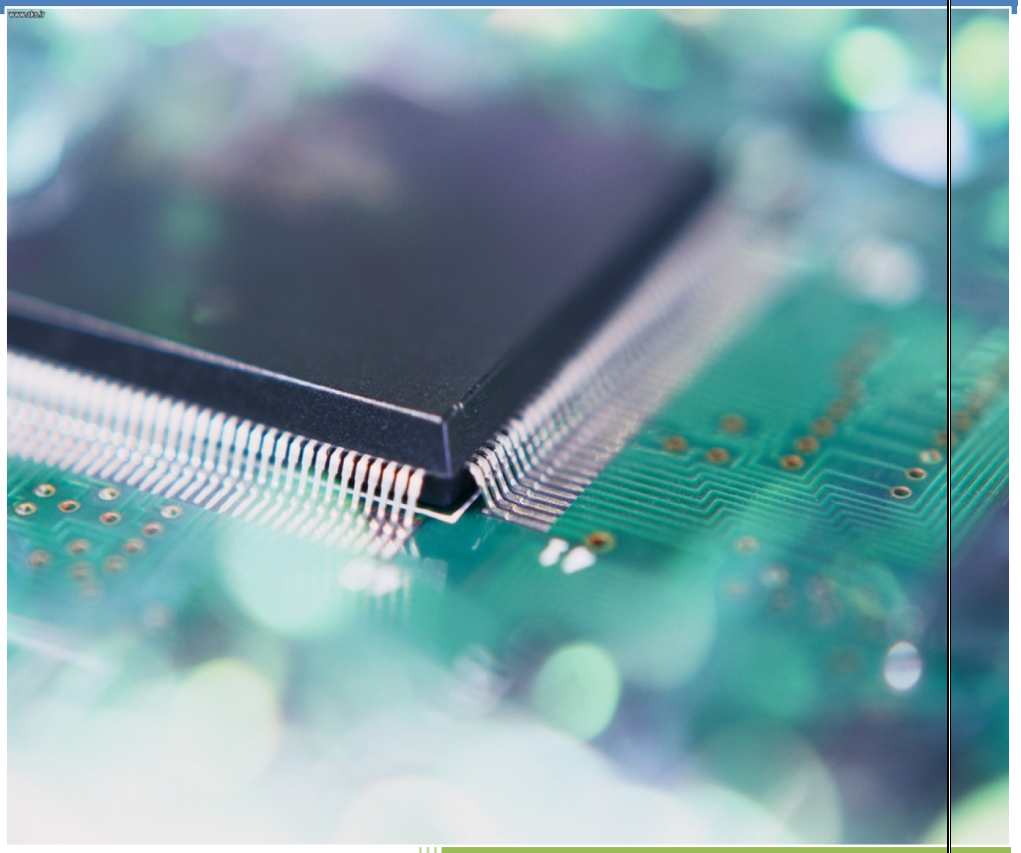
بسمه تعالی

دانشگاه کاشان

دستور کار آزمایشگاه ریزپردازنده



دانشکدهٔ برق و کامپیوتر

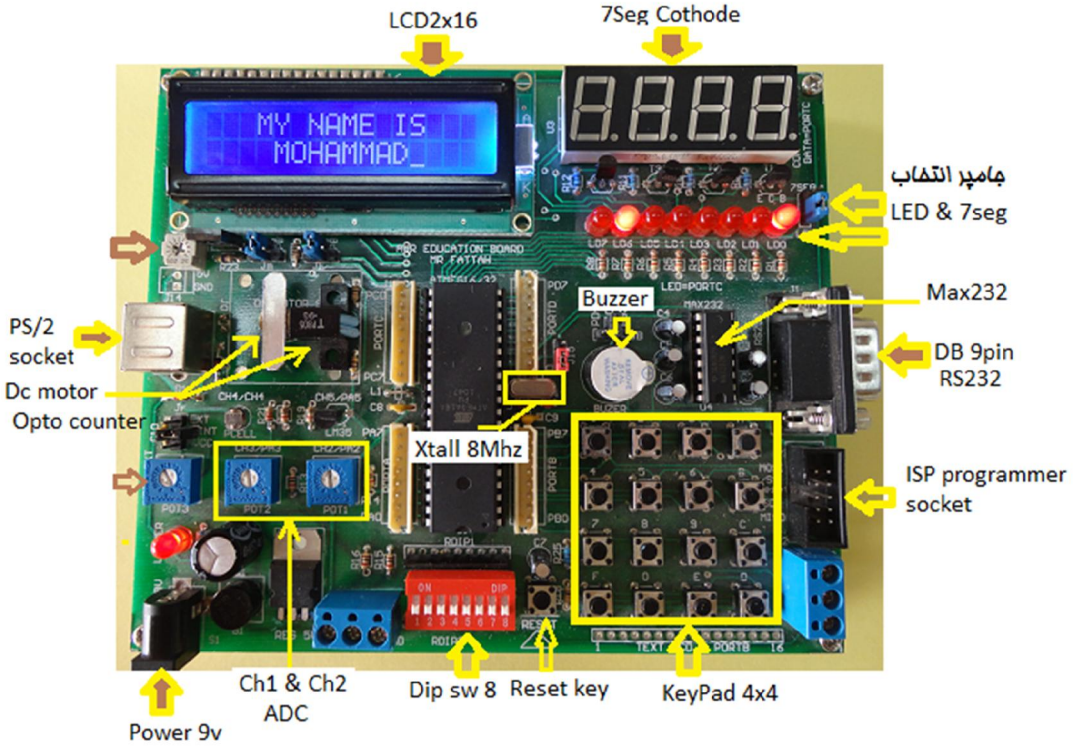
آزمایشگاه ریزپردازنده

محمّدرضا فتّاح

بسمه تعالی

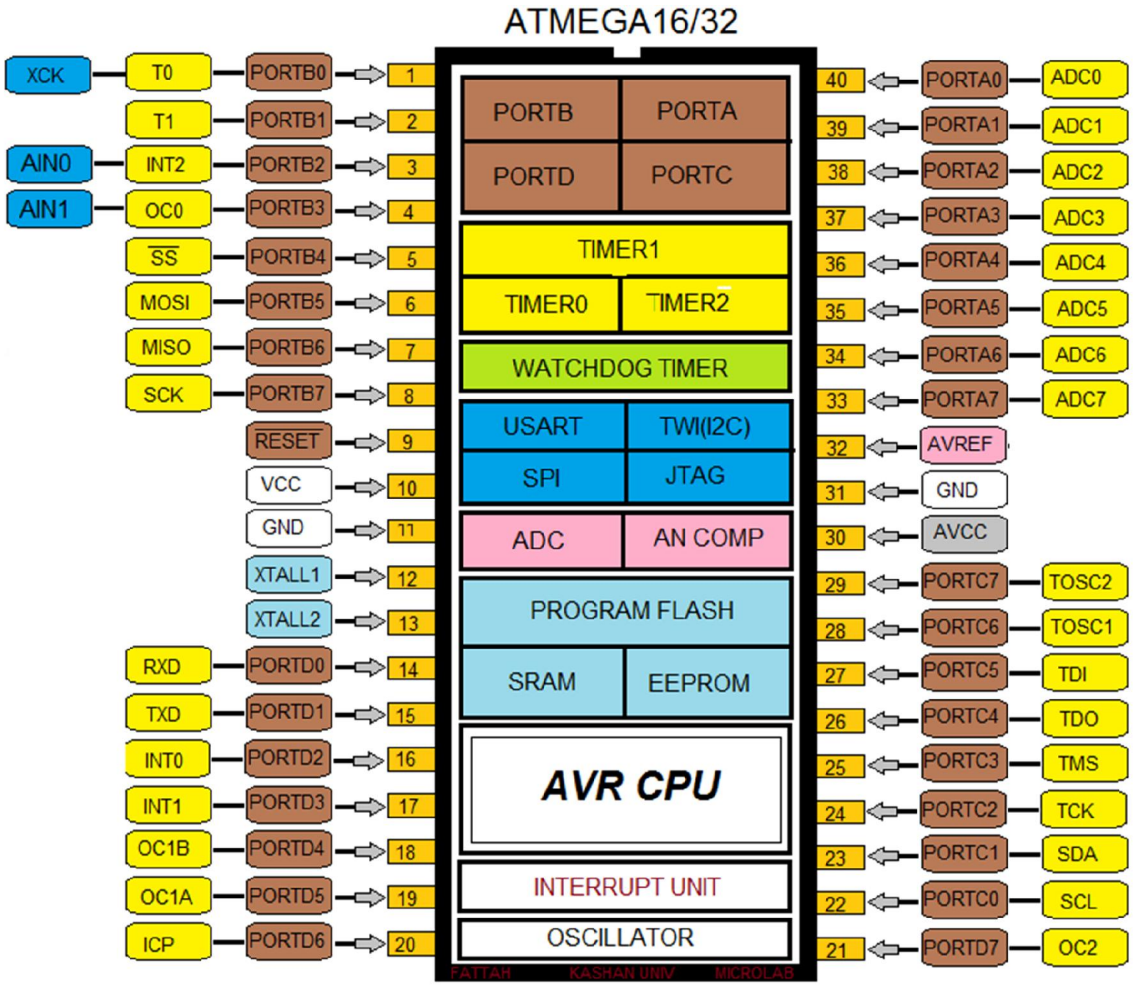
مقدمه :

راهنمای بورد آموزشی AVR



این بورد آموزشی با توجه به امکانات داخلی و قابلیتهای میکروکنترلرهای سری AVR طراحی شده و هستهٔ مرکزی آن میکروکنترلر Atmega16 است . در این بورد سعی شده است که دانشجو بتواند بدون نیاز به بستن مدارهای اضافی از تمام قابلیتهای سخت افزاری و نرم افزاری این میکرو کنترلر استفاده نماید . این میکروکنترلر دارای ۳۲ پایهٔ ورودی/ خروجی دیجیتال است . در صفحهٔ بعد شمای ظاهری همراه با امکانات داخلی آن و عملکرد پایه ها نمایش داده شده است . در صفحات بعد نیز جزئیات و امکانات بورد توضیح داده می شود.

