**Vitor Amadeu Souza**

Desenvolvimento de Projetos

# Datalogger

para medição de

# Sensores

Programado em Arduino no STM32F103C8

**© 2020 by Cerne Tecnologia e Treinamento Ltda.**
**© 2020 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.

**Outubro de 2020**

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**

**Dedicatória**

Como nos meus outros livros, dedico este livro a minha querida esposa Renata Leal.

***"A maioria de nós prefere olhar para fora e não para dentro de si mesmo."***

**Albert Einstein**

**Kits Didáticos e Gravadores da Cerne Tecnologia**

A Cerne tecnologia têm uma linha completa de aprendizado para os MCUs da família PIC, 8051, Holtek, dsPIC, ARM, Arduino, etc. Veja os detalhes de um de nossos kits.




## Kit Cerne Arduino

- Microcontrolador ATMEGA8;
- Comunicação serial RS232;
- Alimentação de 12V;
- Pinos de I/O;
- Gravação ICSP.

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.

Cerne
Conhecimento para o Desenvolvimento

**www.cerne-tec.com.br**

# Sumário


| | |
|---|---|
| **I. Metodologia de desenvolvimento** | **6** |
| **II. Acessando o SD CARD** | **7** |
| **III. Medição de corrente de 4 a 20 mA** | **15** |
| **IV. Medindo Álcool em ppm** | **26** |
| **V. Medindo a concentração de Amônia** | **40** |
| **VI. Amperímetro** | **58** |
| **VII. Barômetro** | **72** |
| **VIII. Medindo a concentração de Benzeno** | **83** |
| **IX. Célula de Carga** | **100** |
| **X. Medição de Monóxido de Carbono** | **116** |
| **XI. Medição de Dióxido de Carbono** | **136** |
| **XII. Decibelímetro** | **156** |
| **XIII. Dinamômetro** | **175** |
| **XIV. Gaussímetro** | **197** |
| **XV. Medindo a concentração de GLP** | **214** |
| **XVI. Medindo a concentração de $H_2$** | **231** |
| **XVII. Higrômetro** | **248** |
| **XVIII. Medindo a concentração de Ozônio** | **259** |
| **XIX. Medição de pH** | **276** |
| **XX. Medição de tensão RMS** | **292** |
| **XXI. Medição de temperatura com NTC** | **304** |
| **XXII. Medindo a radiação ultravioleta** | **315** |

# Capítulo I
# Metodologia de desenvolvimento

## 1. Introdução

Esta literatura é uma continuação da obra *ARM no Arduino – Com base no modelo STM32F103C8 (2019)* do mesmo autor e editora, onde o objetivo é desenvolver um datalogger para registrar os sensores listados no sumário da obra os dados referentes à medição em um SD CARD.

A placa didática utilizada foi a *Blue Pill* STM32F103C8, onde os recursos para testar os circuitos propostos serão conectados através das conexões disponíveis na lateral do kit. Tal kit está à venda no site www.cerne-tec.com.br.

# Capítulo II

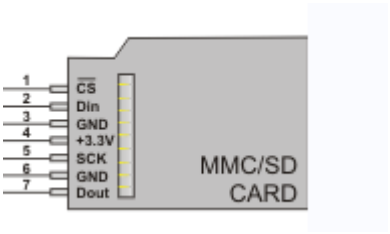
# Acessando o SD CARD

**1. Histórico**

Os cartões do tipo SD CARD são uma evolução dos cartões MMC (Multi Media Card) e normalmente são intercambiáveis. No Brasil encontramos com facilidade e a um preço acessível os cartões do tipo SD, no qual iremos usar nesta literatura. O Cartão SD CARD nasceu de um consórcio formado pela Panasonic (Matsushita Electric), SanDisk e Toshiba em 1999 e hoje disputam com outros fabricantes como a Sony com o seu Memory Stick a liderança de mercado, onde de fato o SD CARD sai na frente.

**2. Pinagem do SD CARD**

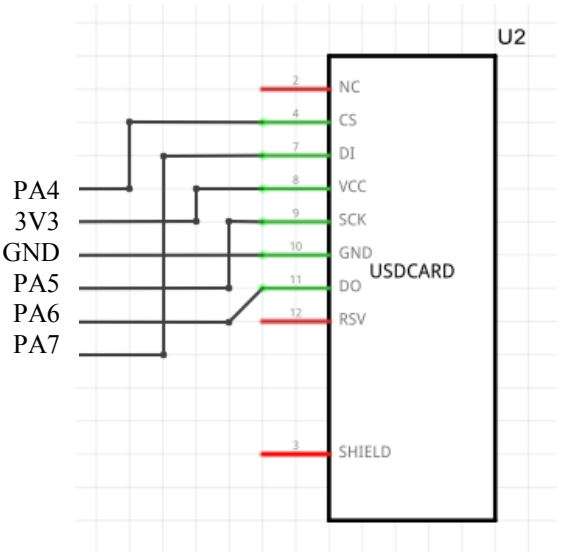
Vejamos na figura abaixo como está disponível a pinagem do SD CARD:



Note que temos 2 pinos de GND, 1 de VCC (3,3V) e outros de comunicação que são os pinos CS, DIN, SCK e DOUT. A comunicação utilizada para comunicar com o STM32 é feito com base no barramento SPI.

