Streaming FireBeetle ESP32 Camera Video to Home Assistant

Will guide you to stream ESP32 Cam footage to Home Assistant.

Difficulté **Facile** | Durée **2 heure(s)** | Catégories **Électronique** | Coût **20 USD ($)**

Sommaire

Introduction

Streaming ESP32-CAM Video to Home Assistant

The **ESP32-CAM** is a versatile and affordable camera module that can be used for various projects. In this tutorial, we'll explore how to set up video streaming from the ESP32-CAM to **Home Assistant**, allowing you to monitor your surroundings remotely.

Matériaux

Outils

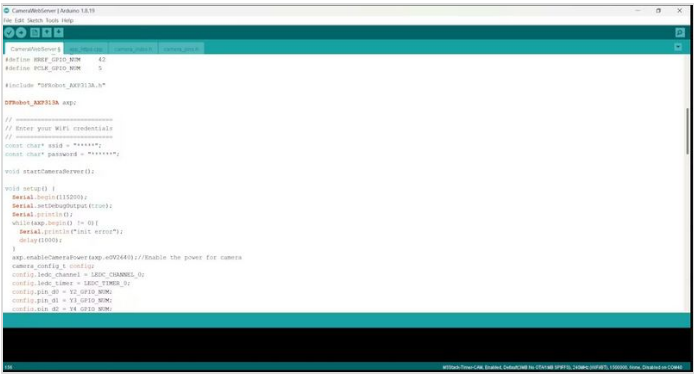
Étape 1 - Setting Up Video Streaming Web Server

Open the Arduino IDE and navigate to the ESP32 examples.
1. Select File->Examples->ESP32->Camera->CameraWebServer example in Arduino IDE.
2. Replace the codes in CameraWebServer with the code below (Note: please fill in your WiFi account and password)

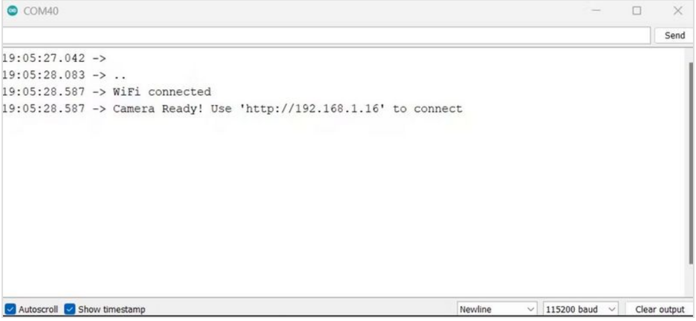
```
#include "esp_camera.h"
#include <WiFi.h>

//
// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//            Ensure ESP32 Wrover Module or other board with PSRAM is selected
//            Partial images will be transmitted if image exceeds buffer size
//
//            You must select partition scheme from the board menu that has at least 3MB APP space.
//            Face Recognition is DISABLED for ESP32 and ESP32-S2, because it takes up from 15
//            seconds to process single frame. Face Detection is ENABLED if PSRAM is enabled as well

// ===================
// Select camera model
// ===================
#define PWDN_GPIO_NUM     -1
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM     45
#define SIOD_GPIO_NUM     1
```

```
#define SIOC_GPIO_NUM     2

#define Y9_GPIO_NUM       48
#define Y8_GPIO_NUM       46
#define Y7_GPIO_NUM       8
#define Y6_GPIO_NUM       7
#define Y5_GPIO_NUM       4
#define Y4_GPIO_NUM       41
#define Y3_GPIO_NUM       40
#define Y2_GPIO_NUM       39
#define VSYNC_GPIO_NUM    6
#define HREF_GPIO_NUM     42
#define PCLK_GPIO_NUM     5

#include "DFRobot_AXP313A.h"

DFRobot_AXP313A axp;

// ===========================
// Enter your WiFi credentials
// ===========================
const char* ssid = "*****";
const char* password = "******";

void startCameraServer();

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();
  while(axp.begin() != 0){
    Serial.println("init error");
    delay(1000);
  }
  axp.enableCameraPower(axp.eOV2640);//Enable the power for camera
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.frame_size = FRAMESIZE_UXGA;
  config.pixel_format = PIXFORMAT_JPEG; // for streaming
  //config.pixel_format = PIXFORMAT_RGB565; // for face detection/recognition
  config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
  config.fb_location = CAMERA_FB_IN_PSRAM;
  config.jpeg_quality = 12;
  config.fb_count = 1;

  // if PSRAM IC present, init with UXGA resolution and higher JPEG quality
  //                      for larger pre-allocated frame buffer.
  if(config.pixel_format == PIXFORMAT_JPEG){
    if(psramFound()){
      config.jpeg_quality = 10;
      config.fb_count = 2;
      config.grab_mode = CAMERA_GRAB_LATEST;
    } else {
      // Limit the frame size when PSRAM is not available
      config.frame_size = FRAMESIZE_SVGA;
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }

#if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
#endif

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

  sensor_t * s = esp_camera_sensor_get();
  // initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1); // flip it back
    s->set_brightness(s, 1); // up the brightness just a bit
    s->set_saturation(s, -2); // lower the saturation
  }
  // drop down frame size for higher initial frame rate
  if(config.pixel_format == PIXFORMAT_JPEG){
    s->set_framesize(s, FRAMESIZE_QVGA);
  }

#if defined(CAMERA_MODEL_M5STACK_WIDE) || defined(CAMERA_MODEL_M5STACK_ESP32CAM)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif

#if defined(CAMERA_MODEL_ESP32S3_EYE)
  s->set_vflip(s, 1);
#endif

  WiFi.begin(ssid, password);
  WiFi.setSleep(false);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
}

void loop() {
  // Do nothing. Everything is done in another task by the web server
  delay(10000);
}
```

