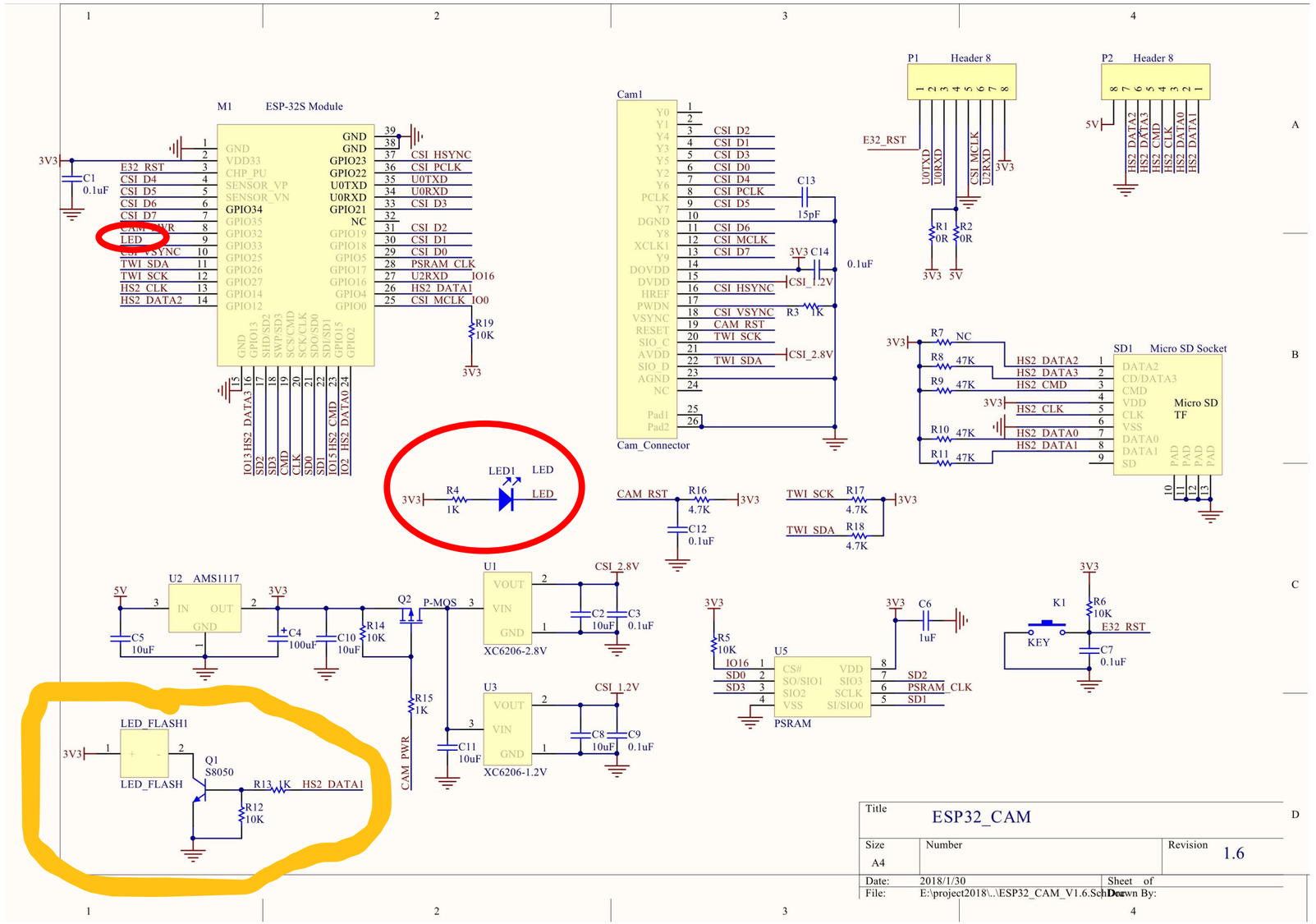
Sangwon Lee

Contents

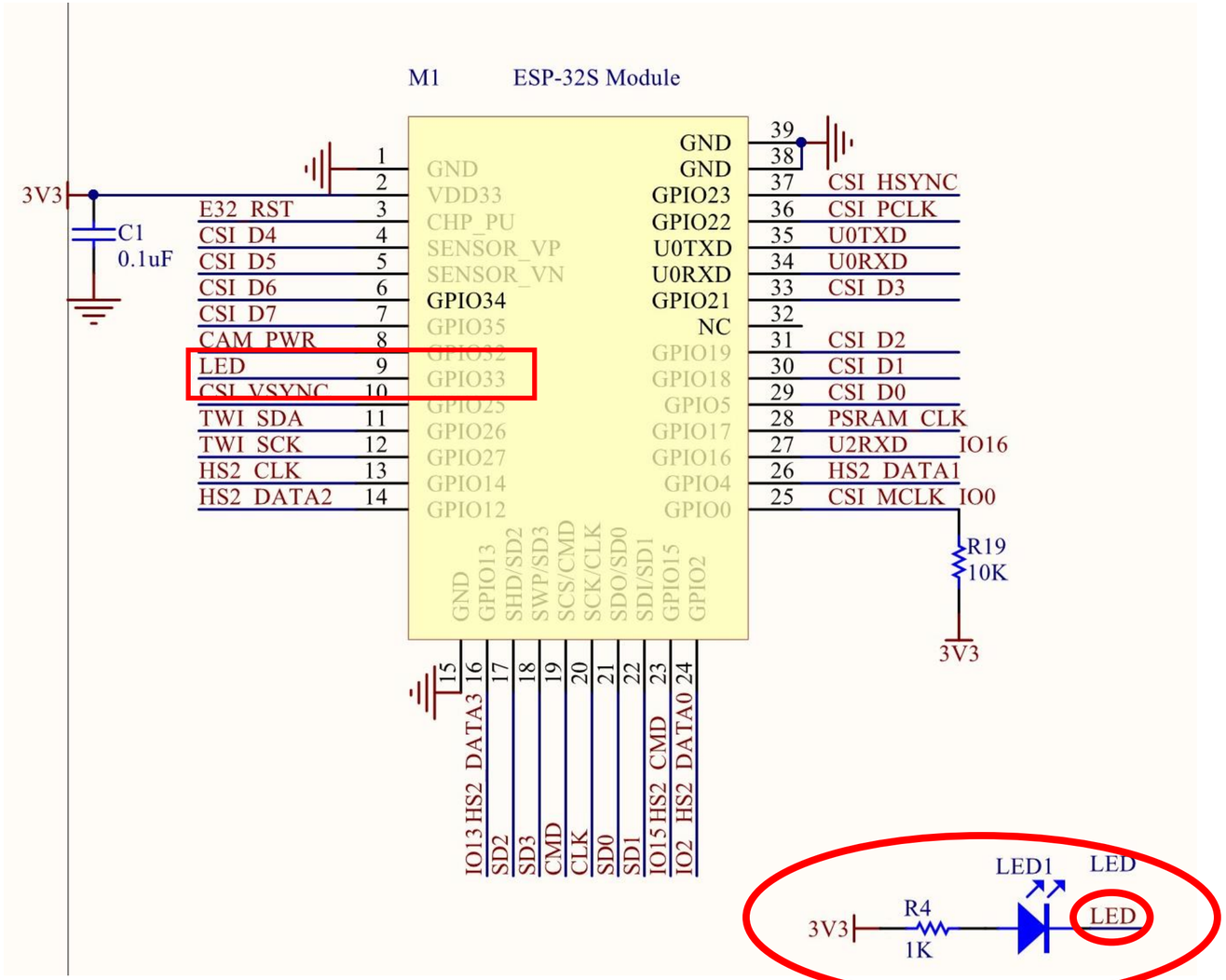
- ESP32-CAM Schematic and LED Pin
- ESP32 PWM Output
- “ledcSetup”, “ledcAttachPin”, “ledcwrite”
- Library Installation - AsyncTCP-master and ESPAsyncWebServer-master
- Setup SPIFFS – File System Uploader
- ESP32 LED Web Server Control
- Motor Web Server Control

