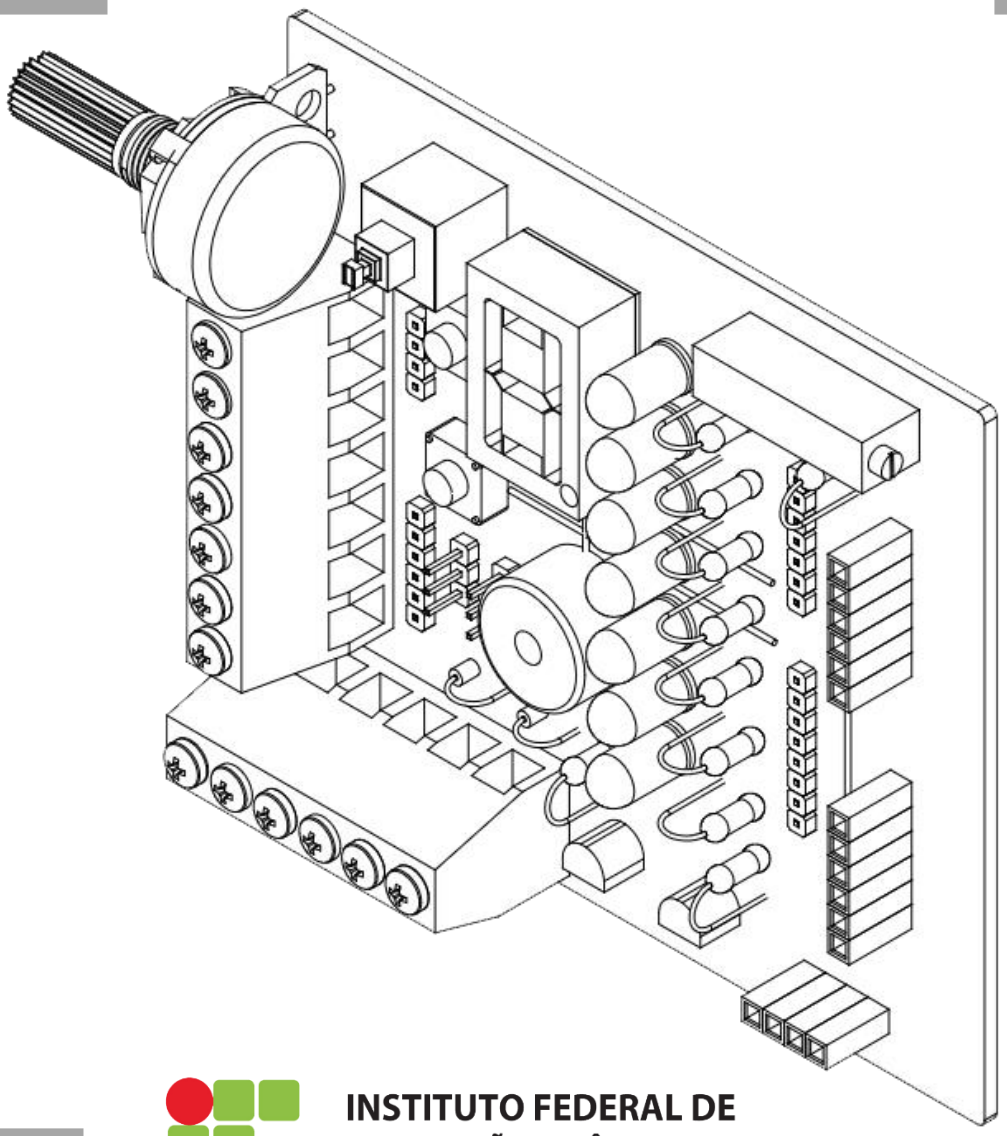
MANUAL DO SHIELD EDU-IFSP



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Câmpus Guarulhos

AVISO

Este manual e quaisquer exemplos aqui contidos são fornecidos na forma em que se encontram, sendo elaborados pelos próprios colaboradores do GERSE – Grupo de Estudos em Robótica e Sistemas Embarcados para fins educacionais.

Sobre o Grupo

Criado em 1 de agosto de 2016, o GERSE – Grupo de Estudos em Robótica e Sistemas Embarcados é um grupo de robótica independente do IFSP – Campus Guarulhos, no qual tem como objetivo principal desenvolver e promover o conhecimento de forma ampla e aberta, entre alunos e comunidade.

GERSE – Grupo de Estudos em Robótica e Sistemas Embarcados

http://www.gerserobotica.com

gerse.robot@gmail.com

Instituto Federal de Educação, Ciências e Tecnologia de São Paulo – Campus Guarulhos

Revisão em março de 2017.

Colaboradores:

Editores:

Gabriel de Oliveira Ferraz

Rogério Daniel Dantas

Imagens:

Eduardo Lima

Programa teste:

Pedro Igor Borçatti

Revisão:

Cláudia Yapuchura Saire

Sumário

Introdução

O Shield Edu-IFSP é uma placa didática criada para a plataforma Arduino UNO de baixo custo, desenvolvida para o ensino de lógica de programação aplicada à robótica, cujo principal objetivo é facilitar o ensino de lógica de programação.

Para isso o Shield Edu-IFSP disponibiliza uma variedade de periféricos de entrada e saída, como: botões, LED's, display de 7 segmentos, LCD e entre outros para serem utilizados juntamente com a placa Arduino.

