گزارش پروژه درس کنترل خطی
Wire cutting machine

اعضای گروه:

علی سعیدی

محمد حسین وهابی

امین کارگر

اجزای پروژه:

- **استپ موتور :**

موتور پله ای نوعی موتور dc است که به صورت گسسته حرکت می‌کند. درون موتور پله ای چندین سیم پیچ وجود دارد که در گروه هایی به نام فاز قرار داده شده اند. با عبور الکتریسیته از هر یک از این فازها، موتور با سرعت یک گام در واحد زمانی به چرخش در می‌آید.

با کنترل کردن گام های استپ موتور توسط یک مولد پالس، مانند یک پی ال سی یا کنترلر CNC یا میکروکنترلر کنترلر استپ موتور، و ارسال آن به درایور استپ موتور، میتوان کنترل سرعت و موقعیت بسیار دقیقی به دست آورد. به همین دلیل استپ موتورها گزینه هایی مناسب برای استفاده در پروژه هایی هستند که نیاز به کنترل حرکت دقیق دارند .استپ موتور ها در اندازه ها و انواع مختلف با مشخصات الکتریکی متفاوت برای کارهای صنعتی مختلفی وجود دارند که بر حسب نیاز می‌توان یکی از آن ها را انتخاب کرد.

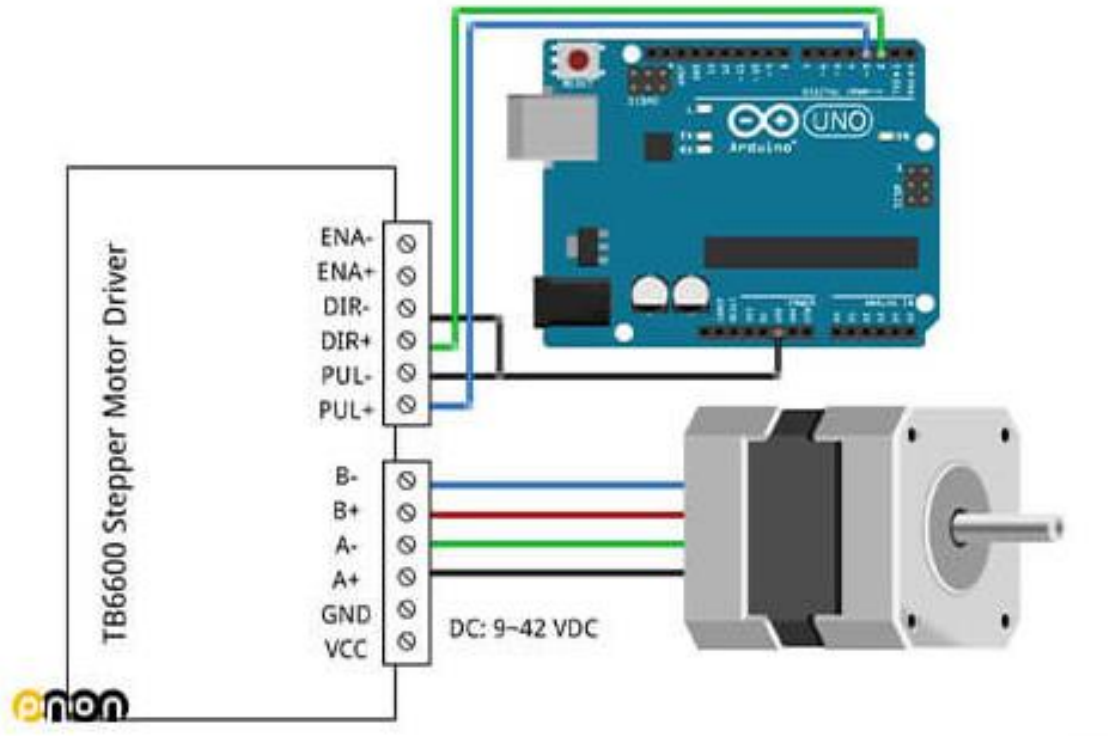
اجزا تشکیل دهنده استپ موتور

