فصل سوم

تجهیزات ورودی و خروجی

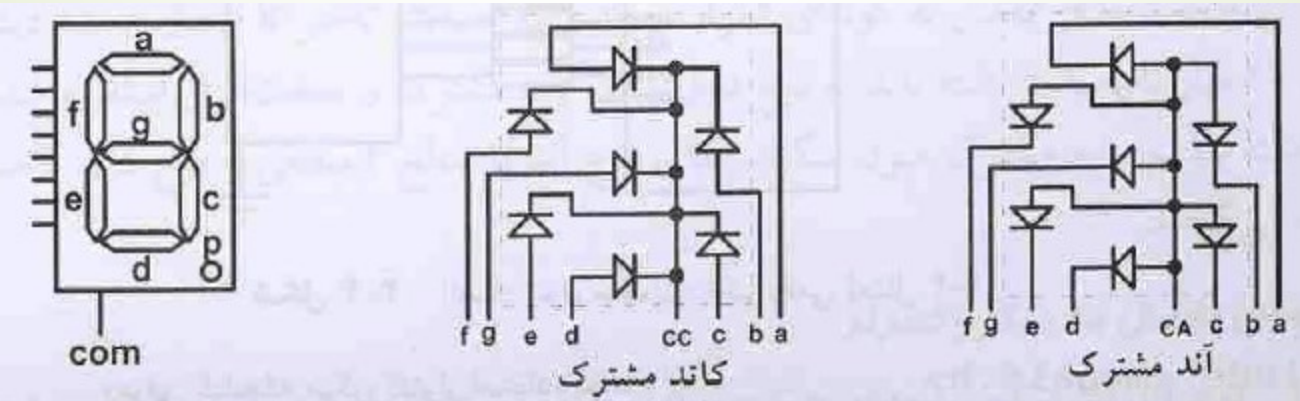
اهداف

1. آشنایی با نمایشگر سون سگمنت تک رقمی
2. بکارگیری روش مالتی پلکسری برای سون سگمنت های چند رقمی
3. آشنایی و نحوی کار با نمایشگر LCD
4. ایجاد کردن کاراکتر دلخواه در LCD
5. اسکن صفحه کلید ۴ × ۴

نمایشگر سون سگمنت (7segment display)

یکی از نمایشگرهای پر استفاده در میکروکنترلرسون سوگمنت ها در ابعاد وارقام مختلف می باشد . این نوع نمایشگر ها از هفت قسمت (LED) تشکیل شده است به همین دلیل به آن سون سگمنت گفته می شود.

این نمایشگر ها به خاطر نوردهی و اندازه فیزیکی آنها نسبت به LCD اجعیت دازند . برای اتصال سون سگمنت به میکروکنترلر نیاز به امانی مانند 7447 یا 7448 است که کد باینری یا BCD را به کد سون سگمنت تبدیل کند اما برای کاهش سخت افزار جانبی می توان کد سون سگمنت را در داخل ایجاد کرد.



شکل ۴-۱ نمای سون سگمنت تک رقمی و ترکیب داخلی آن

نمایشگر سون سگمنت (7segment display)

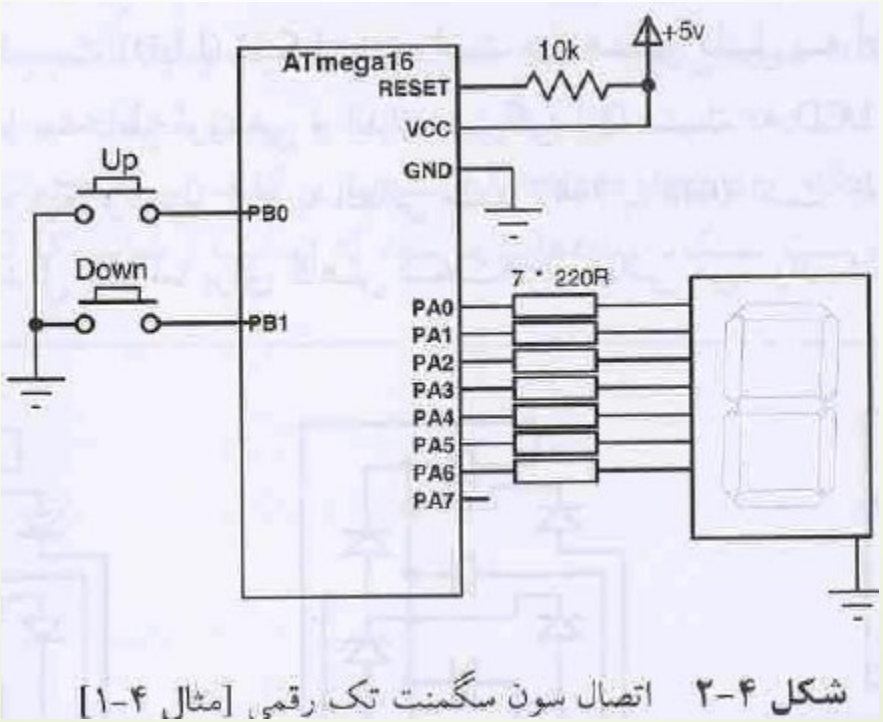
آند مشترک									
عدد	کد	p	g	F	E	d	c	b	A
0	C0	1	1	0	0	0	0	0	0
1	F9	1	1	1	1	1	0	0	1
2	A4	1	0	1	0	0	1	0	0
3	B0	1	0	0	1	0	0	0	0
4	99	1	0	0	1	1	0	0	1
5	92	1	0	0	1	0	0	1	0
6	82	1	0	0	0	0	0	1	0
7	F8	1	1	1	1	1	0	0	0
8	80	1	0	0	0	0	0	0	0
9	90	1	0	0	1	0	0	0	0

کاتد مشترک									
عدد	کد	p	G	f	e	d	c	b	a
0	3F	0	1	1	1	1	1	1	1
1	06	0	0	0	0	0	1	1	0
2	5B	0	1	0	1	1	0	1	1
3	4F	0	1	0	0	1	1	1	1
4	66	0	1	1	0	0	1	1	0
5	6D	0	1	1	0	1	1	0	1
6	7D	0	1	1	1	1	1	0	1
7	07	0	0	0	0	0	1	1	1
8	7F	0	1	1	1	1	1	1	1
9	6F	0	1	1	0	1	1	1	1

