Introdução

Por meio deste documento venho relatar algumas coisas que foram “descobertas” sobre a programação e as funcionalidades do ESP8266 NODEMCU ESP 12-E, essa exploração das funcionalidades do esp8266 ocorreu em sua maior parte por meio de documentos disponibilizados no github ([ESP8266 Community Forum · GitHub](https://github.com/esp8266)).

Para facilitar a exploração da placa foram propostas algumas funcionalidades para servirem de objetivo sendo elas : instalação e configuração de uma plataforma para utilização do esp, utilização das portas analógicas e digitais, Serial, Módulo de WiFi, Sistema de arquivos e criação de Web Server.

Instalação e Configuração

A plataforma escolhida para a programação do ESP8266 foi a IDE Arduino na versão 1.8.5, a versão é importante devido algumas funcionalidades da placa.

Para utilizar o ESP8266 por meio da IDE Arduino, além da instalação normal da IDE e da seleção da porta, necessita que seja instalado um gerenciador da placa. Esse processo é feito da seguinte forma :

* Acesse a aba preferências e adicione o link do gerenciador da placa no campo “URLs Adicionais de Gerenciadores de placa”
* Acesse a aba de ferramentas -> Placa -> Gerenciador de Placas e instale o gerenciador do esp8266
* Por fim acesse ferramentas -> Placa e procure pelo modelo utilizado do esp8266

O código do gerenciador pode ser encontrado em [GitHub - esp8266/Arduino: ESP8266 core for Arduino](https://github.com/esp8266/Arduino).

OBS : O gerenciador utilizado nos testes aqui relatados não é a versão mais recente lançada, URL :

<https://github.com/esp8266/Arduino/releases/download/3.0.2/package_esp8266com_index.json>

Drivers caso o computador não reconheça o esp8266(VCP) :

[CP210x USB to UART Bridge VCP Drivers - Silicon Labs (silabs.com)](https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads)

Utilização dos pinos digitais e analógicos

A conexão do esp8266 com outros dispositivos pode ser feito por meio dos pinos digitais(GPIO) e analógicos(ADC) para a leitura de sinais vindos desses dispositivos ou para a emissão de sinais.

Antes de emitir sinais para os dispositivos é necessário definir a configuração do pino em específico utilizando o seguinte comando :

* pinMode(pino, modo);
* Por padrão o pinMode é INPUT

Os modos disponíveis são INPUT, OUTPUT e INPUT\_PULLUP, sendo definidos em termos básicos : INPUT para ler sinais e OUTPUT para emitir sinais. O modo INPUT\_PULLUP tem uma diferença pequena do INPUT, que não é necessário para utilizações básicas do esp8266.

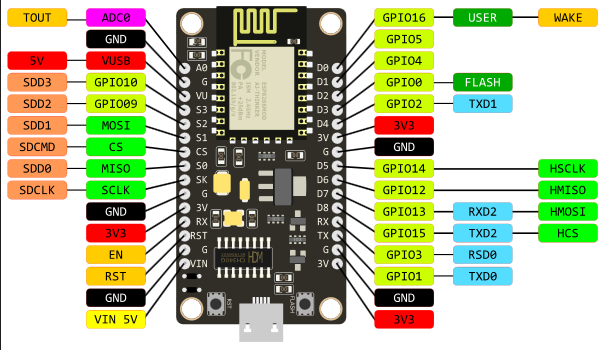
Após a configuração do pino, para emitir sinais :

* digitalWrite(pino, HIGH/LOW); para pinos digitais
* analogWrite(pino, valor); para pinos analógicos

Para ler sinais :

* digitalRead(pino); para pinos digitais
* analogRead(pino); para pinos analógicos

Pinagem do esp8266 :



Observações : Pinos digitais só suportam valores 0 ou 1(LOW ou HIGH) e pino analógicos suportam valores de 0 a 1023 para leitura e de 0 a 255 para emissão.

Serial

O monitor Serial é uma ferramenta da IDE Arduino que pode ser utilizada como um console, para monitorar o andamento do programa, enquanto a placa estiver conectada ao computador.