**3. Esquema elétrico**

A próxima figura mostra a conexão que deve ser feita entre a o SD CARD para conectá-lo a placa *Blue Pill* STM32F103C8 no qual os pinos DIN, DOUT e SCK devem ser ligados aos pinos digitais PA6, PA7 e PA5. O pino CS pode ser um pino digital qualquer, no qual neste exemplo foi utilizado o pino PA4.



## 4. Funções do Arduino

Há diversas funções disponíveis no Arduino que permite o acesso a memória SD CARD. A seguir são apresentadas as mais importantes, utilizadas ao longo da literatura, onde em todos os casos é fundamental que as bibliotecas SD.h e SPI.h estejam declaradas no início do programa.

### SD.begin

Através desta função podemos inicializar o módulo de comunicação SD CARD para realizarmos comunicação com o cartão. Deve-se especificar qual pino digital está sendo utilizado como CS.

```
#include <SPI.h>
#include <SD.h>

void setup()
{
  char estado;
  estado=SD.begin(PA4);
}

void loop()
{
}
```

Caso o retorno tenha valor 1, indica que o cartão está conectado e caso contrário está desconectado.

**SD.open**

Através desta função podemos abrir um arquivo no SD CARD. É importante informar o nome do arquivo e o tipo de acesso que será feito, podendo ser escrita (FILE_WRITE) ou leitura (FILE_READ). O exemplo a seguir abre o arquivo datalog.txt no modo de escrita. Neste caso, os dados são escritos a partir do último registro salvo no SD CARD.

```
File dataFile = SD.open("datalog.txt", FILE_WRITE);
```

**SD.remove**

Remove um arquivo do SD CARD de acordo com o nome informado como parâmetro. Retorna verdadeiro se o processamento ocorreu com sucesso.

```
SD.remove(arquivo);
```

**SD.exists**

Verifica se existe determinado arquivo na memória. Retorna verdadeiro caso exista.

```
SD.exists(arquivo);
```

**datafile.println**

Após a criação do objeto datafile podemos realizar operações de escrita e leitura no cartão. Este comando imprime o conteúdo informado e encerra a string com CR+LF (Enter e nova linha do teclado). Isso é feito usando-se o método println, como mostra o código abaixo.

```
#include <SPI.h>
#include <SD.h>
const int chipSelect = PA4;

void setup()
{
  char estado;
  estado=SD.begin(chipSelect);
  while(estado==0);  //Fica preo se falhou
  File dataFile = SD.open("datalog.txt", FILE_WRITE);

  if (dataFile)
  {
   dataFile.println("Testando");
   dataFile.close();
  }

}

void loop()
{
}
```

O comando println faz o mesmo, porém não termina a string com CR+LF.

**datafile.close**

Encerra o acesso ao arquivo aberto.

**datafile.read**

Lê um byte da posição corrente.

## 5. Exemplos Práticos

Os exemplos a seguir permitem ao leitor entender melhor as funções apresentadas para acessar a memória SD CARD. Após a escrita e teste do exemplo, procure remover o cartão e verificar se a função testada corresponde ao resultado alcançado.

### 5.1 Exemplo 1 - Criando um Arquivo

Neste primeiro experimento veremos como criar um arquivo na memória SD CARD e escrever uma informação no mesmo.

```
#include <SPI.h>
#include <SD.h>
const int chipSelect = PA4;

void setup()
{
  char estado;
  estado=SD.begin(chipSelect);
  while(estado==0);  //Fica preo se falhou
  File dataFile = SD.open("datalog.txt", FILE_WRITE);
```

```
  if (dataFile)
  {
   dataFile.println("Testando");
   dataFile.close();
  }
}
void loop()
{
}
```

### 5.2 Exemplo 2 - Deletando um arquivo

Com o código abaixo poderemos deletar um arquivo gravado na memória SD CARD.

```
#include <SPI.h>
#include <SD.h>
const int chipSelect = PA4;

void setup()
{
  char estado;
  estado=SD.begin(chipSelect);
  while(estado==0);  //Fica preo se falhou
  File dataFile = SD.open("datalog.txt", FILE_WRITE);

  if (dataFile)
  {
   dataFile.println("Testando");
   dataFile.close();
  }
  SD.remove("datalog.txt");
}

void loop()
{
}
```