Tendo em vista que seu principal objetivo é o ensino de lógica de programação, o Shield oferece a oportunidade de pessoas que não tenham o conhecimento prévio de eletrônica a iniciarem seus estudos de lógica de programação, assim a placa poderá ser utilizada futuramente para ensinar programação, sendo voltado para um vasto público.

Dados Técnicos

O Shield Edu-IFSP contém os seguintes componentes para uso:

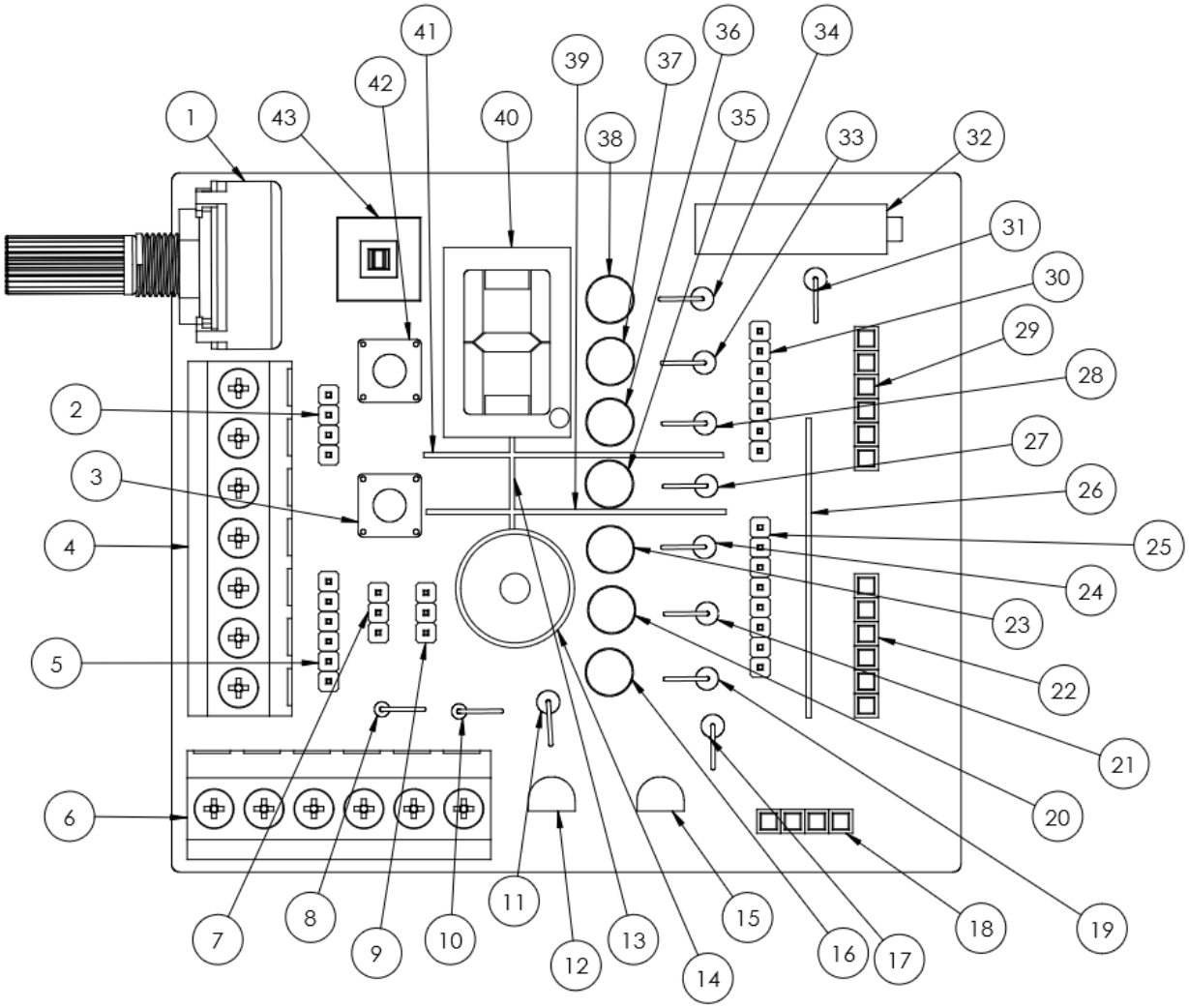


Figura 1 -Shield Edu-IFSP.

Legenda:			
1	Potenciômetro 10k	7	Barramento para conexão
2	Barramento para conexão	8	Diodo 1N4148
3	Chave Táctil	9	Barramento para conexão
4	Saída de uso geral	10	Diodo 1n4148
5	Barramento para conexão	11	Resistor 220 Ohm
6	Saída de uso geral	12	Transistor BC337

13	Jumper (conexão)	29	Entrada LCD
14	Buzzer	30	Barramento para conexão
15	Transistor BC337	31	Resistor 220 Ohm
16	Led azul	32	Trimpot 10k
17	Resistor 220 Ohm	33	Resistor 330 Ohm
18	Entrada Bluetooth	34	Resistor 330 Ohm
19	Resistor 330 Ohm	35	Led vermelho
20	Led verde	36	Led verde
21	Resistor 330 Ohm	37	Led amarelo
22	Entrada LCD	38	Led vermelho
23	Led amarelo	39	Jumper (conexão)
24	Resistor 330 Ohm	40	Display de 7 Seg. Cátodo comum
25	Barramento para conexão	41	Jumper (conexão)
26	Jumper (conexão)	42	Chave Táctil
27	Resistor 330 Ohm	43	Chave com retenção
28	Resistor 330 Ohm		

Utilização dos Botões

Os botões do Shield Edu-IFSP estão ligados internamente para que se utilize a configuração PULLUP.