استاتور: در قسمت استاتور تعدادی سیم پیچ با هم گروه بندی شده اند. مثلا در استپ موتور دو فاز دو گروه سیم پیچ می‌بینید که از داخل با هم اتصال دارند که نهایتا چهار سیم از آن ها بیرون آمده که دو به دو به هر کدام به گروه سیم پیچ ها متصل اند. در استپ موتور سه فاز ، شش سیم خروجی و در استپ موتور ۵ فاز ۱۰ سیم خروجی خواهیم داشت. البته این نکته باید گفته شود که در موتور های پله ای گاها این ده سیم یا شش سیم از داخل با یکدیگر سری یا موازی می‌شوند و بیرون می‌آیند.

روتور: روتور معمولا بسته به اینکه گشتاور موتور پله ای و طول موتور پله ای چقدر است تعداد طبقات کمتر یا بیشتری دارد. مثلا در یک موتور پله ای که ۴ طبقه است اگر می‌خواهد طول بیشتری داشته باشد می‌تواند تا ۸ طبقه شود که در نتیجه آن گشتاور استپ موتور بیشتر می‌شود یا می‌تواند طول آن به ۲ طبقه کاهش پیدا کند که در آن صورت نیز گشتاور استپ موتور نصف خواهد شد.

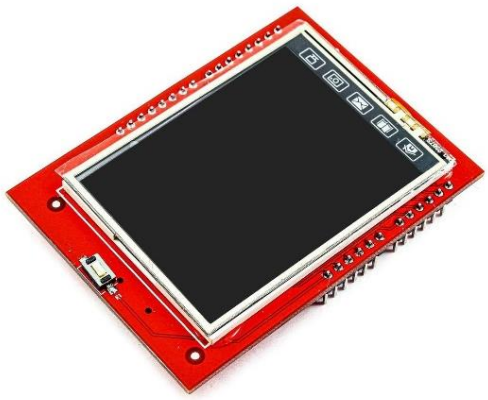
مزایای استپ موتور : با هزینه خیلی کم می‌توان پروژه های کنترل انجام داد. ابعاد کوچکی دارند و در ابعاد کوچک گشتاور زیادی می‌توانند اعمال کنند. سایزبندی های مختلف دارند و بسته به پروژه های مختلف می‌توان از آن ها استفاده کرد. درایورهای آن ها درایور های ساده ای است و کار با آن ها ساده است.

