### Introdução ao NodeMCU-32S ESP-WROOM-32





Por Fernando Koyanagi

### **Características principais**



- Chip com Wi-Fi embutido : padrão 802.11 B/G/N, operando na faixa de 2.4 a 2.5GHz
- Modos de operação : Client, Access Point, Station+Access Point
- ✓ Microprocessador dual core Tensilica Xtensa 32-bit LX6
- ✓ Clock ajustável de 80MHz até 240MHz
- ✓ Tensão de operação : 3.3 VDC
- ✓ Possui SRAM de 512KB
- ✓ Possui ROM de 448KB
- ✓ Possui memória flash externa de 32Mb (4 megabytes)
- Corrente máxima por pino é de 12mA (recomenda-se usar 6mA)
- ✓ Possui 36 GPIOs
- ✓ GPIOs com função PWM / I2C e SPI
- ✓ Possui Bluetooth v4.2 BR / EDR e BLE (Bluetooth Low Energy)



Em <u>www.fernandok.com</u>

Download arquivo PDF dos diagramas Download arquivo INO do código fonte



#### Comparativo entre ESP32, ESP8266 e Arduino R3

	ESP32	ESP8266	ARDUINO UNO R3
Cores	2	1	1
Arquitetura	32 bits	32 bits	8 bits
Clock	160MHz	80MHz	16MHz
WiFi	Sim	Sim	Não
Bluetooth	Sim	Não	Não
RAM	512KB	160KB	2KB
FLASH	16Mb	16Mb	32KB
GPIO	36	17	14
Interfaces	SPI / I2C / UART / I2S / CAN	SPI / I2C / UART / I2S	SPI / I2C / UART
ADC	18	1	6
DAC	2	0	0



### **Tipos de ESP32**





### WiFi NodeMCU-32S ESP-WROOM-32





### **Configurando IDE do Arduino** (windows)

# Vejamos a seguir como configurar a IDE do Arduino para podermos compilar para o ESP-32.

**1. Faça o download dos arquivos através do link** 

# 2. Descompacte o arquivo e copie o conteúdo para o seguinte caminho:

C:/Users/[YOUR\_USER\_NAME]/Documents/Arduino/hardware/espressif/esp32 Obs: caso não exista o diretório "espressif" e "esp32", basta criá-los normalmente.



### **Configurando IDE do Arduino** (windows)

#### 3. Abra o diretório

C:/Users/[YOUR\_USER\_NAME]/Documents/Arduino/hardware/espressif/esp32/tools

#### **Execute o arquivo "get.exe".**



### **Configurando IDE do Arduino** (windows)

4. Após a finalização do "get.exe", plugue o ESP32, aguarde os drivers serem instalados (ou instale manualmente).

Pronto, agora basta escolher a placa do ESP32 em "ferramentas >> placa" e

compilar sou código	🥺 WiFiScan   Arduino 1.8.4				
compilar seu courgoi	Arquivo Editar Sketch Fer	rramentas Ajuda			
		Autoformatação	Ctrl+T		
		Arquivar Sketch			
	WiFiScan	Corrigir codificação e recarregar			
	1 🗆 /*	Monitor serial	Ctrl+Shift+M		
	2 * This sketc	Plotter serial	Ctrl+Shift+L	-	
	3 * Ine API 18 4 * the most o 5 */	WiFi101 Firmware Updater		prary, you need to include:	
	6 #include "WiFi	Placa: "ESP32 Dev Module"	3	<b>▲</b>	
	7	Flash Mode: "QIO"	2	WeMos D1(Retired)	
	9 int intensidad	Flash Frequency: "80MHz"	2	ESPino (ESP-12 Module)	
	10	Flash Size: "4MB (32Mb)"	2	ThaiEasyElec's ESPino	
	11 void setup()	Upload Speed: "921600"	2	WifInfo	
	12 E {	Core Debug Level: "Nenhum"	2	Core Development Module	
	14 Serial.beg	Porta: "COM4"	2	ESP32 Arduino	
	15	Obter informações da Placa		ESP32 Dev Module	
	16 // configu			SparkFun ESP32 Thing	
	17 Wiri.mode	Programador: "AVRISP mkll"	1	Widora AIR	
	19 // descone	Gravar Bootloader		Electronic SweetPeas - ESP320	
	20 WiFi.disconne	ect(); //WIFI_STA é uma con	stante que i	Nano32	
	21 delay(100);			WEMOS LOLIN32	
	<pre>22 23 // Serial.println("Setup done"); 24 } 25 26 void loop() 27 27 27 27 27 27 27 27 27 27 27 27 27</pre>			Dongsen Tech Pocket 32	
				"WeMos" WiFi&Bluetooth Battery	
				ESPea32	
				Noduino Quantum	
	28 // Serial.prin	ntln("scan start");		Node32s	
	29			Hornbill ESP32 Dev	
	30 //realiza o escaneamento das redes disponíveis		oníveis	Hornbill ESP32 Minima	
	31 Int n = WiFi 32 Serial print	.scannetworks(); ln("Escaneamento realizado"		FireBeetle-ESP32	
	33			IntoRobot Fig	



