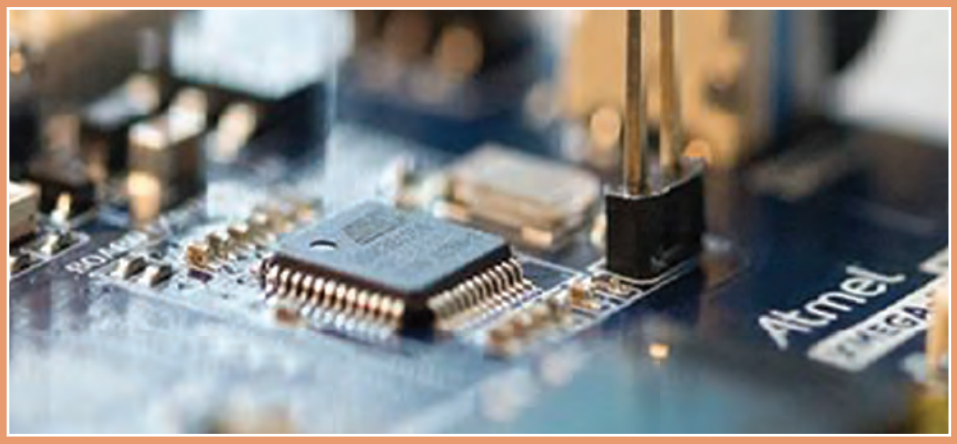
پودمان ۲

برنامه‌نویسی



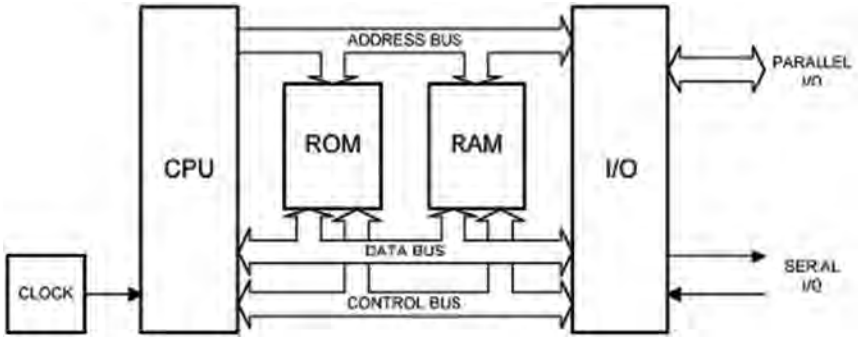
- سخت‌افزار میکروکنترلر را تعریف کند.
- کاربرد میکروکنترلر را نام ببرد.
- درگاه های میکروکنترلر را شناسایی کند.
- بخش‌های مختلف میکروکنترلر را راه‌اندازی کند.
- موج PWM را با فرکانس مشخص تولید کند.

مقدمه‌ای بر میکروکنترلرها

با پیشرفت علم و تکنولوژی در زمینه الکترونیک، مدارهای مجتمع با عنوان میکروپروسسور(ریز پردازنده‌ها) وارد عرصه الکترونیک شدند. در سال‌های قبل از ۱۹۷۱ میلادی، طراحان سیستم‌های الکترونیک، سیستم‌های مورد نظر خود را با استفاده از مدارهای مجتمع دیجیتال و یا آنالوگ و یا ترکیبی از آنها طراحی می‌کردند. در طراحی‌های حرفه‌ای و عمده، طرح این سیستم‌ها جهت ساخت، به شرکت‌های سازنده مدارات مجتمع ارائه می‌شد تا طراحی و ساخته شوند.

پس از این سال‌ها شرکت‌هایی از قبیل Zilog اقدام به ساخت میکروپروسسورها به صورت عمومی نمودند تا کاربران بتوانند با در اختیار داشتن آن طرح‌های مورد نظر خود را به صورت مستقل طراحی و پیاده‌سازی کنند.

این شرکت اولین میکروپروسسور ساخت خود را با نام Z۸۰ روانه بازار نمود. یکی از معایب طراحی با پروسسورها، نیاز به ادوات و بخش‌های جانبی متعدد، به صورت مجزا از هم بود. در شکل زیر دیاگرام بلوکی یک سیستم میکروپروسسوری را مشاهده می‌کنید.



شکل ۱ـ۲ـ دیاگرام بلوکی یک سیستم میکروپروسسوری

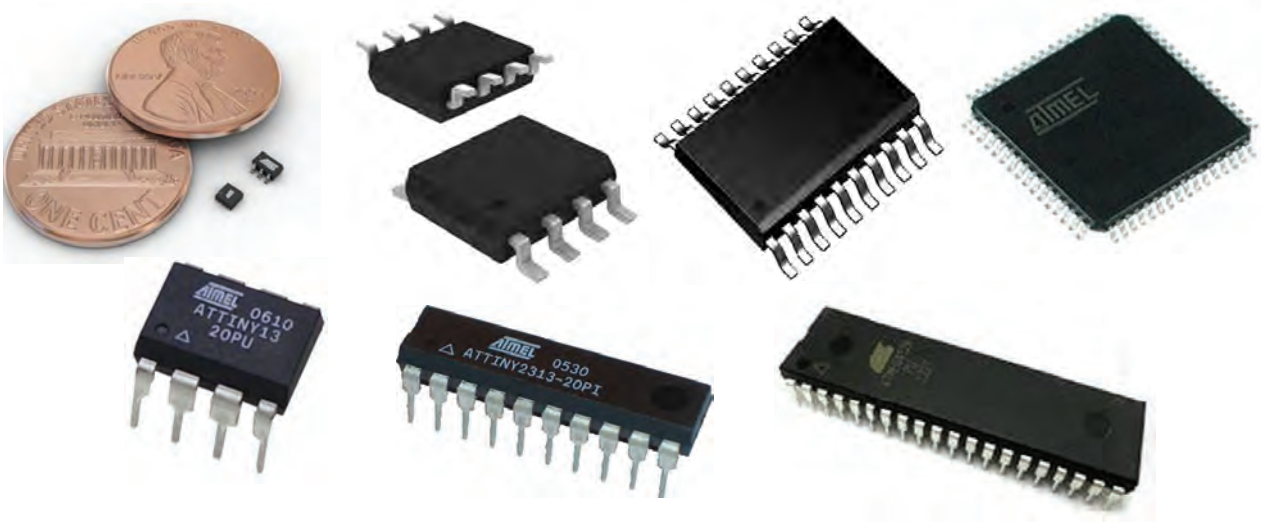
سخت‌افزار میکروکنترلر

در سال ۱۹۸۱ با عرضه محصول جدیدی از شرکت Intel با عنوان میکروکنترلر ، مشکلات مربوط به سیستم‌های پروسسوری مرتفع گردید، شرکت Intel این محصول خود را تحت عنوان خانواده ۸۰۵۱ معرفی و روانه بازار نمود.

این محصول تمام بخش‌های مورد نیاز به عنوان یک رایانه کوچک (از قبیل حافظه موقت، حافظه دائم، درگاه‌های ورودی و خروجی، تایمر/کانتر، مبدل آنالوگ به دیجیتال، ارتباط سریال و ...) را در خود جای داده بود و از این رو در بسیاری موارد از سیستم‌های میکروکنترلری با عنوان "سیستم‌های نهفته" نام برده می‌شود.

بر خلاف میکروکنترلرهای امروزی که در انواع مختلفی از نظر برنامه‌ریزی موجود هستند، میکروکنترلرهای اولیه فقط یک بار قابل برنامه‌ریزی بودند (One Time Program) که این ویژگی، با کلمه اختصاری OTP شناخته می‌شود.

در شکل زیر چند نمونه میکروکنترلر AVR، محصول شرکت اتمل (Attmel) را مشاهده می‌کنید.

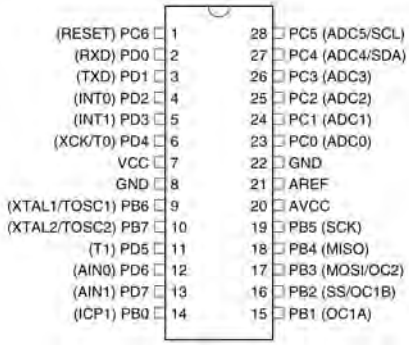


شکل ۲ـ۲ـ شکل ظاهری چند نمونه میکروکنترلر AVR، محصول شرکت اتمل (Attmel)

پایه‌های خروجی برخی میکروکنترلرهای AVR

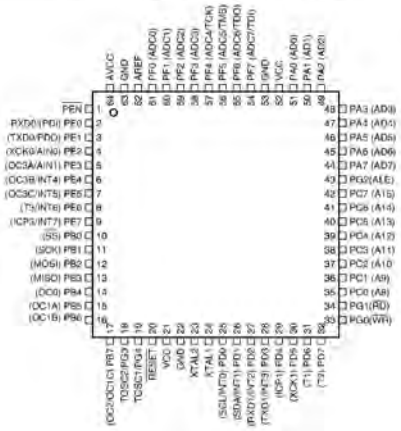
ATmega8

PDIP



ATiny26

ATmega64,ATmega128



ATmega16, ATmega32

PDIP

تعاریف

تعریف بیت (Bit)

کوچک‌ترین واحد حافظه است که می‌تواند یک عدد باینری را در خود ذخیره نماید. بنابراین یک بیت می‌تواند عدد ۰ یا ۱را در درون خود ذخیره نماید.

تعریف بایت (Byte)

هر ۸ بیت که کنار هم قرار گرفته باشند تشکیل یک بایت را می‌دهند. با توجه به اینکه هر بیت می‌تواند ۰ یا ۱ باشد، بنابراین یک بایت می‌تواند از (00000000) هشت عدد ۰ تا (11111111) یعنی هشت عدد ۱ به خود مقدار بگیرد که معادل آن در مبنای دهدهی از ۰ تا ۲۵۵ است.

برای سنجش میزان حافظه‌هایی که متشکل از تعداد زیادی بایت می‌باشند، از "پیشوند"هایی قبل از نام بایت استفاده می‌گردد (کیلو، مگا، گیگا نمونه‌هایی از این پیشوندها می‌باشند). جدول زیر، نام‌ها ، حروف اختصاری و مقادیر پیشوندهای بایت را نشان می‌دهد.

جدول ۱ـ۲ـ نام‌ها، حروف اختصاری و مقادیر پیشوندهای بایت

نام	مخفف	اندازه
Kilo	K	2^{10}=۱,۰۲۴
Mega	M	2^{20}=۱,۰۴۸,۵۷۶
Giga	G	2^{30}=۱,۰۷۳,۷۴۱,۸۲۴
Tera	T	2^{40}=۱,۰۹۹,۵۱۱,۶۲۷,۷۷۶
Peta	P	2^{50}=۱,۱۲۵,۸۹۹,۹۰۶,۸۴۲,۶۲۴
Exa	E	2^{60}=۱,۱۵۲,۹۲۱,۵۰۴,۶۰۶,۸۴۶,۹۷۶
Zetta	Z	2^{70}=۱,۱۸۰,۵۹۱,۶۲۰,۷۱۷,۴۱۱,۳۰۳,۴۲۴
Yotta	Y	2^{80}=۱,۲۰۸,۹۲۵,۸۱۹,۶۱۴,۶۲۹,۱۷۴,۷۰۶,۱۷۶

اجزای داخلی یک میکروکنترلر

تعریف درگاه (PORT)

درگاه یا پورت، وظیفه ارتباط میکروکامپیوتر با دستگاه‌های جانبی را بر عهده دارد. با توجه به اینکه این درگاه‌ها قابلیت ورودی یا خروجی بودن در حالت دیجیتال را دارا هستند. به‌طور کلی اگر در میکروکنترلرها این پورت‌ها در حالت دیجیتال پیکربندی شوند به آنها "ورودی / خروجی دیجیتال" یا همان (Digital I / O) گفته می‌شود.

