ESP32-S3 WROOM-1

I've ordered the ESP32-S3 N16R8 from Aliexpress seems standard. But the pinouts are a bit confusing and take some work to figure out.

This book explains the different aspects of using this IC and how we got there.

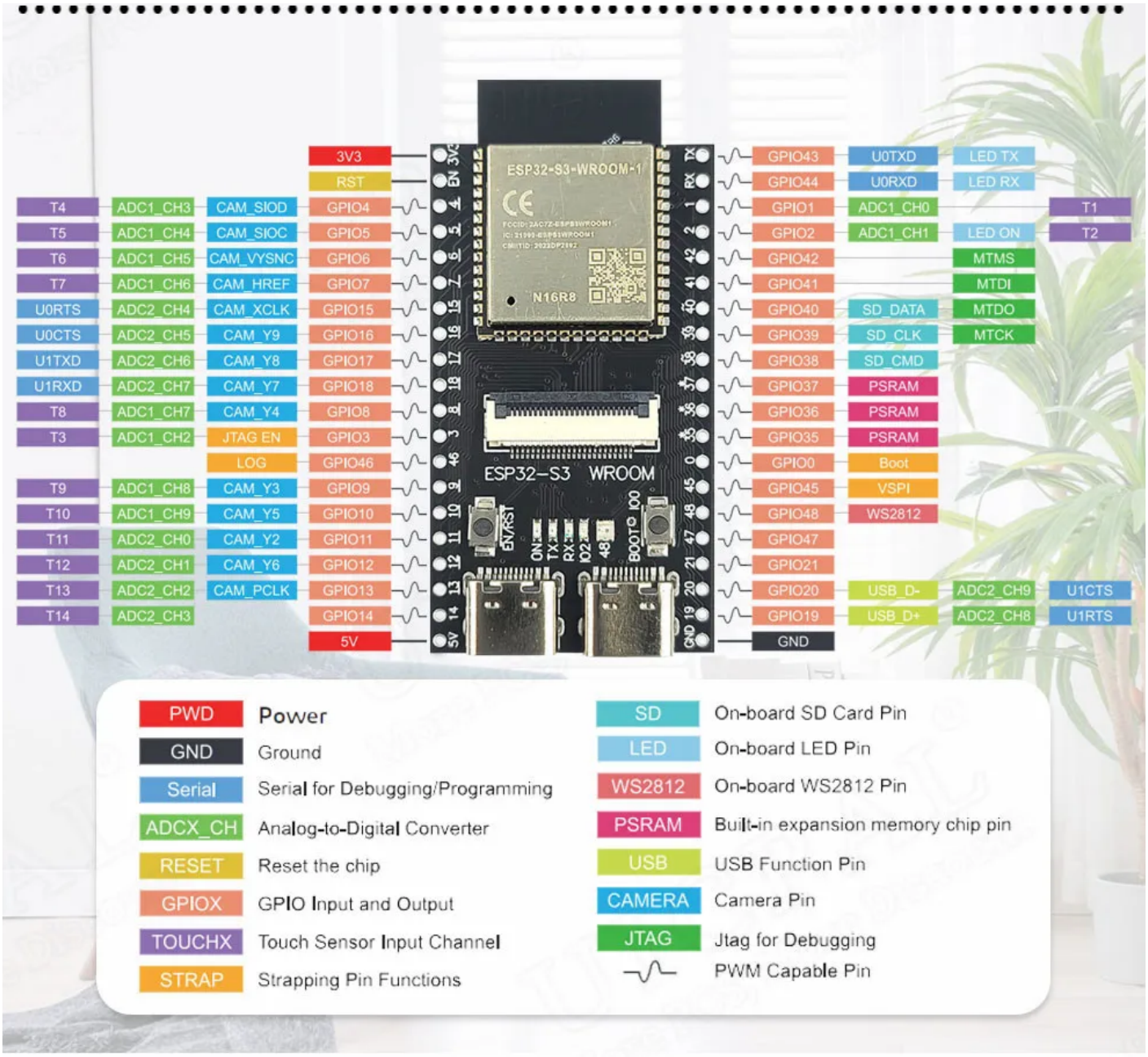
https://www.aliexpress.com/item/1005006434168488.html?spm=a2g0o.order_list.order_list_main.17.488d1802nm70nK

SD Card interfacing

Trying to interface with this onboard SD card slot didn't go as easily as I had expected: it requires a different approach as the basic sample. It only works in "single bit mode".

