## Placa Arduíno MEGA 2560 – IDE – ambiente e linguagem de programação

A IDE (ambiente para desenvolvimento integrado) de desenvolvimento das aplicações para o Arduíno pode ser obtida em https://www.arduino.cc/en/software.

O ambiente da IDE está bem descrito em Arduino - Environment

Uma síntese da linguagem utilizada pode ser obtida em https://www.arduino.cc/reference/pt/.

Nesta nota iremos tratar especificamente de tópicos específicos do ambiente e da linguagem da IDE. A nota prossegue com a apresentação do uso da pinagem apresentada na nota 1, aplicando as funções de entrada e saída apresentadas na nota 2.

### Ambiente da IDE

A Figura 1 apresenta a tela principal da IDE.

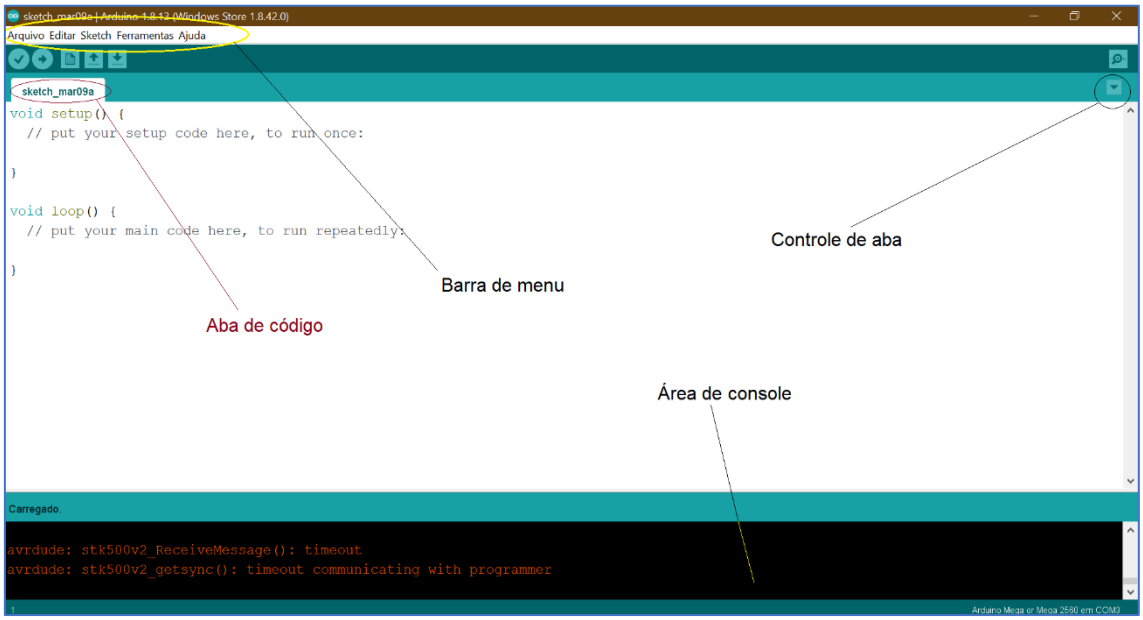


*Figura 1 – Tela inicial da IDE do Arduíno*

A barra de menu é constituída por: *Arquivo*, *Editar*, *Sketch*, *Ferramentas* e *Ajuda*. Vamos nos concentrar somente em alguns elementos da barra de menu.

Os elementos mais simples não estão discutidos. Não discutimos também os elementos que se referem ao programador. O programador é o código de inicialização da placa. Por enquanto, se desejar aprofundar os detalhes, pode começar consultando Arduino - Bootloader.

No menu Sketch, conforme apresenta a Figura 2, focaremos nos elementos detalhados a seguir.