تصویر ۱



تصویر ۲

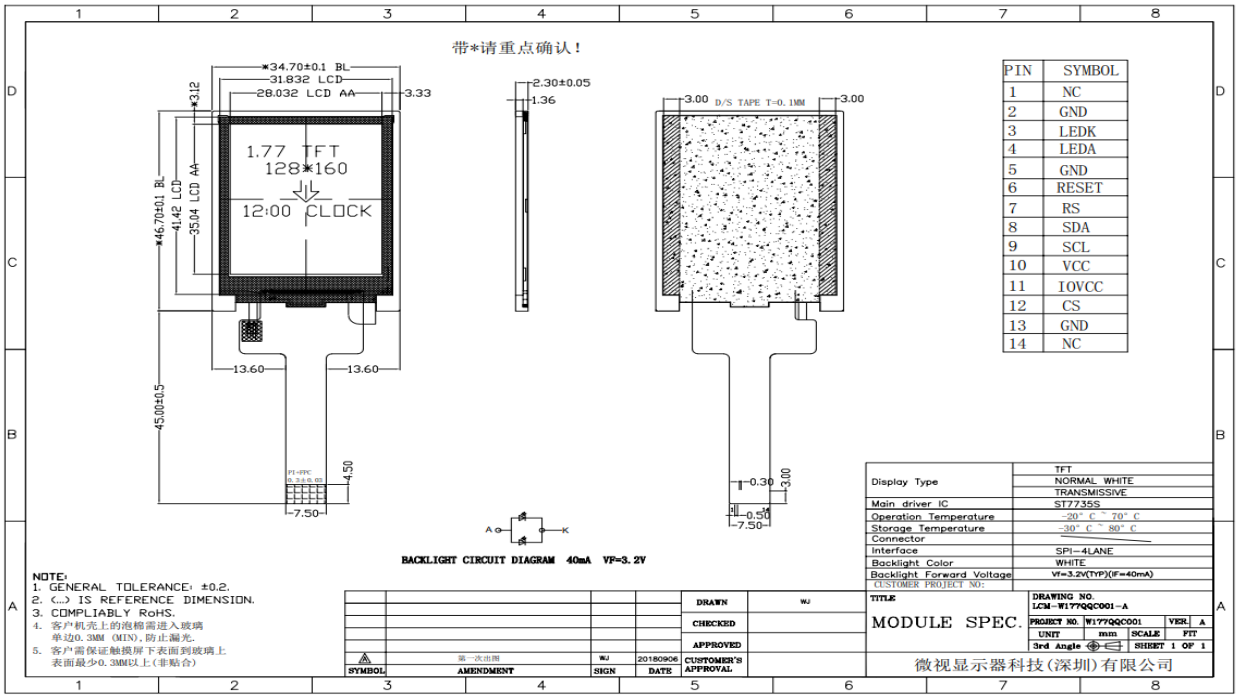
- **صفحه نمایش TFT :**



تصویر ۳

نمایشگر TFT از انواع ال سی دی های کریستال مایع می‌باشد که با یک ترانزیستور به هر پیکسل وصل شده است و همچنین علاوه بر مصرف جریان نسبتاً کم، دارای بک لایت (Backlight) نیز می‌باشد.

داده‌برگ[1] نمایشگر :



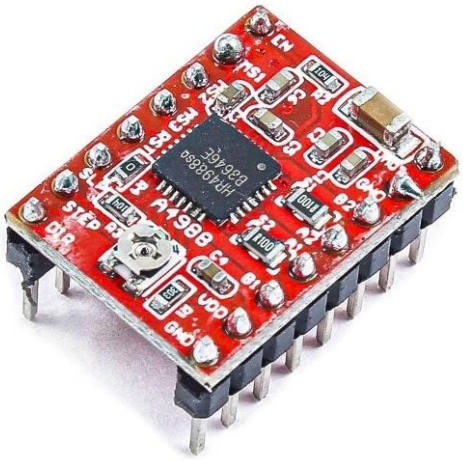
PIN	SYMBOL
1	NC
2	GND
3	LEDK
4	LEDA
5	GND
6	RESET
7	RS
8	SDA
9	SCL
10	VCC
11	IOVCC
12	CS
13	GND
14	NC

Display Type	TFT NORMAL WHITE TRANSMISSIVE
Main driver IC	ST7735S
Operation Temperature	-20° C ~ 70° C
Storage Temperature	-30° C ~ 80° C
Connector	
Interface	SPI-4LANE
Backlight Color	WHITE
Backlight Forward Voltage	Vf=3.2V(TYP)(IF=40mA)
CUSTOMER PROJECT NO:	

تصویر ٤

[1] Datasheet

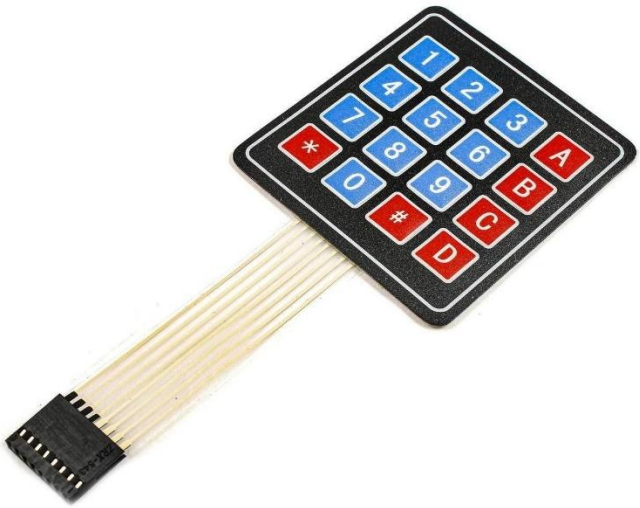
- **درایور استپ موتور :**



تصویر ۵

این ماژول یک برد حامل برای برد درایور میکرواستپ Allegro's A4988 DMOS است که توسط POLOLO در مقابل جریان بیش از حد محافظت میشود.