Os Pull-Ups são utilizados para evitar flutuação em pinos configurados como entradas (INPUT). Em geral, é necessário implementar externamente, mas o Arduino possui Pull-Ups implementados internamente em todos os pinos.

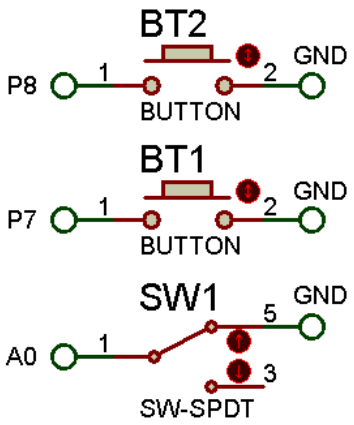


Figura 2 - Diagrama da ligação dos botões

Disposição dos pinos

A tabela a seguir mostra a relação dos pinos do Arduino com o uso dos componentes do Shield.

Saída Digital	**LED's**	**Display de 7 seg.**	**LCD**
Pino D02	Azul	G	RS
Pino D03	Verde	F	E
Pino D04	Amarelo	E	D4
Pino D05	Vermelho	D	D5
Pino D11	Verde	C	D6
Pino D12	Amarelo	B	D7
Pino D13	Vermelho	A	-
Saída Digital	**Saída de UG**	**Driver**	**-**
Pino D09	PWM 1	Transistor	-
Pino D10	PWM 2	Transistor	-
Pino D06	Buzzer/ UG	I/O	-
Entrada Digital	**Botões**	**-**	**-**
Pino D07	Chave Táctil	-	-
Pino D08	Chave Táctil	-	-
Pino A00	Chave c/ retenção	-	-
Entrada Analógica	**Potenciômetro**	**-**	**-**
Pino A01	Pot.	-	-
Comunicação	**Serial**	**-**	**-**
Pino D00	RX	-	-
Pino D01	TX	-	-
Entrada/ Saída	**Uso Geral**	**-**	**-**
Pino A02	UG	-	-
Pino A03	UG	-	-
Pino A04	UG	-	-
Pino A05	UG	-	-
Placa Versão VER.7 – Final			

- **UG – Pino de Uso Geral.**

Abaixo segue o esquema da disposição dos pinos, de acordo com os componentes do Shield Edu-IFSP.

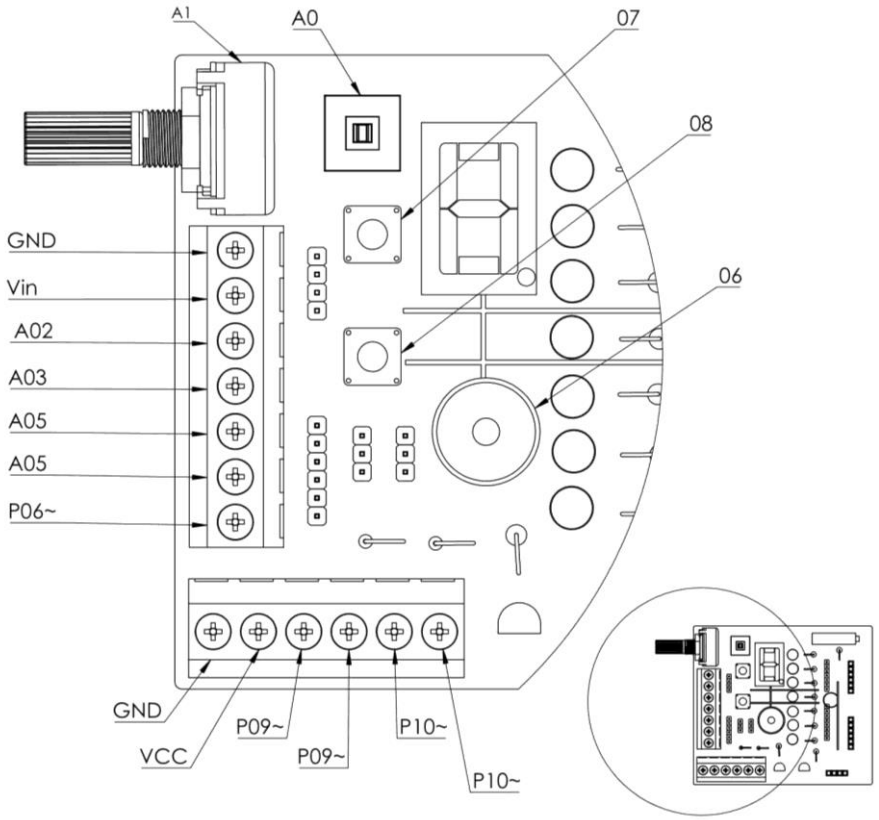


Figura 3 - Disposição dos pinos do Shield.

