به نام خدا

دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)
دانشکده مهندسی کامپیوتر

آزمایشگاه ریزپردازنده

گزارش‌کار آزمایش ۳

استاد درس: مهندس معصوم زاده

مهدی رحمانی/ ۹۷۳۱۷۰۱
محمّد امین رضائی/ ۹۷۳۱۰۲۴

«پیش گزارش»

مشخصات فنی ماژول نمایشگر ال‌سی‌دی کاراکتری 2×16 و دلیل استفاده از پتانسیومتر در مدار

در ابتدا یک معرفی مختصر از این نمایشگرها میکنیم :

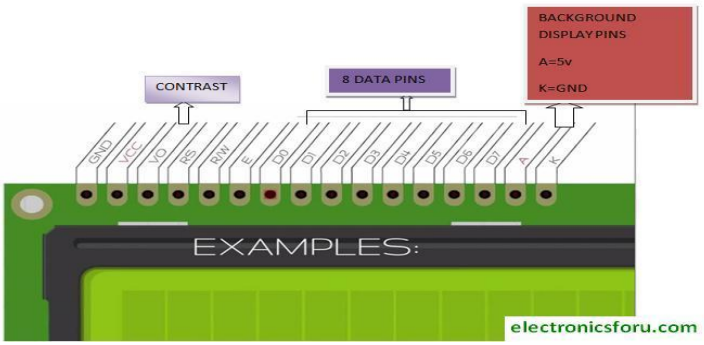
همان گونه که از نام این ماژول آماده و مونتاژ شده، مشخص است این ابزار یک نمایشگر است با دو ردیف 16 تایی کاراکتری جهت نمایش داده ها و به همین دلیل آن را ماژول LCD 2*16 می نامند که در دو مدل سبز و آبی رنگ عرضه می شود. مدل سبز رنگ دارای پس زمینه سبز و متن و کاراکترهای مشکی و مدل آبی رنگ دارای پس زمینه آبی و نوشتار سفید رنگ است. یکی از دلایل پر مصرف بودن این مدل ال سی دی در مقابل نمایشگرهایی همچون سون سگمنت، عدم محدودیت در نمایش کاراکترها می‌باشد. همچنین توانایی ساختن کارکترهای دلخواه را دارند.

از ویژگی های آن:

- تعداد قطعه های به کار رفته: 14 عدد شامل قطعات گوناگون روی برد
- اندازه فیبر نمایشگر: طول 80 میلی متر * عرض 36 میلی متر * عمق 10 میلی متر
- رنگ های موجود در بازار: سبز و آبی به عنوان بکگراند
- شمار پایه ها: 16 عدد
- گونه‌ی صفحه نمایش: نمایشگر LCD کاراکتری تک رنگ
- پشتیبانی رزولیشن: شامل 32 کاراکتر متشکل از دو ردیف 16 تایی
- ولتاژ تغذیه ورودی: 3.5 تا 5 ولت مستقیم DC
- شدت جریان: 16 میلی آمپر ویژه‌ی نور پس زمینه
- میزان دمای حفاظت : 10- تا 60+ درجه سانتی گراد
- میزان دمای کارکرد استاندارد: 0 درجه تا 50+ درجه سانتی گراد

حال در ادامه هرکدام از پین ها را در جدول معرفی میکنیم.

شماره پین	نام	کارکرد
1	GND	زمین یا همان 0 ولت
2	Vcc	ولتاژ تغذیه 5 ولت(به طور دقیق‌تر 4.7ولت تا 5.3 ولت)
3	Vo/VEE	برای تنظیم contrast میباشد(بهترین راه هم استفاده از یک مقاومت متغیر مانند پتانسیومتر میباشد که خروجی پتانسیومتر به آن پین وصل میشود و با چرخش آن میتوان تباین رنگ را تنظیم کرد.)
4	RS (Register Select)	هنگامی که Low باشد command register و زمانی که High باشدdata register را انتخاب میکند.
5	Read/write (RW)	زمانی که Low باشد برای write روی رجیستر و زمانی که High باشد از روی آن Read میکند.
6	Enable	وقتی به این پین یک پالس high to low داده میشود، دیتاها را روی پین‌های مخصوص دیتا میفرستد.
7	DB0	8-bit data pins
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	Led+ (A)	Backlight VCC (5V)
16	Led- (K)	Backlight Ground (0V)



همانطور که در جدول هم گفته شد دلیل استفاده از پتانسیومتر در مدار برای تنظیم contrast صفحه نمایش میباشد. 3 پایه دارد که پایه های چپ و راست باید به VCC و GND وصل شوند. پایه وسط که output میباشد باید به پین شماره‌ی3 از LCD یعنی Vo وصل شود. با چرخش پتانسیومتر میتوان مقدار contrast صفحه LCD که میزان تباین و تضاد رنگ های بک لایت و نوشته ها را مشخص میکند را تنظیم کرد.

