Internet das Coisas

Prof. Vinícius Fernandes Soares Mota

Laboratório ESP8266

Este laboratório tem como objetivo oferecer uma experiência prática com um SOC, no caso o ESP8266. Para este fim, você usará:

1. Um ESP8266
2. Um sensor de temperatura BMP280
3. Arduino IDE → Instalar a biblioteca

O objetivo é aprender:

- Conceitos básicos de dispositivos embarcados
    - Portas analógicas, digitais e I2C;
- Prototipação rápida
- Protocolos de comunicação básicos
    - HTTP/REST
    - MQTT

Adicionalmente, discutiremos padronização formatos e protocolos para envio de dados

Texto puro (e os problemas que isso traz)

Ultralightweigh

JSON

Hardware

ESP8266, Sensor BMP280

Básico: Portas analógicas e digitais

*Texto adaptado de: Curso de Arduino, Alvaro Justen.

No geral, System On Chips de prototipação rápida possuem dois tipos de portas de entrada: analógicas e digitais. Além disso, as portas digitais também servem como portas de saída, funcionando com dois tipos básicos de saída: saída digital comum e saída PWM – o PWM pode ser utilizado para simular uma saída analógica, dentre outras coisas.

Portas digitais:

As portas digitais quando precisamos trabalhar com valores bem definidos de tensão. Como o sistema é binário, tem se apenas duas tensões: 0V e 5V. Dessa forma, as portas digitais trabalham apenas com essas duas tensões – e o software poderá requisitar ao microcontrolador que: Coloque uma determinada porta em 0V; Coloque uma determinada porta em 5V; Leia o valor de uma determinada porta (terá 0V ou 5V como resposta).

Portas analógicas:

Ao contrário das portas digitais, as portas analógicas são apenas de entrada e nelas podemos ter como entrada infinitos valores de tensão (delimitados na faixa de 0V a 5V). Os SOCs possuem os conversores analógico-digitais (ADC – analog-digital converter, do Inglês) com n bits de precisão. Por exemplo, os Arduino e ESP8266 possuem 10 bits de precisão, a precisão das medições de tensão no Arduino é de por volta de 0,005V ou 5mV. Isto quer dizer que uma porta analógica, ao ser lida, retorna um valor entre 0 e 1023.

PWM

PWM significa Modulação por Largura de Pulso (Pulse-Width Modulation, do Inglês) e consiste em manipularmos a razão cíclica de um sinal (conhecida do Inglês como duty cycle) a fim de transportar informação ou controlar a potência de algum outro circuito. Basicamente, teremos um sinal digital que oscila entre 0V e 5V com determinada frequência – funciona como se fosse um clock, porém os tempos em que o sinal permanece em 0V e 5V podem ser diferentes. Duty cycle é a razão do tempo em que o sinal permanece em 5V sobre o tempo total de uma oscilação.

As portas PWM então são capazes de oferecer que as portas digitais funcionem "semelhante" a uma porta analógica. No entanto, a precisão é menor e no ESP8266 e Arduino o valor é mapeado de 0 a 255, com o valor máximo equivalente a 3V no ESP e 5V no Arduino.

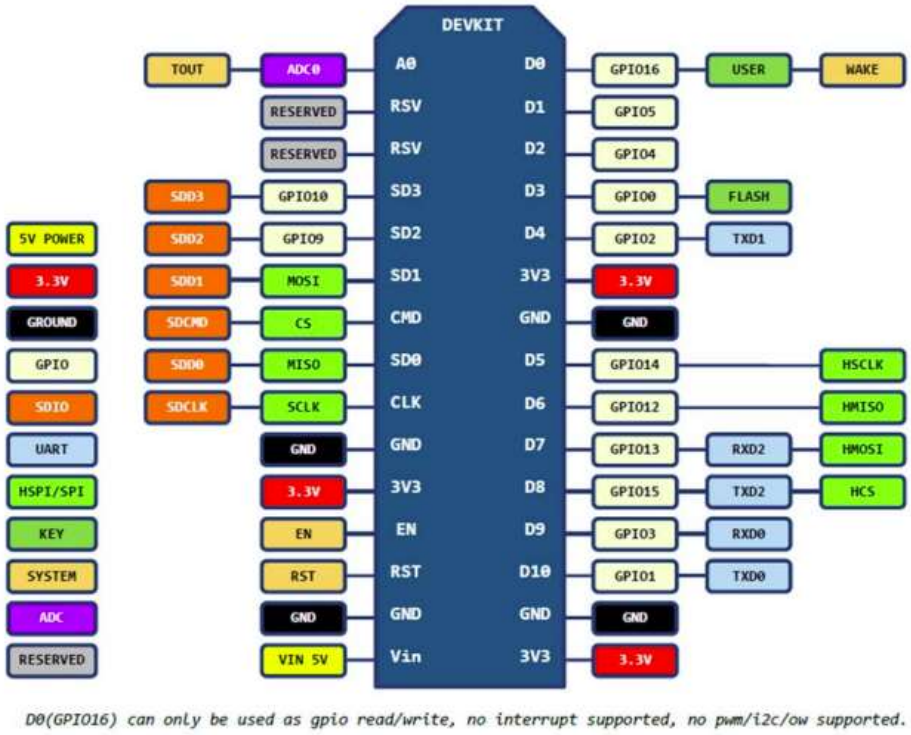
Básico: Datasheet ESP8266 NodeMCU

Microcontroladores da "família" ESP[1] possuem tudo que é necessário para se conectar a Internet. Por este motivo, ganharam o mercado de prototipação rápida para dispositivos embarcados IoT.