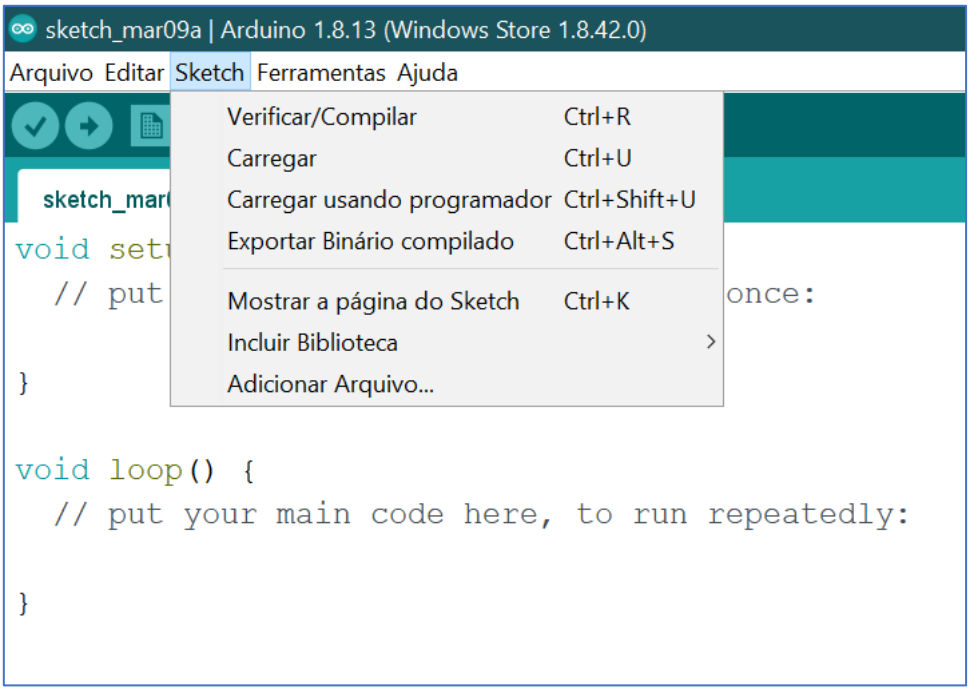


*Figura 2 – Menu > Sketch*

- *Sketch>Verificar/Compilar* – verifica erros no código, compilando-o; reporta na área do console o uso de memória e as variáveis – não carrega o código compilado na placa (sugiro testar o código antes de carregar para verificar possíveis *bugs*);
- *Sketch>Carregar* – compila e carrega o arquivo binário na placa;
- *Sketch>Exportar binário compilado* – salva um arquivo .hex, que pode ser mantido ou enviado para a placa usando outras ferramentas;
- *Sketch>Incluir bibliotecas* – adiciona uma biblioteca ao seu sketch pela inserção do comando *#include* do pré-processador da linguagem; adicionalmente, a partir deste item de menu, pode-se acessar o *Gerenciador de Biblioteca* e importar novas bibliotecas de arquivos .zip, conforme ilustra a Figura 3;

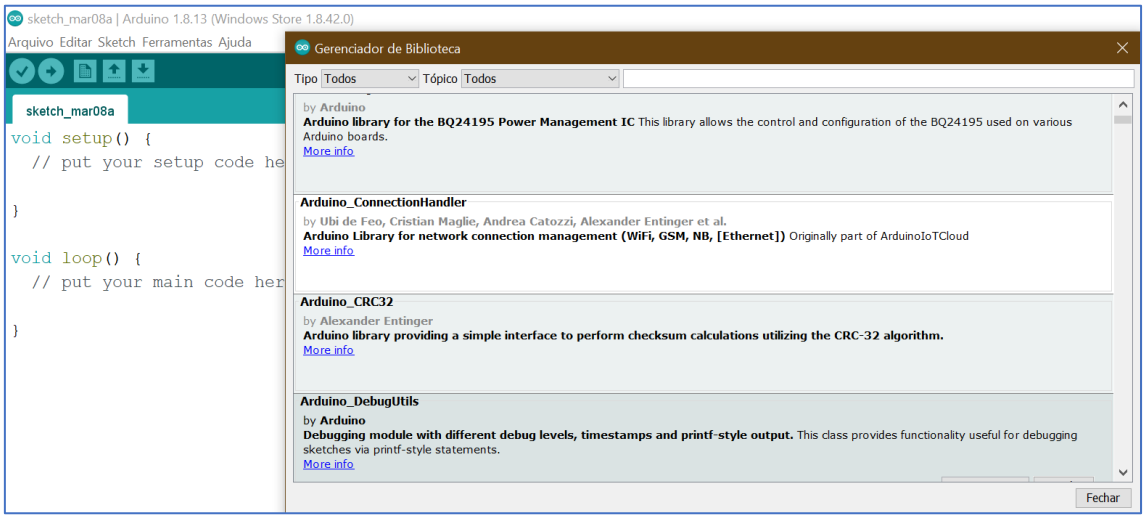


*Figura 3 – Formulário de acesso ao Gerenciador de Biblioteca, pelo menu > Sketch > Incluir Biblioteca*

- *Sketch>Adicionar Arquivo* – adiciona um arquivo fonte ao *sketch*, que é carregado como uma nova aba e pode ser apagado usando o triângulo invertido na parte superior direita da janela.

No menu Ferramentas, conforme apresenta a Figura 4, focaremos nos elementos detalhados a seguir.

