Vitor Amadeu Souza

Desenvolvendo uma aplicação cliente para monitorar

Sensores

com o ESP8266 programado no Arduino e servidor no

VC#

© 2021 by Cerne Tecnologia e Treinamento Ltda.

© 2021 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Julho de 2021

Direitos reservados por:

Cerne Tecnologia e Treinamento Ltda
Produção: Cerne Tecnologia e Treinamento
E-mail da Empresa: cerne@cerne-tec.com.br
Home Page: www.cerne-tec.com.br.com.br
Atendimento ao Consumidor: sac@cerne-tec.com.br
Contato com o Autor: vitor@cerne-tec.com.br



FEITO NO BRASIL

“Até que o Sol não brilhe, acendamos uma vela na escuridão.”

Confúcio

Kits Didáticos e Gravadores da Cerne Tecnologia

A Cerne tecnologia têm uma linha completa de aprendizado para os microcontroladores da família PIC, 8051, Holtek, dsPIC, ARM e etc. Veja os detalhes de cada um nas figuras abaixo:



Kit Cerne PIC12F

- Microcontrolador PIC12F629
- Botões
- Leds
- Gravação ICSP
- E muito mais!

Uma linha completa de componentes para o desenvolvimento de seus projetos eletrônicos como displays, PICs, botões, leds, cristais e etc.

Visite a nossa página na Internet, no endereço www.cerne-tec.com.br e conheça melhor nossos serviços e produtos.

www.cerne-tec.com.br

Cerne
Conhecimento para o Desenvolvimento

Sumário

Introdução.....6
Capítulo I – Conhecendo a NodeMCU.....7
1. Introdução.....7
2. O Arduino IDE.....8
3. Interface do NodeMCU.....11
Capítulo II – Exemplo cliente-servidor.....13
1. Esquema elétrico do cliente.....13
2. Fluxograma do cliente.....14
3. Código fonte do cliente.....14
4. Esquema elétrico do servidor.....17
5. Fluxograma do servidor.....18
6. Código fonte do servidor.....19
Capítulo III – Display LCD Paralelo.....23
Capítulo IV – Medindo a concentração de Amônia.....27
Capítulo V – Amperímetro.....45
Capítulo VI – Barômetro.....59
Capítulo VII – Medindo a concentração de Benzeno.....70
Capítulo VIII – Célula de carga.....85
Capítulo IX – Medição de dióxido de Carbono.....101
Capítulo X – Medição de Monóxido de Carbono.....120
Capítulo XI – Medindo Álcool em ppm.....138
Capítulo XII – Medindo o fluxo de massa.....151
Capítulo XIII – Gaussímetro.....165
Capítulo XIV – Medindo a concentração em GLP.....181
Capítulo XV – Medindo a concentração em H_2.....197
Capítulo XVI – Higrômetro.....212
Capítulo XVII – Medindo a concentração em Ozônio.....222
Capítulo XVIII – Medição de pH.....238
Capítulo XIX – Termômetro.....254
Capítulo XXX – Medindo a radiação ultravioleta.....265
Capítulo XXXI – Voltímetro.....282
Capítulo XXXII – Programação do servidor em VC#.....293

Introdução

A proposta desta literatura é apresentar a conexão cliente-servidor entre dois módulos ESP8266 via WiFi, estando um em modo AP (Access Point) e outro no modo STATION (estação), sendo o alcance médio em área aberta de até 100 m. Logo em seguida, uma aplicação feita em Visual C# (VC#) é deixada como servidor de modo a apresentar as informações enviadas pelo cliente.

O objetivo inicial será ler o estado de um botão no cliente (STATION) e enviar para o servidor (AP) o seu estado, de modo a este indicar através de um led se o botão está pressionado ou não. Em seguida os sensores listados no sumário da obra fica conectado no cliente de modo que o resultado da medição seja enviado para o servidor, apresentando desta forma o resultado no LCD conectado ao mesmo e logo em seguida na aplicação desenvolvida para o computador.

