آزمایش ۱: ارتباط با پورتها

۱–۱ مقدمه

در این آزمایش هدف، آشنایی با پورتهای میکروکنترلر ATMEGA32 و راه‌اندازی LED و نمایشگرهای Seven-Segment می‌باشد.

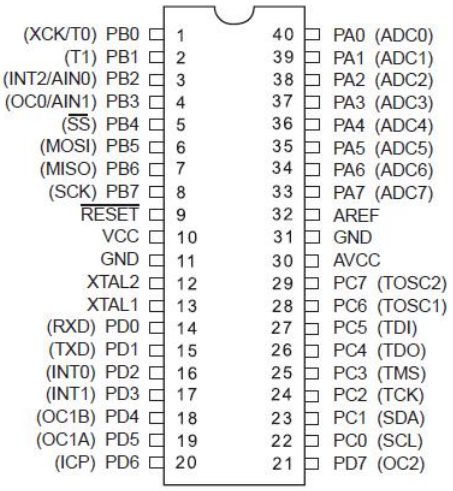
تمامی میکروکنترلرها برای ارتباط با محیط بیرون و سیستمهای جانبی، از بلوک‌های ورودی و خروجی که با نام پورت(Port) شناخته می‌شوند، استفاده می‌نمایند.

MicroLearn.ir

تعداد پورت های برخی از تراشه های AVR

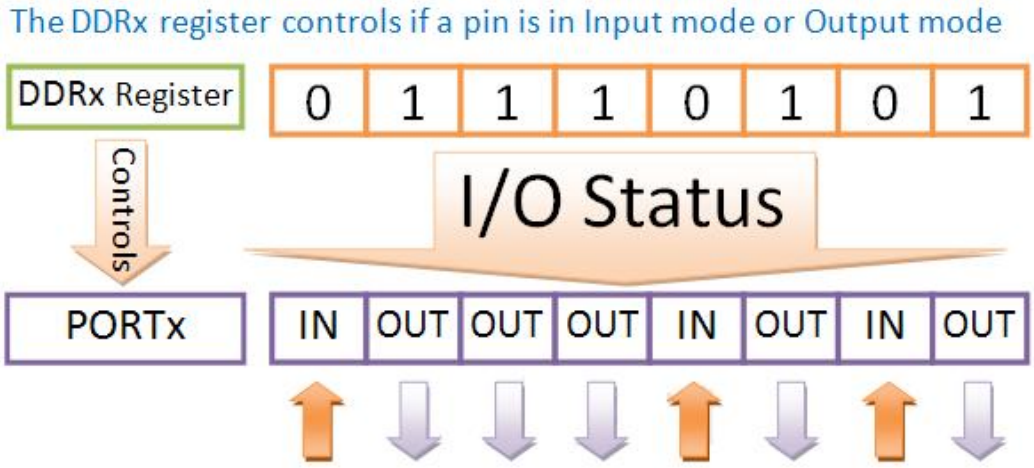
پایه ها	۸ پایه	۲۸ پایه	۴۰ پایه	۶۴ پایه	۱۰۰ پایه
تراشه	Tiny25/45/85	ATmega8/48/88	ATmega16/32	ATmega64/128	ATmega1280
PORT A	✗	✗	✓	✓	✓
PORT B	۶ بیت	✓	✓	✓	✓
PORT C	✗	۷ بیت	✓	✓	✓
PORT D	✗	✓	✓	✓	✓
PORT E	✗	✗	✗	✓	✓
PORT F	✗	✗	✗	✓	✓
PORT G	✗	✗	✗	۵ بیت	۶ بیت
PORT H	✗	✗	✗	✗	✓
PORT J	✗	✗	✗	✗	✓
PORT K	✗	✗	✗	✗	✓
PORT L	✗	✗	✗	✗	✓

همانظور که در شکل زیر مشاهده میکنید تراشه ATMEGA32 دارای ۴ پورت با نام‌های A، B، C و D می‌باشد. هرکدام از این پورتها ۸ بیتی بوده و لذا مجموعا ۳۲پین از پین‌های تراشه به این پورتها اختصاص یافته است.