تعریف گذرگاه (BUS)

وظیفه انتقال سیگنال دیجیتال را برعهده دارد و در عمل چیزی جز مجموعه سیم‌های کنار هم قرار گرفته نیست. در هر میکروکنترلر ۳ نوع گذرگاه وجود دارد که وظیفهٔ انتقال داده، آدرس و سیگنال‌های کنترلی را بر عهده دارد.

انواع گذرگاه

گذرگاه داده (Data Bus)

جهت نقل و انتقال داده‌ها بین ماژول‌های میکرو به‌کار می‌رود و در میکروکنترلرها اصطلاحاً "عرض گذرگاه" حداقل ۸ بیت است. هرچه تعداد خطوط گذرگاه بیشتر باشد سرعت انتقال داده‌ها بیشتر و توان پردازش میکروکنترلر بیشتر خواهد بود.

گذرگاه آدرس (Address Bus)

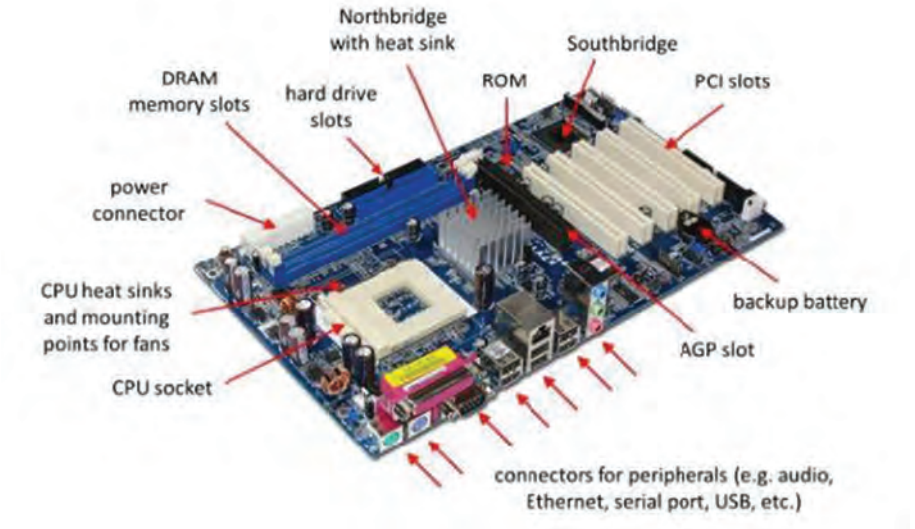
در سیستم‌های رایانه‌ای اعم از میکروکامپیوترها و میکروکنترلرها، شناسایی، انتخاب و ارتباط با هر وسیله(حافظه‌ها، I/O ها) توسط CPU مستلزم اشاره به آدرس آن وسیله می‌باشد، این آدرس باید برای هر کدام از بخش‌های جانبی میکروکنترلر منحصر به فرد باشد. هرچه تعداد خطوط آدرس بیشتر باشد، میکروکنترلر می‌تواند تعداد مکان‌های بیشتری را آدرس‌دهی کند، در نتیجه می‌تواند به تعداد بیشتری I/O و مقدار بیشتری حافظه و ... دسترسی داشته باشد.

گذرگاه کنترل (Control Bus)

شامل خطوط کنترلی دستگاه‌های موجود مثل Write، Read و غیره است. هرچه تعداد خطوط کنترلی بیشتر باشد، میکروکنترلر امکانات کنترلی بیشتری در اختیار برنامه‌نویس قرار می‌دهد.

تعریف میکروکامپیوتر

یک سیستم میکروکامپیوتر حداقل شامل پردازنده (CPU)، حافظه موقت (RAM)، حافظه دائمی (ROM)و (درگاه‌های ورودی/خروجی) می‌باشد که به‌وسیله گذرگاه‌ها به هم ارتباط دارند و همه مجموعه روی یک برد اصلی (Main board) قرار می‌گیرند. تمام کامپیوترهای خانگی امروزی مانند PCها و لپ‌تاپ‌ها از این نوع هستند که علاوه بر واحدهای فوق قطعات و واحدهای دیگری نیز به آنها اضافه شده است مانند شکل زیر:

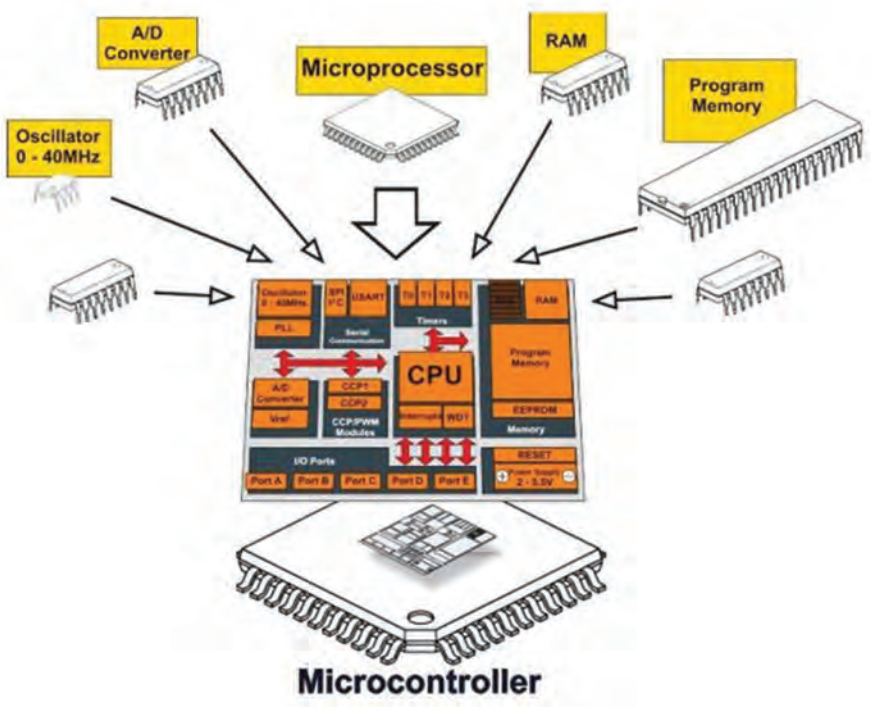


شکل نمایش امکانات و متعلقات یک سیستم میکروکامپیوتری

نحوه ساخت یک مدار مجتمع میکروکنترلر (Microcontroller IC)

تعریف میکروکنترلر (Microcontroller)

هنگامی که قطعات سازنده یک میکروکامپیوتر در یک تراشه و در کنار هم قرار گیرند، یک میکروکنترلر به‌وجود می‌آید. در واقع میکروکنترلر یک آی سی شامل یک CPU به همراه مقدار مشخصی از حافظه‌های RAM و ROM و یا Flash، پورت‌های ورودی/ خروجی و همچنین واحدهای جانبی دیگری نظیر تایمر، رابط سریال، مبدل آنالوگ به دیجیتال و ... می‌باشد. به عبارت دیگر میکروکنترلر یک تراشه الکترونیکی قابل برنامه‌ریزی است که استفاده از آن باعث افزایش سرعت و کارایی مدار و در مقابل، کاهش حجم و هزینه مدار می‌گردد.



معرفی برخی رجیسترهای کاربردی AVR

رجیستر Data Direction

این رجیستر همان‌طور که از نامش مشخص است رجیستر جهت داده پورت بوده و تعیین می‌کند که پورت ورودی است یا خروجی. بدین صورت که اگر روی هرکدام از بیت‌های این رجیستر یک نوشته شود پین متناظر آن پورت خروجی بوده و در غیر این صورت ورودی می‌باشد.

به عنوان مثال اجرای دستور DDRA = 0b10111101; در برنامه، وضعیت ورودی/خروجی پورت A را به صورت زیر پیکربندی می‌نماید.

DDRA	۷	۶	۵	۴	۳	۲	۱	۰
نام بیت	۱	۰	۱	۱	۱	۱	۰	۱
جهت داده	خروجی	ورودی	خروجی	خروجی	خروجی	خروجی	ورودی	خروجی

جدول ۲ـ۲ـ مثال دستور DDRx برای پورت A

رجیستر PORTx

عملکرد این رجیستر بستگی به جهت دادهٔ پورت دارد. در صورتی که به عنوان خروجی پیکربندی شده باشد، صفر یا یک منطقی در این رجیستر ، سطح منطقی پایه خروجی معادل آن را تعیین می‌کند و در صورتی که ورودی باشد با یک کردن هر بیت، مقاومت Pull_up داخلی مربوط به آن پین فعال می‌شود.

به عنوان نمونه در ادامه مثال قبل، با اجرای دستور PORTA = 0b11010100; وضعیت پورت به صورت زیر خواهد بود:

شماره بیت	۷	۶	۵	۴	۳	۲	۱	۰
DDRA	۱	۰	۱	۱	۱	۱	۰	۱
PORTA	۱	۱	۰	۱	۰	۱	۰	۰
جهت داده	خروجی با سطح منطقی یک	ورودی با مقاومت Pull_up	خروجی با سطح منطقی صفر	خروجی با سطح منطقی یک	خروجی با سطح منطقی صفر	خروجی با سطح منطقی یک	ورودی بدون مقاومت Pull_up	خروجی با سطح منطقی صفر

جدول ۳ـ۲ـ مثال دستور PORTx برای پورت A

رجیستر PINx

برای خواندن مقدار هر پین باید محتویات این رجیستر خوانده شود. به عنوان مثال چنانچه PORC را قبلاً به صورت ورودی پیکربندی کرده باشیم و مقدار رجیستر PINC برابر 0b11010000 باشد، سطح منطقی اعمال شده به پین به صورت زیر می‌باشد:

PINC	۷	۶	۵	۴	۳	۲	۱	۰
نام بیت	۱	۱	۰	۱	۰	۰	۰	۰
جهت داده	یک	یک	صفر	یک	صفر	صفر	صفر	صفر

جدول ۴ـ۲ـ وضعیت ورودی‌های منطقی اعمال شده به پایه‌های پورت A

فعال‌سازی پایه های میکروکنترلر به عنوان خروجی

در بسیاری از کاربردهای میکروکنترلرها، تمامی محاسبات و زمان‌سنجی‌ها و شمارش وقایع و .... به فعال کردن و یا غیر فعال کردن یکی از خروجی‌ها منتهی می‌شود، از این رو شاید بهتر باشد، اولین تجربه‌های کاری که می‌توان با یک میکروکنترلر انجام داد فعال‌سازی یک یا چند خروجی باشد. در ادامه این بخش، مثال‌هایی از چند کاربرد راه‌اندازی پایه‌های میکروکنترلر به عنوان ورودی/ خروجی دیجیتال را مشاهده خواهید کرد.