A programação no ESP8266 foi feita usando-se o Arduino, no qual o módulo NodeMCU foi utilizado. É fundamental que o leitor utilize como referência ou possua experiência dos assuntos abordados na literatura *Programando o ESP8266 no Arduino (2016)* do mesmo autor e editora.

Capítulo I

Conhecendo o NodeMCU

1. Introdução

A placa NodeMCU vem equipada com o circuito integrado ESP8266, possibilitando desta forma a conexão com a rede WiFi, sendo ideal para aplicações voltadas a IoT (Internet das Coisas). Ela pode ser programada em diversas linguagens, porém nesta literatura o foco será a sua manipulação através do ambiente Arduino. A próxima figura mostra o aspecto deste módulo.



Fonte: https://en.wikipedia.org/wiki/NodeMCU

Este modelo disponibiliza 11 pinos de I/O (entrada e saída), PWM, uma entrada analógica (ADC) com resolução de 10 bits dentre outros.

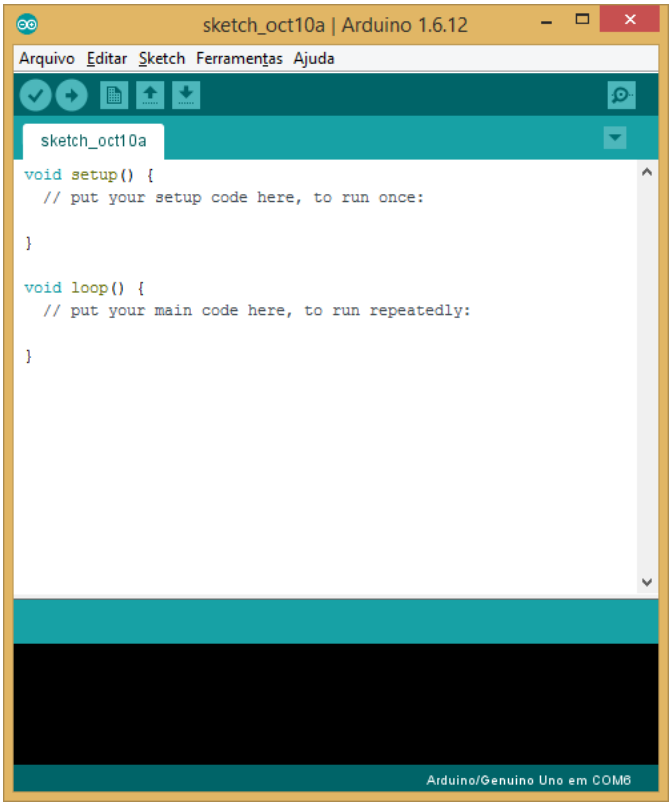
No entanto, é importante frisar a tensão máxima de entrada dos pinos digitais é de 3,3V enquanto que do analógico é de 1V. Da mesma forma, quando configurado como saída o pino pode fornecer 3,3V como tensão máxima.

2. O Arduino IDE

Baixe a versão mais recente do Arduino através do link abaixo.

