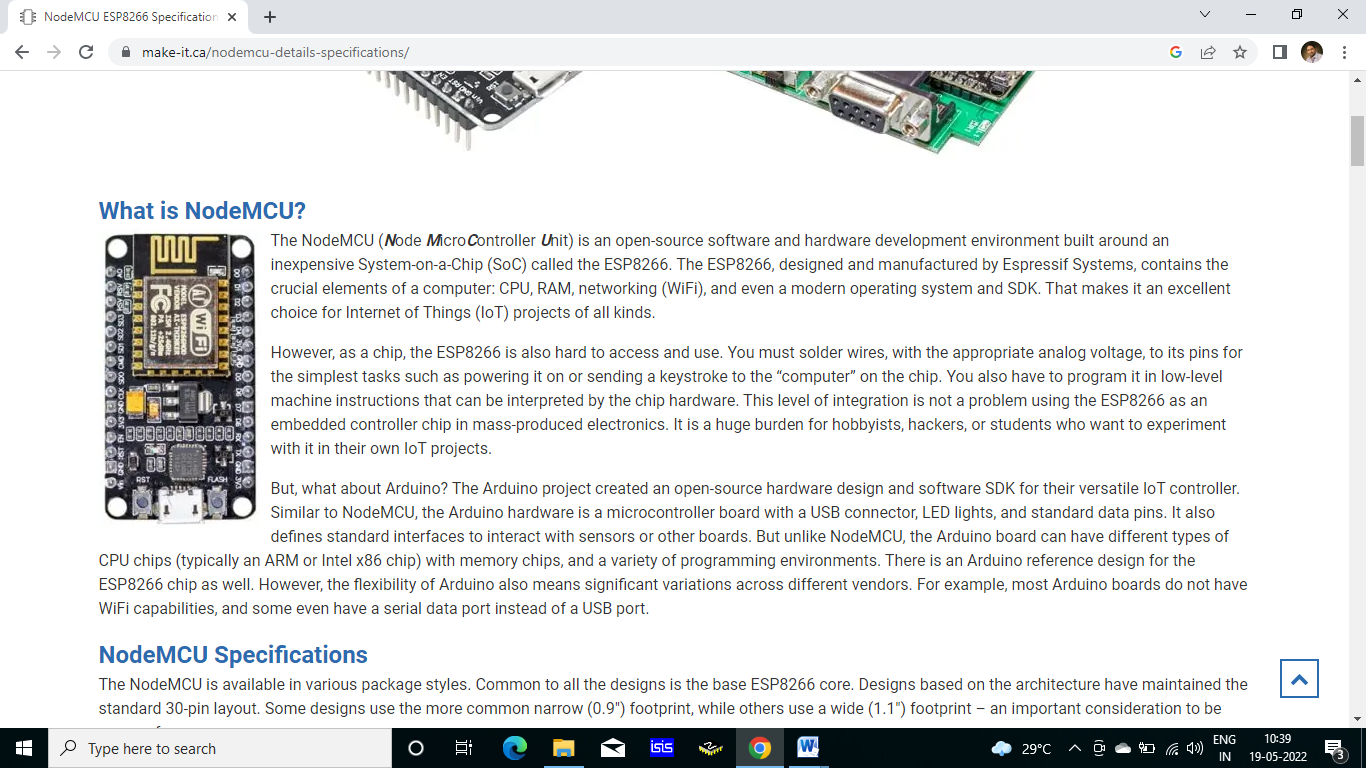
**NodeMCU (ESP8266-12E)**

The NodeMCU (***N***ode ***M***icro***C***ontroller ***U***nit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WiFi), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds.



However, as a chip, the ESP8266 is also hard to access and use. You must solder wires, with the appropriate analog voltage, to its pins for the simplest tasks such as powering it on or sending a keystroke to the “computer” on the chip. You also have to program it in low-level machine instructions that can be interpreted by the chip hardware. This level of integration is not a problem using the ESP8266 as an embedded controller chip in mass-produced electronics. It is a huge burden for hobbyists, hackers, or students who want to experiment with it in their own IoT projects.

But, what about Arduino? The Arduino project created an open-source hardware design and software SDK for their versatile IoT controller. Similar to NodeMCU, the Arduino hardware is a microcontroller board with a USB connector, LED lights, and standard data pins. It also defines standard interfaces to interact with sensors or other boards. But unlike NodeMCU, the Arduino board can have different types of CPU chips (typically an ARM or Intel x86 chip) with memory chips, and a variety of programming environments. There is an Arduino reference design for the ESP8266 chip as well. However, the flexibility of Arduino also means significant variations across different vendors. For example, most Arduino boards do not have WiFi capabilities, and some even have a serial data port instead of a USB port.