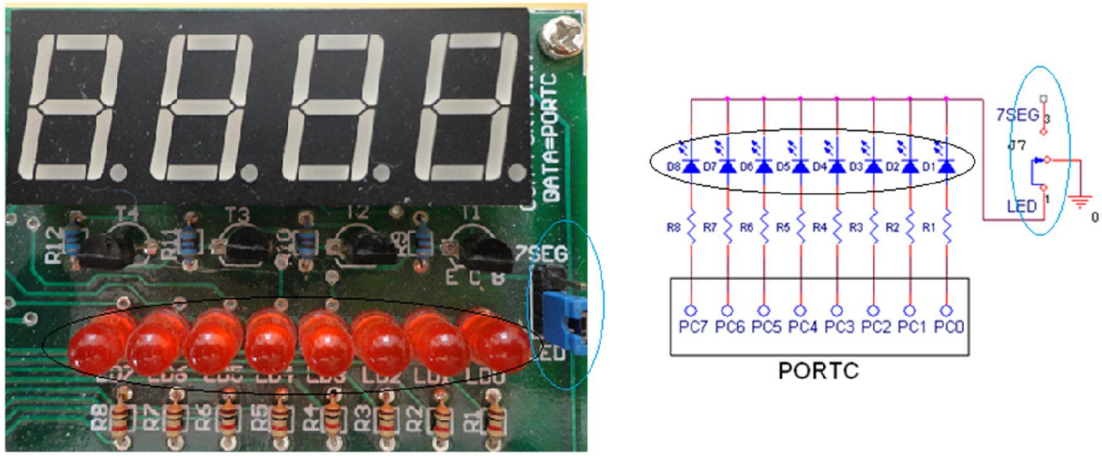
ساختار و عملکرد پایه های میکروکنترلر ATmega16 و یا ATmega32 و امکانات داخلی آن



نمایشگرها :

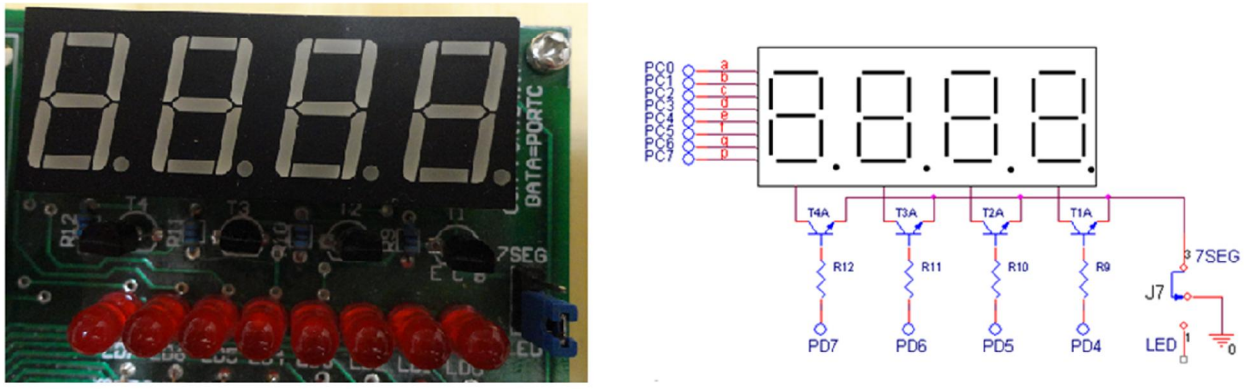
الف – هشت نمایشگر LED (سمت راست بورد)

همانطور که در شکل ۱ نشان داده شده است این LEDها به صورت کاتد مشترک بسته شده اند . آندهای آن نیز به PORTC متصل می باشد. برای فعال کردن این دسته LED لازم است که جامپرJ7 در حالت LED قرار گیرد .



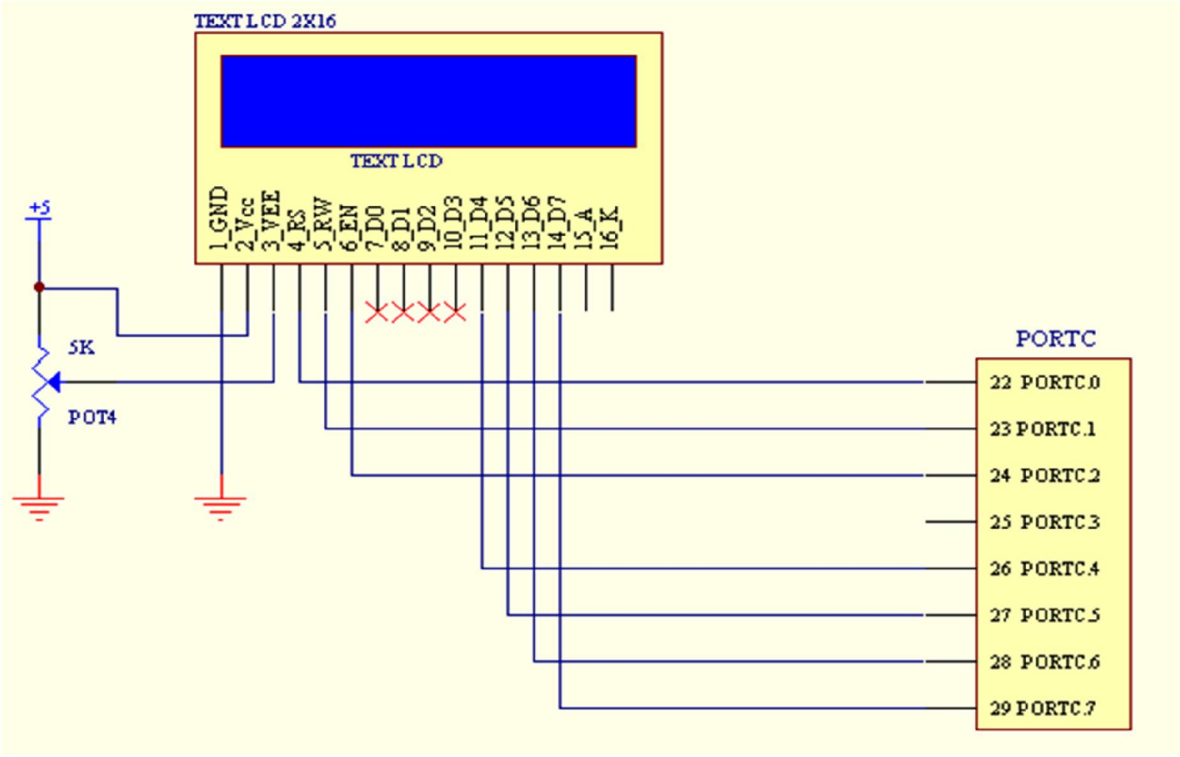
شکل ۱

ب –7_segment چهارتایی : سون سگمنت استفاده شده در این بورد از نوع چهارتایی مالتی پلکس کاتد مشترک است که دارای هشت خط داده و چهار پایهٔ کاتد است .هشت خط دیتای این قطعه به PORTC و چهار خط کنترلی (کاتدها) از طریق چهار ترانزیستور مطابق شکل زیر به PORTD متصل شده است . برای فعال نمودن این المان جامپر J7 باید در حالت 7_seg قرار گیرد .



شکل ۲

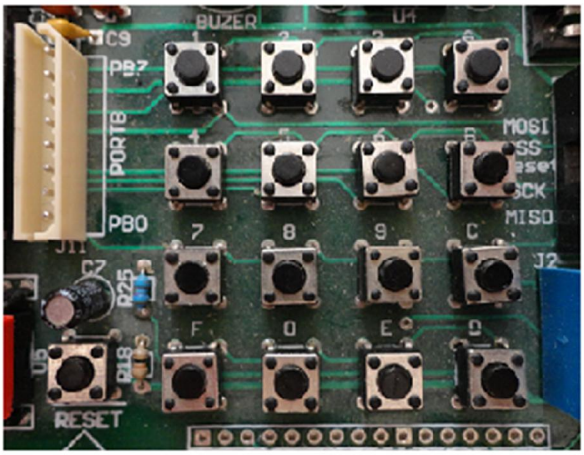
ج – نمایشگر LCD: در این بورد از یک LCD دو خط شانزده کاراکتری استفاده شده است . چگونگی اتصال نمایشگر LCD در شکل زیر نمایش داده شده است . مطابق شکل از هشت خط دیتای LCD ، چهار خط با ارزش به میکروکنترلر متصل است پس برای استفاده از این LCD باید از مود دیتای چهار بیتی استفاده شود . خطوط دیتا و کنترلی LCD به پورت C میکرو کنترلر متصل است .

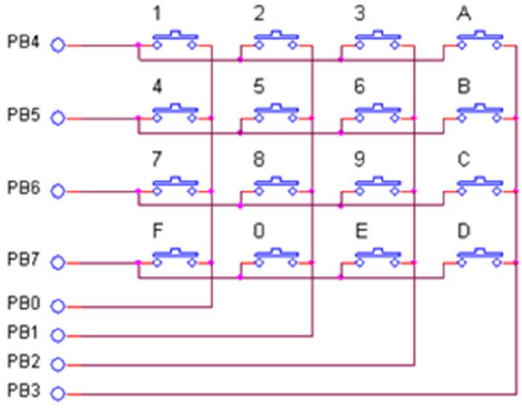


توجه: اگر از bascom برای برنامه نویسی استفاده می کنید باید پایهٔ RW (PORTC.1) را مستقیماً به ولتاژ صفر متصل کنید .

صفحه کلید چهار در چهار:

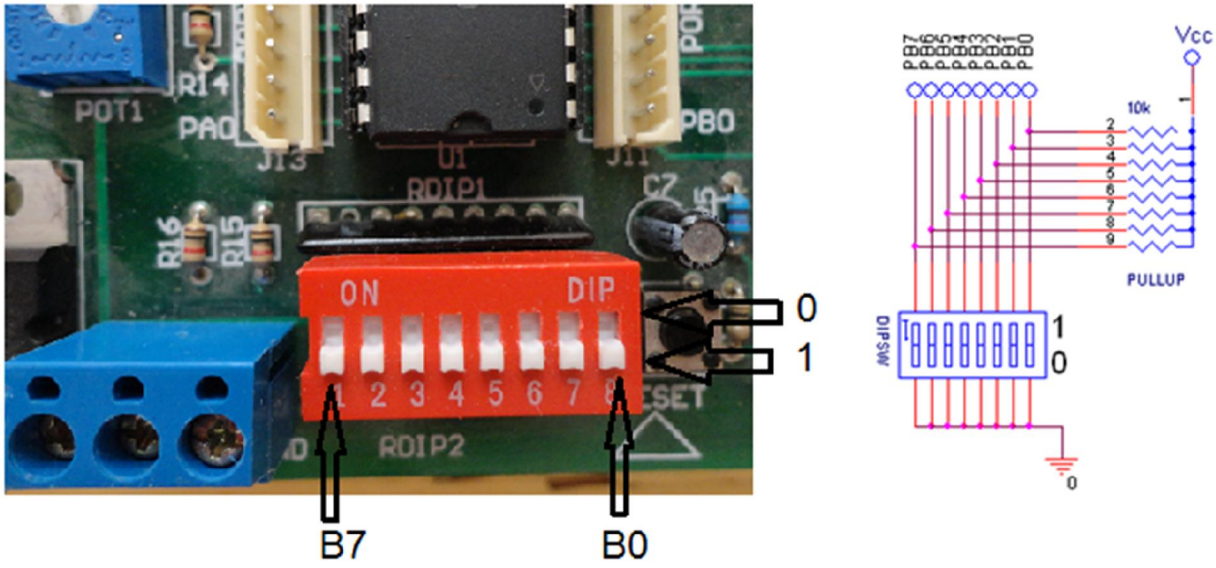
چگونگی اتصال سطرها وستونهای این صفحه کلید در شکل زیر آمده است .همانطور که در شکل مشاهده می شود ستونهای صفحه کلید به چهار بیت پایین PORTB و سطرها به چهار بیت بالای آن متصل است .





DIP_SW هشت تایی :

از این هشت کلید برای دادن اعداد به صورت باینری به پورت Bمی توان استفاده نمود . اگر هریک از کلیدها در حالت ON قرار گیرد وضعیت پورت در حالت منطق صفر خواهد بود . در این بورد آموزشی جهت حفاظت از پورت ،یکطرف کلید از طریق مقاومت ردیفی ۲۲۰ اهم به زمین متصل شده و طرف دیگر به منظور Pullup شدن پورت از طریق مقاومت ردیفی ۱۰ کیلو اهم به تغذیه پنج ولت متصل است .

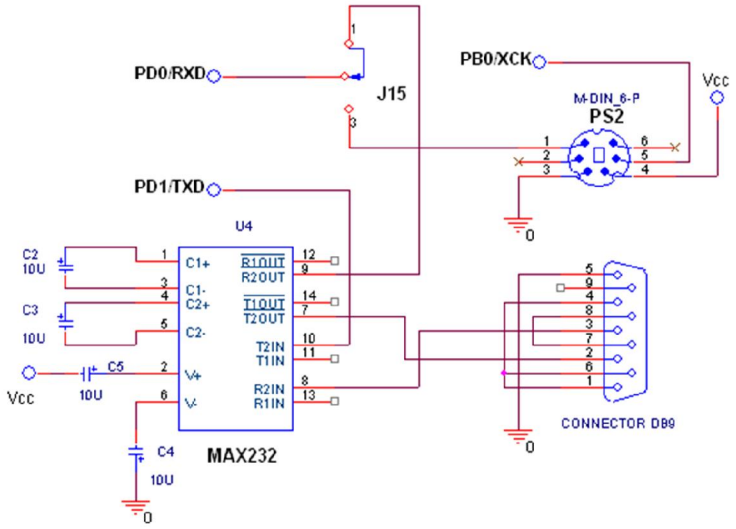


ورودی و خروجی سریال RS232و PS/2:

پایه های RXD و TXD میکروکنترلر از طریق مبدل MAX232 به سوکت ۹ پایه متصل شده است . با استفاده از این سوکت و کابل مخصوص همراه بورد می توان با دستگاههای دیگر مانند کامپیوتر که قابلیت ارتباط RS232 را دارند ارتباط برقرار کرد.

