

DIY surveillance camera with ESP32 camera

In this "DIY Surveillance Camera with ESP32-Camera" project, we will use the ESP32-Camera module to make a DIY surveillance camera to monitor the surrounding environment. The ESP32 camera module will be hosted on a web server, where we can watch the surveillance live, which will help us understand who is in the surveillance area .

Required components:

ESP32-CAM (with OV2640 camera)

FTDI programmer /expansion board with type-c port

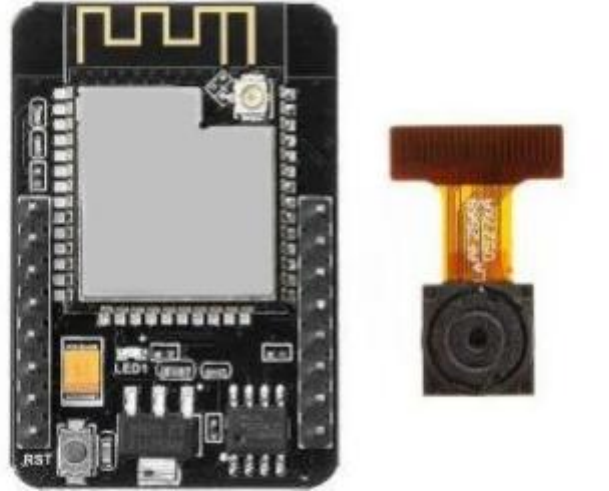
connecting wires

ESP32-CAM 5V power supply

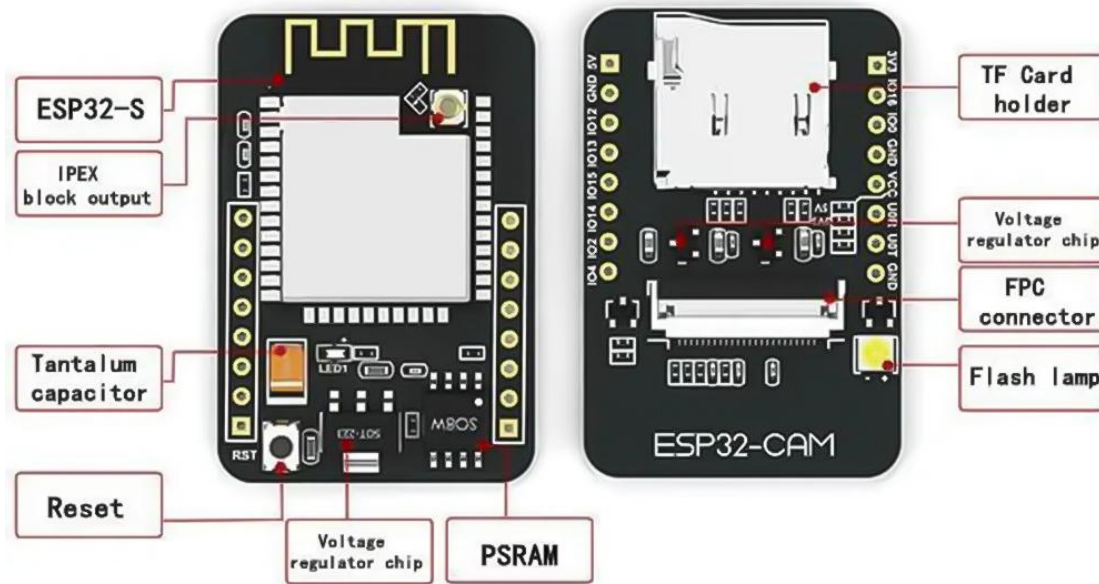
Introduction to ESP32 - Cam:

The ESP32-CAM is a very small camera module with an ESP32-S chip. In addition to the OV2640 camera and several GPIOs for connecting peripherals, it has a microSD card slot that can be used to store images taken with the camera or to

store files to be served to clients.



The components of ESP32-Cam:



ESP32camera features:

Here is a list of features of the ESP32-Cam :

Features the smallest 802.11b / g/n Wi-Fi BT SoC module

Low-power 32-bit CPU, can also be an application processor

The clock speed is 160MHz and the total computing power is up to 600 DMIPS

Built-in 520KB SRAM, external 4MPSRAM

Support UART/SPI/I2C/PWM/ADC/DAC

Supports OV2640 and OV7670 cameras with built-in flash memory

Support WiFi to upload images

Support TF card

Supports multiple sleep modes

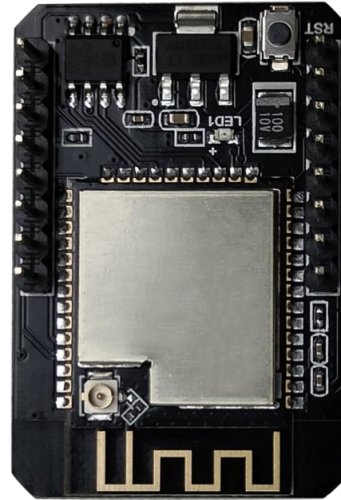
Embedded Lwip and FreeRTOS

Support STA/AP/STA+AP operation mode

Support smart configuration/AirKiss technology

Support serial port local and remote firmware upgrade (FOTA)

ESP32-CamPinout(AI-Thinker module):



There are three GND pins and two power pins: 3.3V or 5V. GPIO 1 and GPIO 3 are serial pins. You need these pins to upload code to the board. Also, GPIO 0 plays an important role as it determines whether the ESP32 is in blink mode or not. When GPIO 0 is connected to GND, the ESP32 is in blink mode.

Video Streaming Server Configuration:

Follow the steps below to build a video streaming web server using ESP32-Cam that you can access on your local network.