Para utilizar o serial é simples :

* Serial.begin(velocidade); para iniciar o serial, a velocidade deve coincidir com a velocidade presente no monitor serial
* Serial.print() ou Serial.println() para mostrar algo no monitor serial
* Para ver o monitor serial é só clicar em uma lupa que fica no canto superior direito da IDE Arduino

Documentação Serial : [Reference — ESP8266 Arduino Core 3.0.2 documentation (arduino-esp8266.readthedocs.io)](https://arduino-esp8266.readthedocs.io/en/3.0.2/reference.html#serial)

Módulo WiFi

Para utilizar funcionalidades relacionadas a rede WiFi foi utilizado a biblioteca ESP8266WiFi.h, essa biblioteca trata o esp8266 tendo dois modos/classes de utilização que podem ser configurados por meio do comando :

* WiFi.mode(WiFi\_STA/WiFi\_AP/WiFi\_AP\_STA); :

Modos de utilização do WiFi :

* Station(STA) : A classe/modo de utilização Station engloba comandos para se conectar/monitorar uma rede WiFi
* Access Point(AP): A classe/modo de utilização Access Point engloba comandos para estabelecer/hostear e configurar uma rede WiFi (Não foi possível se conectar a internet por parte dessas redes hosteadas pelo ESP8266)
* Os dois modos podem ser utilizados juntos

Conectando em uma rede WiFi :

Para conectar o esp8266 em uma rede WiFi basta executar o comando :

* WiFi.begin(ssid, senha);

Observação : Após conectado a uma rede WiFi o comando WiFi.setAutoReconnect(true) habilita a reconexão automática a essa rede.

Definindo IP, gateway e Subnet fixos

As configurações do usuário ao se conectar a rede WiFi podem ser dinâmicas (basta utilizar o comando acima) ou podem estáticas e definidas pelo usuário, para definir essas configurações estáticas basta seguir o exemplo:

IPAddress staticIP(192,168,1,22);

IPAddress gateway(192,168,1,1);

IPAddress subnet(255,255,255,0);

WiFi.config(staticIP, gateway, subnet)

Observações : Para definir o ip, gateway e máscara subnet são necessários que os dados sejam do tipo IPAddress como mostrado acima; O comando WiFi.config() retorna True caso a configuração seja aplicada com sucesso

Monitorando a conexão

Para monitorar a conexão do esp8266 em uma rede basta executar o comando WiFi.status(), o comando retorna :

* WL\_CONNECTED caso a conexão tenha sido bem sucedida
* WL\_NO\_SSID\_AVAIL caso a rede não seja encontrada
* WL\_CONNECT\_FAILED caso a senha fornecida seja incorreta
* WL\_IDLE\_STATUS caso o status da conexão esteja em transição
* WL\_DISCONNECTED caso o esp não esteja conectado em uma rede

Documentação Station Class : [Station Class — ESP8266 Arduino Core 3.0.2 documentation (arduino-esp8266.readthedocs.io)](https://arduino-esp8266.readthedocs.io/en/3.0.2/esp8266wifi/station-class.html)

Estabelecendo uma Rede WiFi :

Para estabelecer rede, pode ser feito definindo IP, gateway e Subnet ou não. O comando que estabelece uma rede WiFi é o seguinte:

* WiFi.softAP(configurações);
* As configurações são as seguintes : SSID(nome da rede), senha, canal, True para esconder a rede, máximo de conexões simultâneas. As configurações devem ser colocadas na ordem e separadas por vírgula

Observações : O SSID pode ter no máximo 32 caracteres e a senha pode ter de 8 a 64 caracteres.

Configurando a Rede :

Para configurar a rede basta executar o comando WiFi.softAPConfig(IP, gateway, Subnet); e vale lembrar que os parâmetros devem ser do tipo IPAddress

Documentação Access Point : [Soft Access Point Class — ESP8266 Arduino Core 3.1.0 documentation (arduino-esp8266.readthedocs.io)](https://arduino-esp8266.readthedocs.io/en/3.1.0/esp8266wifi/soft-access-point-class.html)

Sistema de Arquivos

O ESP8266 conta com um sistema de arquivos que varia de tamanho com o modelo da placa, para utilizá-lo podemos utilizar duas bibliotecas que funcionam de maneira parecida : SPPIFS.h ou LittleFS.h, a escolhida foi a LittleFS.h por receber atualizações(de acordo com a documentação mantida no github)

Instalando o Sistema de Arquivos

Para instalar o sistema de arquivos Little FS :