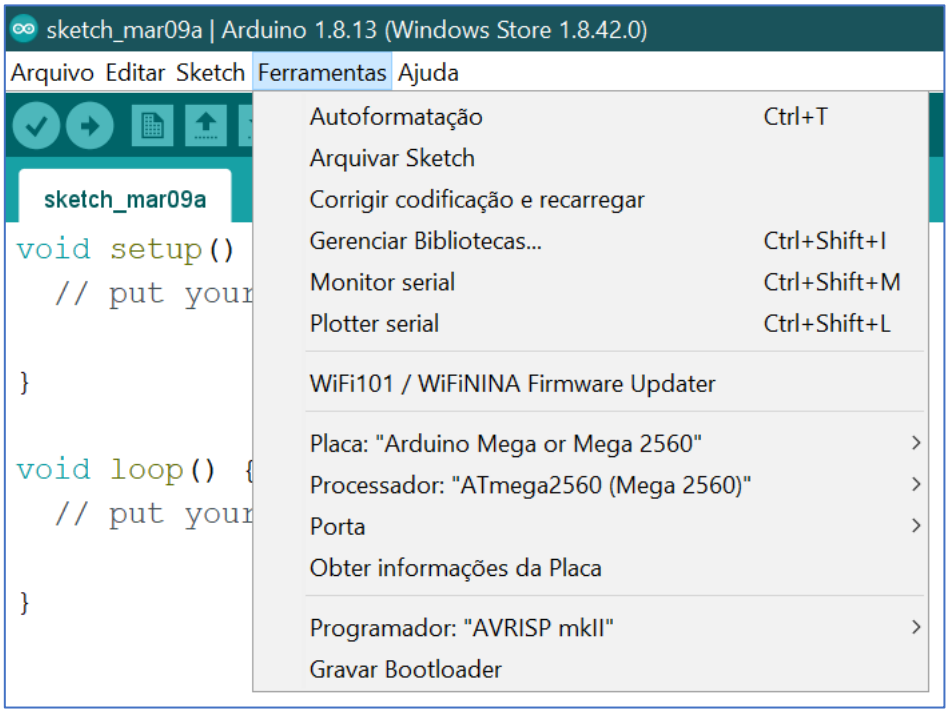


*Figura 4 – Menu Ferramentas*

- *Ferramentas>Corrigir codificação e recarregar* – corrige possíveis discrepâncias entre o mapa de codificação de caracteres do editor e mapas de caracteres de outros sistemas;

- *Ferramentas>Monitor serial* – abre uma janela de monitor serial e inicia a troca de dados com qualquer placa conectada na porta selecionada; normalmente reinicializa a placa se ela suportar a reinicialização pela porta serial – é necessário que a placa esteja alimentada e conectada;
- *Ferramentas>Plotter serial* – abre uma janela que mostra graficamente informações de comunicação serial das portas analógicas e digitais – é necessário que a placa esteja alimentada e conectada.

Os *sketchs* são disponibilizados em *Sketchbook*, uma pasta padrão de uso da IDE, que podem ser acessados em *Arquivo>Sketchbook*. A pasta pode ser configurada em *Arquivo>Preferências*.

Antes de carregar o *sketch* é necessário habilitar a porta serial correta. No Windows pode ser COM1, COM2 ou COM4, ... Para encontrar a porta adequada consultar a aplicação Gerenciador de Dispositivos do Windows.

Uma vez selecionada a porta serial e a placa carregar o *sketch* desejado. A placa irá reiniciar automaticamente.

Após realizada a conexão da placa na porta adequada automaticamente o programador é executado (código de inicialização da placa). É possível ver o LED de TX piscando. A IDE apresentará uma mensagem que o sketch foi carregado ou que houve um erro.

Para ilustrar os pontos específicos da linguagem serão utilizados dois componentes eletrônicos: (i) barra gráfica de 10 segmentos B10BS; (ii) display de 7-segmentos catodo comum - HS – 5161AS. As especificações técnicas da barra gráfica e do display de 7-segmentos pode ser obtida na Internet[1].

**A barra gráfica**

A barra gráfica é um componente simples. Trata-se de um conjunto de LEDs coloridos paralelos, distintos, encapsulados. A Figura 5 apresenta o detalhamento do circuito interno de uma barra gráfica típica.