Install the ESP32 core files in the Arduino IDE :

In this example, we use the Arduino IDE to program the ESP32-Cam board. So you need to install ArduinoIDE and ESP32 add-ons. If you don't already have the ESP32 plugin installed in your computer, follow this tutorial to install it.

Here we will download the core files for the ESP32 board, you need internet to perform this step

Copy this link : https://dl.espressif.com/dl/package_esp32_index.json

Open the Arduino IDE and click File > Preferences

File Edit Sketch Tools Help

New Ctrl+N

Open... Ctrl+O

Open Recent > SP32-Camera

Sketchbook >

Examples >

Close Ctrl+W

Save Ctrl+S

Save As... Ctrl+Shift+S

Page Setup Ctrl+Shift+P

Print Ctrl+P

Preferences Ctrl+Comma

Quit Ctrl+Q

le brownout problems

" //disable brownout problems

credentials

an198910";

57890000000000000987654321"

Paste the provided link as shown. If you already have some links, separate them with commas. Press "OK".

Preferences ×

Settings Network

Sketchbook location:
D:\我的文档\Documents\Arduino Browse

Editor language: English (English) ⌵ (requires restart of Arduino)

Editor font size: 12

Interface scale: ☒ Automatic 100% ⌵ (requires restart of Arduino)

Theme: Default theme ⌵ (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload


Compiler warnings: None ⌵

☐ Display line numbers ☐ Enable Code Folding

☒ Verify code after upload ☐ Use external editor

☒ Check for updates on startup ☒ Save when verifying or uploading

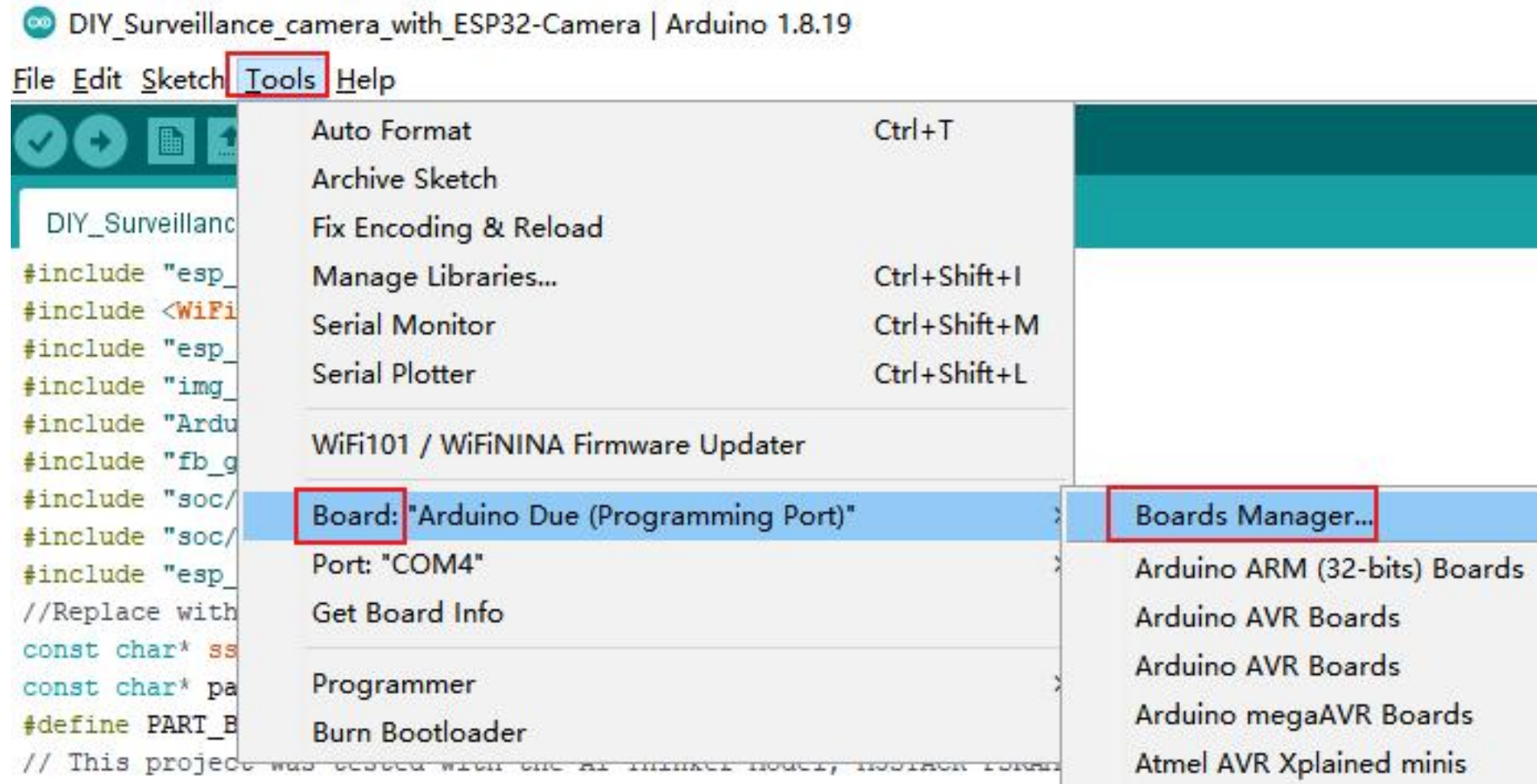
☐ Use accessibility features

Additional Boards Manager URLs: package_esp8266com_index.json, https://dl.espressif.com/dl/package_esp32_index.json 