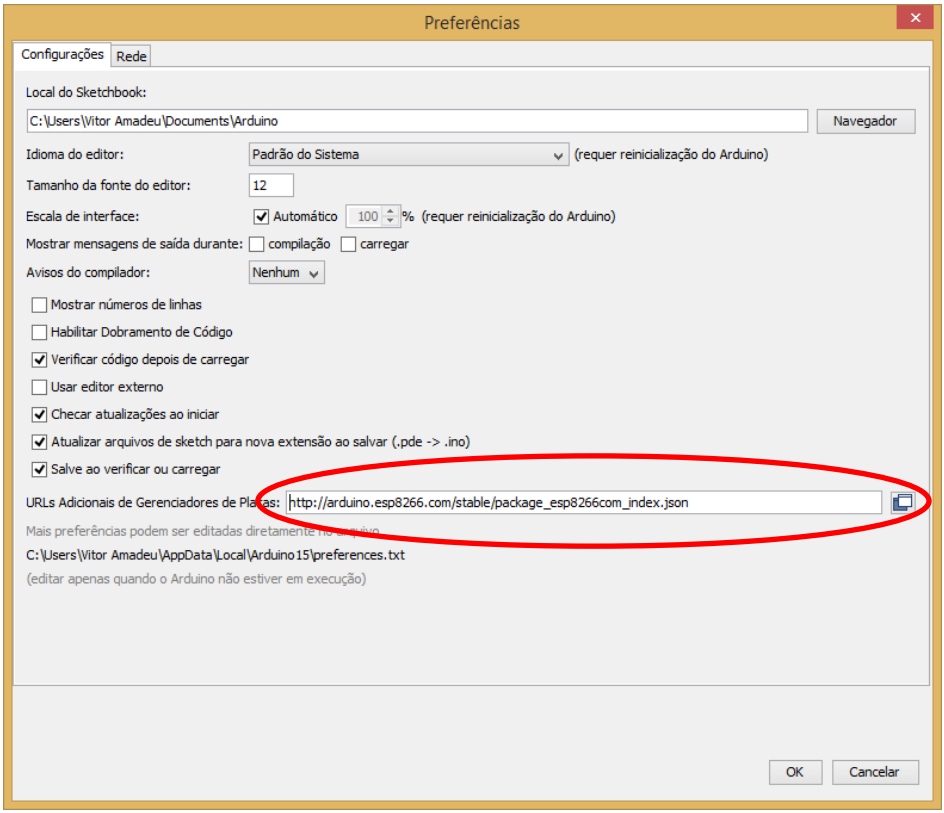
https://www.arduino.cc/en/Main/Software

A versão utilizada nesta literatura foi a 1.6.12, porém versões mais recentes podem ser utilizadas. Após o download e instalação, inicialize o programa, no qual a seguinte tela será apresentada.

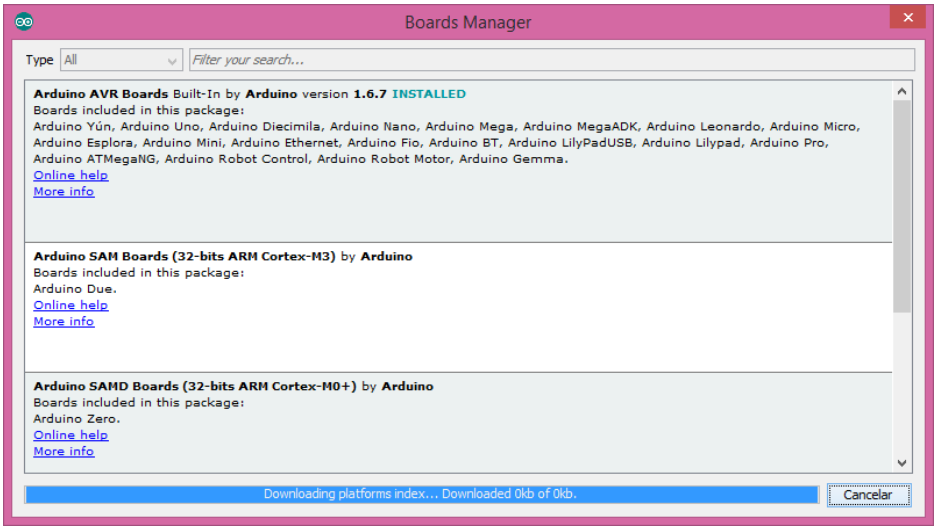


Por default o módulo NodeMCU não vem disponível na ferramenta. Para habilitá-la, vá ao menu Arquivo-> Preferências. Adicione o link a seguir no campo marcado abaixo e pressione o botão ok.