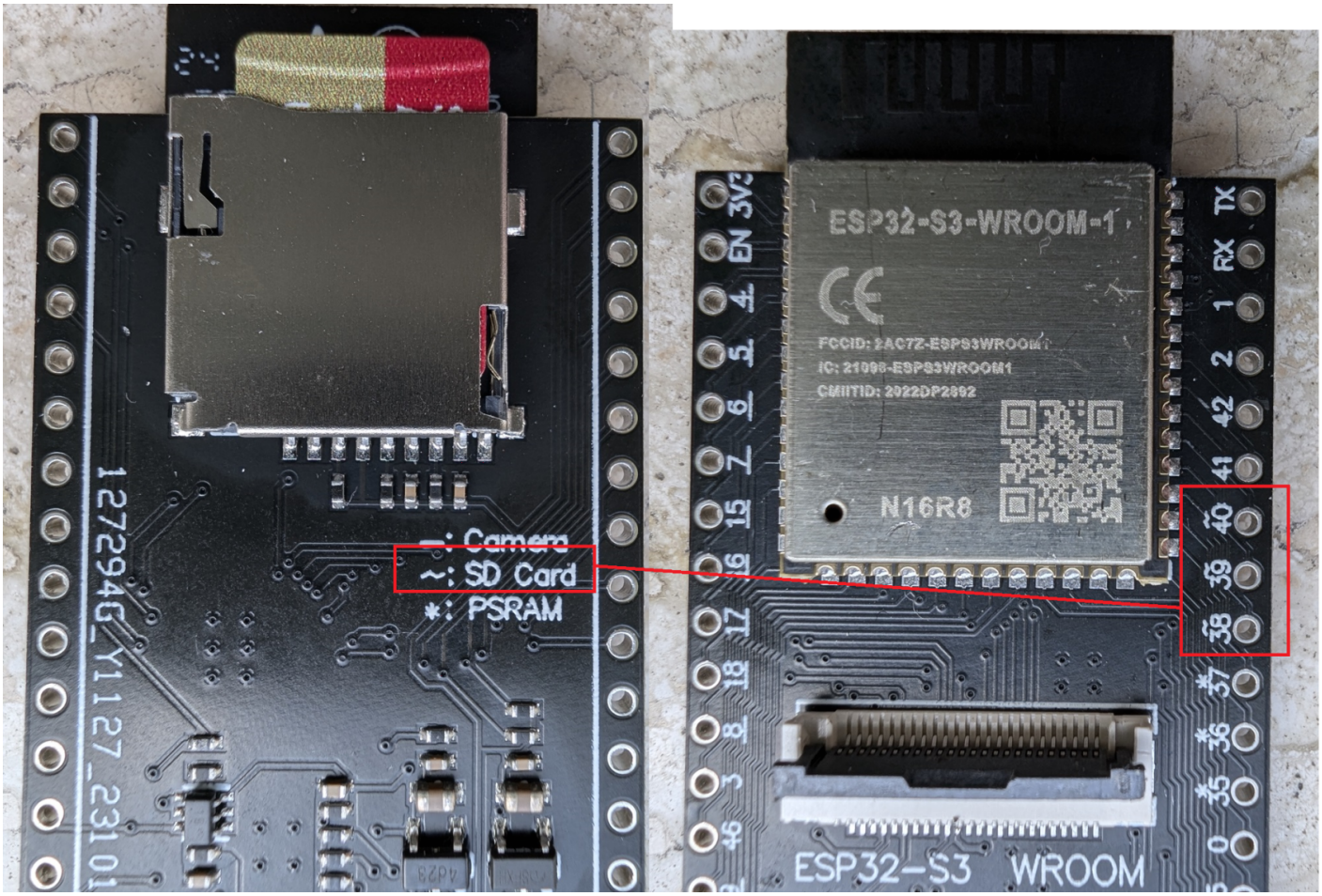
The first thing to do, is to find the pins that are relevant.

I could find for this board this schematic: notice in the right blue blocks we have "SD_DATA", "SD_CLK" and "SD_CMD"



When we inspect the board itself, we can see a small legend that doesn't seem to make so much sense, until we flip the board around and can find the icons next to the port. So this gives a clue

where to start.



We now know there are 3 pins that are used for the SD-card, namely 38, 39, 40.

We can now look for code samples that include these pinouts or write our own.