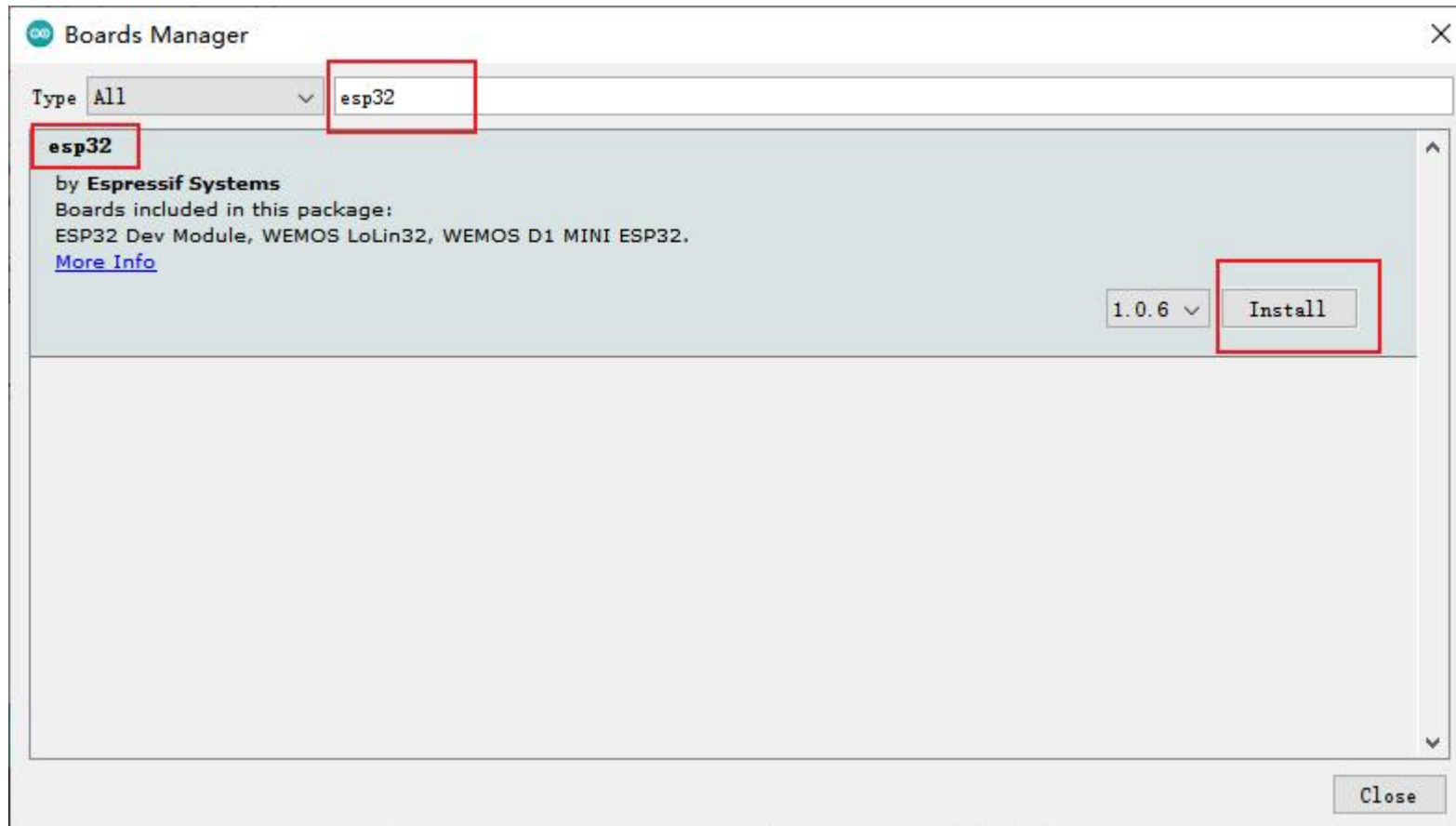
More preferences can be edited directly in the file
C:\Users\Administrator\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

OK Cancel

As shown below, click Tools > Board > Board Manager



esp 32" in the search bar



Select the latest version and click Install. The download will take some time, depending on your internet speed.

Analysis code: DIY surveillance camera with ESP32 camera

Note: Before uploading the code, you need to make some modifications. Insert the network credentials into the following variables and modify them to your own WiFi name and password:

```
//Replace with your network credentials
const char * ssid = "XXX" ; //"XXX" is the name of your WiFi
const char * password = "XXXXXX" ; //Your WiFi password
#define PART_BOUNDARY "123456789000000000000987654321"
```

Then, make sure to select the correct camera module. Our correct choice is the AI-THINKER model.

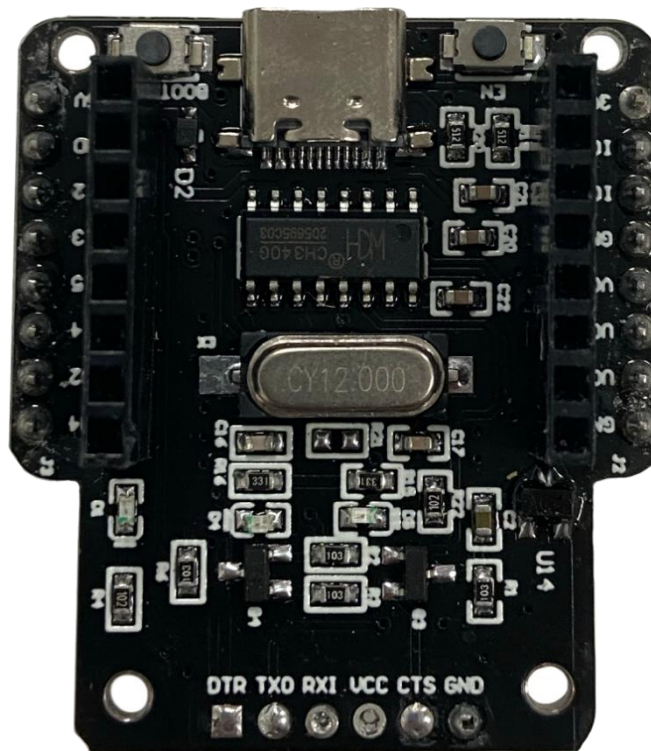
Therefore, please comment all other models and uncomment the appropriate models as follows:

```
// This project was tested with the AI Thinker Model, M5STACK PSRAM Model and M5STACK WITHOUT PSRAM
#define CAMERA_MODEL_AI_THINKER
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WITHOUT_PSRAM
```

Now, the code is ready to be uploaded to the ESP32-Cam module.

