

The camera in your browser with the ESP32-S3

FireBeetle 2 ESP32-S3 and OV2640: the board takes a shot when the browser asks, one photo per second. PSRAM, settings and the manufacturer's pins.

By **Ada Intelisano** · informaticait.it technical team



Online guide,
always up to date

IN SHORT

- Camera cable in the CAM connector, pin 1 toward the white dot.
- Memory: 16 MB and octal PSRAM, a partition with 3 MB for the sketch.
- The page asks for /photo every second; the board answers with a JPEG.
- For real video there is Espressif's CameraWebServer example.

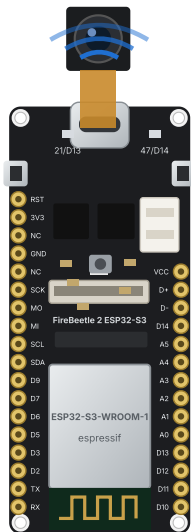
Time: 1 hour

Level: next steps

Before: guides no. 56 and 59

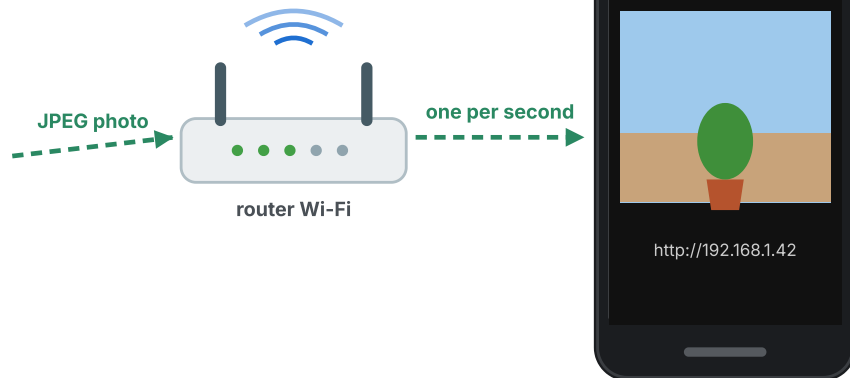
A small Wi-Fi camera: the board takes a photo every time the browser asks for one, and the page asks for one per second. To keep an eye on your front door, a bird feeder, a 3D printer at work. We use the FireBeetle 2 ESP32-S3 sold with its own camera.

1 The board



FireBeetle 2 ESP32-S3 + OV2640

The camera in your browser



informaticait.it

- **ESP32-S3** module with 16 MB of memory for sketches and 8 MB of **PSRAM**, the extra memory that images need.
- 2-megapixel **OV2640** camera (up to 1600×1200).
- The camera's flat cable goes into the **CAM** connector, below the USB socket, with pin 1 toward the white dot.

New word: PSRAM

An extra fast memory next to the processor. Photos take up a lot of memory: without PSRAM you can only use small images.

2 The settings

This board has to be described properly to the compiler. In PlatformIO the file below does it: it says the memories are 16 MB and "octal" PSRAM, and asks for a memory layout with room for large sketches.

```
; "Next steps" series - camera in the browser with the FireBeetle 2 ESP32-S3 (DFR0975,
N16R8, OV2640)
; The board has 16 MB of flash and 8 MB of "octal" PSRAM: the board... lines replace the
default values (4 MB, no PSRAM).
[env:esp32s3]
platform = espressif32
board = dfrobot_firebeetle2_esp32s3
framework = arduino
monitor_speed = 115200
board_build.arduino.memory_type = qio_opi
board_upload.flash_size = 16MB
board_build.partitions = default_16MB.csv
build_flags =
    -DBOARD_HAS_PSRAM
    -DARDUINO_USB_CDC_ON_BOOT=1
```

In **Arduino IDE** choose the board "DFRobot Firebeetle 2 ESP32-S3" and, in the Tools menu: **USB CDC On Boot: Enabled, PSRAM: OPI PSRAM, Flash Size: 16MB, Partition Scheme: 16M Flash (3MB APP/9.9MB FATFS).**