روشن کردن یک LED

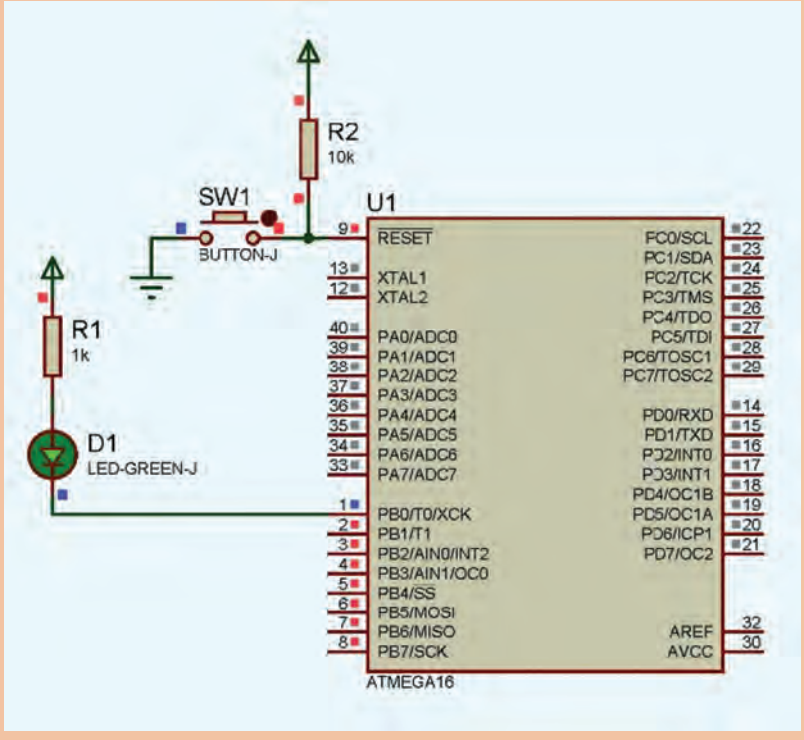
مثال ۲-۱

روشن کردن یک LED

در این مثال، یکی از خروجی‌های میکروکنترلر به عنوان خروجی دیجیتال تعریف شده و آن را بدون در نظر گرفتن هیچ شرط و یا نتایج محاسبات و ...، با یک دستور ساده فعال خواهیم کرد.

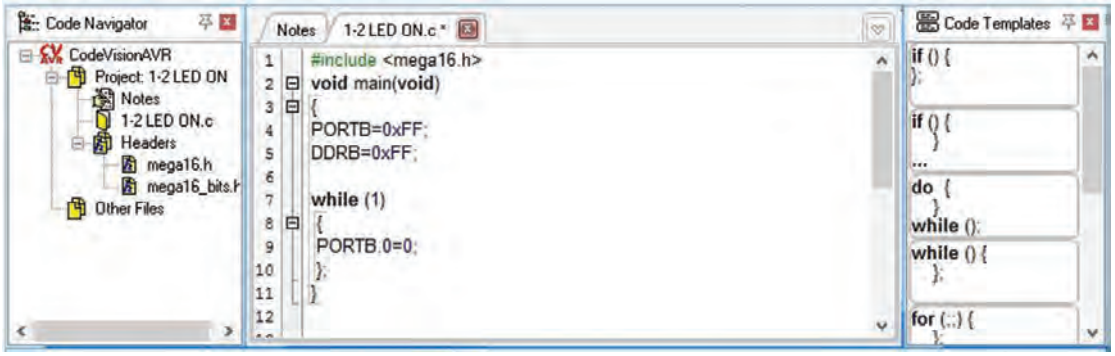
نکته

حالت خروجی، فعال پایین یا همان Active Low در نظر گرفته شده است. در این شیوه راه‌اندازی، ایجاد سطح منطقی صفر بر روی پایه خروجی، سبب فعال شدن قطعه یا وسیله تحت کنترل می‌گردد.



مدار شماتیک برای روشن کردن یک LED با استفاده از یک پایه خروجی میکروکنترلر

با توجه به حالت حافظه قفل شونده (Latch) در میکروکنترلرها، از زمان صدور یک دستور در مورد وضعیت یک بیت و یا یک پایه ورودی/خروجی دیجیتال، تا ارسال دستور جدید به همین بیت و یا پایه، آخرین وضعیت آن ثابت باقی خواهد ماند. اثر این دستورات، تا زمان باقی بودن تغذیه میکروکنترلر ادامه خواهد داشت.

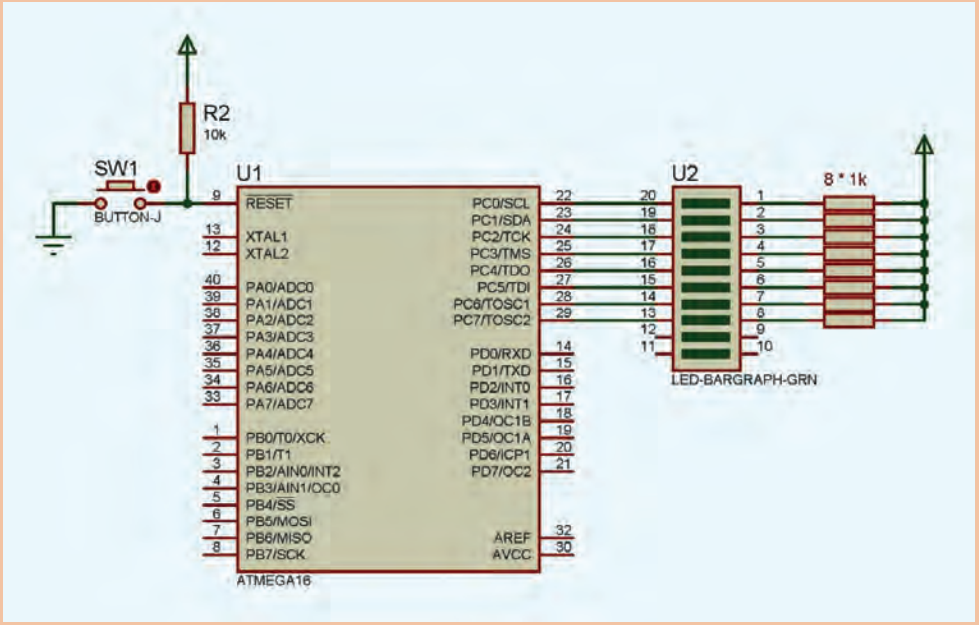


تصویر متن برنامه برای روشن کردن یک LED با استفاده از یک پایه خروجی میکروکنترلر

با توجه به تصویر بالا، برای خاموش شدن LED باید دستور جدیدی جهت این کار به رجیسترهای مربوط به پایه مورد نظر ارسال شود. مسئله قابل توجه این است که این دستور باید به اندازه کافی با دستور قبلی فاصله زمانی داشته باشد، در غیر این صورت چشم انسان قادر به مشاهده روشن شدن LED نبوده و در کسر کوچکی از ثانیه LED روشن شده و با دستور بعدی بلافاصله خاموش می‌شود.

فعالیت ۱

با توجه به شکل زیر:



الف) برنامه‌ای بنویسید که در آن LEDها بر روی ۴ بیت اول از پورت C خاموش و ۴ بیت بعدی آن روشن شده و در همان حالت باقی بمانند.
ب) هنرجویان برنامه‌ای بنویسند که LEDها بر روی پورت C به صورت فعال پایین، یک در میان(کنترل بیتی خروجی) روشن شده و در همان حالت باقی بماند.
ج) عدد AA هگزا دسیمال یا معادل باینری آن (10101010) را در مدار صفحهٔ قبل روی پورت C قرار دهید و نتیجه را مشاهده کنید.
د) عدد ۵۵ هگزا دسیمال یا معادل باینری آن (01010101) را در مدار صفحهٔ قبل روی پورت C قرار دهید و نتیجه را مشاهده کنید.

توجه

با استفاده از کلید Reset، برنامه را چند بار اجرا نموده و از صحت عملکرد برنامه اطمینان حاصل کنید.

کاربرد تأخیر در تغییر وضعیت منطقی پایه‌های خروجی

یکی از روش‌های مرسوم برای برنامه‌ریزی زمانی دستورات در میکروکنترلرها، استفاده از دستورات ایجاد تأخیر در برنامه است. با استفاده از دستورات تأخیر دهنده، میکروکنترلر در وضعیت فعلی باقی مانده و پس از پایان زمان تأخیر، دستور بعدی را اجرا می‌کند.
در نرم افزار CODEVISION در محدوده میلی‌ثانیه از دستور ( )Delay_ms و در محدوده میکروثانیه از دستور ( )Delay_us برای ایجاد تأخیر زمانی استفاده می‌شود. برای استفاده از این دستور در برنامه، لازم است فایل سرآمد Delay.h در ابتدای برنامه فراخوانی شود. در داخل پرانتز، عدد مورد نظر برای تأخیر در برنامه نوشته می‌شود. این عدد می‌تواند از صفر تا ۶۵۵۳۵ مقدار داشته باشد. شکل زیر نحوه کاربرد این دستور را در بین خطوط برنامه نشان می‌دهد.

نکته

در یک برنامه‌نویسی اصولی، استفاده از تأخیرهای طولانی مرسوم نیست.

Notes | delay.c*

```
1  #include <mega16.h>
2  #include <delay.h>
3
4  void main(void)
5  {
6  .
7  .
8  delay_ms(500);
9  .
10 .
11
12 while (1)
13 {
14
15 };
16 }
17
```

نحوه کاربرد دستور Delay در بین خطوط یک برنامه به زبان C

به عنوان مثال، می‌توان برای پر کردن مخازن مایعات، دستور باز شدن شیر برقی را با استفاده از یک پایه خروجی میکروکنترلر صادر نموده و پس از گذشت مدت زمان مشخص، دستور بسته شدن آن را صادر نمود. به یک چنین سیستم کنترلی، "**سیستم کنترل حلقه باز**" گفته می‌شود، در این مثال اگر فشار مایع و یا مقدار اولیه آن در مخزن و ... تغییر کند ممکن است مخزن پر نشود و یا مایع از آن سر ریز کند. (بیان این مثال، صرفاً جنبه آموزشی داشته و به جهت مطرح کردن اشکالات سیستم‌های کنترلی حلقه باز مطرح گردید، به طور معمول، کاربردهایی از این دست، برمبنای "**سیستم کنترل حلقه بسته**" طرح می‌گردند که در ادامه همین بخش به آن پرداخته شده است.

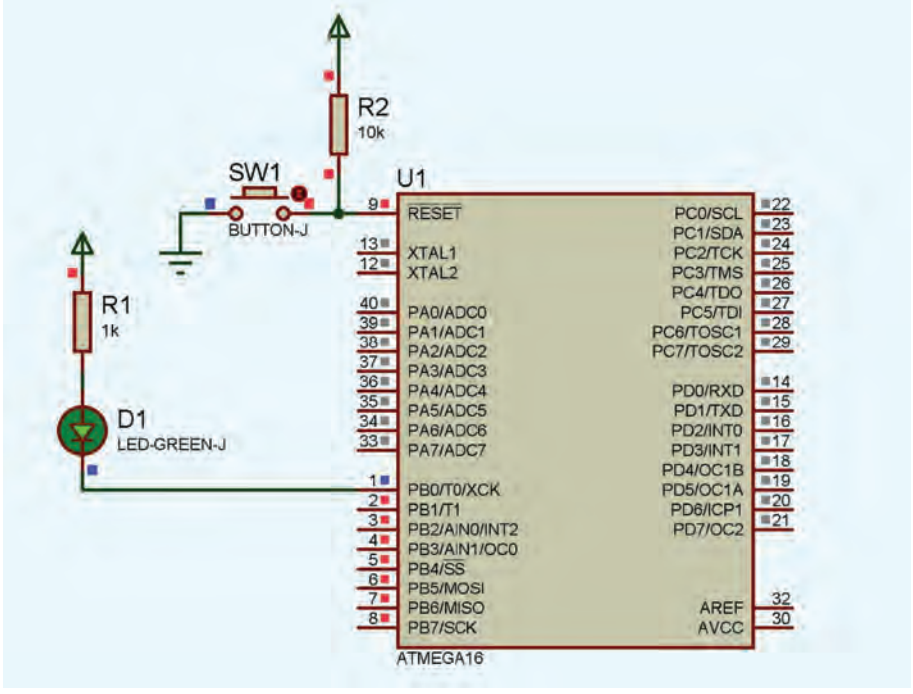
LED با چشمک زن

مثال ۲-۲

ایجاد حالت چشمک زن برای روی یک LED

در یک سیستم میکروکنترلری اگر خواسته باشیم که یک فرایند کنترلی، طی زمان‌بندی خاص (یک یا چند مرحله‌ای) و به صورت تکراری صورت پذیرد، در متن برنامه میکروکنترلر، دستورات برنامه را در حلقه تکرار اصلی برنامه (که معروف به حلقه تکرار بی‌نهایت است) می‌نویسیم.

