Vitor Amadeu Souza

Apresentando no display

TFT

para medição de

Sensores

Programado no Arduino

© 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.

Agosto 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

A minha querida esposa Renata Leal Souza.

"Para quem tudo perde, ainda resta Deus."

Alfred de Musset

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 Arduino

- Gravação On-Board
- Comunicação Serial RS232
- Entrada de 12 V

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

Sumário

Capítulo I – Metodologia de desenvolvimento 7

1. Introdução 7

Capítulo II – Tela TFT 8

1. TFT 8

2. Pinagem 9

3. Alterando a cor de fundo 10

4. Desenhando linhas 14

5. Desenhando retângulos 16

6. Desenhando círculos 17

7. Imprimindo textos 18

8. Carregando imagens 20

Capítulo III – Medindo Álcool em ppm 22

Capítulo IV – Medindo a concentração de Amônia 33

Capítulo V – Amperímetro 48

Capítulo VI – Anemômetro 60

Capítulo VII – Barômetro 68

Capítulo VIII – Medindo a concentração de Benzeno 76

Capítulo IX – Célula de Carga 92

Capítulo X – Medição de Monóxido de Carbono 105

Capítulo XI – Medição de Dióxido de Carbono 122

Capítulo XII – Contador Geiger 139

Capítulo XIII – Dinamômetro 160

Capítulo XIV – Medindo o fluxo de massa 169

Capítulo XV – Gaussímetro 181

Capítulo XVI – Medindo a concentração de GLP 195

Capítulo XVII – Medindo a concentração de H_2 208
Capítulo XVIII – Medição de umidade 222
Capítulo XIX – Luxímetro 230
Capítulo XX – Medindo a concentração de Ozônio 241
Capítulo XXI – Termômetro com NTC 256
Capítulo XXII – Medindo a radiação ultravioleta 264
Capítulo XXIII – Medição de corrente de 4 a 20 mA 280
Capítulo XXIV – Decibelímetro 289
Capítulo XXV – Voltímetro 306

Capítulo I
Metodologia de desenvolvimento

1. Introdução

Esta literatura é uma continuação da obra *Arduino – Prático e Objetivo* (2011) e *Programação para Arduino – Avançado* (2014) do mesmo autor e editora, onde o objetivo é controlar uma tela de TFT usando o Arduino para em seguida apresentar na tela o resultado da medição dos sensores listados no sumário da obra.

A placa didática utilizada foi a Arduino UNO em conjunto com um display TFT que faz uso do controlador ST7735, onde tais kits e acessórios estão à venda no site www.cerne-tec.com.br.

Capítulo II
Tela TFT

1. TFT

O TFT (Thin-Film-Transistor liquid-crystal display) utilizado nesta experiência é o modelo cuja imagem está apresentada a seguir.



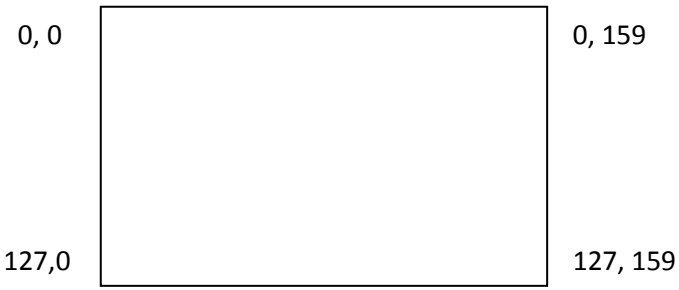
Fonte: https://www.arduino.cc/en/Guide/TFT#toc3

Este display possui uma tela de 1,8º e possui resolução 160x128 pixels (HxV). Utiliza um controlador, cujo código é ST7735 e a alimentação é feita em 5V.

Outros modelos podem ser utilizados, desde que o controlador seja compatível. O software Arduino a partir da versão 1.0.5 já dispõe de bibliotecas para acessá-lo.

A tela de TFT citada possui encaixe para cartão SD CARD, onde imagens podem ser gravadas no formato bitmap (bmp) para serem carregadas na tela. Para isso, faz-se necessário habilitar a conexão através do barramento SPI com o cartão. Este display pode ser encaixado como um shield a placa Arduino.