- Reference:
  https://randomnerdtutorials.com/esp32-web-server-slider-pwm/

ESP32 CAM Schematic



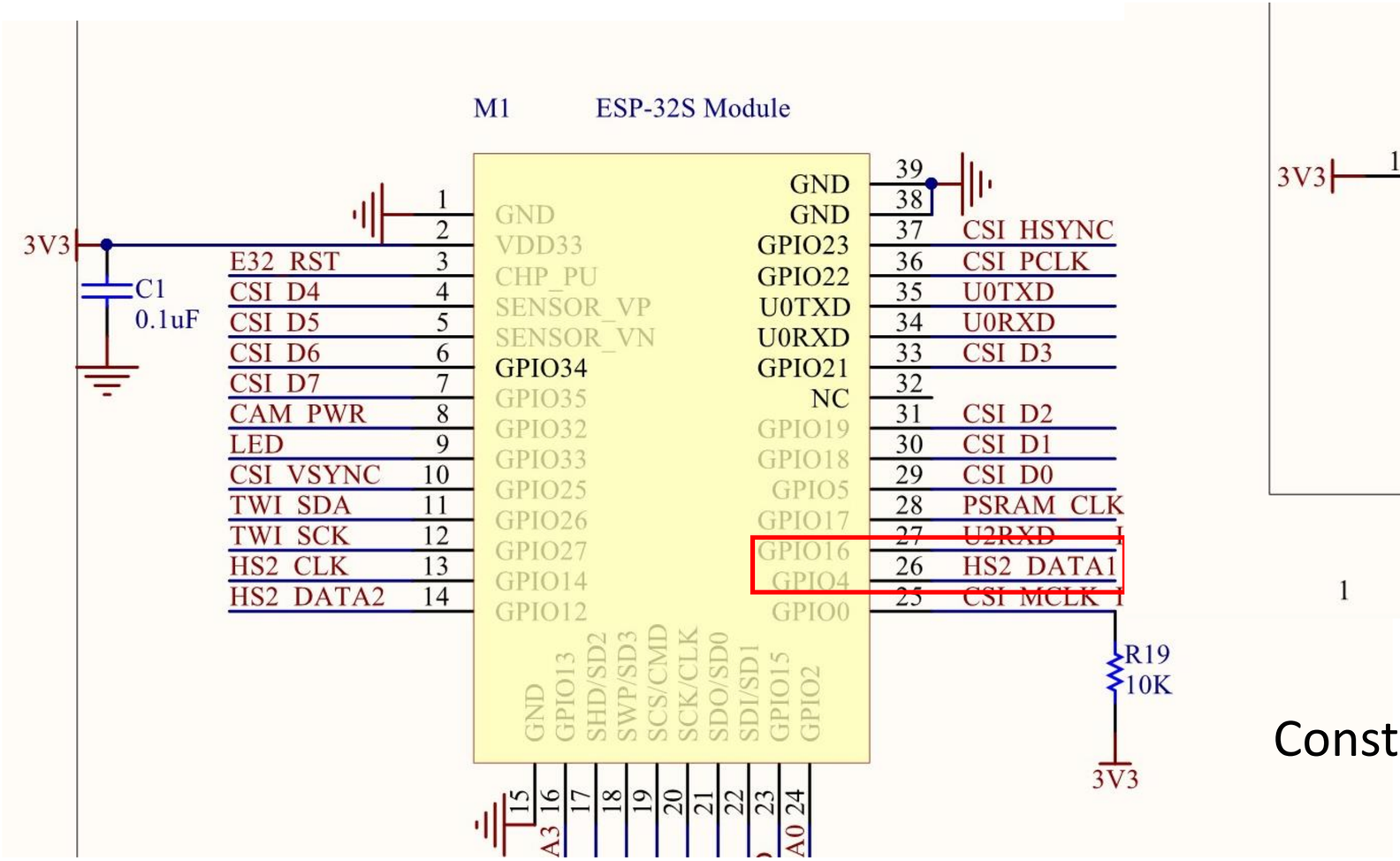
ESP32 CAM Internal LED Pin

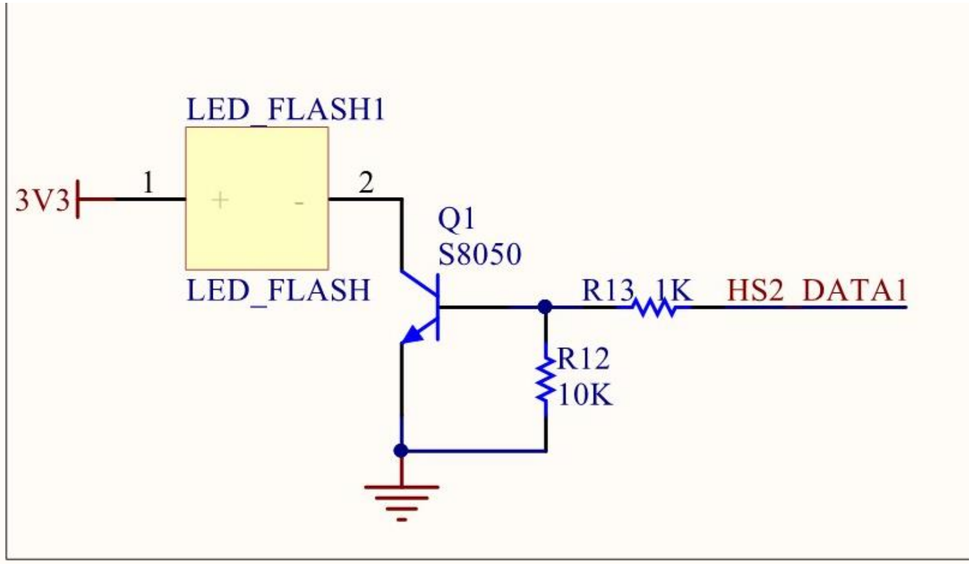


~~Const int ESP32CAM_LED_PIN = 33~~

Const int ESP32CAM_LED_PIN = 4

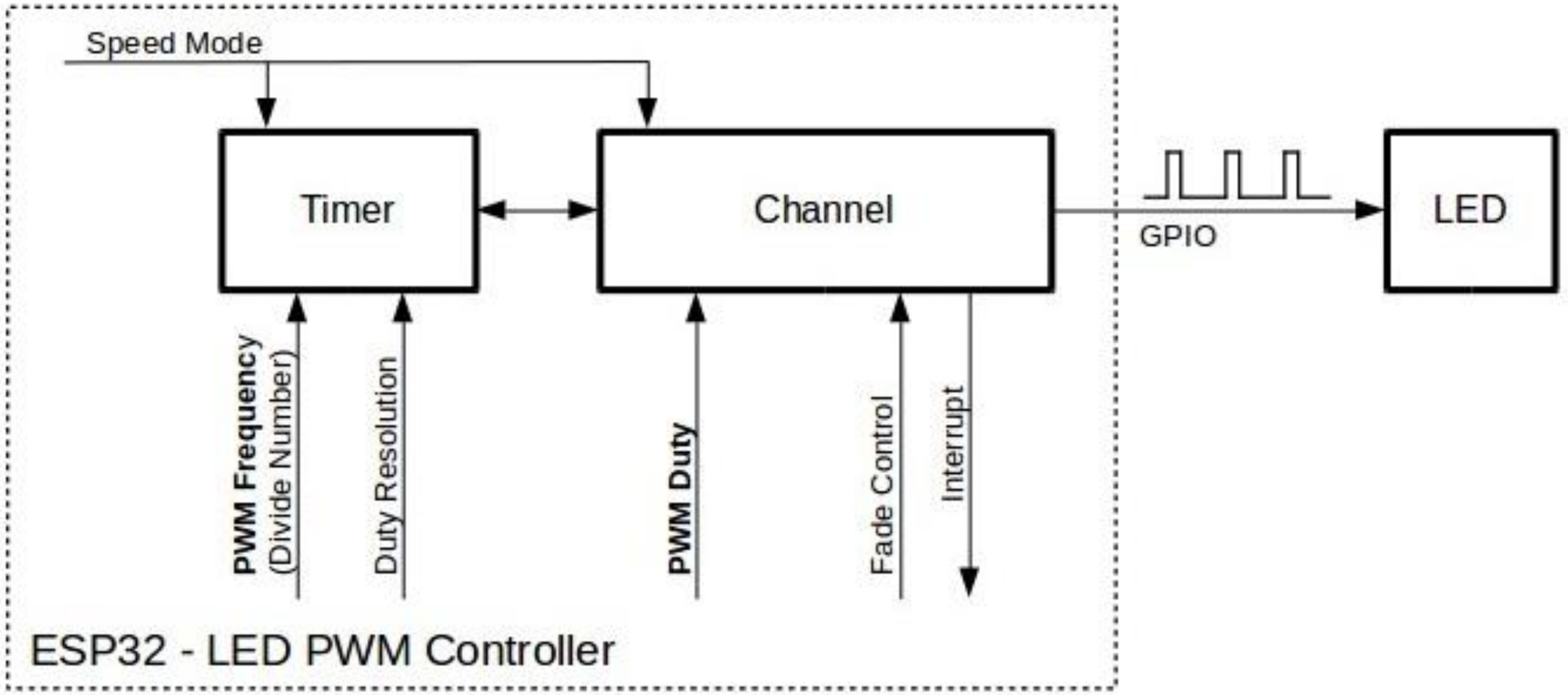
ESP32 CAM Internal LED Pin