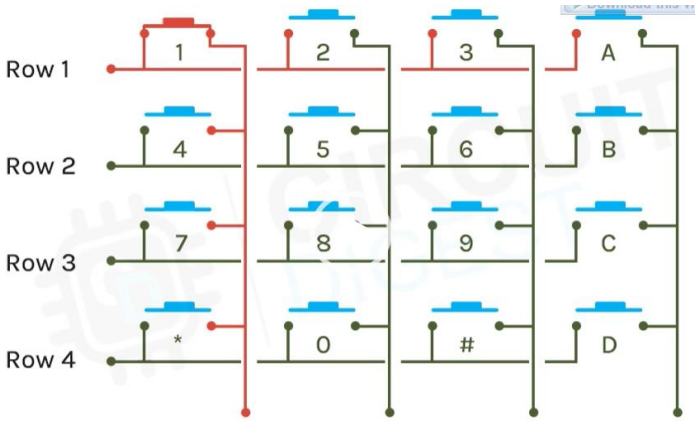
- **صفحه کلید ماتریسی:**



تصویر ۶

هنگامی که دکمه ای روی صفحه کلید غشایی فشار داده می شود، یک ردیف را به یک ستون متصل می کند و اجازه می دهد جریان الکتریکی کمی بین آنها جریان یابد. این یک مدار الکتریکی ایجاد می کند که می تواند توسط میکروکنترلر

شناسایی شود، که سپس میکروکنترلر می تواند تعیین کند که کدام دکمه بر اساس ترکیب ردیف و ستون متصل شده است. فشار دکمه تغییری در ولتاژ یا جریان ایجاد می کند که میکروکنترلر می تواند آن را تشخیص داده و به عنوان سیگنال ورودی تفسیر کند. صفحه کلید ممکن است از یک روش اسکن استفاده کند که در آن میکروکنترلر یک سری پالس به هر ردیف می فرستد و ولتاژ هر ستون را اندازه گیری می کند تا مشخص کند کدام دکمه فشرده شده است.



تصویر ۷

- **برد آردوینو UNO:**

تصویر ۸

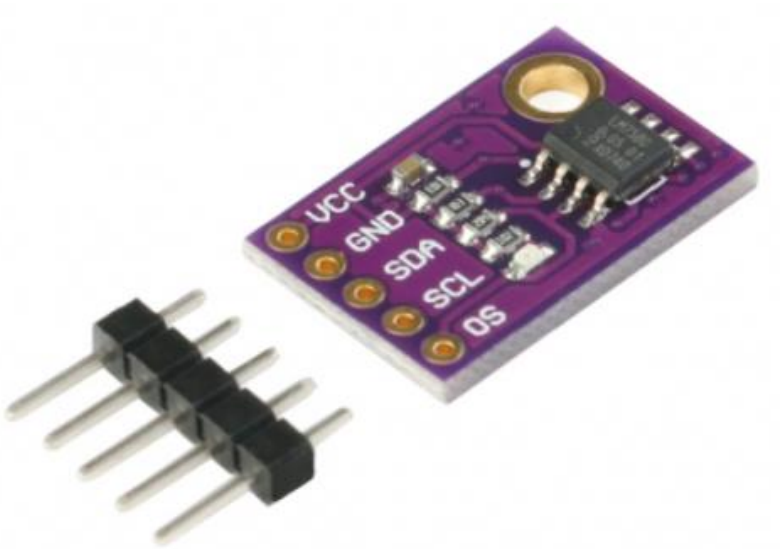
این برد برای پیاده سازی منطق کد نویسی شده به ماژول ها استفاده شده است.