The below example shows working code to write text to the SDCard.

```
#include "esp_vfs_fat.h"
#include "driver/sdmmc_host.h"
#include "sdmmc_cmd.h"
#include "SD.h"

static const char *TAG = "example";
#define MOUNT_POINT "/sdcard"
#define EXAMPLE_MAX_CHAR_SIZE    64

esp_err_t ret;
void setup() {
```

```
  Serial.begin(115200);
  // Initialize SDMMC host driver
  sdmmc_host_t host = SDMMC_HOST_DEFAULT();
  sdmmc_slot_config_t slot_config = SDMMC_SLOT_CONFIG_DEFAULT();
  slot_config.width = 1;
  slot_config.clk = GPIO_NUM_39;
  slot_config.cmd = GPIO_NUM_38;
  slot_config.d0 = GPIO_NUM_40;

  esp_vfs_fat_sdmmc_mount_config_t mount_config = {
    .format_if_mount_failed = true,
    .max_files = 5,
    .allocation_unit_size = 16 * 1024
  };
  sdmmc_card_t* card;
  esp_err_t err = esp_vfs_fat_sdmmc_mount("/sdcard", &host, &slot_config, &mount_config, &card);

  if (err != ESP_OK) {
    Serial.printf("Failed to mount SD card (%s)\n", esp_err_to_name(err));
    return;
  }
  sdmmc_card_print_info(stdout, card);

  // First create a file.
    const char *file_hello =  MOUNT_POINT"/hello.txt";
    char data[EXAMPLE_MAX_CHAR_SIZE];
    snprintf(data, EXAMPLE_MAX_CHAR_SIZE, "%s %s!\n", "Hello", card->cid.name);
    ret = s_example_write_file(file_hello, data);
    if (ret != ESP_OK) {
        return;
    }
}
void loop()
{

}
static esp_err_t s_example_write_file(const char *path, char *data)
{
    ESP_LOGI(TAG, "Opening file %s", path);
    FILE *f = fopen(path, "w");
```

```
    if (f == NULL) {
        ESP_LOGE(TAG, "Failed to open file for writing");
        return ESP_FAIL;
    }
    fprintf(f, data);
    fclose(f);
    ESP_LOGI(TAG, "File written");

    return ESP_OK;
}

static esp_err_t s_example_read_file(const char *path)
{
    ESP_LOGI(TAG, "Reading file %s", path);
    FILE *f = fopen(path, "r");
    if (f == NULL) {
        ESP_LOGE(TAG, "Failed to open file for reading");
        return ESP_FAIL;
    }
    char line[EXAMPLE_MAX_CHAR_SIZE];
    fgets(line, sizeof(line), f);
    fclose(f);

    // strip newline
    char *pos = strchr(line, '\n');
    if (pos) {
        *pos = '\0';
    }
    ESP_LOGI(TAG, "Read from file: '%s'", line);

    return ESP_OK;
}
```