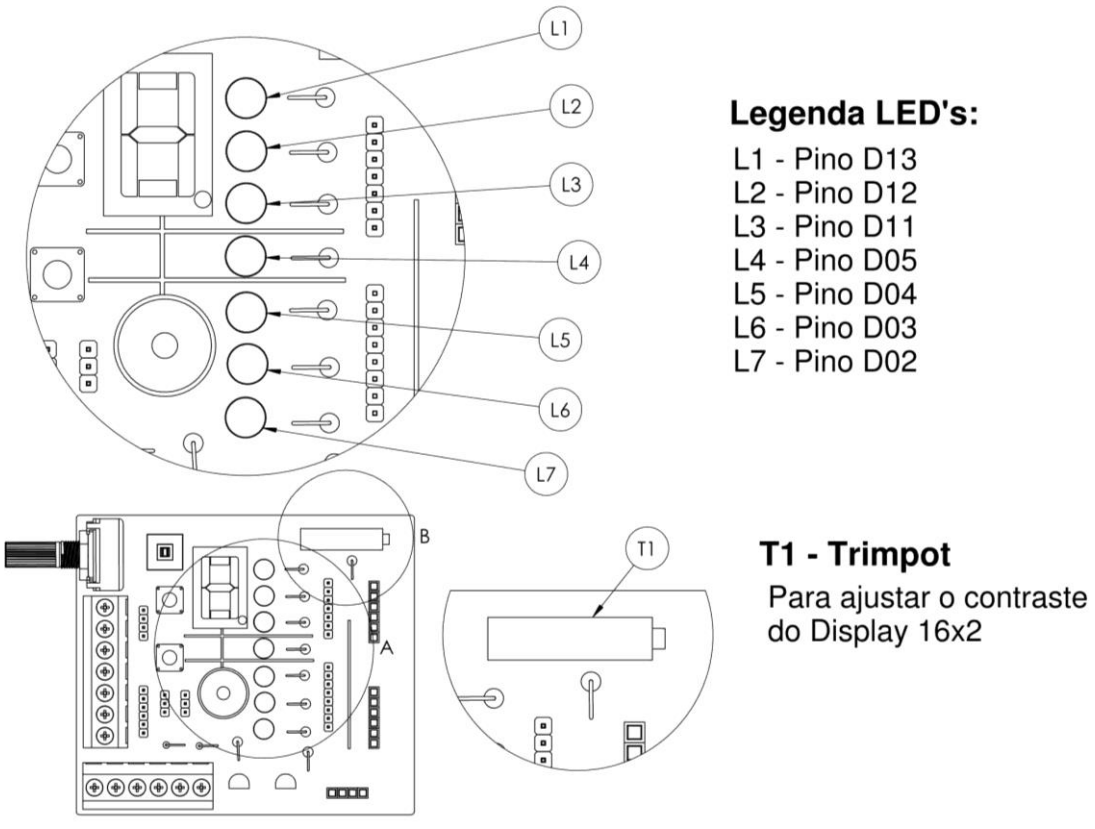


Figura 4 - Disposição dos pinos dos LED's.

Jumpers

Para melhor utilização dos pinos de entrada e saída do Arduino, o Shield Edu-IFSP possui dois jumpers, onde você pode alternar entre os LED's e o Display de 7 segmentos e entre o Buzzer e uma saída de uso geral PWM.

Abaixo segue o esquema de ligação dos jumpers:

Para selecionar o display de 7 segmentos, faça uma conexão do pino DISPLAY 7 SEG. com o pino GND;

Para selecionar os LED's, faça uma conexão do pino LEDS com o pino GND;

Para selecionar o Buzzer, faça uma conexão do pino BUZZER com o pino GND;

Para selecionar o pino de entrada e saída de uso geral D06 PWM do Arduino, faça uma conexão do pino P06 com o pino GND.