در این مثال یک پایه خروجی را به طور مداوم و با زمان مساوی، فعال و غیر فعال می‌کنیم. جهت نمایش وضعیت خروجی پایه، یک LED به صورت"Active Low" بر روی این پایه قرار گرفته است. در یک پروژه عملی، با کمک رله‌ها و یا ترانزیستورها و ... می‌توانیم قطعات دیگری مانند انواع موتورهای الکتریکی، لامپ‌ها ، هیترها ، شیرهای برقی، جک‌ها و ... را به همین شیوه راه‌اندازی نماییم.



طرح شماتیک مدار چشمک زن LED با استفاده از یک پایه خروجی میکروکنترلر

Notes | 2-2 Blinking LED.c*

```
#include <mega16.h>
#include <delay.h>

void main(void)
{
PORTB=0xFF;
DDRB=0xFF;

while (1)
 {
 PORTB.0=0;
 delay_ms(500);
 PORTB.0=1;
 delay_ms(500);
 };
}
```

شکل ۳ـ۲ـ تصویر متن برنامه چشمک زن LED با استفاده از یک پایه خروجی میکروکنترلر

فعالیت ۲

الف) هنرجویان با کمک مربی برنامه‌ای بنویسند که یک LED بر روی بیت صفر از پورت B به صورت فعال پایین، روشن شده و پس از ۲ ثانیه روشن ماندن خاموش شده و در همان حالت باقی بماند.

ب) برنامه‌ای بنویسید که پس از گذشت ۵ ثانیه از شروع برنامه، یک LED بر روی PORTB.0 روشن شده و در همان حالت باقی بماند.

ج) در یک میکسر مواد شیمیایی در یک کارخانه تولید رنگ، لازم است؛ شیر برقی ماده A به مدت ۱دقیقه و شیر برقی ماده B به مدت ۳۰ ثانیه باز باشد، پس از گذشت مدت زمان ۲ دقیقه جهت ترکیب مواد، شیر تخلیه میکسر به مدت ۲ دقیقه باز باشد تا مواد ترکیب شده از میکسر خارج شود و دوباره همین روند تکرار شود، برنامه‌ای مناسب برای عملکرد این میکسر رنگ بنویسید.

د) برنامه‌ای مناسب برای کنترل وضعیت چراغ راهنمایی یک چهار راه بنویسید که فقط دارای دو وضعیت سبز و قرمز در هر سمت بوده و پس از گذشت ۳۰ ثانیه وضعیت آن تغییر کند و این کار را به صورت مداوم ادامه دهد.

نکته

به عنوان راهنمایی، به خاطر داشته باشید در برنامه‌هایی که لازم است عمل و یا مجموعه عملیاتی، فقط یک بار انجام شده و در پایان به یک حالت ماندگار برسد، کافیست در متن برنامه، دستورات کنترلی، بلافاصله بعد از تابع اصلی نوشته شود.

پاسخ میکروکنترلر به محرک‌های خارجی

کنترل وضعیت پایه‌های خروجی با توجه به وضعیت پایه‌های ورودی

یکی دیگر از کاربردهای میکروکنترلرها در ارتباط با ورودی/خروجی‌های دیجیتال، اعمال کنترلی با استفاده از دادن فرمان‌های خارجی به میکروکنترلرهاست.

زمانی که دکمه خاموش/روشن یک نمایشگر رایانه و یا تلفن هوشمند و یا دکمه کنترل سرعت بر روی سیستم تهویه اتوماتیک را می‌فشاریم، در حقیقت با تغییر وضعیت پایه‌های ورودی، سیستم کنترلی را وادار به بررسی و تصمیم‌گیری براساس فرمان‌های داده شده می‌نماییم.

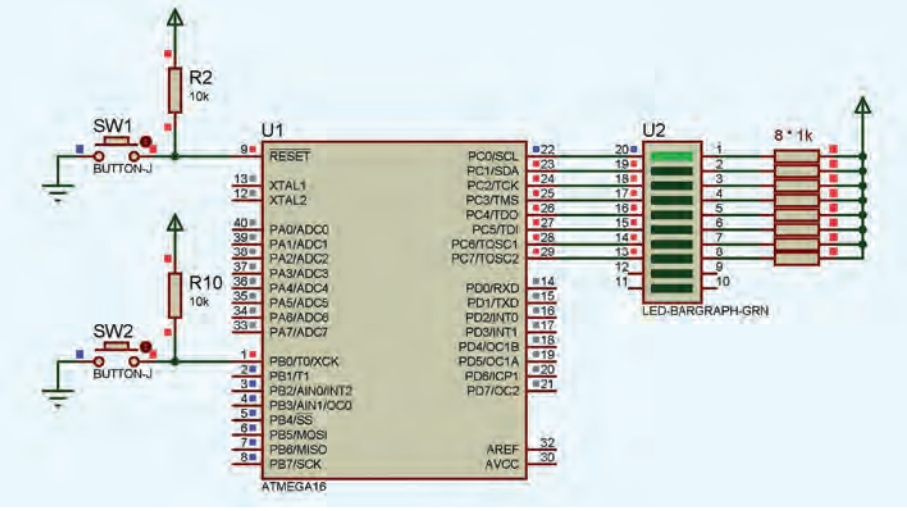
مثال ۲-۳

فعال شدن یک پایه خروجی با فعال نمودن یک پایه ورودی

برای انجام این کار، در ابتدای برنامه، باید یکی از پایه‌ها را به صورت ورودی و پایه دیگری را به صورت خروجی تعریف نموده و سپس با بررسی دایمی وضعیت پایه ورودی، وضعیت منطقی پایه خروجی را کنترل نماییم.

یکی از روش‌های مرسوم برای بررسی وضعیت منطقی ورودی استفاده از دستور شرطی if است. عملکرد کلی این دستور برای بررسی وضعیت یک پایه ورودی به زبان ساده به صورت زیر است:

((اگر پایه ورودی در وضعیت ورودی مورد نظر بود (صفر یا یک منطقی) آنگاه، وضعیت پایه خروجی را به حالت مورد نظر (صفر یا یک منطقی) تغییر بده.))



شکل ۴-۲- طرح شماتیک مدار کنترل یک پایه خروجی با فعال نمودن یک پایه ورودی

Notes | 3-2 Controled Output By an Input.c

```
#include <mega16.h>

void main(void)
{
PORTB=0x00;
DDRB=0xFE;
PORTC=0xFF;
DDRC=0xFF;


while (1)
 {
  if (PINB.0==0)
  {
  PORTC.0=0;
  };

 };
}
```

شکل ۵-۲- تصویر متن برنامه کنترل یک پایه خروجی با فعال نمودن یک پایه ورودی

با مشاهده عملکرد مدار در می‌یابیم که پس از یک بار فشردن کلید، پایه خروجی فعال شده و تا زمانی که دستور غیر فعال شدن را به آن نداده باشیم، در وضعیت فعال باقی خواهد ماند، در برنامه مثال ۲ـ۳ دستوری برای غیر فعال کردن پایه خروجی در نظر گرفته نشده است.
به طور کلی اینکه با فعال شدن یک ورودی، چه اتفاقی باید بیفتد و یا اینکه اثر آن اتفاق تا چه زمانی برقرار باشد و یا اینکه با غیر فعال شدن مجدد آن چه اتفاقی بیفتد (یا تغییری در روند برنامه رخ ندهد) کاملاً به نیازهای پروژه و تعریف پروژه مورد نظر بستگی دارد.

مفهوم الگوریتم

نکته

در علم ریاضی و برخی علوم دیگر مانند علوم رایانه و برنامه‌نویسی، به روشی که در آن به طور متوالی، یک فرایند پایه برای حل یک مسئله، تکرار شود اصطلاحاً "**الگوریتم**" گفته می‌شود. مسایل مطرح شده در مثال‌ها و فعالیت‌های این بخش در حقیقت بخشی از الگوریتم برنامه مورد نظر هستند. در یک الگوریتم ممکن است مسایل به صورت کلی و یا با تمام جزییات مطرح شوند، طرح یک الگوریتم به صورت کلی، عمدتاً از سوی سفارش دهنده طرح (کاربر) و تکمیل الگوریتم با تمام جزئیات، اصولاً جزء توانایی‌های برنامه‌نویس می‌باشد.

عملکرد سنسورهای سطح مایعات

نکته

یکی از روش‌های کنترل سطح مایعات در صنعت، استفاده از انواع سوئیچ‌های سطح (Level Switch) است. سوئیچ‌های سطح ممکن است ویژه نصب بر روی دیواره افقی یا عمودی مخازن و یا به صورت شناور(Float Switch) طراحی شده باشند. همچنین این سوئیچ‌ها ممکن است دارای خروجی نرمال باز و یا نرمال بسته باشند و گاهی نیز دارای دو سیستم سوئیچ برای کنترل ۲ سطح باشند.

این تجهیزات در حقیقت رسیدن مایع به یک سطح مشخص را تشخیص می‌دهند، علت به کار بردن لفظ "سنسور" (Sensor) در مورد این نوع سوئیچ‌ها نیز همین موضوع است.
شکل‌های زیر برخی از انواع این تجهیزات را نمایش می‌دهد.

برخی از انواع سوئیچ‌های سطح که بر روی دیواره افقی مخزن قابل نصب هستند

سنسورهای سطح شناور کابلی

فعالیت ۳

الف) برای کنترل یک شیر برقی روشویی، برنامه‌ای بنویسید که با تحریک (Active Low) یک پایه ورودی میکروکنترلر توسط سنسور مادون قرمز، یک پایه خروجی فعال شده (Active Low) و شیر برقی را باز نگه دارد و تا ۲ ثانیه بعد از قطع شدن تحریک ورودی فعال بماند.
ب) یک مخزن ویژه نگهداری مایع، دارای دو سنسور، یکی در قسمت بالای مخزن (High Level) و یکی در قسمت پایین مخزن (Low Level) و نیز دو شیر برقی در بالا و پایین مخزن است. با توجه به شکل و نکات زیر، برنامه‌ای بنویسید که در صورت فعال شدن سنسور سطح پایین مخزن، شیر برقی توسط یک پایه خروجی میکروکنترلر باز شده و با فعال شدن سنسور سطح بالای مخزن، پایه خروجی غیر فعال و شیر برقی بسته شود.