[1] As especificações da barra gráfica estão disponíveis em https://www.usinainfo.com.br/leds/barra-grafica-de-led-10-segmentos-colorida-4827.html. O datasheet do display pode ser obtido em Display 7 Segmentos Catodo Comum - HS-5161AS Vermelho.pdf (awsli.com.br)

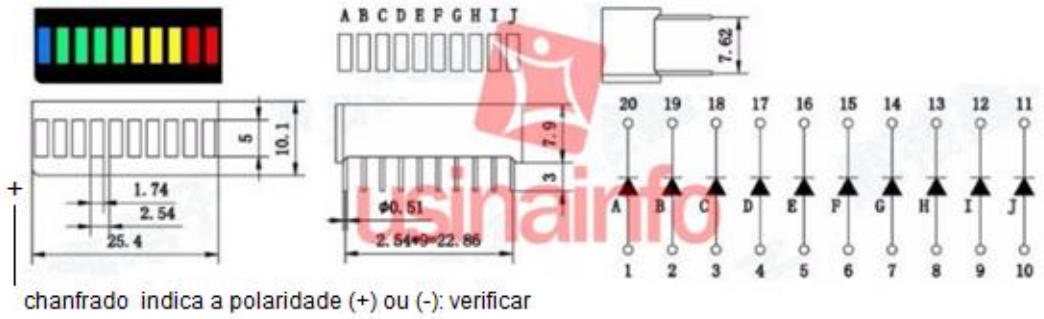


*Figura 5 – Barra gráfica: desenho com dimensões (observe a polaridade) e esquema elétrico*

As especificações técnicas da barra são as seguintes:

- Segmentos coloridos
- 10 segmentos
- Tensão reversa – 5V
- Corrente direta – 30 mA

Lembrando que é importante utilizar em série com diodos um resistor com a função de limitar a corrente e, consequentemente, não danificar o componente. É importante calcular o resistor adequado ao circuito. Pode-se utilizar a fórmula

$$Rs \geq \frac{5V - 1V}{20mA} = 165\Omega$$

Os valores decorrem do seguinte: tensão de alimentação, 5V, valor HIGH da saída digital da placa; tensão de operação do diodo, 1V (diodos de silício possuem tipicamente 0,7V); corrente de operação do diodo, 20 mA, 2/3 da corrente máxima – limite arbitrado para o projeto. Utilizaremos no circuito resistores de 200 Ω.

O esquema elétrico utilizado está apresentado na Figura 6.

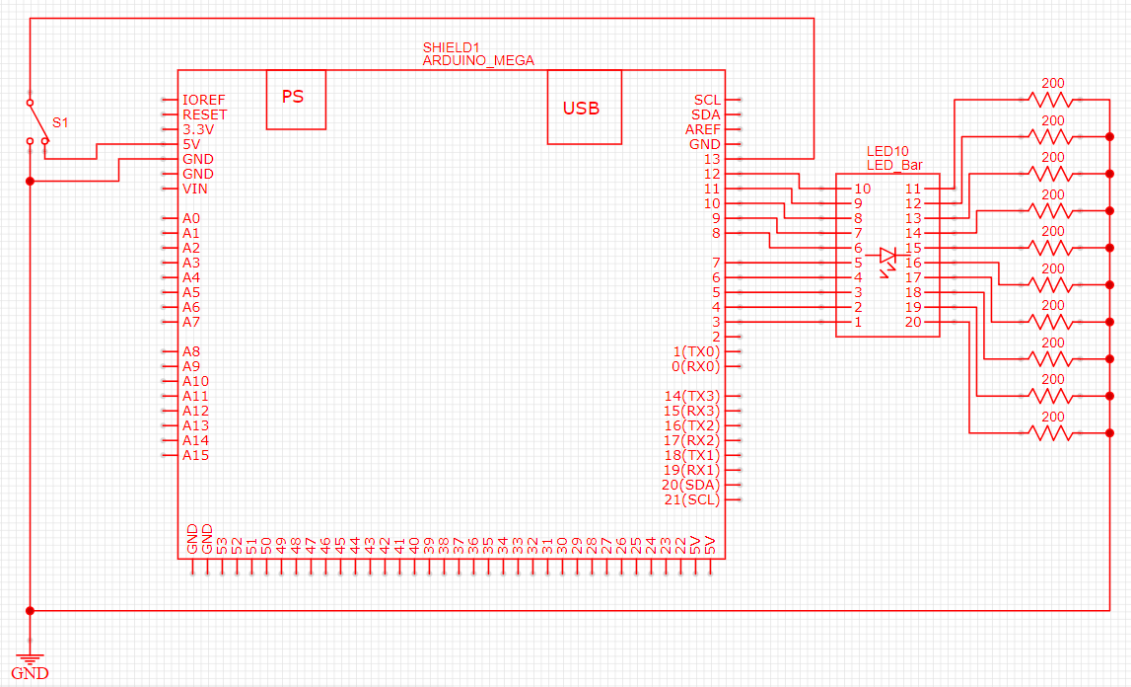


*Figura 6 – Esquema elétrico do circuito dos códigos 1 e 2*

O esquema mostra as ligações das saídas digitais 3 a 12 conectadas aos pinos 1 a 10 (anodo) da barra de LEDs, assegurando uma polarização positiva quando o sinal de saída da placa for HIGH (5V).