http://arduino.esp8266.com/stable/package_esp8266com_index.json



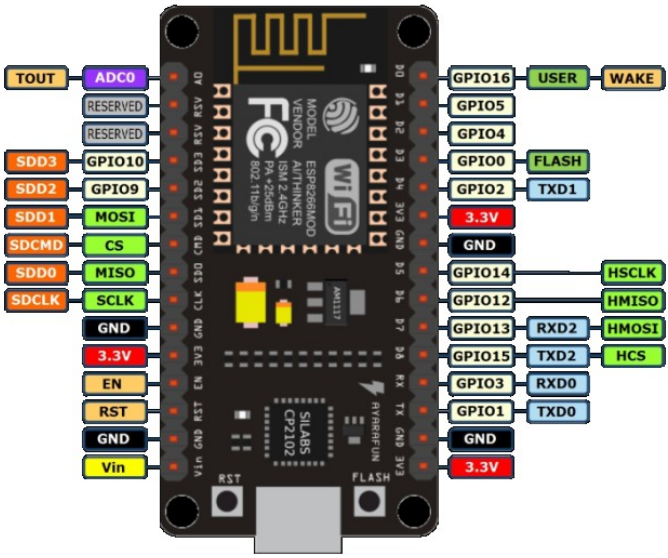
Vá ao menu Ferramenta-> Placa -> Boards Manager..., onde a seguinte tela será apresentada.



Nesta lista são apresentados todos os modelos de placa suportadas pelo ambiente Arduino, tanto as instaladas como não instaladas. Localize a opção esp8266 by ESP8266 Community e clique no botão *Instalar* para inicializar a instalação do pacote. Após este passo, observe que o modelo NodeMCU 1.0 fica disponível através do menu Ferramenta-> Placa.

3. Interface do NodeMCU

Os pinos de interface do NodeMCU está disponível através dos conectores marcados na figura a seguir.



Fonte: http://justjibba.net/getting-nodemcu-1-0-to-work-on-a-mac/

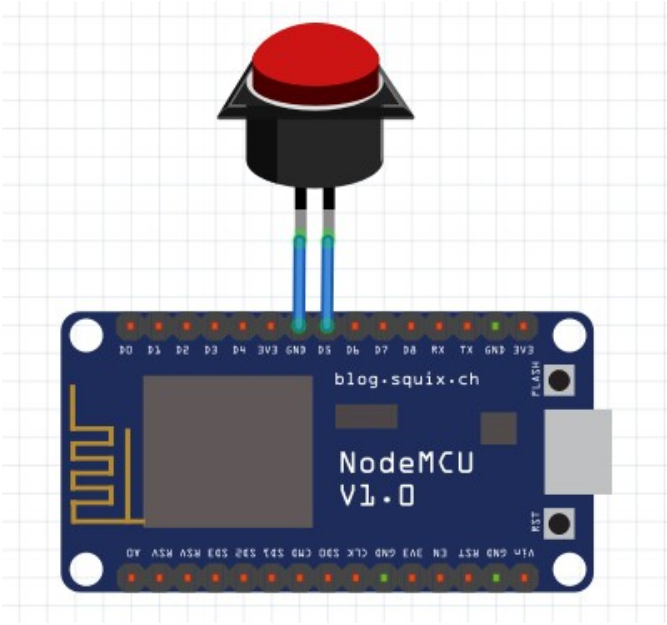
A descrição e funções destes pinos estão elencadas no próprio silk da placa, onde os pinos do tipo GPIO podem ser configurados para serem entrada ou saída em que em ambos os casos o sinal será do tipo digital. O pino analógico é identificado como ADC0.