تعریف مختصر توابع مورد نیاز از کتابخانه LiquidCrystal مانند:

- LiquidCrystal()

یک متغیر از جنس LiquidCrystal میشه به کمک آن ساخت. نمایشگر با استفاده از 4 یا 8 خط داده قابل کنترل است. اگر مورد اول است ، اعداد پین را برای d0 تا d3 حذف کنید و آن خطوط را بدون ارتباط بگذارید. پین RW می تواند به زمین متصل شود به جای اینکه به پین آردوینو متصل شود. اگر چنین است ، آن را از پارامترهای این تابع حذف کنید.

سینتکس هایی که برای کار با این تابع میتوان داشت به شکل زیر میباشند:

- ✓ LiquidCrystal(rs, enable, d4, d5, d6, d7)
- ✓ LiquidCrystal(rs, rw, enable, d4, d5, d6, d7)
- ✓ LiquidCrystal(rs, enable, d0, d1, d2, d3, d4, d5, d6, d7)
- ✓ LiquidCrystal(rs, rw, enable, d0, d1, d2, d3, d4, d5, d6, d7)

پارامترهای مورد استفاده را هم در زیر توضیح میدهیم:

rs: شماره‌ی پینی از بورد آردوینو که به پین RS از LCD وصل میشود.

rw: شماره‌ی پینی از بورد آردوینو که به پین RW از LCD وصل میشود.(اختیاری)

enable: شماره‌ی پینی از بورد آردوینو که به پین enable از LCD وصل میشود.

d0, d1, d2, d3, d4, d5, d6, d7: شماره‌هایی از پین‌های بورد آردوینو که به پین‌های متناظر دیتا در LCD وصل میشوند.

- begin()

رابط کاربری را روی صفحه LCD آغاز می کند و ابعاد (عرض و ارتفاع) نمایشگر را مشخص می کند. لازم است begin() قبل از هر دستور دیگری در کتابخانه LCD فراخوانی شود.

سینتکس کار با آن به صورت زیر است:

✓ *lcd*.begin(cols, rows)

حال پارامترهای به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

cols: تعداد ستون‌هایی که نمایشگر دارد.

rows: تعداد سطرهایی که نمایشگر دارد.

- clear()

صفحه LCD را خالی کرده و cursor را در گوشه بالا سمت چپ قرار میدهد.
سینتکس کار با آن به صورت زیر است:

✓ *lcd*.clear()

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

- setCursor()

جای کرسر LCD را تعیین میکند. یعنی محلی را تنظیم کنید که متن بعدی که میخواهیم در LCD نوشته شود از آنجا نمایش داده شود.

سینتکس کار با آن به صورت زیر است:

✓ *lcd*.setCursor(col, row)

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

cols: تعداد ستون‌هایی که نمایشگر دارد.

rows: تعداد سطرهایی که نمایشگر دارد.

- write()

یک کاراکتر روی LCD مینویسد.

سینتکس کار با آن به صورت زیر است:

✓ *lcd*.write(data)

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

data: کاراکتری که میخواهیم روی صفحه نمایش بنویسیم تا نمایش دهد.

همچنین این تابع مقدار تعداد بایت های نوشته شده را برمیگرداند و استفاده از آن اختیاری میباشد.

- print()

به کمک این میتوان یک متن (text) را داخل LCD نوشت.
سینتکس‌های کار با آن به صورت زیر است:

- ✓ *lcd*.print(data)
- ✓ *lcd*.print(data, BASE)

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

data: دیتا و متنی که میخواهیم روی LCD نمایش دهد.

