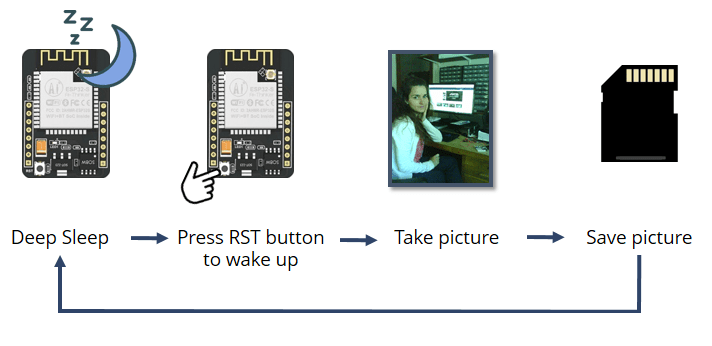
**Project Overview**



* The ESP32-CAM is in deep sleep mode
* Press the RESET button to wake up the board
* The camera takes a photo
* The photo is saved in the microSD card with the name: pictureX.jpg, where X corresponds to the picture number
* The picture number will be saved in the ESP32 flash memory so that it is not erased during RESET and we can keep track of the number of photos taken.

**Formatting MicroSD Card**

The first thing we recommend doing is formatting your microSD card. You can use the Windows formatter tool or any other microSD formatter software.

|  |  |
| --- | --- |
| **1.** Insert the microSD card in your computer. Go to **My Computer** and right click in the SD card. Select **Format** |  |
| **2.**A new window pops up. Select **FAT32**, press **Start** to initialize the formatting process and follow the onscreen instructions. |  |

IMPORTANT!!!

- Select Board "ESP32 Wrover Module"

- Select the Partion Scheme "Huge APP (3MB No OTA)

- GPIO 0 must be connected to GND to upload a sketch

- After connecting GPIO 0 to GND, press the ESP32-CAM on-board RESET button to put your board in flashing mode

\*\*\*\*\*\*\*\*\*/

#include "esp\_camera.h"

#include "esp\_timer.h"

#include "img\_converters.h"

#include "Arduino.h"

#include "fb\_gfx.h"

#include "fd\_forward.h"

#include "fr\_forward.h"

#include "FS.h" // SD Card ESP32

#include "SD\_MMC.h" // SD Card ESP32

#include "soc/soc.h" // Disable brownour problems

#include "soc/rtc\_cntl\_reg.h" // Disable brownour problems

#include "dl\_lib.h"

#include "driver/rtc\_io.h"

#include <EEPROM.h> // read and write from flash memory

// define the number of bytes you want to access

#define EEPROM\_SIZE 1

// Pin definition for CAMERA\_MODEL\_AI\_THINKER

#define PWDN\_GPIO\_NUM 32

#define RESET\_GPIO\_NUM -1

#define XCLK\_GPIO\_NUM 0

#define SIOD\_GPIO\_NUM 26

#define SIOC\_GPIO\_NUM 27

#define Y9\_GPIO\_NUM 35

#define Y8\_GPIO\_NUM 34

#define Y7\_GPIO\_NUM 39

#define Y6\_GPIO\_NUM 36

#define Y5\_GPIO\_NUM 21

#define Y4\_GPIO\_NUM 19

#define Y3\_GPIO\_NUM 18

#define Y2\_GPIO\_NUM 5

#define VSYNC\_GPIO\_NUM 25

#define HREF\_GPIO\_NUM 23

#define PCLK\_GPIO\_NUM 22

int pictureNumber = 0;

void setup() {

WRITE\_PERI\_REG(RTC\_CNTL\_BROWN\_OUT\_REG, 0); //disable brownout detector

Serial.begin(115200);

//Serial.setDebugOutput(true);

//Serial.println();

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.pixel\_format = PIXFORMAT\_JPEG;

if(psramFound()){

config.frame\_size = FRAMESIZE\_UXGA; // FRAMESIZE\_ + QVGA|CIF|VGA|SVGA|XGA|SXGA|UXGA

config.jpeg\_quality = 10;

config.fb\_count = 2;

} else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

// Init Camera

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}

//Serial.println("Starting SD Card");

if(!SD\_MMC.begin()){

Serial.println("SD Card Mount Failed");

return;

}

uint8\_t cardType = SD\_MMC.cardType();

if(cardType == CARD\_NONE){

Serial.println("No SD Card attached");

return;

}

camera\_fb\_t \* fb = NULL;

// Take Picture with Camera

fb = esp\_camera\_fb\_get();

if(!fb) {

Serial.println("Camera capture failed");

return;

}

// initialize EEPROM with predefined size

EEPROM.begin(EEPROM\_SIZE);

pictureNumber = EEPROM.read(0) + 1;

// Path where new picture will be saved in SD Card

String path = "/picture" + String(pictureNumber) +".jpg";

fs::FS &fs = SD\_MMC;

Serial.printf("Picture file name: %s\n", path.c\_str());

File file = fs.open(path.c\_str(), FILE\_WRITE);

if(!file){

Serial.println("Failed to open file in writing mode");

}

else {

file.write(fb->buf, fb->len); // payload (image), payload length

Serial.printf("Saved file to path: %s\n", path.c\_str());

EEPROM.write(0, pictureNumber);

EEPROM.commit();

}

file.close();

esp\_camera\_fb\_return(fb);

// Turns off the ESP32-CAM white on-board LED (flash) connected to GPIO 4

pinMode(4, OUTPUT);

digitalWrite(4, LOW);

rtc\_gpio\_hold\_en(GPIO\_NUM\_4);

delay(2000);

Serial.println("Going to sleep now");

delay(2000);

esp\_deep\_sleep\_start();

Serial.println("This will never be printed");

}

void loop() {

}

Initialize an int variable called pictureNumber that that will generate the photo name: picture1.jpg, picture2.jpg, and so on.

int pictureNumber = 0;

All our code is in the setup(). The code only runs once when the ESP32 wakes up (in this case when you press the on-board RESET button).

Define the camera settings:

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.pixel\_format = PIXFORMAT\_JPEG;

Use the following settings for a camera with PSRAM (like the one we’re using in this tutorial).

if(psramFound()){

config.frame\_size = FRAMESIZE\_UXGA; // FRAMESIZE\_ + QVGA|CIF|VGA|SVGA|XGA|SXGA|UXGA

config.jpeg\_quality = 10;

config.fb\_count = 2;

}

If the board doesn’t have PSRAM, set the following:

else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12;

config.fb\_count = 1;

}

Initialize the camera:

// Init Camera

esp\_err\_t err = esp\_camera\_init(&config);

if (err != ESP\_OK) {

Serial.printf("Camera init failed with error 0x%x", err);

return;

}