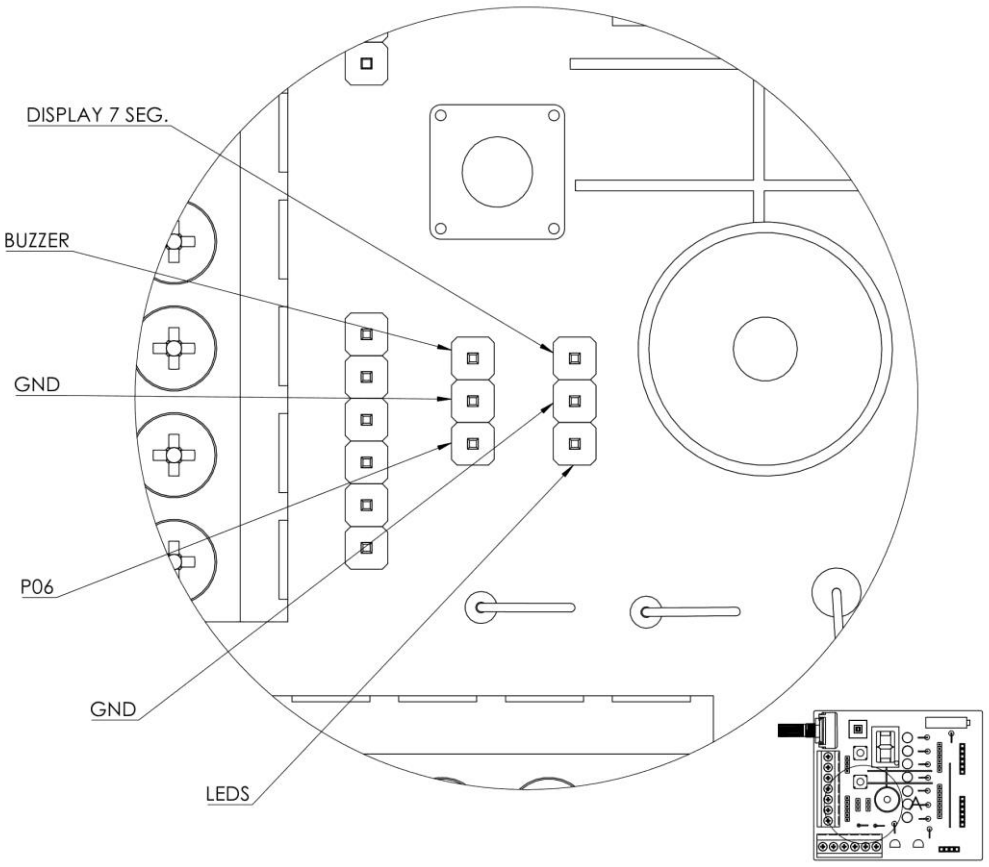


Figura 5 - Conexões do jumper.

Display de 7 Segmentos

A seguir temos o esquema do Display de 7 segmentos (Cátodo comum) da placa Shield Edu-IFSP.

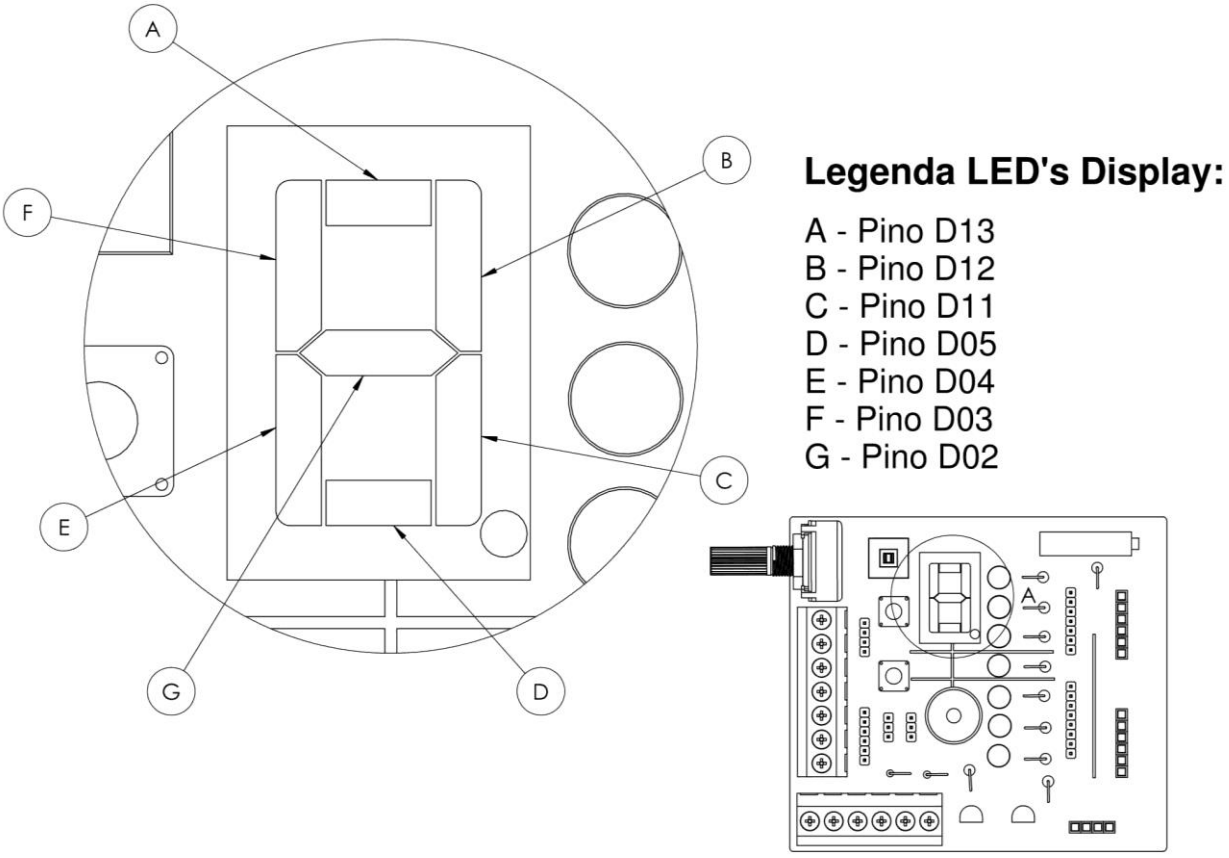


Figura 6 - Disposições dos pinos do Display de 7 segmentos.

Controlando LCD 16x2 com o Shield Edu-IFSP

Esse **display LCD** tem 16 colunas e 2 linhas, com backlight (luz de fundo) azul e letras na cor branca. Para conexão, são 16 pinos, na qual usamos 12 para uma conexão básica, já incluindo as conexões de alimentação (pinos 1 e 2), backlight (pinos 15 e 16) e contraste (pino 3).

