**ESP32 Pinout Reference: Which GPIO pins should you use?**

ESP32 has 48 pins with multiple functions. Not all pins are exposed in all ESP32, and some cannot be used.

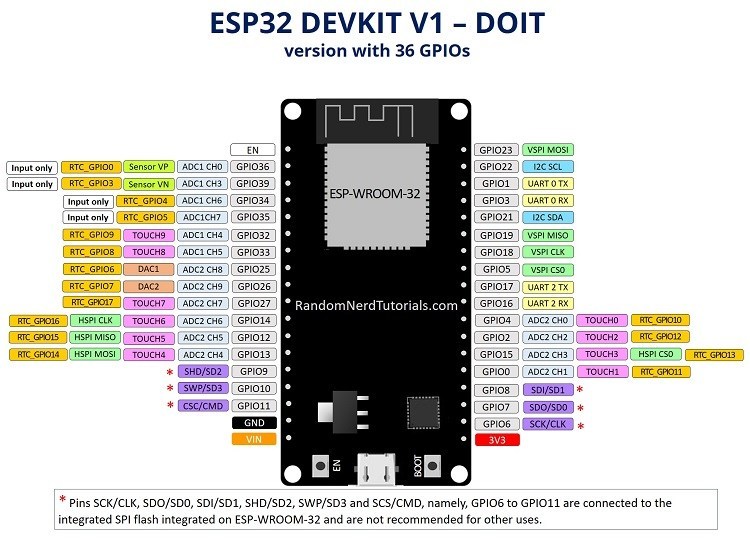
**ESP32 Peripherals**

The ESP32 peripherals include:

* 18 Analog-to-Digital Converter (ADC) channels
* 3 SPI interfaces
* 3 UART interfaces
* 2 I2C interfaces
* 16 PWM output channels
* 2 Digital-to-Analog Converters (DAC)
* 2 I2S interfaces
* 10 Capacitive sensing GPIOs

The ADC and DAC features are assigned to specific static pins. However, you can decide which pins are UART, I2C, SPI, PWM, etc.

Although you can define the pins properties on the software, there are pins assigned by default as shown in the ESP32 DevKit.

[](https://i1.wp.com/randomnerdtutorials.com/wp-content/uploads/2018/08/ESP32-DOIT-DEVKIT-V1-Board-Pinout-36-GPIOs-updated.jpg?ssl=1)

### **Pins HIGH at Boot**

Some GPIO’s change its state to HIGH or output PWM signals at boot or reset. This means that if you have outputs connected to these GPIOs you may get unexpected results when the ESP32 resets or boots.

* GPIO 1
* GPIO 3
* GPIO 5
* GPIO 6 to GPIO 11 (connected to the ESP32 integrated SPI flash memory – not recommended to use).
* GPIO 14
* GPIO 15

### **SPI**

By default, the pin mapping for SPI is:

**SPI MOSI MISO CLK CS**

VSPI GPIO 23 GPIO 19 GPIO 18 GPIO 5

HSPI GPIO 13 GPIO 12 GPIO 14 GPIO 15

### **SPI flash integrated on the ESP-WROOM-32**

GPIO 6 to GPIO 11 are exposed in some ESP32 development boards. However, these pins are connected to the integrated SPI flash on the ESP-WROOM-32 chip and are not recommended for other uses. So, don’t use these pins in your projects:

* GPIO 6 , SCK / CLK
* GPIO 7 , SDO / SD0
* GPIO 8 , SDI / SD1
* GPIO 9 , SHD / SD2`
* GPIO 10 , SWP / SD3
* GPIO 11 , CSC / CMD

### 

### **I2C**

The ESP32 has two I2C channels and any pin can be set as SDA or SCL. When using the ESP32 with the Arduino IDE, the default I2C pins are:

Both I2C interfaces are connected to GPIO matrix, so you can select arbitrary IO pins for SDA and SCL (NOT GPIO34 – 39 though)

* GPIO 21 , SDA
* GPIO 22 , SCL

If you want to use other pins, when using the wire library, you just need to call:

Wire.begin(SDA, SCL); // like Wire.begin(17, 18);

### **Strapping Pins**

The ESP32 chip has the following strapping pins:

* GPIO 0
* GPIO 2
* GPIO 4
* GPIO 5 ( must be HIGH during boot )
* GPIO 12 ( must be LOW during boot )
* GPIO 15 ( must be HIGH during boot )

These are used to put the ESP32 into bootloader or flashing mode.

However, if you have peripherals connected to those pins, you may have trouble trying to upload new code, flashing the ESP32 with new firmware or resetting the board. If you have some peripherals connected to the strapping pins and you are getting trouble uploading code or flashing the ESP32, it may be because those peripherals are preventing the ESP32 to enter the right mode.

