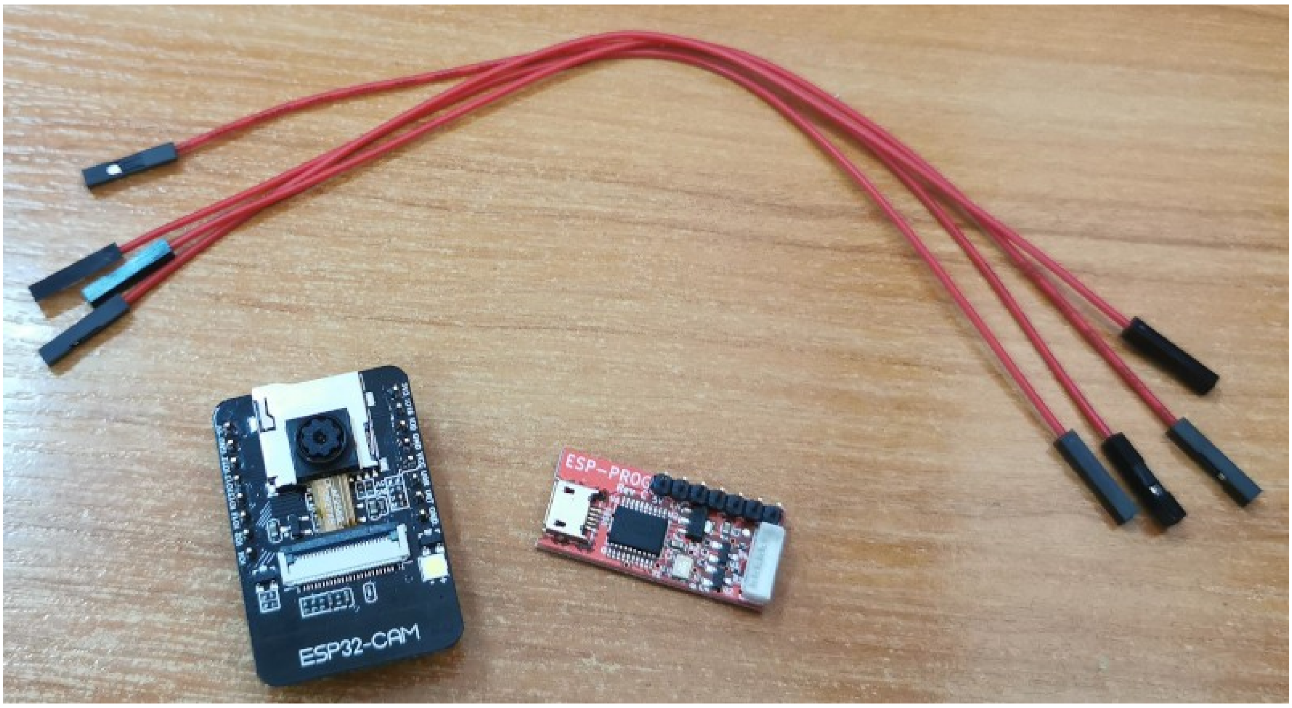
Using ESP32-CAM with Arduino IDE Tutorial

What you need: ESP32-CAM, ESP-PROG-C, USB-CABLE-A-MICRO-1.8M



1. Install Arduino

Go to arduino.cc and download latest revision for your OS.

2. CH340 patch for Linux

CH340 Linux drivers has bad PLL settings for all kernels before 5.5.

If your system happen to be with Linux Kernel before 5.5. here is the GitHub repository with the patch to install.

If you do not have this patch CH340 will work, but will not be able to communicate at speed over 115200 bps, with the patch up to 2Mbps communication is possible.

3. Wire cables:

You need to connect ESP32-CAM and ESP-PROG this way:

ESP32-CAM **GND** ----> ESP-PROG **GND**

ESP32-CAM **3.3V** ----> ESP-PROG **3.3V**

ESP32-CAM **U0T** ----> ESP-PROG **RXD**

ESP32-CAM **U0R** ----> ESP-PROG **TXD**

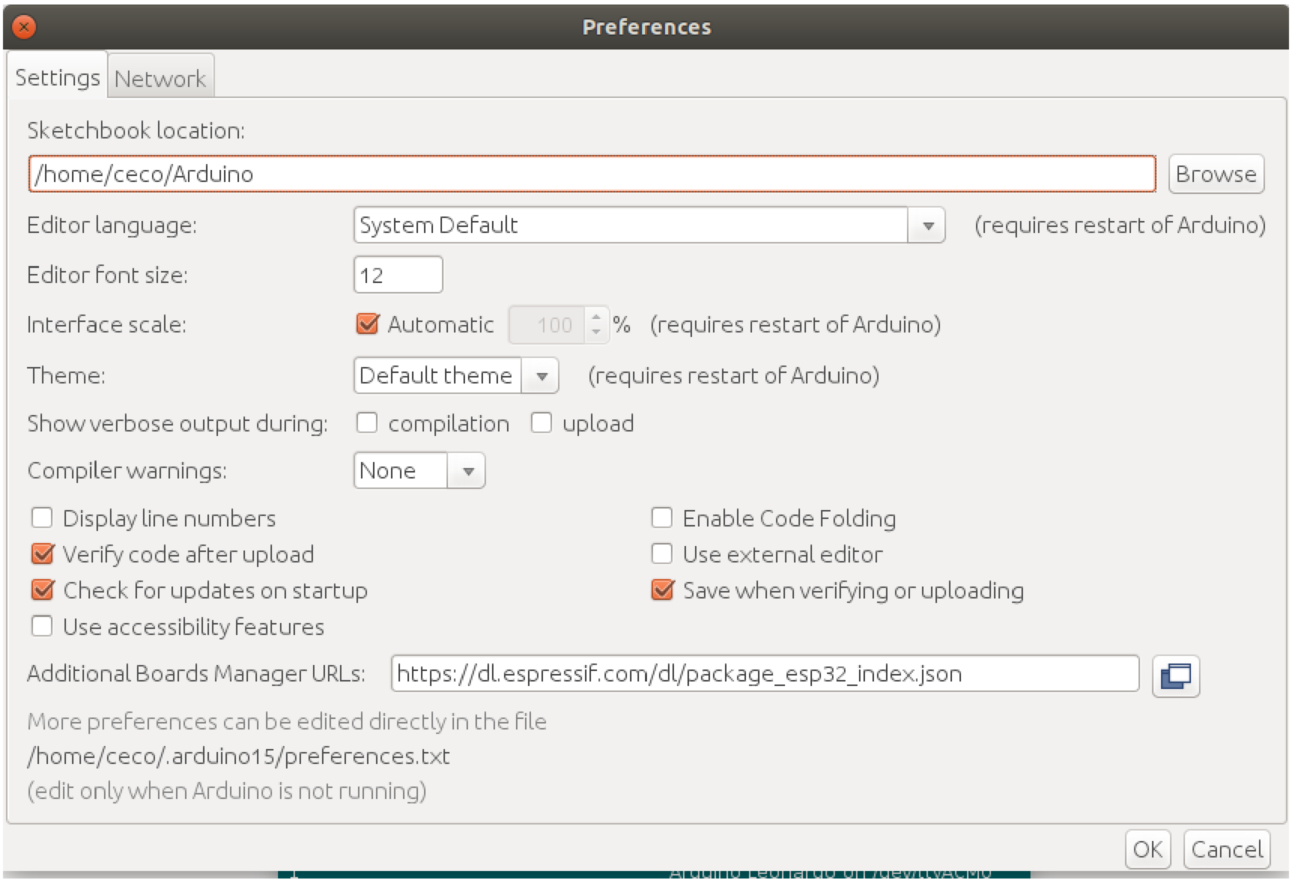
For firmware uploading you need one more connection, which is necessary ESP32 to go in Bootloader mode:

ESP32-CAM **IO0** ----> ESP32-CAM **GND**

4. Configure Arduino for ESP32-CAM

Run Arduino. In the Files-Preferences add:

https://dl.espressif.com/dl/package_esp32_index.json



Sketchbook location:
/home/ceco/Arduino

Editor language: System Default (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

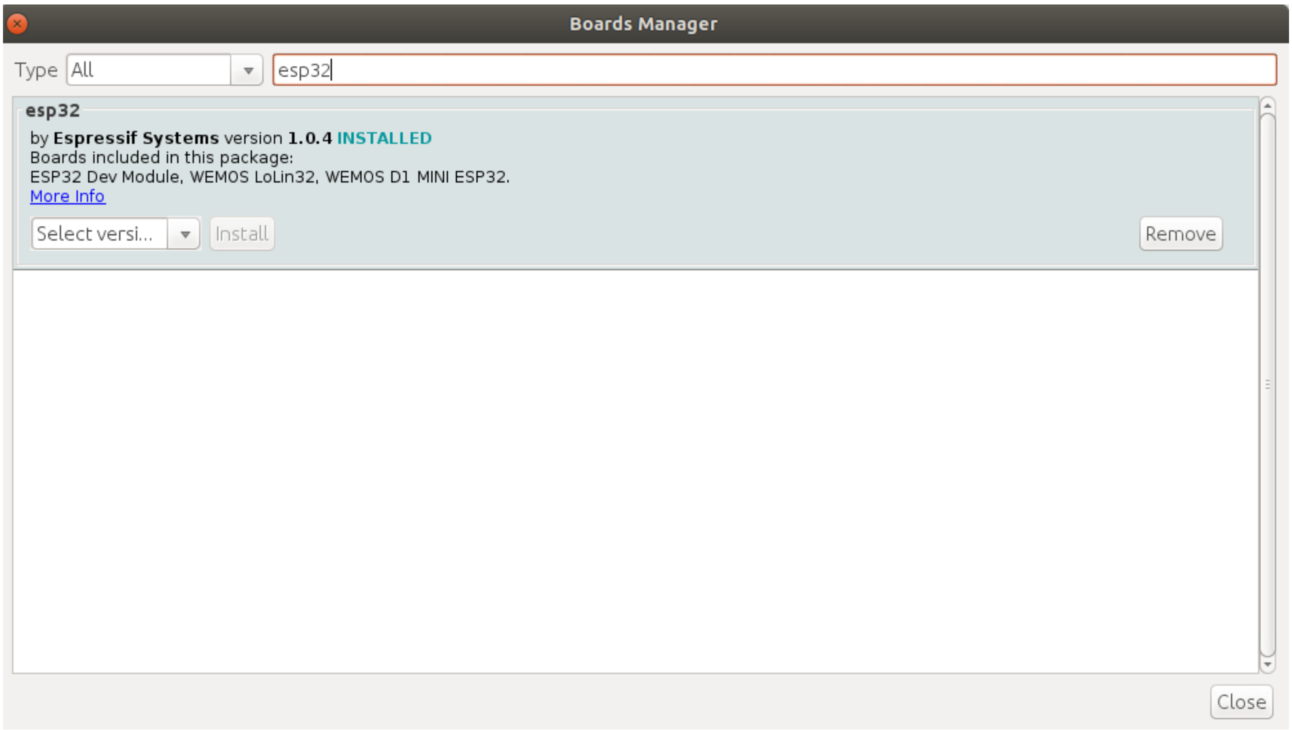
Compiler warnings: None

□ Display line numbers
☑ Verify code after upload
☑ Check for updates on startup
□ Use accessibility features
□ Enable Code Folding
□ Use external editor
☑ Save when verifying or uploading