### 5.3 Exemplo 3 - Lendo um conteúdo de um arquivo

Este exemplo lê um conteúdo gravado no SDCARD e envia para o PC o seu conteúdo via canal serial de comunicação RS 232. O código está apresentado a seguir:

```
#include <SPI.h>
#include <SD.h>
const int chipSelect = PA4;

void setup()
{
  char estado;
  estado=SD.begin(chipSelect);
  while(estado==0);  //Fica preo se falhou
  File dataFile = SD.open("datalog.txt", FILE_WRITE);

  if (dataFile)
  {
   dataFile.println("Testando");
   Serial.begin(9600);
   while (dataFile.available())
   {
      Serial.write(dataFile.read());
   }
   dataFile.close();
  }
}

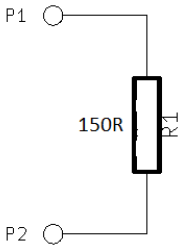
void loop()
{
}
```

# Capítulo III

# Medição de corrente de 4 a 20 mA

**1. Conversor corrente-tensão**

Para realizar a medição da corrente faz-se necessário um conversor de corrente em tensão, já que o ARM é capaz de medir apenas tensão pelo conversor ADC. Tal efeito é alcançado com o emprego de um resistor de 150Ω acoplado a entrada de corrente, como poder ser visto na figura a seguir.




Pela 1º Lei de Ohm, a tensão é dada da seguinte forma:

$$V = RI$$

Onde V é a tensão em Volts, R é a resistência em Ω e I á a corrente em Amperes. Sendo assim, com a resistência de 150 Ω e com

a entrada máxima de corrente que é de 20 mA, observa-se a seguinte tensão na entrada analógica do ARM.

$$V = 150 * 0{,}02$$

$$V = 3{,}0V$$

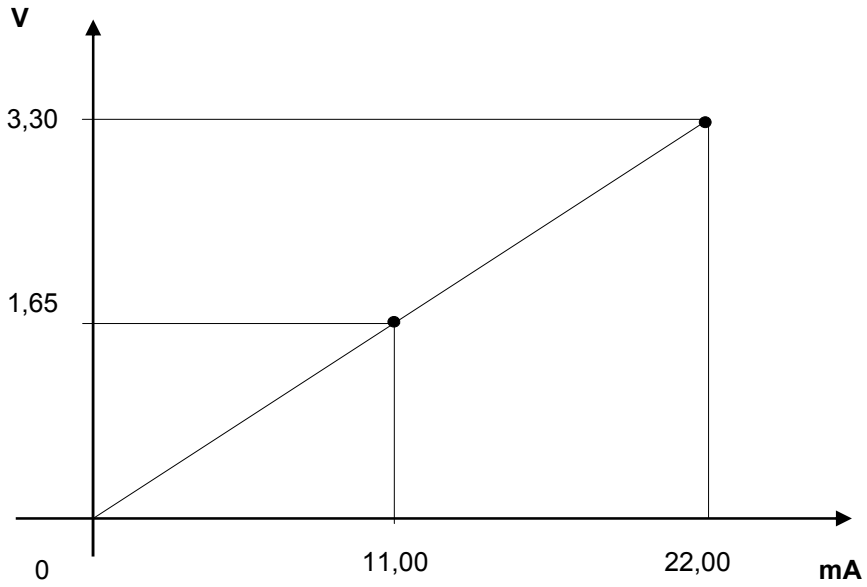
Quando a corrente mínima for medida (4 mA) a seguinte tensão é observada.

$$V = 150 * 0{,}004$$

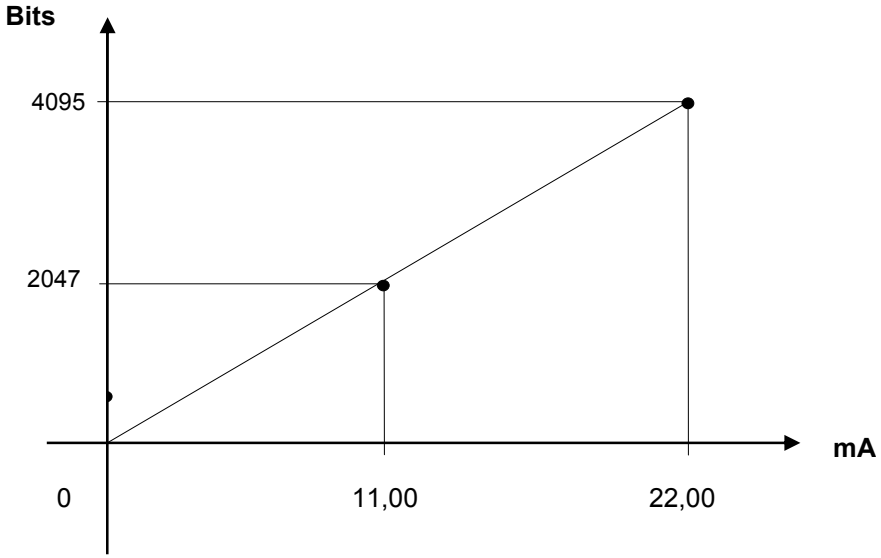
$$V = 0{,}6V$$

É assim que sabe-se a corrente na entrada do ARM, passando primeiro pelo conversor corrente-> tensão e em seguida medindo o mesmo de forma linear.

