**Input only pins**

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

* GPIO 34
* GPIO 35
* GPIO 36
* GPIO 39

**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. On most development boards with built-in USB/Serial, you don’t need to worry about the state of these pins. The board puts the pins in the right state for flashing or boot mode. More information on the [ESP32 Boot Mode Selection can be found here](https://github.com/espressif/esptool/wiki/ESP32-Boot-Mode-Selection).

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 from entering the right mode. Read the [Boot Mode Selection documentation](https://github.com/espressif/esptool/wiki/ESP32-Boot-Mode-Selection) to guide you in the right direction. After resetting, flashing, or booting, those pins work as expected.

**Pins HIGH at Boot**

Some GPIOs change their 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

**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 Peripherals and I/O

Although the ESP32 has total 48 GPIO pins, only 25 of them are broken out to the pin headers on both sides of the development board. These pins can be assigned to all sorts of peripheral duties, including:

|  |  |
| --- | --- |
| 15 ADC channels | 15 channels of 12-bit SAR ADC’s. The ADC range can be set, in firmware, to either 0-1V, 0-1.4V, 0-2V, or 0-4V |
| 2 UART interfaces | 2 UART interfaces. One is used to load code serially. They feature flow control, and support IrDA too! |
| 25 PWM outputs | 25 channels of PWM pins for dimming LEDs or controlling motors. |
| 2 DAC channels | 8-bit DACs to produce true analog voltages. |
| 3 SPI & 1 I2C interfaces | There are 3 SPI and 1 I2C interfaces to hook up all sorts of sensors and peripherals. |
| 9 Touch Pads | 9 GPIOs feature capacitive touch sensing. |

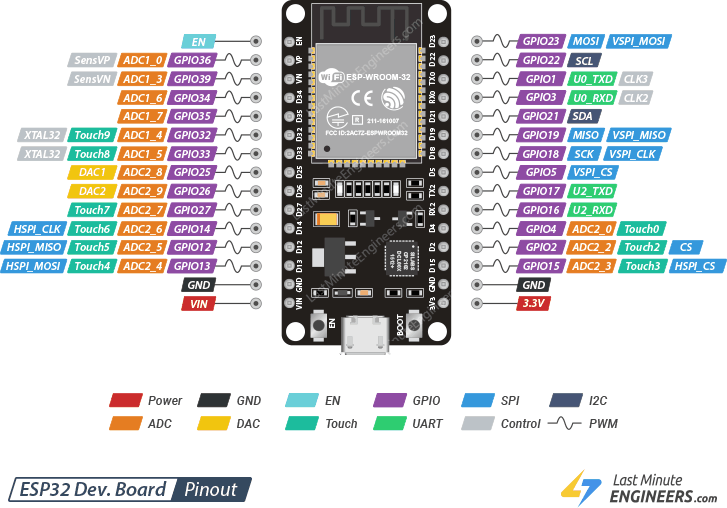
Thanks to the ESP32’s pin multiplexing feature (Multiple peripherals multiplexed on a single GPIO pin). Meaning a single GPIO pin can act as an ADC input/DAC output/Touch pad.

You can get extensive information about ESP32 from the datasheet.

[ESP32 Datasheet](https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf)

ESP32 Pinout

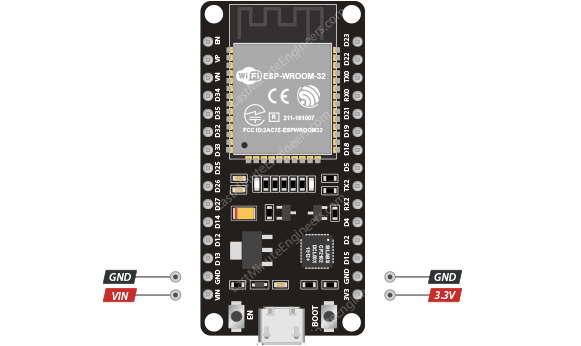
The ESP32 development board has total 30 pins that interface it to the outside world. For the sake of simplicity, pins with similar functionality are grouped together. The connections are as follows:



Let us analyze the ESP32 pins and their functions in more detail one by one.