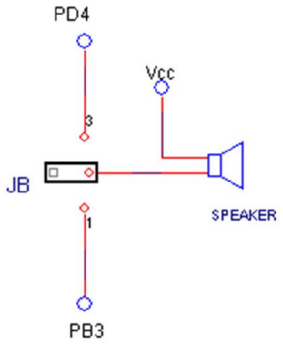
اتصال PS/2 :

از طریق این سوکت می توان صفحه کلید کامپیوتر را به میکروکنترلر متصل نمود. شکل زیر چگونگی اتصال را مشخص کرده است .



المان تولید صدا (BUZZER) :

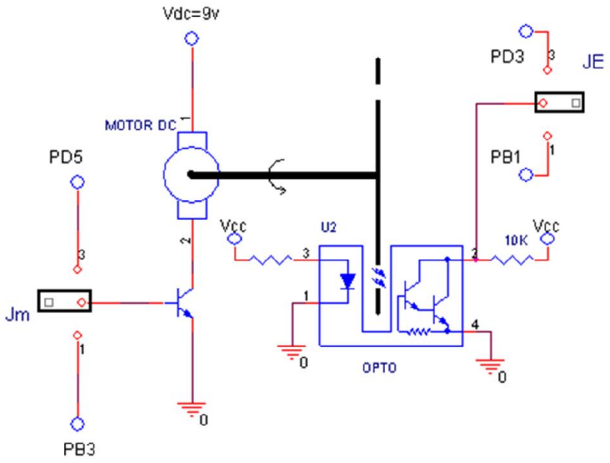
بازر از طریق جامپر J3 به PORTD.4 و یا PORTB.3 متصل می شود که با تغییر وضعیت جامپر می توان منبع تحریک بازر را تغییر داد . اگر یکی از این دو پایه در منطق صفر باشد بازر روشن و در غیر این صورت خاموش خواهد بود .



موتور DC و اینکودر :

بورد آموزشی AVR دارای یک موتور DCاست که با ولتاژ ۹ ولت تغذیه می شود . مطابق شکل می توان با تحریک بیس ترانزیستور T5 موتور را خاموش و یا روشن نمود . بیس ترانزیستور از طریق جامپر J17می تواند به PORTB3 و PORTD.5 متصل شود که از طریق این دو پورت می توان موتور را تحریک کرد .

همچنین این بورد دارای یک اینکودر است که می توان با استفاده از آن دور موتور را اندازه گیری نمود . خروجی اینکودر از طریق جامپر J16 به PORTB.1 و PORTD.3 متصل می شود که با یکی از این دو پورت می توان پالسهای خروجی اینکودر را دریافت نمود .



تجهیزات آنالوگ متصل به PORTA :

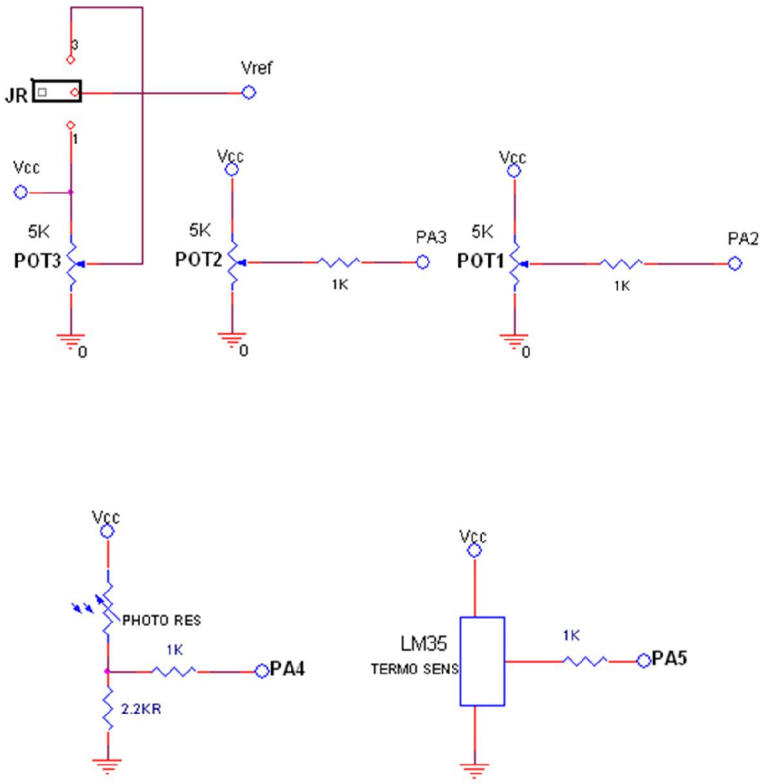
ترمینال سه پایه : دو پایه از این ترمینال به PORTA.0 و PORTA.1 متصل است که با استفاده از آن می توان سیگنالهای خارجی آنالوگ با دامنهٔ ۵ ولت را به ورودی پورت اعمال نمود .

پتانسیومترهای POT2 و POT3 : با استفاده از این دو پتانسیومتر که به PORTA.2 و PORTA.3 متصل هستند می توان ولتاژ متغیر از صفر تا پنج ولت را اعمال نمود .

فتورزیستور(مقاومت متغیر با شدت نور) : خروجی مدار حاوی این المان به PORTA.4 متصل شده است . با استفاده از این مدار می توان شدت نور را اندازه گیری نمود .

سنسور دما (LM35) : خروجی این المان ولتاژ متغیر با دما است. این خروجی به PORTA.5 متصل شده است .

۵- مرجع ولتاژ : با استفاده از جامپر J9 می توان مرجع ولتاژ را تغییر داد . البته باید توجه داشته باشید که اگر مرجع ولتاژ را مرجع داخلی میکرو انتخاب می کنید جامپر نباید در حالتهای Vcc و یا حالت متغیر باشد .



آزمایش اول

آشنایی با پورتهای ورودی و خروجی:

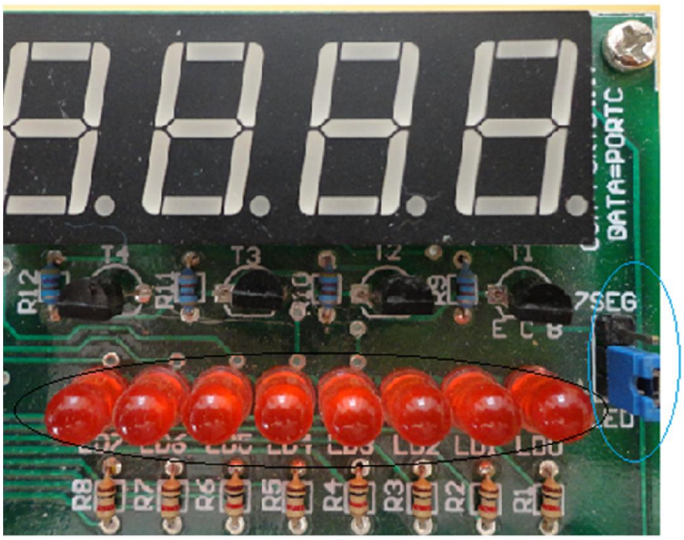
پیش گزارش : ندارد

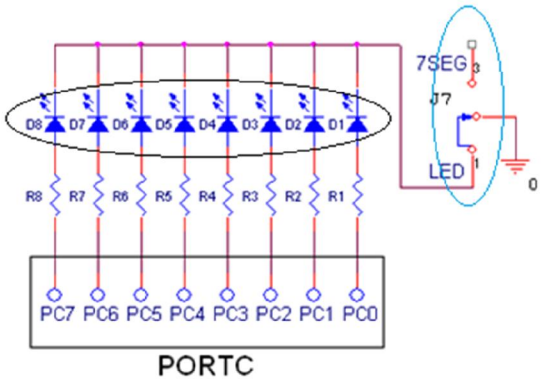
قبل از نوشتن برنامه ها و انجام آزمایش ، ضمیمهٔ آزمایش اول را مطالعه فرمایید .

آزمایش ۱: اجرای یک برنامهٔ نمونه

برنامهٔ شمارندهٔ باینری هشت بیتی را در محیط AVRSTUDIO نوشته و سپس آنرا کامپایل کنید . در صورت نداشتن خطا ، فایل HEX آنرا روی بورد پروگرام کنید و اجرای آنرا ببینید .

توجه : برای فعال شدن LED های متصل به PORTC جامپر J7 باید در وضعیت LED قرار گیرد .





آزمایش۲ : شمارندهٔ دهدهی

با استفاده آز دستورات CPI و BRNE و یا BREQ برنامهٔ قبل را برای ساخت شمارنده ای که اعداد صفرتا ۹ را شمارش نماید تغییر دهید. خروجی را روی چهار LED اول نمایش دهید.

سوال ۱: تفاوت دستور CPI و CP در چیست ؟ همچنین تفاوت عملکرد این دو با دستور تفریق(SUB) چگونه است ؟

سوال ۲ : دستورات شرطی برای اعداد علامت دار و بدون علامت را بصورت جداگانه بنویسید .

آزمایش۳ : شمارندهٔ دهدهی دو رقمی

یک شمارندهٔ دهدهی دو رقمی (رجیستر R0 رقم یکان و R1 رقم دهگان) را روی LED ها نمایش دهید بطوریکه رقم یکان (R0) روی چهار LED کم ارزش و رقم دهگان (R1) روی چهار LED پرارزش قرار گیرد .

به عملکرد دستورSWAPتوجه کنید.

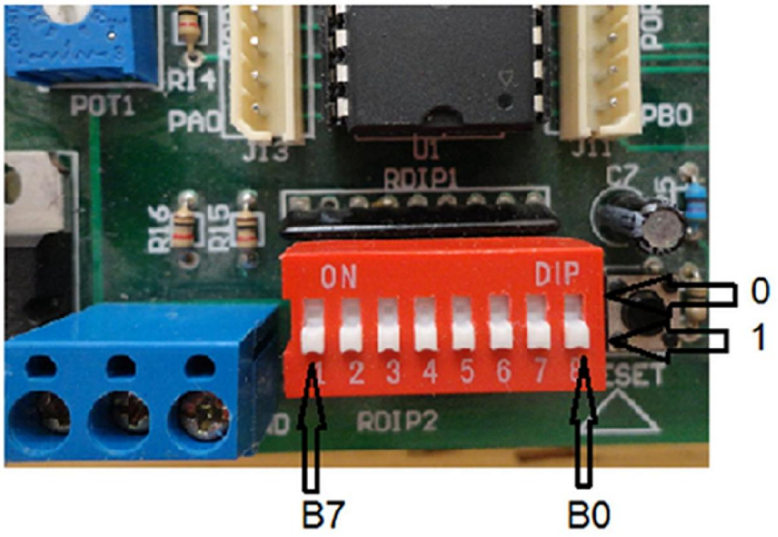
آزمایش ۴ : شمارندهٔ جانسون

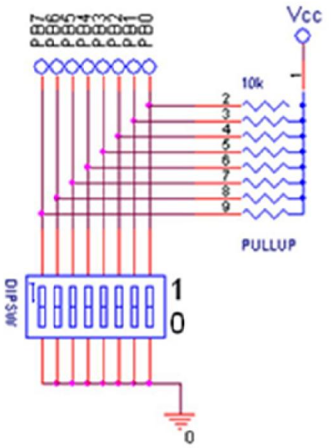
برنامه ای بنویسید که شمارندهٔ جانسون هشت بیتی را روی LEDها نمایش دهد(LEDها از سمت راست یک به یک روشن می شود تا اینکه هر هشت LED روشن شود و سپس از سمت راست به همان ترتیب خاموش می شود)

توصیه می شود برای این برنامه از دستورات شیفت و چرخش به چپ (LSL و ROL) و همچنین دستورات پرس شرطی (SBRS وSBRC) استفاده نمایید .