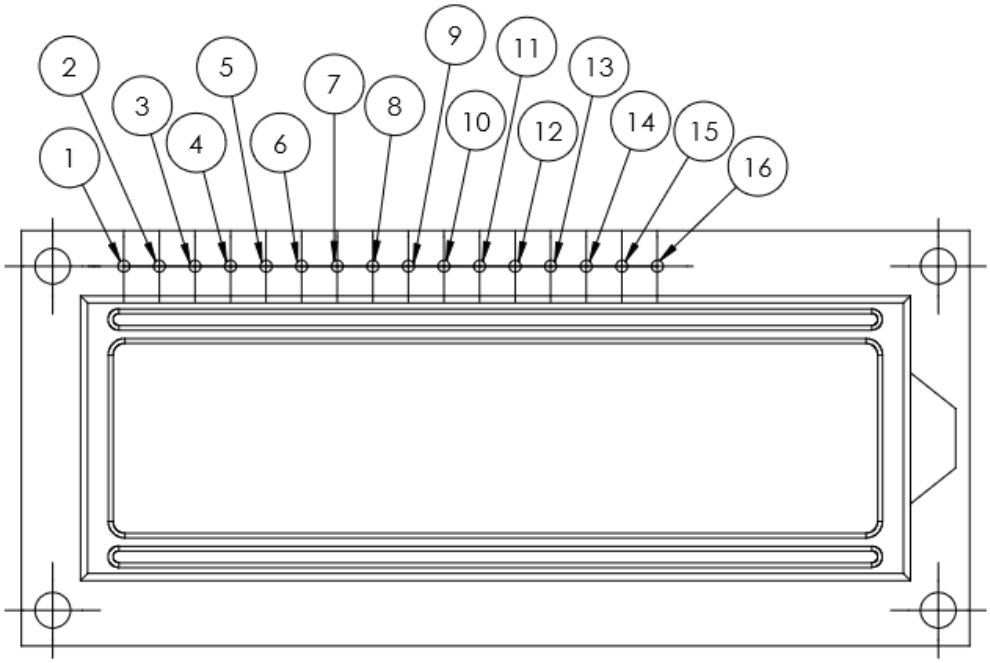


Figura 7 - Display LCD 16x2.

Conexões LCD 16x2 – HD44780		
Pino LCD	**Função**	**Ligação no Shield Edu-IFSP**
1	Vss	GND
2	Vdd	Vcc 5V
3	V0	Pino central potenciômetro
4	RS	Pino D02 Arduino
5	RW	GND
6	E	Pino D03 Arduino
7	D0	Não conectado
8	D1	Não conectado
9	D2	Não conectado
10	D3	Não conectado
11	D4	Pino D04 Arduino
12	D5	Pino D05 Arduino
13	D6	Pino D11 Arduino
14	D7	Pino D12 Arduino
15	A	Vcc 5V
16	K	GND

Programa de controle do LCD:

O controle desse display pode ser feito utilizando-se a biblioteca **LiquidCrystal**, já embutida na IDE do Arduino.

No início do programa, definimos os pinos que serão utilizados pelo display, nesse formato:

LiquidCrystal lcd (<pino RS>, <pino enable>, <pino D4>, <pino D5>, <pino D6>, <pino D7>);

Na utilização do Shield Edu-IFSP, definimos da seguinte maneira:

LiquidCrystal lcd (2, 3, 4, 5, 11, 12);