- **مبنع تغذیه سوئیچینگ:**

تصویر ۹

این منبع تغذیه برای تامین ولتاژ های مختلف مورد نیاز برای هر ماژول استفاده شده است (۱۲ و ۵ و ۳,۳ ولت)

- **سنسور دمای LM75** :



تصویر ۱۰

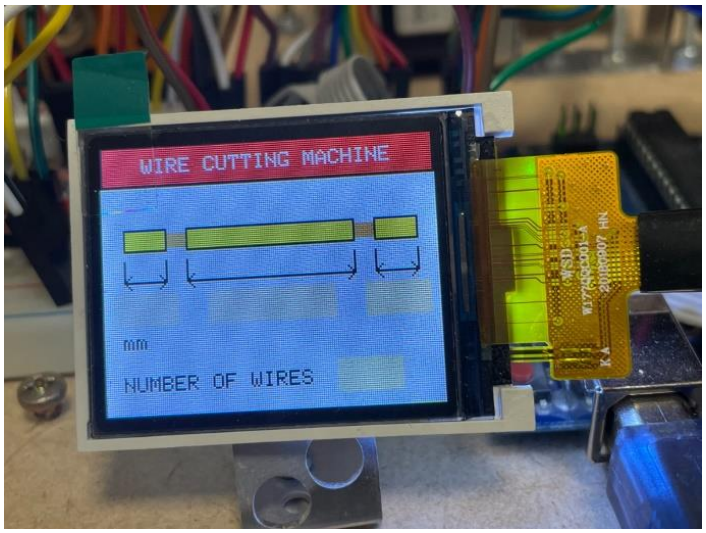
این سنسور دما برای خواندن دمای موتور ها و دادن فیدبک به آردوینو استفاده می‌شود.

توضیح عملکرد پروژه :

به طور خلاصه در این پروژه هدف ما این است که تعدادی سیم به اندازه‌ی دلخواه ما بریده شود که سر و ته آن سیم ها هم به اندازه دلخواه روکش سیم از آن جدا شده باشد.

در لحظه راه‌اندازی سیستم نمایشگر به این صورت خواهد بود:

تصویر ۱۱



تصویر ۱۲

در صفحه باز شده باید ۴ عدد توسط کاربر وارد بشود که روی صفحه نمایش مسخص شده اند:

- میزان برداشته شدن روکش اول سیم

- میزان طول وسط سیم
- میزان برداشته شدن روکش انتهای سیم
- تعداد سیم هایی که به این اندازه نیاز داریم

و بعد از وارد کردن عدد با زدن # روی صفحه کلید ماتریسی دستگاه شروع به کار خواهد کرد.

برنامه نوشته شده در آردوینو :

در ابتدا کتابخانه های مورد نیاز شامل توابع LCD و صفحه‌کلید ماتریسی و توابع ریاضی تعریف می‌شوند .

```
#include <LCDWIKI_GUI.h>
#include <LCDWIKI_SPI.h>
#include <Keypad.h>
#include <math.h>
```