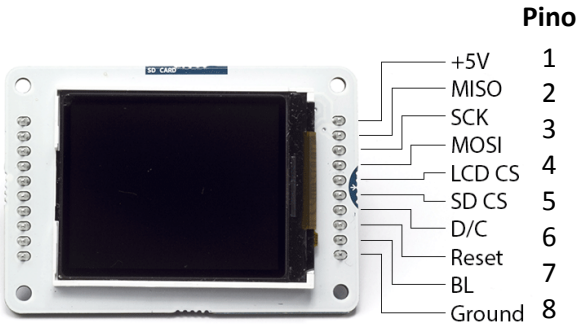
Por ter uma resolução de 160x128 pixels, podemos ver esta tela como um plano cartesiano, cujos pontos estão apresentados a seguir.



A resolução de cada pixel do display é de 16 bits, sendo 5 bits para R (Red-Vermelho) e B (Blue-Azul) e 8 para G (Green-Verde).

2. Pinagem

A pinagem deste display está apresentada abaixo.



Fonte: https://www.arduino.cc/en/Guide/TFT#toc3

A tela é alimentada em 5V através dos pinos 1 e 8. Os pinos MISO (2), SCK(3), MOSI(4) fazem a comunicação SPI entre o Arduino e o display e cartão SD. A seleção de comunicação se será feita com o TFT ou SD depende dos pinos digitais LCD CS (5) e SD CS (6). Os pinos DC e RESET também devem ser utilizados para conectar o TFT ao Arduino.

Quaisquer pinos podem ser utilizados para CS, DC e RESET, no entanto, para a comunicação SPI recomenda-se utilizar os pinos default para esta comunicação, disponível no conector ICSP da placa Arduino em razão da sua maior velocidade de transação.

3. Alterando a cor de fundo

Este experimento troca a cor da tela (background) de acordo com o parâmetro passado através do método `background` após a

inicialização do objeto `tela`. Acompanhe a seguir a listagem do código.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,0,0);
}

void loop()
{

}
```

Inicialmente são feitas as inclusões referentes às bibliotecas de comunicação SPI e TFT. Em seguida, os pinos referentes à conexão com a tela gráfica como LCD_RS, DC e RESET são definidos no programa e o objeto `tela` é criado em função dos pinos definidos anteriormente.

No bloco `setup()` o TFT é inicializado pelo método begin e a cor de fundo do display é alterado pelo método `background()` já que o parâmetro passado é 0,0,0. Tal parâmetro refere-se às cores

R, G e B (vermelho, verde e azul) onde cada parâmetro pode ser alterado de 0 a 255 em que quanto mais próximo de 255 maior a intensidade da cor especificada e quanto mais próximo de 0 maior é a sua ausência. Ao passar o parâmetro 0,0,0 foi informada a ausência das três cores fundamentais, logo a tela fica escura.

Para deixar a tela com o fundo branco, teste o código a seguir.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
}

void loop()
{

}
```

Diferente do exemplo anterior este deixa todas as cores presentes, onde o resultado é o fundo branco. Acompanhe a seguir o programa para deixar toda a tela vermelha.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,0,0);
}

void loop()
{

}
```

O próximo exemplo deixa a tela verde.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,255,0);
}

void loop()
{
```

```
}
```

E o próximo exemplo com a cor azul.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(0,0,255);
}

void loop()
{

}
```

4. Desenhando linhas

Ao criar o objeto tela uma série de métodos são disponibilizados. Uma delas é a `line(`x_i`,`y_i`,`x_f`,`y_f`)` que possibilita o desenho de uma linha de acordo o ponto inicial `(`x_i`,`y_i`)` e final `(`x_f`,`y_f`)` informado como parâmetro. O método `stroke(R,G,B)` permite a alteração da cor no qual a linha será impressa.

Acompanhe a seguir o programa para desenhar uma linha diagonal vermelha na tela do TFT.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(255,0,0);
  tela.line(0,0,159,127);
}

void loop()
{

}
```

O próximo exemplo desenha uma linha horizontal verde na tela do TFT.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,255,0);
  tela.line(0,127,159,127);
}

void loop()
{

}
```

5. Desenhando retângulos

O método `rect(`x_i`,`y_i`,largura,altura)` possibilita o desenho de um retângulo a partir do ponto inicial `(`x_i`,`y_i`)` de acordo com a `largura` e `altura` especificadas como parâmetro. O método `stroke(R,G,B)` permite a alteração da cor da linha que envolve o retângulo enquanto o método `fill(R,G,B)` a sua cor interna.

Acompanhe a seguir o programa para desenhar um retângulo com cor interna vermelha e linha de contorno azul.