آزمایش۵ : خواندن آز پورت

برنامه ای بنویسید که مقدار دیپ سوییچ متصل به PORTBرا بخواند و به صورت وارونه روی LED ها نمایش دهد(بیت کم ارزش روی LED پرارزش و بیت پرارزش روی LED کم ارزش نمایش داده شود) .





ضمیمهٔ آزمایش اول :

الف - پورتهای میکروکنترلر ATmega16/32 :

این میکروکنترلر دارای چهار پورت هشت بیتی بنامهای A ، B ، C و Dاست که هر بیت از این پورتها می تواند به صورت مجزا مورد استفاده قرار گیرد . پس می توان گفت این تراشه دارای ۳۲ دریچهٔ I/Oاست . (به شکل تراشه در ابتدای دستور کار توجه کنید).

در این تراشه هر پورت هشت بیتی دارای دو رجیستر و یک بافر برای تعیین وضعیت پورت و انتقال اطلاعات است که در زیر توضیح داده می شود:

رجیستر DDRx : این رجیستر جهت انتقال اطلاعات (ورودی یا خروجی) را مشخص می‌کند . اگر در هر بیت این رجیستر مقدار یک نوشته شود پایهٔ معادل این بیت در پورت به عنوان خروجی و اگر مقدار صفر نوشته شود آن پایه به عنوان ورودیتعریف می‌گردد .

مثال :

```
LDI   R16 , $0F
OUT   DDRA , R16
```

این دو دستور مشخص می‌کند که چهار بیت کم اَرزش پورت A بعنوان خروجی و چهار بیت پرارزش بعنوان ورودی تعریف شده است .

رجیستر PORTx : مقدار نوشته شده در این پورت روی پایه های پورت قرار میگیرد به شرط اینکه پورت بعنوان خروجی تعریف شده باشد . اگر پایه در حالت ورودی تعریف شده باشد مقدار نوشته شده در این ثبات مشخص کنندهٔ وضعیت فعال و یا غیر فعال مقاومت Pullup است .

مثال :

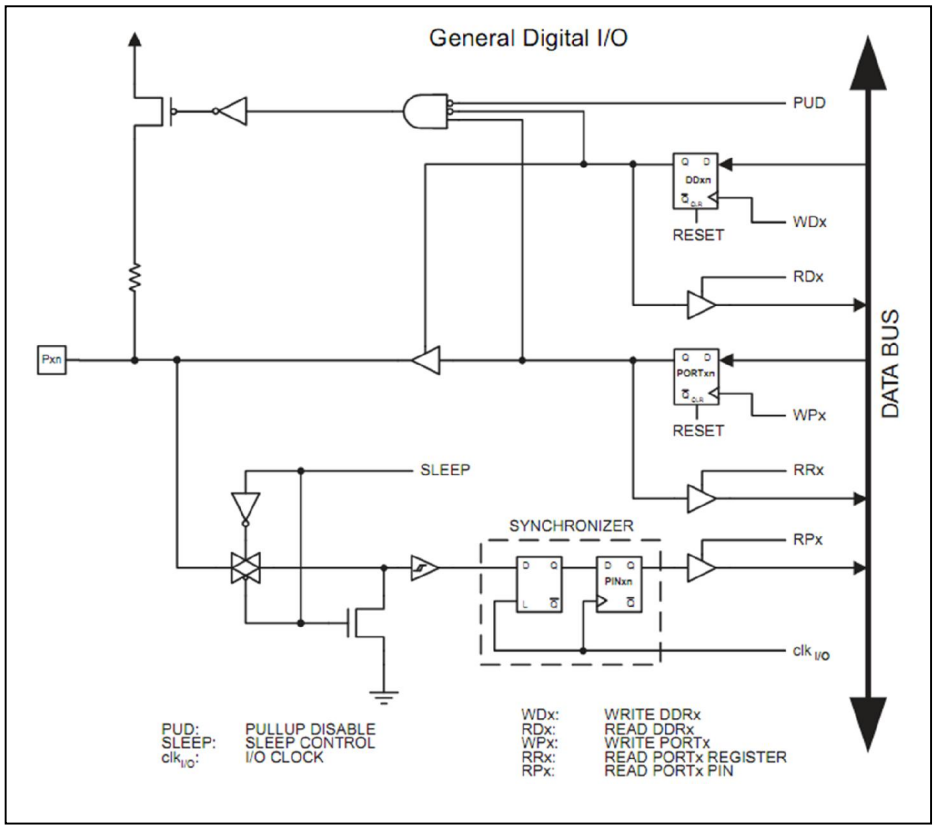
```
LDI   R16 , $F5
OUT   PORTB , R16
```

این دستورات مقدار باینری (11110101)را روی پایه های پورت B قرار می‌دهد . توجه داشته باشید که اگر پورت در حالت خروجی تعریف نشده باشد مقدار نوشته شده در این رجیستر روی پایه های پورت قرار نخواهد گرفت .

بافر PINx : اگر پورت را بعنوان ورودی تعریف کرده باشیم مقداری که روی پایه های پورت قرار دارد را می توانیم اَز طریق این بافر دریافت کنیم . به دستور زیر توجه کنید :

```
IN   R16 , PINB
```

این دستور مقدار قرار گرفته روی پایه های پورت B را در ثبات R16 قرار می دهد .



البته به این نکته نیز باید توجه داشته باشید که اگر پورت را بعنوان ورودی تعریف می‌کنید باید مقاومت داخلی Pullup آنرا فعال کنید و یا اینکه یک مقاومت خارجی بین هر کدام از پایه های پورت و ولتاژ Vcc=5v قرار دهید . مقدار این مقاومت می تواند از 1KΩ تا 100KΩ باشد . به مثال زیر توجه کنید :

```
LDI   R16 , $0F

OUT   DDRA , R16

LDI   R16 , $F0

OUT   PORTA , R16
```

دو دستور اول ، چهار پایهٔ کم ارزش پورت A را بعنوان خروجی و چهار بیت بالا را بعنوان ورودی تعریف می‌کند . دو دستور بعدی با نوشتن مقدار یک در چهار بیتبالای رجیستر PORTA باعث فعال شدن مقاومت Pullup داخلی چهار پایهٔ بالای پورت A می‌شود . و همینطور مقدار (0000) را روی چهار پایهٔ کم ارزش پورت که بعنوان خروجی تعریف شده‌است قرار می‌دهد .

ب - فرکانس کار میکروکنترلر:

در میکروکنترلها ، مرجع تولید فرکانس می تواند بصورت داخلی و یا خارجی تعیین گردد . در بورد آموزشی آزمایشگاه این کار توسط یک کریستال 8MHz خارجی اَنجام می گیرد . با توجه به مقدار این کریستال می توان فرکانس کاری و یا زمان

اجرای یک دستور را مشخص نمود . در میکروکنترلرهای AVR اکثر دستورات در یک سیکل و بعضی از آنها در دو و یا سه سیکل اجرا می شوند . زمان یک سیکل برابر با عکس فرکانس کاری میکروکنترلر است :

Tcycel=1/8 µsec

ج – جدول آدرس پورتهای AVR

I\O PORT:

Port	Register	Function	Port-Address
A	PORTA	Data Register	0x1B
	DDRA	Data Direction Register	0x1A
	PINA	Input Pins Address	0x19
B	PORTB	Data Register	0x18
	DDRB	Data Direction Register	0x17
	PINB	Input Pins Address	0x16
C	PORTC	Data Register	0x15
	DDRC	Data Direction Register	0x14
	PINC	Input Pins Address	0x13
D	PORTD	Data Register	0x12
	DDRD	Data Direction Register	0x11
	PIND	Input Pins Address	0x10

Port Pin Configurations:

DDxn	PORTxn	PUD (inSFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

د- رجیستر وضعیت

Status-Register, Accumulator flags

Port	Function	Port-Address	RAM-Address
SREG	Status Register Accumulator	0x3F	0x5F

7	6	5	4	3	2	1	0
I	T	H	S	V	N	Z	C

آزمایش دوم

محاسبات ریاضی

پیش گزارش :

۱- زیر برنامه ای بنویسید که دو عدد دو بایتی را تفریق کند . بایت کم اَرزش و پراَرزش عدد اول به ترتیب در رجیسترهای R6 و R7 و عدد دوم در رجیسترهای R8 و R9 قرار دارد . حاصل جمع نیز باید در رجیسترهای R6 ،R7 و R8قرار گیرد .

۲- زیر برنامه ای بنویسید که دو عدد یک بایتی را بر هم تقسیم کند . این اَعداد در رجیسترهای R6 و R7 قرار دارند . خارج قسمت در R6و باقیمانده را درR7قرار دهید .

۳- زیر برنامه ای بنویسید که یک عدد دو بایتی را در یک عدد یک بایتی ضرب کند . عدد دو بایتی به ترتیب ارزش در رجیسترهای R6 و R7 و عدد یک بایتی در رجیستر R8 قرار دارد . حاصل ضرب را نیز در همین رجیسترها به ترتیب ارزش قرار دهید .

۴- زیر برنامه ای بنویسید که با استفاده از زیر برنامۀ تقسیم ، عدد باینری موجود در رجیستر R16 را به معادل BCD تبدیل کند . اعداد یکان ، دهگان و صدگان به ترتیب در رجیسترهای R6 ،R7 و R8 قرار گیرد .(با استفاده از روش تقسیم بر ده)

آزمایش ۱ : تفریق دو بایتی

برنامه ای بنویسید که ابتدا دو عدد دلخواه دو بایتی را در ثباتهای R6 ، R7 و R8 ، R9 ذخیره کرده ، سپس با استفاده اَز زیر برنامۀ یک ، آن دو را تفریق کند و حاصل تفریق را به ترتیب اَرزش و هر یک را به مدت یک ثانیه روی LEDهای متصل به PORTC نمایش دهد .

آزمایش ۲ : ضرب دو بایت در یک بایت

برنامه ای بنویسید که ابتدا یک عدد دلخواه دو بایتی و یک عدد دلخواه یک بایتی را در مکانی مناسب ذخیره کرده ، سپس با استفاده اَز زیر برنامۀ سه ، آن دو را در هم ضرب کند و حاصل ضرب را به ترتیب اَرزش و هر یک را به مدت یک ثانیه روی LEDهای متصل به PORTC نمایش دهد .

آزمایش ۳ : تبدیل باینری به دهدهی

برنامه ای بنویسید که ابتدا یک عدد باینری را اَز کلید هشت تایی متصل به PORTB بخواند و سپس مقدار معادل BCD آنرا با استفاده از زیر برنامه چهار به دست آورده و اعداد یکان و دهگان را به ترتیب روی چهار LED اول و دوم به مدت یک ثانیه نمایش دهد سپس عدد صدگان را روی چهار LED اول به مدت یک ثانیه جایگزین کرده و نمایش دهد .