**NodeMCU Specifications**

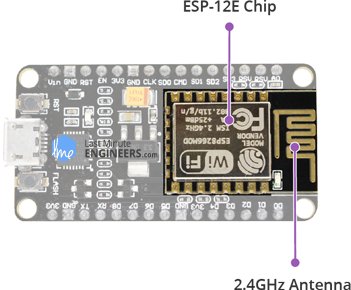
The NodeMCU is available in various package styles. Common to all the designs is the base ESP8266 core. Designs based on the architecture have maintained the standard 30-pin layout. Some designs use the more common narrow (0.9″) footprint, while others use a wide (1.1″) footprint – an important consideration to be aware of.

The most common models of the NodeMCU are the Amica (based on the standard narrow pin-spacing) and the LoLin which has the wider pin spacing and larger board. The open-source design of the base ESP8266 enables the market to design new variants of the NodeMCU continually.

The Internet of Things (IoT) has been a trending field in the world of technology. It has changed the way we work. Physical objects and the digital world are connected now more than ever. Keeping this in mind, Espressif Systems (A Shanghai-based Semiconductor Company) has released an adorable, bite-sized Wi-Fi enabled microcontroller – ESP8266, at an unbelievable price! For less than $3, it can monitor and control things from anywhere in the world – perfect for just about any IoT project.

ESP-12E Module

The development board equips the ESP-12E module containing ESP8266 chip having Tensilica Xtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS.



There’s also 128 KB RAM and 4MB of Flash memory (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays.

The ESP8266 Integrates 802.11b/g/n HT40 Wi-Fi transceiver, so it can not only connect to a WiFi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. This makes the ESP8266 NodeMCU even more versatile.

**Technical Specification of NodeMCU (ESP8266-12E)**

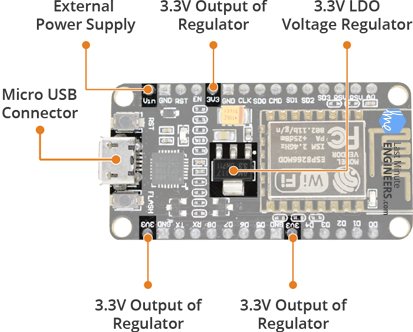
|  |  |
| --- | --- |
| Microcontroller | Esp8266 – 32 bit |
| Flash memory (ROM) | 4MB |
| RAM | 64Kb |
| Wi-Fi Built-In | 802.11 b/g/n |
| Digital I/O pins | 11 |
| Analog Input Pins | 1 |
| UART | 1 |
| SPI | 1 |
| I2C | 1 |
| Operating voltage | 3.3v |
| Input Voltage | 4.5 – 10V |
| ADC voltage Range | 0 -3.3V |
| USB Connector | microUSB |

**Power Requirement**

As the operating voltage range of ESP8266 is 3V to 3.6V, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as 80mA during RF transmissions. The output of the regulator is also broken out to one of the sides of the board and labeled as 3V3. This pin can be used to supply power to external components.

**Power Requirement**

* Operating Voltage: 2.5V to 3.6V
* On-board 3.3V 600mA regulator
* 80mA Operating Current
* 20 µA during Sleep Mode



Power to the ESP8266 NodeMCU is supplied via the on-board MicroB USB connector. Alternatively, if you have a regulated 5V voltage source, the VIN pin can be used to directly supply the ESP8266 and its peripherals.

**Warning:**

The ESP8266 requires a 3.3V power supply and 3.3V logic levels for communication. The GPIO pins are not 5V-tolerant! If you want to interface the board with 5V (or higher) components, you’ll need to do some level shifting.

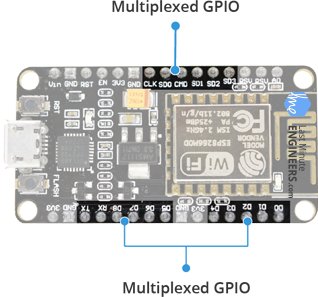
**Peripherals and I/O**

The ESP8266 NodeMCU has total 17 GPIO pins 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:

* ADC channel – A 10-bit ADC channel.
* UART interface – UART interface is used to load code serially.
* PWM outputs – PWM pins for dimming LEDs or controlling motors.
* SPI, I2C & I2S interface – SPI and I2C interface to hook up all sorts of sensors and peripherals.
* I2S interface – I2S interface if you want to add sound to your project.