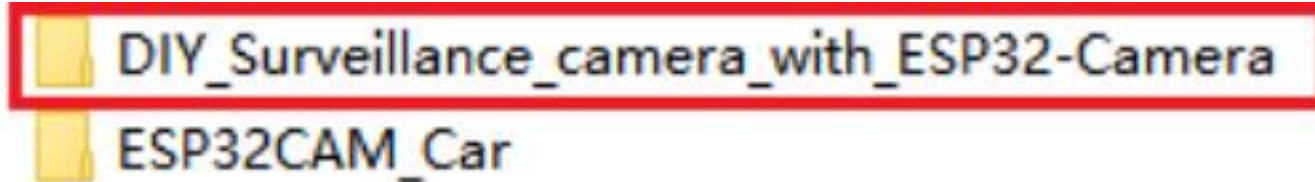
ESP32-CAM code upload:

Connect the ESP32-CAM development board to the computer through the type-C port of the expansion board . The expansion board is shown in the following figure:

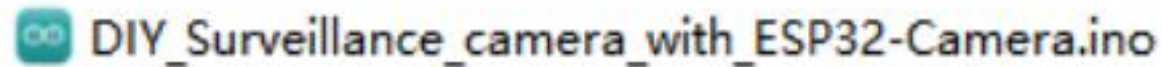


Upload code :

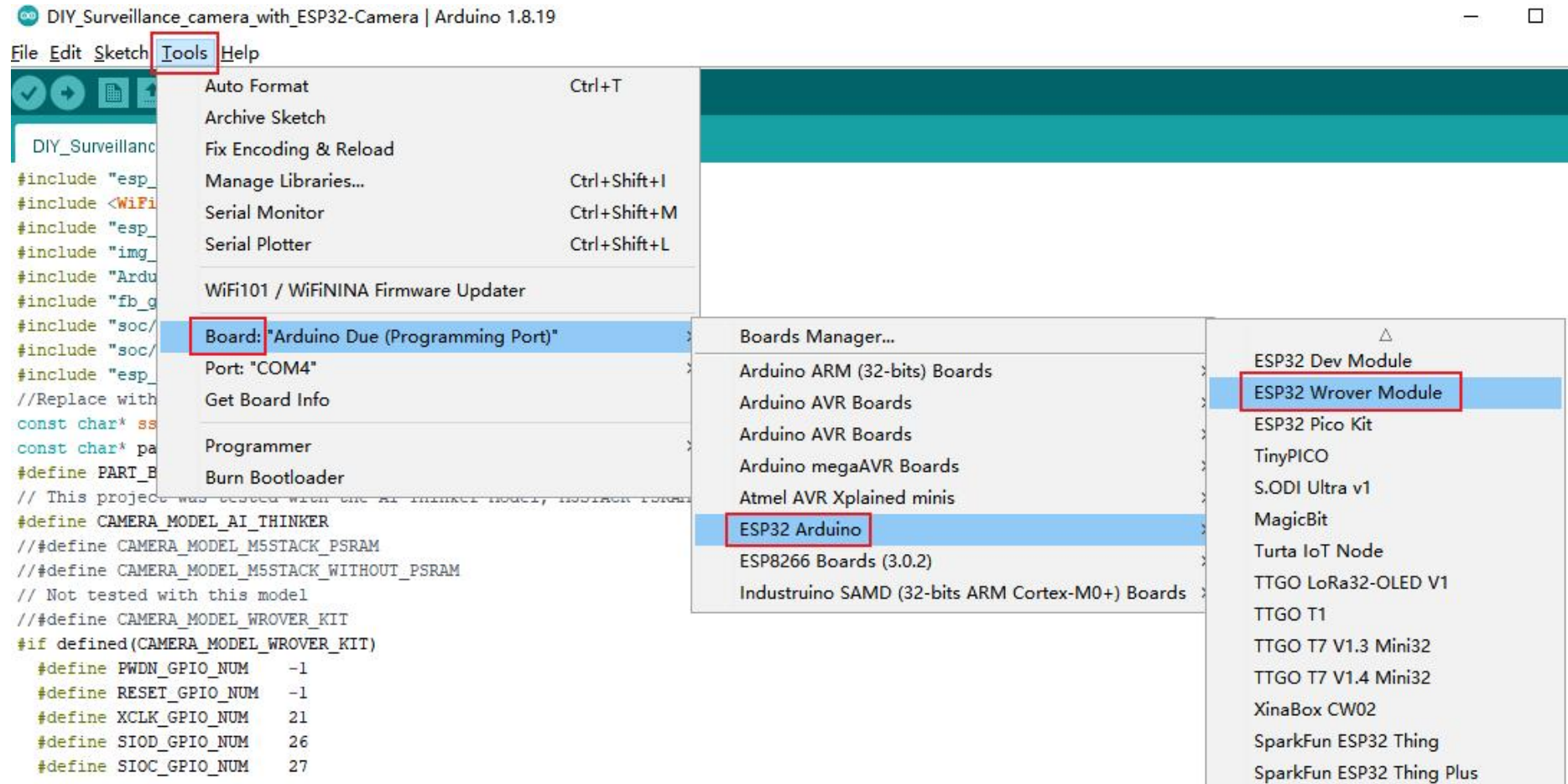
Open folder path 4. Tutorial -Arduino\4-Arduino Code\DIY_Surveillance_camera_with_ESP32-Camera