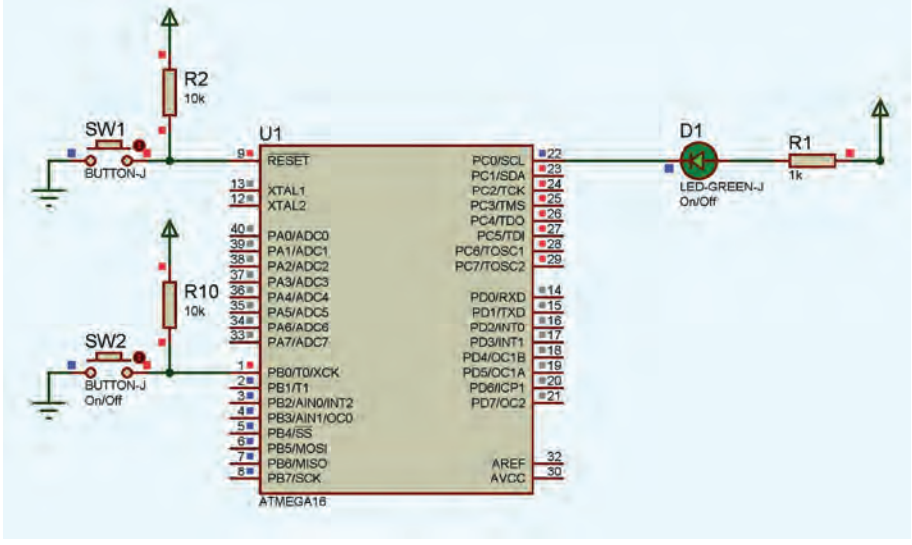
عملکرد معکوس‌کنندگی ۱ بیت یا یک متغیر چند بیتی

تغییر وضعیت (Toggle) خروجی با تحریک یک پایه ورودی

مثال ۲-۴

کنترل خاموش و روشن یک وسیله با استفاده از عملکرد (Toggle)

کلمه "Toggle" به معنی تغییر وضعیت، در حقیقت نوعی عملکرد معکوس‌کننده دارد، به طور مثال با اعمال این عملگر بر روی متغیری با ارزش "0" منطقی، ارزش متغیر برابر با "1" منطقی خواهد شد و به همین ترتیب با هر بار عملکرد این عملگر، متغیر مورد نظر به عبارتی NOT می‌شود.
این ویژگی در برنامه‌نویسی، معمولاً جهت سوئیچ بین دو وضعیت از پیش تعیین شده کاربرد دارد و از آن برای عملکردهایی مانند شروع/ توقف، روشن/ خاموش، چپ‌گرد/ راست‌گرد، بالاشمار/ پایین‌شمار و ... استفاده می‌شود . در زبان C این عملکرد با استفاده از کاراکتر (~) قابل دسترسی است.



تصویر شماتیک مدار کنترل روشن/ خاموش با استفاده از دستور Toggle

تصویر متن برنامه مدار کنترل روشن/ خاموش با استفاده از دستور Toggle

فعالیت ۴

الف) با اضافه کردن دستور while به برنامه مثال ۲ـ۴، آن را بازنویسی نموده و با حفظ عملکرد صحیح برنامه، مقدار تأخیر در (خط هفدهم) برنامه را به حداقل رسانیده و یا آن را حذف نمایید.

ب) تغییرات مناسب در مدار مثال ۲ـ۴، برای کنترل چپ‌گرد/ راست‌گرد یک موتور DC اعمال نموده (استفاده از یک رله با دو جفت کنتاکت باز/ بسته و ترانزیستور راه‌انداز رله ...) و با استفاده از برنامه همین مثال (بدون تغییر) عملکرد مدار جدید را مشاهده نمایید.

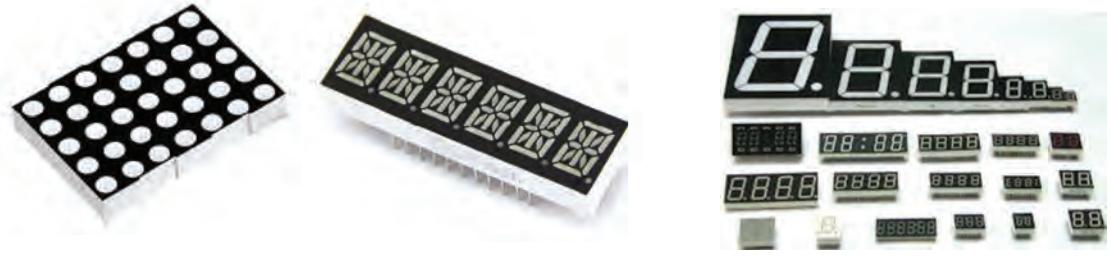
ج) برنامه‌ای بنویسید که در آن با فعال شدن یک پایه ورودی، اگر پس از گذشت ۲ ثانیه هنوز پایه ورودی فعال باقی مانده بود، یک پایه خروجی با هر بار تکرار این عمل، بین حالت فعال و غیر فعال تغییر وضعیت دهد (عملکرد خاموش و روشن تلفن همراه و ...).

نمایش حروف و اعداد بر روی نمایشگر هفت قطعه‌ای

نمایش اعداد بر روی نمایشگر هفت قطعه‌ای (7seg)

در بسیاری از پروژه‌های میکروکنترلری، نمایش اطلاعات ورودی، وضعیت و یا نتایج محاسبات، کمک بسیار بزرگی به روند کنترل و بررسی چگونگی عملکرد سیستم می‌نماید، یکی از مناسب‌ترین نمایشگرها، نمایشگرهای LED هستند. در شکل زیر برخی از انواع این نمایشگرها را مشاهده می‌کنید.

همان‌طور که در شکل مشاهده می‌کنید برخی از این نمایشگرها متشکل از چند کاراکتر ویژه ثابت هستند و برخی دیگر دارای کاراکترهایی هستند که قابلیت نمایش تعدادی حروف و اعداد را دارا هستند، هر یک از این کاراکترها به طور معمول دارای هفت و یا چهارده قطعه است که هر یک با استفاده از یک LED روشن می‌شوند. ترتیب خاموش و یا روشن بودن این LEDها، کاراکتر مورد نظر را به وجود می‌آورد.

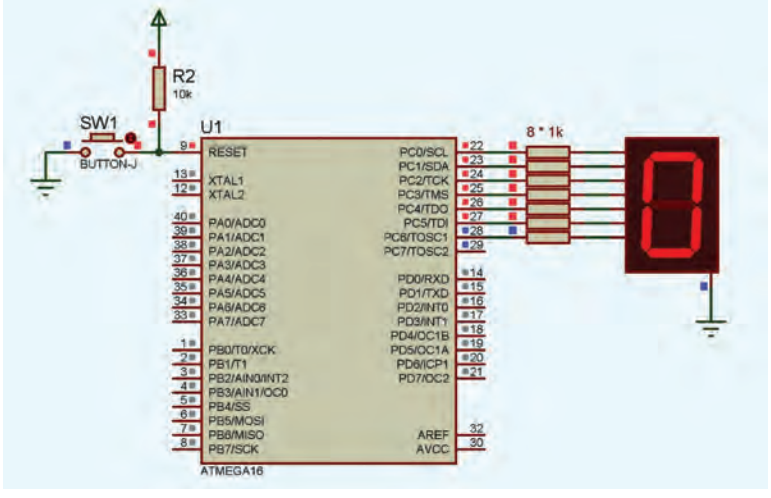


شکل ۱۶ـ۲ـ برخی از انواع نمایشگرهای LED

نمایشگر هفت قطعه‌ای

شمارنده تک رقمی ۰ تا ۹ با استفاده از نمایشگر هفت قطعه‌ای

با وجود پیشرفت در زمینه تولید انواع نمایشگرهای رنگی با کیفیت بالا، مزیت ارزان بودن، وضوح بالا و قابلیت تشخیص از فاصله دور نسبت به سایر نمایشگرها، استفاده از نمایشگرهای هفت قطعه‌ای هنوز هم در بین طراحان و تولیدکنندگان رواج دارد، شکل زیر مربوط به یک شمارنده تک رقمی از ۰ تا ۹ با فاصله زمانی یک ثانیه است.



مدار شماتیک شمارنده تک رقمی با نمایشگر هفت قطعه‌ای

```
#include <mega16.h>
#include <delay.h>

flash char digits[16]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,
0x7D,0x07,0x7F,0x6F,0x77,0x7C,0x39,0x5E,0x79,0x71};

unsigned char i;

void main(void)
{
DDRC = 0xFF;
PORTC = 0x00;

while (1)
  {
   for (i=0;i<10;i++)
   {
   PORTC = digits[i];
   delay_ms(1000);
   };
  };
}
```

شکل ۷ـ۲ـ تصویر برنامه شمارنده تک رقمی با نمایشگر هفت قطعه‌ای

فعالیت ۵

الف) با استفاده از نمایشگر هفت قطعه‌ای دو رقمی ، یک شمارنده از ۰ تا ۹۹ طراحی نموده و عملکرد آن را مشاهده کنید.
ب) با استفاده از سه نمایشگر هفت قطعه‌ای ، کلمه "ALI" را نمایش دهید.

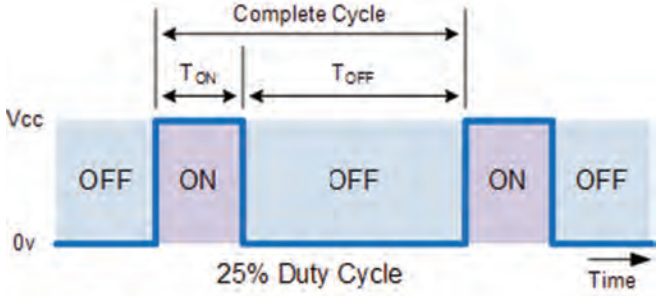
کنترل توان الکتریکی DC بار، با استفاده از تکنیک PWM

یکی از قابلیت‌های میکروکنترلرها کنترل توان DC به وسیله سیگنال PWM است (کنترل دیجیتال). این تکنیک به سبب کاهش توان تلفاتی در قطعات قدرت سیستم‌های کنترل توان DC دارای اهمیت ویژه است. در این شیوه از کنترل سوئیچ قدرت که نقش تأمین توان الکتریکی بار را دارد، اصولاً در حالت قطع و اشباع کار می‌کند همین مسئله موجب کاهش قابل توجه توان تلفاتی، خصوصاً در بارهای پر مصرفی مانند موتورهای الکتریکی، سلونوئیدها، هیترها، لامپ‌های التهابی و ... می‌گردد.
در یک سیستم کنترل توان DC با تکنیک PWM، انرژی منتقل شده به بار، با تغییر نسبت زمان روشن بودن به زمان خاموشی قطعه قدرت کنترل می‌شود. این نسبت با نام دوره چرخه کار (Duty Cycle) یا به اختصار D.C شناخته می‌شود.

تعریف Duty Cycle

در یک موج مربعی، نسبت زمان روشن بودن (t_{on}) به کل زمان تناوب ($T= t_{on} + t_{off}$) با عنوان Duty Cycle نام برده می‌شود.

$$D.C = \frac{t_{on}}{T}$$



شکل ۸ـ۲ـ نمایش زمان روشن، زمان خاموش یک موج مربعی

نکته

* حاصل این کسر همیشه به طور ذاتی باید عددی بین صفر تا یک باشد.
** در مواقعی که لازم باشد این فاکتور بر حسب درصد بیان شود مقدار آن را در عدد ۱۰۰ ضرب می‌کنند که در این صورت (٪) D.C مورد نظر است،

$$D.C(\%)=\frac{t_{on}}{T}\times ۱۰۰=\frac{OCRX}{TCNTX\ top\ value-TCNTX}\times ۱۰۰$$

*** مقدار TCNTX top value در حالت‌های مختلف و رجیسترهای مختلف می‌تواند ۲۵۶، ۵۱۲، ۱۰۲۴ (یا بسته به نوع میکروکنترلر، مقادیر دیگری) باشد.