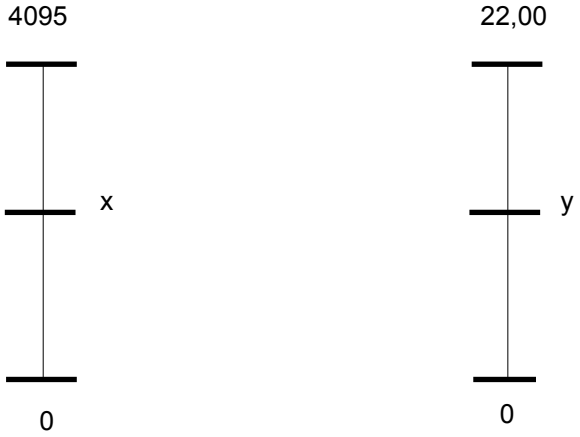
A medição é feita na faixa de 0 a 22,00 mA. A tensão de saída Vout é proporcional a corrente de entrada, onde caso haja a corrente mínima de 0 mA na saída do conversor observa-se a tensão de 0 V. Caso haja a corrente máxima de 22,00 mA a tensão no pino Vout é de 3,3 V e assim linearmente, conforme sugere o gráfico abaixo.




O conversor ADC do ARM é de 12 bits, fornecendo uma resolução de 4096 pontos, pois $2^{12}$ é igual a este valor. O sensor de corrente está ligado a uma das entradas analógicas do ARM, no qual fornece o resultado binário na faixa de 0 a 4095. Sendo assim é possível modelar o seguinte gráfico, no qual converte a escala de corrente para os bits referentes à conversão ADC.




Desta forma a seguinte relação matemática é observada entre o resultado da conversão ADC e a medição dada em URA.



$$\frac{x-0}{4095-0} = \frac{y-0}{22{,}00-0}$$

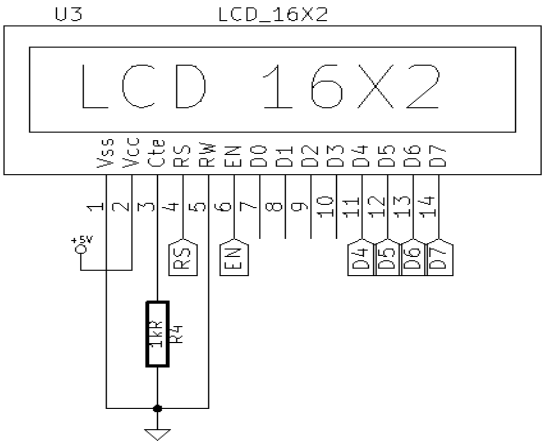
$$\frac{x}{4095} = \frac{y}{22{,}00}$$

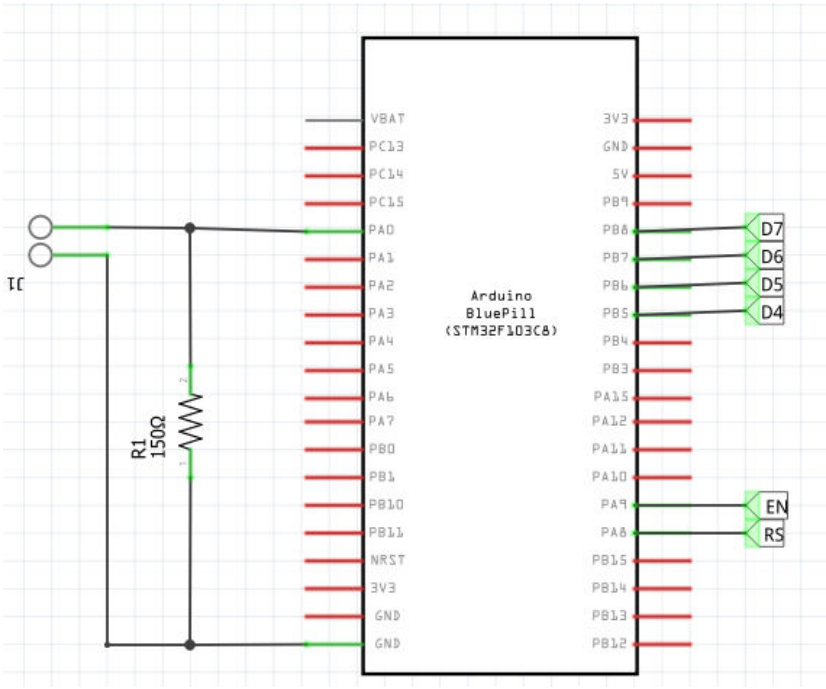
$$y = 22{,}00 * \frac{x}{4095}$$

Ou seja, após a leitura da entrada analógica, deve-se executar o cálculo matemático acima de forma a obter o valor da corrente e apresentá-la no display.