Then upload the code to the FireBeetle ESP32 S3 board and look for the serial terminal response.



```
#define HREF_GPIO_NUM     42
#define PCLK_GPIO_NUM     5

#include "DFRobot_AXP313A.h"

DFRobot_AXP313A axp;

// ===========================
// Enter your WiFi credentials
// ===========================
const char* ssid = "******";
const char* password = "*******";

void startCameraServer();

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();
  while(axp.begin() != 0){
    Serial.println("init error");
    delay(1000);
  }
  axp.enableCameraPower(axp.eOV2640);//Enable the power for camera
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
```



```
19:05:27.042 ->
19:05:28.083 -> ..
19:05:28.587 -> WiFi connected
19:05:28.587 -> Camera Ready! Use 'http://192.168.1.16' to connect
```

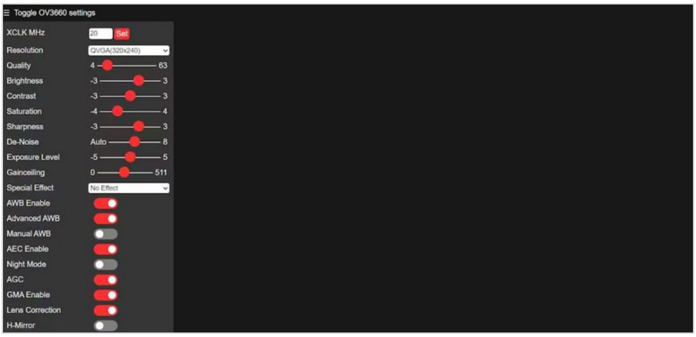
Étape 2 - Get PCBs for Your Projects Manufactured

PCBWay 6th Project Design Contest
Electronic & Mechanical Design
PARTICIPATE NOW
PCBWay
Instant Quote
PCBWay 6th Project Design Contest
Electronic & Mechanical Design
$5.00
$30

You must check out PCBWAY for ordering PCBs online for cheap! You get 10 good-quality PCBs manufactured and shipped to your doorstep for cheap. You will also get a discount on shipping on your first order. Upload your Gerber files onto PCBWAY to get them manufactured with good quality and quick turnaround time. PCBWay now could provide a complete product solution, from design to enclosure production. Check out their online Gerber viewer function. With reward points, you can get free stuff from their gift shop. Also, check out this useful blog on PCBWay Plugin for KiCad from here. Using this plugin, you can directly order PCBs in just one click after completing your design in KiCad.