Capítulo II

Exemplo cliente-servidor

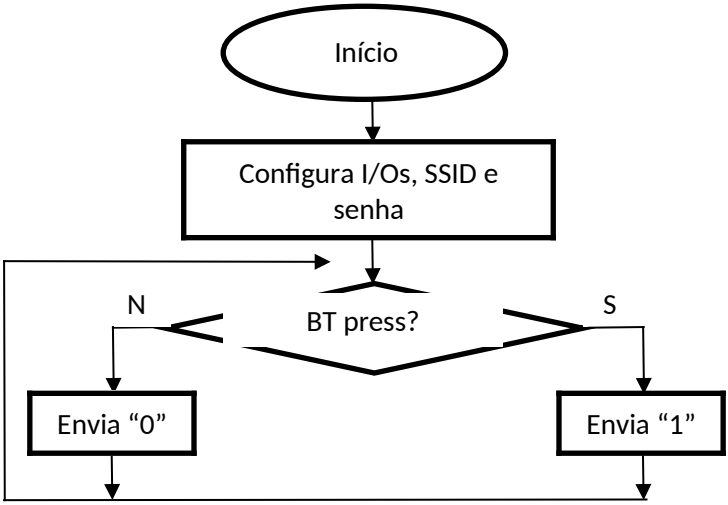
1. Esquema elétrico do cliente

No cliente há um botão conectado ao pino GPIO14, como ilustra a próxima figura.



2. Fluxograma do cliente

A lógica do programa consiste em configurar o nome da rede (SSID) e senha de acordo com o definido no outro módulo em modo AP. Após a conexão ser estabelecida com sucesso, dentro do loop principal é feita a leitura constante do estado do botão, de modo a enviar em seguida para o outro terminal o seu estado “0” ou “1”.



3. Código fonte do cliente

A seguir está apresentado o código fonte que implementa o algoritmo anterior em mod cação está logo após o código.

```
#include <ESP8266WiFi.h>

const char* ssid     = "Cerne";         //SSID da rede
const char* password = "Tecnologia";    //Senha da rede
const char* host = "192.168.4.1";       //IP do servidor

#define   INP1    14

void setup()
{
  pinMode(INP1, INPUT_PULLUP);

  WiFi.mode(WIFI_STATION);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED);
}

/*    *     *     *     *     *     *     *     *
 *                 Loop principal               *
 *    *     *     *     *     *     *     *     */

void loop()
{
  WiFiClient client;

  if (client.connect(host,80))
  {
   if(digitalRead(INP1)==0)
   client.println("1");
   else
   client.println("0");
  }
  delay(100);
}
```

Inicialmente é feita a inclusão da biblioteca padrão para comunicação WiFi e definido o SSID e senha da rede do ESP8266 oposto que funcionará em modo AP. Também é definido o IP do servidor que é fixo e tem o valor citado assim como a entrada do sensor conectada ao pino GPIO14.

```
#include <ESP8266WiFi.h>

const char* ssid     = "Cerne";         //SSID da rede
const char* password = "Tecnologia";    //Senha da rede
const char* host = "192.168.4.1";       //IP do servidor

#define   INP1    14
```

No bloco setup é configurado o pino INP1 como entrada e configura-se o módulo ESP8266 para operar em modo STATION. Define-se o SSID e senha a se conectar e aguarda para que este complete-se na linha seguinte.

```
void setup()
{
  pinMode(INP1, INPUT_PULLUP);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED);
}
```