Os pinos 11 a 20 da barra de LEDs (catodo) estão conectados, cada um, a um resistor de 200 Ω, proporcionando uma corrente DC no diodo em polarização direta de cerca de 20 mA, bastante inferior ao limite de 30 mA.

A chave está conectada aos pinos de Gnd e 5V de saída de alimentação da placa. O sinal selecionado pela chave é o sinal de entrada do pino 13 da placa. O sinal de entrada no pino 13 da placa é que seleciona a barra de LEDs para operar como temporizador ou como uma palavra fixa de 10-bits.

A Figura 7 apresenta a montagem do protótipo na placa de *protoboard*.

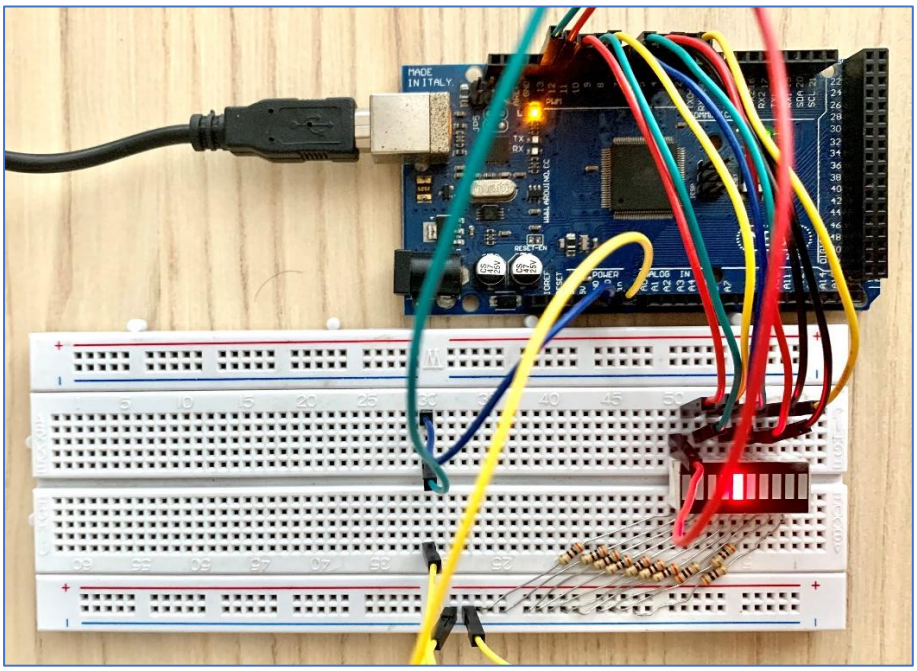

*Figura 7 – protótipo em protoboard do circuito para implementação dos códigos 1 e 2*

## O display de 7-segmentos

- Tensão de operação: 5 VDC
- Cor dos LEDs: vermelha
- Polaridade: catodo comum
- Tensão típica: 1,8 V
- Tensão reversa máxima: 5 V

Um display de 7-segmentos é um componente que é composto por sete LEDs e um ponto decimal (DP) que podem ser ligados ou desligados individualmente, de modo a formar um caractere alfanumérico. A Figura 8 ilustra os segmentos e o ponto decimal do componente.

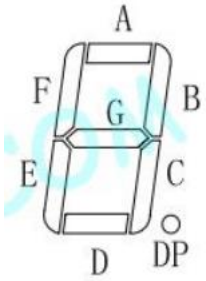


*Figura 8 – Display de 7-segmentos com identificação dos segmentos e ponto decimal (DP)*

A Figura 8 apresenta a identificação de cada um dos segmentos do componente, que varia de A a G, bem como do ponto decimal (DP).

O display a ser usado é de catodo comum. Isso significa que os LEDs que compõem cada segmento possuem o catodo conectado ao mesmo referencial de tensão. Os displays de anodo comum possuem o anodo de todos os seus LEDs constituintes ligados ao mesmo referencial.

A Figura 9 ilustra as conexões de dos LEDs do display e a ligação comum de catodo. Observe que a, b, c, d, e, f, g são os pinos de entrada do display (além do DP). O valor do caractere apresentado no display será resultado dos valores apresentados nessa pinagem.

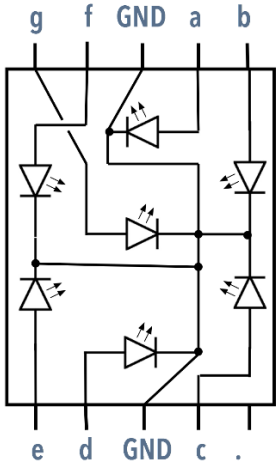


*Figura 9 – LEDs e conexão em display de 7-segmentos de catodo comum*

Os caracteres produzidos dependem da combinação na pinagem de entrada. A Tabela 1 apresenta nas colunas da esquerda a pinagem de entrada (A a G, DP), cada pino podendo assumir dois valores possíveis: HIGH (H) ou LOW (L). Apresenta o caractere produzido pelo display na coluna mais à direita.