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

More preferences can be edited directly in the file
/home/ceco/.arduino15/preferences.txt
(edit only when Arduino is not running)

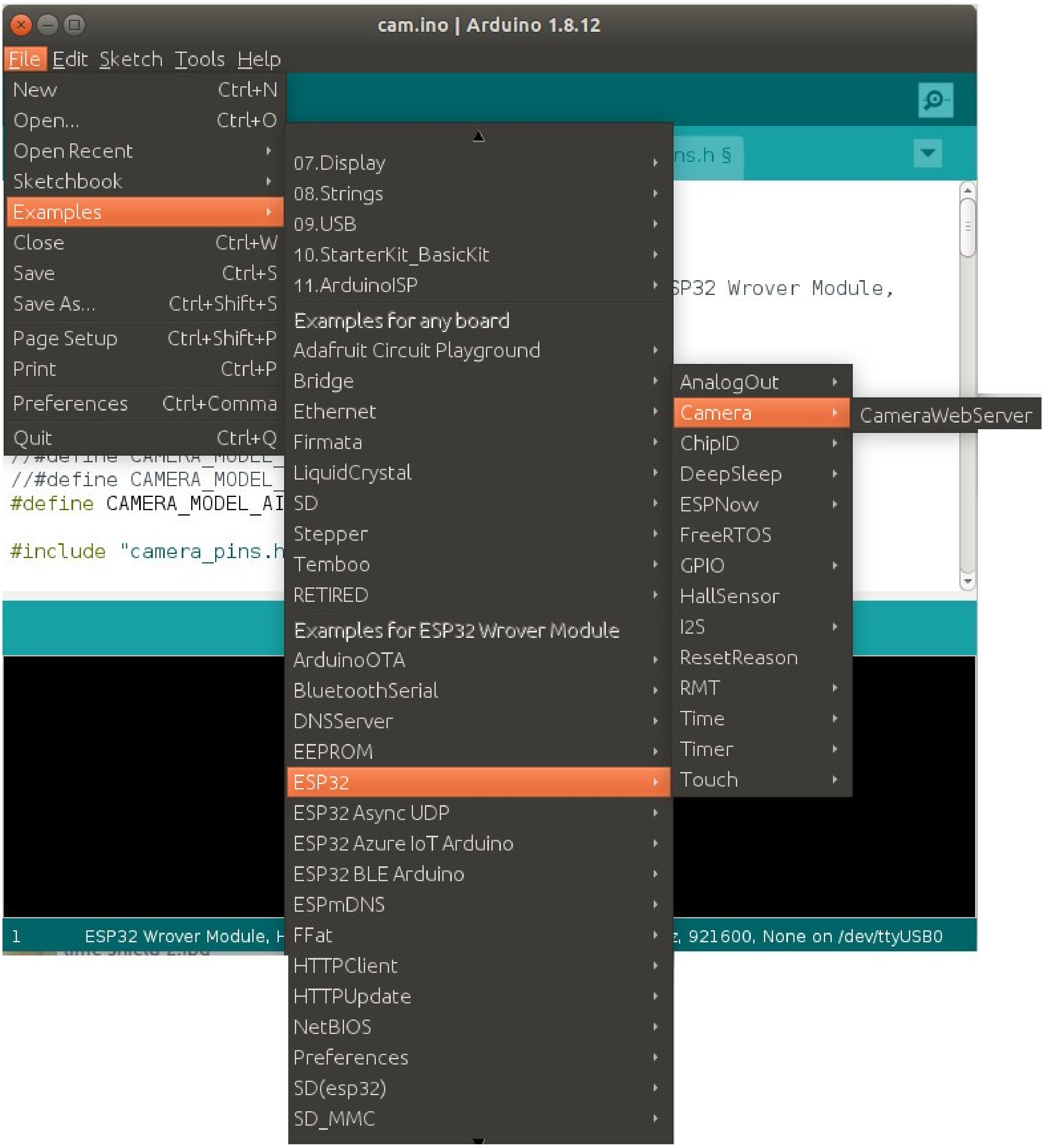
In Tools-Board-Board Manager search for ESP32 and install it



esp32

by **Espressif Systems** version **1.0.4** **INSTALLED**
Boards included in this package:
ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32.
More Info