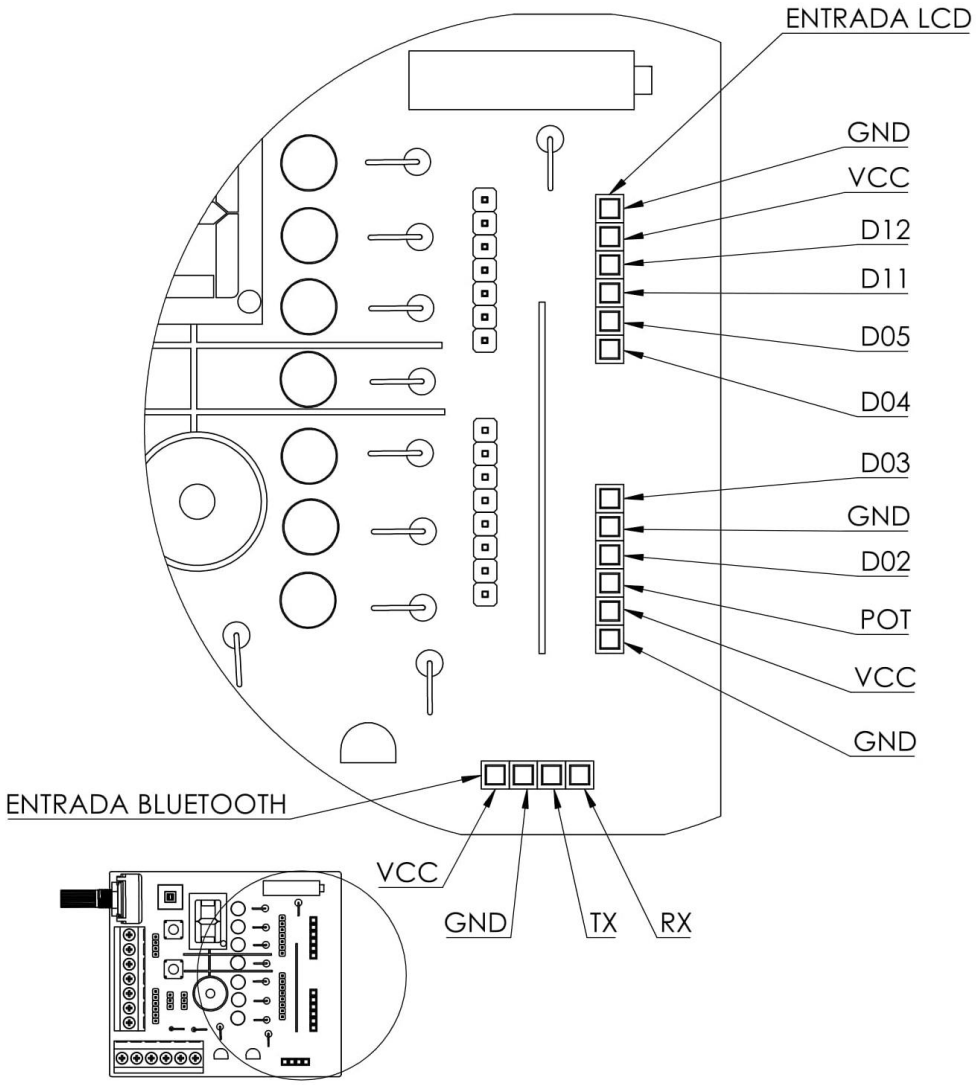


Figura 8 - Disposição dos pinos do LCD e Bluetooth,

Módulo Bluetooth HC-06

O módulo Bluetooth HC-06 é usado para comunicação wireless entre o Arduino e algum outro dispositivo com bluetooth. As informações recebidas pelo módulo são passadas ao Arduino via comunicação serial.

Abaixo segue o esquema de ligação do módulo Bluetooth HC-06.

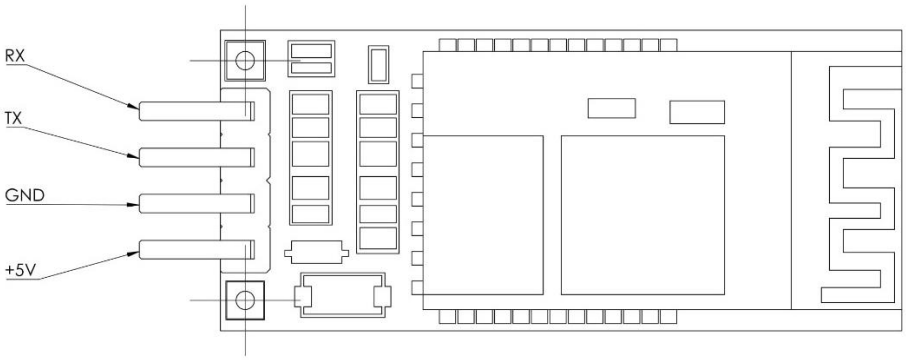


Figura 9 - Módulo Bluetooth HC-06.

Programa Teste

Esse programa realiza o teste de funcionamento dos periféricos do Shield Edu-IFSP.

Botão 1: quando pressionado mostra uma numeração de 0 a 9 no display de 7 segmentos, esse valor é referente a posição do potenciômetro.

Botão 1: quando solto imprime “< GERSE >” no display LCD 16x2.

Botão 2: quando pressionado escreve um valor PWM no pino do Buzzer, esse valor é referente a posição do potenciômetro.

Botão 3: quando pressionado e **Botão 1** solto escreve um valor PWM nos pinos D09 e D10 do Arduino e no display LCD 16x2, esse valor é referente a posição do potenciômetro.

Dependendo da posição dos jumpers serão os LED’s que funcionaram ao em vez do display de 7 segmentos.