### WiFi Scan

Vejamos a seguir um exemplo de como buscar as redes WiFi disponíveis próximas ao ESP-32, assim como a intensidade do sinal de cada uma delas. A cada escaneamento, também vamos descobrir qual a rede com a melhor intensidade de sinal.



# Vamos ao código

Primeiramente vamos incluir a biblioteca "WiFi.h", ela será necessária para nos permitir trabalhar com a placa de rede do nosso dispositivo.

#include "WiFi.h"

A seguir, vamos declarar duas variáveis que serão utilizadas para guardar o SSID (nome) da rede e a intensidade do sinal.



# Setup

Na função *setup()*, definiremos o modo de comportamento WiFi do nosso dispositivo. Nesse caso, como o objetivo é procurar por redes disponíveis, vamos configurar nosso dispositivo para trabalhar como **"estação"**.

```
void setup()
   // Instrução para inicializar o Serial, utilizaremos apenas para log no monitor.
   Serial.begin(115200);
   // configurando o modo de operação do WiFi como estação
   WiFi.mode(WIFI_STA); //WIFI_STA é uma constante que indica o modo estação
   // desconecta do access point caso ele já esteja conectado
   WiFi.disconnect();
```





Na função *loop()*, vamos fazer a busca pelas redes disponíveis e em seguida imprimir no log as redes encontradas. Para cada uma dessas redes faremos a comparação para encontrar a com maior intensidade de sinal.

```
void loop()
  //realiza o escaneamento das redes disponíveis
  int n = WiFi.scanNetworks();
  Serial.println("Escaneamento realizado");
  //verifica se encontrou alguma rede
  if (n == 0) {
    Serial.println("Nenhuma rede encontrada");
  else {
```

Loop

#### Continuando a função loop

. . .

```
void loop()
  . . .
  else {
      redeSSID = "";
      intensidadeSinal= -9999;
      Serial.print(n);
      Serial.println(" redes encontradas");
      for (int i = 0; i < n; ++i) {</pre>
       // imprime no log cada uma das redes encontradas
         Serial.print(WiFi.SSID(i)); //nome da rede
         Serial.print(": ");
         Serial.print(WiFi.RSSI(i)); //intensidade do sinal
```





```
void loop()
   ...
  else {
       . . .
      // faz a comparação para saber se a rede encontrada tem melhor sinal do que a melhor
rede até o momento.
       if(abs(WiFi.RSSI(i)) < abs(intensidadeSinal))</pre>
          intensidadeSinal = WiFi.RSSI(i);
          redeSSID = WiFi.SSID(i);
          Serial.print("REDE COM MELHOR SINAL ENCONTRADA: ");
          Serial.print(redeSSID);
          Serial.print(" - SINAL : ");
          Serial.println(intensidadeSinal );
     }//for
```



delay(5000); // deixa um intervalo de 5 segundos para fazer um novo escaneamento

Loop

#### "if(abs(WiFi.RSSI(i)) < abs(intensidadeSinal))"

Repare que na instrução acima, utilizamos *abs()*, essa função pega o valor absoluto (ou seja, não negativo) do número. No nosso caso fizemos isso para achar o menor entre os valores na comparação, pois, a intensidade do sinal é dada como um número negativo, e quanto mais próxima de zero, melhor o sinal.



Link dos arquivos de configuração https://github.com/espressif/arduino-esp32