BASE: پایه ای که می توان اعداد را در آن چاپ کرد: BIN برای باینری (پایه 2) ، DEC برای اعشاری (پایه 10) ، OCT برای هشت (پایه 8) ، HEX برای هگزادسیمال (پایه 16).

همچنین این تابع مقدار تعداد بایت های نوشته شده را برمیگرداند و استفاده از آن اختیاری میباشد.

- noDisplay()

صفحه LCD را خاموش می کند ، بدون اینکه متنی که در حال حاضر روی آن نشان داده شده است را از دست بدهد.
سینتکس کار با آن به صورت زیر است:

- ✓ *lcd*.noDisplay()

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

- scrollDisplayLeft()

محتویات نمایشگر (متن و cursor) را به اندازه‌ی یک space به سمت چپ میفرستد.
سینتکس کار با آن به صورت زیر است:

✓ *lcd*.scrollDisplayLeft()

حال پارامتر به کار رفته را توضیح میدهیم:

lcd: یک متغیر از جنس LiquidCrystal میباشد.

- autoscroll()

پیمایش خودکار LCD را روشن می کند. این امر باعث می شود که هر کاراکتر خروجی برای نمایش روی صفحه ، کاراکترهای قبلی را یک space شیفت دهد. اگر جهت متن فعلی از چپ به راست باشد (پیش فرض) ، صفحه به سمت چپ scroll می کند. اگر جهت فعلی از راست به چپ باشد ، صفحه به سمت راست scroll می کند. این امر منجر می‌شود هر کاراکتر جدید در یک مکان یکسان در LCD چاپ شود

سینتکس کار با آن به صورت زیر است:

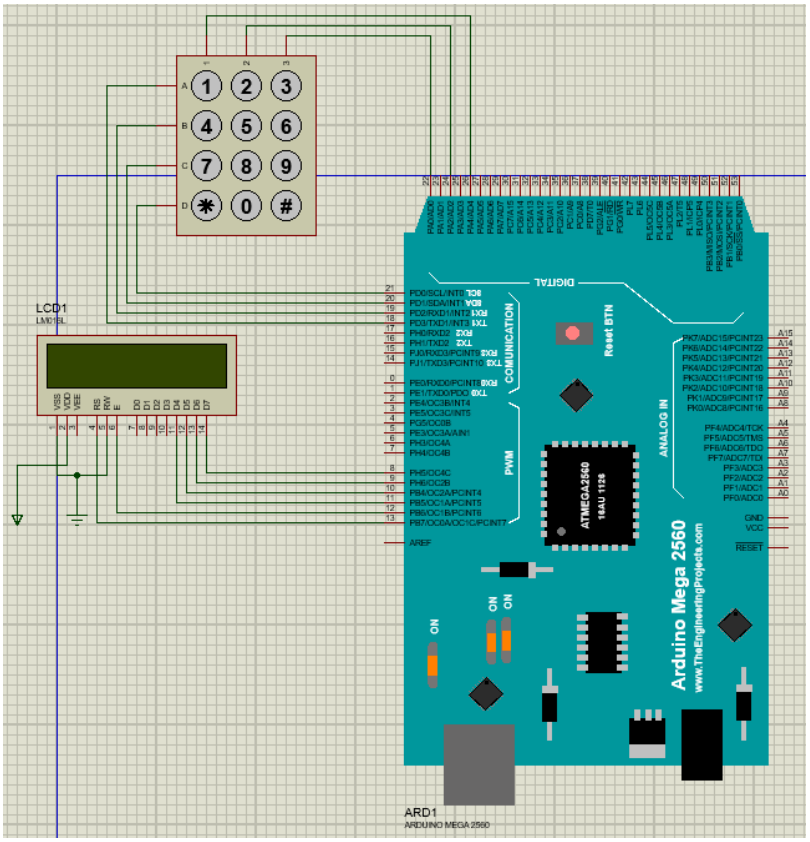
✓ *lcd*.autoscroll()

حال پارامتر به کار رفته را توضیح میدهیم:

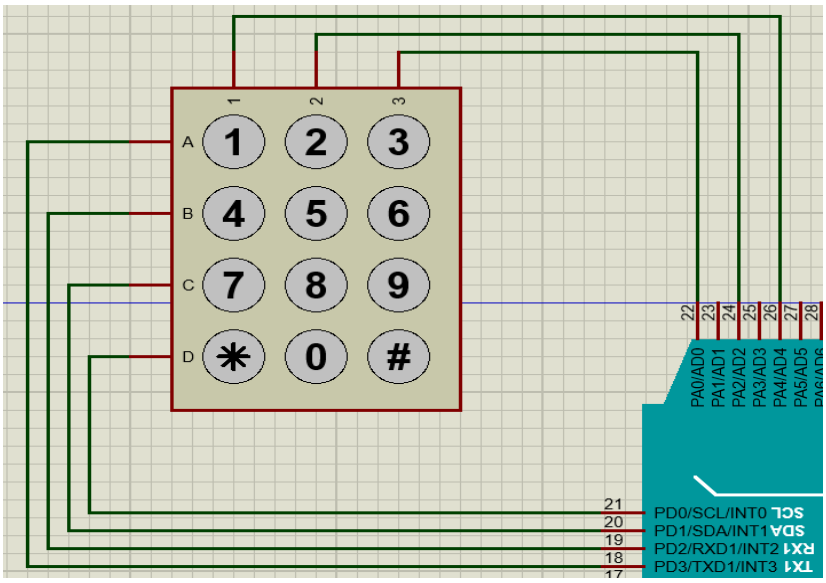
lcd: یک متغیر از جنس LiquidCrystal میباشد.

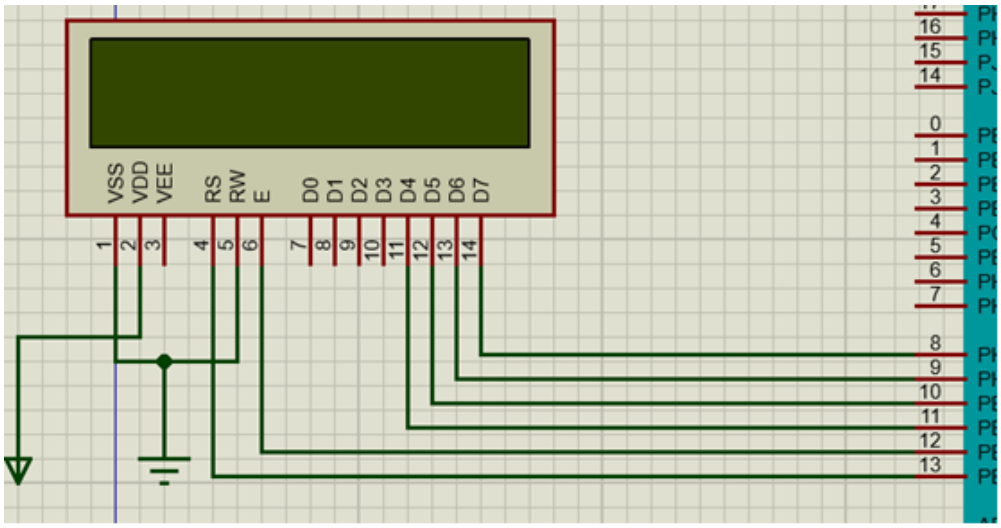
«گزارش آزمایش»

ابتدا برای بخش های یک، دو و چهار آزمایش، مدار زیر را به کمک برد آردوبینو مگا ۲۵۶۰، یک ال سی دی ۱۶ در ۲ و یک صفحه کلید ماتریسی میبندیم:



حال به پورت های ورودی و خروجی نگاه دقیق‌تری می‌اندازیم:





حال به بررسی سوال اول میپردازیم:

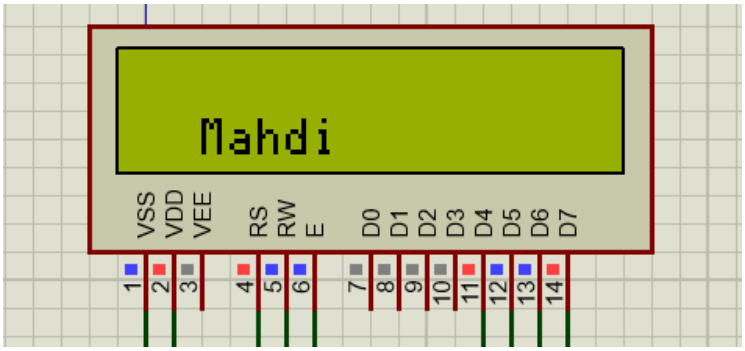
سوال اول) همانطور که در عکس زیر می‌بینید، ابتدا یک صفحه نمایش را از بین componentهای پروتئوس اضافه کرده و به برد Arduino Mega 2560 وصل کنید. با استفاده از توابع کتابخانه‌ای برنامه‌ای بنویسید که اسم خودتان در ابتدای سطر اول LCD کاراکتری نوشته شده و هر ثانیه یک خانه به جلو حرکت کند و وقتی به انتهای خط اول رسید به خط دوم رفته و باز به سمت جلو حرکت نماید. سپس به طور پیوسته این کار را تکرار نماید.