جدول ۴-۱ کدهای سون سگمنت نوع آند مشترک و کاتد مشترک

نمایشگر سون سگمنت (7segment display)

مثال ۴-۱ : برنامه ای بنویسید که وضعیت دو کلید فشاری UP وDown را برسی نماید و مفدار عددی نمایشگر تک رقمی نوع کاتد مشترک را افزایش و کاهش دهد؟



شکل ۴-۲ اتصال سون سگمنت تک رقمی [مثال ۴-۱]

نمایشگر سون سگمنت (7segment display)

```
#include  <mega16.h>
#include  <delay.h>
Flash  unsigned  char display [ ] ={ // تعریف آرایه ثابت حاوی کد های سون سگمنت
0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,0x7d, 0x7f, 0x6f };
Void  main ( ) {                    // تابع اصلی برنامه
Unsigned  char  i;                  // تعرف متغیر شمارشگر
PORTA=0x3f;                 //ابتدا بر روی سون سگمنت عدد صفر را نشان می دهیم
  DDRA=0xff;              //پورت A متصل به سون سگمنت را خروجی تعیین می کنیم
     PORTB=0x03;                    // فعال کردن مقاومت pull-up پایه های متصل به کلید ها
      DDRB=0x00;                  // تعیین پورت B به عنوان ورودی
While (1) {                       // حلقه بی نهایت برای خواندن مکرر وضعیت کلید ها
    if (PINB.0==0 && I<9) {       // اگر کلید UP فشرده شود و متغیر شمارشگر کوچکتر از ۹ باشد
i++;                              // یک واحد به متغیر شمارشگر اضافه می کنیم
                                  // UPتست برای رها شدن کلید
While (PINB.0==0);
```

}

نمایشگر سون سگمنت (7segment display)

```
If (PINB.1==0 && I!=0 )  // اگر کلید down فشرده شود و متغیر شمارشگر مخالف صفر باشد
                         {
i--;                     // یک واحد متغیر شمارشگر را کاهش می کنیم
                    While //Down تست برای رها شدن کلید
                                (PINB.0==0);
}
PORTA=display [ i ] ; کد سون سگمنت معادل متغیر شمارشگر ، به خروجی ارسال می
        //شود.
} ;
}
```

نمایشگر سون سگمنت (7segment display)

روش مالتی پلکسری (Multiplexing Method)

اگر نیاز باشد از نمایشگر های سون سگمنت چند رقمی استفاده کنیم این مسئله بوجود می آید که با محدودیت پورت میکروکنترلر مواجه می شویم . برای این منظور از روش refresh یا تازه سازی استفاده می شود این عمل بر اساس خطای چشم انسان صورت می گیرد .

اگر یک LED را ۵۰ بار در ثانیه روشن و خاموش کنیم چشم انسان LED را مدام روشن می بیند .بنابراین اگر پایه های مشترک سون سگمنت را در اختیار میکروکنترلر قرار دهیم و کدها را به صورت خروجی یک دیکدر ، با پایه های مشترک ارسال کنیم و مانند یک مالتی پلکسر داده های نمایشی را با زمانی مشخص تسهیم کنیم ، می توانیم ادعا کنیم که تمام داده های نمایشی را توانسته ایم نمایش دهیم.

روش بدین صورت است که هر بار یکی از سگمنت ها را انتخاب کرده و دیتای آن سگمنت را ارسال می کنیم و این عمل را برای سگمنت های بعد ی نبه گونه ای تکرار می کنیم که حدا کثر تاخیر تا ارسال مجدد دیتای سگمنت اول ۲۰، میلی ثانیه باشد .

البته باید به نوع فعال شدن پایه مشترک و سخت افزار استفاده شده ، توجه لازم را داشته باشیم .

پایه مشترک سون سگمنت های نوع آند با سطح 1 منطقی و نوع کاتد با سطح 0 منطقی فعال می گردند.

نمایشگر سون سگمنت (7segment display)