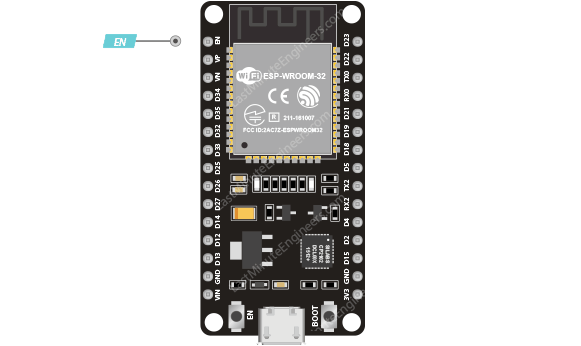
Power Pins

There are two power pins viz. VIN pin & 3.3V pin. The VIN pin can be used to directly supply the ESP32 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pin is the output of an on-board voltage regulator. This pin can be used to supply power to external components. GND is a ground pin of ESP32 development board.



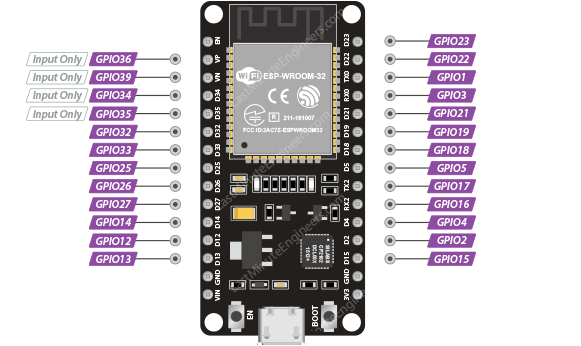
Enable Pin

EN Pin is used to enable ESP32. The chip is enabled when pulled HIGH. When pulled LOW the chip works at minimum power.



GPIO Pins

ESP32 development board has 25 GPIO pins which can be assigned to various functions programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance.



Input Only GPIOs

Pin D34, D35, VP and VN cannot be configured as outputs; they can be used as either digital inputs, analog inputs, or for other unique purposes. Also note that they do not have internal pull-up or pull-down resistors, like the other GPIO pins.

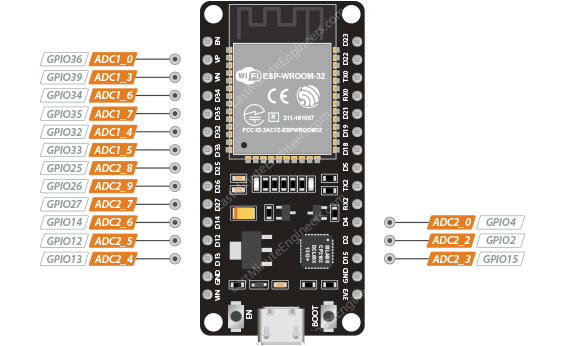
Also GPIO pins VP and VN are an integral part of the ultra-low-noise pre-amplifier for the ADC, which help to configure the sampling time and noise of the pre-amp.

ESP32 Interrupt Pins

All GPIOs can be configured as [interrupts](https://lastminuteengineers.com/handling-esp32-gpio-interrupts-tutorial/).

ADC Pins

The ESP32 has fifteen 12-bit ADC input channels. These are GPIOs that can be used to convert the analog voltage on the pin to a digital number.

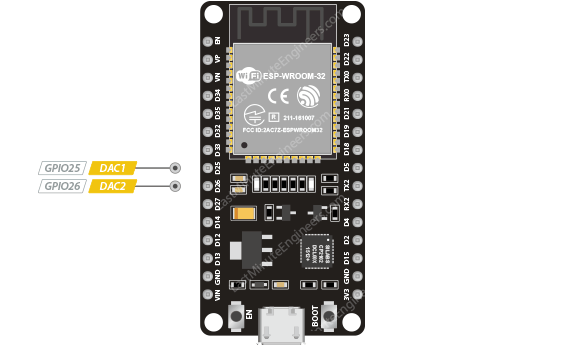


The ADC on the ESP32 is a 12-bit ADC meaning it has the ability to detect 4096 (2^12) discrete analog levels. In other words, it will map input voltages between 0 and the operating voltage 3.3V into integer values between 0 and 4095. For example, this yields a resolution between readings of: 3.3 volts / 4096 units or, 0.0008 volts (0.8 mV) per unit.

You also have the ability to set the ADC resolution and ADC range of your channels in code.

DAC Pins

The ESP32 features two 8-bit DAC channels that can be used to convert digital signals into true analog voltages. It can be used as a “digital potentiometer” to control analog devices.



This dual-DAC on the ESP32 has 8-bit resolution, which means that values between 0 and 256 will produce an analog voltage somewhere between 0 and 3.3V.