3 The sketch

Write your network name and password. The camera pin numbers are the ones DFRobot and Espressif give for this board: do not touch them.

```
// Camera in the browser: the FireBeetle 2 ESP32-S3 takes a photo when the page asks for one.
// The page asks for one per second: almost a video. The LED on the board lights up with every
shot.
#include <WiFi.h>
#include <WebServer.h>
#include "esp_camera.h"
#include "img_converters.h"

const char* NETWORK = "your-network-name";
const char* PASSWORD = "your-network-password";
const int LED = 21;           // the LED on the board (D13)

WebServer server(80);
bool directJpeg = true;      // does the sensor already send JPEG? (see the guide)

camera_config_t cameraConfig(pixformat_t format) {
    camera_config_t c = {};
    // the camera connections on the FireBeetle 2 ESP32-S3 (DFRobot wiki and Espressif example)
    c.pin_pwdn = -1; c.pin_reset = -1; c.pin_xclk = 45;
    c.pin_sccb_sda = 1; c.pin_sccb_scl = 2;
```

```

c.pin_d7 = 48; c.pin_d6 = 46; c.pin_d5 = 8; c.pin_d4 = 7;
c.pin_d3 = 4; c.pin_d2 = 41; c.pin_d1 = 40; c.pin_d0 = 39;
c.pin_vsync = 6; c.pin_href = 42; c.pin_pclk = 5;
c.xclk_freq_hz = 200000000;
c.ledc_timer = LEDC_TIMER_0; c.ledc_channel = LEDC_CHANNEL_0;
c.pixel_format = format;
c.frame_size = FRAMESIZE_VGA; // 640 x 480: a good compromise
c.jpeg_quality = 12; // from 0 to 63: lower = nicer and heavier
c.fb_count = 2; // two images in memory: smoother
c.fb_location = CAMERA_FB_IN_PSRAM; // the images live in the PSRAM
c.grab_mode = CAMERA_GRAB_LATEST; // always the latest image, not an old one
return c;
}

void photo() {
    digitalWrite(LED, HIGH);
    camera_fb_t* fb = esp_camera_fb_get();
    if (!fb) {
        server.send(500, "text/plain", "Capture failed");
        digitalWrite(LED, LOW);
        return;
    }
    if (directJpeg) {
        server.send_P(200, "image/jpeg", (const char*)fb->buf, fb->len);
    } else { // uncompressed image: we convert it to JPEG
        uint8_t* jpg = nullptr;
        size_t len = 0;
        frame2jpg(fb, 80, &jpg, &len);
        server.send_P(200, "image/jpeg", (const char*)jpg, len);
        free(jpg);
    }
    esp_camera_fb_return(fb); // gives the memory back for the next shot
    digitalWrite(LED, LOW);
}

void page() {
    server.send(200, "text/html",
        "<!DOCTYPE html><html><head><meta name='viewport' content='width=device-width, initial-scale=1'><title>Camera</title></head>"
        "<body style='margin:0;background:#111;text-align:center'><img id='f' src='/photo' "
        "style='width:100%;max-width:640px'>"
        "<script>setInterval(())=>{document.getElementById('f').src='/photo?'+Date.now()},1000)"
        "</script></body></html>");
}

void setup() {
    Serial.begin(115200);
    pinMode(LED, OUTPUT);
    if (!psramFound()) Serial.println("Warning: PSRAM not active, check the settings");
    camera_config_t c = cameraConfig(PIXFORMAT_JPEG);
    if (esp_camera_init(&c) != ESP_OK) { // some versions of the board want the RGB565 format
        directJpeg = false;
        c = cameraConfig(PIXFORMAT_RGB565);
        c.frame_size = FRAMESIZE_QVGA;
        if (esp_camera_init(&c) != ESP_OK) {
            Serial.println("Camera not found: power the board off and on again");
            return;
        }
    }
}

```

```

    }
  }
  WiFi.mode(WIFI_STA);
  WiFi.begin(NETWORK, PASSWORD);
  while (WiFi.status() != WL_CONNECTED) delay(500);
  Serial.print("Open http://");
  Serial.println(WiFi.localIP());
  server.on("/", page);
  server.on("/photo", photo);
  server.begin();
}

void loop() {
  server.handleClient();
  delay(2);
}

```

New word: quality and memory

jpeg_quality goes from 0 to 63: the lower it is, the nicer and heavier the photo. fb_count = 2 keeps two images ready in the PSRAM and CAMERA_GRAB_LATEST always gives the latest one, not an old one.

4 Try it

✓ What you should see

"Open http://..." appears in the Serial Monitor. Open it from your phone: you see the camera image, updated every second. The board's LED flashes with every shot.

If it doesn't work

- "Camera not found": the flat cable is not pushed all the way in or is upside down; DFRobot also recommends powering off and on again.
- "PSRAM not active": the memory settings are not the ones from step 2.
- The upload does not start: hold down BOOT, press and release RST, then release BOOT, as DFRobot advises.

5 Two things to know

- **Board versions:** DFRobot notes that version 1.0 needs a library to switch on the camera's power supply, and for 1.3 it recommends the RGB565 format. The sketch tries JPEG first and, if that fails, switches to RGB565 by itself and converts the photos.
- **Real video:** for smooth streaming there is Espressif's **CameraWebServer** example (File → Examples → ESP32 → Camera), choosing the model CAMERA_MODEL_DFRobot_FireBeetle2_ESP32S3: the video is at :81/stream.
- **Privacy:** the page has no password. Keep it on your home network and do not point the camera where other people are without telling them.

6

Challenge yourself

Try it yourself

- Add the over-the-air update from guide no. 59: the mounted camera can be updated without taking it down.
- With the board's button (GPIO 47), save the best photo in a variable and show it at /latest.
- Raise the resolution to SVGA (800×600) and see how much it slows down.

✓ What you learned

- images need PSRAM and a sketch with plenty of room;
- the board answers /photo with a JPEG, and the page asks for it every second;
- camera pins and memory settings come from the manufacturer.

Sources: DFRobot, FireBeetle 2 ESP32-S3 wiki (DFR0975) and camera tutorial · Espressif, arduino-esp32: CameraWebServer, camera_pins.h, boards.txt · Espressif, esp32-camera (esp_camera.h) · PlatformIO, dfrobot_firebeetle2_esp32s3 board and board_... options · informaticait guides no. 56 and 59