نحوه اتصال به میکروکنترلر

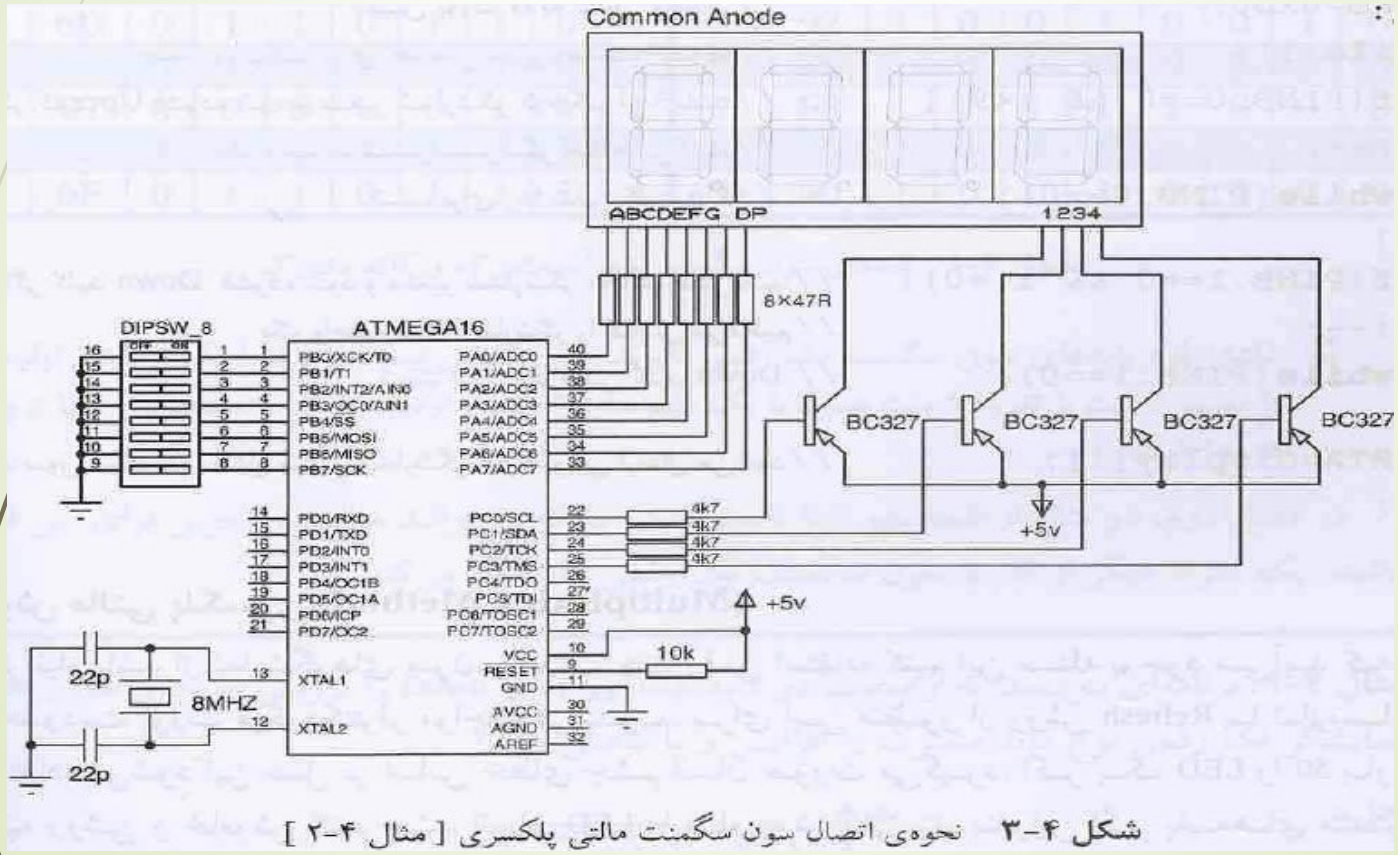
1. **نوع آند مشترک :** پایه های a تا g وp به یک پورت دلخواه متصل می گردد و پایه های مشترک توسط ترانزیستور مطابق شکل ۴-۳ به چهار پایه از یک پورت دلخواه دیگر متصل می گردد.

2. **نوع کاتد مشترک :** پایه های a تا g و p با مقاومت های 150 اهم به یک پورت دلخواه متصل شده و پایه های مشترک ، مستقیما به چهار پایه از یک پورت دلخواه دیگر متصل می گردد . زیرا با مقاوت ۱۵۰ اهم ، حداکثر از هر پایه پورت 34mA، کشیده خواهد شد.

نمایشگر سون سگمنت (7segment display)

مثال ٤-٢ : یک عدد ۸ بیتی از یک دسته کلید متصل به پورت B بخوانید و آن عدد را بر روی نمایشگر سون سگمنت از نوع آند مشترک چهار رقمی به روش مالتی پلکسری نمایش دهید.

حل:



شکل ۴-۳ نحوه‌ی اتصال سون سگمنت مالتی پلکسری [مثال ۴-۲]

نمایشگر سون سگمنت (7segment display)

```
#include <mega16.h>
#include <delay.h>
Unsigned char part1=0, part2=0, part3=0, part4=0;      //متغیر های سگمنت ها
Flash unsigned char c7seg[]={                          //کد های سون سگمنت ذخیره شده در حافظه ثابت
0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80,0x90};
Void HEX_to_seg(unsigned char k) {     // تابع تبدیل عدد از هگزاد به کد سون سگمنت
Part4= c7seg[k%10];                    // تبدیل یکان به کد معادل سون سگمنت
Part3= c7seg[k/10%10];                 // تبدیل دهگان به کد معادل سون سگمنت
Part2= c7seg[k/10/10];                 // تبدیل صدگان به کد معادل سون سگمنت
Part1=c7seg[0] ;                       //عدد ۸ بیتی است و سگمنت سمت چپ همیشه صفر است
}
Void main () {
Unsigned char number=0 ;               // تعریف متغیر ۸ بیتی برای خواندن ورودی
PORTA=0xFF;                            // مقدار اولیه جهت خاموش بودن تمام سگمنت ها
                                       // تعیین پورت A به عنوان خروجی
DDRA=0xFF;
```

نمایشگر سون سگمنت (7segment display)

```
// فعال کردن مقاومت pull-up پورت  B متصل به کلید ها
PORTB=0xFF;
// تعیین پورت B به عنوان ورودی
DDRB=0x00;
 PORTC=0xFF;                    // مقدار اولیه جهت خاموش بودن تمام سگمنت ها
 DDRC=0x0F;                     // تعیین نیبل کم ارزش پورت C به عنوان خروجی
      while (1) {    // حلقه بی نهایت جهت خواندن ورودی و نمایش مکرر عدد خوانده شده
Number = PINB;                  // خواندن پورت ورودی متصل به کلید ها
HEX_to_seg(number);             //فراخوانی تابع جهت تبدیل کد سون سگمنت
PORTC=0b00001110 ;              //انتخاب سگمنت یکان
PORTA=part1;                    //ارسال عدد یکان
Delay_ms (5);                   // تاخیر زمانی به مدت ۵ میلی ثانیه جهت تازه سازی
PORTC=0b00001101;               //انتخاب سگمنت دهگان
      PORTA=part2;              // ارسال عدد دهگان
Delay_ms (5);                   // تاخیر زمانی به مدت ۵ میلی ثانیه جهت تازه سازی
PORTC=0b00001011;               //انتخاب سگمنت صدگان
```

نمایشگر سون سگمنت (seven segment display)

```
PORTA=part3;                ارسال عدد صدگان//
Delay_ms (5);               تاخیر زمانی به مدت ۵ میلی ثانیه جهت تازه سازی //
PORTC=0b00000111;           انتخاب سگمنت هزارگان//
PORTA=part4;                ارسال عدد هزارگان//
Delay_ms (5);               تاخیر زمانی به مدت ۵ میلی ثانیه جهت تازه سازی //
} ;
}
```

نمایشگر LCDمتنی

امروزه در اکثر پروژه های میکروکنترلری از نمایشگر LCD استفاده می شود. این نمایشگر توانایی نمایش تمام حروف های موجود بر روی صفحه کلید مانیتور را دارد و از مزیت های آن نسبت به سون سگمنت این است که ارقام و حروف های بیشتری را می توان نمایش داد و نیازی به تازه سازی اطلاعات جهت نمایش نمی باشد یعنی اگر یک کاراکتر را به LCD ارسال کنیم ، آن کاراکتر نمایش داده می شود و نیازی نیست دوباره همان کاراکتر را ارسال کنیم تا از بین نرود.

