Writing an wisdom ESP32 operating algorithm.

## Step 1: Key Features

Chip with built-in WiFi: standard 802.11 B / G / N, operating in the range of 2.4 to 2.5GHz

Modes of operation: Client, Access Point, Station + Access Point

Dual core microprocessor Tensilica Xtensa 32-bit LX6

Adjustable clock from 80MHz up to 240MHz

Operating voltage: 3.3 VDC

It has SRAM of 512KB

Features 448KB ROM

It has external flash memory of 32Mb (4 megabytes)

Maximum current per pin is 12mA (it is recommended to use 6mA)

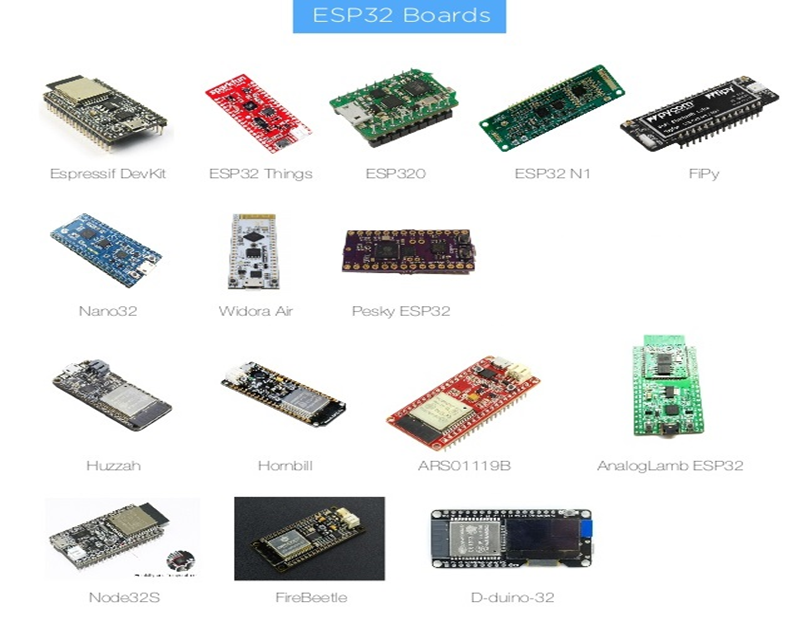
It has 36 GPIOs

GPIOs with PWM / I2C and SPI functions

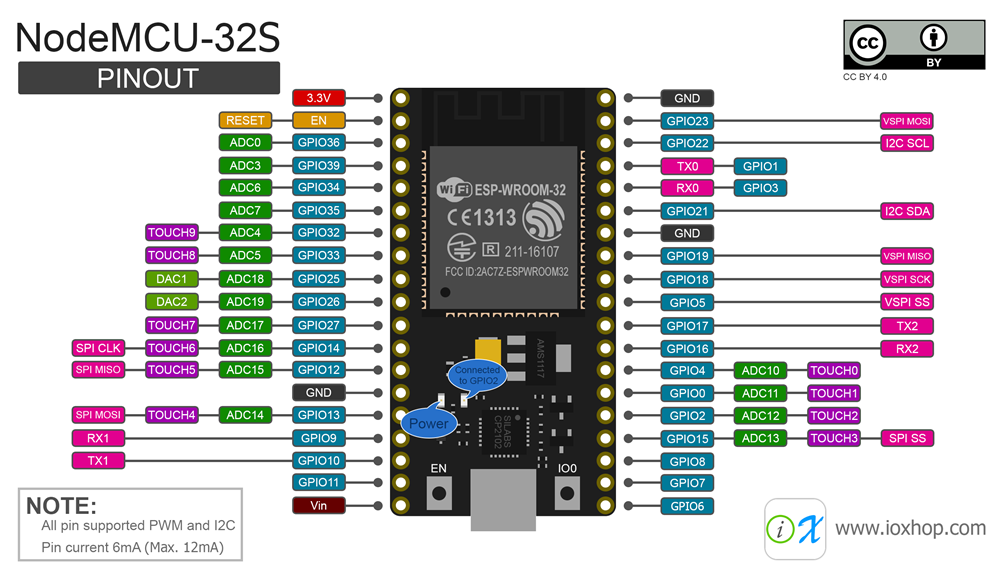
It has Bluetooth v4.2 BR / EDR and BLE (Bluetooth Low Energy)