تعریف پین های قرارگیری موتور , تعریف صفحه‌کلید ماتریسی

توابع راه‌اندازی و ساخت محیط گرافیکی نمایشگر

```
const int stepdir = 3;
const int stepPin = 2;
const int cutdir = 10;
const int cutPin = 12;

const byte ROWS = 4;
const byte COLS = 3;


char hexaKeys[ROWS][COLS] = {
  { '1', '2', '3' },
  { '4', '5', '6' },
  { '7', '8', '9' },
  { '<', '0', '>' }
};

byte rowPins[ROWS] = { 4, 5, 6, 7 };
byte colPins[COLS] = { 8, 9, A2 };
Keypad keypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

//paramters define
#define MODEL ST7735S
```

```
#define CS A5
#define CD A3
#define RST A4
#define LED A0

long int num1 = 0;
long int num2 = 0;
long int num3 = 0;
long int num4 = 0;

LCDWIKI_SPI my_lcd(MODEL, CS, CD, RST, LED);
unsigned long show_start(void) {
  my_lcd.Set_Draw_color(255, 255, 255);
  my_lcd.Fill_Rectangle(0, 0, my_lcd.Get_Display_Width() - 1, my_lcd.Get_Display_Height() - 1);
  my_lcd.Set_Draw_color(255, 0, 0);
  my_lcd.Fill_Round_Rectangle(my_lcd.Get_Display_Width() / 2 - 1 - 72 + 1, my_lcd.Get_Display_Height() / 2 - 1
- 52 + 1, my_lcd.Get_Display_Width() / 2 - 1 + 72 - 1, my_lcd.Get_Display_Height() / 2 - 1 + 52 - 1, 5);
  my_lcd.Set_Text_colour(255, 255, 255);
  my_lcd.Set_Text_Size(2.5);
  my_lcd.Set_Text_Mode(1);
  my_lcd.Print_String("WIRE", CENTER, 35);
  my_lcd.Print_String("CUTTING", CENTER, 53);
  my_lcd.Print_String("MACHINE", CENTER, 71);
  delay(5000);
}
unsigned long menu(void) {
  my_lcd.Set_Draw_color(255, 0, 0);
  my_lcd.Fill_Rectangle(0, 0, my_lcd.Get_Display_Width() - 1, 20);
  my_lcd.Set_Draw_color(0, 0, 0);
  my_lcd.Draw_Rectangle(0, 0, my_lcd.Get_Display_Width() - 1, 19);
  my_lcd.Set_Text_colour(255, 255, 255);  //text on top
  my_lcd.Set_Text_Size(1.9);
  my_lcd.Set_Text_Mode(1);
  my_lcd.Print_String("WIRE CUTTING MACHINE", CENTER, 6);
  my_lcd.Set_Draw_color(255, 255, 255);
  my_lcd.Fill_Rectangle(0, 20, my_lcd.Get_Display_Width() - 1, my_lcd.Get_Display_Height() - 1);

  my_lcd.Set_Draw_color(255, 240, 0);
  my_lcd.Fill_Rectangle(10, 40, 30, 50);
  my_lcd.Fill_Rectangle(40, 40, 120, 50);
  my_lcd.Fill_Rectangle(130, 40, 150, 50);
  my_lcd.Set_Draw_color(0, 0, 0);  //yellow part
  my_lcd.Draw_Rectangle(10, 40, 30, 50);
  my_lcd.Draw_Rectangle(40, 40, 120, 50);
  my_lcd.Draw_Rectangle(130, 40, 150, 50);
```

```
 my_lcd.Set_Draw_color(224, 141, 40);
 my_lcd.Fill_Rectangle(31, 42, 39, 48);  //copper part
 my_lcd.Fill_Rectangle(121, 42, 129, 48);

 my_lcd.Set_Draw_color(0, 0, 0);
 my_lcd.Draw_Line(10, 55, 10, 65);
 my_lcd.Draw_Line(30, 55, 30, 65);
 my_lcd.Draw_Line(40, 55, 40, 65);
 my_lcd.Draw_Line(120, 55, 120, 65);
 my_lcd.Draw_Line(130, 55, 130, 65);
 my_lcd.Draw_Line(150, 55, 150, 65);
 my_lcd.Draw_Line(10, 65, 30, 65);
 my_lcd.Draw_Line(40, 65, 120, 65);
 my_lcd.Draw_Line(130, 65, 150, 65);
 my_lcd.Set_Text_colour(0, 0, 0);  //black lines
 my_lcd.Set_Text_Size(1.8);
 my_lcd.Set_Text_Mode(1);
 my_lcd.Print_String("<", 9, 62);
 my_lcd.Print_String("<", 39, 62);
 my_lcd.Print_String("<", 129, 62);
 my_lcd.Print_String(">", 26, 62);
 my_lcd.Print_String(">", 116, 62);
 my_lcd.Print_String(">", 146, 62);

 my_lcd.Set_Draw_color(255, 255, 150);
 my_lcd.Fill_Rectangle(5, 70, 35, 85);
 my_lcd.Fill_Rectangle(50, 70, 110, 85);
 my_lcd.Fill_Rectangle(125, 70, 155, 85);
 my_lcd.Set_Draw_color(0, 0, 0);  //text box
 my_lcd.Draw_Rectangle(10, 40, 30, 50);
 my_lcd.Draw_Rectangle(40, 40, 120, 50);
 my_lcd.Draw_Rectangle(130, 40, 150, 50);
 my_lcd.Set_Text_colour(0, 0, 0);
 my_lcd.Set_Text_Size(1.9);
 my_lcd.Set_Text_Mode(1);
 my_lcd.Print_String("mm", 9, 90);

 my_lcd.Print_String("NUMBER OF WIRES", 9, 110);
 my_lcd.Set_Draw_color(255, 255, 150);
 my_lcd.Fill_Rectangle(110, 105, 140, 120);


 // delay(5000);
}
```