نمایشگر های LCD دارای دستوراتی می باشند که کاربر می تواند بنا به نیاز خود از آنها استفاده کند.

در جدول ۴-۳ فهرستی از دستورات LCD آورده شده که دستورات برای تمام LCD ها در ابعاد مختلف یکسان است.

نمایشگر LCDمتنی

دستور	عملکرد	دستور	عملکرد
0x01	صفحه نمایش پاک شود	0x0E	نمایش روشن و مکان نما روشن
0x02	بازگشت به مکان اول	0x0F	نمایش روشن و مکان نما چشمک بزنند.
0x04	جابجایی مکان نما به چپ	0x10	جابجایی محل مکان نما به چپ
0x06	جابجایی مکان نما به راست	0x14	جابجایی محل مکان نما به راست
0x05	جابجایی کاراکترها به راست	0x18	همهٔ کاراکترهای نمایش داده شده، یک خانه به چپ جابجا می‌شوند.
0x07	جابجایی کاراکترها به چپ	0x1C	همهٔ کاراکترهای نمایش داده شده، یک خانه به راست جابجا می‌شوند.
0x08	نمایش خاموش و مکان نما خاموش	0xC0	مکان نما به آغاز خط دوم می‌رود.
0x0A	نمایش خاموش و مکان نما روشن	0x38	سازماندهی ۸ بیتی و ماتریس ۵×۸
0x0C	نمایش روشن و مکان نما خاموش	0x28	سازماندهی ۴ بیتی و ماتریس ۵×۸

جدول ۴-۳ دستورات نمایشگر LCD

کتابخانه lcd.h

فایل کتابخانه ای LCD دارای توابعی به صورت زیر است و باید ابتدای برنامه معرفی گردد.

Void_lcd_ready(void);

این تابع پرچم مشغول LCDرا بررسی می کند.

Void _lcd _write_data(unsigned char data);

برای ارسال فرمان به LCD استفاده می شود.

Void _lcd_write_byte (unsigned char addr,unsigned char data);

برای نوشتن یک بایت کاراکتر دلخواه یا نمایشی در RAM داخلی استفاده می شود.

unsigned char lcd_read_byte (unsigned char addr);

برای خواندن یک بایت کاراکتر دلخواه یا نمایشی در RAM داخلی استفاده می شود.

Void lcd _gotoxy (unsigned char x, unsigned char y);

برای تعیین موقعیت (ستون x و برای تعیین سطر y)نمایش کاراکتر LCD است.

Void lcd_clear(void);

این تابع کل صفحه نمایش LCD را پاک می کند.

Void lcd_putchar (char c);

برای ارسال یک کاراکتر به LCD استفاده می شود.

Void lcd_puts (char *str);

برای نمایش یک رشته کاراکتر ذخیره شده در RAM میکروکنترلر به LCD استفاده میشود.

Void lcd_putsf (char flash *str);

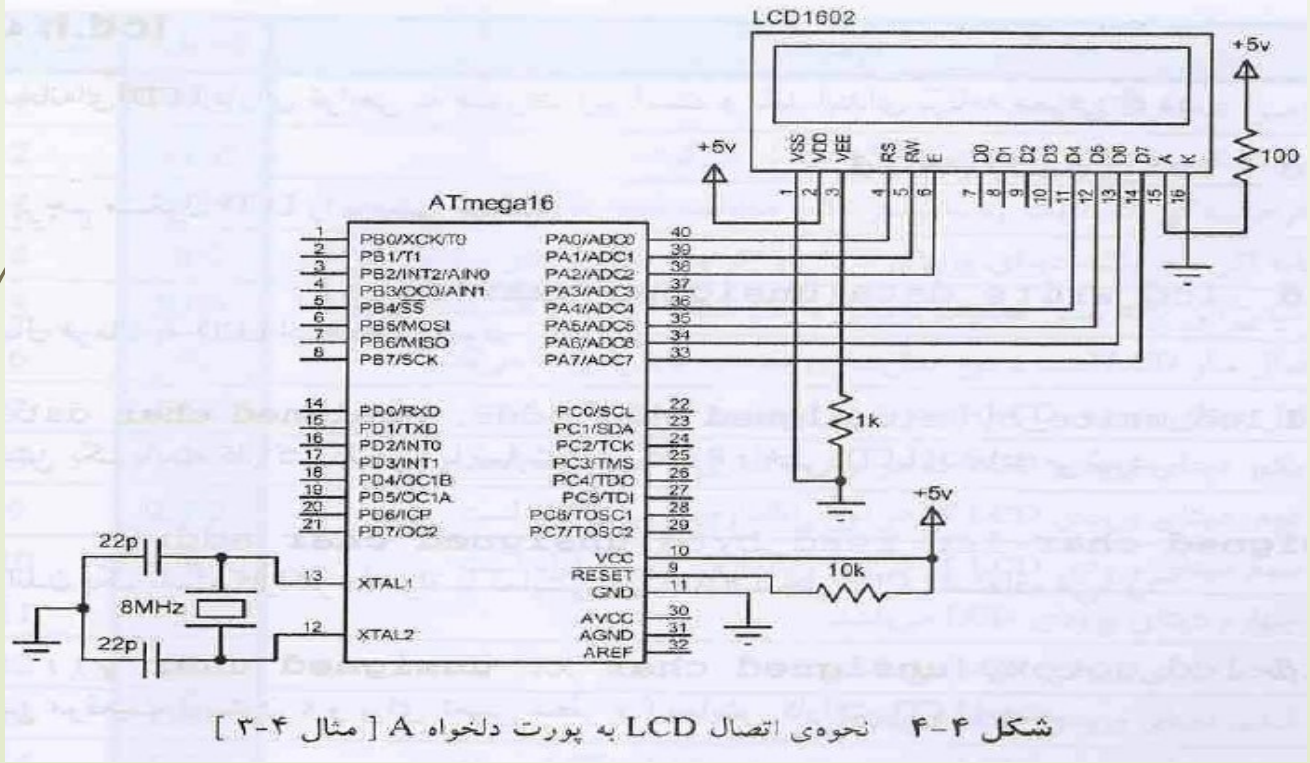
برای نمایش یک رشته کاراکتری ذخیره شده در flash میکروکنترلر به LCD استفاده می شود.

