ESP32-CAM AI-Thinker Pinout Guide: GPIOs Usage Explained

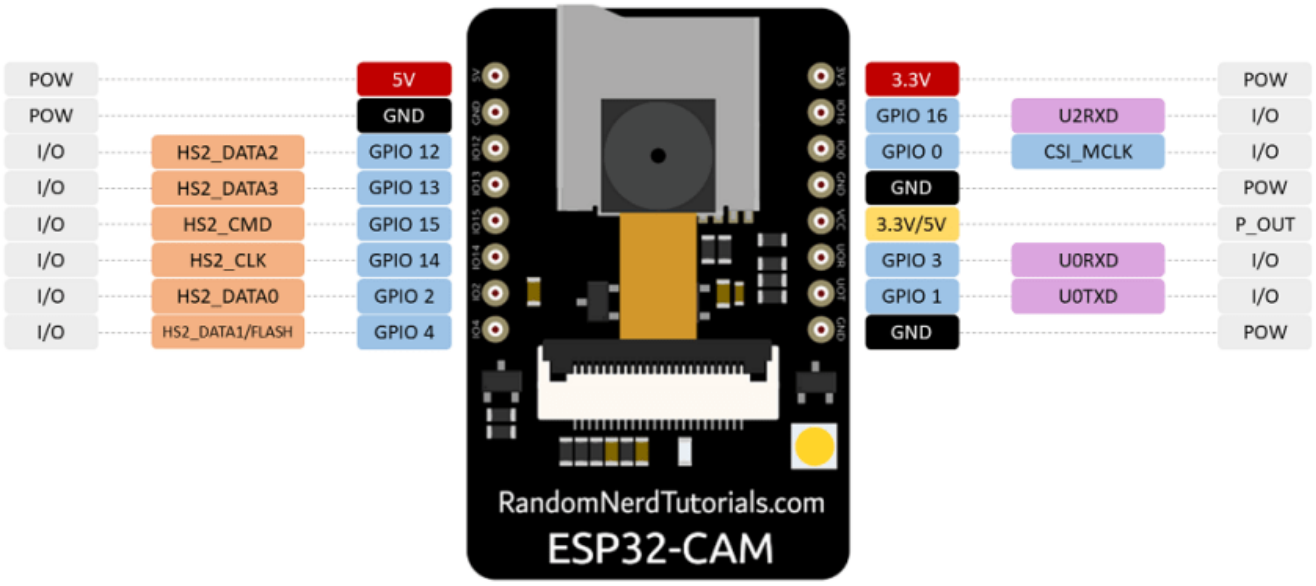
randomnerdtutorials.com/esp32-cam-ai-thinker-pinout

March 10, 2020

The ESP32-CAM is a development board with an ESP32-S chip, an OV2640 camera, microSD card slot and several GPIOs to connect peripherals. In this guide, we'll take a look at the ESP32-CAM GPIOs and how to use them.

Pinout Diagram

The following image shows the pinout diagram for the ESP32-CAM AI-Thinker.



Schematic Diagram

The following figure shows the schematic diagram for the ESP32-CAM.