آزمایش سوم

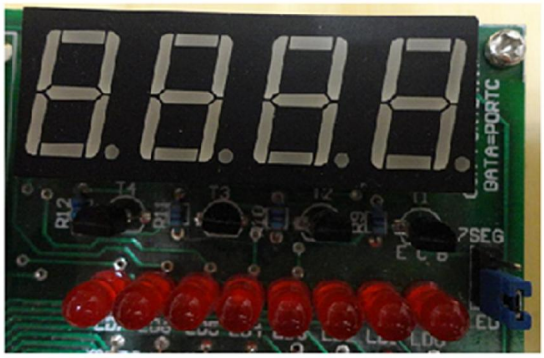
نمایش اعداد توسط سون سگمنت چهار تایی

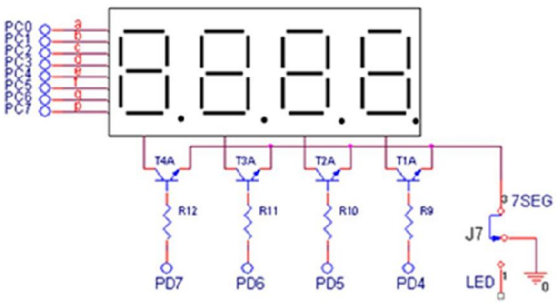
قبل از نوشتن برنامه ها و انجام آزمایش ، ضمیمهٔ آزمایش سوم را مطالعه فرمایید .

پیش گزارش:

۱- زیر برنامه‌ای بنویسید که یک عدد باینری چهاربیتی موجود در ثبات R16 را به معادل کد 7_seg آن تبدیل کند و مقدار تبدیل را در R16 قرار دهد . از روش جدول جستجو (Lookup Table) استفاده کنید.

۲- زیر برنامه‌ای بنویسید که یک عدد چهار رقمی که رقمهای آن به ترتیب در رجیسترهای R0 ، R1 ، R2 و R3 موجود است را توسط سون سگمنت چهارتایی روی بورد یکبار نمایش دهد و سپس از زیر برنامه خارج شود.(هر سون سگمنت بمدت سه میلی ثانیه نمایش داده شود)





آزمایش ۱ : نمایش اعداد

با استفاده از زیر برنامه اول ، برنامه ای بنویسید که چهار بیت اول سمت راست کلید هشت تایی (PORTB) را بخواند و عدد معادل هگز آنرا روی سون سگمنت اول نمایش دهد .فرض براین است که فقط یک سون سگمنت داریم .برای این منظور ابتدا پایهٔ کاتد سون سگمنت اول را فعال کنید. این کار با نوشتن مقدار یک در پایهٔ PORTD.4 انجام می شود.

آزمایش ۲ : نمایش اعداد چند رقمی

با استفاده از زیربرنامهٔ دوم و زیر برنامهٔ تبدیل باینری به دهدهی(جلسه قبل) برنامه ای بنویسید که یک عدد باینری هشت بیتی را از کلید هشت تایی بخواند و مقدار دهدهی آنرا روی سون‌سگمنتها نمایش دهد .

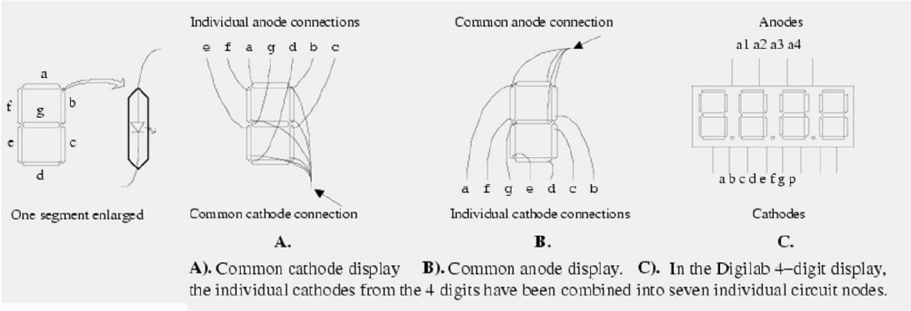
آزمایش ۳ : نمایش اعداد بصورت چرخشی

برنامه‌ای بنویسید که اعداد 0 تا 9 را به صورت چرخشی روی چهار سون‌سگمنت نمایش دهد به طوریکه اعداد از سون‌سگمنت سمت راست وارد و از سمت چپ خارج شوند .مقدار اولیه نمایش داده شده روی سون سگمنت صفر باشد.

ضمیمهٔ آزمایش سوم

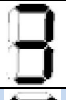
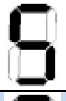
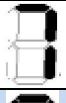
برای نمایش اعداد در مدارهای الکترونیک از سون سگمنت استفاده می شود. هر سون سگمنت از هفت LEDبرای نمایش عدد که از a تا g نامگذاری می شود و از یک LED با نام p برای نقطهٔ اعشار استفاده می کند . سون سگمنتها به دو صورت کاتد مشترک و آند مشترک تولید می شوند.

سون سگمنتها بصورت دو ، سه و یا چهارتایی نیز ساخته می شوند . سون سگمنت بورد آزمایشگاه از نوع چهار تایی مالتی پلکس و کاتد مشترک است .هشت خط دیتا برای هر چهار سون سگمنت مشترک است و هر کدام خط کاتد مخصوص به خود را داراست.در این نوع سون سگمنت در هر لحظه فقط می توان یکی از سون سگمنتها را فعال و یک عدد را نمایش داد . شیوهٔ نمایش روی این نوع از سون سگمنت در زیر توضیح داده شده است .



A). Common cathode display **B).** Common anode display. **C).** In the Digilab 4–digit display, the individual cathodes from the 4 digits have been combined into seven individual circuit nodes.

اگر یک لامپ در بازۀ زمانی بیست میلی ثانیه یک بار روشن و خاموش شود و این کار بصورت مداوم ادامه یابد چشم انسان لامپ را همیشه روشن خواهد دید . حال اگر بخواهیم روی یک بلوک سون سگمنت چهارتایی یک عدد چهار رقمی نمایش دهیم باید ابتدا بازۀ 20msecرا به چهار قسمت تقسیم کنیم و در هر قسمت زمانی یکی از سون سگمنتها را فعال کرده و رقم مورد نظر را روی آن نمایش دهیم . در این صورت در یک بازۀ 20msec یک سون سگمنت در مدت زمان 5msec روشن و در 15msec بعد خاموش خواهد بود . هر چه نسبت زمان روشن بودن به خاموش بودن کمتر باشد سون سگمنت کم نورتر به نظر خواهد رسید . برای نمایش هر عدد روی سون سگمنت ابتدا باید کد سون سگمنت آن عدد را بدست آورد سپس همین کد را روی پایه های دیتای سون سگمنت قرار داد . برای این منظور از جدول زیر استفاده کنید .

عدد	نمایش	کد هگز	وضعیت سگمنت							
			p	g	f	e	d	c	b	a
0		3f	0	0	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		6e	0	1	1	0	1	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		67	0	1	1	0	1	1	1	1

آزمایش چهارم

نمایشگر LCD

پیش گزارش :

۱- یک زیر برنامه با نام Lcd_command بنویسید که فرمان موجود در رجیستر R16 را برای LCD ارسال کند.

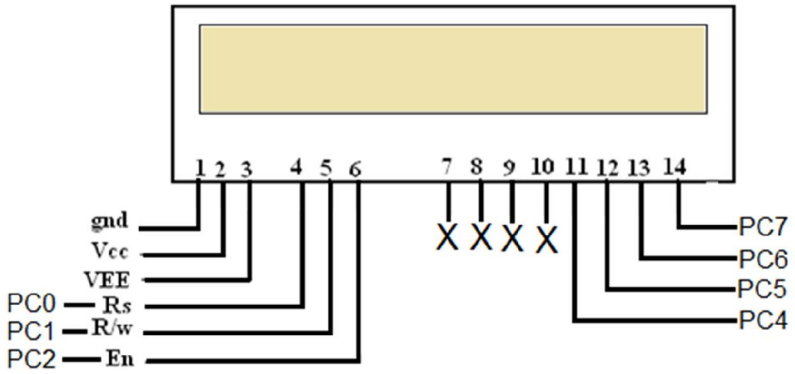
۲- زیر برنامه ای با نام Lcd_putchar بنویسید که کاراکتر نمایشی موجود در رجیستر R16 را برای LCD ارسال کند.

۳- زیر برنامه ای با نام Lcd_puts بنویسید که جمله ای را از مکانی مشخص از حافظهٔ برنامه (Lookup Table) بخواند و آنرا روی LCD چاپ نماید . انتهای جمله با عدد 0 مشخص شده است . مانند مثال زیر:

.DB “Lcd Display” , 0

۴- زیر برنامه ای با نام Lcd_gotoxy بنویسیدکه با اجرای آن کرسر به مکانی از LCD منتقل شود که موقعیت آن توسط رجیسترهای R16 (موقعیت افقی در خط) و R17 (شمارهٔ خط) مشخص شده است .

توجه : مطابق شکل فقط چهار خط بالای دیتای LCD به میکروکنترلر متصل است .



آزمایش ۱ : نمایش کاراکتر روی LCD

با استفاده از زیر برنامه های نوشته شده در پیش گزارش،برنامه ای بنویسید که نام خانوادگی خود و هم گروهیتان را به ترتیب در خط اول و دوم LCD نمایش دهد.

آزمایش ۲ : نمایش اعداد روی LCD

برنامه ای بنویسید که دیپ سوییچ را بخواند سپس مقدار آنرا بصورت باینری روی خط اول و مقدار دهدهی آنرا روی خط دوم LCD نمایش دهد .برای حالت دوم از زیر برنامهٔ تبدیل باینری به دهدهی که در جلسات قبل نوشته اید استفاده نمایید .

آزمایش ۳ : نمایش حروف فارسی

برنامه ای بنویسید که جملهٔ **" به نام خدا"** را روی خط اول LCD نمایش دهد .

ضمیمهٔ آزمایش چهارم :

LCD کاراکتری روی بورد آموزشی از دو خط نمایشی که هر خط ظرفیت نمایش ۱۶ کارکتر را دارد تشکیل شده است . همچنین این LCD دارای ۸ خط برای تبادل اطلاعات بین خود و میکروکنترلر است و دارای سه خط کنترلی Rs برای تعیین نوع اطلاعات ارسالی(فرمان و یا کاراکتر نمایشی) ، RW برای تعیین جهت جریان اطلاعات (خواندن و یا نوشتن) و En برای نگهداشتن اطلاعات (Latch) در بافر ورودی/خروجی LCD (حساس به لبهٔ پایین رونده) .

LCD کاراکتری می‌تواند در دو مود انتقال اطلاعات به صورت هشت بیتی و چهار بیتی عمل کند. بورد آموزشی موجود در آزمایشگاه فقط از چهار خط برای ارتباط با LCD استفاده می کند(به شکل موجود در دستور کار آزمایش چهار توجه کنید)

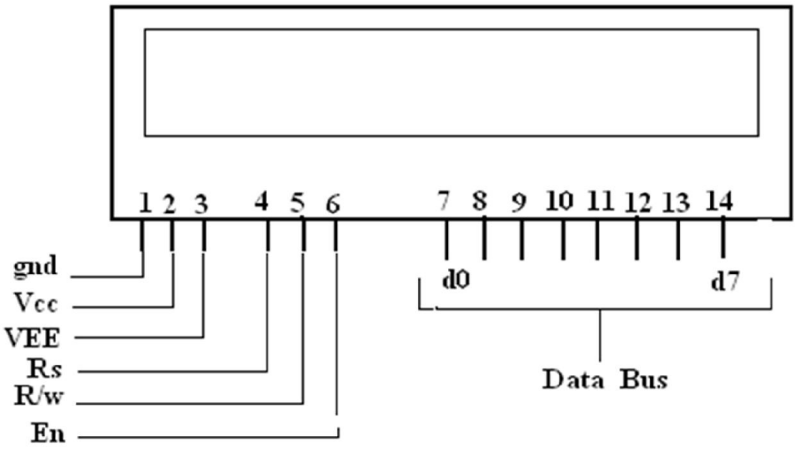
- ارسال فرمان : برای ارسال فرمان به LCD باید خط Rs و Rw را در حالت '0' و خط En را در حالت '1' نگه دارید. سپس اطلاعات ارسالی را روی خط اطلاعات قرارداده و خط En را به حالت '0' برگردانید(لبهٔ پایین رونده). چون در این بورد از مود چهار بیتی استفاده می کنیم مراحل گفته شده در دو مرحله ابتدا برای چهار بیت بالا و سپس برای چهاربیت پایین فرمان انجام شود.
- ارسال کاراکتر نمایشی : این مرحله کاملاً مشابه مرحلهٔ قبل است فقط خط Rs در حالت '1' قرار می گیرد .