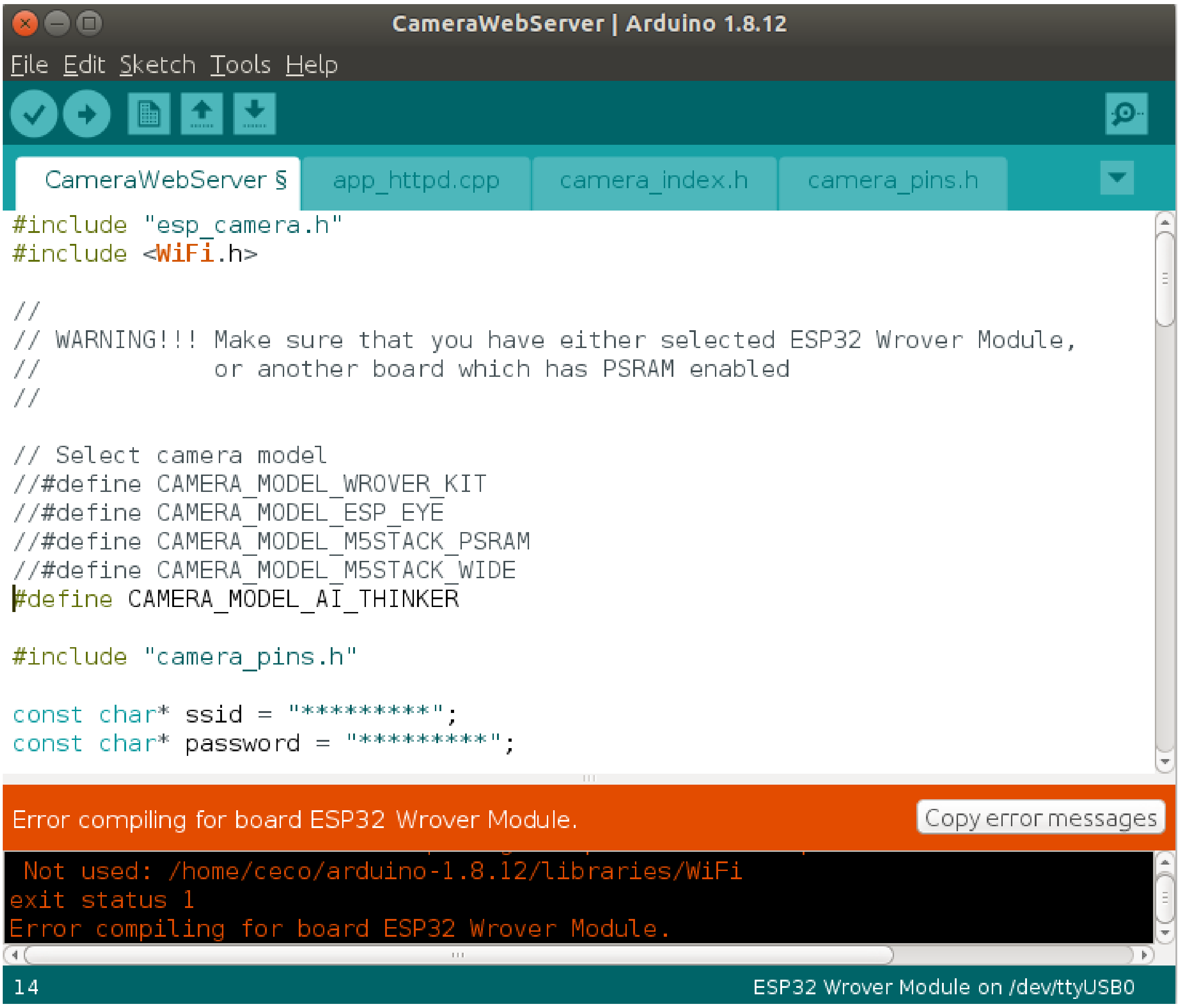
Exit Arduino and Run it again so it loads new board data. From File-Examples select ESP32-Camera-CameraWebServer



```
//#define CAMERA_MODEL_
//#define CAMERA_MODEL_
#define CAMERA_MODEL_AI

#include "camera_pins.h
```

1 ESP32 Wrover Module, H z, 921600, None on /dev/ttyUSB0

In select camera model uncomment CAMERA_MODEL_AI_THINKER and comment all other:



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

//
// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,
//            or another board which has PSRAM enabled
//

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"

const char* ssid = "*********";
const char* password = "*********";
```

Error compiling for board ESP32 Wrover Module.