Programa de Testes

```
/*
*Programa teste do Shield Edu-IFSP
*
*GERSE – Grupo de Estudos em Robótica e Sistemas Embarcados
*http://www.gerserobotica.com
*gerse.robot@gmail.com
*IFSP – Campus Guarulhos
*
*Desenvolvedor:
*Pedro Igor Borçatti
*
*março de 2017
*/

#include <LiquidCrystal.h> // BIBLIOTECA PARA DISPLAY LCD

#define    A        13     // DISPLAY DE 7 SEGMENTO A OU LED A
#define    B        12     // DISPLAY DE 7 SEGMENTO B OU LED B, LCD 16X2 D7
#define    C        11     // DISPLAY DE 7 SEGMENTO C OU LED C, LCD 16X2 D6
#define    D        5      // DISPLAY DE 7 SEGMENTO D OU LED D, LCD 16X2 D5
#define    E        4      // DISPLAY DE 7 SEGMENTO E OU LED E, LCD 16X2 D4
#define    F        3      // DISPLAY DE 7 SEGMENTO F OU LED F, LCD 16X2 E
#define    G        2      // DISPLAY DE 7 SEGMENTO G OU LED G, LCD 16X2 RS

#define    POT      A1     // ADC1, POTENCIÔMETRO

#define    BUZZER   6      // BUZZER OU BORNE (PWM6)

#define    ADC2     A2     // BORNE
#define    ADC3     A3     // BORNE
#define    ADC4     A4     // BORNE
#define    ADC5     A5     // BORNE
#define    MOTOR1   9      // BORNE (BC548 - VCC BATERIA E COLETOR)
#define    MOTOR2   10     // BORNE (BC548 - VCC BATERIA E COLETOR)

#define    BT1      A0     // BOTÃO COM TRAVA
#define    BT2      7      // BOTÃO 1
#define    BT3      8      // BOTÃO 2


float      CALC  = 0;      // VARIÁVEL PARA CALCULO
byte       NUM   = 0;      // VARIÁVEL PARA PWM

boolean    bt1   = false;  // VARIÁVEL DO BOTÃO 1
boolean    bt2   = false;  // VARIÁVEL DO BOTÃO 2
boolean    bt3   = false;  // VARIÁVEL DO BOTÃO 3

boolean    cls   = false;  // FLAG PARA CONTROLE DO DISPLAY LCD 16X2

const boolean LCD7[10][7] =  // NÚMEROS DE 0 A 9 PARA DISPLAY DE 7 SEGMENTOS
              {
             // {A,B,C,D,E,F,G}
                {1,1,1,1,1,1,0},    // 0
                {0,1,1,0,0,0,0},    // 1
                {1,1,0,1,1,0,1},    // 2
                {1,1,1,1,0,0,1},    // 3
                {0,1,1,0,0,1,1},    // 4
                {1,0,1,1,0,1,1},    // 5
//CONTINUA NA PROXIMA PAGINA...
```

```
                {1,0,1,1,1,1,1},    // 6
                {1,1,1,0,0,0,0},    // 7
                {1,1,1,1,1,1,1},    // 8
                {1,1,1,1,0,1,1},    // 9
             // {A,B,C,D,E,F,G}
              };

//INICIANDO DISPLAY LCD
LiquidCrystal lcd(2, 3, 4, 5, 11, 12);

void setup()
{
  lcd.begin(16, 2);           // CONFIGURANDO TAMANHO DO DISPLAY LCD 16X2

  pinMode(A,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 13
  pinMode(B,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 12
  pinMode(C,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 11
  pinMode(D,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 5
  pinMode(E,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 4
  pinMode(F,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 3
  pinMode(G,      OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 2
  pinMode(BUZZER, OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 6
  pinMode(MOTOR1, OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 9
  pinMode(MOTOR2, OUTPUT);    // CONFIGURANDO COMO SAÍDA O PINO 10

  pinMode(BT1,INPUT_PULLUP);  // CONFIGURANDO COMO ENTRADA O PINO A0 EM PLLU UP
  pinMode(BT2,INPUT_PULLUP);  // CONFIGURANDO COMO ENTRADA O PINO 7  EM PLLU UP
  pinMode(BT3,INPUT_PULLUP);  // CONFIGURANDO COMO ENTRADA O PINO 8  EM PLLU UP
  pinMode(POT,    INPUT);     // CONFIGURANDO COMO ENTRADA O PINO A1
  //pinMode(ADC2, );          // PINO DISPONÍVEL NO BORNE
  //pinMode(ADC3, );          // PINO DISPONÍVEL NO BORNE
  //pinMode(ADC4, );          // PINO DISPONÍVEL NO BORNE
  //pinMode(ADC5, );          // PINO DISPONÍVEL NO BORNE

}

void loop()
{

    bt1 = !digitalRead(BT1);  // AQUISITANDO VALOR DO BOTÃO 1
    bt2 = !digitalRead(BT2);  // AQUISITANDO VALOR DO BOTÃO 2
    bt3 = !digitalRead(BT3);  // AQUISITANDO VALOR DO BOTÃO 3

    CALC = analogRead(POT);   // AQUISITANDO VALOR DO POTENCIÔMETRO

    if(bt1)                   // SE BOTÃO 1 APERTADO
    {
      if(cls)                 // CONTROLE DO DISPLAY LCD 16X2
      {
          lcd.clear();        // LIMPAR DISPLAY LCD 16X2
          delay(20);

          cls = false;        // FLAG DE CONTROLE P. DISPLAY LCD 16X2
      }

      NUM = (CALC/1023)*9;    // 0 -> 0   1023 -> 9
      digitalWrite(A,  LCD7[NUM][0]);
      digitalWrite(B,  LCD7[NUM][1]);
      digitalWrite(C,  LCD7[NUM][2]);
      digitalWrite(D,  LCD7[NUM][3]);
//CONTINUA NA PROXIMA PAGINA...
```

```
      digitalWrite(E,  LCD7[NUM][4]);
      digitalWrite(F,  LCD7[NUM][5]);
      digitalWrite(G,  LCD7[NUM][6]);
    }
    else          // SE BOTÃO 1 NÃO APERTADO
    {

      if(!cls)    // CONTROLE DO DISPLAY LCD 16X2
      {
          digitalWrite(A,  LOW);
          digitalWrite(B,  LOW);
          digitalWrite(C,  LOW);
          digitalWrite(D,  LOW);
          digitalWrite(E,  LOW);
          digitalWrite(F,  LOW);
          digitalWrite(G,  LOW);

          lcd.clear();
          delay(10);
          lcd.print("<     GERSE    >"); // ESCREVER EM DISPLAY LCD 16X2
          delay(10);

          cls = true;
      }


     if(bt3)                              // SE BOTÃO 3 PRESSIONADO
     {
        NUM = (CALC/1023)*255;            // 0 -> 0   1023 -> 255
        analogWrite(MOTOR1, NUM);         // SETAR VALOR DO PWM NO MOTOR1
        analogWrite(MOTOR2, NUM);         // SETAR VALOR DO PWM NO MOTOR2

        lcd.setCursor(0, 1);      // POSICIONAR CURSOR NA COLUNA 0, LINHA 1
        lcd.print("   ");
        lcd.setCursor(0, 1);
        lcd.print(NUM);           // ESCREVER VALOR DE NUM NO DISPLAY LCD 16X2
      }
    }


    if(bt2)                               // SE BOTAO2 PRESSIONADO
    {
      NUM = (CALC/1023)*255;
      analogWrite(BUZZER, NUM);           // SETAR VALOR DO PWM NO BUZZER
    }

    delay(100);                           // ESPERAR 100ms
}
```