برای راه‌اندازی LCD مراحل زیر بترتیب در یک زیر برنامه بنام Inith_Lcd اجرا شود :

۱- فرامین 0x33 ، 0x32 ، 0x28 بترتیب توسط زیر برنامهٔ Lcd_command ارسال شود . این فرامین برای راه اندازی LCD در مود چهار بیتی و فعال شدن هر دو خط نمایشی LCD مورد استفاده قرار می گیرد .

۲- فعال کردن مکان نما (Cursor) با ارسال یکی از فرامین 0x0c (مکان نما خاموش) ، 0x0e (مکان نما بصورت زیر خط) و یا 0x0f (مکان نما بصورت چشمک زن) .

۳- در مرحلهٔ آخر باید یک بار محتویات حافظهٔ LCD پاک شود . برای این منظور فرمان 0x01 را ارسال می‌کنیم .



Rs : برای انتخاب ارسال فرمان یا اطلاعات نمایشی :

Rs= 0(اطلاعات ارسالی بعنوان فرمان محسوب می شود)

Rs=1 (اطلاعات ارسالی بعنوان کاراکتر برای نمایش در نظر گرفته میشود)

R/w : برای انتخاب خواندن و یا نوشتن اطلاعات (R/w=0 حالت نوشتن و R/w=1 حالت خواندن)

En : پایه latch شدن اطلاعات در بافر LCD (حساس به لبه پایین رونده)

جدول فرامین :

INSTRUCTION	Decimal	Hexadecimal
Function set (8-bit interface, 2 lines, 5*7 Pixels)	56	38
Function set (8-bit interface, 1 line, 5*7 Pixels)	48	30
Function set (4-bit interface, 2 lines, 5*7 Pixels)	40	28
Function set (4-bit interface, 1 line, 5*7 Pixels)	32	20
Scroll display one character right (all lines)	28	1E
Scroll display one character left (all lines)	24	18
Home (move cursor to top/left character position)	2	2
Move cursor one character left	16	10
Move cursor one character right	20	14
Turn on visible underline cursor	14	0E
Turn on visible blinking-block cursor	15	0F
Make cursor invisible	12	0C
Blank the display (without clearing)	8	08
Restore the display (with cursor hidden)	12	0C
Clear Screen	1	01
Set cursor position (DDRAM address)	128 + addr	80+ addr
Set pointer in character-generator RAM (CG RAM address)	64 + addr	40+ addr

جدول آدرس LCD :

Characters	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
First line (H)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
Second line(H)	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53
Third line (H)	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27
Fourth line (H)	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67

ASCII TABLE

		MSB							
		0000	0001	0010	0011	0100	0101	0110	0111
LSB	0000	NUL	DLE	SP	0	@	P		p
	0001	SOH	DC1	!	1	A	Q	a	q
	0010	STX	DC2		2	B	R	b	r
	0011	ETX	DC3	#	3	C	S	c	s
	0100	EOT	DC4	$	4	D	T	d	t
	0101	ENQ	NAK	%	5	E	U	e	u
	0110	ACK	SYN	&	6	F	V	f	v
	0111	BEL	ETB		7	G	W	g	w
	1000	BS	CAN	(	8	H	X	h	x
	1001	HT	EM	)	9	I	Y	I	y
	1010	LF	SUB	*	:	J	Z	j	z
	1011	VT	ESC	+	;	K	[	k	{
	1100	FF	FS	,	<	L	\	l	|
	1101	CR	GS		=	M	]	m	}
	1110	SO	RS	.	>	N	^	n	~
	1111	SI	US		?	O	-	o	DEL

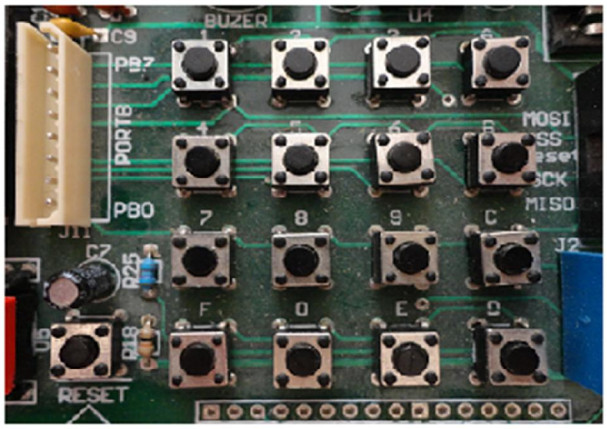
آزمایش پنجم

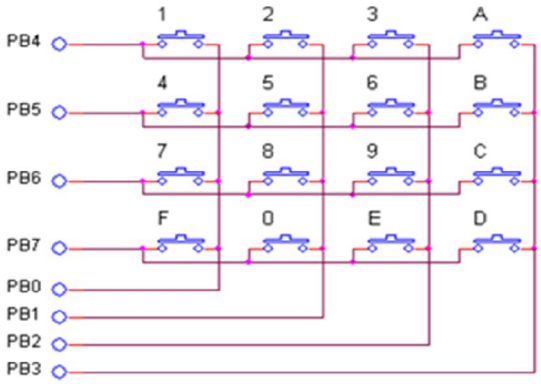
خواندن صفحه کلید ماتریسی

پیش گزارش :

۱- زیر برنامه ای بنویسید که یک شمارندهٔ دسیمال چهار رقمی تولید کند . به این صورت که در هر بار فراخوان زیر برنامه یک واحد به عدد چهار رقمی اضافه شود و سپس از زیر برنامه خارج گردد . رقمهای یکان تا هزارگان این عدد به ترتیب در رجیسترهای R0 ، R1 ، R2 و R3 قرار دارد

۲- زیر برنامه ای بنویسید که صفحه کلید را بخواند اگر کلیدی فشار داده شده بود کد کلید در مبنای شانزده درغیر این صورت مقدار 80H را در رجیستر R16 قرار داده و از زیر برنامه خارج شود .





آزمایش ۱ : شمارش رویداد فشار دادن کلید

با استفاده از زیر برنامهٔ اول برنامه ای بنویسید که تعداد فشار دادن کلید 1 از صفحه کلید شانزده تایی را شمارش نماید . برای خواندن تک کلید ابتدا باید یک پایه از کلید در وضعیت صفر باشد و پایه دیگر به پورتی که در حالت ورودی قرار گرفته متصل شود. بنابراین برای استفاده از کلید '1' باید پورت PB4 را خروجی تعریف کرده و حالت آنرا صفر قرار دهید همچنین پایهٔ PB0 را در وضعیت ورودی قرار داده و مقاومت Pullup آنرا فعال کنید .(فرض کنید که فقط همین یک کلید روی بورد وجود دارد)

آزمایش ۲ : خواندن صفحه ماتریسی

با استفاده از زیر برنامهٔ دوم ، برنامه ای بنویسید کد کلید فشار داده شده از صفحه کلید شانزده تایی در مبنای شانزده روی سون سگمنت نمایش داده شود به صورتیکه با هر بار فشار دادن یک کلید ، کد کلید قبلی روی نمایشگر به سمت چپ منتقل شده و کد کلید جدید روی نمایشگر اول جایگزین گردد .

آزمایش ۳ : نمایش موقعیت کلید روی LCD

برنامه ای بنویسید که با فشار دادن هر کلید از صفحه کلید ماتریسی ، کد کلید زوج روی خط اول LCD و در موقعیت متناسب با عدد کلید و کد کلید فرد روی خط دوم ودر موقعیت متناسب با عدد کلید نمایش داده شود . مثلاً با فشار دادن کلید 8 ، کد مربوطه در موقعیت هشتم از خط اول و یا کلید 3 روی خط دوم و روی موقعیت سوم قرار گیرد . البته محتویات صفحه LCD نباید در نمایش هر کاراکتر پاک شود .

آزمایش ششم

تایمرها

پیش گزارش : برنامهٔ آزمایشهای زیر

آزمایش ۱ : تولید پالس مربعی با استفاده از تایمر یک در مود مقایسه

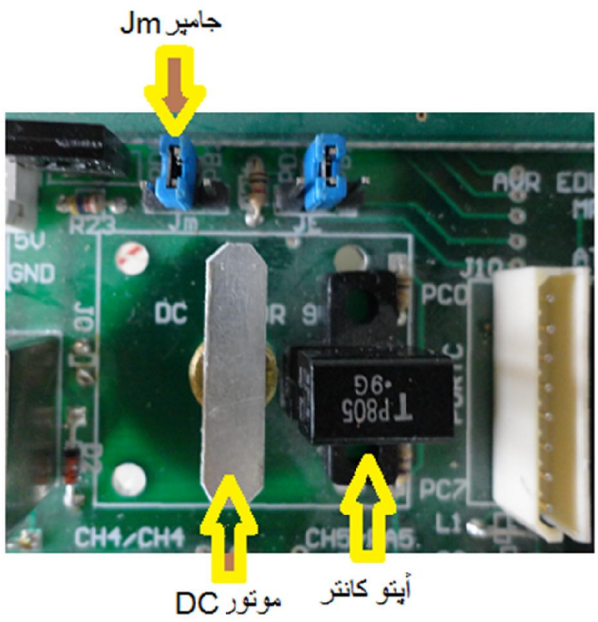
با استفاده از مود مقایسه (CTC) تایمر یک، برنامه‌ای بنویسید که یک پالس مربعی با فرکانس قابل تغییر روی پایه PD.5 تولید کند . مقدار فرکانس توسط کلید هشت‌تایی متصل به PORTB مشخص می‌شود . به ازاء اعداد صفر ، یک ، دو و سه به ترتیب فرکانسهای صد هرتز ، یک ، پنج و ده کیلو هرتز تولید شود . خروجی را توسط اسیلوسکوپ مشاهده کنید.

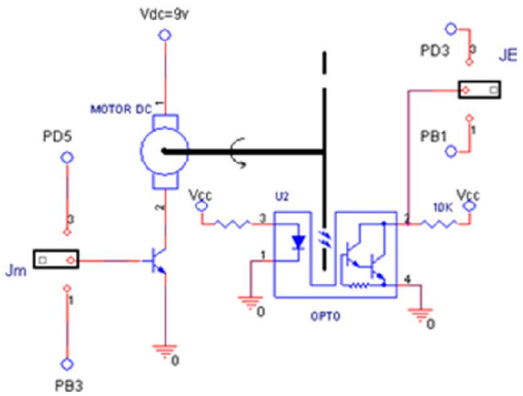
آزمایش ۲ : ساخت کرنومتر

با استفاده از زیر برنامه های آزمایش مربوط به 7_seg و با استفاده از تایمر یک ، یک کرنومتر با دقت صدم ثانیه بسازید . این کرنومتر با فشار دادن کلید 1 از صفحه کلید ماتریسی زمانگیری را شروع می کند و با فشار دادن همین کلید در بار دوم متوقف می شود . مقدار ماکزیمم شمارش 99.99 ثانیه خواهد بود .

آزمایش۳ : کنترل سرعت موتور با روش PWM ، کنترل توسط کلید فشاری

با استفاده از مود PWM تایمر یک و همچنین کلید ۱ و ۲ از صفحه کلید ماتریسی برنامه‌ای بنویسید که بتواند یک پالس مربعی با فرکانس ثابت پنج کیلوهرتز و Duty Cycle متغیر تولید کند .در این برنامه با فشار دادن کلید 1، dutycycle افزایش و با کلید 2 کاهش یابد(هر تغییر ۱۰ واحد) . خروجی را ابتدا توسط اسکوپ مشاهده کنید سپس با قرار دادن جامپر Jm درحالت PD.5 سرعت موتور DC روی بورد را کنترل کنید .