```
 Not used: /home/ceco/arduino-1.8.12/libraries/WiFi
exit status 1
Error compiling for board ESP32 Wrover Module.
```

Then enter SSID and PASSWORD for your WiFi router.

In Tools-Board select : ESP32Wrover Module

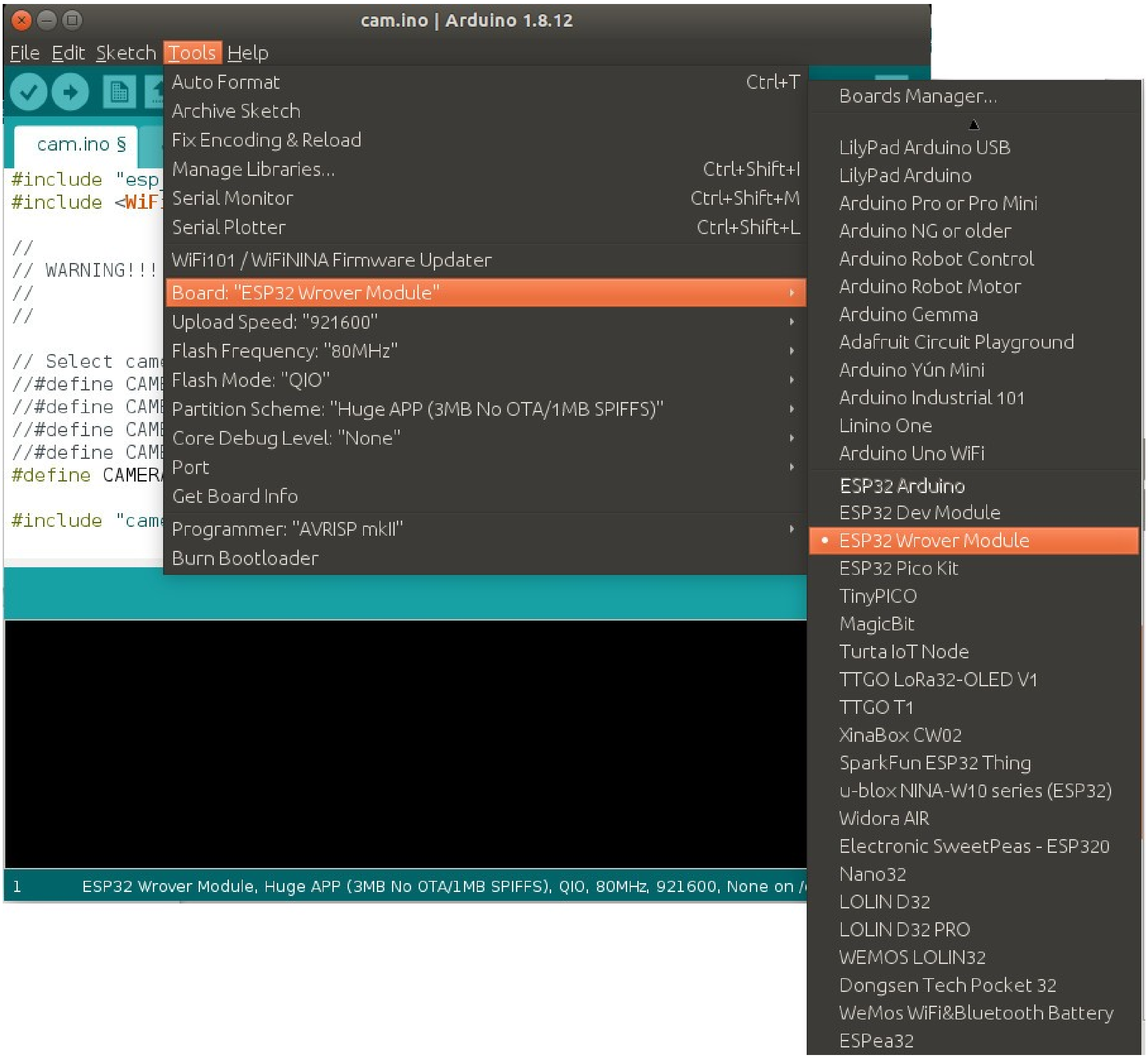
Speed 921600

Flash Frequency 80Mhz

Flash Mode QIO

Partition Scheme: Huge app

Port: the port where your ESP-PROG USB Serial is connected it may be „ttyUSB0“ if you are running Linux or COMxx if you run Windows

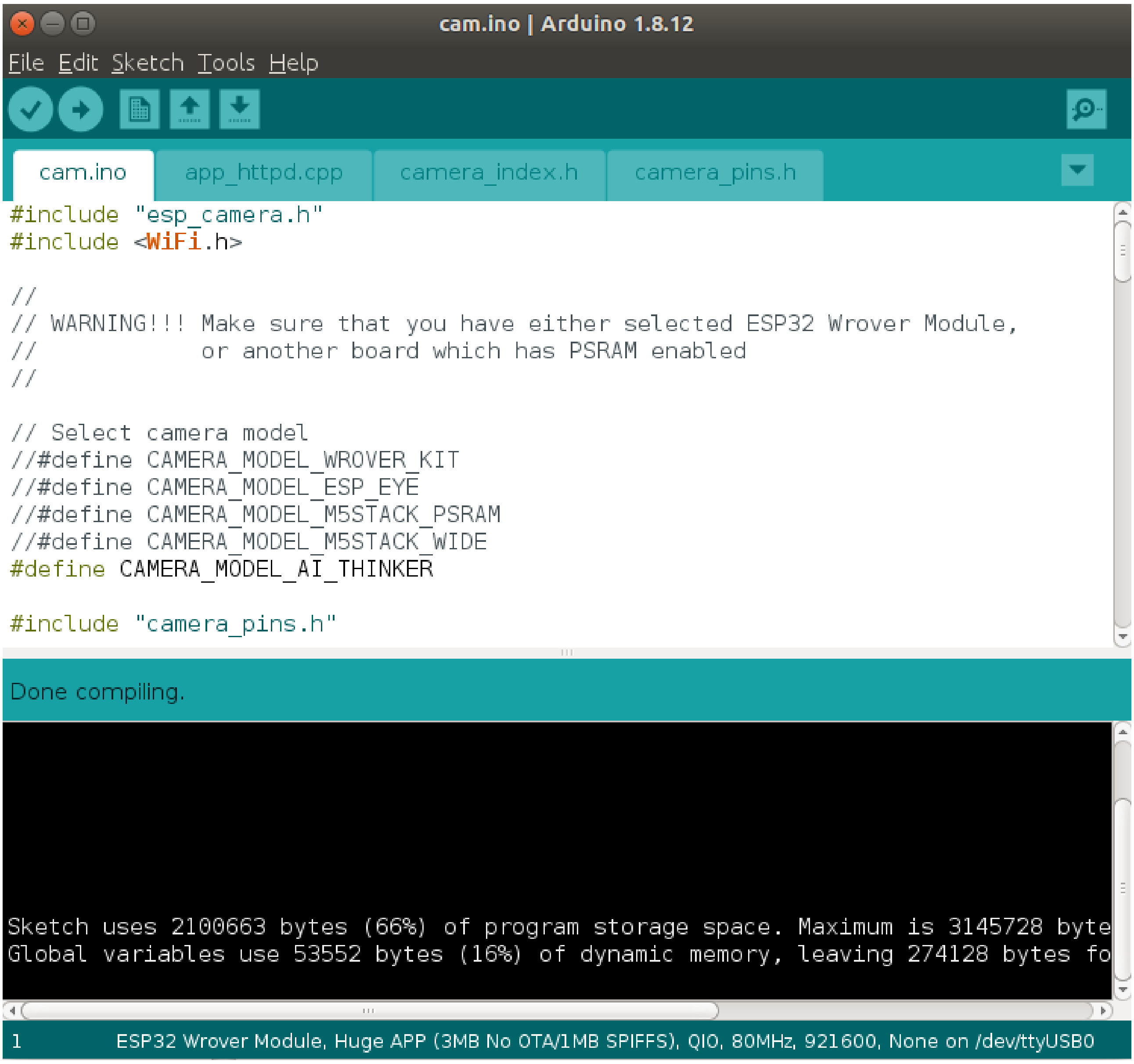


Board: "ESP32 Wrover Module"
Upload Speed: "921600"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Partition Scheme: "Huge APP (3MB No OTA/1MB SPIFFS)"
Core Debug Level: "None"
Programmer: "AVRISP mkII"

```
#include "esp
#include <WiFi

//
// WARNING!!!
//
//

// Select came
//#define CAME
//#define CAME
//#define CAME
//#define CAME
#define CAMERA