**2. Esquema elétrico**

A próxima figura apresenta o esquema elétrico, no qual observa-se o conversor corrente-tensão conectado a entrada analógica do ARM e este conectado a um display LCD.






### 3. Código fonte

A seguir o código fonte para medir a fonte de corrente de 4 a 20 mA e apresentar no LCD o resultado da medição. Após a leitura do canal analógico é feita a conta matemática apresentada anteriormente para se obter o resultado em mA e atualizar no LCD tal resultado.

```
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *         Cerne Tecnologia           *
 *       www.cerne-tec.com.br         *
 *   Projetos em Arduino -Corrente    *
 *  *  *  *  *  *  *  *  *  *  *  *  */
#include <LiquidCrystal.h>
LiquidCrystal lcd(PA8, PA9, PB5, PB6, PB7, PB8);

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *      Função de inicialização      *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void setup(void)
{
  lcd.begin(16,2);
}
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Função de loop           *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void loop(void)
{
  int corrente;
  corrente=(22.00*analogRead(0))/4095;
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print(corrente,DEC);
  lcd.print(" mA");
  delay(1000);
}
```

## 4. Datalogger

Um datalogger permite salvar periodicamente os dados apurados em uma memória não volátil, permitindo sua averiguação posterior usando um computador em forma gráfica, por exemplo. Desta forma, é possível estabelecer padrões de comportamento de

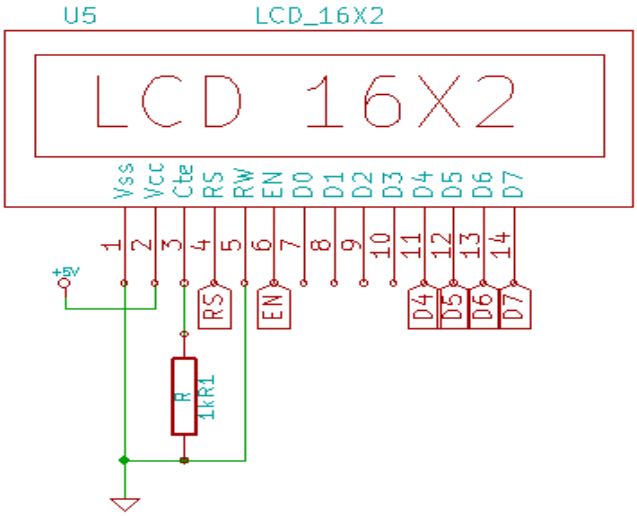
acordo com o horário, dia, mês, ano etc, sendo fundamental para levantamento estatístico.

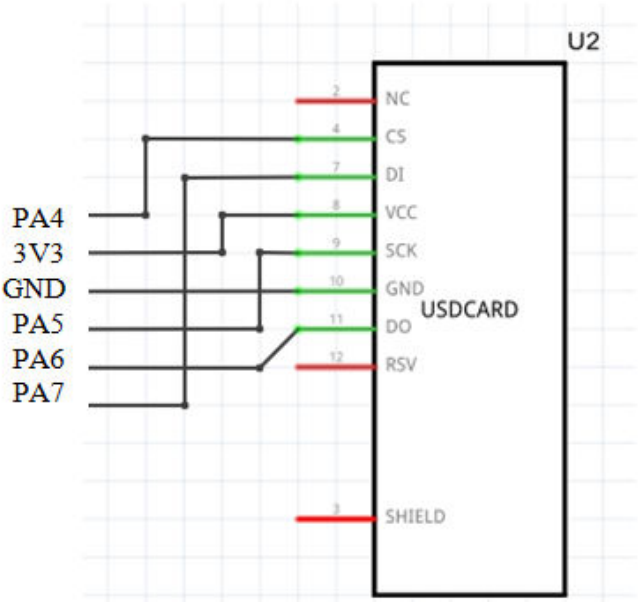
Neste exemplo, os dados adquiridos do sensor estudado são salvos na memória SD CARD a cada 1 s.