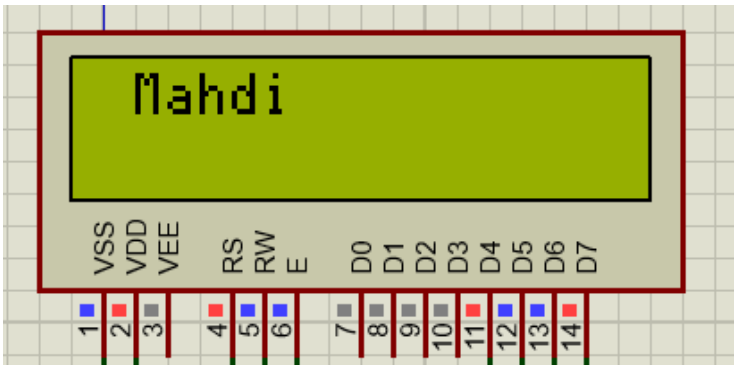
کد این سوال به این صورت میباشد:

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
int i = 0;
String my_name = "Mahdi";
int len = my_name.length();

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
}

void loop() {
  int line_no = i/(17-len)%2;
  int char_no = i++%(17-len);
  lcd.clear();
  lcd.setCursor(char_no, line_no);
  lcd.print(my_name);
  delay(100);

}
```

ابتدا کتابخانه کریستال را اضافه کرده و پورت های متصل به ال سی دی را مشخص میکنیم. سپس رشته نام را مقدار دهی کرده و سپس طول رشته را برابر len میگذاریم. در بخش setup هم طول و عرض کاراکتری ال سی دی را مشخص میکنیم.

حال در بخش loop دو پارامتر line_no و char_no ار مقدار دهی میکنیم و روی آن مدام پیمایش میکنیم. هرچه مقدار i بیشتر شود، موقعیت cursor جلو رفته و برای تنظیم موقعیت بعدی از دو فرمول برای دو پارامتر بالا استفاده میکنیم،در موقعیت عرض، باید هرگاه Mahdi به انتها رسید به خط بعدی برود و برعکس، پس هرگاه به موقعیت ۱۲ طول رسیدیم، باید خط عوض کنیم. در موقعیت طول هم هرگاه به موقعیت ۱۲ رسیدیم باید برویم اول خط.(توجه شود مهدی ۵ کاراکتر دارد و برای همین گفتیم موقعیت ۱۲، ولی کد برای هر اسمی کار میکند)





سوال دوم) یک صفحه کلید از پیش‌فرض‌های پروتئوس اضافه کنید و به برد نصب کنید. با استفاده از آن، رمزی را دریافت کنید و روی LCD نمایش دهید، و با زدن کلید * باید درستی این رمز را (مقدار صحیح رمز به صورت پیش‌فرض باید شماره دانشجویی شما باشد) بررسی کنید. در صورت درستی رمز عبور پیغام "correct password" در خط دوم نمایش داده شود و در غیر اینصورت پیغام "wrong password" نمایش داده شود.

مدار این سوال همان مدار قبلی می‌باشد. حال به بررسی کد میپردازیم:

```
#include <LiquidCrystal.h>
#include <Keypad.h>

// initialize the LCD library with the numbers of the interface pins
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
// initialize the KEYPAD library with the numbers of the interface pins
const int ROW_NUM = 4; //four rows
const int COLUMN_NUM = 3; //three columns
static const uint8_t ANALOG_PINS[] = {A0,A1,A2,A3,A4,A5,A6,A7,A8};
//DEFINE KEYPAD BUTTONS
char keys[ROW_NUM][COLUMN_NUM] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};
//DEFINE KEYPAD CONNECTED PORTS
byte pin_rows[ROW_NUM] = {18, 19, 20, 21}; //connect to the row pinouts of the keypad
byte pin_column[COLUMN_NUM] = {26, 24, 22}; //connect to the column pinouts of the keypad
Keypad keypad = Keypad( makeKeymap(keys), pin_rows, pin_column, ROW_NUM, COLUMN_NUM );
// end of initialize