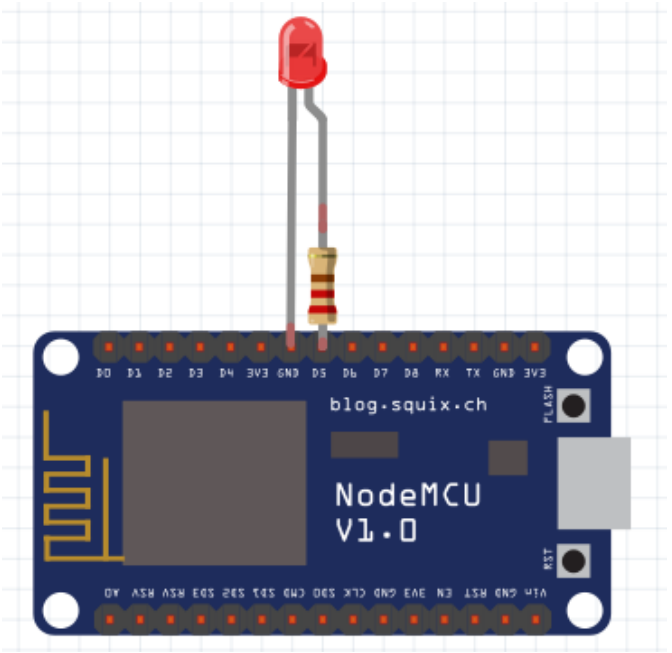
No bloco loop lê-se constantemente o estado do botão INP1, de modo a enviar o caracter “1” ou “0” para o servidor caso a conexão tenha sido estabelecida com o sucesso usando o IP do HOST e a porta (80) informada.

```
void loop()
{
  WiFiClient client;

  if (client.connect(host,80))
  {
   if(digitalRead(INP1)==0)
   client.println("1");
   else
   client.println("0");
  }
  delay(100);
}
```

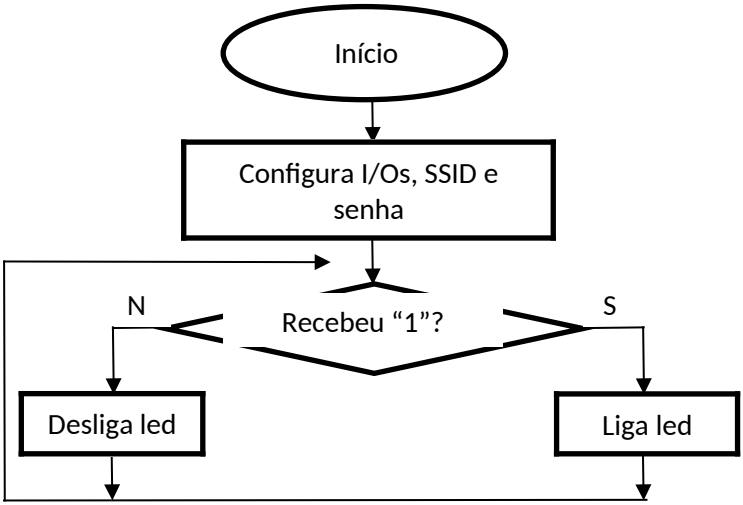
4. Esquema elétrico do servidor

No servidor há um led conectado ao pino GPIO14, como ilustra a próxima figura.



5. Fluxograma do servidor

A lógica do programa consiste em configurar o nome da rede (SSID) e senha, para em seguida aguardar a conexão do ESP8266 em modo STATION. O IP que o módulo disponibiliza em modo AP é fixo e tem valor 192.168.4.1. Em seguida aguarda-se o envio de algum caracter do cliente, onde dependendo do seu estado o led é ligado ou não.



6. Código fonte do servidor

A seguir está apresentado o código fonte que implementa o algoritmo anterior em mo o está detalhada logo após o código.

```
#include <ESP8266WiFi.h>

const char* ssid     = "Cerne";         //SSID da rede
const char* password = "Tecnologia";    //Senha da rede

#define   OUT1    14

WiFiServer server(80);

void setup()
{
  pinMode(OUT1, OUTPUT);
```

```
  digitalWrite(OUT1, 0);

  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password);
  server.begin();
}

/*    *     *     *     *     *     *     *     *
 *                 Loop principal               *
 *    *     *     *     *     *     *     *     */

void loop()
{
  WiFiClient client = server.available();

  if (!client)
    return;

  while(!client.available())
    delay(1);

  String req = client.readStringUntil('\r');
  Serial.println(req);
  client.flush();

  if (req.indexOf("1") != -1)
    digitalWrite(OUT1, HIGH);

  if (req.indexOf("0") != -1)
    digitalWrite(OUT1, LOW);
}
```