```
#include <SPI.h>
#include <TFT.h>
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,0,255);
  tela.fill(255,0,0);
  tela.rect(0,0,50,50);
}

void loop()
{

}
```

6. Desenhando círculos

O método `circle(`x_i`,`y_i`,raio)` possibilita o desenho de um círculo a partir do ponto inicial (x_i, y_i) de acordo com o `raio` especificado no parâmetro. O método `stroke(R,G,B)` permite a alteração da cor da linha que envolve o círculo enquanto o método `fill(R,G,B)` a sua cor interna.

Acompanhe a seguir o programa para desenhar um círculo com cor interna vermelha e linha de contorno azul.

```
#include <SPI.h>
#include <TFT.h>
#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
```

```
void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(0,0,255);
  tela.fill(255,0,0);
  tela.circle(50,50,20);
}

void loop()
{

}
```

7. Imprimindo textos

O método `text(texto,`x_i`,`y_i`)` possibilita escrever um `texto` a partir do ponto inicial `(`x_i`,`y_i`)`. O tamanho da letra por default é de 5 pixels de comprimento e 8 de altura.

Acompanhe a seguir o programa para escrever um texto na tela TFT.

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


#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
```

```
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.text("Cerne Tec",0,50);
}

void loop()
{

}
```

O método `setTextSize(tamanho)` possibilita aumentar a altura da letra de 10 em 10 pixels. Por exemplo, caso `setTextSize(1), setTextSize(2) e setTextSize(3)` a altura será de 10, 20 e 30 pixels respectivamente, como mostra o próximo exemplo.

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

#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
  tela.background(255,255,255);
  delay(1000);
  tela.stroke(255,0,0);
  tela.setTextSize(1);
  tela.text("Cerne",0,0);
  tela.setTextSize(2);
```

```
  tela.text("Tecnologia",0,20);
  tela.setTextSize(1);
  tela.text("cerne-tec.com.br",0,50);
}

void loop()
{

}
```

8. Carregando imagens

O próximo exemplo mostra como carregar uma imagem proveniente do SD CARD. Tal imagem deve estar no formato bmp e ter resolução de 160x128 pixels. Para este exemplo foi considerada a imagem salva com o nome arduino.bmp.

Observe que foi declarado um novo objeto chamado imagem proveniente da classe PImage além de um pino adicional para controle de seleção do SD CARD poder ser utilizado. No bloco `setup()` a imagem é carregada usando o método image do objeto `tela` declarado anteriormente.

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

#define SD_CS    11
#define LCD_CS   10
#define DC       9
```

```
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);
PImage imagem;

void setup()
{
  while (!SD.begin(SD_CS));
  tela.begin();
  tela.background(255,255,255);
  imagem = tela.loadImage("arduino.bmp");
  if (imagem.isValid() == true)
    tela.image(imagem, 0, 0);
}

void loop()
{

}
```

Neste programa o mesmo apenas continua sua execução caso não haja erro na conexão com a memória SD CARD, identificado através do comando `while (!SD.begin(SD_CS))`. A imagem somente é carregada caso esteja na formatação correta.

Capítulo III
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 Arduino 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.

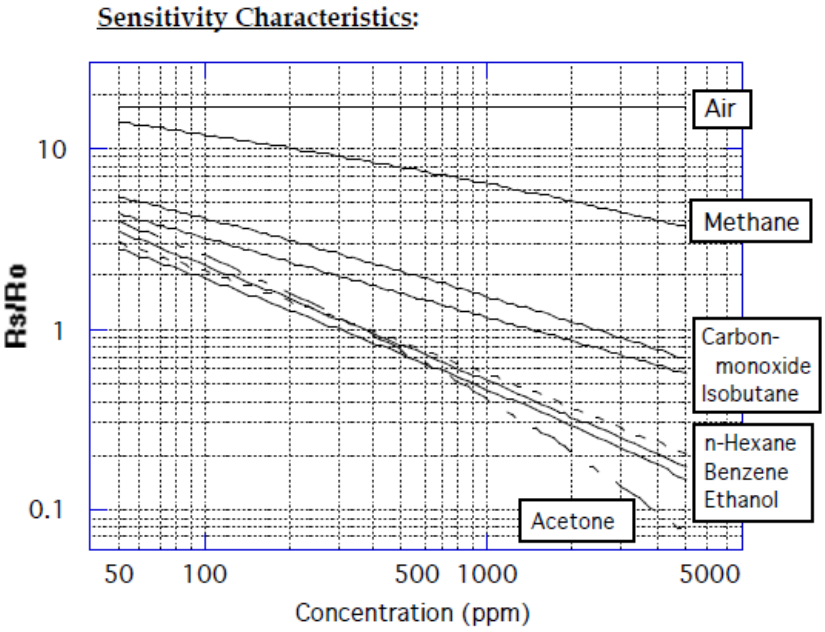


Figura 2.1: Resposta de acordo com a concentração de solvente
Fonte: TGS 822 Datasheet

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.