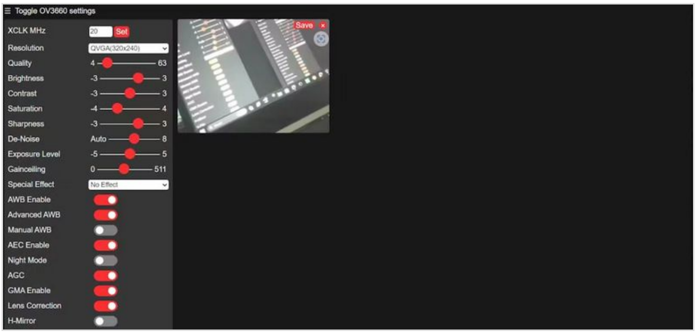
Étape 3 - Setting Up Video Stream Properties

Next, open up the IP address that shows in the serial terminal and look for the camera web server properties.
Set up the resolution. Go with the lower one to get a high FPS.
Finally, add the:81/stream in the camera IP address.
Now it will directly show you the camera feed.



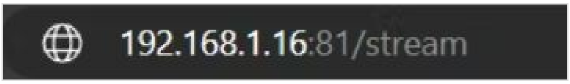
Toggle OV3660 settings

XCLK MHz	20 Set
Resolution	QVGA(320x240)
Quality	4 – 63
Brightness	-3 – 3
Contrast	-3 – 3
Saturation	-4 – 4
Sharpness	-3 – 3
De-Noise	Auto – 8
Exposure Level	-5 – 5
Gainceiling	0 – 511
Special Effect	No Effect
AWB Enable	●
Advanced AWB	●
Manual AWB	○
AEC Enable	●
Night Mode	○
AGC	●
GMA Enable	●
Lens Correction	●
H-Mirror	○



Toggle OV3660 settings

XCLK MHz	20 Set
Resolution	QVGA(320x240)
Quality	4 – 63
Brightness	-3 – 3
Contrast	-3 – 3
Saturation	-4 – 4
Sharpness	-3 – 3
De-Noise	Auto – 8
Exposure Level	-5 – 5
Gainceiling	0 – 511
Special Effect	No Effect
AWB Enable	●
Advanced AWB	●
Manual AWB	○
AEC Enable	●
Night Mode	○
AGC	●
GMA Enable	●
Lens Correction	●
H-Mirror	○

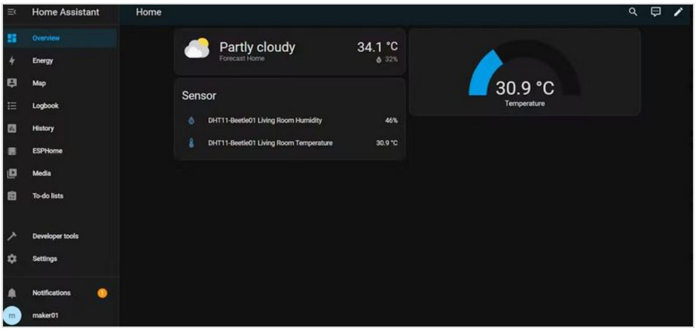


192.168.1.16:81/stream

Étape 4 - Integrating with Home Assistant

- Open **Home Assistant**.
- Next, navigate to the overview and add a picture card.
- Enter the IP address of your ESP32-CAM following:81/stream and click **Finish**.
- And that's it! You've successfully set up video streaming from the ESP32-CAM to Home Assistant.

Feel free to customize and enhance this project further. Happy monitoring! ??

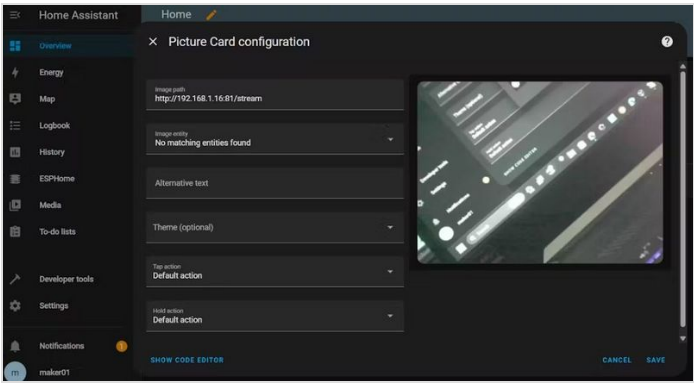


Partly cloudy 34.1 °C
Forecast Home 32%

Sensor

DHT11-Beetle01 Living Room Humidity	46%
DHT11-Beetle01 Living Room Temperature	30.9 °C

30.9 °C
Temperature



Picture Card configuration

Image path: http://192.168.1.16:81/stream

Image entity: No matching entities found

Alternative text

Theme (optional)

Tap action: Default action

Hold action: Default action