Camera interfacing

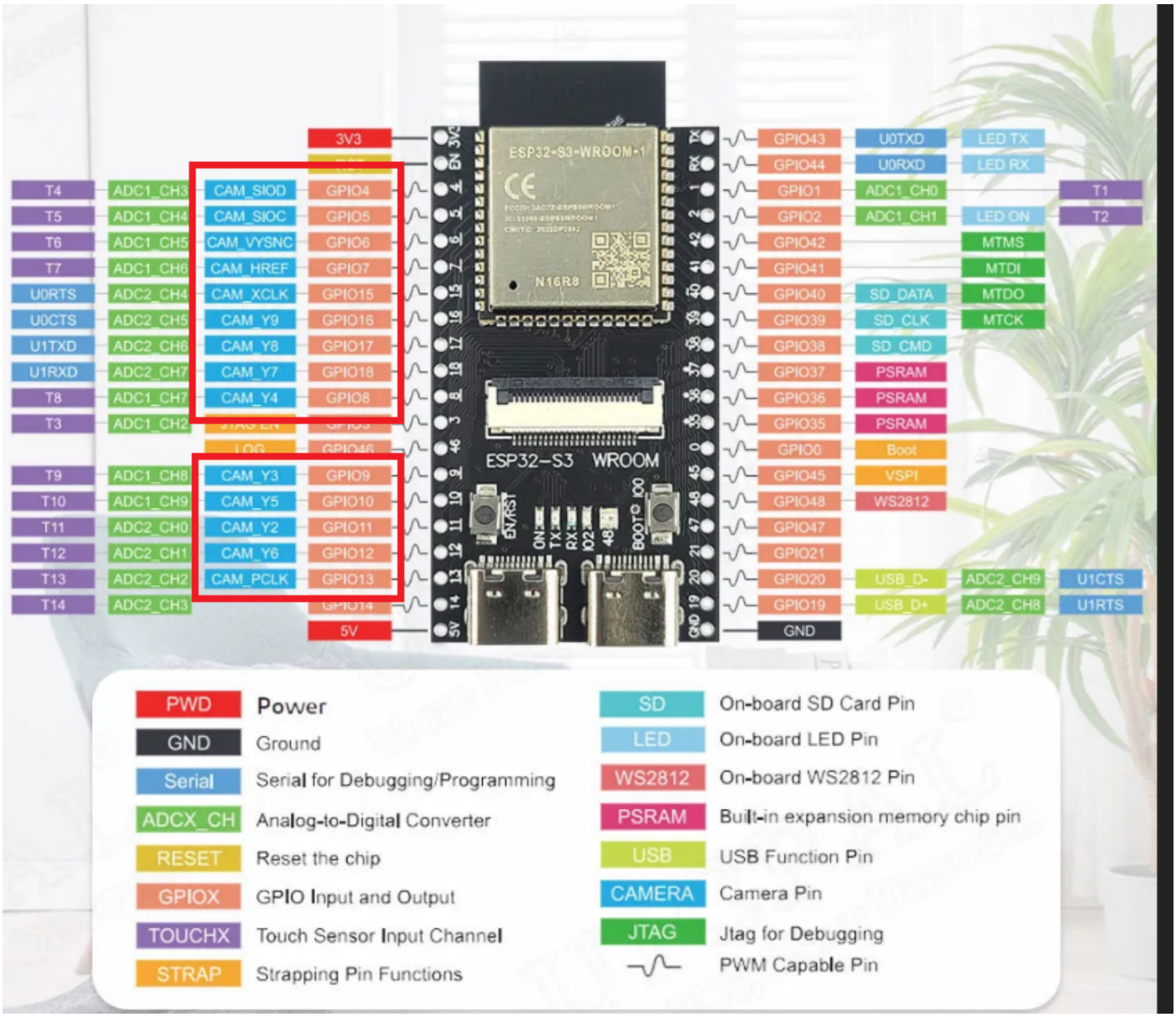
How to use the camera

Configuration and initialization

There are plenty of examples how to interface with the camera, but you'll find that you'll spend time trying to figure out "why it doesn't work. Where many posts will suggest "you have a faulty model", it's very likely you didn't find the correct pin-out schematic.

Or you have selected the wrong module for example.

So we have to check the schematic back.



We then can infer thee next configuration:

```
#define PWDN_GPIO_NUM -1
#define RESET_GPIO_NUM -1

#define XCLK_GPIO_NUM 15
#define SIOD_GPIO_NUM 4
#define SIOC_GPIO_NUM 5

#define Y2_GPIO_NUM 11
#define Y3_GPIO_NUM 9
#define Y4_GPIO_NUM 8
#define Y5_GPIO_NUM 10
#define Y6_GPIO_NUM 12
#define Y7_GPIO_NUM 18
```

```
#define Y8_GPIO_NUM 17
#define Y9_GPIO_NUM 16

#define VSYNC_GPIO_NUM 6
#define HREF_GPIO_NUM 7
#define PCLK_GPIO_NUM 13
```