Unsigned char lcd_init(unsigned char lcd_columns);

برای مقدار دهی اولیه و تعیین ستون LCD این تابع در ابتدای برنامه فراخوانی می شود.

مثال ٤-٣ : ابتدا بر روی سطر اول رشته ثابت " LCD test " را نمایش داده و سپس بر روی سطر دوم رشته ثابت "ATmega16" را نمایش دهید و به انتهای سطر اول رفته و کاراکتر * را نمایش دهید و فرمان چشمک زدن مکان نما را ارسال نمائید.

حل :



شکل ۴-۴ نحوه‌ی اتصال LCD به پورت دلخواه A [مثال ۴-۳]

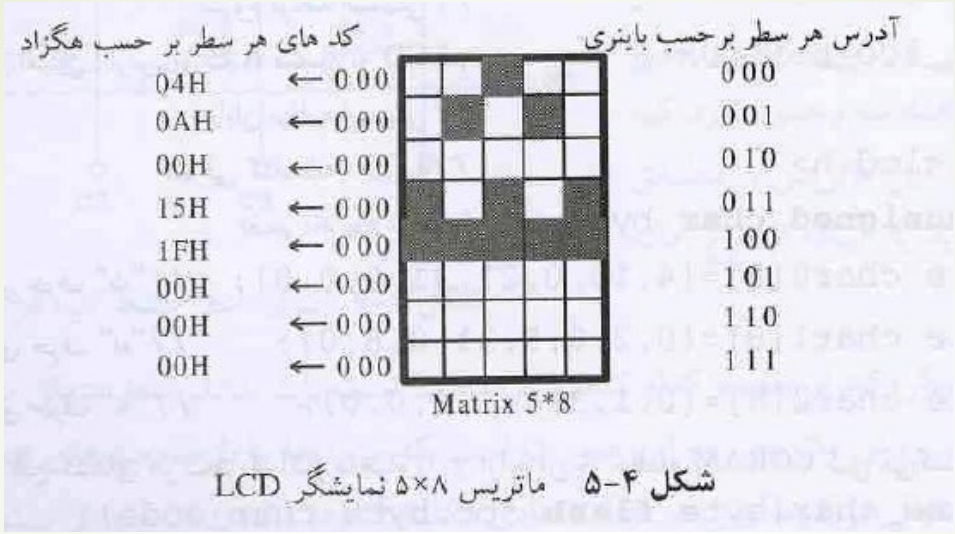
نمایشگر LCDمتنی

```
#include <mega16.h>
#asm                                    //شروع برنامه اسمبلی
    .equ_lcd_port=0x1B                  تعیین آدرس پورت A متصل به LCD :
#endasm                                 // پایان برنامه اسمبلی
        #include <lcd.h>                // معرفی کتابخانه نمایشگر LCD
Void main () {
          Lcd_init (16);                // تعیین lcd با 16 ستون و مقدار دهی اولیه
Lcd_clear ();                           // پاک کردن صفحه نمایش ال سی دی
Lcd_gotoxy(2,0);                        // رفتن به ستون سوم و سطر اول ال سی دی
Lcd_putsf("lcd test");                  //نمایش رشته کاراکتری ذخیره شده در حافظه فلاش
Lcd_gotoxy(5,1);                        //رفتن به ستون ششم و سطر دوم ال سی دی
Lcd_putsf("ATmega16");                  //نمایش رشته کاراکتری ذخیره شده در حافظه فلاش
```

```
Lcd_gotoxy(14,0);                رفتن به سطر اول و ستون پانزدهم ال سی دی//
Lcd_putchar( '*');               نمایش کاراکتر *//
Lcd_gotoxy(14,0);                رفتن به سطر اول و ستون پانزدهم ال سی دی//
Lcd_ready();                     خواندن پرچم مشغولال سی دی//
Lcd_write_data(0x0f);  ارسال فرمان نمایش روشن و مکان نما پشمک
        بزند//
While (1);                       حلقه بی نهایت و بیکار برنامه //
}
```

ایجاد کردن کاراکتر دلخواه در LCD متنی

در LCD های متنی ،حافظه ای بنام CGRAM وجود دارد که می توان ۸ کاراکتر دلخواه در آن ایجاد کرد .
نمایشگر های LCD از ماتریس های کوچک ۵*۸ یا ۵*۷ ایجاد شده اند بنابراین برای نمایش کاراکتر دلخواه باید کد مربوط به سطر ماتریس را بدست بیاوریمور در حافظه CGRAM آن کاراکتر را ایجاد کنیم.
به طور نمونه حرف "ش" را به صورت شکل ۴-۵ ایجاد می کنیم و کد هر سطر را بدست می آوریم ودر آرایه برنامه قرار می دهیم.
به طور نمونه حرف "ش" را به صورت شکل ۴-۵ ایجاد می کنیم و کد هر سطر را بدست می آوریم ودر آرایه برنامه قرار می دهیم.