### **Digital to Analog Converter (DAC)**

There are 2 x 8 bits DAC channels on the ESP32 to convert digital signals into analog voltage signal outputs. These are the DAC channels:

* DAC1 , GPIO25
* DAC2 , GPIO26

### **Analog to Digital Converter (ADC)**

The ESP32 has 18 x 12 bits ADC input channels (while the ESP8266 only has 1x 10 bits ADC). These are the GPIOs that can be used as ADC and respective channels:

* ADC2\_CH0 , GPIO 4
* ADC2\_CH1 , GPIO 0
* ADC2\_CH2 , GPIO 2
* ADC2\_CH3 , GPIO 15
* ADC2\_CH4 , GPIO 13
* ADC2\_CH5 , GPIO 12
* ADC2\_CH6 , GPIO 14
* ADC2\_CH7 , GPIO 27
* ADC2\_CH8 , GPIO 25
* ADC2\_CH9 , GPIO 26
* ADC1\_CH0 , GPIO 36
* ADC1\_CH1 , GPIO 37
* ADC1\_CH2 , GPIO 38
* ADC1\_CH3 , GPIO 39
* ADC1\_CH4 , GPIO 32
* ADC1\_CH5 , GPIO 33
* ADC1\_CH6 , GPIO 34
* ADC1\_CH7 , GPIO 35

### **NOTE: ADC2 pins cannot be used when Wi-Fi is used. So, if you’re using Wi-Fi and you’re having trouble getting the value from an ADC2 GPIO, you may consider using an ADC1 GPIO instead, that should solve your problem.**

### **Capacitive touch GPIOs**

ESP32 has 10 internal capacitive touch sensors. These can sense variations in anything that holds an electrical charge, like the human skin. So they can detect variations induced when touching the GPIOs with a finger. These can replace mechanical buttons. The capacitive touch pins can also be used to wake up the ESP32 from deep sleep.

Those internal touch sensors are connected to these GPIOs:

* T0 , GPIO 4
* T1 , GPIO 0
* T2 , GPIO 2
* T3 , GPIO 15
* T4 , GPIO 13
* T5 , GPIO 12
* T6 , GPIO 14
* T7 , GPIO 27
* T8 , GPIO 33
* T9 , GPIO 32

### **Input only pins**

GPIOs 34 to 39 are GPIs – input only pins. These pins don’t have internal pull-ups or pull-down resistors. They can’t be used as outputs, so use these pins only as inputs:

* GPIO 34
* GPIO 35
* GPIO 36 SVP
* GPIO 39 SVN

### **PWM**

The ESP32 LED PWM controller has 16 independent channels that can be configured to generate PWM signals with different properties. All pins that can act as outputs can be used as PWM pins (GPIOs 34 to 39 can’t generate PWM).

To set a PWM signal, you need to define these parameters in the code:

* Signal’s frequency;
* Duty cycle;
* PWM channel;
* GPIO where you want to output the signal.

### **Enable (EN)**

Enable (EN) is the 3.3V regulator’s enable pin. It’s pulled up, so connect to ground to disable the 3.3V regulator. This means that you can use this pin connected to a pushbutton to restart your ESP32, for example.

### **GPIO current drawn**

The absolute maximum current drawn per GPIO is 40mA according to the “Recommended Operating Conditions” section in the ESP32 datasheet.

### **ESP32 Built-In Hall Effect Sensor**

The ESP32 also features a built-in Hall Effect sensor to detect the magnetic field in its surroundings.

**GPIO In Out Mapped Your Notes Pin Notes**

0 p.up OK outputs PWM signal at boot

1 TX pin OK debug output at boot

2 OK OK connected to on-board LED

3 OK RX pin HIGH at boot

4 OK OK

5 OK OK V.SPI CS outputs PWM signal at boot

6 x x connected to the integrated SPI flash

7 x x connected to the integrated SPI flash

8 x x connected to the integrated SPI flash

9 x x connected to the integrated SPI flash

10 x x connected to the integrated SPI flash

11 x x connected to the integrated SPI flash

12 OK OK H.SPI MISO boot fail if pulled high

13 OK OK H.SPI MOSI

14 OK OK H.SPI CLK outputs PWM signal at boot

15 OK OK H.SPI CS outputs PWM signal at boot

16 OK OK

17 OK OK

This shows what pins are best to use as inputs, outputs and which ones you need to be cautious.

GREEN Pins are OK to use.

YELLOW are OK, but pay attention because they may have unexpected behavior mainly at boot.

RED not recommended for inputs or outputs but may work in some applications.

18 OK OK V.SPI CLK

19 OK OK V.SPI MISO

21 OK OK I2C SDA

22 OK OK I2C SCL

23 OK **OK** V.SPI MOSI

25 OK OK

26 OK OK

27 OK OK

32 OK OK

33 OK OK

34 OK - input only NO Internal Pullup or pulldown resistors

35 OK - input only NO Internal Pullup or pulldown resistors

36 OK - SVP input only NO Internal Pullup or pulldown resistors

39 OK - SVN input only NO Internal Pullup or pulldown resistors