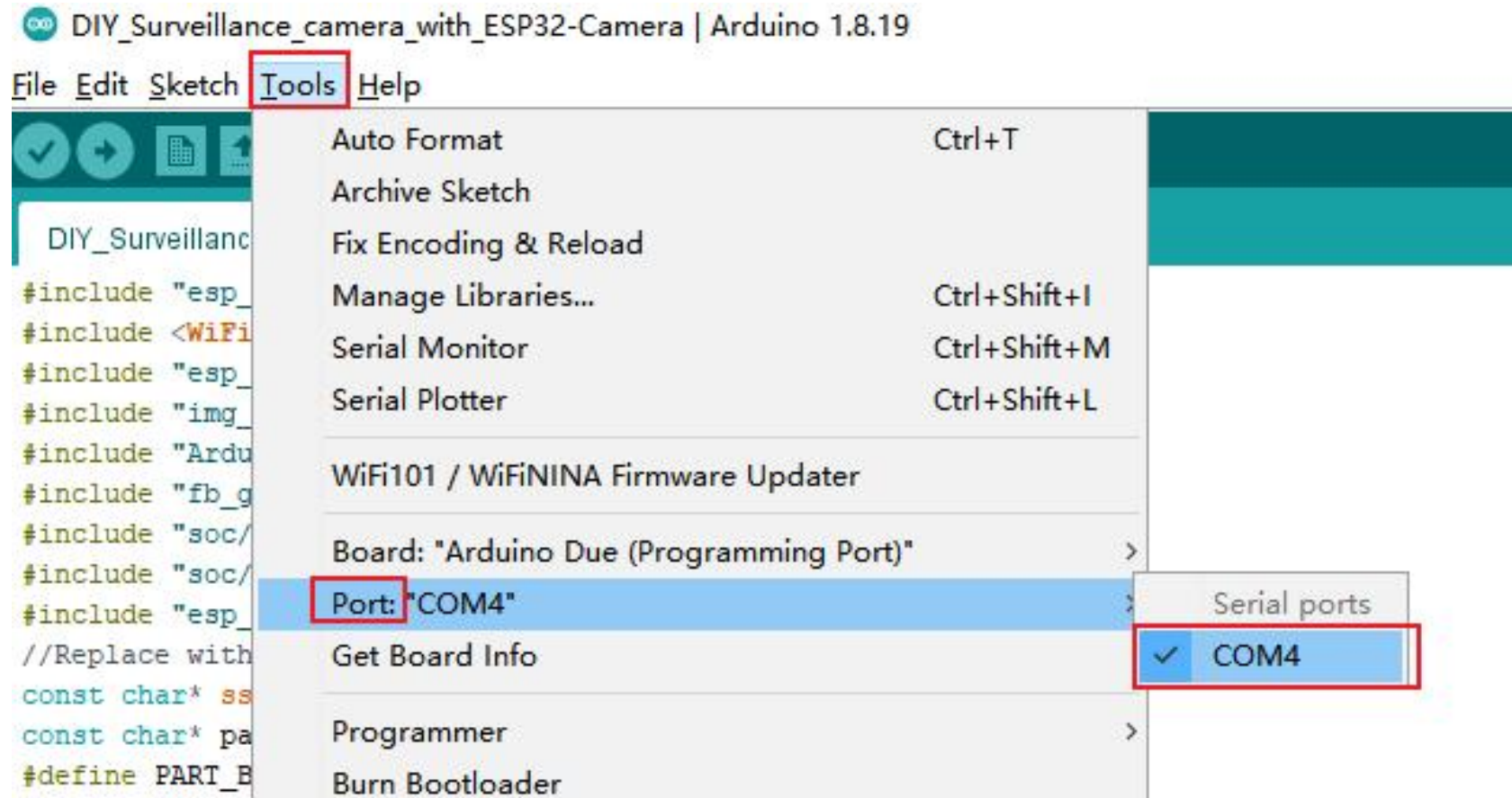
Open the .ino program in the Arduino IDE