//set default password
int i = 0;
String password = "9731701";
String input;
int len = password.length();
```

مانند آزمایش ۲ و سوال یک،ابتدا پورت ها را مشخص کرده و صفحه کلید را میسازیم و سپس یک پسوورد(9731701) را معین کرده و یک پارامتر input را تعریف میکنیم و طول پسوورد را درون len میریزیم.

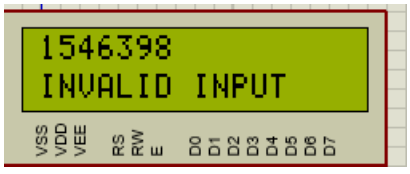
```
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);


}

void loop() {
  //GET CHARACTER OF KEY PRESSED
  char key = keypad.getKey();
  if (key){
    if (key == '*'){
      if (i == 7)
            checkPassword(input, password);
    }
    else if (i < 7 && isDigit(key)){
      input += key;
      lcd.setCursor(i++,0);
      lcd.print(key);
    }
    else{
      lcd.setCursor(0,1);
      lcd.print("INVALID INPUT");
      delay(262);
      lcd.setCursor(0,1);
      lcd.print("                ");
    }

  }

}
```

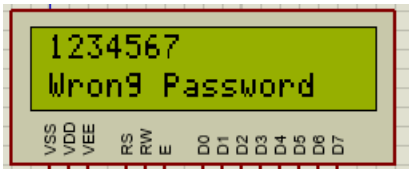
در بخش setup ، ابعاد ال سی دی را مشخص می‌کنیم.در بخش حلقه، اول چک میکنیم ببینیم ببینیم keyیا ورودی داریم یا خیر، اگر داشتیم،اگر ورودی کلید ستاره بود، نگاه میشود که اگر ما ۷ بار ورودی گرفته باشیم(طول رمز هفت است)،می‌آید و تابع checkpassword را صدا میزند و پسوورد اصلی و ورودی را به آن میدهد. اگر وروی کلید ما int بود و کمتر از ۷ بار ورودی گرفته بودیم،ورودی کلید را در input ریخت و کرسر را یکی جلو میبریم وکلید را چاپ میکنیم و عرض آن همیشه صفر میباشد.در غیر دو صورت بالا مینویسیم invalid input و بعد از تاخیر کوچکی به کمک یک روش بسیار اصولی ناپدیدش میکنیم☺.

حال تابع checkpassword را بررسی میکنیم:

```
void checkPassword(String input, String password){
  lcd.setCursor(0,1);
  if (input == password)
    lcd.print("Correct Password");
  else
    lcd.print("Wrong Password");