هرکدام از این پورتهای چهارگانه دارای سه رجیستر اختصاصی با نامهای PORTX، PINX و DDRX می‌باشند. از آنجاییکه هریک از پورتها ۸ بیت طول دارند، لذا هرکدام از این رجیسترها نیز ۸ بیتی می‌باشند.

۱) رجیستر DDRX (**Data Direction Register**) برای تعیین نوع ورودی یا خروجی بودن پورت استفاده می‌شود. در پارامتر "DDRX" حرف X به نام پورت اشاره می‌کند و می‌تواند یکی از حروف A، B، C یا D باشد. بیتهای رجیستر مربوط به پایه متناظر از پورت مربوطه می‌باشند. با صفر قرار دادن هر بیت، پین متناظر از نوع ورودی تعریف شده و با یک قرار دادن آن، از نوع خروجی تعریف خواهد شد.

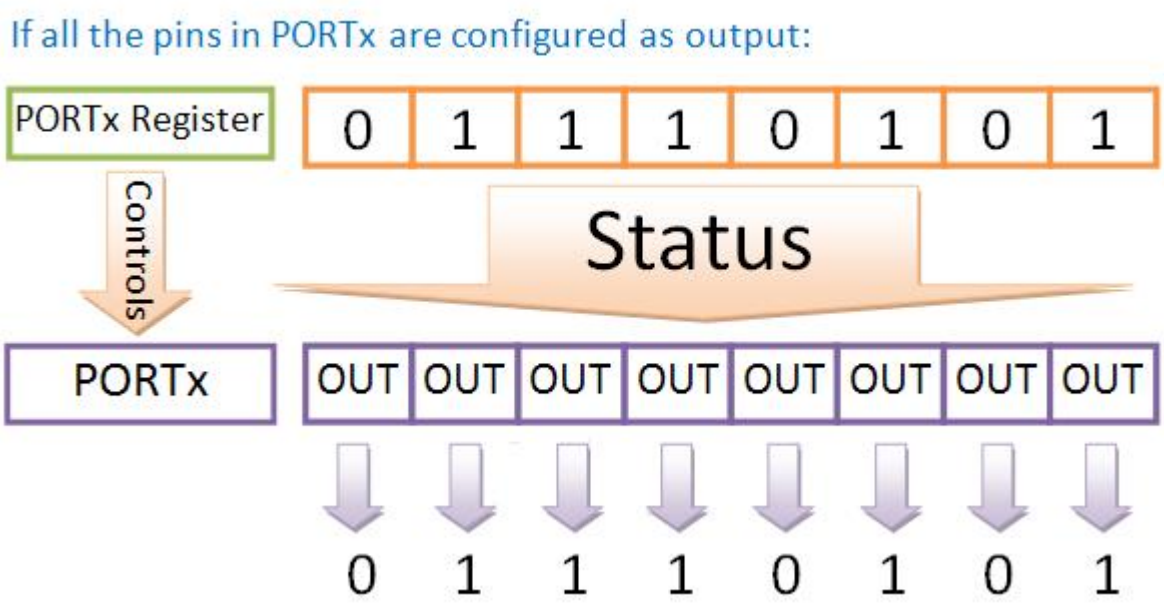


به عنوان مثال فرض کنید می‌خواهیم پین ۳ و ۷ از پورت B را به عنوان خروجی و بقیه ی پین ها را به عنوان ورودی تنظیم کنیم. در زبان C باید بنویسیم :

DDRB = 0b10001000;

۲) رجیستر PORTX :یکی دیگر از رجیسترهایی که وضعیت صفر/یک پین را کنترل می‌کند رجیستر PORTx است. هر بیت در این رجیستر پین متناظر در پورت را کنترل می کند. به عنوان مثال بیت پنجم از این رجیستر پین پنجم از پورت مورد نظر را کنترل خواهد کرد. این رجیستر بر حسب این که پورت به عنوان خروجی و یا ورودی تنظیم شده باشد (توسط تنظیم رجیستر DDRx)، دو وظیفه دارد:

حالت ۱) صفر یا یک کردن پین (در صورتی که پین مورد نظر در نقش خروجی عمل کند):



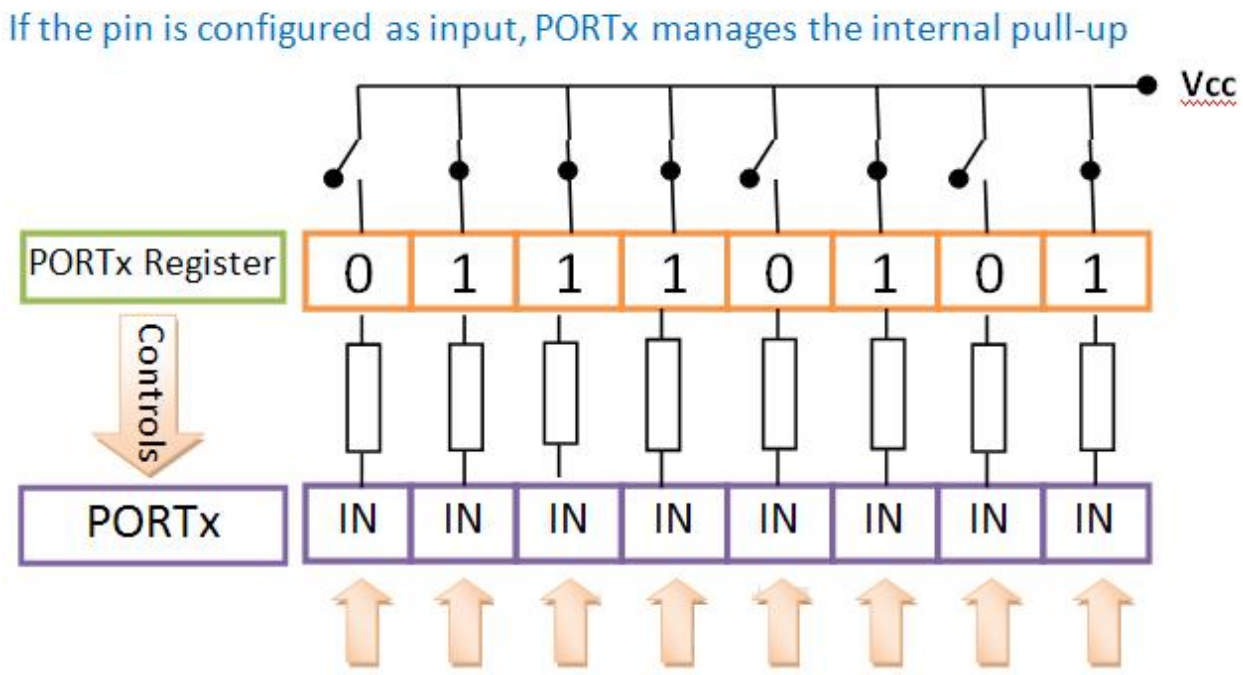
همان طور که در شکل بالا مشاهده می شود، در حالتی که پین به عنوان خروجی عمل کند (بیت مربوط به آن در رجیستر DDRx روی یک تنظیم شده باشد) با یک کردن بیت مربوط به آن پین در رجیستر PORTx ، مقدار آن پین در خروجی یک (high) خواهد شد. و به طور عکس با صفر کردن بیت مربوط به آن پین در رجیستر PORTx مقدار آن پین در خروجی صفر (low) خواهد شد. به برنامه زیر که به زبان C نوشته شده‌اند توجه کنید:

```
DDRB = 0b10001111;          /* Configuring I/O pins of  portb*/
```

```
PORTB = 0b10001010;       /* Write this byte to PORTB*/
```

در برنامه‌های بالا بیت‌های یک، سه و هفت ست (set) و بیت‌های صفر و دو ریست (reset) می شوند. دیگر بیت‌ها در حالت امپدانس بالا قرار می‌گیرند چرا که به عنوان پین های ورودی تنظیم شده اند.