It's then easy to follow most examples online, after you set the camera configuration.
Be aware that the order of the parameters is fixed, else the compiler will give an error "***designator order for field XXX does not match declaration order in YYY***"

```
 camera_config_t config =  {
  .pin_pwdn = PWDN_GPIO_NUM,
  .pin_reset = RESET_GPIO_NUM,
  .pin_xclk = XCLK_GPIO_NUM,
  .pin_sscb_sda = SIOD_GPIO_NUM,
  .pin_sscb_scl = SIOC_GPIO_NUM,
  .pin_d7 = Y9_GPIO_NUM,
  .pin_d6 = Y8_GPIO_NUM,
  .pin_d5 = Y7_GPIO_NUM,
  .pin_d4 = Y6_GPIO_NUM,
  .pin_d3 = Y5_GPIO_NUM,
  .pin_d2 = Y4_GPIO_NUM,
  .pin_d1 = Y3_GPIO_NUM,
  .pin_d0 = Y2_GPIO_NUM,
  .pin_vsync = VSYNC_GPIO_NUM,
  .pin_href = HREF_GPIO_NUM,
  .pin_pclk = PCLK_GPIO_NUM,
  .xclk_freq_hz = 20000000,
  .ledc_timer = LEDC_TIMER_0,
  .ledc_channel = LEDC_CHANNEL_0,
  .pixel_format = PIXFORMAT_JPEG,//YUV422,GRAYSCALE,RGB565,JPEG
  .frame_size = FRAMESIZE_UXGA,//QQVGA-UXGA, For ESP32, do not use sizes above QVGA when not JPEG. The
performance of the ESP32-S series has improved a lot, but JPEG mode always gives better frame rates.
  .jpeg_quality = 12, //0-63, for OV series camera sensors, lower number means higher quality
  .fb_count = 1, //When jpeg mode is used, if fb_count more than one, the driver will work in continuous mode.
  .grab_mode = CAMERA_GRAB_WHEN_EMPTY//CAMERA_GRAB_LATEST. Sets when buffers should be filled
  //init with high specs to pre-allocate larger buffers
  };
```

The function to initialize the camera is then simple as this:

```
void startCamera()
{

  if (psramFound()) {
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

  // Camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    ESP.restart();
  }
  else
  {
    Serial.printf("Camera init succeeded");
  }
}
```

Errors encountered

error 0x105(ESP_ERR_NOT_FOUND)

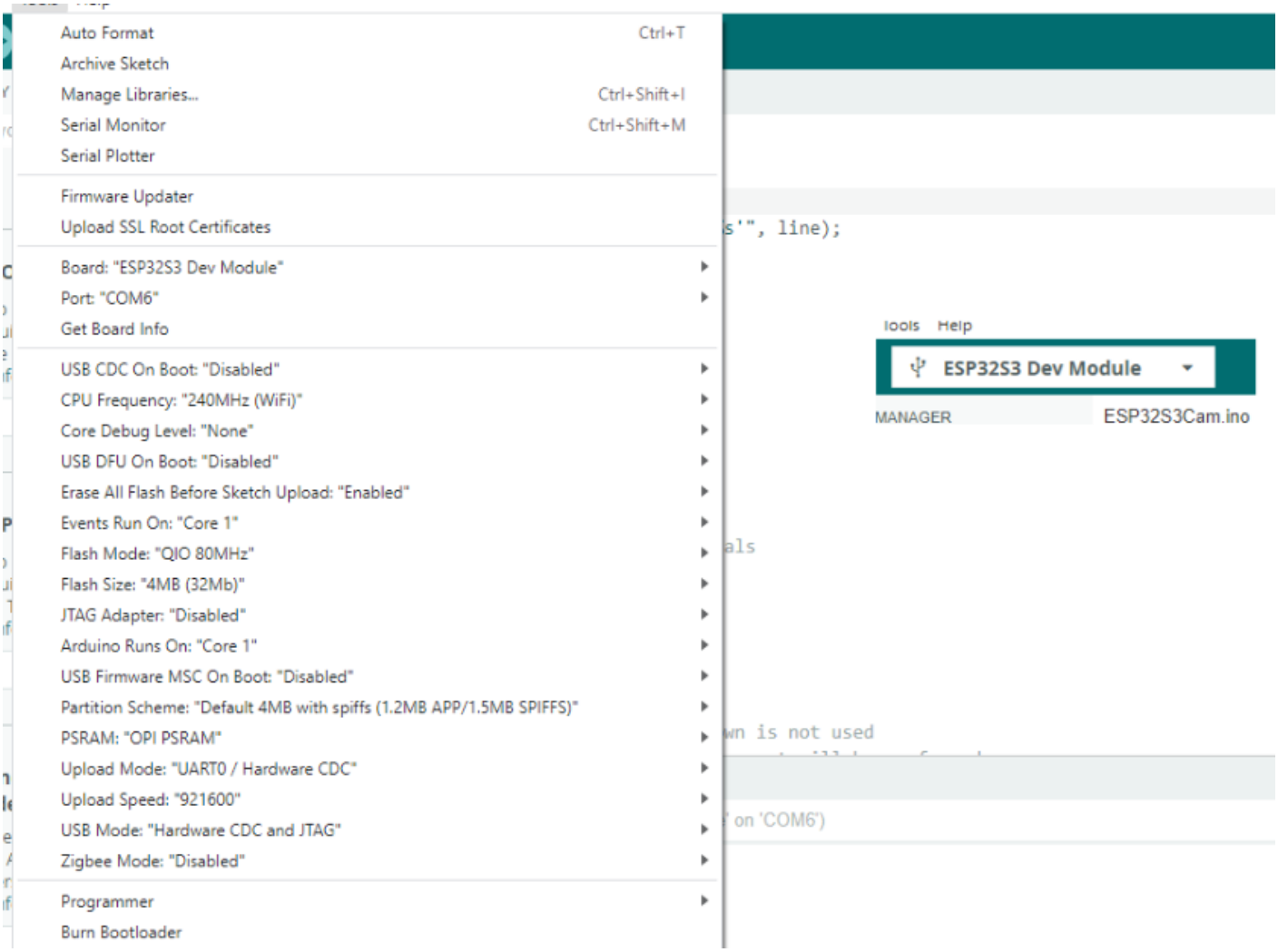
This error can mean either:

- The camera isn't seated properly
- The pin configuration is wrong
- You've selected the wrong board

```
E (893) camera: Camera probe failed with error 0x105(ESP_ERR_NOT_FOUND)
Camera init failed with error 0x105
```

In the case of the ESP32-S3-WROOM, selecting WROOM2 generates this error.

These settings work properly:


Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
Firmware Updater
Upload SSL Root Certificates
Board: "ESP32S3 Dev Module"
Port: "COM6"
Get Board Info
USB CDC On Boot: "Disabled"
CPU Frequency: "240MHz (WiFi)"
Core Debug Level: "None"
USB DFU On Boot: "Disabled"
Erase All Flash Before Sketch Upload: "Enabled"
Events Run On: "Core 1"
Flash Mode: "QIO 80MHz"
Flash Size: "4MB (32Mb)"
JTAG Adapter: "Disabled"
Arduino Runs On: "Core 1"
USB Firmware MSC On Boot: "Disabled"
Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
PSRAM: "OPI PSRAM"
Upload Mode: "UART0 / Hardware CDC"
Upload Speed: "921600"
USB Mode: "Hardware CDC and JTAG"
Zigbee Mode: "Disabled"
Programmer
Burn Bootloader
s'", line);
ESP32S3 Dev Module
MANAGER
ESP32S3Cam.ino
wn is not used
on 'COM6')

Errors encountered

designator order for field 'xxx' does not match declaration order in 'yyy'

This error occurs when the "order of declared members" is not in the right order.
I don't know the rules of this language enough, but it would appear there is a fixed order when you do an inline declaration:

This is correct, for example:

```
camera_config_t config =  {
    .pin_pwdn = PWDN_GPIO_NUM,
    .pin_reset = RESET_GPIO_NUM,
    .pin_xclk = XCLK_GPIO_NUM,
....
```

While this one (reset and pwdn are reversed) generates the error:

```
camera_config_t config =  {
    .pin_reset = RESET_GPIO_NUM,
    .pin_pwdn = PWDN_GPIO_NUM,
    .pin_xclk = XCLK_GPIO_NUM,
    ....
```

Webserver

Getting started

This is a simple example, how to use esp_http_server - probably there are libraries to handle it different.
Yet this displays how to connect to the Wifi and start a webserver.
And bind a handler to the endpoint "/" - where we reply with a string "hello world"

```
#include "esp_http_server.h"
#include

// Replace with your network credentials
const char* ssid = "YourWifi";
const char* password = "YourPassWord";


httpd_handle_t stream_httpd = NULL;
httpd_handle_t camera_httpd = NULL;

  static esp_err_t file_handler(httpd_req_t *req) {

       Serial.printf("File handler");

    char ret_homepage[255] = "";

   strcpy(ret_homepage,"Hello world");

  /* Set some custom headers */
  httpd_resp_set_hdr(req, "Connection", "close");
  httpd_resp_set_hdr(req, "Cache-Control", "no-cache");

  /* Send response with custom headers and body set as the
   * string passed in user context*/
  const char *resp_str = (const char*) ret_homepage;

  Serial.print(ret_homepage);
  httpd_resp_send(req, resp_str, HTTPD_RESP_USE_STRLEN);
  ESP_LOGI(TAG, "Response sent for home page request.Time:%s",esp_log_system_timestamp());
  return ESP_OK;
  }
void StartServer(){
  httpd_config_t config = HTTPD_DEFAULT_CONFIG();
  config.server_port = 80;

     httpd_uri_t index_uri = {
    .uri       = "/",
    .method    = HTTP_GET,
```

```
    .handler   = file_handler,
    .user_ctx  = NULL
  };

    Serial.printf("Starting  server on port: '%d'\n", config.server_port);
    if (httpd_start(&stream_httpd, &config) == ESP_OK) {
        httpd_register_uri_handler(stream_httpd, &index_uri);
    }
}



void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println("Started");



Serial.println("Wifi");
  // Wi-Fi connection
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");


  Serial.print("Camera Stream Ready! Go to: http://");
  Serial.println(WiFi.localIP());


  Serial.println("Start  Server");
  StartServer();

}

void loop() {
Serial.println("Loop");
//s_listFiles("/sdcard");
// captureImage();
delay(10000);

}
```