Multiplexed I/Os

1. ADC channels
2. 2 UART interfaces
3. PWM outputs
4. SPI, I2C & I2S interface



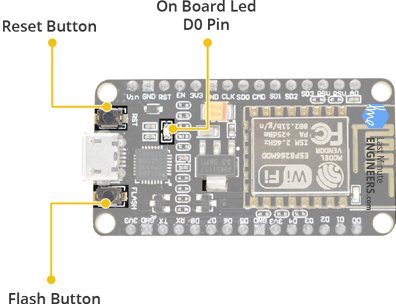
The ESP8266’s pin multiplexing feature (Multiple peripherals multiplexed on a single GPIO pin). Meaning a single GPIO pin can act as PWM/UART/SPI.

**On-board Switches & LED Indicator**

The ESP8266 NodeMCU features two buttons. One marked as RST located on the top left corner is the Reset button, used of course to reset the ESP8266 chip. The other FLASH button on the bottom left corner is the download button used while upgrading firmware

**Switches & Indicators**

* RST – Reset the ESP8266 chip
* FLASH – Download new programs
* Blue LED – User Programmable



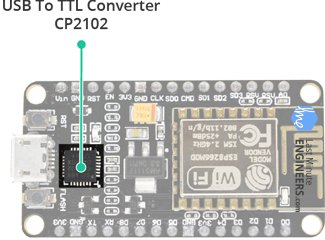
The board also has a LED indicator which is user programmable and is connected to the D0 pin of the board.

**Serial Communication**

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.

Serial Communication

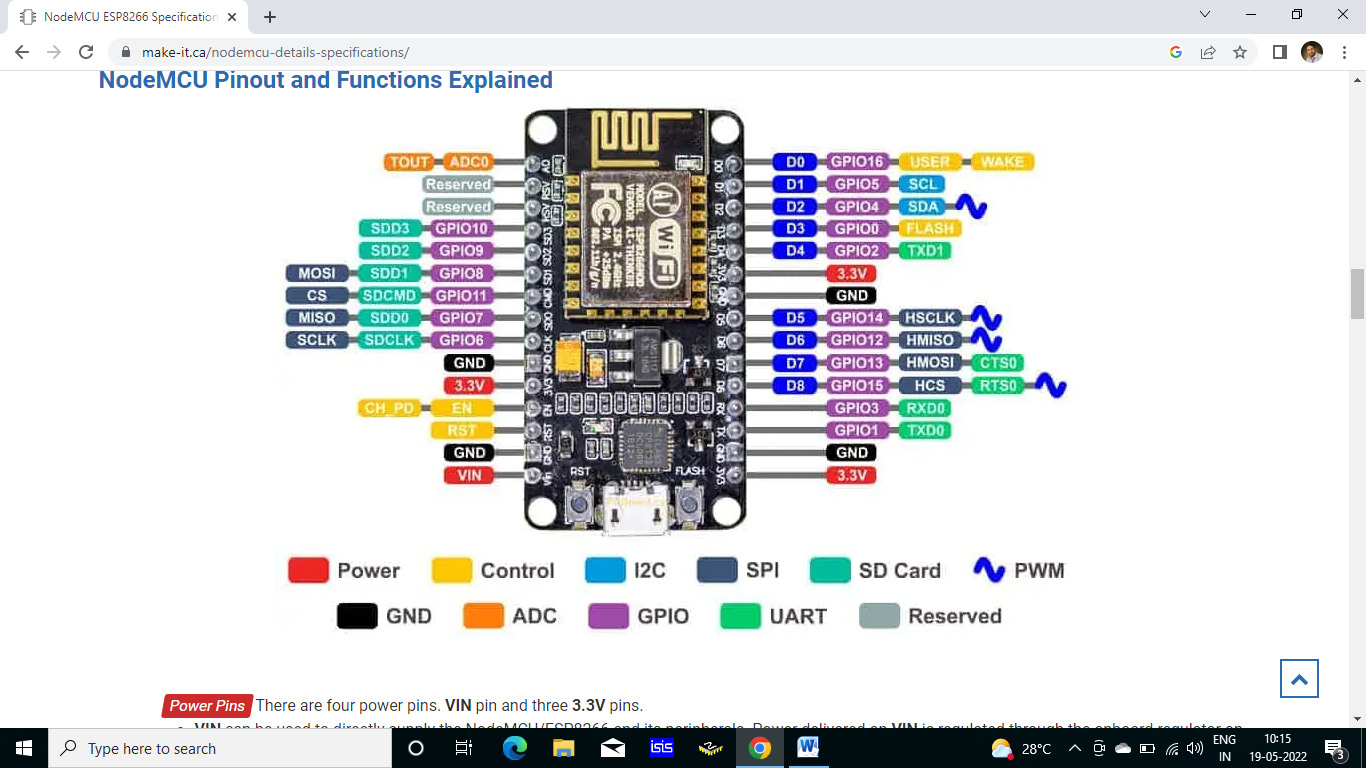
* CP2102 USB-to-UART converter
* 4.5Mbps communication speed
* Flow Control support



If you have an older version of CP2102 driver installed on your PC, we recommend upgrading now.

