

آموزش میکرو کنترلرهای AVR به زبان بیسیک
و
آموزش کار با شبیه ساز پروتیوس

نویسنده و گردآورنده: پیمان خدّامی

WRITER AND PRODUCER: PEYMAN KHODDAMI

B | A | S | C | O | M

PEYMAN KHODDAMI

فهرست : شماره صفحه

مقدمه: 5

فصل اول (آشنایی مختصر با محیط بسکام)

آشنایی مختصر با محیط بسکام: 7

منوی file 7

منوی edit 8

منوی View 8

منوی program 9

اصلا حات پر کاربرد 7

فصل دوم (آشنایی مختصر با بیسیک و کار با lcd و پورت ها)

مراحل نوشتن یک برنامه جدید(بدنه ی یک برنامه): 11

دستورات مربوط به پورت ها (کار با پورتها) 15

دستورات تاخیر 18

دستورات حلقه و پرش 18

Lcd کاراکتری(دستورات مربوط به راه اندازی ، فارسی نویسی و...) 23

فصل سوم (معرفی سایر دستورات بیسیک)

اعداد و متغییر ها در بسکام 33

دستورات مربوط به کار با رشته ها 45

دستورات حلقه و پرش و شرط 55

دستورات اجرایی (این دستورات ، دستورات خاص برای کامپایلر هستند که برای اجرای بهتر برنامه استفاده میشوند) 68

زیر برنامه ها و فراخوانی توابع 74

توابع ریاضی و محاسباتی 79

PEYMAN KHODDAMI

- 94-----توابع تبدیل کدها و متغییر ها به یکدیگر-----
فصل چهارم (راه اندازی امکانات جانبی)
- 97-----دستور debounce(اتصال کلید به میکرو)-----
98-----دستور PULSEOUT(ایجاد یک پالس بر روی یک پایه)-----
99-----دستور PULSEIN(اندازی گیری دوره ی تناوب پالس موجود بر روی یک پایه)-----
99-----دستور SOUND(ایجاد پالس برای راه اندازی بازر بر روی یک پایه)-----
100-----دستور ENCODER (ENCODER نوعی کلید دوطرفه میباشدکه..)-----
102-----دستور DTMFOUT (ایجاد پالس شماره گیری(تلفن)با avr)-----
105-----راه اندازی magnetic card (فقط کافی است شما ان را از شکاف...)-----
122-----LCD گرافیکی(بر روی این نوع lcd میتوان تصاویر ، متن و... را نمایش داد و...)-----
129-----استفاده از کلید وکیبرد و کی پد و ... (روشهای راه اندازی کیبرد، کیبرد کامپیوتر و کلید فشاری و...)-----
126-----اتصال avr به عنوان کیبرد به کامپیوتر(چگونه یک کیبرد بسازیم.....)-----
131-----اتصال avr به عنوان موس به کامپیوتر—چگونه یک موس بسازیم)-----
134-----مبدل آنالوگ به دیجیتال(adc)(برای تبدیل کمیت های آنالوگ به دیجیتال از این مورد استفاده میشود)-----
137-----راه اندازی سروو موتور (نوعی موتور پر قدرت است که میتواند در یک زاویه خاص بچرخد...)-----
140-----راه اندازی WATCHDOG (تایمری است که میتواند تا یک زمان خاص بشمارد و میکرو را ریست کند ، این تایمر...)-----
141-----راه اندازی وقفه های خارجی-(چگونه یک پایه همیشه چک شود)-----
143-----راه اندازی گیرنده rc5(گیرنده و فرستنده های مادون قرمز که انها را با نام تجاری گیرنده و فرستنده های rc5 میشناسند رواج فوق ...)-----
147-----ساخت کنترل تلویزیون و سیدی sony (توسط دستور زیر میتوان دستورات مخصوص کنترل تلوزیون و cd سونی ...)-----
150-----راه اندازی گیرنده RC6 (این پروتکل ، برخلاف RC5 در اکثر دستگاههای صوتی تصویری جدید(تمامی دستگاههای CD چینی را پشتیبانی...)-----
152-----اندازه گیری یک خازن یا مقاومت(شما با استفاده از دستور زیر میتوانید مقدار ثابت زمانی مقاومت و خازنی که به پایه دلخواه میکرو...)-----
153-----مقایسه کننده آنالوگ(مقایسه کننده آنالوگ مقادیر ولتاژ آنالوگ موجود بر روی دو پایه خود را)-----
154-----تایمر/ کانتر(تایمر کانتر چیست...؟)-----
154-----تایمر/ کانتر 0 (راه اندازی تایمرکانتر صفر در مد تایمر – کانتر ...)-----
158-----تایمر کانتری یک (راه اندازی در مد تایمر –کانتر –pwm مقایسه ای و...)-----
176-----راه اندازی تایمر/کانتر دو (راه اندازی در مد تایمر –کانتر –pwm مقایسه ای و...)-----