Read querystring

Alter the config, so the uri_match_fn is wildcard match

```
  httpd_config_t config = HTTPD_DEFAULT_CONFIG();
  config.server_port = 80;
  config.uri_match_fn = httpd_uri_match_wildcard;

     httpd_uri_t index_uri = {
   .uri       = "/*",
   .method    = HTTP_GET,
   .handler   = file_handler,
   .user_ctx  = NULL
  };
```

In the file handler, one can now start to retrieve the query string url.com?test=anser

```
 char query[100];
    size_t query_len = sizeof(query);
    // Get the query string
    if (httpd_req_get_url_query_str(req, query, query_len) == ESP_OK) {
        ESP_LOGI(TAG, "Found URL query => %s", query);

        // Parse specific query parameters
        char param[32];
        if (httpd_query_key_value(query, "param", param, sizeof(param)) == ESP_OK) {
            ESP_LOGI(TAG, "Found URL query parameter => param=%s", param);
        }
    }
```

This comes together in this sample:

```
#include "esp_http_server.h"
#include
static const char *TAG = "example";
// Replace with your network credentials
const char* ssid = "YourWifi";
const char* password = "YourPass";
```

```
httpd_handle_t stream_httpd = NULL;
httpd_handle_t camera_httpd = NULL;

 static esp_err_t file_handler(httpd_req_t *req) {

char query[100];
   size_t query_len = sizeof(query);
   // Get the query string
   if (httpd_req_get_url_query_str(req, query, query_len) == ESP_OK) {
       ESP_LOGI(TAG, "Found URL query => %s", query);

       // Parse specific query parameters
       char param[32];
       if (httpd_query_key_value(query, "param", param, sizeof(param)) == ESP_OK) {
           ESP_LOGI(TAG, "Found URL query parameter => param=%s", param);
       }
   }
      Serial.printf("File handler");

   char ret_homepage[255] = "";

 strcpy(ret_homepage,"Hello world");

 /* Set some custom headers */
 httpd_resp_set_hdr(req, "Connection", "close");
 httpd_resp_set_hdr(req, "Cache-Control", "no-cache");

 /* Send response with custom headers and body set as the
  * string passed in user context*/
 const char *resp_str = (const char*) ret_homepage;

 Serial.print(ret_homepage);
 httpd_resp_send(req, resp_str, HTTPD_RESP_USE_STRLEN);
 ESP_LOGI(TAG, "Response sent for home page request.Time:%s",esp_log_system_timestamp());
 return ESP_OK;
 }
void StartServer(){
 httpd_config_t config = HTTPD_DEFAULT_CONFIG();
 config.server_port = 80;
 config.uri_match_fn = httpd_uri_match_wildcard;

     httpd_uri_t index_uri = {
   .uri       = "/*",
   .method    = HTTP_GET,
   .handler   = file_handler,
   .user_ctx  = NULL
 };

   Serial.printf("Starting  server on port: '%d'\n", config.server_port);
   if (httpd_start(&stream_httpd, &config) == ESP_OK) {
       httpd_register_uri_handler(stream_httpd, &index_uri);
   }
}
```

```
void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println("Started");



Serial.println("Wifi");
  // Wi-Fi connection
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");


  Serial.print("Camera Stream Ready! Go to: http://");
  Serial.println(WiFi.localIP());


  Serial.println("Start  Server");
  StartServer();

}

void loop() {
Serial.println("Loop");
//s_listFiles("/sdcard");
// captureImage();
delay(10000);

}
```