## 5. Esquema elétrico

A próxima figura apresenta o esquema elétrico do projeto. Observe que ele é composto dos tópicos referentes ao acesso ao SD CARD e o sensor estudado.

| Display LCD | Blue Pill |
|---|---|
| VCC | 5V |
| VSS | GND |
| VO | Em série com resistor de 1kΩ ao GND |
| RS | PA8 |
| EN | PA9 |
| D4 | PB5 |
| D5 | PB6 |
| D6 | PB7 |
| D7 | PB8 |
| SENSOR | PA0 |
| CS | PA4 |
| SCK | PA5 |
| DO | PA6 |
| DI | PA7 |


U5
LCD_16X2
LCD 16X2
Vss
Vcc
Cte
RS
RW
EN
D0
D1
D2
D3
D4
D5
D6
D7
1
2
3
4
5
6
7
8
9
10
11
12
13
14
+5V
RS
EN
D4
D5
D6
D7
R
1kR1


U2
NC
CS
DI
VCC
SCK
GND
DO
RSV
SHIELD
USDCARD
PA4
3V3
GND
PA5
PA6
PA7

## 6. Código fonte

O código fonte listado a seguir é uma junção dos anteriores, onde os dados obtidos do anemômetro são salvos periodicamente no SD CARD.

```
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Cerne Tecnologia          *
 *        www.cerne-tec.com.br        *
 *             Datalogger             *
 *  *  *  *  *  *  *  *  *  *  *  *  */
#include <LiquidCrystal.h>
#include <SPI.h>
#include <SD.h>
const int chipSelect = PA4;
LiquidCrystal lcd(PA8, PA9, PB5, PB6, PB7, PB8);

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *      Declaração de variáveis       *
 *  *  *  *  *  *  *  *  *  *  *  *  */
File dataFile;

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *      Função de inicialização       *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void setup(void)
{
  char estado;
  lcd.begin(16,2);
  estado=SD.begin(chipSelect);
  while(estado==0);
  dataFile = SD.open("datalog.txt", FILE_WRITE);
}

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Função de loop            *
```

```
 *  *  *  *  *  *  *  *  *  *  *  *  */
void loop(void)
{
  float i;
  i=(analogRead(0)*22.0)/4095;
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("  Amperimetro   ");
  lcd.setCursor(0,5);
  lcd.print(i,DEC);
  lcd.print(" A");
  dataFile.println(i);
  delay(1000);
}
```

# Capítulo IV

# Medindo Álcool em ppm

## 1. Sensor TGS 822

O sensor TGS 822, desenvolvido pela Figaro Sensor (www.figarosensor.com), permite medir uma série de vapores orgânicos como Metano, Monóxido de Carbono, Isobutano, Hexano, Benzeno, Álcool e Acetona. Cada um destas soluções, apresenta uma curva diferente de concentração em ppm (Parte Por Milhão), na faixa de 50 a 5000 ppm.

Este sensor apresenta alta sensibilidade, estabilidade, baixo custo e sua interface elétrica é relativamente simples. Suas principais aplicações são na detecção de álcool (podendo ser usado como bafômetro), detecção de vazamento de gás, detectores de solvente em fábrica etc. O elemento sensor é composto de Dióxido de Estanho ($SnO_2$), que varia sua condutividade de acordo com a concentração do gás presente em seu ambiente de instalação. Com o aumento da concentração de gás no ar, sua condutividade aumenta, em que desta forma um ARM pode mensurar a concentração de determinado gás neste ambiente.

A próxima figura mostra a resposta de sensibilidade do sensor, de acordo com a concentração em ppm do solvente orgânico que estiver sendo medido pelo mesmo.

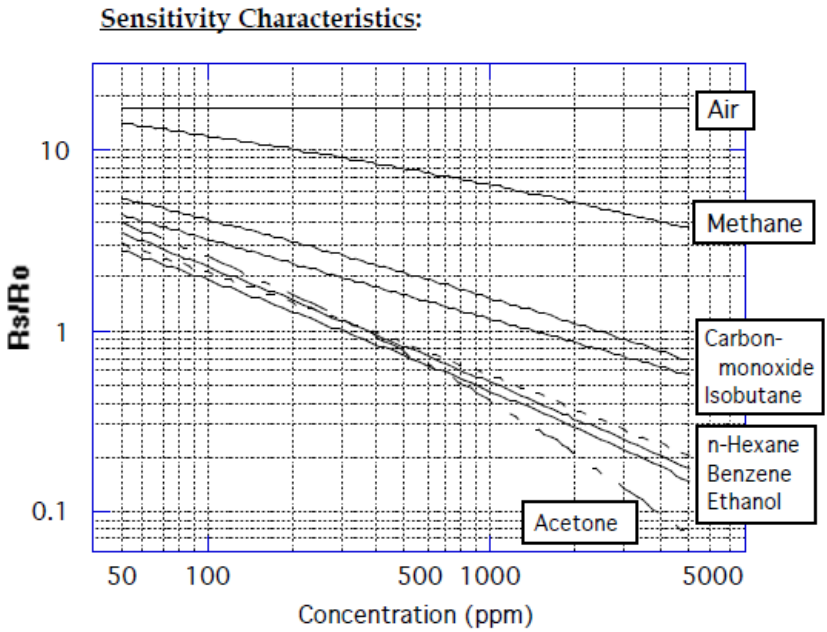



**Figura 2.1:** Resposta de acordo com a concentração de solvente
**Fonte:** TGS 822 Datasheet

A resposta do sensor varia de acordo com a temperatura e umidade relativa (R.H) do ambiente em que o mesmo estiver instalado, de acordo com a figura a seguir fornecida pelo próprio datasheet do componente.