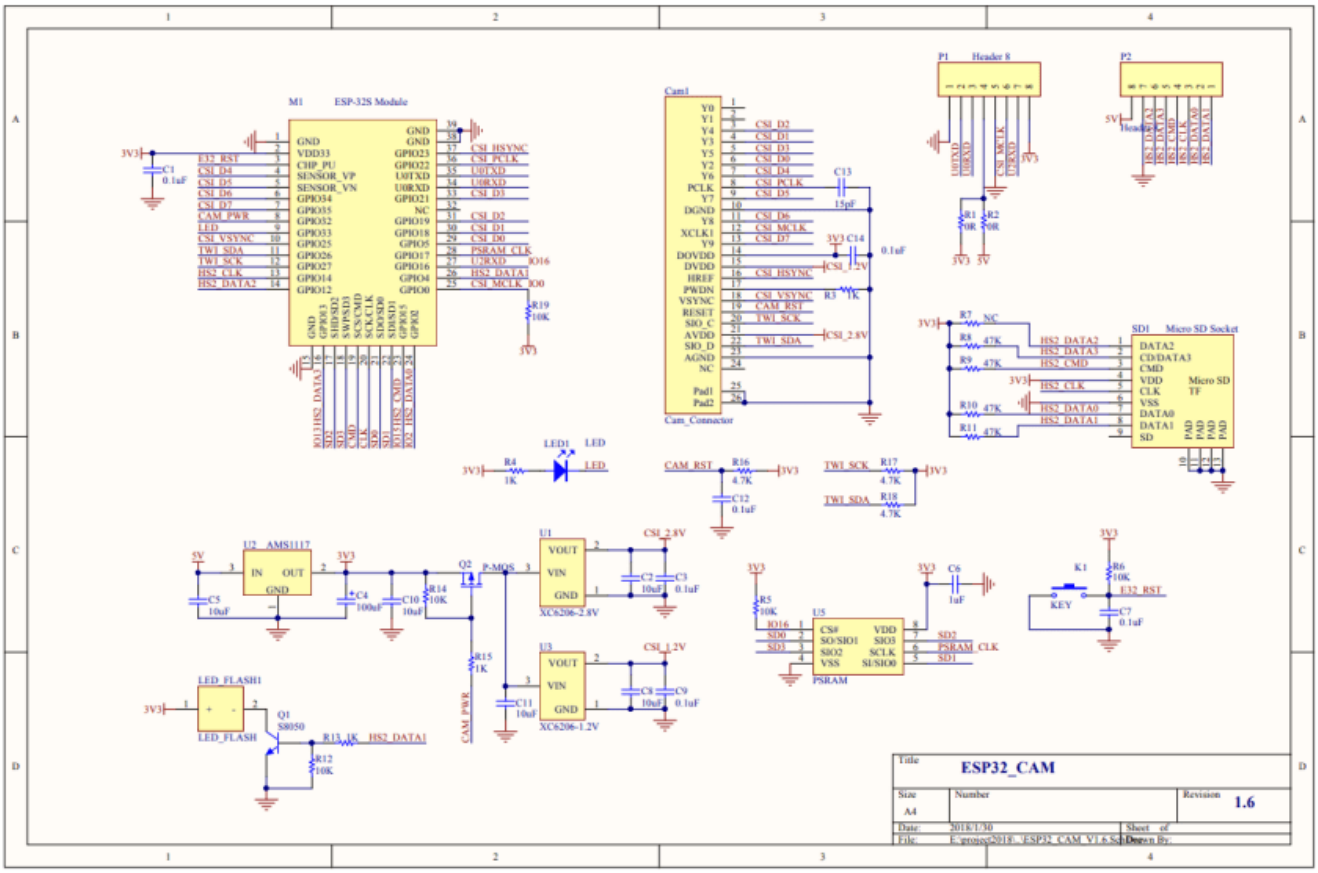


Image Source

You can download a PDF file with better resolution on this GitHub repository.

Power Pins

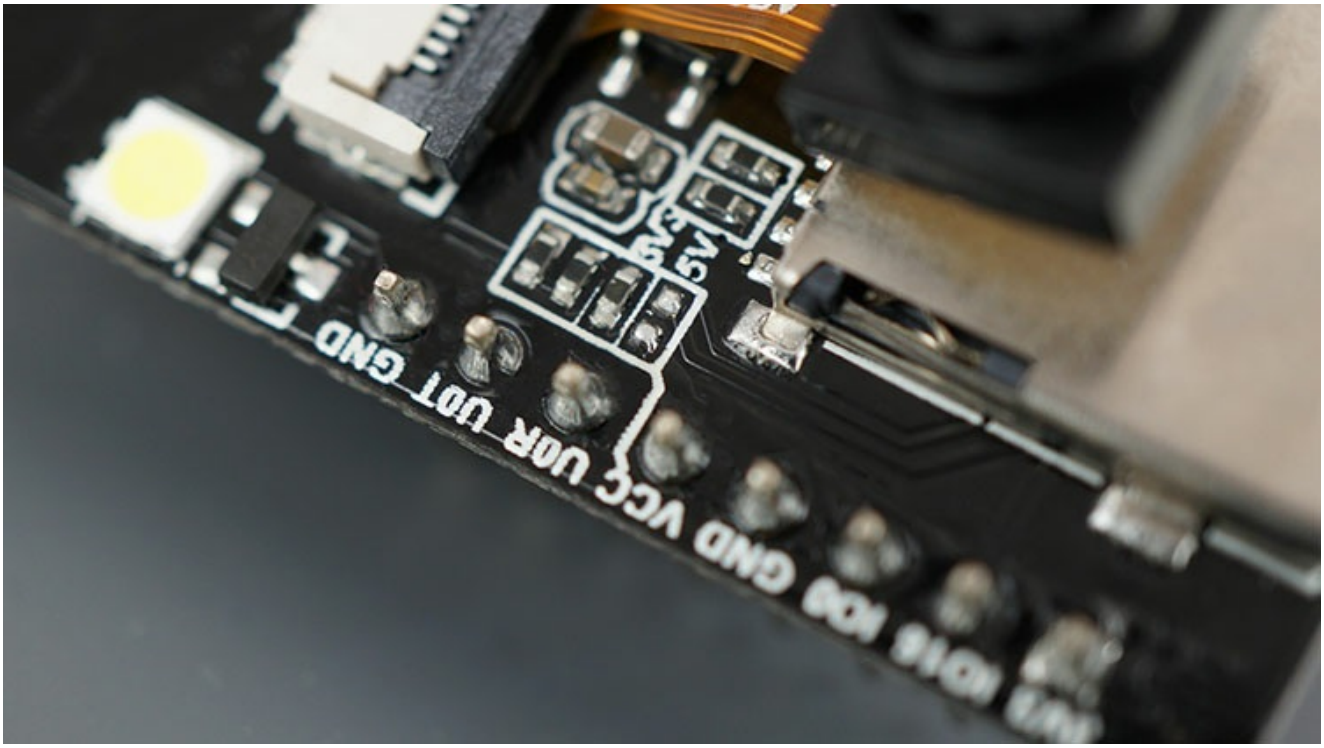
The ESP32-CAM comes with three GND pins (colored in black color) and two power pins (colored with red color): 3.3V and 5V.