* Baixar a ferramenta no link : <https://github.com/earlephilhower/arduino-esp8266littlefs-plugin/releases>
* Ir até o sketchbook(Acesse a aba de preferências e veja o local do sketchbook), criar uma pasta tools e extrair a ferramenta lá dentro
* Reiniciar a IDE
* Antes de fazer upload de arquivos crie uma pasta chamada “data” dentro da pasta da sketch que você está codificando e coloque os arquivos que vão ser guardados no esp lá dentro.
* Para fazer upload dos arquivos vá em Ferramentas -> ESP8266 LittleFS Data Upload

Observações : Dentro da pasta data são necessárias outras pastas onde serão dispostos os arquivos, pois sem essas pastas o sistema não reconhece o caminho dos arquivos

Acessando e lendo Arquivos :

A forma mais segura de ler arquivos com LittleFS que foi encontrada é :

#include "LittleFS.h"

void setup() {

LittleFS.begin();

File f = LittleFS.open("/Path/leia.txt", "r");

char leitura;

while(f.available()){

leitura = f.read();

printf("%c", leitura);

}

}

Estabelecendo algumas funcionalidades importantes:

* LittleFS.begin() inicia o sistema de arquivos;
* LittleFS.end() fecha o sistema de arquivos;
* LittleFS.format() formata o sistema de arquivos;
* LittleFS.open(caminho, modo) abre um arquivo e retorna uma variável do tipo File
* LittleFS.exists(caminho) retorna True se o diretório/arquivo existe
* LittleFS.remove(caminho) deleta o arquivo especificado
* nomeArquivo.read() mostra um carácter de um arquivo
* nomeArquivo.available() retorna true caso o arquivo não tenha acabado/ caso o arquivo esteja disponível para leitura
* nomeArquivo.close() fecha o arquivo

Documentação Sistemas de Arquivos : [Filesystem — ESP8266 Arduino Core 3.0.2 documentation (arduino-esp8266.readthedocs.io)](https://arduino-esp8266.readthedocs.io/en/3.0.2/filesystem.html#file-system-object-spiffs-littlefs-sd-sdfs)

Observações : os modos disponíveis para utilizar no LittleFS.open() estão especificados na documentação

Web Server

Para criar Web Servers com o esp8266 antes de tudo é necessário que esteja conectado em uma rede com acesso a internet, e para criá-los será utilizado a biblioteca ESP8266WebServer.h.

Criando um Web Server simples

Para criar um Web Server simples utilizando a biblioteca ESP8266WebServer.h deve-se :

* Criar o objeto do servidor : ESP8266WebServer nomeServidor(porta);
* Definir duas funções : handleRoot() e handleNotFound();
* Iniciar o servidor com o comando nomeServidor.on(“/”, handleRoot();
* Definir o que vai acontecer caso seja requisitado algo inexistente com nomeServidor.onNotFound(handleNotFound());
* Acesse o Web Server pelo IP(para consultar execute WiFi.localIP());

Definindo funções do Web Server e construindo páginas HTML

Para criar a função handleRoot()ou handleNotFound() deve-se :

* Criar o que vai ser retornado na função
* Executar o comando nomeServidor.send(200, tipo, variavel);

Para criar uma página html basta colocar os textos e tags em uma variável do tipo string e executar o comando nomeServidor.send()

Executando funções pelo Web Server

Fugindo padrão de handleRoot(), podemos definir funções para serem requisitadas no servidorWeb e para isso :

* Crie uma função com as funcionalidades dentro dela
* Execute o comando nomeServidor.on(“/nomeQualquer”, nome função);
* Acesse o servidor pelo ip seguido por “/nomeQualquer”

Exibindo Páginas HTML armazenadas pelo Sistema de Arquivos

Para utilizar uma página HTML armazenada com LittleFS basta ler o arquivo e armazenar numa String e seguindo o procedimento padrão de estabelecimento de Web Server.

Considerações Finais

É válido salientar que essas poucas funcionalidades foram estabelecidas como objetivos iniciais e agora são necessários novos objetivos, e também vale salientar que nem todos os comandos relacionados às funcionalidades listadas foram citados e por isso foi deixado o link das documentações específicas.

Documentação Geral ESP8266 3.0.2 : [Welcome to ESP8266 Arduino Core’s documentation! — ESP8266 Arduino Core 3.0.2 documentation (arduino-esp8266.readthedocs.io)](https://arduino-esp8266.readthedocs.io/en/3.0.2/index.html)