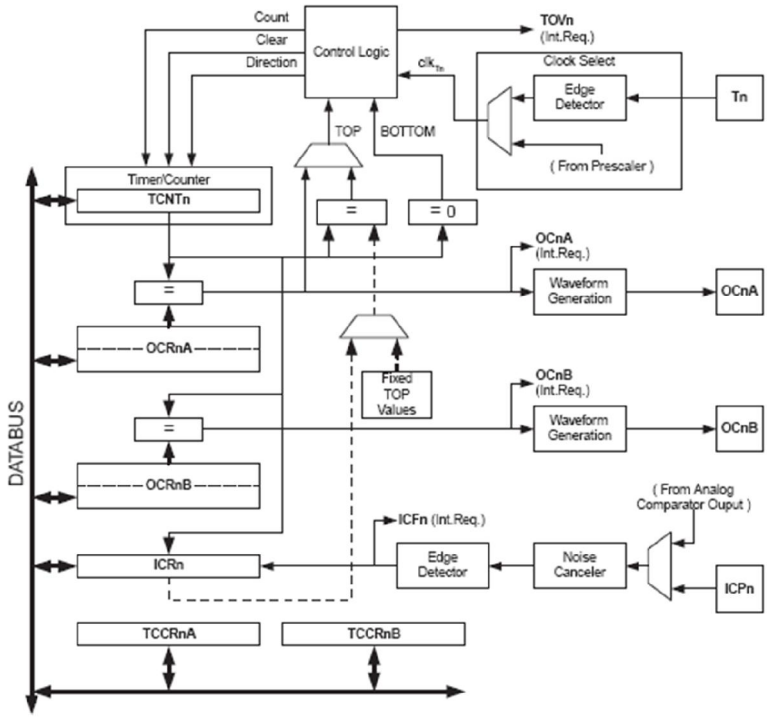




ضمیمهٔ آزمایش ششم : شکل وجداول مربوط به تایمر/کانتر یک

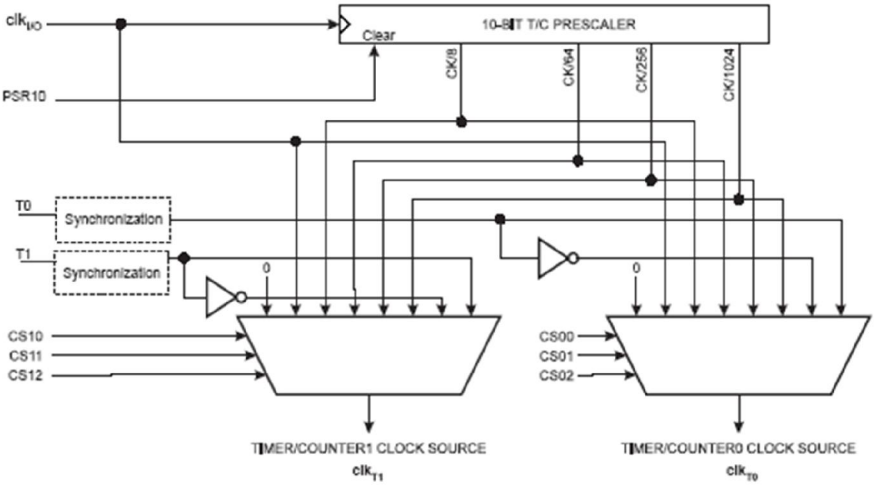
۱- نمای کلی تایمر/کانتر یک

Figure 40. 16-bit Timer/Counter Block Diagram[1]



۲- مقسم فرکانس تایمر/کانتر یک و صفر

Figure 39. Prescaler for Timer/Counter0 and Timer/Counter1[1]



۳- رجیستر اصلی تایمر/کانتر یک

Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۴- رجیستر مقایسه کانال A و B تایمر/کانتر یک

Bit	7	6	5	4	3	2	1	0	
	OCR1A[15:8]								OCR1AH
	OCR1A[7:0]								OCR1AL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	OCR1B[15:8]								OCR1BH
	OCR1B[7:0]								OCR1BL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۵- رجیستر تسخیر تایمر/کانتر یک

Bit	7	6	5	4	3	2	1	0	
	ICR1[15:8]								ICR1H
	ICR1[7:0]								ICR1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۶- رجیسترهای کنترلی تایمر/کانتر یک

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	W	W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۷- رجیستر پرچمهای تایمر/کانترهای صفر ، یک و دو

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۸- رجیستر وقفه TIMSK

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

۹- جداول حالات کاری مختلف تایمر/کانتر یک

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	Reserved	–	–	–
14	1	1	1	0	Fast PWM	ICR1	TOP	TOP
15	1	1	1	1	Fast PWM	OCR1A	TOP	TOP

Table 44. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on compare match
1	0	Clear OC1A/OC1B on compare match (Set output to low level)
1	1	Set OC1A/OC1B on compare match (Set output to high level)

Table 45. Compare Output Mode, Fast PWM(1)

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM13:0 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on compare match, set OC1A/OC1B at TOP
1	1	Set OC1A/OC1B on compare match, clear OC1A/OC1B at TOP

Table 46. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM (1)

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 9 or 14: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM13:0 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on compare match when up-counting. Set OC1A/OC1B on compare match when downcounting.
1	1	Set OC1A/OC1B on compare match when up-counting. Clear OC1A/OC1B on compare match when downcounting.

جدول بردارهای وقفه :

Table 18. Reset and Interrupt Vectors

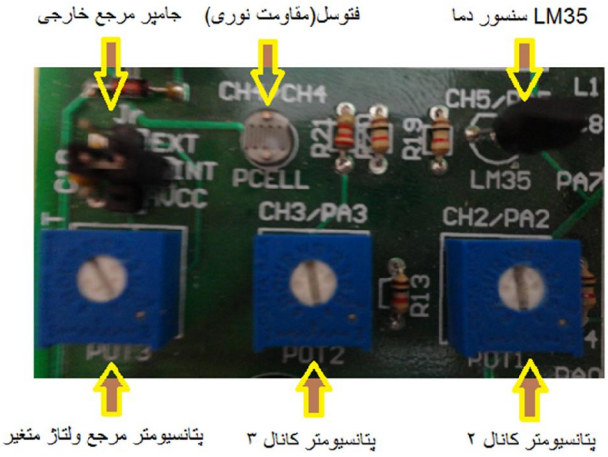
Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	SPM_RDY	Store Program Memory Ready

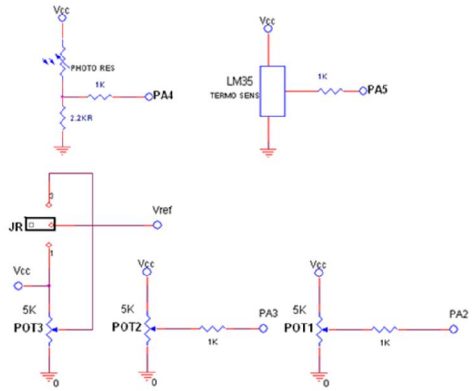
مبدل آنالوگ به دیجیتال

پیش گزارش : برنامهٔ آزمایشهای ۱ و ۲ و ۳ زیر

آزمایش ۱ : ولتمتر با نمایشگر LCD

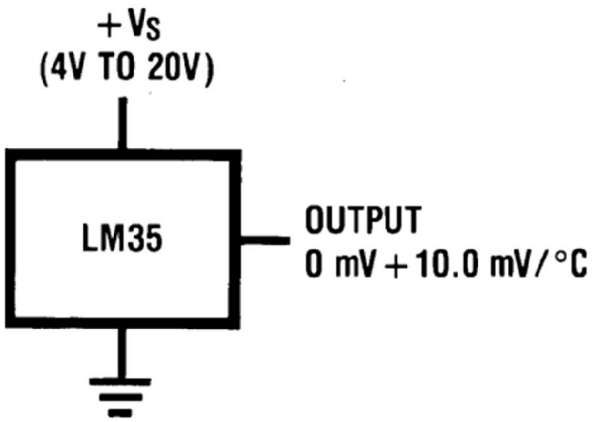
برنامه‌ای بنویسید که مقدار ولتاژ پتانسیومتر متصل به کانال ۲ مبدل آنالوگ به دیجیتال PORTA را با دقت یک دهم اعشار روی نمایشگر LCD نمایش دهد . (مرجع ولتاژ را AVCC (5v) انتخاب کنید)





آزمایش ۲ : دماسنج و ولتمتر روی نمایشگرسون سگمنت

- می‌خواهیم مقادیر خروجی ولتاژ سنسوردما (LM35) بر حسب درجهٔ سانتیگراد و پتانسیومتر POT3 بر حسب ولت و بصورت جداگانه و همزمان روی نمایشگر سون سگمنت نمایش دهیم . بطوریکه دما با دقت یک درجه سانتی گراد روی دو سون سگمنت سمت راست و ولتاژ با دقت یک دهم روی دو سون سگمنت سمت چپ نمایش داده شود. (به راهنمای سنسور LM35 توجه کنید) .



ضمیمهٔ آزمایش هفتم : شکل و جداول کاری قسمت مبدل آنالوگ به دیجیتال

میکرو کنترلر Atmega 16 دارای یک مبدل آنالوگ به دیجیتال هشت کاناله است . این مبدل می تواند مقدار آنالوگ را به مقدار دیجیتال ده بیتی تبدیل کند . کانالهای ورودی مبدل روی پایه های PORTA قرار دارد . برای راه اندازی این مبدل باید مراحل زیر توسط برنامه اجرا شود :

۱- پایهٔ متناظر کانال انتخابی روی PORTA را بعنوان ورودی تعریف کنید . مقاومت Pullup داخلی نباید فعال گردد .

۲- مبدل آنالوگ به دیجیتال را با '1' کردن بیت ADEN از ثبات ADCSRA فعال کنید . همچنین سرعت تبدیل اطلاعات با تعیین فرکانس ورودی مبدل توسط بیتهای صفر تا سه همین ثبات مشخص شود . این مقادیر را از جدول ۲ انتخاب نمایید .

۳- شمارهٔ کانال ورودی در بیتهای صفر تا چهار از ثبات ADMUX نوشته شود . همچنین مرجع ولتاژ توسط بیتهای شش و هفت تعیین گردد . این مقادیر می تواند از جدول ۳ انتخاب شود .این مقدار با توجه ماکزیمم دامنه سیگنال ورودی می تواند AVCC و یا مرجع داخلی 2.56v باشد .

۴- برای شروع عملیات تبدیل می توانید بیت ADSC از ثبات ADCSRA را در حالت '1' قرار دهید سپس منتظر بمانید تا بیت ADIF از همین ثبات در حالت '1' قرار گیرد . اکنون می توانید مقدار تبدیل را از ثبات شانزده بیتی ADC بخوانید .