#include "came
```

1 ESP32 Wrover Module, Huge APP (3MB No OTA/1MB SPIFFS), QIO, 80MHz, 921600, None on /

Compile and see if there are no errors:



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

//
// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,
//            or another board which has PSRAM enabled
//

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"
```

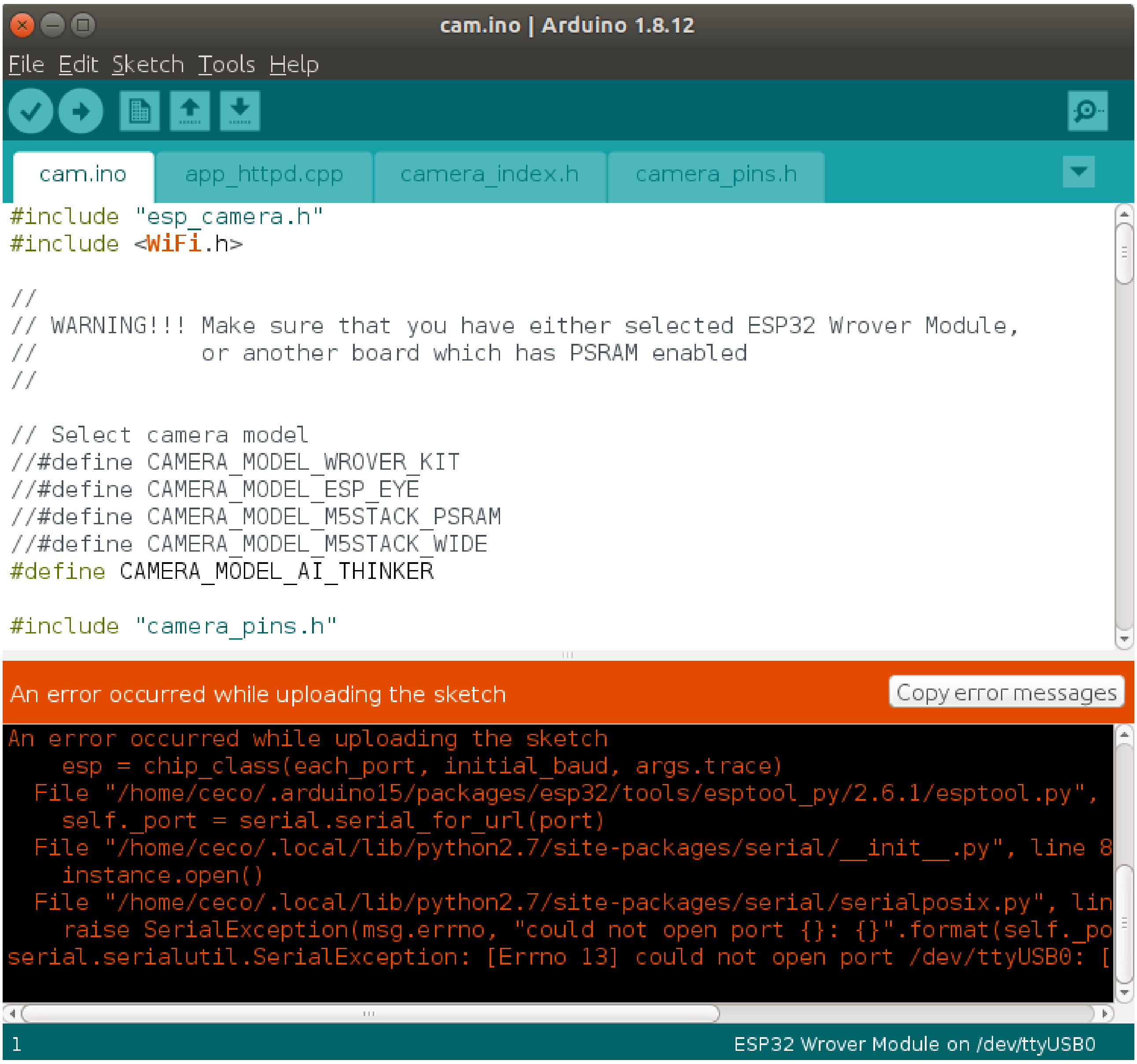
Done compiling.

```
Sketch uses 2100663 bytes (66%) of program storage space. Maximum is 3145728 byte
Global variables use 53552 bytes (16%) of dynamic memory, leaving 274128 bytes fo
```

1 ESP32 Wrover Module, Huge APP (3MB No OTA/1MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0

Then Press RESET button on ESP32-CAM, release it and hit Upload button on Arduino IDE.

If you get this error under Linux:



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

//
// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,
//            or another board which has PSRAM enabled
//