*Tabela 1 – Pinagem do display e respectivas palavras do código de caracteres*

| A | B | C | D | E | F | G | DP | Caractere |
|---|---|---|---|---|---|---|---|---|
| H | H | H | H | H | H | L | L | 0 |
| L | H | H | L | L | L | L | L | 1 |
| H | H | L | H | H | L | H | L | 2 |
| H | H | H | H | L | L | H | L | 3 |
| L | H | H | L | L | H | H | L | 4 |
| H | L | H | H | L | H | H | L | 5 |
| H | L | H | H | H | H | H | L | 6 |
| H | H | H | L | L | L | L | L | 7 |
| H | H | H | H | H | H | H | L | 8 |
| H | H | H | L | L | H | H | L | 9 |
| H | H | H | L | H | H | H | L | A |
| H | L | L | H | H | H | H | L | b |
| | | | | | | | | ... |

A Figura 10 apresenta um esquema de ligação da utilização da placa com um display de 7-segmentos.

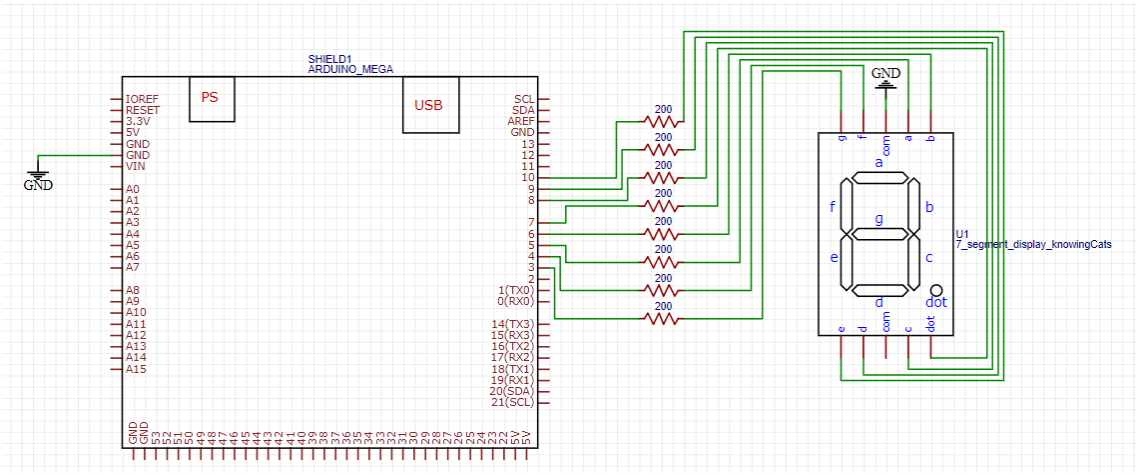


*Figura 10 – Um esquema de ligação da placa Arduíno 2560 com display de 7-segmentos*

A Figura 11 apresenta um protótipo da utilização da placa com um display de 7-segmentos em *protoboard*. Os resistores utilizados têm a finalidade de limitar a corrente sobre os diodos que constituem o display de 7-segmentos.