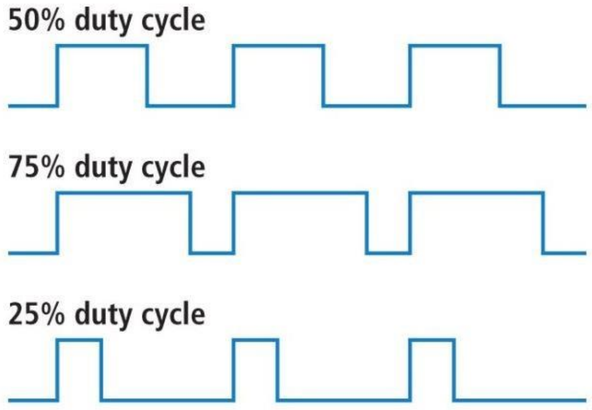
Const int ESP32CAM_LED_PIN = 4

ESP32 PWM Output

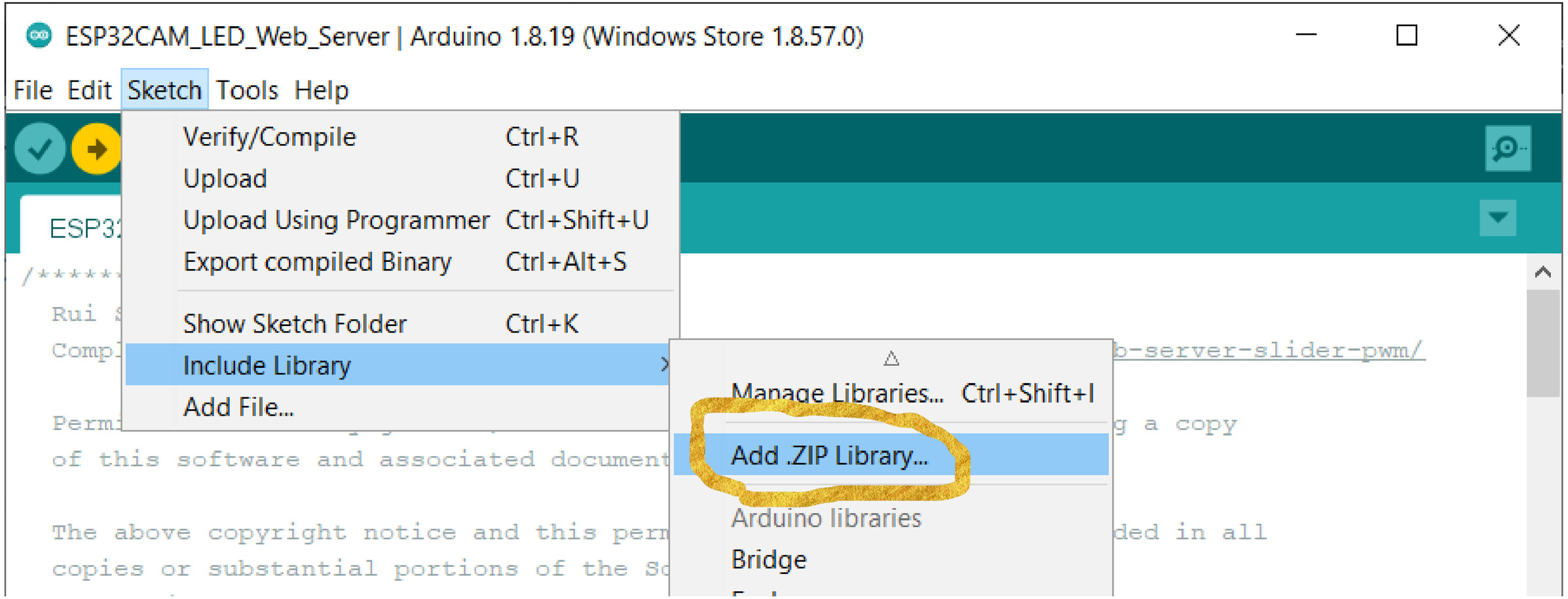


ledcSetup, ledcAttachPin, ledcwrite

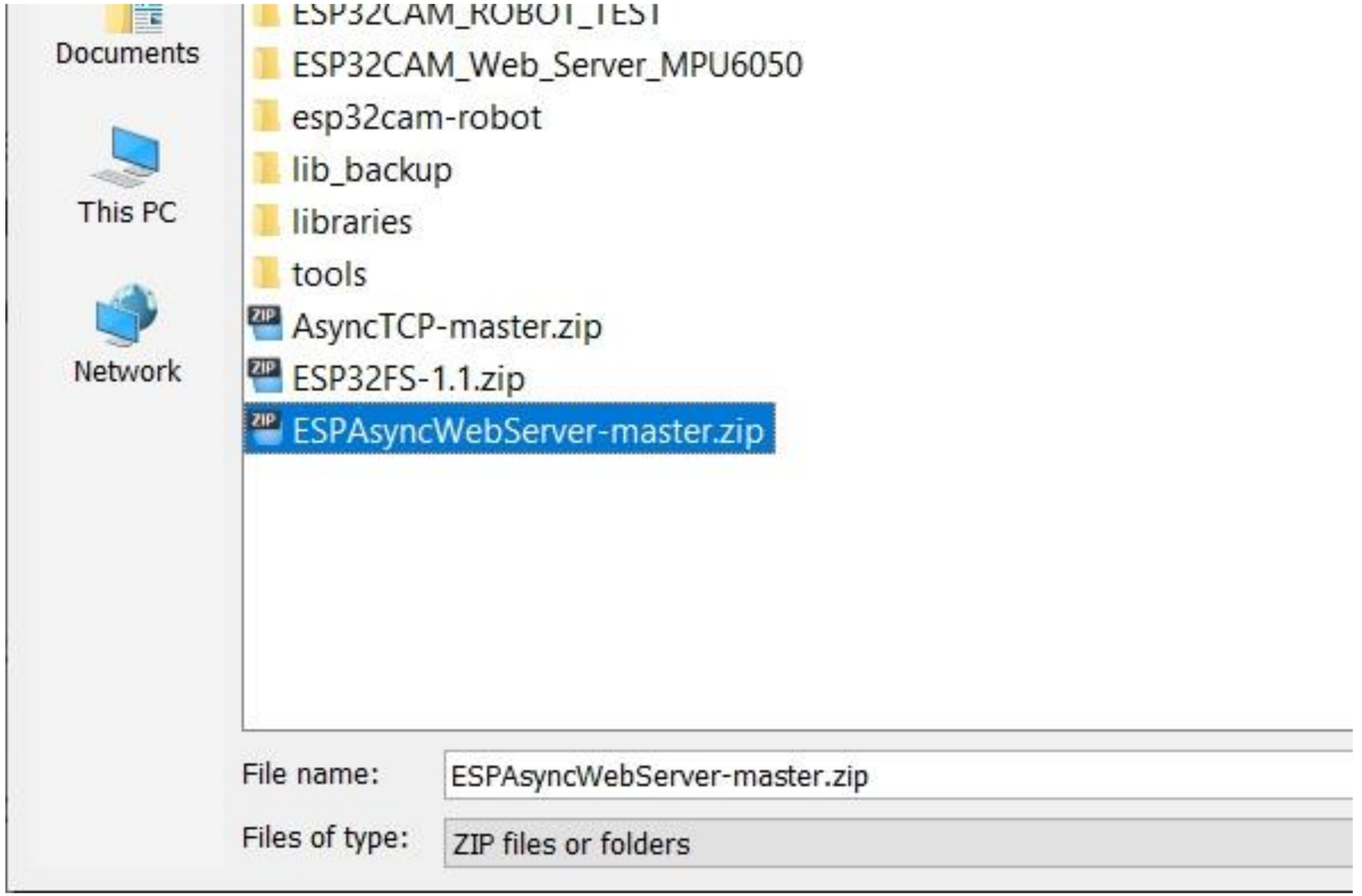
```
// ledcSetup (ledcChannel_n, frequency, resolution)
//     : ledcChannel_n is the internal PWM generator number
//     : frequency is the frequency of ledcChannel_n
//     : resolution is the 100% duty value
//
// for example,
// PWM generator 0 and 1 kHz frequency and 8-bit resolution
// ledcSetup(0, 1000, 8)

// ledcAttachPin(GPIO_n, ledcChannel_n)
//     : connect PWM generator to real GPIO pin
//     : ledcAttachPin(33, 0)

// ledcwrite(ledcChannel_n, duty)
//     : change PWM generator duty cycle
//     : ledcwrite(0, 127)
```



