This page gives some information on how to get started using a ESP8266-based board. It is intended for the newbie with past experience in arduino-like mcu who wants to get a first hand at ESP8266.

Procuring a ESP8266 based board

Which ESP board?

ESP8266 usually comes assembled on a board, the most popular ones having been the **ESP-NN** by vendors such as *AIThinkers*.*Olimex* has also started to sell their own boards, and you can find several others too. The boards integrate an ESP8266 chip plus a Flash memory chip (Flash RAM), a crystal, and usually on on-board WiFi antenna. The Flash RAM size and chip model can vary.

Among the ESP-XX boards, the most notable to get started is the**ESP-01**, which has a male 2×4 0.1“ pitch connector that can be readily used with Dupont wires. This board exposes serial communication pins (RX/TX), and the minimal 4 control pins, GPIO0, GPIO2, CH_PD and ReSeT, plus VCC and GND of course. With this board you will have very limited access to all-purpose GPIOs. It also has an on-PCB Wifi Antenna etched on the PCB, which has sufficient gain for shorter range applications. Note that the ESP-01 connector on its own cannot straddle the two sides of a standard breadboard.

The other ESP-NN modules expose more GPIO pins, but are packaged in SMT packages, at best with a 2mm-pitch notches. Some modules have etched antenna, some have ceramic antenna, and some have just a socket or a pin. A favorite for that is the ESP-07, which has a shield (with FCC marking, but probably not registered at FCC)

Where to get ESP modules from?

Modules can be found in several places, just Google for ESP8266 or ESP-01. eBay has a number of sellers, and some are found on AliExpress or the big name DIY electronics vendors in US or Europe. Prices vary in the $3-$5 range for boards imported from China and slightly more if sourced from an in-country supplier. European vendors such as Olimex offer modules too.

Vendors such as nodeMCU also offer integrated development boards that are breadboard-friendly, include USB connectivity onboard and present a reasonable number of I/O pins to the connector. Their version 2 board is based on the ESP-12E and is available on eBay etc. for around $12-15.

If you are using a bare ESP-XX module then you will also need to connect to the 3.3V TTL serial port during development. There are many USB to serial converter modules available which present a virtual serial connection to your host PC or laptop and connect to the chip's serial interface. These are based on either the FT232, CH340G or CP2102 chip. Note:

- The FT232 is made by FTDI, but there are a number of unlicensed FT232 clones available and some modules use these. These FT232 clones can cause problems on Windows PCs because the FTDI-supplied FT232 driver now includes detection logic and will not communicate to unlicensed clones.
- The modules based on the CH340G or CP2102 chips do not have this issue.
- How must by a serial module what can drive TTL at 3.3V levels. Many are 3.3V/5V jumpered or DIP-switchable.
- Some modules also provide a 3.3V VCC line but these are typically at 300mA or so and cannot provide enough power to run the ESP chip during WifI operation. So a separate 3.3V power supply is needed. A common approach is to use a 5.5V to 3V power module, and again there are many options available online for a few $.

Got an ESP-NN board in the mail, now what ?

Caution: The first and most important thing to note – especially for Arduino developers – is that ESP8266 chip only uses **3.3V** for power and I/O. If you attempt to connect directly to 5V devices then at best this may simply not work and at worst this will cause permanent or fatal damage to the chip.

When getting to know and use this device, the best approach is to use a USB to 3.3V TTL converter and a 3.3v power supply. Avoid working with any devices that require or output 5V levels. Get everything working and talking first.

Once you have done this then consider how to connect any 5V devices and interfaces if you must. You will need to implement level shifting to connect to 5V devices. There are plenty of resources on the internet (for example (1), (2)) that will help you do this, and also low cost modules such as by-direction level shifters that are available for a few $, Also be aware of the difference between unbuffered interfaces (such as I2C) and buffered ones (such as SPI) as the level shifting apporach is different for these.

Also note that if you use an external power supply, then be sure to connect its ground to your USB TTL's ground, otherwise it may have trouble.

Basic wiring

So, let's assume that you have:

1. an ESP-01
2. a USB/TTL converter
3. a 3.3V power supply
4. a breadboard and Dupont wires
5. 1k and 2.2K resistors

Note that some USB UART adapters internal 3.3v supply are rated at 1A and can therefore provide enough current to power the ESP8266, but in general an external 3.3v power supply is recommended.