Inicialmente é feita a inclusão da biblioteca padrão para comunicação WiFi e definido o SSID e senha da rede do ESP8266. É definida a saída no qual o led está conectado e um objeto para tratamento do servidor.

```
#include <ESP8266WiFi.h>

const char* ssid     = "Cerne";        //SSID da rede
const char* password = "Tecnologia";   //Senha da rede

#define   OUT1    14

WiFiServer server(80);
```

No bloco setup é configurado o pino OUT1 como saída e configura-se o módulo ESP8266 para operar em modo AP. Define-se o SSID e senha a inicializa-se o servidor.

```
void setup()
{
  pinMode(OUT1, OUTPUT);
  digitalWrite(OUT1, 0);

  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password);
  server.begin();
}
```

No bloco loop lê-se constantemente se há algum pedido de algum cliente. Caso haja, faz-se a leitura da string e verifica-se em seguida seu conteúdo, de modo a ligar ou não o led.

```
void loop()
{
  WiFiClient client = server.available();

  if (!client)
    return;
```

```
  while(!client.available())
    delay(1);

  String req = client.readStringUntil('\r');
  Serial.println(req);
  client.flush();

  if (req.indexOf("1") != -1)
    digitalWrite(OUT1, HIGH);

  if (req.indexOf("0") != -1)
    digitalWrite(OUT1, LOW);
}
```

Após a montagem do esquema e escrita de cada um dos programas, faça o teste com os módulos ESP8266 estando um em modo STATION e outro AP para verificar e comprovar o seu funcionamento.

Capítulo III

Display LCD Paralelo

1. Introdução

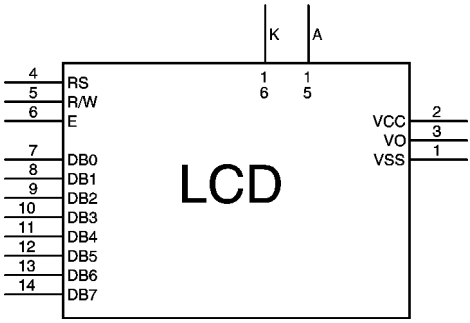
Os displays do tipo LCD são muito usados para mostrar informações do estado de uma máquina. Na figura a seguir, é apresentado aspecto típico desta interface.



Existem vários tipos de display, como os de 2x20, que significa 2 linhas por 20 colunas, 4x20, 4x16 e etc. Neste exemplo, será usado um display do tipo 2x16, onde haverá duas linhas e 16 colunas. Será apresentado um texto no LCD, de forma a entender como montar um programa no NodeMCU para comandá-lo.

2. Montando o Hardware

Na figura a seguir, é apresentado a pinagem típica de um display LCD.



Siga a tabela apresentada a seguir para ligar o display LCD à placa NodeMCU .

Display LCD	Placa NodeMCU
VCC	5V
VSS	GND
VO	Em série com resistor de 1kΩ ao GND
RS	GPIO16
EN	GPIO5
D4	GPIO4
D5	GPIO0
D6	GPIO2
D7	GPIO14

Os pinos não citados não são usados.

3. Programando o Arduino

Foi utilizada as funções disponíveis em uma biblioteca do Arduino para que o exemplo fique funcional. A biblioteca é a `LiquidCrystal.h` que contém as funções de inicialização e escrita no LCD. Além disso, será necessário utilizar a função `lcd.begin` para configurar o tipo de display conectado ao NodeMCU . A seguir é apresentada tal função.

```
lcd.begin(colunas,linhas);
```

Em colunas, define-se o número de colunas no qual o display possui e em linhas o número de linhas. Após este passo, utiliza-se a função `lcd.setCursor` para posicionar a impressão de determinada informação. A seguir, tal função é apresentada.

```
lcd.setCursor(coluna,linha);
```

Onde caso coluna esteja em 0, a coluna 1 estará selecionada e caso esteja em 1, a coluna 2 e assim sucessivamente. Mesma ideia é válida para a opção linha. Finalmente, para imprimir um valor no LCD, utiliza-se a função `lcd.print` como apresentado a seguir.

```
lcd.print(texto);
```

A seguir os passos para mostrar uma mensagem na 1° e 2° linha do display lcd.