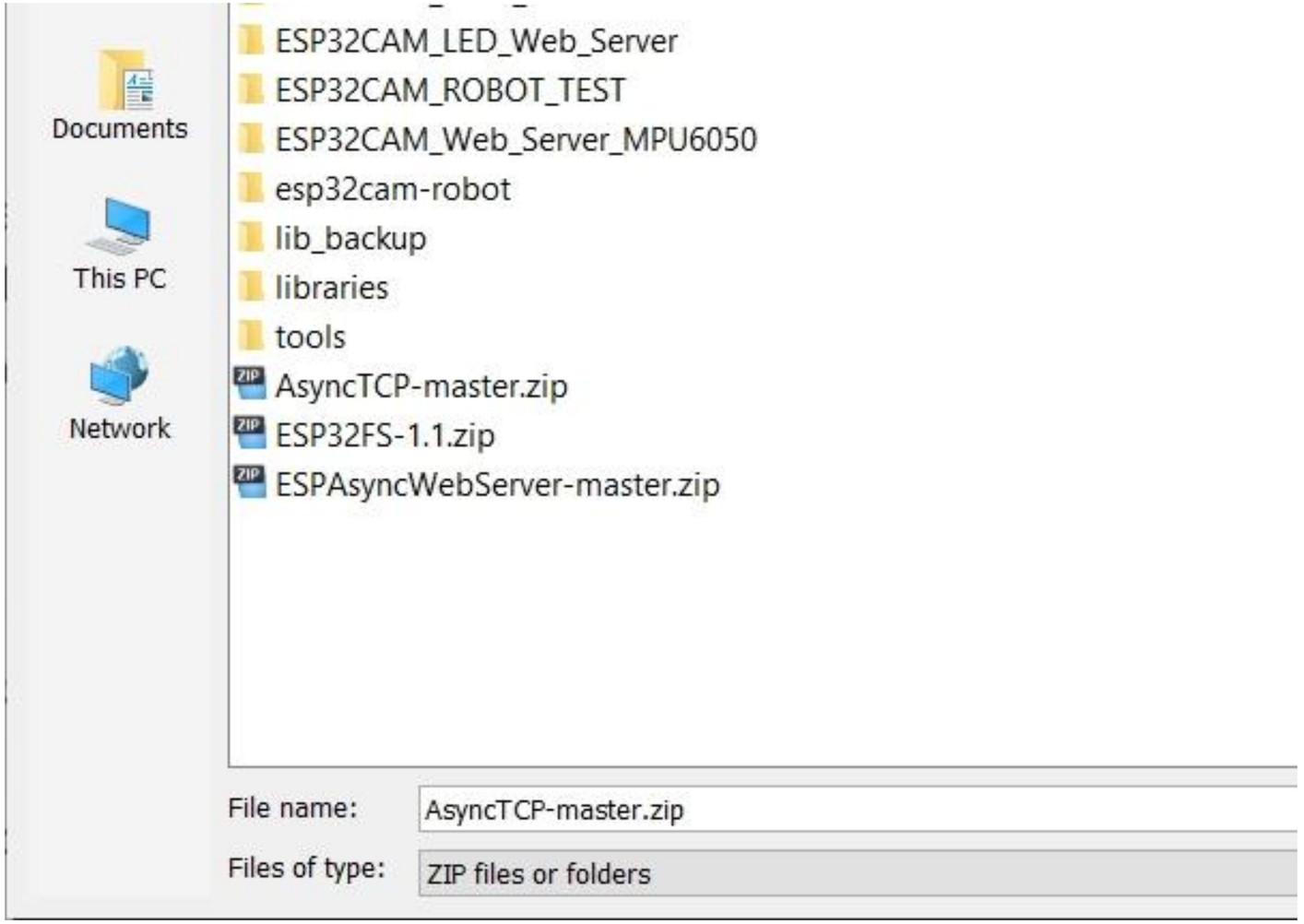
Add .zip libraries



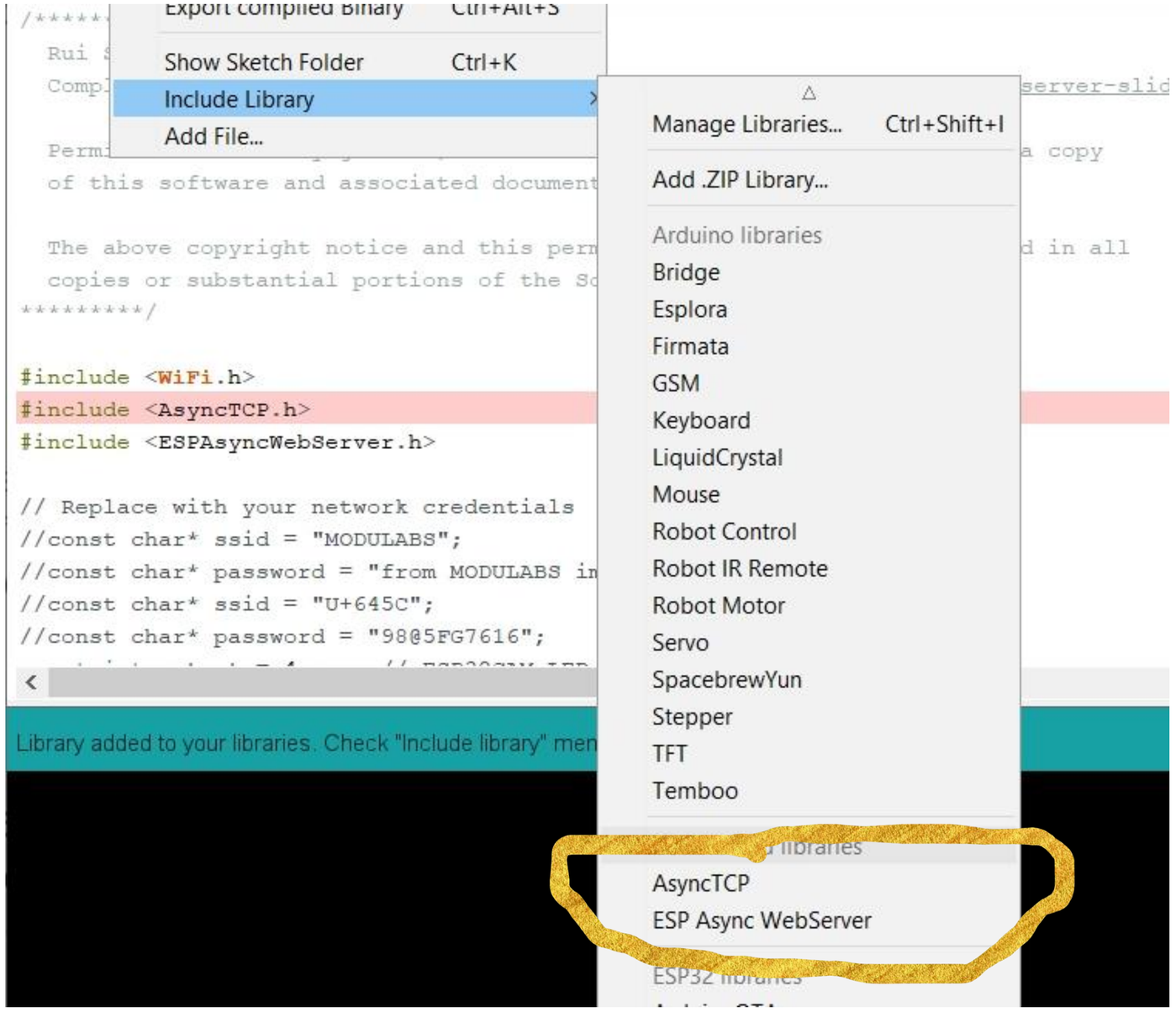
Add .zip libraries



Add .zip libraries



Add .zip libraries



ESP32CAM LED Web Server Control

- Reference:
  https://randomnerdtutorials.com/esp32-web-server-slider-pwm/

- Source:
  https://raw.githubusercontent.com/RuiSantosdotme/Random-Nerd-Tutorials/master/Projects/ESP32/ESP32_Slider_PWM.ino

ESP32CAM LED Web Server Control

```
#include <WiFi.h>
#include <AsyncTCP.h>
#include <ESPAsyncWebServer.h>

// Replace with your network credentials
const char* ssid = "MODULABS";
const char* password = "from MODULABS import future";