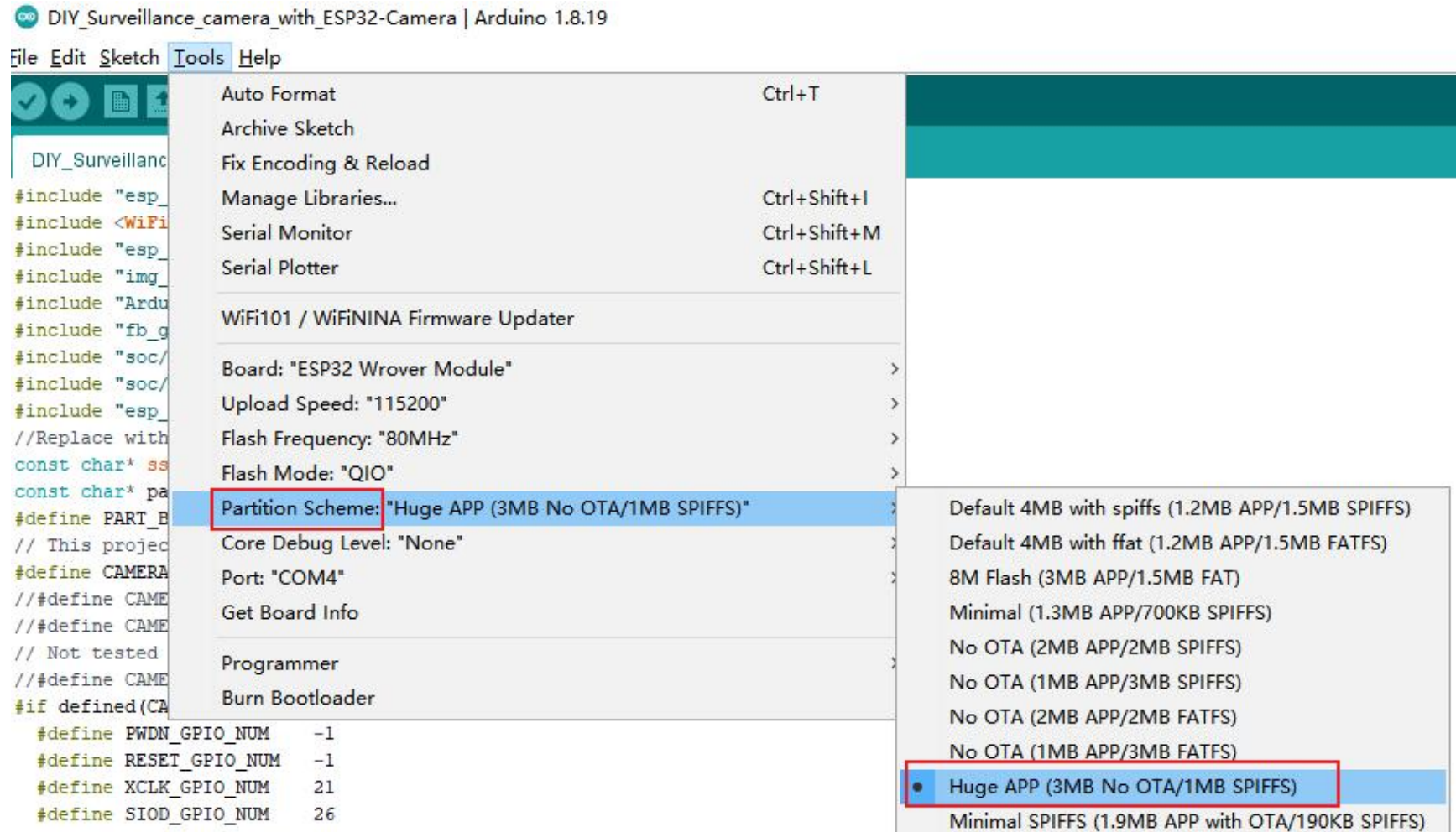
Connect ESP32-CAM to computer, go to Tools > Board > ESP32 Arduino in Arduino IDE , select ESP32 Wrover Module



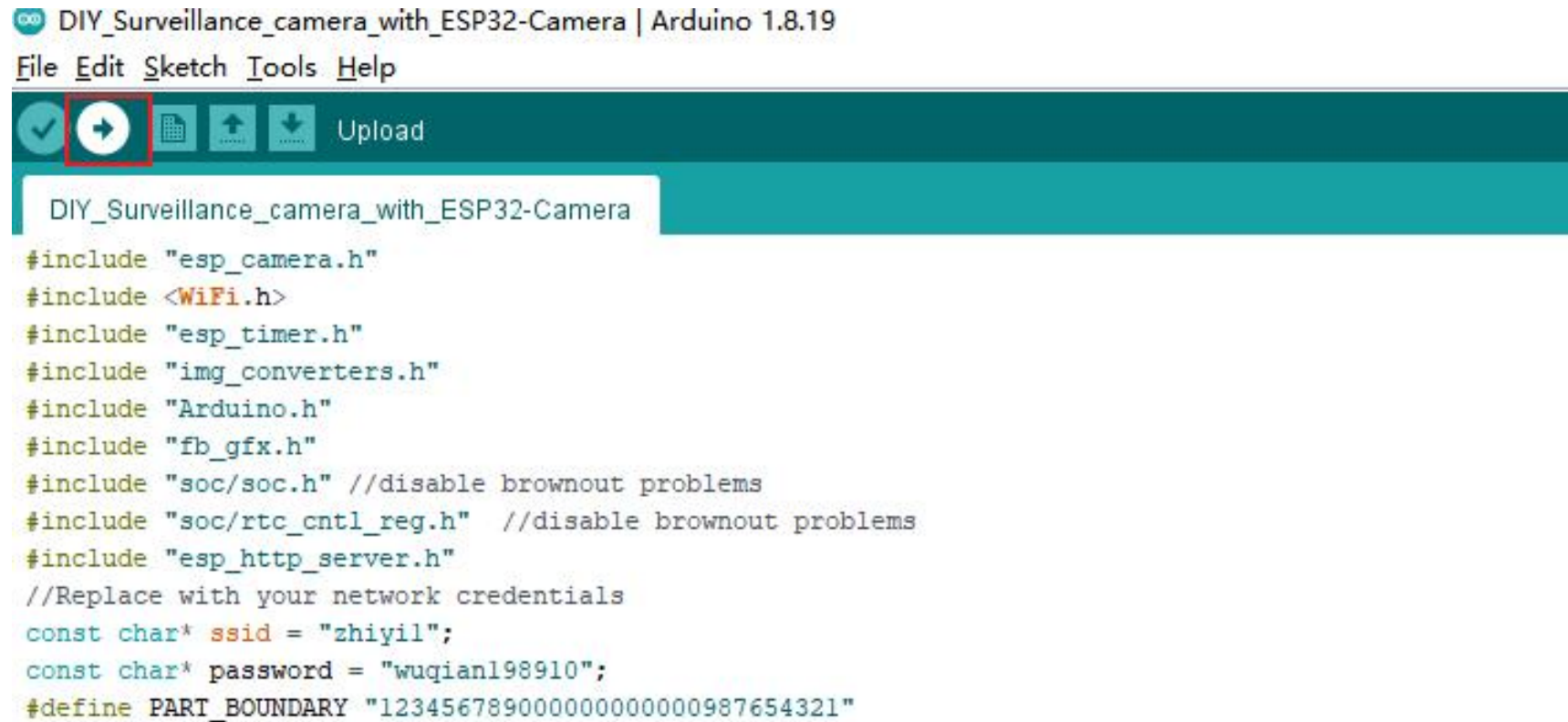
Go to Tools > Port and select the COM port to which the ESP32 is connected (the COM number that everyone can choose is not necessarily the same, you may be COM3 or others)