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(16, 5, 4, 0, 2, 14);

void setup()
{
  lcd.begin(16, 2);
  lcd.setCursor(0,0);
  lcd.print("Placa Arduino");
  lcd.setCursor(0,1);
  lcd.print("Exemplo LCD");
}

void loop()
{

}
```

Capítulo IV

Medindo a concentração de Amônia

1. Efeitos da Amônia ao ser humano

O gás amônia ou amônia, cuja fórmula molecular é NH_3 está disposto geometricamente da seguinte forma.

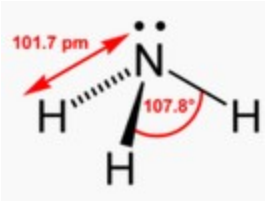


Figura 3.1: Representação da amônia
Fonte: http://pt.wikipedia.org/wiki/Am%C3%B4nia

É um gás tóxico em temperatura ambiente e pressão atmosférica e na presença da umidade torna-se corrosivo. É um gás inflamável, incolor, com odor irritante e possui alta solubilidade em meio aquoso, como a água. Possui diversas aplicações, como agente refrigerante industrial e como fertilizante.

Em altas concentrações, produz tosses fortes devido a sua ação nas vias respiratórias além de causar queimadura nos olhos. Além

disso, a vítima sofrerá irritação dos pulmões podendo evoluir a um edema pulmonar em pouco tempo, sendo necessária a rápida remoção para não agravar tal situação. Em caso de ingestão, provocará corrosão da boca, garganta e estômago.

Segundo o anexo 11 da NR 15 (Norma Regulamentadora 15), a concentração máxima para uma exposição semanal de até 48 horas é de 20 ppm. Caso este limite seja extrapolado, o ambiente é considerado medianamente insalubre.

Os principais riscos oferecidos pela Amônia são ligados a saúde, conforme mostra a classificação de rotulagem da figura a seguir.

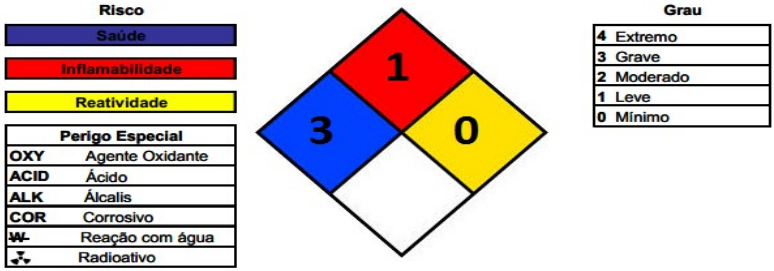


Figura 3.2: Riscos associados a Amônia
Fonte: http://pt.wikipedia.org/wiki/Am%C3%B4nia

Note pelo quadro que os riscos associados a reatividade são mínimos (0), enquanto que o de inflamabilidade é leve (1) e saúde grave (3).

Face ao exposto, torna-se importante dispor de equipamentos que meçam a concentração de Amônia de maneira contínua e possam alertar os seus usuários caso níveis críticos sejam atingidos.

Neste capítulo, será desenvolvido um dispositivo capaz de medir a concentração de Amônia no ambiente e que envia pela serial tal valor em ppm (parte por milhão).

2. Sensor MQ-135

O sensor MQ-135, desenvolvido pela Hanwei Eletronics (www.hwsensor.com), permite medir a concentração em ppm da Amônia (NH_3) com alta repetibilidade e seletividade. A próxima figura mostra o encapsulamento deste sensor.



Figura 3.4: Encapsulamento do MQ-135
Fonte: https://www.sparkfun.com/products/10916

O MQ-135 opera como um resistor variável, que acompanha a variação de Amônia no ambiente. A sua resposta é linear e quanto