const int output = 4;     // ESP32CAM LED is connected to the GPIO4

String sliderValue = "0";

// setting PWM properties
const int freq = 1000;
const int ledChannel = 7;  // ledc channel(PWM control unit) number 7
const int resolution = 8;   // ledc pwm resolution
```

ESP32CAM LED Web Server Control

```
void setup(){

  // Serial port for debugging purposes
  Serial.begin(115200);

  // configure LED PWM functionalitites
  ledcSetup(ledChannel, freq, resolution);                    // 7, 1000, 8

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(output, ledChannel);                          // 4, 7

  ledcWrite(ledChannel, sliderValue.toInt());                 // 7, 0 ~ 255
```

ESP32CAM LED Web Server Control

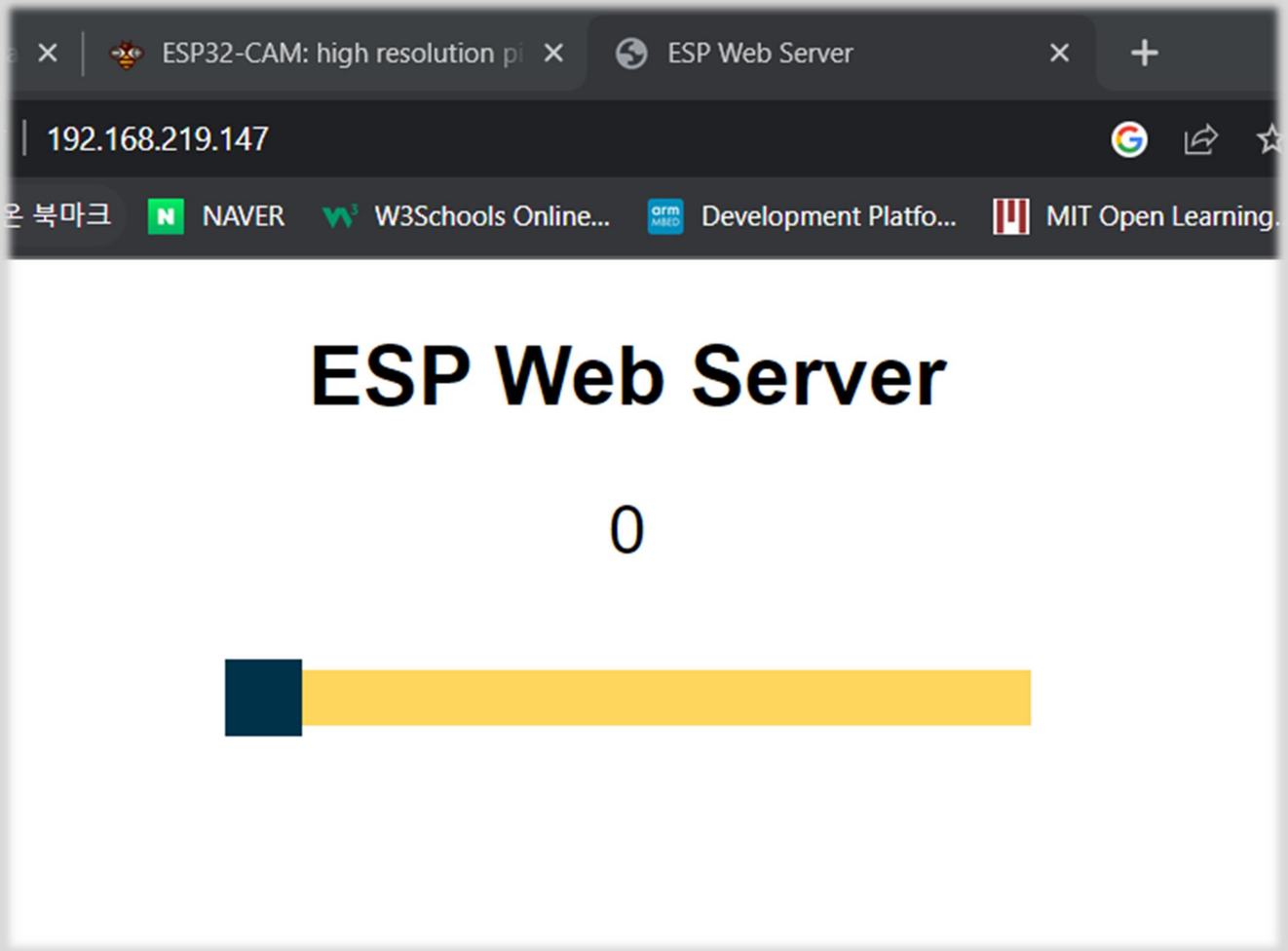
```
// Send a GET request to <ESP_IP>/slider?value=<inputMessage>
 server.on("/slider", HTTP_GET, [] (AsyncWebServerRequest *request) {
  String inputMessage;
  // GET input1 value on <ESP_IP>/slider?value=<inputMessage>
  if (request->hasParam(PARAM_INPUT)) {
    inputMessage = request->getParam(PARAM_INPUT)->value();
    sliderValue = inputMessage;
    ledcWrite(ledChannel, sliderValue.toInt());
  }
  else {
    inputMessage = "No message sent";
  }
  Serial.println(inputMessage);
  request->send(200, "text/plain", "OK");
 });
```