موج مربعی با مدولاسیون پهنای پالس (PWM)

تولید موج PWM با فرکانس مشخص

عملکرد واحد PWM در میکروکنترلرها به این صورت است که با شروع شمارش رجیستر timerX، خروجی این واحد به حالت (۱) منطقی می‌رود و تا زمانی که مقدار رجیستر تایمر (TCNTX) با مقدار مقایسه OCRX برابر شود در وضعیت (۱) منطقی باقی می‌ماند، به محض برابری مقدار تایمر با رجیستر مقایسه‌ای OCRX (رجیستر مقدار PWM مورد نظر)، خروجی این واحد تغییر وضعیت داده و به حالت (۰) منطقی می‌رود و تا زمان سرریز رجیستر TCNTX (رجیستر کنترلی تایمر/ کانتر مورد نظر) در حالت (۰) منطقی باقی می‌ماند.

زمان (۱) بودن پالس خروجی (t_{on})، برابر ضرب مقدار OCRX در طول زمان یک کلاک پالس ورودی به واحد تایمر است و کل زمان تناوب یک پالس (T)، برابر حداکثر ظرفیت عددی رجیستر TCNTX، در طول زمان یک کلاک پالس ورودی واحد تایمر است.

$$\begin{cases} t_{on}=OCRX\times Clock\ Pulse\ time \\ T=TCNTX\ top\ value\times Clock\ Pulse\ time \end{cases}$$

$$T=\frac{۱}{f_{out}}$$

$$f_{out}=\frac{۱}{TCNTX\ top\ value\times Clock\ Pulse\ time}=\frac{Clock\ Pulse\ Frequency}{TCNTX\ top\ value}$$

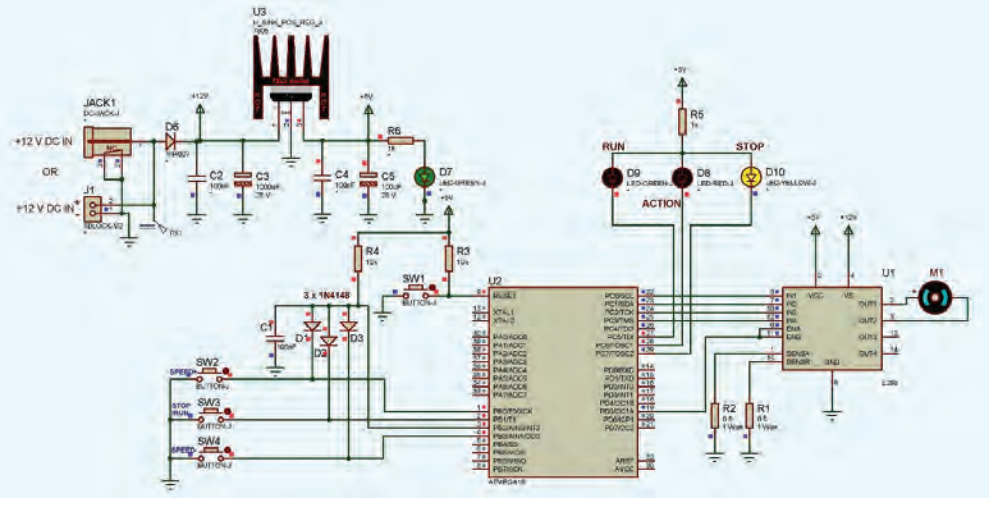
با استفاده از مطالب بالا (و رابطه عکس فرکانس با زمان) می‌توان گفت فرکانس کلاک ورودی ، بر مقدار حداکثر رجیستر تایمر مربوطه تقسیم می‌شود. چون به همین تعداد پالس نیاز است تا رجیستر از صفر به حداکثر رسیده و به ازای هر بار سر ریز این رجیستر، یک پالس در خروجی ظاهر می‌شود.
اگر وضعیت خروجی را در حالت Non Inverting قرار داده باشیم پالس خروجی مستقیماً در پایه خروجی OCRX در میکروکنترلر ظاهر خواهد شد و در صورت انتخاب وضعیت Inverting قرینه این پالس در خروجی (مکمل D.C) ظاهر خواهد شد .

به طور مثال اگر پالسی با درصد D.C معادل ۲۵ درصد در واحد PWM ساخته شده باشد، با انتخاب وضعیت Inverting در خروجی یک پالس ۷۵ درصد در خروجی ظاهر می‌شود.

مثال ۲-۶

کنترل دور موتور DC با استفاده از تکنیک PWM

در این مثال با استفاده از میکروکنترلر ATMEGA۱۶ که دارای ۴ عدد کنترلر PWM سخت‌افزاری است، سرعت گردش یک موتور DC را تحت کنترل قرار می‌دهیم، از آنجا که توان مورد نیاز برای راه‌اندازی یک موتور DC، بیشتر از توان خروجی پورت‌های میکروکنترلر می‌باشد، برای راه‌اندازی موتور از آی‌سی راه‌انداز L۲۹۸ به جهت تقویت ولتاژ و جریان سیگنال PWM ساخته شده توسط میکروکنترلر استفاده شده است.



مدار شماتیک کنترل دور موتور DC با استفاده از تکنیک PWM

Notes | 6-2 MEGA16 PWM SPeed Controler for DC Motors.c

```
#include <mega16.h>
#include <delay.h>

bit RUN;
unsigned char SPEED=0;

interrupt [EXT_INT2] void ext_int2_isr(void)
{
//////////SPEED CONTROL////////
 if(PINB.0==0 && SPEED<10)
 {
 SPEED++;
 }
 if(PINB.3==0 && SPEED>0)
 {
 SPEED--;
 }
/////////////RUN////////////////
 if(PINB.1==0)
 {
 RUN=~RUN;
 }
//////////////////////////////////
 PORTC.6=0;
 delay_ms(200);
 PORTC.6=1;
}
```

تصویر متن برنامه کنترل دور موتور DC با استفاده از تکنیک PWM بخش اول

Notes | 6-2 MEGA16 PWM SPeed Controler for DC Motors.c

```
28
29
30  void main(void)
31  {
32
33  PORTB=0x0F;
34  DDRB=0x00;
35
36  PORTC=0xE0;
37  DDRC=0xFF;
38
39  PORTD=0x00;
40  DDRD=0x20;
41
42  TCCR1A=0x81;
43  TCCR1B=0x0B;
44  TCNT1H=0x00;
45  TCNT1L=0x00;
46  ICR1H=0x00;
47  ICR1L=0x00;
48  OCR1AH=0x00;
49  OCR1AL=0x00;
50  OCR1BH=0x00;
51  OCR1BL=0x00;
52
53  GICR|=0x20;
54  MCUCR=0x00;
```

شکل۹ـ۲ـ تصویر متن برنامه کنترل دور موتور DC با استفاده از تکنیک PWM بخش دوم

Notes | 6-2 MEGA16 PWM SPeed Controler for DC Motors.c

```
52
53  GICR|=0x20;
54  MCUCR=0x00;
55  MCUCSR=0x00;
56  GIFR=0x20;
57
58  #asm("sei")
59
60  while (1)
61        {
62        if (RUN==0)
63         {
64         PORTC=0x60;
65         PORTD.5=0;
66         OCR1AL=0x00;
67         }
68        else
69         {
70         PORTC=0xC1;
71         OCR1AL=SPEED*25;
72         };
73        };
74  }
75
76
```

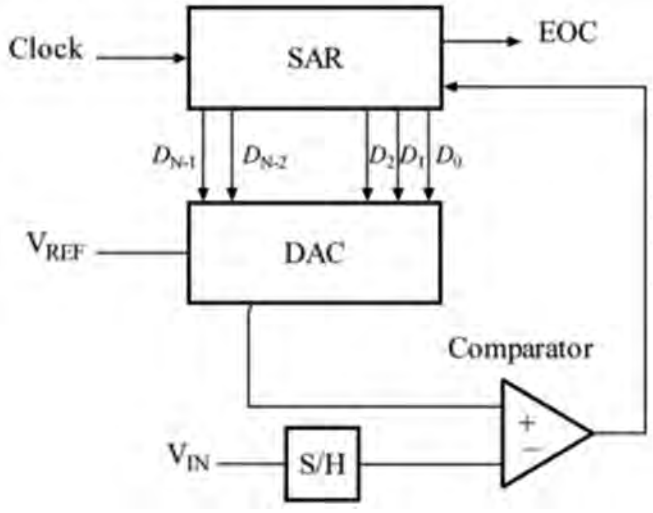
شکل۱۰ـ۲ـ تصویر متن برنامه کنترل دور موتور DC با استفاده از تکنیک PWM بخش سوم

مبدل آنالوگ به دیجیتال (ADC)

عمده روش‌هایی که برای تبدیل آنالوگ به دیجیتال وجود دارند عبارت‌اند از: تبدیل آنی یا Flash، روش تقریب‌های متوالی یا Successive Approximation Register (SAR)، مبدل‌های Delta-Encoded Adc or Counter Ramp، Pipelined Adc ،Delta Sigma ،Digital Ramp و غیره که از میان میکروکنترلرهای AVR از روش تقریب‌های متوالی استفاده می‌کنند.

دیاگرام بلوکی روش تقریب‌های متوالی

بلوک دیاگرام ساده شدهٔ این مبدل به صورت زیر است:



برخی از مشخصات ADC در میکروکنترلر AVR

- ± ۲ LSB Absolute Accuracy
- ۶۵-۲۶۰ µs Conversion Time
- Up to ۱۵ kSPS at Maximum Resolution
- ۸ Multiplexed Single Ended Input Channels
- ۷ Differential Input Channels
- ۲ Differential Input Channels with Optional Gain of ۱۰x and ۲۰۰x
- ۰-VCC ADC Input Voltage Range
- Selectable ۲.۵۶V ADC Reference Voltage
- Free Running or Single Conversion Mode
- ADC Start Conversion by Auto Triggering on Interrupt Sources
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

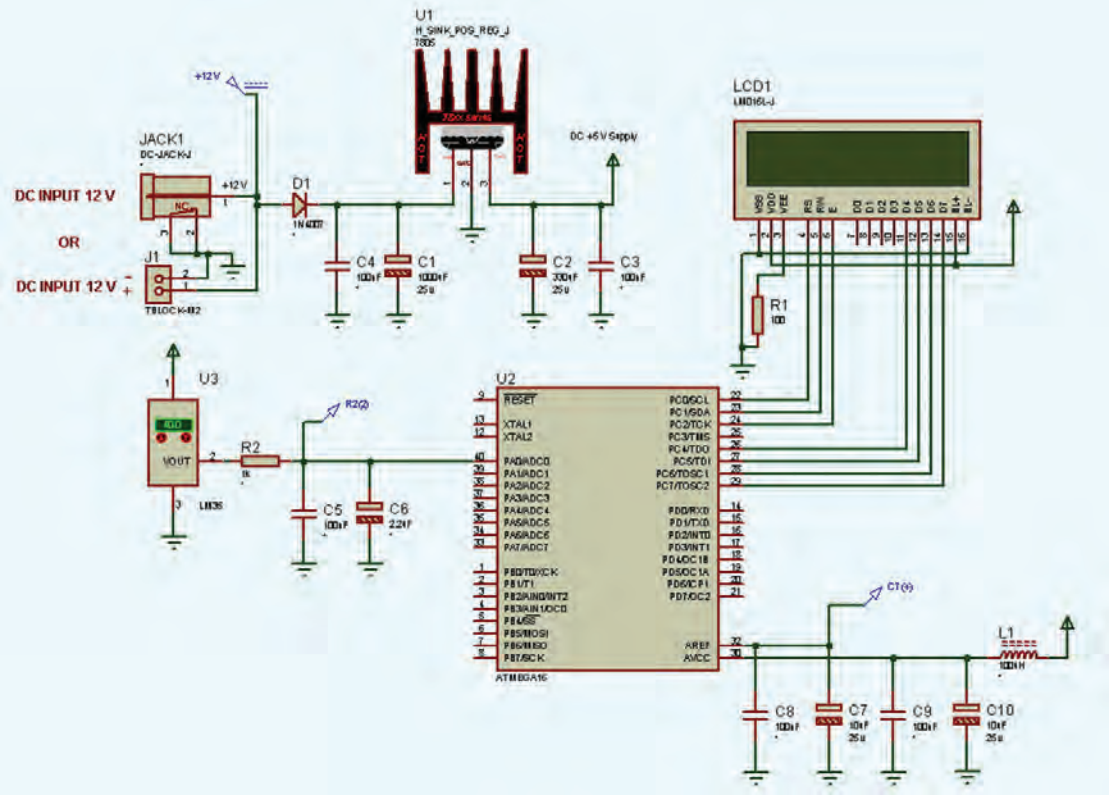
پایه‌های ورودی مبدل آنالوگ به دیجیتال

پین‌های ورودی ADC عملکرد دوم PORAT می‌باشند که به صورت مالتی پلکس شده به ADC اعمال می‌شوند. ولتاژ ورودی بین صفر تا ولتاژ مرجع بوده و ولتاژ مرجع از سه منبع AVCC، پین AREF و ولتاژ داخلی ۲.۵۶ ولت قابل تأمین می‌باشد. جهت کاهش نویز مؤثر بر روی واحد ADC، تغذیهٔ آن به صورت جداگانه از پین AVCC تأمین می‌شود. ولتاژ این پین نباید بیشتر از ۰.۳ ولت با VCC تفاوت داشته باشد. در صورتی که از VCC به عنوان AVCC استفاده می‌شود، می‌توان به‌وسیلهٔ یک فیلتر LC ، این پایه را به VCC متصل نمود.