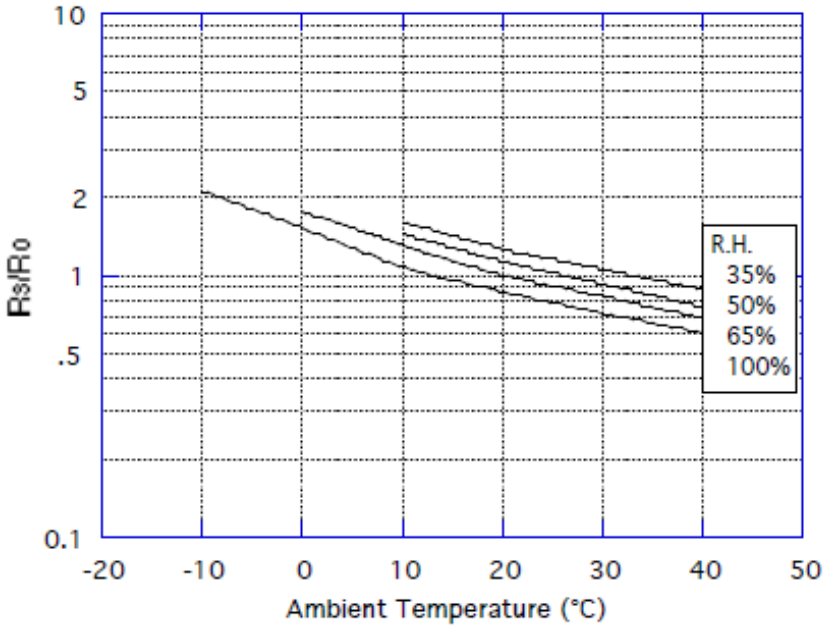


Figura 2.2: Influência da variação de umidade e temperatura
Fonte: TGS 822 Datasheet

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.

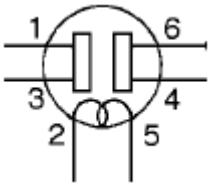


Figura 2.3: Pinagem do TGS 822
Fonte: TGS 822 Datasheet

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 Arduino mensurá-la.

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

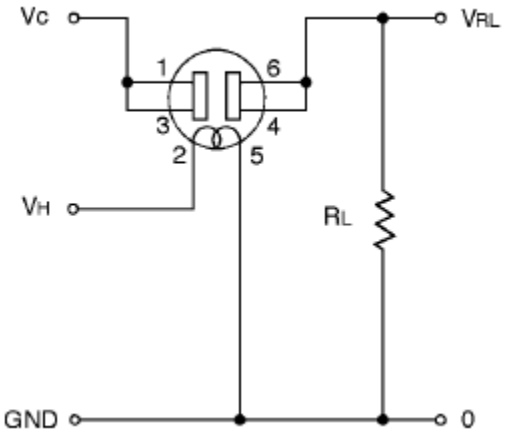


Figura 2.4: Circuito típico com TGS 822
Fonte: TGS 822 Datasheet

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 Arduino (CAD), estando na faixa de 0 a 5 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.