  delay(662);
  clearLCD();
}

void clearLCD(){
  lcd.clear();
  i = 0;
  input = "";
}
```

این تابع،کرسر را تنظیم کرده و ورودی ساخته شده در لوپ را با پسوورد اصلی چک میکند و پیغام مناسب را چاپ میکند و با کمک تابع clearLCD، i را صفر کرده و ورودی را خالی میکند تا در تلاش های بعدی دچار مشکل نشویم.





چون مدار بخش ۳ متفاوت است اکنون به بخش چهار میپردازیم:

سوال چهارم) با استفاده از توابع کتابخانه ای برنامه ای بنویسید که یک کاراکتر را در ایتدای سطر اول LCDبنویسد و هر ثانیه در حالی که یک خانه به جلو حرکت می کند، به خط بعدی برود(اگر در خط اول بود به خط دوم برود و برعکس). سپس بطور پیوسته این کار را تکرار نماید.

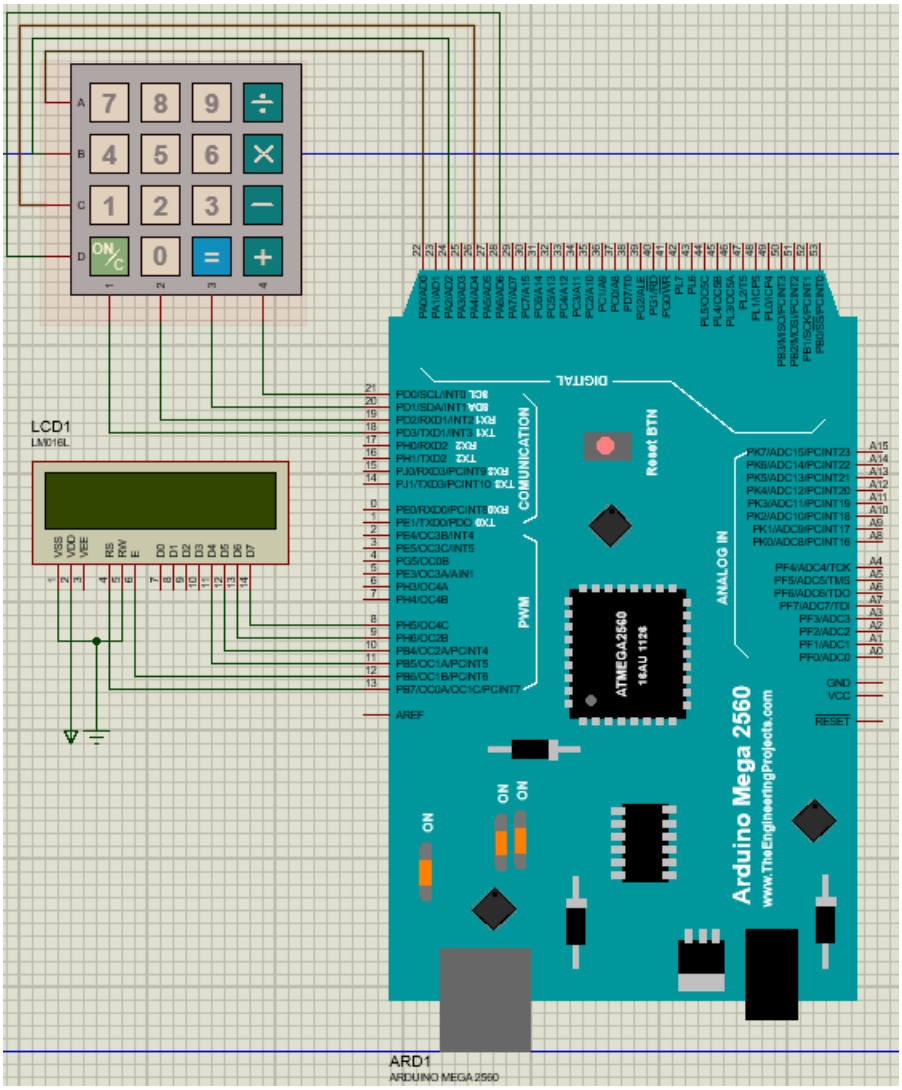
برای این سوال کدی میزنیم که به صورت زیگزاگی حرف M را برایمان چاپ کند در ال سی دی.

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
int i = 0;

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
}

void loop() {
  int line_no = i%2;
  int char_no = i++%16;
  lcd.clear();
  lcd.setCursor(char_no, line_no);
  lcd.print("M");
  delay(100);

}
```

در قسمت حلقه موقعیت cursor را دائما آپدیت کرده و حرف M را چاپ میکنیم با تاخیر ۱۰۰ میلی ثانیه.

سوال سوم) یک کد ماشین حساب بنویسید که بر اساس نوشته‌های کلید (که در LCD مانند 23+59 نوشته شود) دستورات محاسباتی بگیرد و در خط دوم LCD به صورت خروجی نشان دهد. ماشین حساب باید جمع، تفریق، ضرب و تقسیم را انجام دهد.

این بخش از ما یک ماشین حساب میخواهد. ابتدا مدارش را میبینیم:



این بار ازیک صفحه کلید ماشین حساب ماتریسی استفاده میکنیم که عرض آن یک واحد بزرگتر میباشد.