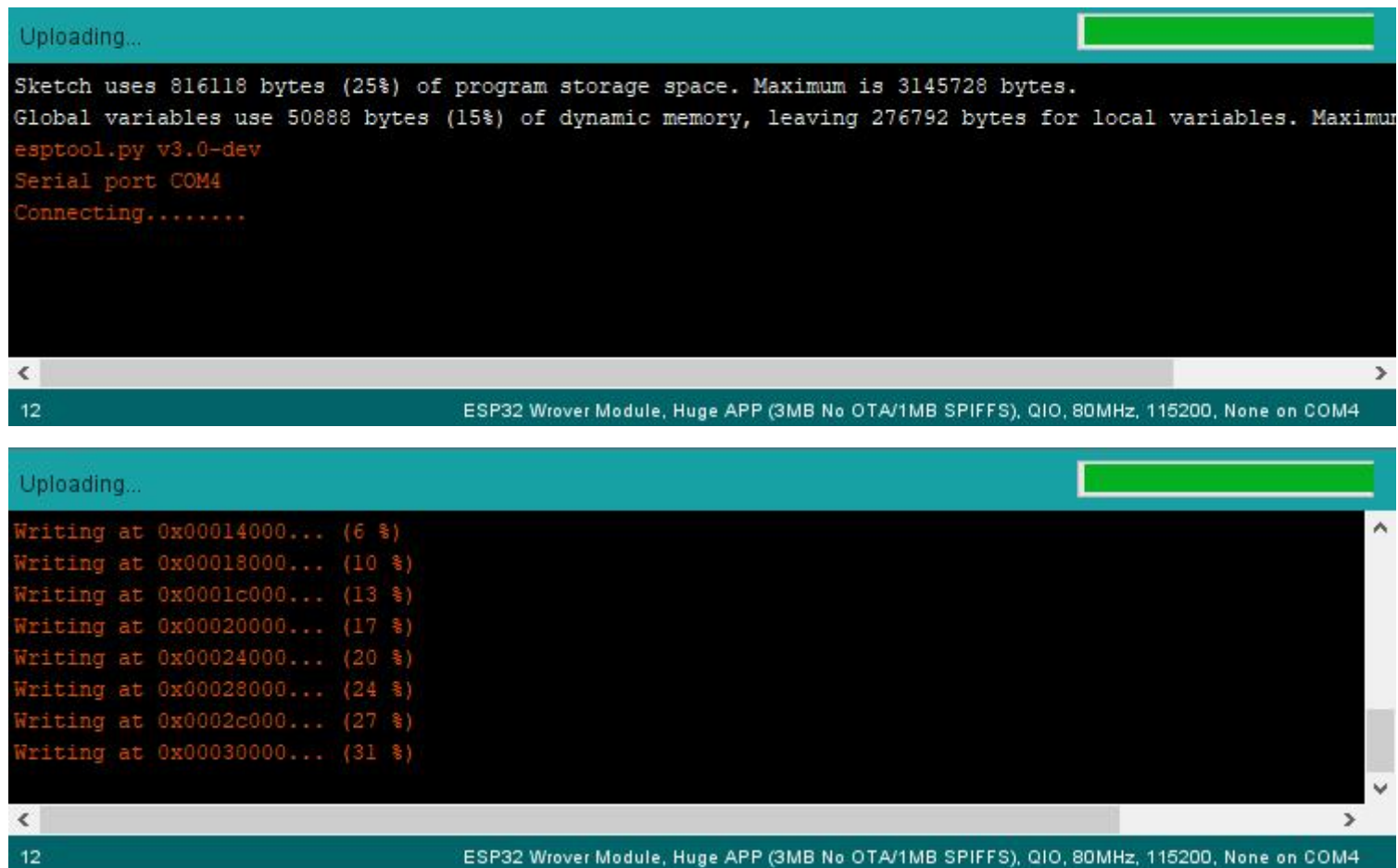
In Tools > Partition Scheme , select " Huge APP (3MB or OTA)"



Click the "Upload" button to burn the code



You can see the code burning situation at the bottom of the window



```
Uploading...  
Sketch uses 816118 bytes (25%) of program storage space. Maximum is 3145728 bytes.  
Global variables use 50888 bytes (15%) of dynamic memory, leaving 276792 bytes for local variables. Maximum  
esptool.py v3.0-dev  
Serial port COM4  
Connecting.....  
  
12 ESP32 Wrover Module, Huge APP (3MB No OTA/1MB SPIFFS), QIO, 80MHz, 115200, None on COM4
```