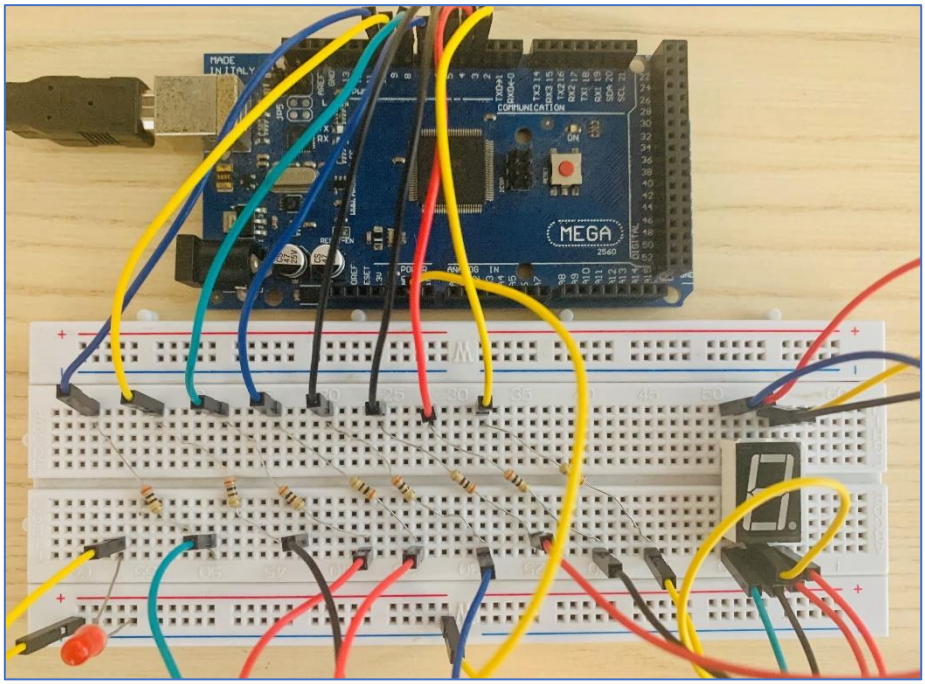


*Figura 11 – Protótipo da utilização da placa Arduíno 2560 com display de 7-segmentos.*

## Códigos para a barra gráfica na linguagem do Arduíno

Após fazer algumas pesquisas, levantei que a linguagem utilizada pela IDE é a linguagem C, com algumas adaptações. Uma avaliação preliminar me permite concordar com essa informação.  A documentação de referência pode ser obtida em https://www.arduino.cc/reference/pt/.

A linguagem de programação do Arduino pode ser dividida em três partes principais: estruturas, valores (variáveis e constantes) e funções. Todas as partes estão muito bem descritas nas referências.

Os códigos apresentados a seguir utilizam a mesma arquitetura física, apresentada no esquema.

Código 1) O circuito lê uma entrada selecionada por uma chave liga-desliga. Se a chave for posicionada na posição liga (HIGH), a barra produz como saída uma sequência colorida; se a chave for posicionada na posição desliga (LOW), a barra produz como saída uma outra sequência colorida.

*Código 1 – Seleção de uma palavra de 10-bits com chave liga-desliga*

```
// O programa ilustra a utilizacao do display de 7-segmentos
// Integra a nota 3 produzida a respeito da utilização da placa Arduíno MEGA 2560

// Declaracao de variaveis globais
int val=0;

void setup() {
 pinMode(3,OUTPUT);  //Configura o led1 - pino 3 - como saída
 pinMode(4,OUTPUT);  //Configura o led2 - pino 4 - como saída
 pinMode(5,OUTPUT);  //Configura o led3 - pino 5 - como saída
 pinMode(6,OUTPUT);  //Configura o led4 - pino 6 - como saída
 pinMode(7,OUTPUT);  //Configura o led5 - pino 7 - como saída
 pinMode(8,OUTPUT);  //Configura o led6 - pino 8 - como saída
 pinMode(9,OUTPUT);  //Configura o led7 - pino 9 - como saída
 pinMode(10,OUTPUT); //Configura o led8 - pino 10 - como saída
 pinMode(11,OUTPUT); //Configura o led9 - pino 11 - como saída
 pinMode(12,OUTPUT); //Configura o led10 - pino 12 - como saída
}

void loop() {
  val=digitalRead(13);
  if (val==HIGH){
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,LOW);
    digitalWrite(6,LOW);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);
    digitalWrite(9,LOW);
    digitalWrite(10,HIGH);
    digitalWrite(11,HIGH);
    digitalWrite(12,LOW);
  }
  else {
    digitalWrite(3,LOW);
    digitalWrite(4,LOW);
    digitalWrite(5,HIGH);
    digitalWrite(6,HIGH);
    digitalWrite(7,LOW);
    digitalWrite(8,LOW);
    digitalWrite(9,HIGH);
    digitalWrite(10,LOW);
    digitalWrite(11,HIGH);
```

```
    digitalWrite(12,LOW);
  }
}
```

Os comandos são os mesmos da linguagem C. Utiliza-se somente a estrutura condicional composta *if...else* da linguagem C. Além dela, o código 1 não contém nenhuma outra estrutura especial do C, somente as funções tratadas na nota 2.

Quanto à resposta eletrônica do circuito, observe que, embora a resposta esteja sendo apresentada graficamente para a barra de LEDs, uma solução similar poderia ser adotada para fazer uma transmissão paralela em banda base, utilizando barramento ligado aos pinos 3 a 12 da placa, de palavras de um código de 10-bits.

No funcionamento do circuito pode-se observar que após a chave ser trocada de posição os comandos dentro da função *loop()* prosseguem sendo executadas até o final. Somente após encerrada, no novo loop a resposta muda.

Código 2) O circuito é um contador, temporizado a cada 0,5 segundos, que produz um efeito visual na barra gráfica. Se a chave liga-desliga estiver selecionada na posição liga (HIGH) a sequência é apresentada nos LEDs da barra, de 1 a 10. Em caso contrário, é apresentada uma palavra binária.