حال به کد میرسیم:

```
#include <LiquidCrystal.h>
#include <Keypad.h>

// initialize the LCD library with the numbers of the interface pins
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
// initialize the KEYPAD library with the numbers of the interface pins
const int ROW_NUM = 4; //four rows
const int COLUMN_NUM = 4; //three columns
static const uint8_t ANALOG_PINS[] = {A0,A1,A2,A3,A4,A5,A6,A7,A8};
//DEFINE KEYPAD BUTTONS
char keys[ROW_NUM][COLUMN_NUM] = {
  {'7','8','9', '/'},
  {'4','5','6', '*'},
  {'1','2','3', '-'},
  {'c','0','=', '+'}
};
//DEFINE KEYPAD CONNECTED PORTS
byte pin_rows[ROW_NUM] = {22, 24, 26, 28}; //connect to the row pinouts of the keypad
byte pin_column[COLUMN_NUM] = {18, 19, 20, 21}; //connect to the column pinouts of the keypad
Keypad keypad = Keypad( makeKeymap(keys), pin_rows, pin_column, ROW_NUM, COLUMN_NUM );
// end of initialize

//define inputs and which input
String inputs[3];
char op = NULL;
int i = 0 , no = 0;
```

در این بخش ابتدا صفحه کلید را ساخته وپورت ها را مشخص میکنیم متغیر های مورد نیاز از جمله یک رشته سه بخشی که بعدا دو عدد و یک عملگر در آن ریخته می‌شود ساخته میشود.

```
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  //BEGIN SERIAL
  Serial.begin(9600);

}

void loop() {
  //GET CHARACTER OF KEY PRESSED
  char key = keypad.getKey();
  checkKey(key);
}
```

در این تکه از کد ابعاد ال سی دی را مشخص میکنیم و سریال را آغاز میکنیم.سپس در حلقه کلید فشرده شده را گرفته و به تابع check key که در ادامه به آن میپردازیم، پاس میدهد.

```
void checkKey(char key){
  if (key){
    if (key == 'c')
      clearLCD();
    else if (isDigit(key)){
      inputs[no] += key;
      lcd.setCursor(i++,0);
      lcd.print(key);
    }
    else if (key == '=' and op != NULL){
      int result = calculate(inputs[0],inputs[1],op);
      lcd.setCursor(0,1);
      lcd.print(result);
    }
    else if (op == NULL){
      op = key;
      no ++;
      lcd.setCursor(i++,0);
      lcd.print(key);
    }
    else{
      lcd.setCursor(0,1);
      lcd.print("INVALID INPUT");
      delay(100);
      lcd.setCursor(0,1);
      lcd.print("                ");
    }


  }

}
```

در اینجا اول میبینیم که ورودی داریم، اگر داشتیم ابتدا چک میکنیم اگر c بود تابع clearLCD را که بعدامیبینیم فراخوانی میکنیم و برنامه و ال سی دی خالی میشود. سپس چک میکنیم اگر کلید فشرده شده عدد باشد، کاراکتر ورودی وارد بخشی میشود که در آن هستیم و با no پیمایش میشود.سپس cursor جلو میرود و کاراکتر مورد نظر چاپ میشود.

سپس چک میشود اگر عملگر نال نبود، یعنی یکی از ۴ عمل بود و ورودی = بود، تابع calculate فراخوانده میشود و عملگر و دو عدد به آن پاس داده میشود و نتیجه در خط ۲ چاپ میشود.

حال ورودی اگر عدد بود یا = یا c و op نال بود، شرط داریم، پس اگر وارد اینها نشود،چک میشود اگر هنوز عملگر نداشتیم، این ورودی عمل گر است و سپس no را اضافه میکنیم و سپس عملگر را چاپ میکنیم.

در شرط آخر اگر هیچکدام از حالت های بالا نبود، میگوییم ورودی معتبر نیست و سپس پیام را پاک میکنیم به همان روش حرفه ای☺.

```
unsigned long calculate(String input1, String input2, char operat){
  unsigned long in1 = ( unsigned long ) (input1.toInt());
  unsigned long in2 = ( unsigned long ) (input2.toInt());
  switch (operat){
    case '+':
      return in1 + in2;
      break;
    case '-':
      return in1 - in2;
      break;
    case '*':
      return in1 * in2;
      break;
    case '/':
      return in1 / in2;
      break;

  }
}

void clearLCD(){
  lcd.clear();
  inputs[0] = "";
  inputs[1] = "";
  no = 0;
  i = 0;
  op = NULL;
}
```

در آخر هم تابع calculate و clearlcd را میبینیم.