You can power the ESP32-CAM through the 3.3V or 5V pins. However, many people reported errors when powering the ESP32-CAM with 3.3V, so we always advise to **power the ESP32-CAM through the 5V pin**.

Power output pin

There's also the pin labeled on the silkscreen as **VCC** (colored with a yellow rectangle). You should not use that pin to power the ESP32-CAM. That is an output power pin. It can either output 5V or 3.3V.

In our case, the ESP32-CAM outputs 3.3V whether it is powered with 5V or 3.3V. Next to the VCC pin, there are two pads. One labeled as 3.3V and other as 5V.



If you look closely, you should have a jumper on the 3.3V pads. If you want to have an output of 5V on the VCC pin, you need to unsolder that connection and solder the 5V pads.

Serial Pins

GPIO 1 and GPIO 3 are the serial pins (TX and RX, respectively). Because the ESP32-CAM doesn't have a built-in programmer, you need to use these pins to communicate with the board and upload code.

The best way to upload code to the ESP32-CAM is using an FTDI programmer.

Learn how to upload code to the ESP32-CAM AI-Thinker.

You can use GPIO 1 and GPIO 3 to connect other peripherals like outputs or sensors after uploading the code. However, you won't be able to open the Serial Monitor and see if everything is going well with your setup.

GPIO 0

GPIO 0 determines whether the ESP32 is in flashing mode or not. This GPIO is internally connected to a pull-up 10k Ohm resistor.

When GPIO 0 is connected to GND, the ESP32 goes into flashing mode and you can upload code to the board.

GPIO 0 connected to GND » ESP32-CAM in flashing mode

To make the ESP32 run “normally”, you just need to disconnect GPIO 0 from GND.

GPIO 16

GPIO 16 is by default a UART pin. However, you can use it as an input. It is internally connected to a 10k Ohm pull-up resistor. So, it is in high state as default.

Note that **GPIO 16 is not an ADC pin**, so you can’t read analog sensors on this pin.

Additionally, **GPIO 16 is not an RTC GPIO**, so it can’t be used as an external wake-up source.

MicroSD Card Connections

The following pins are used to interface with the microSD card when it is on operation.

MicroSD card	ESP32
CLK	GPIO 14
CMD	GPIO 15
DATA0	GPIO 2
DATA1 / flashlight	GPIO 4
DATA2	GPIO 12
DATA3	GPIO 13

If you’re not using the microSD card, you can use these pins as regular inputs/outputs. You can take a look at the ESP32 pinout guide to see the features of these pins.

All these GPIOs are RTC and support ADC: GPIOs 2, 4, 12, 13, 14, and 15.

Flashlight (GPIO 4)

The ESP32-CAM has a very bright built-in LED that can work as a flash when taking photos. That LED is internally connected to GPIO 4.