تعریف تابع برای گرفتن مقادیر از کاربر :

```
unsigned long keypadd(void) {
  while (1) {
    //   char Key = customKeypad.getKey();
    char x = 'k';
    int t = 5;
    my_lcd.Set_Text_colour(0, 0, 0);
    my_lcd.Set_Text_Size(1.8);
    my_lcd.Set_Text_Mode(1);
    //    my_lcd.Draw_Char(5,30,Key,1,1,1,0);
  }
}
```

نوشتن اعداد گرفته شده از کاربر :

```
int number(int x, int y) {
  int num = 0;

  //  char Key = customKeypad.getKey();
  //  num = (num*10)+(Key-'0');
  my_lcd.Set_Text_colour(210, 210, 210);
  my_lcd.Set_Text_Size(1.8);
  my_lcd.Set_Text_Mode(1);
  my_lcd.Print_Number_Int(num, x, y, 1, ' ', 10);
}
```

تعیین حالت پین ها نرخ تبادل داده[1] و فراخوانی نمایشگر

```
void setup() {
  pinMode(stepPin, OUTPUT);
  pinMode(stepdir, OUTPUT);
  pinMode(cutPin, OUTPUT);
  pinMode(cutdir, OUTPUT);
  Serial.begin(9600);
  my_lcd.Init_LCD();
  my_lcd.Fill_Screen(0x0);
  my_lcd.Set_Rotation(1);
}
```

شروع حلقه برنامه

[1] Baud rate

۴ حلقه تعریف شده که در انتظار ورودی کاربر هستند بدین صورت که ۴ ورودی عدد به عنوان اندازه سیم و تعداد آن دریافت می‌کند. در هر مرحله اگر کاربر کلید ستاره(*) را فشار دهد تمام مقادیر بازنشانی خواهند شد(بازگشت به خط با پرچم again1) یا کلید مربع(#) را به عنوان تایید مقادیر دریافت می‌نماید .

```
void loop() {
 // digitalWrite(cutdir, LOW);
  //digitalWrite(cutPin, LOW);
show_start();
again1:
 menu();

 while (1) {

  char key = keypad.getKey();

  if (key && key != '>' && key != '<') {
   num1 = (num1 * 10) + (key - '0');  //num1
   my_lcd.Set_Text_colour(0, 0, 0);
   my_lcd.Set_Text_Size(1.8);
   my_lcd.Set_Text_Mode(1);
   my_lcd.Print_Number_Int(num1, 10, 74, 1, ' ', 10);
  }
  if (key == '>') break;
  if (key == '<') {
   num1 = 0;
   num2 = 0;
   num3 = 0;
   num4 = 0;
   goto again1;
  }
 }

 while (1) {

  char key = keypad.getKey();

  if (key && key != '>' && key != '<') {
   num2 = (num2 * 10) + (key - '0');  //num2
   my_lcd.Set_Text_colour(0, 0, 0);
   my_lcd.Set_Text_Size(1.8);
   my_lcd.Set_Text_Mode(1);
   my_lcd.Print_Number_Int(num2, 55, 74, 1, ' ', 10);
  }
```

```
  if (key == '>') break;
  if (key == '<') {
    num1 = 0;
    num2 = 0;
    num3 = 0;
    num4 = 0;
    goto again1;
  }
}

while (1) {

  char key = keypad.getKey();

  if (key && key != '>' && key != '<') {
    num3 = (num3 * 10) + (key - '0');
    my_lcd.Set_Text_colour(0, 0, 0);  //num3
    my_lcd.Set_Text_Size(1.8);
    my_lcd.Set_Text_Mode(1);
    my_lcd.Print_Number_Int(num3, 130, 74, 1, ' ', 10);
  }
  if (key == '>') break;
  if (key == '<') {
    num1 = 0;
    num2 = 0;
    num3 = 0;
    num4 = 0;
    goto again1;
  }
}

while (1) {
  char key = keypad.getKey();

  if (key && key != '>' && key != '<') {
    num4 = (num4 * 10) + (key - '0');
    my_lcd.Set_Text_colour(0, 0, 0);  //num4
    my_lcd.Set_Text_Size(1.8);
    my_lcd.Set_Text_Mode(1);
    my_lcd.Print_Number_Int(num4, 115, 109, 1, ' ', 10);
  }
  if (key == '>') break;
  if (key == '<') {
    num1 = 0;
    num2 = 0;
```

```
    num3 = 0;
    num4 = 0;
    goto again1;
   }
 }
```