## Step 2: Comparison Between ESP32, ESP8266 and Arduino R3

## Step 3: Types of ESP32

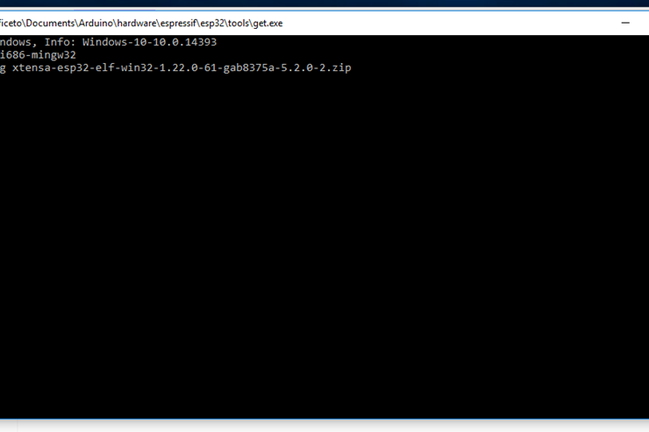


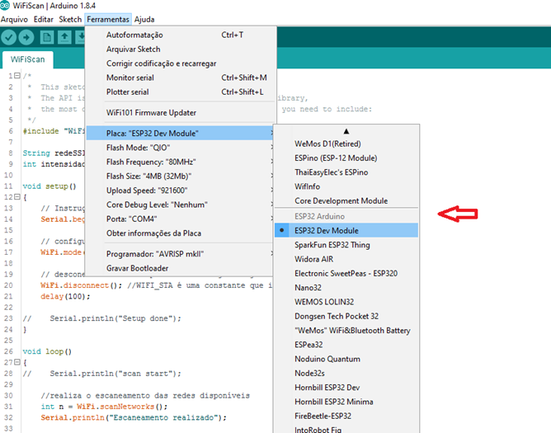
## Step 4: WiFi NodeMCU-32S ESP-WROOM-32



This is the diagram of ESP that we are using in our assembly. It is a chip that has lots of appeal and power. They are several pins you choose whether they want to work as digital analog, analog digital or even if that work the door as digital.

## Step 5: Configuring Arduino IDE (Windows)





## Step 6: WiFi Scan

Here's an example of how to look for available WiFi networks near the ESP-32, as well as the signal strength of each of them. With each scan, we will also find out which network has the best signal strength.

## Step 7: Code

First let's include the library "WiFi.h", it will be necessary to allow us to work with the network card of our device.

#include "WiFi.h"

Here are two variables that will be used to store the network's SSID (name) and signal strength.

String networkSSID = "";  
int strengthSignal = -9999;

## Step 8: Setup

In the setup () function, we will define the WiFi behavior mode of our device. In this case, since the goal is to search for available networks, we will configure our device to work as a "station".

void setup()  
{  
 // Initialize Serial to log in Serial Monitor  
 Serial.begin(115200);

// configuring the mode of operation of WiFi as station  
 WiFi.mode(WIFI\_STA);//WIFI\_STA is a constant indicating the station mode

// disconnect from the access point if it is already connected  
 WiFi.disconnect();   
 delay(100);

// Serial.println("Setup done");  
}

## Step 9: Loop

In the loop () function, we will search for the available networks and then print the log in the found networks. For each of these networks we will make the comparison to find the one with the highest signal strength.

void loop()  
{  
// Serial.println("scan start");  
  
 // performs the scanning of available networks

int n = WiFi.scanNetworks();  
 Serial.println("Scan performed");

//check if you have found any network   
 if (n == 0) {  
 Serial.println("No network found");  
 } else {  
 networkSSID = "";  
 strengthSignal= -9999;  
 Serial.print(n);  
 Serial.println(" networks found\n");  
 for (int i = 0; i < n; ++i) {  
 //print on serial monitor each of the networks found  
 Serial.print("SSID: ");  
 Serial.println(WiFi.SSID(i)); //network name (ssid)  
 Serial.print("SIGNAL: ");  
 Serial.print(WiFi.RSSI(i)); //signal strength  
 Serial.print("\t\tCHANNEL: ");  
 Serial.print((int)WiFi.channel(i));  
 Serial.print("\t\tMAC: ");  
 Serial.print(WiFi.BSSIDstr(i));  
 Serial.println("\n\n");  
   
   
 if(abs(WiFi.RSSI(i)) < abs(strengthSignal))  
 {  
 strengthSignal = WiFi.RSSI(i);  
 networkSSID = WiFi.SSID(i);  
 Serial.print("NETWORK WITH THE BEST SIGNAL FOUND: ( ");  
 Serial.print(networkSSID);  
 Serial.print(" ) - SIGNAL : ( ");  
 Serial.print(strengthSignal );  
 Serial.println(" )");  
 }   
   
 delay(10);  
 }  
 }  
 Serial.println("\n------------------------------------------------------------------------------------\n");

// interval of 5 seconds to perform a new scan  
 delay(5000);  
}

## Step 10: Files