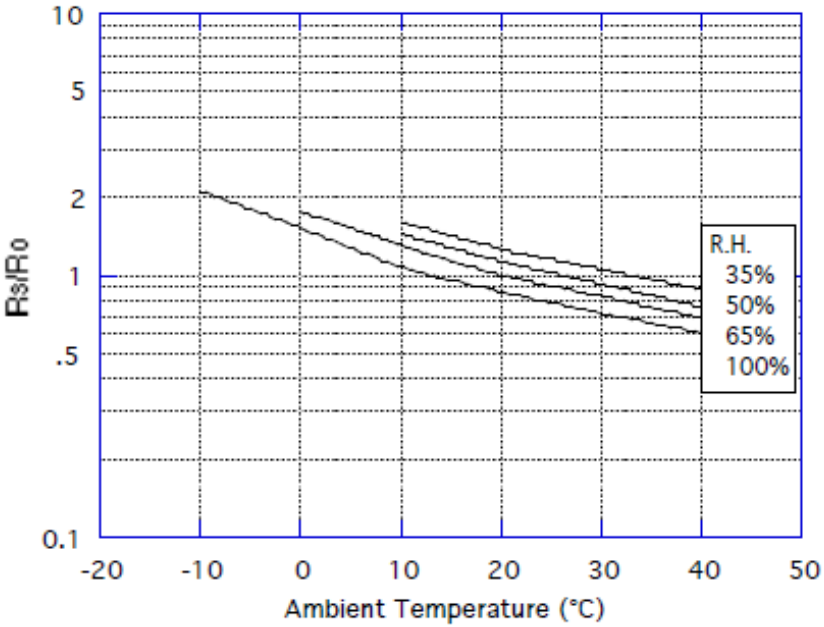



**Figura 2.2:** Influência da variação de umidade e temperatura
**Fonte:** TGS 822 Datasheet

Em determinados projetos que necessitem de maior precisão na resposta da medição de gases, pode ser necessário haver um sensor de umidade e temperatura, de modo a ajustar a resposta do TGS 822 de acordo com estas variações.

Este sensor é composto de 6 pinos, conforme pode ser visto na figura a seguir.

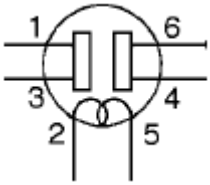



**Figura 2.3:** Pinagem do TGS 822
**Fonte:** TGS 822 Datasheet

A função da cada um dos pinos está descrita a seguir.

| Pino | Função | Descrição |
|---|---|---|
| 1 e 3 | $V_C$ | Alimentação positiva. Máximo de 24 Vdc. |
| 2 | $V_H$ | Tensão de aquecimento. Tensão de 5V DC ou AC |
| 5 | GND | Terra da fonte |
| 4 e 6 | $V_{RL}$ | Tensão variável de acordo com a concentração de gases |

**Fonte: TGS 822 Datasheet**

Normalmente, os pinos 1, 2 e 3 ficam ligados juntos na tensão de 5 V. A saída $V_{RL}$ fica acoplada a um resistor de carga ($R_L$), pois a resistência interna do sensor, ou seja, sua condutividade, varia segundo esta concentração, permitindo ao ARM mensurá-la.

A próxima figura demonstra um circuito típico a ser montado para uso deste sensor.

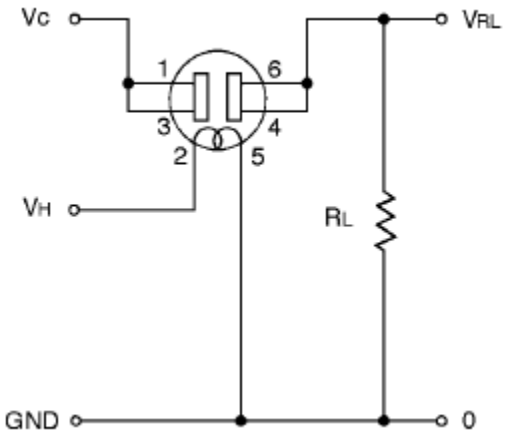



**Figura 2.4:** Circuito típico com TGS 822
**Fonte:** TGS 822 Datasheet

Um valor usual para $R_L$ é de 10 kΩ. Observe pela curva do gráfico 2.1 que a concentração em ppm é encontrada de acordo com a relação $R_S/R_0$. O valor de $R_S$ é definido como a resistência do sensor de acordo com a concentração de gás ambiente enquanto que $R_0$ para uma concentração de 50 ppm de Álcool, tipicamente 5.000Ω. O valor de $R_S$ será encontrado segundo a fórmula a seguir.

$$R_S = \left( \frac{V_C}{V_{RL}} - 1 \right).R_L$$

Considerando $V_C$=5V e $R_L$=10.000Ω, o cálculo fica da seguinte forma.

$$R_S = \left( \frac{5}{V_{RL}} - 1 \right).10000$$

O valor de $V_{RL}$ é encontrado através da leitura do conversor analógico do ARM (CAD), estando na faixa de 0 a 3,3 V. Após encontrar $R_S$, basta encontrar a relação deste valor com $R_0$ (que vale 5.000Ω) para em seguida encontrar a concentração em ppm.