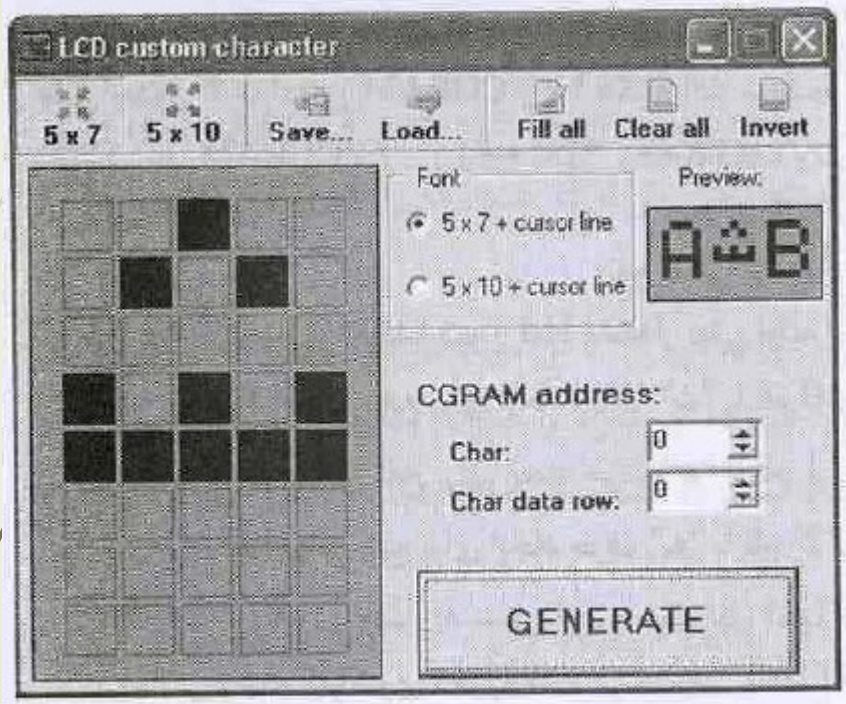
شکل ۴-۵ ماتریس ۵×۸ نمایشگر LCD

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	0	0	0	0	0	0
				0	0	1	0	0	1
				0	1	0	0	1	0
				0	1	1	0	1	1
				1	0	0	1	0	0
				1	0	1	1	0	1
				1	1	0	1	1	0
				1	1	1	1	1	1
شرط دستور		شرط دستیابی به آدرس CGRAM		محل ذخیره سازی کاراکتر (نام کاراکتر برای فراخوانی)			آدرس هر سطر از کاراکتر دلخواه		
				آدرس CGRAM					

جدول ۴-۵ نحوه‌ی دستیابی CGRAM برای ایجاد کاراکتر دلخواه

نمایشگر LCDمتنی

شکل ۴-۶ نمونه هایی از یک برنامه کامپیوتری جهت در آوردن کد های کاراکتر دلخواه را نمایش می دهد.
با کلیک چپ بر روی هر خانه از ماتریس می توان آن را روشن و یا خاموش کرد



شکل ۴-۶ پنجره نرم‌افزار ایجاد کننده کاراکتر دلخواه

مثال ٤-٤: کلمه "شنبه" را در حافظه گرافیکی LCD ایجاد کرده و آن را نمایش دهید . از سخت افزار شکل ۴-۴ استفاده کنید.

حل:

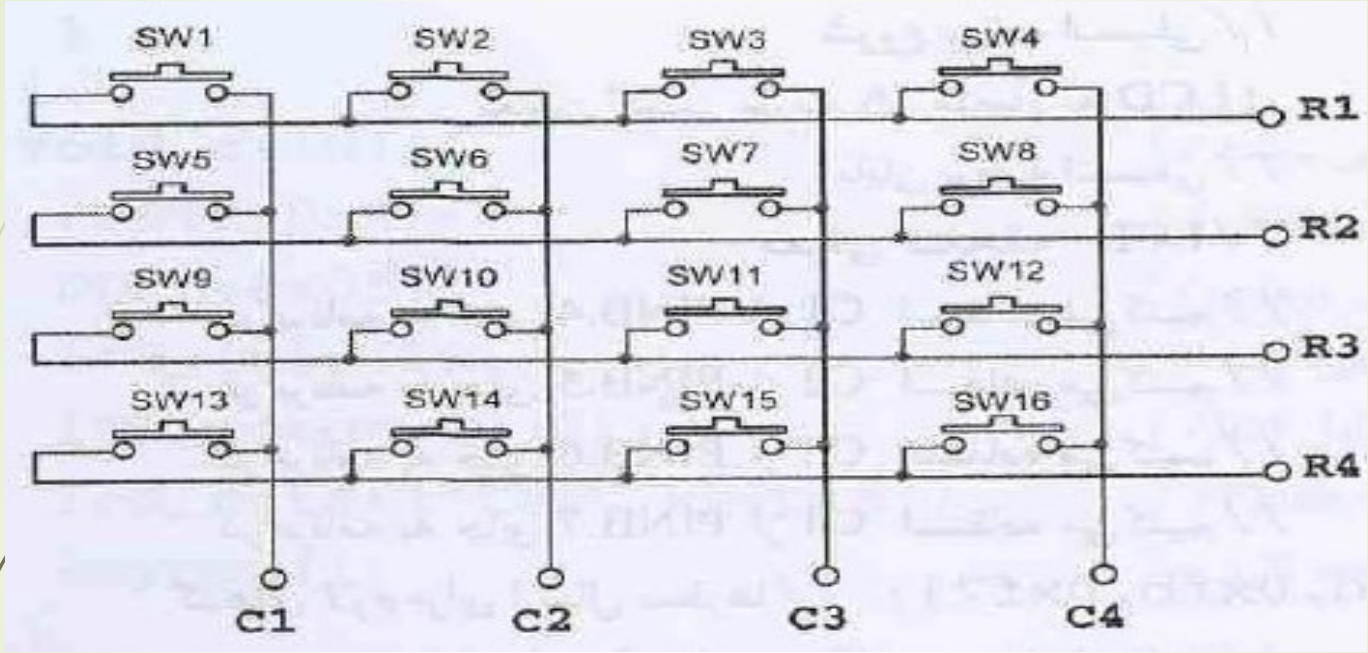
```
#include <mega16.h>
#asm                                    //شروع برنامه اسمبلی
                                        تعیین آدرس پورت A متصل به LCD :
                                        .equ_lcd_port=0x1B
#endasm                                 // پایان برنامه اسمبلی
#include  <lcd.h>                       // معرفی کتابخانه ال سی دی
Typedef  unsigned   char  byte;         //تغیر نام نوع داده
Flash byte  char0[8]={4,10,0,21,31,0,0,0};  کدهای حرف ش
   //
Flash byte  char1[8]={0,2,0,5,31,0,8,0}; کدهای حرف
          //نب
Flash type  char2[8]={0,1,3,3,1,0,0,0};  //کدهای حرف ه
```

```
-- >>> این تابع کد های هر سطر کاراکتر دلخواه را از آرایه آن به حافظه CGRAM کپی میکند<<< -- //
Void  define _char (byte flash *pc,byte char_code) {
Byte  i,a;                                          //معرفی دو متغیر ۸ بیتی بدون علامت
          a=(char_code<<3) 0x40;                    //طبق جدول ۴-۵ آدرس CGRAM از ۴۰ هگزاد آغاز می شود.
For  (i=0;  i<8;  i++) lcd_write_byte(a++, *pc++);  //نوشتن در CGRAM
}
Void  main () {
Lcd  _init(16);                                     //تعیین تعداد ستون  ال سی دی و مقدار دهی اولیه
    Define_char (char0,0);                          //فراخوانی تابع برای ایجاد حرف "ش"  با نام 0
          Define_char (char1,1);                    //فراخوانی تابع برای ایجاد حرف "نب"  با نام 1
          Define_char (char2,2);                    //فراخوانی تابع برای ایجاد حرف "ه"  با نام 2
Lcd_gotoxy(0,0);                                    //رفتن به ستون اول و سطر اول ال سی دی
Lcd_putsf ("display day: ") ;                       //نمایش رشته کاراکتری ذخیره شده در حافظه فلاش
Lcd_putchar (2) ;                                   //نمایش حرف  ه
Lcd_putchar(1) ;                                    //نمایش حرف  نب
Lcd_putchar(0) ;                                    //نمایش حرف  ش
While (1) ; {                                       //حلقه بی نهایت
};      }
```