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"
```

An error occurred while uploading the sketch

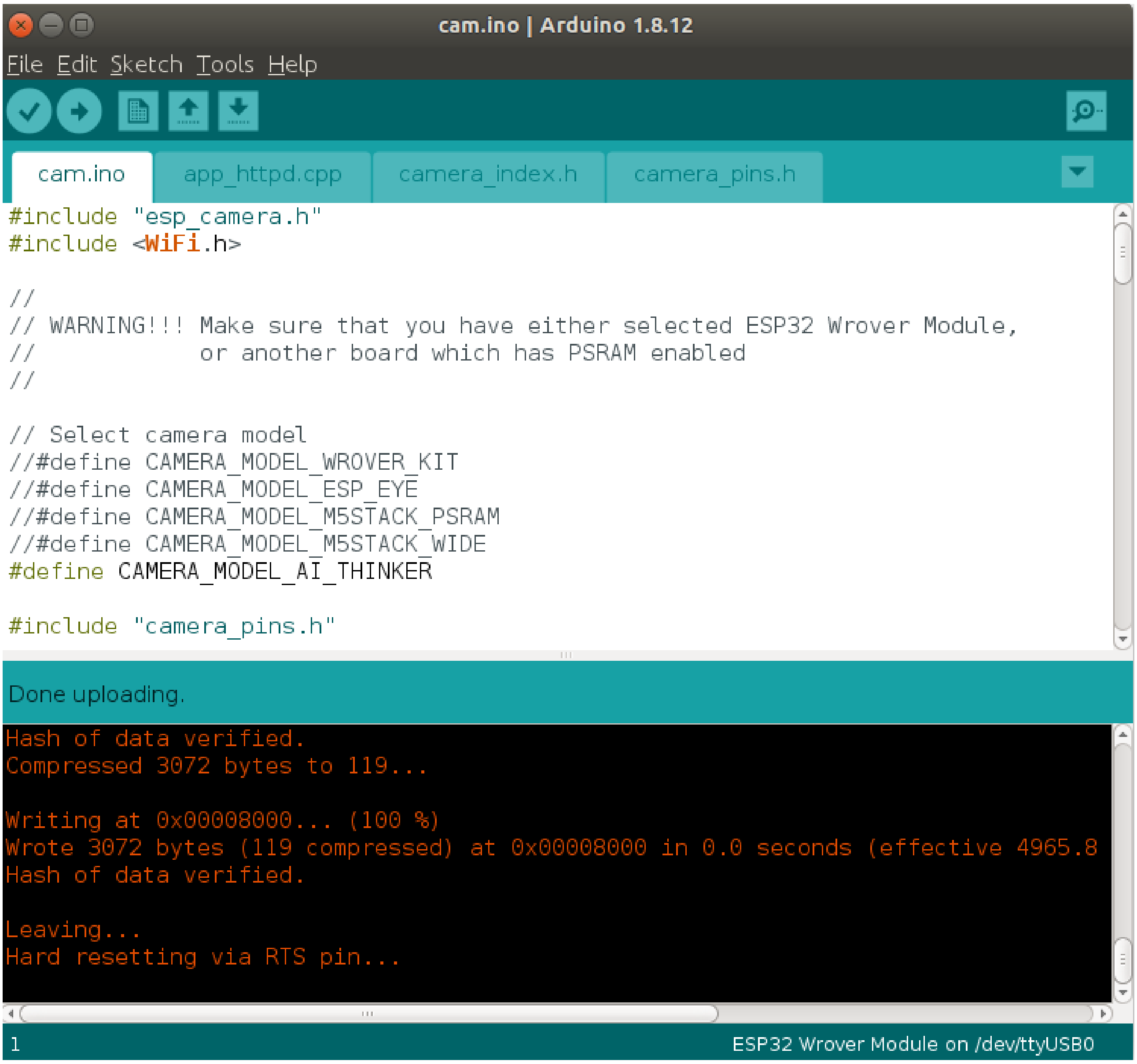
```
An error occurred while uploading the sketch
    esp = chip_class(each_port, initial_baud, args.trace)
  File "/home/ceco/.arduino15/packages/esp32/tools/esptool_py/2.6.1/esptool.py",
    self._port = serial.serial_for_url(port)
  File "/home/ceco/.local/lib/python2.7/site-packages/serial/__init__.py", line 8
    instance.open()
  File "/home/ceco/.local/lib/python2.7/site-packages/serial/serialposix.py", lin
    raise SerialException(msg.errno, "could not open port {}: {}".format(self._po
serial.serialutil.SerialException: [Errno 13] could not open port /dev/ttyUSB0: [
```

This means that the access to ttyUSB0 is not enabled for your user and you have to run in terminal

```
sudo chown youruser /dev/ttyUSB0
```

Where „youruser“ can be seen and try again.

If everything is OK you will see this picture:

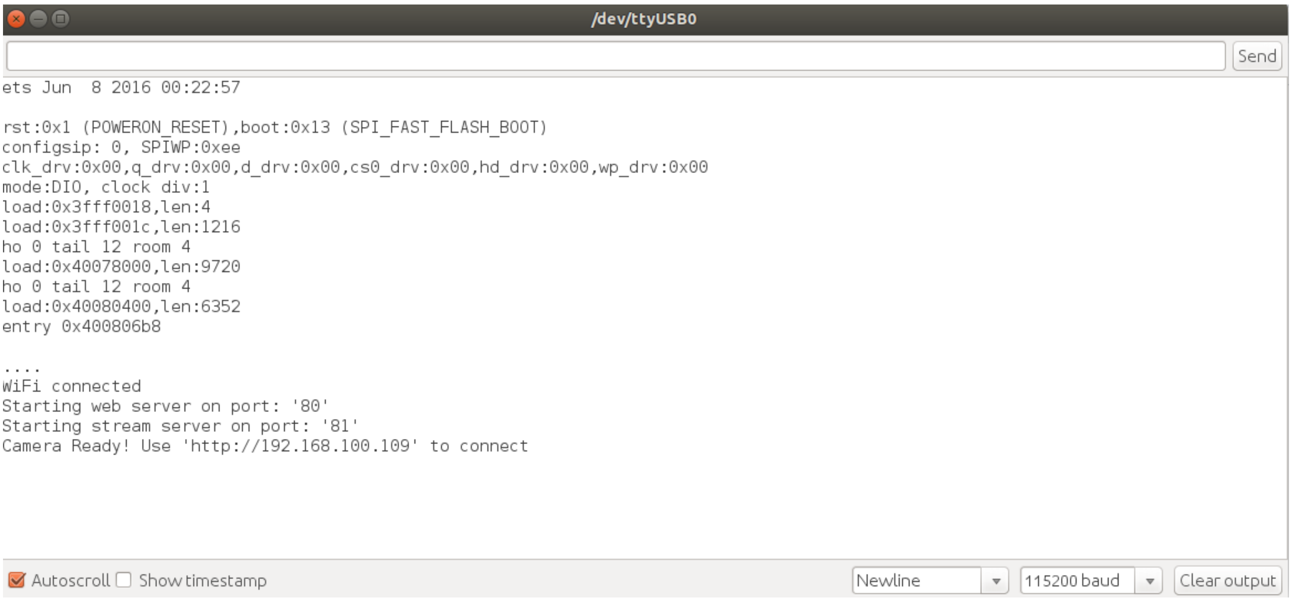


```
#include "esp_camera.h"
#include <WiFi.h>

//
// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,
//            or another board which has PSRAM enabled
//

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"
```

Done uploading.