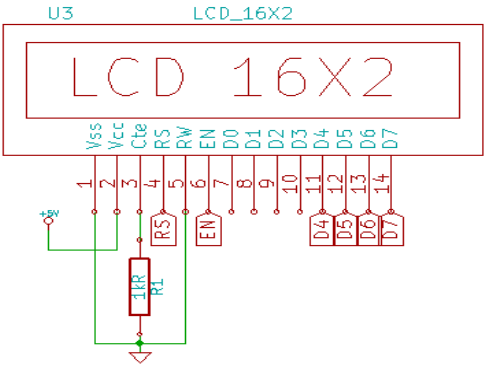
Chamando de *x* a relação entre $R_S/R_0$ e *ppm* a concentração, a função a seguir descreve como é encontrada a concentração.

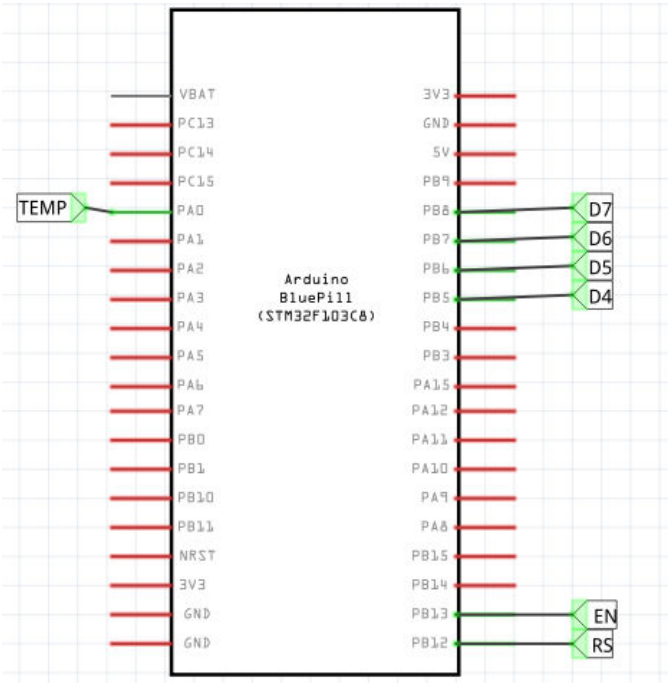
$$ppm = 169.(\log(x))^4 - 147.(\log(x))^3 + 207.(\log(x))^2 - 496.\log(x) + 332$$

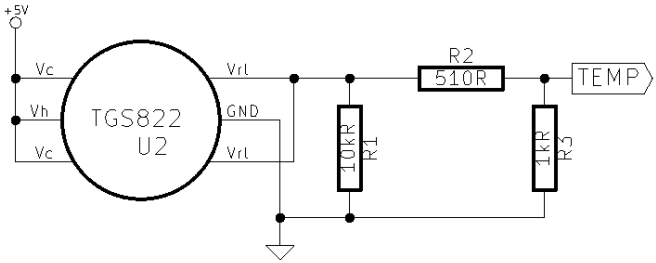
A descrição de como esta função foi encontrada está disponível no artigo de Jiana Shin, onde o link está disponível na seção Referências desta literatura.

**2. Esquema elétrico**

O esquema elétrico para este exemplo está apresentado a seguir, onde observa-se o display LCD que mostra o valor da concentração em ppm de Álcool e o sensor que é a fonte de dados para o ARM.


U3
LCD_16X2
LCD 16X2
Vss
Vcc
Cte
RS
RW
EN
D0
D1
D2
D3
D4
D5
D6
D7
1
2
3
4
5
6
7
8
9
10
11
12
13
14
+5V
RS
EN
D4
D5
D6
D7
1KR
R1


VBAT
PC13
PC14
PC15
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB10
PB11
NRST
3V3
GND
GND
3V3
GND
5V
PB9
PB8
PB7
PB6
PB5
PB4
PB3
PA15
PA12
PA11
PA10
PA9
PA8
PB15
PB14
PB13
PB12
TEMP
D7
D6
D5
D4
EN
RS
Arduino
BluePill
(STM32F103C8)




**3. Fluxograma**

O fluxograma que rege o funcionamento do programa está apresentado a seguir. Observe que inicialmente é feita a inicialização do LCD e no loop principal a leitura contínua da entrada analógica e em seguida a conversão para ppm, para que finalmente o LCD seja atualizado.