اسکن صفحه کلید 4 × 4

در بعضی از پروژه ها لازم است تا میکروکنترلر اطلاعاتی را توسط کاربر جهت تنظیم کردن ،در یافت کند.
به طور نمونه یک میکروکنترلر ۴۰ پایه ، دارای ۳۲ خط ورودی و خروجی است. بنابراین برای اتصال ۱۶ کلید فشاری ،به نصفی از ورودی و خروجی ها نیاز داریم ولی می توان ۱۶ کلید فشاری را طوری به میکروکنترلر وصل نمود که با ۸ پایه ،بتواند کلید ها را تشخیص دهد برای این منظور از روش ماتریسی استفاده می شود.
نمونه یک صفحه کلید ۴ × ۴ در شکل ۴-۷ نمایش داده شده است.

صفحه کلید ٤ × ٤



برای اینکه میکروکنترلر بفهمد کدام کلید فشرده شده است باید بداند ،کلید متعلق به کدام سطر و کدام ستون است. روش ماتریسی به این صورت است که ستون ها را با مقاومت pull-up داخلی یا بیرونی به 5v+ متصل می کنیم.

صفحه کلید ٤ × ٤

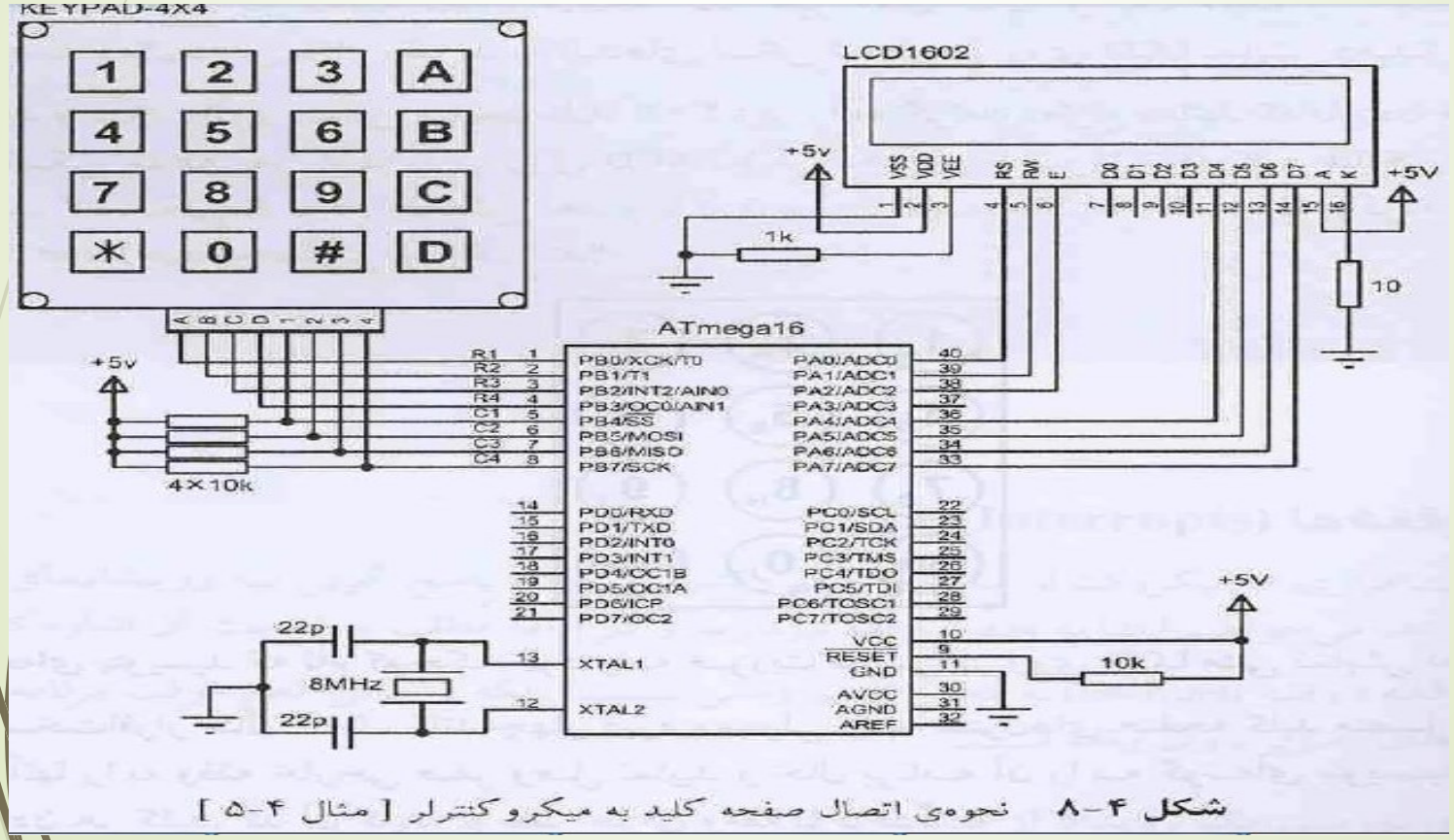
در حالت عادی (کلید فشرده نشده) هر چهار ستون در وضعیت 1 منطقی هستند و جهت اسکن صفحه کلید باید به سطرها به صورت دیکدر اکتیو low دیتا بفرستین و برای ستون ها را برای صفر شدن بررسی نمائیم.