**ESP8266 NodeMCU Pinout**

The ESP8266 NodeMCU has total 30 pins that interface it to the outside world. The connections are as follows:



Power Pins There are four power pins. **VIN** pin and three **3.3V** pins.

* + - **VIN** can be used to directly supply the NodeMCU/ESP8266 and its peripherals. Power delivered on **VIN** is regulated through the onboard regulator on the NodeMCU module – you can also supply 5V regulated to the **VIN** pin
    - **3.3V** pins are the output of the onboard voltage regulator and can be used to supply power to external components.

GND are the ground pins of NodeMCU/ESP8266

I2C Pins are used to connect I2C sensors and peripherals. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins NodeMCU/ESP8266 has 17 GPIO pins which can be assigned to functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins NodeMCU/ESP8266 has 2 UART interfaces (UART0 and UART1) which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins NodeMCU/ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

* + - 4 timing modes of the SPI format transfer
    - Up to 80 MHz and the divided clocks of 80 MHz
    - Up to 64-Byte FIFO

SDIO Pins NodeMCU/ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs (100 Hz and 1 kHz).

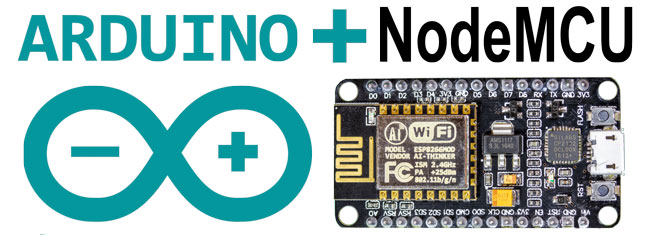
Control Pins are used to control the NodeMCU/ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

* + - **EN:** The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
    - **RST:** RST pin is used to reset the ESP8266 chip.
    - **WAKE:** Wake pin is used to wake the chip from deep-sleep.

Tiny Sine WaveControl Pins are used to control the NodeMCU/ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

* + - **EN:** The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
    - **RST:** RST pin is used to reset the ESP8266 chip.
    - **WAKE:** Wake pin is used to wake the chip from deep-sleep.

## NodeMCU Compatibility with Arduino IDE



The NodeMCU offers a variety of development environments, including compatibility with the Arduino IDE (Integrated Development Environment). The NodeMCU/ESP8266 community took the IDE selection a step further by creating an Arduino add-on. If you’re just getting started programming the ESP8266 or even an established developer, this is the highly recommended environment. Visit our dedicated page on setting up and configuring the Arduino IDE for a NodeMCU ESP8266.

**Installing the ESP8266 Core on Windows OS**

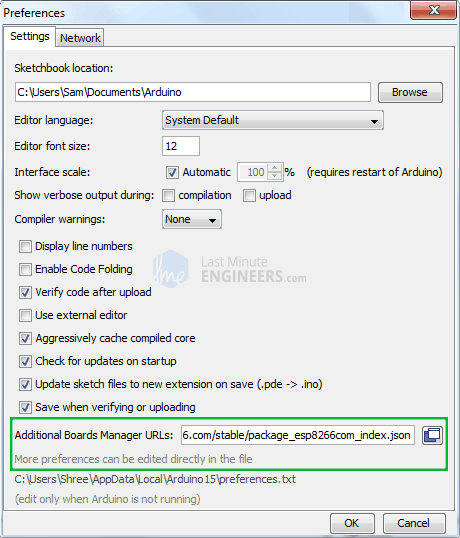
Let’s proceed with installing ESP8266 Arduino core.

The first thing is having latest Arduino IDE (Arduino 1.6.4 or higher) installed on your PC. If don’t have it, we recommend upgrading now.