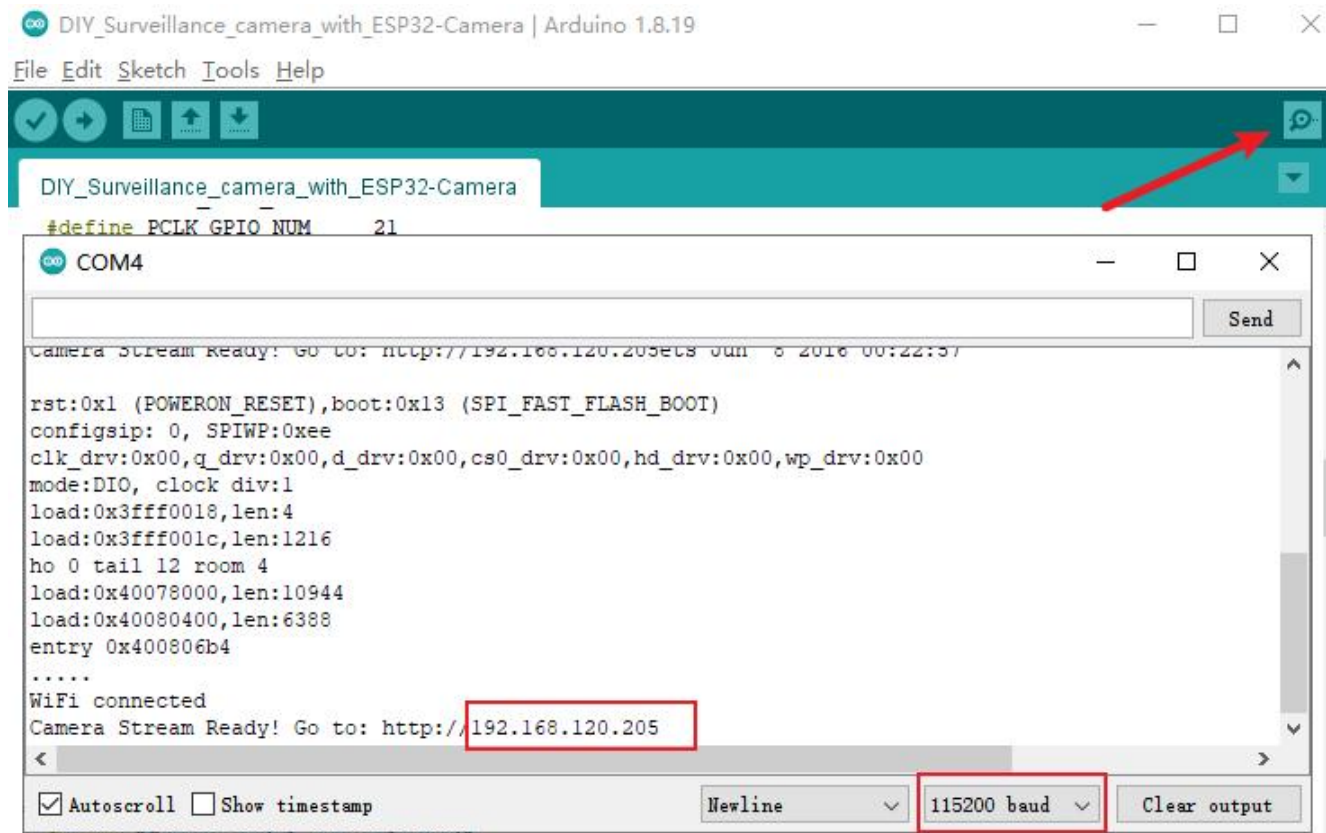
```
Uploading...  
Writing at 0x00014000... (6 %)  
Writing at 0x00018000... (10 %)  
Writing at 0x0001c000... (13 %)  
Writing at 0x00020000... (17 %)  
Writing at 0x00024000... (20 %)  
Writing at 0x00028000... (24 %)  
Writing at 0x0002c000... (27 %)  
Writing at 0x00030000... (31 %)
```

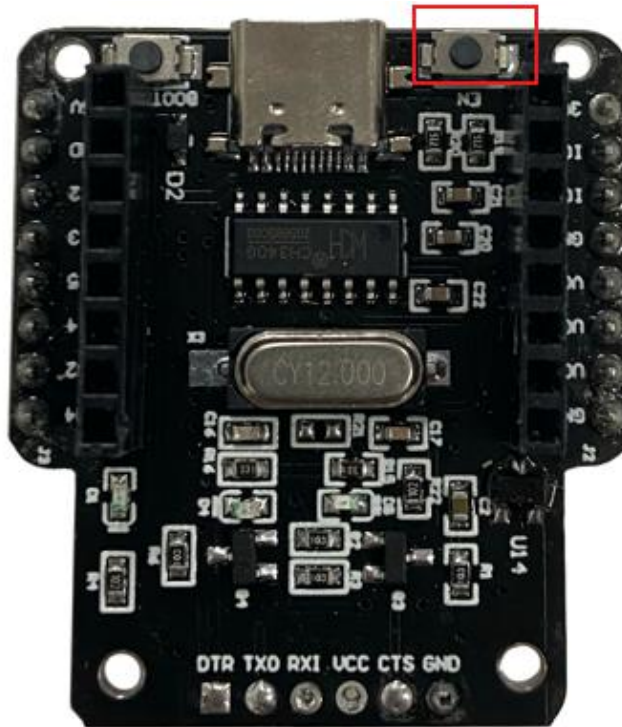
12 ESP32 Wrover Module, Huge APP (3MB No OTA/1MB SPIFFS), QIO, 80MHz, 115200, None on COM4

After a few seconds, the code should be successfully uploaded to your board.

Once the code is successfully uploaded, the ESP32-CAM will reset.

upper right serial monitor at a baud rate of 115200 . Press the reset button on the ESP32-CAM circuit board . When the ESP32-CAM is successfully connected to the WiFi, you can see the http address. Everyone gets the same address. Here it is "192.168.120.205", please remember it.





Videostreaming server:

Now you can access the camera streaming server on your local network. Connect the device to your own WiFi (the same local area network) as described above, open your browser, enter the IP address "192.168.120.205" of the ESP32-CAM, and then the motor determines to load the video streaming page.