ارسال کد سطرها با سرعت بالایی انجام می گیرد . در شکل ۴-۷ فرض کنید R1 را صفر کرده و مابقی سطر ها را یک کرده ایم و در این حالت کلید SW1 فشرده می شود و ستون C1 را صفر می کند.

همچنین برای لرزش گیری ،از یک تاخیر زمانی به مدت 50ms استفاده می کنیم و کلید فشرده شده را تست می کنیم که این کلید رها شده باشد و به معنی دوبار فشردگی نیز محسوب نشود.

مثال ٤-٥: برنامه ای بنویسید که صفحه کلید ۴ × ۴ را اسکن کرده و کد مربوط به هر کلیدی را بر روی LCD نمایش دهد در این برنامه ستون ها را ورودی و سطرها را خروجی در نظر بگیرید.

حل:



شکل ۴-۸ نحوه‌ی اتصال صفحه کلید به میکروکنترلر [مثال ۴-۵]

صفحه کلید ٤ × ٤

```
#include  <mega16.h>;
#include  <delay.h>;
#asm
  #equ_lcd_port=0x1B                تعیین آدرس پورت A متصل به ال سی دی//
   #define c1 PINB.4                در برنامه جاری به جای PINB.4 از c1 استفاده می کنیم//
   #define c2 PINB.5                در برنامه جاری به جای PINB.5 از c2 استفاده می کنیم//
   #define c3 PINB.6                در برنامه جاری به جای PINB.6 از c3 استفاده می کنیم//
   #define c1 PINB.7                در برنامه جاری به جای PINB.7 از c4 استفاده می کنیم//
  falsh  char  row []={0xfe,0xfd,0xfb,0xf7};          // کد های لازم برای ارسال سطرها
  flash  char data_key[]={                          قرار دادن کد های هر کلید در حافظه ثابت
' 1' , '2' , '3' , 'A' ,                          // کد های اسکی سطر اول صفحه کلید
' 4' , '5' , '6' , 'B' ,                          // کد های اسکی سطر دوم صفحه کلید
' 7' , '8' , '9' , 'C' ,                          // کد های اسکی سطر سوم صفحه کلید
' *' , '0' , '#' , 'D' ,                          // کد های اسکی سطر چهارم صفحه کلید
Unsigned char  ac ,table;                         // معرفی متغیر های کلی و ۸ بیتی
     unsigned  int  r;                            // معرفی متغیر کلی و ۱۶ بیتی
```

صفحه کلید ٤ × ٤

```
Void keypad () {                                  //تابع اسکن صفحه کلید
                                  // LCDرفتن به ستون اول و سطر دوم
Lcd_gotoxy(0,1);
 lcd_putsf(“ ~ ”);                                // نمایش کاراکتر  فلش
          While (1) {                             // حلقه بی نهایت برای اسکن مداوم
        For  (r=0; r<4; r++) {                    // ایجاد یک حلقه ،برای  ارسال  ۴  دیتا دیکدری به سطر ها
Ac=4;                                  //. متغیر ac را ۴ قرار می دهیم و بعدا آن را تست می کنیم
 PORTB=row [r];          //متغیر r توسط حلقه افزایش و کد های سطر ها را بر می گرداند و به پورت می فرستد
DDRB=0x0f;                                        // نیبل پایینی خروجی و نیبل بالایی ورودی
       If  (c1==0)     ac=0;                      //اگر کلیدی از ستون اول فشرده شد متغیر ac=0
       If  (c2==0)     ac=1;                      //اگر کلیدی از ستون اول فشرده شد متغیر ac=1
        If  (c3==0)     ac=2;                     //اگر کلیدی از ستون اول فشرده شد متغیر ac=2
       If  (c4==0)    ac=3;                      //اگر کلیدی از ستون اول فشرده شد متغیر ac=3
          if (!(ac==4)) {                         //اگر متغیر acبرابر با 4 نباشد نشان دهنده فشردگی کلید است.
table=data_key[ (r*4) +ac] ;                      //این فرمول کد اسکی کلید را بر می گرداند
Lcd_putchar (table) ;                             // نمایش کاراکتر برگشتی از آرایه
```

صفحه کلید ٤ × ٤

```
While (c1==0) {}                    //تست برای فشردگی کلید های ستون اول
  While (c2==0) {}                  //تست برای فشردگی کلید های ستون دوم
While (c3==0) {}                    //تست برای فشردگی کلید های ستون سوم
While (c4==0) {}                    //تست برای فشردگی کلید های ستون چهارم
  delay_ms(50) ;                    //تاخیر زمانی به مدت ۵۰ میلی ثانیه
}
}
}
}
Void main ( ) {
 PORTB=0xff ;                       // فعال کردن مقاومت بالا کش داخلی
 DDRB=0x0 ;                         //نیبل پاینی خروجی و نیبل بالایی ورودی
lcd_init (16) ;                     //تعیین ستون ال سی دی و مقدار دهی اولیه
lcd_gotoxy ( 0, 0 ) ;               //رفتن به سطر اول و ستون اول ال سی دی
lcd_putsf ( "test  keypad" ) ;      // نمایش رشته کاراکتری ذخیره شده در حافظه فلاش
Keypad () ;                         //فراخوانی یابع اسکن صفحه کلید
}
```