*Código 2 – Temporizador com intervalos de 0,5 seg controlado por chave liga-desliga*

```
// O programa ilustra a utilizacao do display de 7-segmentos
// Integra a nota 3 produzida a respeito da utilização da placa Arduíno MEGA 2560

// Declaracao de variaveis globais
int val=0;

void setup() {
 int i; //i representa o pino da placa
 for (i=3;i<=12;i++){
  pinMode(i,OUTPUT); //Configura o segmento i-2 - pino i - como saída
 }
}

void loop() {
  val=digitalRead(13);
  int i;
  if (val==HIGH){
    delay(500);
    digitalWrite(3,HIGH);
    delay(500);
    for (i=3;i<=11;i++){
      digitalWrite(i,LOW);
      digitalWrite(i+1,HIGH);
      delay(500);
    }
    digitalWrite(12,LOW);
  }
```

```
  else {
    for (i=3;i<=12;i++){
      switch(i){
        case 5:case 6:case 9:case 11:
          digitalWrite(i,HIGH);
          break;
      }
    }
  }
}
```

Os comandos são os mesmos da linguagem C. Diferentemente do código anterior, além da estrutura *if...else*, o código 2 utiliza a estrutura de repetição *for* e a estrutura múltipla condicional *switch...case*.

Quanto à resposta eletrônica do circuito, observe que a resposta esteja sendo apresentada graficamente para a barra de LEDs. A contagem não está sendo apresentada digitalmente, que levaria à solução 0x000 a 0x009 (0000000000-0000000001-0000000010-...-00000001001), para evidenciar o funcionamento da barra de LEDs. Utilizando-se na sequência binária crescente poder-se-ia aplicar a saída como temporizador de 1024x0,5 seg (cerca de 8,5 min).

Código 3) O circuito lê serialmente uma palavra de 10-bits na entrada (b0,b1,b2,...,b9). Cada bit é lido dentro de um intervalo de 3 segundos. Bit a bit é apresentado na barra de LEDs após ser lido. Após toda a palavra ser lida o código gera uma sequência de 50 palavras binárias crescentes a partir da palavra lida, finalizando com a palavra 0x3FF.

*Código 3 – Contador binário de 10-bits*

```
// O programa ilustra a utilizacao do display de 7-segmentos
// Integra a nota 3 produzida a respeito da utilização da placa Arduíno MEGA 2560
#include <math.h>

// Declaracao de variaveis globais
int val;

void setup() {
 int i; //i representa o pino da placa
 for (i=3;i<=12;i++){
  pinMode(i,OUTPUT); //Configura o segmento i-2 - pino i - como saída
 }
}

void loop() {
  int i,j,bit;
  int vIni=0;
  for (i=0;i<=9;i++){//Lê a sequencia de bits
   val=digitalRead(13);
   if (val==HIGH)//Gera a palavra decimal equivalente à palavra de 10-bits lida
    vIni+=1*pow(2,i);
   delay(3000);
   digitalWrite(i+3,val);//Gera a saida no pino i+3 do bit i do valor lido
  }
  delay(4000);
```

```
  vIni++;//Incrementa o valor lido
  for (j=0;j<=49;j++){//Gera a sequência de 50 palavras de 10-bits e apresenta na barra de
LEDs
    for (i=9;i>=0;i--){//Converte a palavra decimal em digital
      bit=vIni>>i;//Determina o bit resultante do deslocamento
      if (bit&1)//Gera a saída no pino i+3 do bit i da palavra de 10-bits
        digitalWrite(i+3,HIGH);
      else
        digitalWrite(i+3,LOW);
    }
    delay(500);
    vIni++;
  }
  for (i=0;i<=9;i++){//Ilumina toda a barra após gerar a sequência
    digitalWrite(i+3,HIGH);
  }
  delay(5000);
  for (i=0;i<=9;i++){//Apaga toda a barra antes de ler a próxima palavra de 10-bits
    digitalWrite(i+3,LOW);
  }
}
```

O código utiliza a biblioteca *math.h* da linguagem C. Entre as funções da biblioteca, utiliza em particular a função *pow(base,expoente)*. A primeira parte do código se encarrega de determinar o valor decimal da palavra binária e gerar a saída para a barra de LEDs.

Na seção subsequente do código, as 50 palavras seguintes à palavra binária de entrada são geradas em ordem crescente e carregadas nas saídas da barra de LEDs.

Observe que o número decimal é incrementado em cada loop e o binário equivalente é calculado e a palavra é carregada na saída. Para isso, utiliza-se o operador *x>>y*, que realiza a operação de deslocamento de x à direita y bits. Além disso, utiliza-se o operador *x&y* - que realiza a operação lógica de conjunção de x e y (and) bit a bit, para testar se o bit é 0 ou 1.