A fabricante possui o microcontrolador embarcado com CPU, rede sem fio, memória flash. Para facilitar a prototipação, diversos fabricantes produzem módulos que já exportam os pinos de entradas e saídas do ESP de maneira fácil. Neste laboratório, utilizaremos o Esp8266 Node MCU V3[2]. Com as seguintes especificações:

- Voltage:3.3V.
- Wi-Fi Direct (P2P), soft-AP.
- Current consumption: 10uA~170mA.
- Flash memory attachable: 16MB max (512K normal).
- Integrated TCP/IP protocol stack.
- Processor: Tensilica L106 32-bit.
- Processor speed: 80~160MHz. • RAM: 32K + 80K.

• GPIOs: 17 (multiplexed with other functions).
• Analog to Digital: 1 input with 1024 step resolution.
• +19.5dBm output power in 802.11b mode
• 802.11 support: b/g/n. • Maximum concurrent TCP connections: 5.



[1] https://espressif.com/

[2] https://www.theengineeringprojects.com/Downloads/Datasheet/NodeMCU-V3.pdf

Básico: Instalação Arduino IDE

Neste laboratório, utilizaremos o Arduino IDE. É necessário que instale o suporte ao ESP8266 no Arduino IDE

Entre http://arduino.esp8266.com/stable/package_esp8266com_index.json no campo *Additional Board Manager URLs* nas preferências do Arduino IDE v1.6.4+.

No Arduino IDE, deve-se selecionar que está usando o ESP8266 e os demais valores padrões devem funcionar. Em caso de dúvidas, seguir a seção 3 do datasheet do NodeMCU[2].

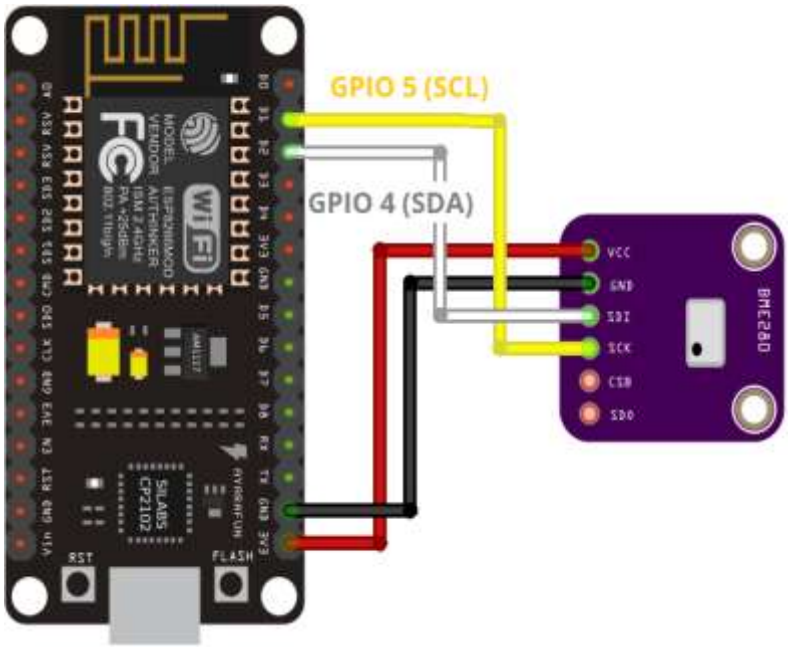
Rodar o “blink test” para ver se está tudo funcionando.

Obs: No Windows há um problema com drive que pode ser resolvido instalando o drier CH340 (https://www.blogdarobotica.com/2020/05/26/instalando-driver-serial-para-nodemcu-com-chip-ch340/)

1: Obter dados de temperatura e pressão

Set-up

Fazer as seguintes conexões:



Analisar o código disponível no AVA.

Rodar o código.

Temperatura e pressão via ESP8266

Vamos agora analisar o código do exemplo de server HTTP do ESP8266. Para isto, abra o código File → Examples → ESP8266WebServer → AdvancedWebserver

Observe que você deverá configurar uma rede sem fio. Use as informações do access point disponibilizada em aula (caso houver) ou seu celular como hotspot.

Acesse o endereço IP do ESP8266 do seu navegador (eventualmente será acessado apenas do seu celular). Como o servidor web configura um DNS, pode ser possível acessar via esp8266.local (Em uma máquina Windows, precisará instalar o Bounjour e, se estiver em uma máquina Linux, precisará instalar o Avahi).

1. Sua missão é apresentar a temperatura e pressão via HTTP.

2. Defina serviços que permitam obter cada uma das informações separadamente ou que todas juntas.

Para agrupar informações, existem diversos padrões como exemplificado abaixo. Fica a escolha do desenvolver utilizar o modo com mais verbosidade para dar semântica aos campos, útil em aplicações que compartilham dados e facilita o entendimento para quem recebe os dados, por outro lado, gastam mais bytes de transmissão.

Exemplos Formatos típicos:

CSV: temperatura, umidade, pressao ou com verbosidade: temp, valor, umi, valor, pressao, valor

Ultralight: Similar ao CSV mas usa pipe '|' como separador: temp|umidade|pressao

JSON: Formato que permite hierarquia entre os membros (https://json.org/)

Exemplo abaixo utiliza verbosidade

```
{
  "sensor": {
        "temp" : valor,
        "umidade" : valor,
        "pressao" : valor,
  }
}
```

XML. Just Don't use for IoT app, please....

3. Utilize a aplicação curl (linux) para obter as informações.

Considerações finais

Ao implementar serviços HTTP, estamos implementado nossa primeira API para IOT. Observe que para o desenvolvedor da aplicação, fica independente qual sensor foi utilizado para obter as informações.

O exemplo utilizado permite fazer páginas web completas e com isso, incentiva o desenvolvedor a fazer a aplicação fortemente acoplada ao dispositivo. O objetivo esperado dessa aula é justamente entender que o desenvolvimento deve ser desacoplado.