ESP32CAM LED Web Server Control

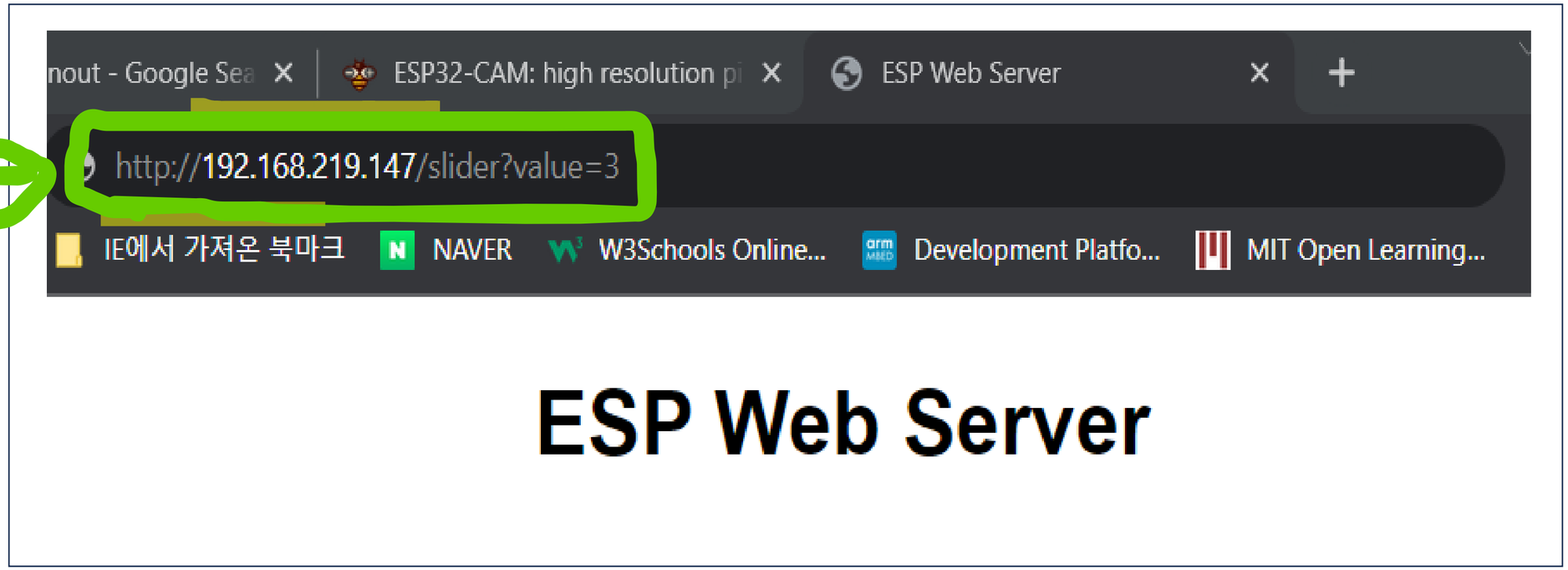
```
<body> <h2>ESP Web Server</h2>
    <p><span id="textSliderValue">%SLIDERVALUE%</span></p>
    <p><input type="range" onchange="updateSliderPWM(this)" id="pwmSlider“
     min="0" max="255" value="%SLIDERVALUE%" step="1" class="slider"></p>

<script> function updateSliderPWM(element) {
    var sliderValue = document.getElementById("pwmSlider").value;
    document.getElementById("textSliderValue").innerHTML = sliderValue;
    console.log(sliderValue);
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "/slider?value="+sliderValue, true);
    xhr.send(); }
</script>

</body>
```

ESP32 will receive a request like this **”/slider?value=SLIDERVALUE”**

ESP32CAM LED Web Server Control



ESP32CAM LED Web Server Control



http://YOUR_ESP32CAM_IP_ADDRESS/slider?value=10