فیلتر کاهش نویز (داخلی)

مدارات داخلی میکروکنترلر با ایجاد نویز باعث کاهش دقت مقدار خوانده شده توسط ADC می‌شوند، برای بهبود این مشکل می‌توان در زمان تبدیل میکروکنترلر را به یکی از Mode‌های کم‌توان Idle یا ADC Noise Reduction برده تا عملیات تبدیل بعد از خاموش شدن CPU انجام شود.

مدار زیر، تصویر شماتیک برای مبدل ADC به صورت استاندارد را نمایش می‌دهد، در این مدار ولتاژ خروجی حاصل از یک سنسور LM۳۵ به کانال صفر ADC متصل شده و مقدار قرائت شده پس از تبدیل به واحد اصلی (درجه سانتیگراد) بر روی نمایشگر LCD نمایش داده شده است.



متن برنامه نمایش دمای سنسور LM۳۵ بر روی LCD

```
#include <mega16.h>
#include <delay.h>
#include <stdlib.h>
#include <stdio.h>
#define ADC_VREF_TYPE 0xE0
#asm
   .equ __lcd_port=0x15;
#endasm
#include <lcd.h>
#include <stdio.h>

unsigned char i[30];
int temp=0;
int c=0;

unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

void main(void) {

lcd_init(16);

ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x82;

 while (1){

        lcd_gotoxy(0,0);

        for (c=0;c<10;c++)
        {
        temp=temp+read_adc(0);
        }
        temp=temp/10;
        sprintf(i,"temp=%d \xdf\C",temp);
        lcd_puts(i);
        delay_ms(1000);
 }
}
```

فعالیت ۶

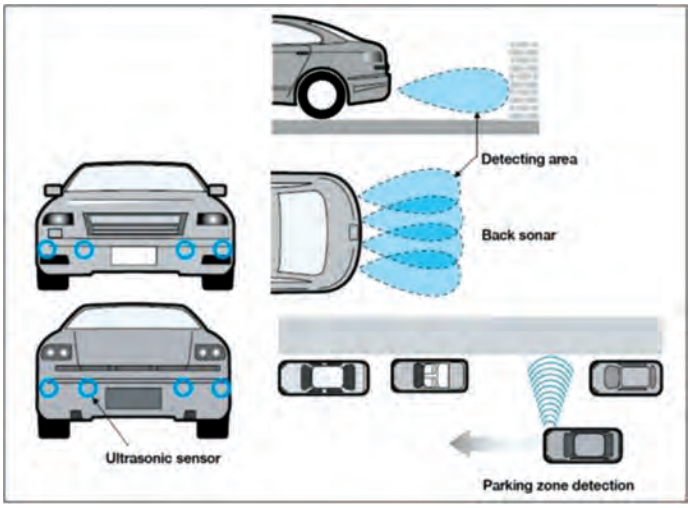
الف) با افزودن دستورات شرطی به متن برنامه بالا، آن را به نحوی بازنویسی نمایید که با رسیدن دما به عددی که طی نوشتن برنامه تعیین می‌کنید، PORTB.۰ست (فعال بالا یا "۱" منطقی) شود.

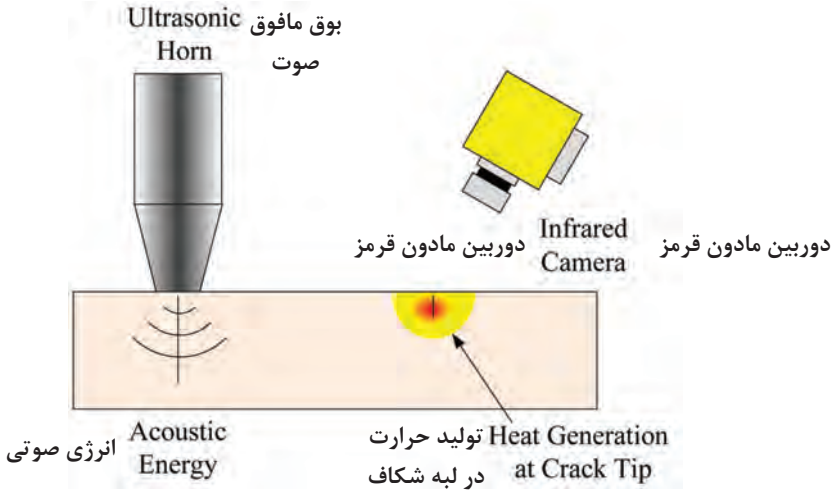
ب) به عنوان یک برنامه کاربردی برای تهویه کلی یک سوله پرورش دام، برنامه‌ای بنویسید که دمای یک سنسور LM۳۵ را قرائت نموده، با کاهش دما از میزان تعیین شده در برنامه PORTB.۰ فعال و با کمک رله یک هیتر صنعتی را فعال نماید و با افزایش آن PORTB.۱ با کمک یک رله یک فن صنعتی را راه‌اندازی نماید.

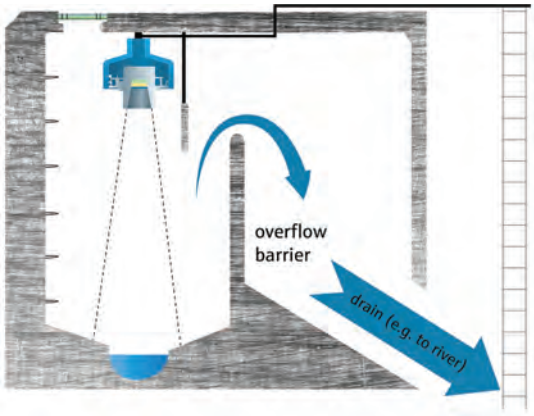
ماژول فاصله‌سنج SRF۰۵ (ارتباط I۲C)

حسگرهای فراصوت مشابه رادار یا ردیاب صوتی، تشخیص ویژگی‌های هدف از طریق تحلیل بازتاب امواج رادیویی یا صوتی می‌باشد. حسگرهای فراصوت امواج صوتی با فرکانس بالا ایجاد می‌کنند. این حسگرها با محاسبه زمان بین فرستادن سیگنال و گرفتن بازتاب، فاصله جسم را محاسبه می‌کنند. از این فناوری می‌توان در اندازه‌گیری فاصله بین ماژول با اجسام مقابل آن، ارتفاع و حجم مایع در مخازن، اندازه‌گیری قد اشخاص و سنسور دنده عقب خودرو و کشف جای پارک خالی برای خودروهای هوشمند استفاده کرد.

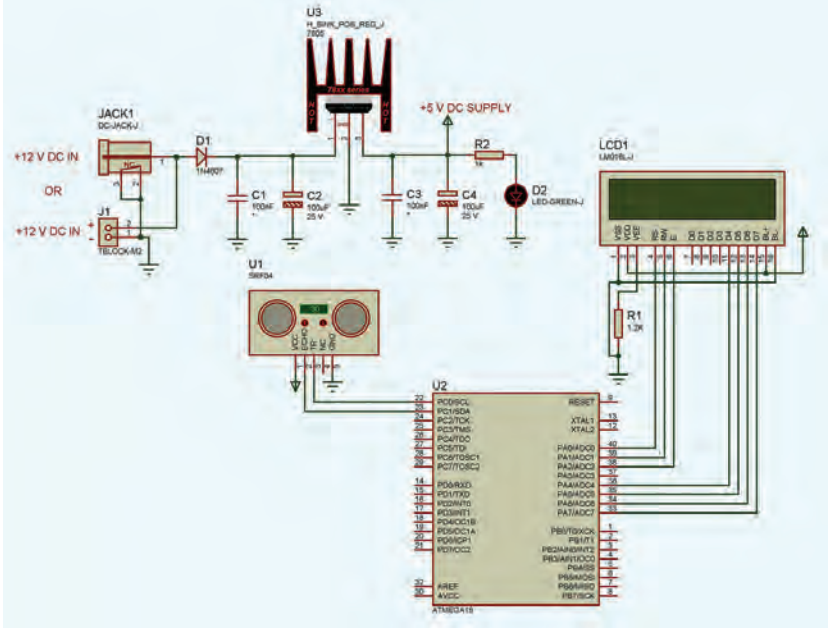
در تصاویر زیر برخی از کاربردهای ماژول‌های فاصله‌سنج با مدارات توسعه یافته را مشاهده می‌کنید.







مدار زیر نمونه‌ای از مدار راه‌اندازی ماژول SRF05 به کمک میکروکنترلر ATMEGA16 به منظور سنجش فاصله و نمایش آن بر روی نمایشگر کریستال مایع می‌باشد.