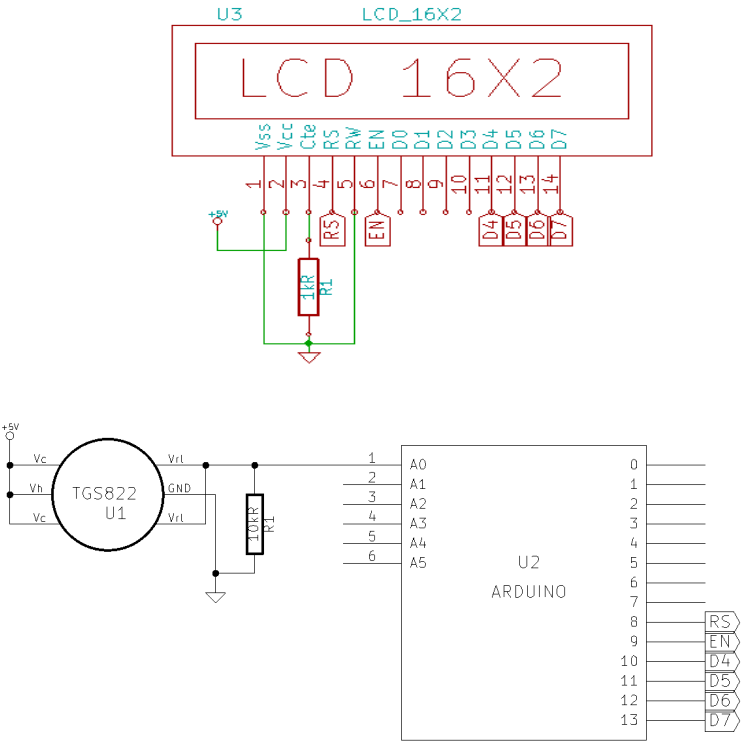
Chamando de *x* a relação entre R_S/R_0 e *ppm* a concentração, a função a seguir descreve como é encontrada a concentração.

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

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.

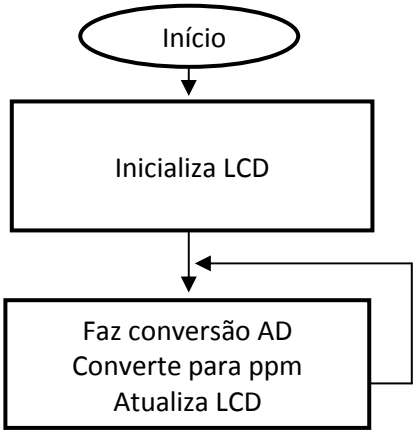
2. Esquema elétrico

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 Arduino.



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.



4. Código fonte

A seguir o código fonte em Arduino que mede a concentração em ppm do Álcool em determinado meio através do sensor TGS 822 e apresenta no LCD o resultado da medição em ppm. Após a leitura analógica do canal 0 encontra-se a relação R_S/R_0 para se aplicar a função apresentada, passando como parâmetro esta relação. O retorno da função é a concentração em ppm para que em seguida, este resultado seja apresentado no LCD.

```
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *          Cerne Tecnologia          *
 *        www.cerne-tec.com.br        *
 *    Projetos em Arduino - Álcool    *
 *  *  *  *  *  *  *  *  *  *  *  *  */
#include <LiquidCrystal.h>
LiquidCrystal lcd(8,9,10,11,12,13);
```

```
/*  *  *  *  *  *  *  *  *  *  *  *  *
 *       Função de inicialização       *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void setup(void)
{
  lcd.begin(16,2);
}

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *           Função de loop            *
 *  *  *  *  *  *  *  *  *  *  *  *  */
void loop(void)
{
  lcd.clear();
  lcd.setCursor(4,0);
  lcd.print(le_ppm_alcool());
  lcd.print(" ppm");
  delay(1000);
}

/*  *  *  *  *  *  *  *  *  *  *  *  *
 *  Função para ler ppm do ambiente   *
 *  *  *  *  *  *  *  *  *  *  *  *  */
float le_ppm_alcool(void)
{
  float ppm,ad,vrl,rs,rs_ro,x;

  ad = analogRead(0);      //Lê AD de 0 a 1023
  vrl=(ad*5.0)/1023;       //Encontra Vrl
  rs=((5.0/vrl)-1)*10000; //Encontra Rs
  rs_ro=rs/5000;           //Encontra relação Rs/Ro
  x=log(rs_ro);          //Acha o log
  ppm=169*(x*x*x*x)-147*(x*x*x)+207*(x*x)-496*x+332;
                           //Equação para achar ppm
  return(ppm);             //Concentração em ppm
}
```

5. Objetivo

Este tópico é uma junção dos dois últimos, no qual é apresentado no TFT o resultado da medição do sensor.

6. Esquema elétrico

É o mesmo do exemplo de TFT com a adição do sensor estudado anteriormente.

7. Código fonte

Acompanhe a seguir o código proposto.

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


#define LCD_CS   10
#define DC       9
#define RESET    8

TFT tela = TFT(LCD_CS,DC,RESET);

void setup()
{
  tela.begin();
}
void loop()
{
   tela.background(255,255,255);
   tela.text(String(le_ppm_alcool()) + "ppm",0,50);
   delay(5000);
```