سپس حلقه ای تعریف شده تا به تعداد سیم خواسته شده تکرار شود و سیم ببرد.

در داخل حلقه حلقه های دیگری نیز قرار دارند تا موتور سیم را به طول ورودی وارد کند. مقادیر اندازه با استفاده از توابع مربوط به stepmotor و فرمول آن به تعداد گام های چرخش موتور تبدیل می‌شود.

پس وارد شدن سیم در هر مرحله موتور برش‌دهنده سیم را برش می‌دهد و بدین ترتیب سیم با اندازه های خواسته شده تحویل داده می‌شود.

```
digitalWrite(stepdir, LOW);

 for (int x = 0; x < num4; x++) {

   if (x == 0) {
    delay(1000);
    digitalWrite(cutdir, LOW);
    for (int y = 0; y < 25; y++) {
     digitalWrite(cutPin, HIGH);
     delayMicroseconds(3000);
     digitalWrite(cutPin, LOW);
     delayMicroseconds(3000);
    }
    delay(500);
    digitalWrite(cutdir, HIGH);
    for (int y = 0; y < 25; y++) {
     digitalWrite(cutPin, HIGH);
     delayMicroseconds(3000);
     digitalWrite(cutPin, LOW);
     delayMicroseconds(3000);
    }
    delay(1000);
   }
   for (int x = 0; x < floor((num1 * 621) / 100); x++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(3000);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(3000);
   }
```

```
delay(200);
digitalWrite(cutdir, LOW);
for (int y = 0; y < 25; y++) {
  digitalWrite(cutPin, HIGH);
  delayMicroseconds(3000);
  digitalWrite(cutPin, LOW);
  delayMicroseconds(3000);
}
delay(200);
digitalWrite(cutdir, HIGH);
for (int y = 0; y < 25; y++) {
  digitalWrite(cutPin, HIGH);
  delayMicroseconds(3000);
  digitalWrite(cutPin, LOW);
  delayMicroseconds(3000);
}
delay(200);


for (int x = 0; x < floor((num2 * 621) / 100); x++) {
  digitalWrite(stepPin, HIGH);
  delayMicroseconds(3000);
  digitalWrite(stepPin, LOW);
  delayMicroseconds(3000);
}

delay(200);
digitalWrite(cutdir, LOW);
for (int x = 0; x < 25; x++) {
  digitalWrite(cutPin, HIGH);
  delayMicroseconds(5000);
  digitalWrite(cutPin, LOW);
  delayMicroseconds(5000);
}
delay(200);
digitalWrite(cutdir, HIGH);
for (int x = 0; x < 25; x++) {
  digitalWrite(cutPin, HIGH);
  delayMicroseconds(5000);
  digitalWrite(cutPin, LOW);
  delayMicroseconds(5000);
}
delay(200);

for (int x = 0; x < floor((num3 * 621) / 100); x++) {
```

```
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(3000);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(3000);
   }
   delay(200);
   digitalWrite(cutdir, LOW);
   for (int x = 0; x < 25; x++) {
    digitalWrite(cutPin, HIGH);
    delayMicroseconds(5000);
    digitalWrite(cutPin, LOW);
    delayMicroseconds(5000);
   }
   delay(200);
   digitalWrite(cutdir, HIGH);
   for (int x = 0; x < 25; x++) {
    digitalWrite(cutPin, HIGH);
    delayMicroseconds(5000);
    digitalWrite(cutPin, LOW);
    delayMicroseconds(5000);
   }
   delay(200);
  }
}
```