۵- رجیسترهای کنترلی

Analog to Digital Converter
ADC Control and Status

7	6	5	4	3	2	1	0
ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

Register A – ADCSRA

ADCSRA		
Bit	Name	Function
7	ADEN: ADC Enable	Writing this bit to one enables the ADC
6	ADSC: ADC Start Conversion	write this bit to one to start each conversion
5	ADATE: ADC Auto Trigger Enable	When this bit is written to one, Auto Triggering of the ADC is enabled
4	ADIF: ADC Interrupt Flag	This bit is set when an ADC conversion completes and the Data Registers are updated
3	ADIE: ADC Interrupt Enable	When this bit is written to one and the I-bit in SREG is set, the ADC Conversion Complete Interrupt is activated.
2	ADPS2:0: ADC Prescaler Select Bits	Table : ADC Prescaler Selections

جدول ۱

ADC Prescaler Selections			
ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

جدول ۲

ADC Multiplexer Selection Register – ADMUX

7	6	5	4	3	2	1	0
REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0

Bit	Name	Function
7:6	REFS1:0: Reference Selection Bits	Table : Voltage Reference Selections for ADC
5	ADLAR: ADC Left Adjust Result	The ADLAR bit affects the presentation of the ADC conversion result in the ADC Data Register
4:0	MUX4:0: Analog Channel and Gain Selection Bits	**Table :** Input Channel and Gain Selections

Table : Voltage Reference Selections for ADC		
REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

جدول ۳

Input Channel and Gain Selections				
MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A	N/A	N/A
00001	ADC1	N/A	N/A	N/A
00010	ADC2	N/A	N/A	N/A
00011	ADC3	N/A	N/A	N/A
00100	ADC4	N/A	N/A	N/A
00101	ADC5	N/A	N/A	N/A
00110	ADC6	N/A	N/A	N/A
00111	ADC7	N/A	N/A	N/A
01000	N/A	ADC0	ADC0	10x
01001	N/A	ADC1	ADC0	10x
01010(1)	N/A	ADC0	ADC0	200x
01011(1)	N/A	ADC1	ADC0	200x
01100	N/A	ADC2	ADC2	10x
01101	N/A	ADC3	ADC2	10x
01110(1)	N/A	ADC2	ADC2	200x
01111(1)	N/A	ADC3	ADC2	200x
10000	N/A	ADC0	ADC1	1x
10001	N/A	ADC1	ADC1	1x
10010	N/A	ADC2	ADC1	1x
10011	N/A	ADC3	ADC1	1x
10100	N/A	ADC4	ADC1	1x
10101	N/A	ADC5	ADC1	1x
10110	N/A	ADC6	ADC1	1x
10111	N/A	ADC7	ADC1	1x
11000	N/A	ADC0	ADC2	1x
11001	N/A	ADC1	ADC2	1x
11010	N/A	ADC2	ADC2	1x
11011	N/A	ADC3	ADC2	1x
11100	N/A	ADC4	ADC2	1x
11101	N/A	ADC5	ADC2	1x
11110	1.22 V (V_{BG})	N/A	N/A	N/A
11111	0 V (GND)	N/A	N/A	N/A

جدول ۴

Special FunctionIO Register –
SFIOR

7	6	5	4	3	2	1	0
ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10

Bit	Name	Function
7:5	ADTS2:0: ADC Auto Trigger Source	Table : ADC Auto Trigger Source Selections
4	Res: Reserved Bit	

Table : ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

۶- نحوهٔ قرارگیری دیتا در رجیستر مبدل با توجه به بیت ADLAR در رجیستر ADMUX

The ADC Data Register –
ADCL and ADCH

ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

آزمایش هشتم

پورت سریال Rs232

پیش گزارش :

۱- زیر برنامه‌ای بنویسید که مقدار موجود در رجیستر R16 را از طریق پورت سریال ارسال کند .

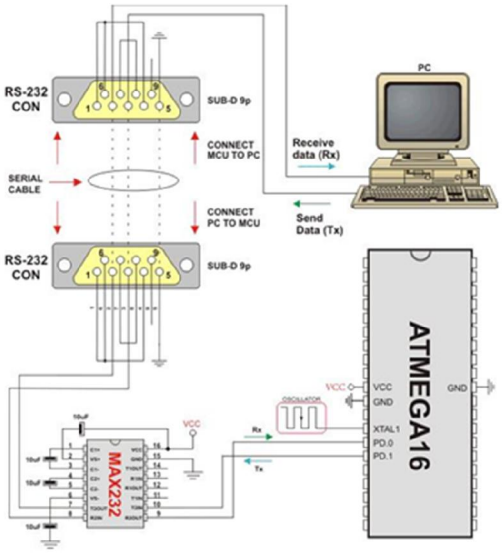
۲- زیر برنامه‌ای بنویسید که مقدار ارسال شده به پورت سریال میکروکنترلر را خوانده و در رجیستر R16 قرار دهد . در صورت نیامدن اطلاعات منتظر بماند .

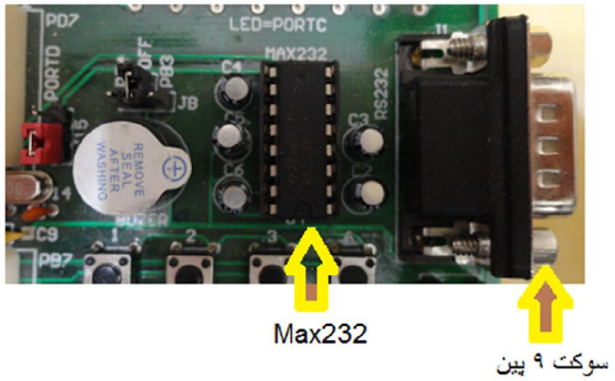
آزمایش ۱ : ارسال و دریافت همزمان اطلاعات از طریق پورت سریال

برنامه‌ای بنویسید که ابتدا عدد صفر را از طریق میکرو به دستگاه جانبی ارسال کند و سپس منتظر دریافت یک مقدار از طرف دستگاه جانبی بماند در صورت دریافت ، آنرا روی 7_seg اول از سمت راست نمایش دهد سپس بعد از مدت یک ثانیه یک واحد به آن اضافه کرده ، آنرا به دستگاه جانبی باز گرداند .

(Baud Rate = 2400)

توجه : در این آزمایش ابتدا کابل ارتباط بین میکرو و دستگاه جانبی را که دارای دو سوکت ۹پین می‌باشد را به سوکت ۹پین موجود روی بورد متصل کنید در طرف دیگر کابل با استفاده از یک سیم مسی پایه های با شمارهٔ ۲ و ۳ را به هم متصل کنید تا در این حالت میکرو هم دریافت کننده و هم ارسال کننده باشد .

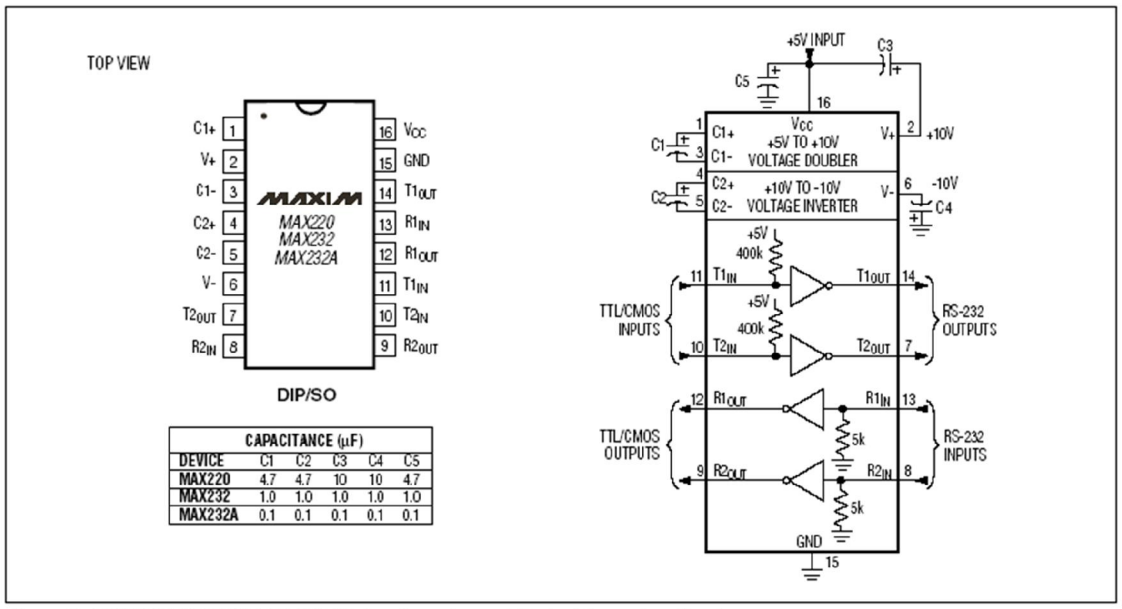




آزمایش ۲ : تبادل اطلاعات بین کامپیوتر و بورد میکروکنترلر

-برنامهٔHyper Terminal موجود در سیستم عامل (Windows(start\allprog\accessories\communications\Hyper،کد آسکی کلید فشار داده شده روی صفحه کلید کامپیوتر را روی پورت سریال کامپیوتر قرار می‌دهد و همچنین می تواند اطلاعات دریافتی از طریق پورت سریال را روی مانیتور نمایش دهد . اکنون با استفاده از این برنامه و برنامهٔ ارتباط سریال که برای میکروکنترلر می‌نویسید می‌خواهیم کد کلید فشار داده شده روی صفحه کلید کامپیوتر را روی LCD و کد کلید فشار داده شده روی صفحه کلید ماتریسی بورد را روی مانیتور کامپیوتر نمایش دهیم . (Baud Rate = 2400)

توجه : مانند شکل قبل سوکت ۹ پایه مربوط به پورت سریال بورد را با استفاده از کابل موجود در آزمایشگاه به COM1 و یا COM2 کامپیوتر متصل کنید .



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

ضمیمهٔ آزمایش هفتم : رجیسترهای ورودی / خروجی و کنترلی USART

۱- رجیستر ورودی/خروجی

Bit	7	6	5	4	3	2	1	0	
	RXB[7:0]								UDR (Read)
	TXB[7:0]								UDR (Write)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The USART Transmit Data Buffer Register and USART Receive Data Buffer Registers share the same I/O address referred to as USART Data Register or UDR. The Transmit Data Buffer Register (TXB) will be the destination for data written to the UDR Register location. Reading the UDR Register location will return the contents of the Receive Data Buffer Register (RXB).

For 5-, 6-, or 7-bit characters the upper unused bits will be ignored by the Transmitter and set to zero by the Receiver.

۲- رجیسترهای کنترلی

Bit	7	6	5	4	3	2	1	0	
	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM	UCSRA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	UCSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	0	0	0	0	1	1	0	

۳- جداول

UMSEL	Mode
0	Asynchronous Operation
1	Synchronous Operation

UPM1	UPM0	Parity Mode
0	0	Disabled
0	1	Reserved
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

USBS	Stop Bit(s)
0	1-bit
1	2-bit

UCSZ2	UCSZ1	UCSZ0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

UCPOL	Transmitted Data Changed (Output of TxD Pin)	Received Data Sampled (Input on RxD Pin)
0	Rising XCK Edge	Falling XCK Edge
1	Falling XCK Edge	Rising XCK Edge

۴- رجیستر تنظیم Buad Rate و جدول محاسبهٔ آن

Bit	15	14	13	12	11	10	9	8	
	URSEL	–	–	–	UBRR[11:8]				UBRRH
	UBRR[7:0]								UBRRL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Table 60. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{OSC}}{16(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{OSC}}{8(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{OSC}}{2(UBRR+1)}$	$UBRR = \frac{f_{OSC}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps).

BAUD Baud rate (in bits per second, bps)

f_{OSC} System Oscillator clock frequency

UBRR Contents of the UBRRH and UBRRL Registers, (0 - 4095)

Table 70. Examples of UBRR Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 8.0000 MHz				f_{osc} = 11.0592 MHz				f_{osc} = 14.7456 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	207	0.2%	416	-0.1%	287	0.0%	575	0.0%	383	0.0%	767	0.0%
4800	103	0.2%	207	0.2%	143	0.0%	287	0.0%	191	0.0%	383	0.0%
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4k	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2k	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8k	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4k	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6k	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8k	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2k	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4k	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250k	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	0	0.0%	1	0.0%	–	–	2	-7.8%	1	-7.8%	3	-7.8%
1M	–	–	0	0.0%	–	–	–	–	0	-7.8%	1	-7.8%
Max [1]	0.5 Mbps		1 Mbps		691.2 kbps		1.3824 Mbps		921.6 kbps		1.8432 Mbps	

1. UBRR = 0, Error = 0.0%