```
Hash of data verified.
Compressed 3072 bytes to 119...

Writing at 0x00008000... (100 %)
Wrote 3072 bytes (119 compressed) at 0x00008000 in 0.0 seconds (effective 4965.8
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

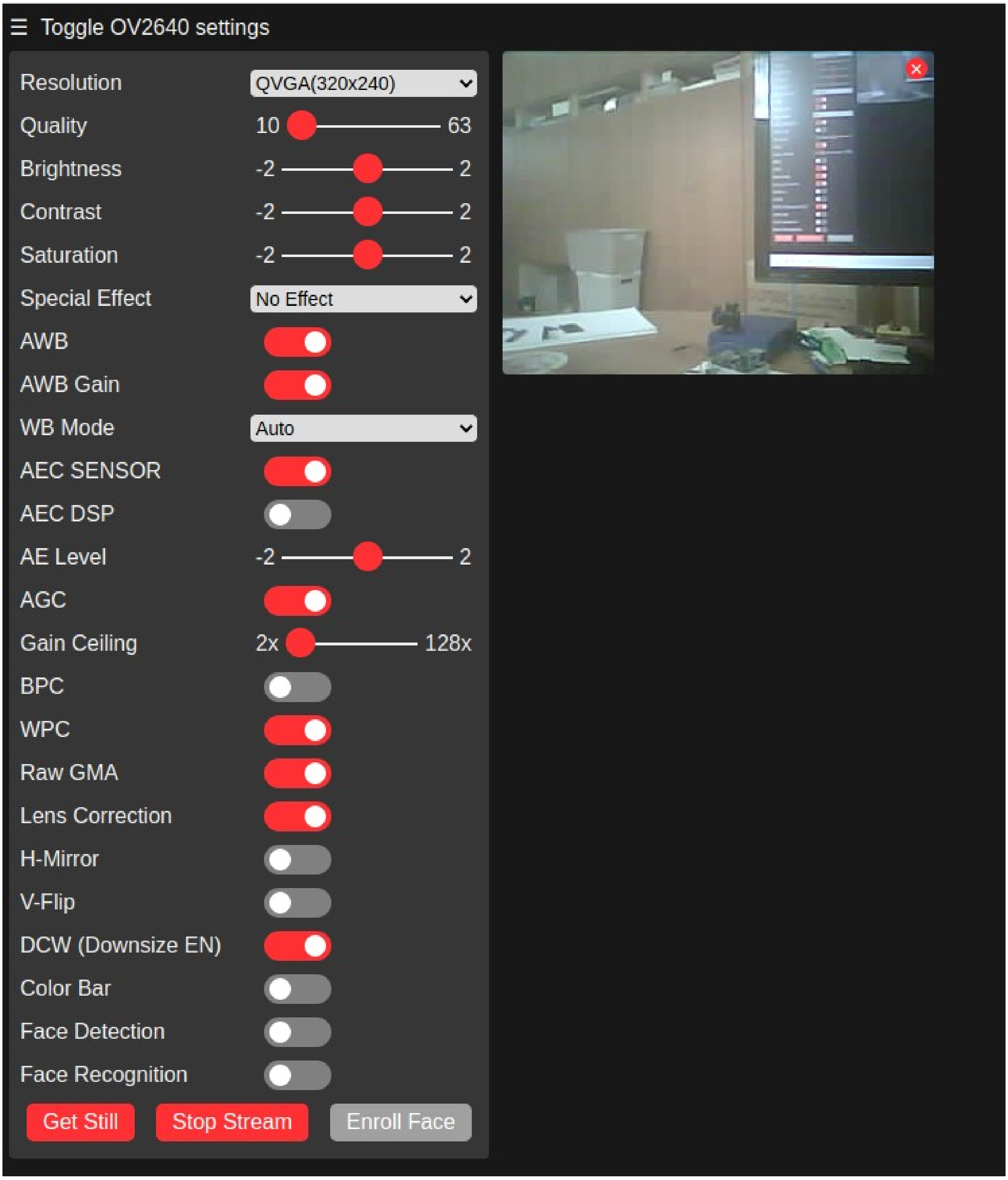
Now you have to disconnect ESP32-CAM IO0 and GND and press reset. In serial monitor you will see this message:



```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8

....
WiFi connected
Starting web server on port: '80'
Starting stream server on port: '81'
Camera Ready! Use 'http://192.168.100.109' to connect
```

☑ Autoscroll □ Show timestamp

when you open http://192.168.100.109/ you will see



☰ Toggle OV2640 settings

Resolution	QVGA(320x240)
Quality	10 — 63
Brightness	-2 — 2
Contrast	-2 — 2
Saturation	-2 — 2
Special Effect	No Effect
AWB	●
AWB Gain	●
WB Mode	Auto
AEC SENSOR	●
AEC DSP	○
AE Level	-2 — 2
AGC	●
Gain Ceiling	2x — 128x
BPC	○
WPC	●
Raw GMA	●
Lens Correction	●
H-Mirror	○
V-Flip	○
DCW (Downsize EN)	●
Color Bar	○
Face Detection	○
Face Recognition	○

Now you can play with the different settings!