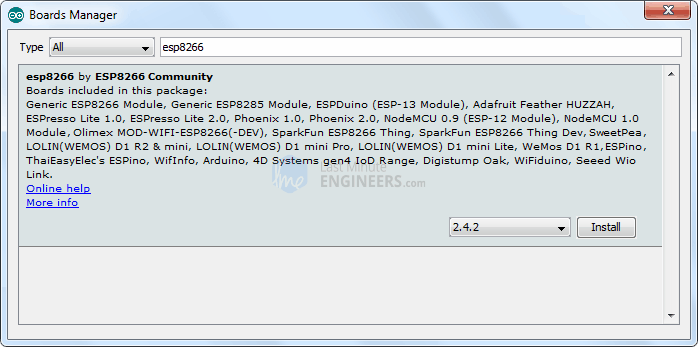
[Latest Arduino IDE](https://www.arduino.cc/en/Main/Software)

To begin, we’ll need to update the board manager with a custom URL. Open up Arduino IDE and go to File > Preferences. Then, copy below URL into the Additional Board Manager URLs text box situated on the bottom of the window:

http://arduino.esp8266.com/stable/package\_esp8266com\_index.json



Hit OK. Then navigate to the Board Manager by going to Tools > Boards > Boards Manager. There should be a couple new entries in addition to the standard Arduino boards. Filter your search by typing esp8266. Click on that entry and select Install.



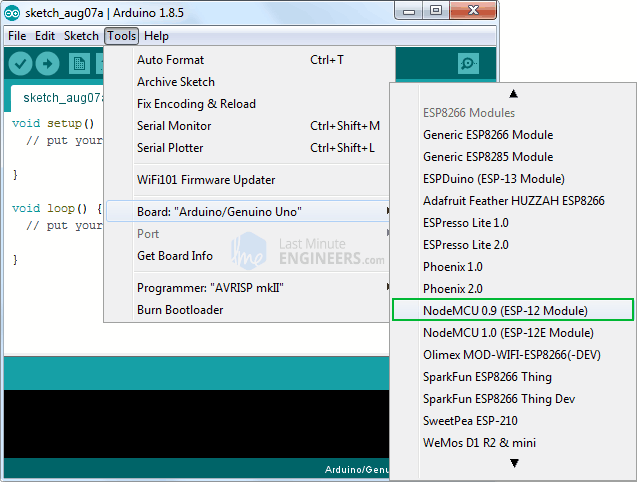
The board definitions and tools for the ESP8266 include a whole new set of gcc, g++, and other reasonably large, compiled binaries, so it may take a few minutes to download and install (the archived file is ~110MB). Once the installation has completed, a small *INSTALLED* text will appear next to the entry. You can now close the Board Manager.

**Arduino Example: Blink**

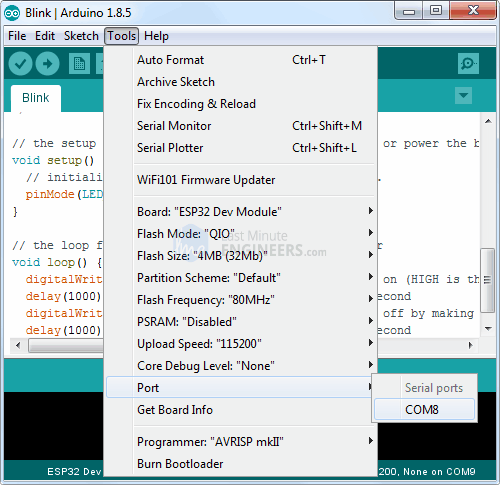
To make sure ESP8266 Arduino core and the NodeMCU are properly set up, we’ll upload the simplest sketch of all – The Blink!

We will use the on-board LED for this test. As mentioned earlier in this tutorial, D0 pin of the board is connected to on-board Blue LED & is user programmable. Perfect!

Before we get to uploading sketch & playing with LED, we need to make sure that the board is selected properly in Arduino IDE. Open Arduino IDE and select NodeMCU 0.9 (ESP-12 Module) option under your Arduino IDE > Tools > Board menu.



Now, plug your ESP8266 NodeMCU into your computer via micro-B USB cable. Once the board is plugged in, it should be assigned a unique COM port. On Windows machines, this will be something like COM#, and on Mac/Linux computers it will come in the form of /dev/tty.usbserial-XXXXXX. Select this serial port under the Arduino IDE > Tools > Port menu. Also select the Upload Speed : 115200



**Warning:**

More attention needs to be given to selecting board, choosing COM port and selecting Upload speed. You may get espcomm\_upload\_mem error while uploading new sketches, if failed to do so.

Once you are done, try the example sketch below.

void setup()

{

pinMode(D0, OUTPUT);

}

void loop()

{

digitalWrite(D0, HIGH);

delay(500);

digitalWrite(D0, LOW);

delay(500);

}

Once the code is uploaded, LED will start blinking. You may need to tap the RST button to get your ESP8266 to begin running the sketch.