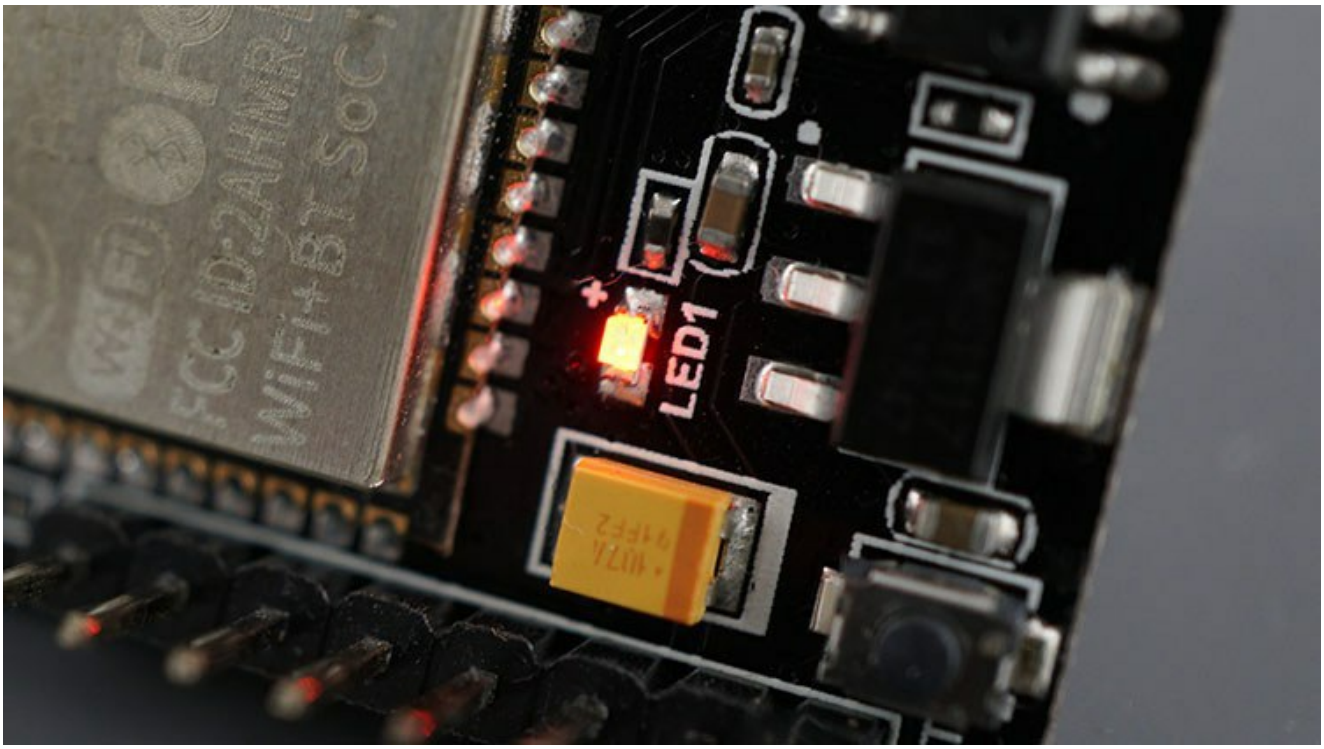
That GPIO is also connected to the microSD card slot, so you may have troubles when trying to use both at the same time – the flashlight will light up when using the microSD card.

Note: one of our readers shared that if you initialize the microSD card as follows, you won't have this problem because the microSD card won't use that data line.*

```
SD_MMC.begin("/sdcard", true)
```

* we found that this works and that the LED will not make that flash effect. However, the LED remains on with low brightness – we're not sure if we are missing something.

GPIO 33 – Built-in Red LED



Next to the RST button, there's an on-board red LED. That LED is internally connected to GPIO 33. You can use this LED to indicate that something is happening. For example, if the Wi-Fi is connected, the LED is red or vice-versa.

That LED works with inverted logic, so you send a LOW signal to turn it on and a HIGH signal to turn it off.

You can experiment uploading the following snippet and see if you get that LED glowing.

```
void setup() {
  pinMode(33, OUTPUT);
}

void loop() {
  pinMode(33, LOW);
}
```

Camera Connections

The connections between the camera and the ESP32-CAM AI-Thinker are shown in the following table.

OV2640 CAMERA	ESP32	Variable name in code
D0	GPIO 5	Y2_GPIO_NUM
D1	GPIO 18	Y3_GPIO_NUM
D2	GPIO 19	Y4_GPIO_NUM
D3	GPIO 21	Y5_GPIO_NUM
D4	GPIO 36	Y6_GPIO_NUM
D5	GPIO 39	Y7_GPIO_NUM
D6	GPIO 34	Y8_GPIO_NUM
D7	GPIO 35	Y9_GPIO_NUM
XCLK	GPIO 0	XCLK_GPIO_NUM
PCLK	GPIO 22	PCLK_GPIO_NUM
VSYNC	GPIO 25	VSYNC_GPIO_NUM
HREF	GPIO 23	HREF_GPIO_NUM
SDA	GPIO 26	SIOD_GPIO_NUM
SCL	GPIO 27	SIOC_GPIO_NUM
POWER PIN	GPIO 32	PWDN_GPIO_NUM

So, the pin definition for the ESP32-CAM AI-Thinker on the Arduino IDE should be as follows:

```
#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
```

Wrapping Up

We hope you've found this guide for the ESP32-CAM GPIOs useful. If you have any tips or more info about the ESP32-CAM GPIOs, write a comment below.

We have several projects with the ESP32-CAM that you may like:

Thanks for reading.