متن برنامه نمونه جهت راه‌اندازی ماژول فاصله‌سنج و نمایش بر روی LCD

```
#include <mega16.h>
#include <delay.h>
#include <stdlib.h>

#include <alcd.h>

int OVF0_Count;
float d = 0.0;
char str[20];

interrupt [TIM0_OVF] void timer0_ovf_isr(void){TCNT0=0;OVF0_Count++;}


void main(void)
{

PORTC=0x00;
DDRC=0x01;

TCCR0=0x02;
TCNT0=0x00;
OCR0=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x01;

lcd_init(16);
```

ادامه متن برنامه

```
// Global enable interrupts
#asm("sei")

while (1)
      {
      //ÇÑÓÇá íå çáó Êå çíå ÊÑí
       PORTC.0=1;delay_us(20);PORTC.0=0;

       while(PINC.1==0){};
       OVF0_Count=0;TCNT0=0;TCCR0=0x02;
       while(PINC.1 == 1){};
       TCCR0=0x00;

       if(OVF0_Count*256.0+TCNT0>30000.0){lcd_clear();lcd_gotoxy(0,0);lcd_puts("chizi nist. o_O");}
       else
       {
           d=0;d=(OVF0_Count*256.0+TCNT0)*0.1716;
           lcd_clear();
           lcd_gotoxy(0,0);lcd_puts("  X=");
           lcd_gotoxy(5,0);ftoa(d,2,str);lcd_puts(str);
           lcd_gotoxy(13,0);lcd_puts("cm");
           lcd_gotoxy(0,1);lcd_puts(" SFR04&05 SENSOR");
           delay_ms(200);
       }

      }
}
```

از کاربردهای دیگر امواج فراصوت، ردیاب‌های صوتی (Sonar)، بخورها (Humidifier)، فراوانگاری (سونوگرافی فراصوت) تصویربرداری ۳ یا ۴ بعدی از درون اشیاء، سنسورهای تشخیص حرکت، دزدگیرها و آزمایشات غیر مخرب (Nondestructive testing)، تشخیص نشت گاز و یا تشخیص محل شکستگی یا ترک و ... می‌شود. ماژول التراسونیک SRF05 در واقع نمونه تکامل یافته SRF04 است و با هدف افزایش انعطاف‌پذیری و افزایش رنج از ۳ متر به ۴ متر و کاهش قیمت طراحی شده است. عملکرد جدید (Mode در صورت اتصال به پین زمین) این ماژول امکان استفاده از تنها یک پین برای تریگر و اکو به‌طور هم‌زمان می‌دهد. در نتیجه در تعداد پین مصرفی از میکروکنترلر صرفه‌جویی می‌شود. وقتی پین Mode بدون اتصال رها می‌شود، ماژول SRF05 با استفاده از پین‌های جداگانه تریگر و اکو همانند SRF04 عمل می‌کند.

فعالیت ۷

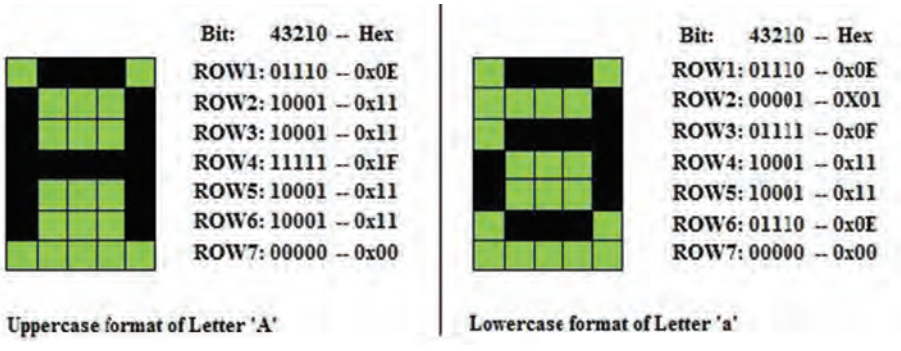
الف) با استفاده از ماژول SRF۰۵، مداری بسازید که فاصله سپر عقب اتومبیل با اشیاء یا اشخاص را بر روی نمایشگر نشان دهد و در صورت تشخیص فاصله کمتر از ۶۰ سانتی‌متر به کمک یک LED یا بازر، هشدار مناسب ایجاد نماید.

الف) در مثال ۲ـ۶ تغییرات لازم جهت افزودن امکان کنترل چپ‌گرد/ راست‌گرد را در مدار شماتیک و متن برنامه اعمال نموده و نتیجه را مشاهده کنید.
ب) با توجه به مثال ۲ـ۶ مداری جهت کنترل دو عدد موتور DC طراحی و برنامه مناسب با آن را بنویسید.

انواع نمایشگرهای کریستال مایع

نمایشگر کریستال مایع یا LCD

LCD مخفف عبارت "Liquid Crystal Display" به معنای **صفحه نمایش کریستال مایع** است. کریستال‌های مایع موادی هستند که ظاهر مایع دارند، اما مولکول‌های آنها آرایش خاصی نسبت به یکدیگر دارند. به همین دلیل کریستال مایع خصوصیاتی شبیه به مایع و جامد داشته و به همین دلیل با چنین اسم متناقضی خوانده می‌شوند.
برای ساخت LCD دو شیشه پلاروید را با ۹۰ درجه اختلاف نسبت به یکدیگر قرار می‌دهند و یک کریستال مایع بین آنها می‌گذارند. وقتی کریستال به **جریان برق** وصل نباشد؛ نور از قطبشگر اول می‌گذرد و وارد کریستال مایع می‌شود، جهتش ۹۰ درجه تغییر کرده و به همین دلیل از قطبشگر دوم هم عبور کرده و به چشم می‌رسد. اما وقتی که جریان به کریستال وصل باشد، نور دیگر چرخشی نخواهد داشت و نمی‌تواند از کریستال دوم عبور کند.
در بسیاری از موارد، سیستم‌های کنترلی، با یک نمایشگر تک رنگ (Monochrome)، اطلاعات کافی برای ارتباط با کاربر را در اختیار او قرار می‌دهند.
ساده‌ترین انواع این نمایشگرها، نمونه‌هایی معروف به نمایشگر کریستال مایع کاراکتری (CHARACTER LCD یا Alphanumeric LCD) می‌باشد. این نمایشگر، به طور معمول دارای تعدادی کاراکتر ماتریسی (به طور معمول ۵×۷ یا ۵×۸) در یک یا چند سطر است که هر کدام از این کاراکترها می‌توانند با روشن و خاموش نگه داشتن خانه‌های خود حرف یا شکلی را نمایش دهند.



یک کاراکتر ماتریسی، مربوط به نمایشگرهای کریستال مایع کاراکتری

در بازار الکترونیک، نمایشگرهایی به صورت 4×40, 4×20, 2×20, 2×16, 1×16, 2×8, 1×8 داریم، به طور مثال، منظور از 1×16 در حقیقت شانزده کاراکتر در یک سطر است. شکل زیر کاراکترهای یک نمایشگر را نشان می‌دهد که سطر بالا تمام پیکسل‌ها روشن و سطر پایین تماماً خاموش هستند.

کاراکترهای یک نمایشگر کریستال مایع کاراکتری ۲×۱۶

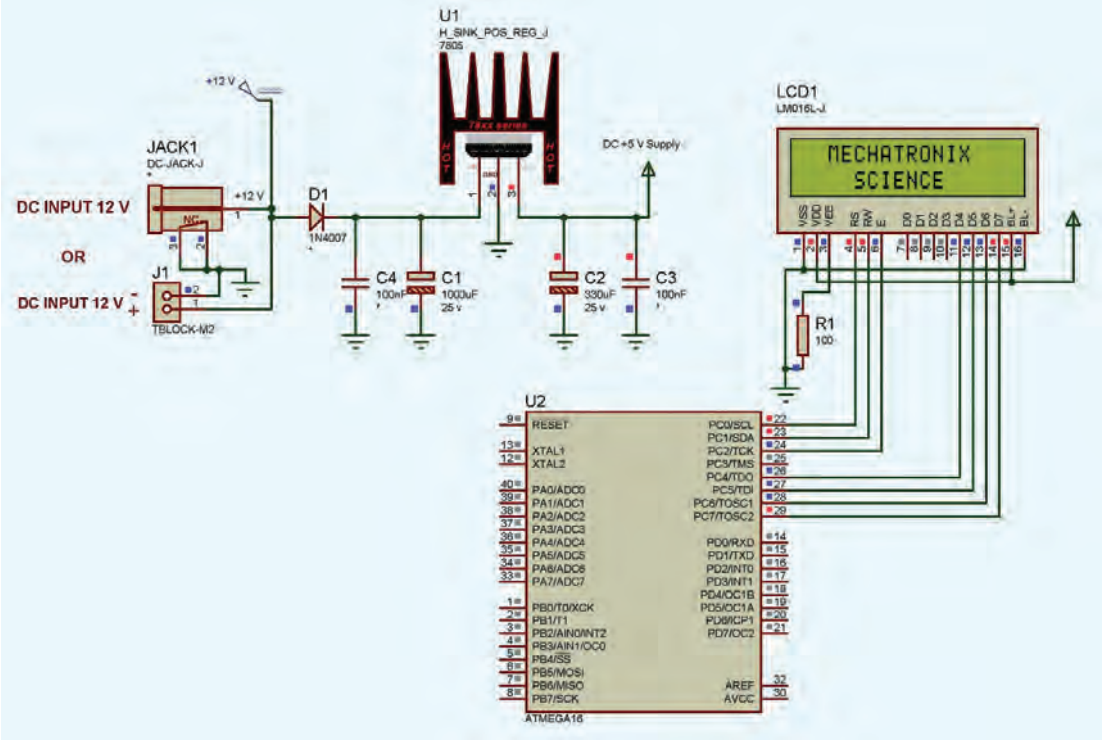
در شکل زیر برخی از انواع نمایشگر کریستال مایع کاراکتری را مشاهده می‌کنید.

برخی از انواع نمایشگر کریستال مایع کاراکتری

مثال ۲-۷

نمایش متن بر روی Alphanumeric LCD

در این مثال یک متن ثابت بر روی نمایشگر 2×16 نمایش داده شده است. برای نمایش محتوای یک متغیر لازم است ابتدا آن را تبدیل به یک رشته کد نماییم.



مدار شماتیک نمایشگر Alphanumeric LCD 16×2

فعالیت ۸

الف) با توجه به مثال ۲ـ۷، برنامه‌ای بنویسید که در آن کلمه "SCIENCE" در خط دوم متحرک به سمت چپ و راست خود شود.

ب) با استفاده از مدار بالا، برنامه یک شمارنده ۰ تا ۹۹ بر روی نمایشگر LCD بنویسید.

ج) تصویر یک باتری قلمی به صورت ایستاده، یک دوشاخه برق و یک فن را بر روی نمایشگر LCD نمایش دهید.

ارزشیابی پایانی پودمان دوم

۱ با توجه به پیشرفت تکنولوژی و امکان ساخت میکروکنترلرهایی با قابلیت هزاران بار برنامه‌ریزی مجدد، به نظر شما علت استفاده از میکروکنترلرها به صورت OTP خصوصاً در تولید انبوه تجهیزات میکروکنترلری چه عواملی می‌تواند باشد؟

۲ تفاوت اصلی نوشتن دستورات یک برنامه، بلافاصله بعد از تابع اصلی (main) با نوشتن دستورات برنامه در حلقه تکرار بی‌نهایت (while)، در برنامه چیست ؟

۳ در صورت منطقی و مجاز بودن استفاده از تأخیرهای طولانی، با استفاده از دستور ( )delay_ms ، حداکثر به چه مقدار تأخیر زمانی می‌توان دسترسی داشت؟ با استفاده از دستور ( )delay_us چطور؟

۴ در یک برنامه‌نویسی اصولی به ویژه در صنعت، وضعیت فعال ورودی‌ها و خروجی‌ها با منطق Active Low طراحی و ساخته می‌شوند، این موضوع تا حدی حائز اهمیت است که گاهی به قیمت افزایش خطوط برنامه و یا افزایش برخی قطعات جانبی، رعایت می‌شود. مزایای این شیوه طراحی را بنویسید.

۵ در برنامه مثال ۲ـ۴ اگر دستور delay_ms(۵۰۰) در خط هفدهم، حذف شود چه اتفاقی می‌افتد، این مسئله تحت عنوان چه نامی در برنامه‌نویسی شناخته می‌شود؟

۶ مزیت مشترک نمودن تمامی آندها و یا کاتدها در نمایشگرهای هفت قطعه‌ای چیست؟

۷ جمعی از هنرجویان، کاربرد و مزایای آی سی MAX 6955 را با استفاده از برگه اطلاعات آن، در قالب یک تحقیق گروهی، بررسی نموده و به صورت کنفرانس در کلاس ارائه نمایند.

۸ علت کاهش قابل توجه تلفات قطعات قدرت، در سیستم‌های کنترل PWM را مختصر شرح دهید.