حالت ۲) فعال/غیرفعال کردن مقاومت‌های پول آپ داخلی (در صورتی که پین مورد نظر در نقش ورودی عمل کند):



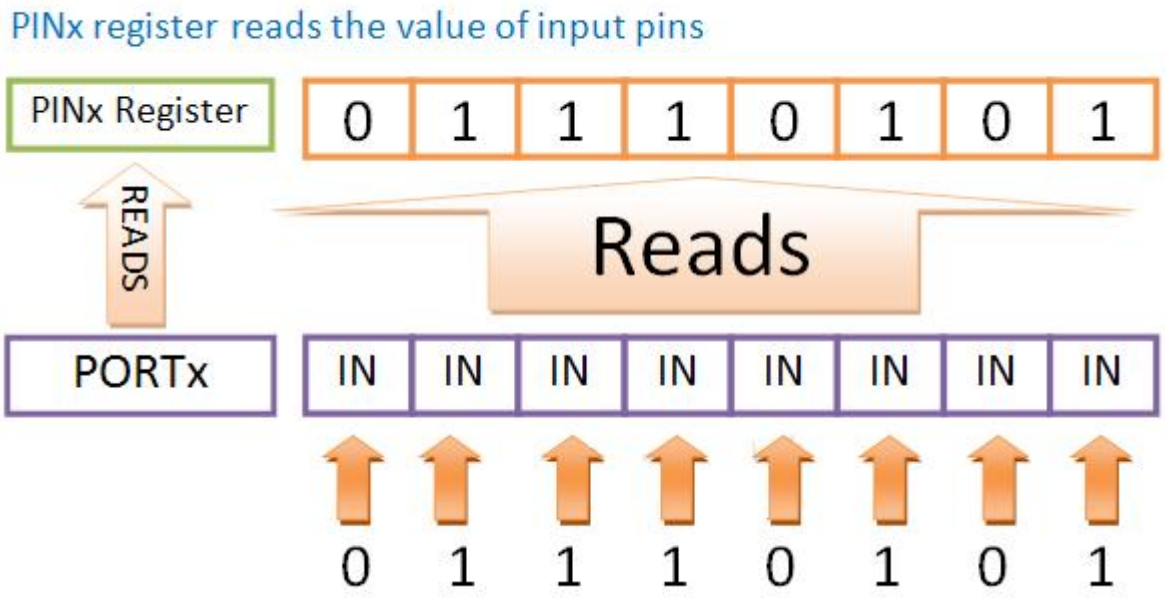
در میکروکنترلر AVR همه ی پین ها به مقاومت پول آپ مجهزند. در حالتی که یک پین در نقش ورودی عمل کند، مقاومت پول آپ مربوط به آن را می‌توان با برنامه‌ریزی بیت مربوط به آن در رجیستر PORTx فعال یا غیر فعال کرد. در صورتی که آن را با یک برنامه‌ریزی کنیم مقاومت پول آپ فعال می‌شود و به طور عکس اگر آن را با صفر برنامه‌ریزی کنیم مقاومت پول آپ غیر فعال می شود. فعال کردن مقاومت‌های پول آپ ورودی را به حالت یک می برد. با فعال کردن مقاومت‌های پول آپ از نویزی شدن ورودی جلوگیری می کنیم.

در برنامه زیر که به زبان C نوشته شده اند، پورت B به عنوان ورودی برنامه‌ریزی شده و سپس مقاومت‌های پول آپ پین های ۱، ۳، ۵، ۶ و ۷ آن فعال شده اند.

```
DDRB = 0b00000000;          /* Configuring I/O pins of portb*/
PORTB = 0b01110101;         /*enable internal pull-up resistors on output pins 1 , 3 , 5 , 6 , 7*/
```

✓ به طور پیش‌فرض پین های AVR در حالت ورودی و امپدانس بالا قرار دارند. یعنی هم بیت‌های رجیستر DDRx و هم بیت‌های رجیستر PORTx با صفر برنامه‌ریزی شده اند.