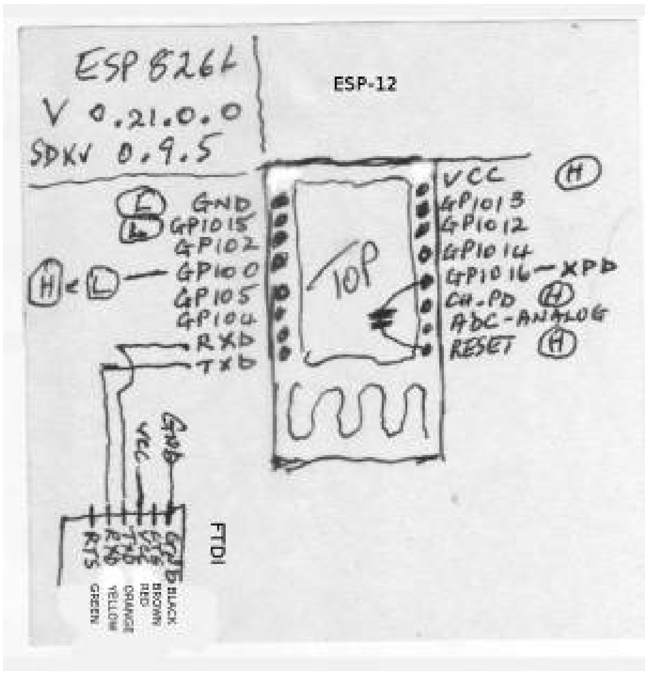
Looking at the (8 pin) board from the top (chips and antenna side), with the antenna to the right, the pins from the top down in the edge row are:

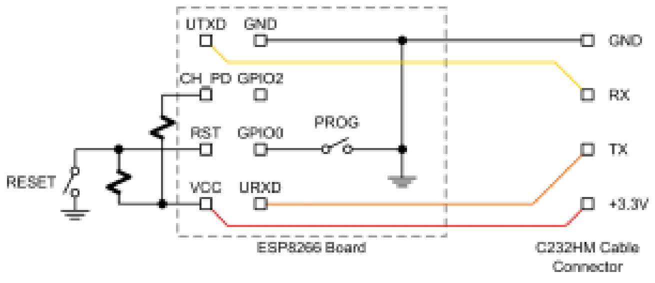
- TXD (goes to the 3.3V RxI of the UART USB adapter to the PC)
- CH_PD (enable/power down, must be pulled to 3.3v directly or via resistor)
- REST (reset, must be pulled to 3.3v)
- VCC (3.3v power supply)

The inner row (top and bottom have square pads)

- GND (connect to power ground)
- GPIO 2
- GPIO 0 (leave open or pull up for normal, pull down to upload new firmware)
- RXD (goes to the 3.3V TxO of the UART USB adapter to the PC)

Other modules have more pins, but these will be included. If GPIO 15 is broken out, then it must be pulled low at power-on or after reset to get out of the boot-loader.





If there is power, The red LED will come on, and the blue LED will flash briefly as it comes out of reset – it flashes when the UART is transmitting.

If you have problems with the module occasionally resetting once you've attached other devices then this might be a voltage regulation issue so try adding a 220 µF capacitor across Vcc and Gnd.

Talk to me, baby

Once setup, you will want to start talking to the ESP. For that, a terminal emulator on windows will do. Tera Term, Putty, Hyperterminal or nearly anything else will work. One caution is if you detach your UART device or it goes offline (e.g. loses power or the ESP causes the chip to reset), it might not clear and reset until you exit or close or restart the program. On Linux and OSX, the standard screen command does this job.

First, your board might talk at any of several baud rates. The ones to try first are 9600 and 115200. Then 57600 and/or 76800 (38400 * 2). Note the noise when you reset the device (pull the RST pin to ground) is typically some bootup messages at 76800. But there should be a ”`ready`“ message at the selected baud rate if your UART Rx is wired correctly.

If your UART Tx rate is correct, then try typing ”`AT`“ followed by Ctrl-M and Ctrl-J (*both carriage return and linefeed are needed*). This should return an ”`OK`“. And you are now ready to start working with the ESP8266. It should echo the characters as you type them.

Boot up

(Note that by “**HI**GH”, this means pulled up to **Vcc**. (This can be directly connected to 3.3v power but a

1k pull-up resistor is normally recommended.) **L**OW” means the equivalent pulled down to **Gnd**.

If one of the boot-mode pins is not set correctly, when you reset, you won't get the ready message. To diagnose this, you can see what the bootloader is putting out by setting the baud rate to 76800. Then you can pull the RST pin **L**ow, then **H**igh, and see what the output is.

It will display a message giving the reason for the reset. It should give a loading message and some addresses if everything is correct. If it doesn't give any more messages, or “Waiting for host”, or something else, one of the other pins is not connected properly.

The CH_PD must be **H**IGH, as well as the RST line and GPIO 2 (or left floating, i.e. unconnected)

GPIO 0 is interrogated by the bootrom loader and if this is **L**OW then it enters flashing new firmware model; if **H**IGH then it transfers to normal execution mode,

For the modules with extra GPIOs (more than the 8 pin), GPIO 15 must be pulled low.

If everything here is right, you should get a bootloader loading message, then nothing or some noise as the baud rate changes to normal.

AT Commands

The first AT command you will want to try is:
`AT+GMR`
it will output the firmware revision number similar to:
`0018000902-AI03`.
`000902` means firmware version 0.9.2, `0018` is the version level of the AT command support.

Then, you may want to list the **A**ccess **P**oints that are visible from your ESP module:

- Set Wifi mode to both **A**ccess **P**oint and **STA**tion:
  `AT+CWMODE=3`
- **L**ist **A**ccess **P**oints:
  `AT+CWLAP`

And connect to one of the listed APs:

- **J**oin an **A**ccess **P**oint:
  `AT+CWJAP=“SSID”,“password”`