۳) رجیستر PINX :



رجیستر PINx شامل وضعیت تمام پین های مربوط به آن پورت است. اگر پین مربوطه به عنوان ورودی تنظیم شده باشد، بیت مربوط به آن پین در رجیستر PINx سطح منطقی آن پین را نشان می دهد. و اگر پین مربوطه به عنوان خروجی تنظیم شده باشد، بیت مربوط به آن پین در رجیستر PINx شامل آخرین داده ی خروجی بر روی آن پین می باشد. برنامه‌های زیر وضعیت پورت D را خوانده و ذخیره می کنند:

```
DDRD = 0b00000000;          /* Set all pins as input */
unsigned char status;       /* Define a 8 bit integer variable */
status = PIND;              /* Store the current value of PIND in status*/
```

۱-۲ مراحل اجرای آزمایش

در این آزمایش تمامی پروژه ها بدون استفاده از قابلیت CodeWizard تولید خواهد شد.

۱-۲-۱ راه‌اندازی LED

یک پروژه جدید در نرم‌افزار باز نمایید. یک source جدید باز نموده، کد زیر را در آن نوشته و به پروژه اضافه نمایید. در برنامه زیر پورت A به عنوان خروجی تعریف شده و سپس مقدار صفر برروی پینهای آن قرار داده می‌شود. در ادامه پین اول صفر شده و با تأخیرهای ۱ ثانیه برروی پورت شیفت داده می‌شود. پس از ساختن پروژه و تولید فایلهای برنامه، کد Hex را برروی میکروکنترلر پروگرام نمایید. خروجیهای پورت A را به LEDهای موجود برروی برد متصل نموده و نتیجه را مشاهده نمایید. خواهید دید که در هر ثانیه یکی از LEDها روشن بوده و در انتها به مدت ۱ ثانیه همه LEDها خاموش خواهند بود. این روند به صورت متناوب تکرار خواهد شد.

```
#include <mega32.h>
#include <delay.h>
void main(){
  DDRA=0xFF;
  PORTA=0x00;
  while(1){
    int i;
    PORTA.0=1;
    for (i=0; i<8; i++){
      delay_ms(1000);
      PORTA = PORTA << 1;}}}
```

۱-۲-۲ راه‌اندازی نمایشگر Seven-Segment

در این بخش از آزمایش، به جای LED از نمایشگر Seven-Segment نوع کاتد-مشترک استفاده خواهد شد. یک پروژه جدید همانگونه که در بخش قبل توضیح داده شده، باز نموده و برنامه زیر را به پروژه اضافه نمایید.

```
#include <mega32.h>
#include <delay.h>
char Data=0;
char Seven_Segment[10]={0x7E,0x0C,0xB6,0x9E,0xCC,0xDA,0xFA,0x0E,0xFE,0xDE};
void main(){
   DDRB=0xFF;
   PORTB=0x00;
   while(1){
      PORTB=Seven_Segment(Data);
      delay_ms(1000);
      if (Data == 9)  Data=0;
      else Data++;}}
```

پس از اجرای برنامه و تولید فایلهای نهایی، فایل Hex را برروی برد آزمایشگاه پروگرام نموده و نتایج را مشاهده نمایید. خواهید دید که اعداد ۰ تا ۹ با تأخیر ۱ ثانیه برروی نمایشگر نشان داده می‌شوند. در برنامه نوشته شده پایه های ۰ تا ۷ پورت B را به ترتیب به پایه‌های ممیز، a، b، c، d، e، f و g نمایشگر متصل نمایید.

تمرین: برنامه بخش ۱-۲-۱ را طوری تغییر دهید که LED ها بصورت کاملا تصادفی با تاخیر ۱ ثانیه روشن و خاموش شوند.