PEYMAN KHODDAMI

- 187-----راه اندازی تایمر/کانتر سه(راه اندازی در مد تایمر -کانتر -pwm مقایسه ای و...)
- 202-----rtc (Real Time Counter) (شمارش گر زمان واقعی-)
- 211-----ارتباط سریال rs232 (اتصال دو میکرو از طریق دو سیم...)
- 217-----ارتباط سریال spi (ارتباط چند میکرو از طریق 4 سیم ...)
- 224-----ارتباط سریال i2c یا 2-wire(اتصال چند میکرو از طریق دو سیم...)
- 231-----ارتباط سریال 1 WIRE(اتصال چند میکرو از طریق 1 سیم..)
- 234-----کار با حافظه داخلی میکرو (eeprom):

ضمائم:

- 236-----ضمیمه 1 : طریقه ی نصب بسکام-----
- 240-----ضمیمه 2 : آشنایی با محیط شبیه سازی بسکام(simulate)-----
- 246-----ضمیمه 3 : شبیه سازی میکرو کنترل avr با برنامه پروتوس (آشنایی مقدماتی)-----
- 253-----ضمیمه 4 : پروگرام کردن میکرو : (معرفی منوی send to chip)-----
- 258-----ضمیمه 5 : راه اندازی انواع موتور ها ی dc و پله ای بوسیله ی avr و زبان بیسیک-----
- 269-----ضمیمه 6: راه اندازی 7 سگمنت(این قطعات از 8 led تشکیل می شود که 7تا از آنها نمایشگر هستند ویکی...)
- 283-----ضمیمه 7: خطا های بسکام-----
- 287-----ضمیمه 8 : کلید های میانبر در بسکام-----
- 288-----ضمیمه 9 : اندازه گیری ولتاژ های منفی زیاد مثبت و...با ADC (کار با OP-AMP)-----

PEYMAN KHODDAMI

مقدمه :

سلام

این آموزش از هفت بخش اصلی زیر تشکیل شده است:

- 1- آشنایی با محیط نرم افزار بسکام
 - 2- آشنایی مختصر با بیسیک و کار با lcd و پورت ها
 - 3- معرفی سایر دستورات بیسیک
 - 4- راه اندازی لوازم جانبی avr
 - 5- ضmann (یا ضمیمه ها)
- در بخش اول شما با محیط بسکام آشنا میشوید، شما یاد میگیرید در این محیط چگونه با پروژه ها کار کنید ، آنها را ویرایش کنید و ایراد های احتمالی را چگونه رفع کنید.
- در بخش بعدی طریقه کار با lcd و طریقه کار با پورت ها گفته میشود تا شما بتوانید با مثالهای بخش های بعدی کار کنید.
- در بخش سوم سایر دستورات بیسیک گفته میشود.
- در بخش چهارم طریقه کار با امکانات جانبی خارجی (مانند lcd گرافیکی ،...) و امکانات جانبی داخلی (مانند adc و...) گفته میشود.
- در بخش پنجم ضmannی برای درک بهتر مطالب آورده شده است ، این ضmann شامل : آموزش مقدماتی پروتوس ، نحوه پروگم کردن میکرو ، ... میباشد.
- شما برای یاد گیری avr به دو نرم افزار بسکام avr (bascom avr) و نرم افزار شبیه سازی پروتوس (Proteus) نیاز دارید ، که میتوانید آنها را از طریق اینترنت تهیه کنید(در قسمت ضmann نحوه شبیه سازی میکرو کنترلر ها با پروتوس گفته شده است).

اما در صورت بروز هرگونه مشکل در رابطه با این کتاب و یا در میان گذاشتن نظرات، پیشنهادات و انتقادات، می توانید از طریق زیر با من در ارتباط باشید:

<http://PEYMANKHODDAMI.IR>

khoddamipeyman@gmail.com

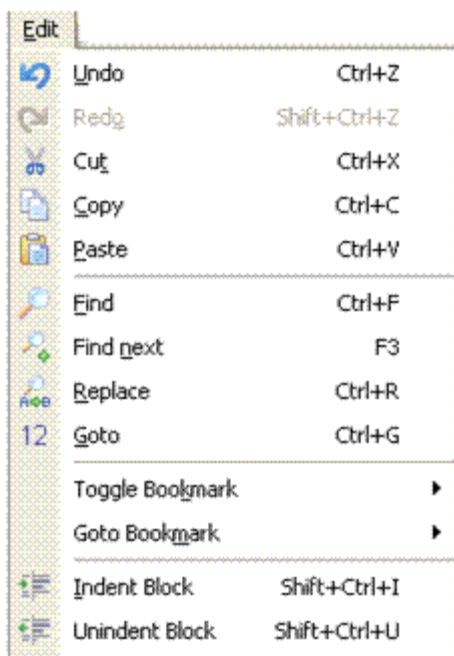
PEYMAN KHODDAMI

Avr ها میکرو کنترلر های 8 بیتی ساخت شرکت اتمل میباشند . این میکرو کنترلر ها بیشترین سهم بازار ایران را بخود اختصاص داده اند . از جمله مزایای این میکرو ها میتوان به موارد زیر اشاره کرد:

- 1- کارای بالا و توان مصرفی کم
- 2- دارای 100 تا 140 دستور که اکثر آنها در یک سیکل اجرا میشوند
- 3- قابلیت کار در فرکانس 1 تا 20 مگا هرتز (به جز کریستال ساعت)
- 4- حافظه ی eeprom و فلش و sram با دفعات نوشتن بالا.
- 5- چندین تایمر /کانتر
- 6- چندین کانال خروجی pwm
- 7- چندین کانال مبدل آنالوگ به دیجیتال 10 بیتی.
- 8- دارای rtc (نوعی ساعت است که زمان و تاریخ را مستقل از عملکرد میکرو محاسبه میکند)با اسیلاتور مجزا
- 9- مقایسه کننده آنالوگ داخلی
- 10- Usart قابل برنامه ریزی
- 11- Watchdog قابل برنامه ریزی با اسیلاتور داخلی
- 12-ارتباط سریال isp برای برنامه ریزی (پروگرام کردن)داخل مدار (هنگامی که میکرو داخل مدار است با پروگرامر isp میتوانید میکرو را برنامه ریزی کنید ،برای برنامه ریزی از چهار خط miso و mosi و sck و reset استفاده میشود)
- 13-قابلیت ارتباط سریال isp به صورت master یا slave
- 14-قابلیت ارتباط jtag (یک نوع ارتباط است که از طریق ان می توان کلیه حافظه های قابل برنامه ریزی میکرو را خواند یا نوشت)
- 15-Reset شدن میکرو بعد از روشن شدن
- 16-دارای چندین مد بیکاری
- 17-چندین منبع تولید پالس
- 18-نوسانساز داخلی
- 19-تنوع در امکانات ، بسته بندی و قیمت و....
- 20-و دیگر موارد که این میکرو را متمایز میسازد ، شما میتواند دیگر اطلاعات را از دیتا شیت ایسی استخراج کنید.

PEYMAN KHODDAMI

<< منوی Edit :



که دارای گزینه های زیر است:

1-Undo و Redo: این دو گزینه برای دست یابی به آخرین تغییرات انجام شده می باشد.

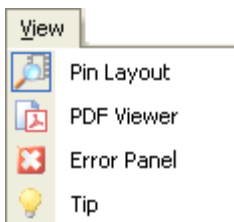
2-Copy و Paste و Cut: این سه گزینه برای برداشتن یا کپی کردن قسمتی از متن به جای دیگر میباشد.

3-Find و Findnext: این دو گزینه برای پیدا کردن قسمتی از متن در برنامه می باشد. نحوه کار به این صورت است که بعد از انتخاب گزینه Find ، پنجره جدیدی باز می شود که باید در قسمت Text to find مورد نظر را تایپ کنید. بعد روی ok کلیک کنید تا متن مورد نظر در برنامه انتخاب شود. Findnext. متن های که در خط های بعدی برنامه وجود دارد پیدا میکند.

4-replace: با این گزینه شما می توانید متنی را جایگزین متن موجود در برنامه نمایید، یعنی در قسمت text to find متن یا کلمه مورد جستجو که باید توسط متن یا کلمه دیگری جایگزین شود را تایپ کنید و در قسمت replace with متنی را که باید جایگزین شود تایپ می کنیم .

5-دو گزینه بعدی برای گذاشتن علامت در خطوط مختلف و پرش به آنها می باشد.

5-IndentBlock و UnindentBlock: این دو گزینه متن انتخاب شده را به اندازه يك tab به چپ یا راست منتقل میکند.



<< منوی view:

1-Pin layout: با انتخاب این گزینه پنجره ای باز میشود ، که در آن پایه های میکرو مورد استفاده و شکل آن آورده شده است ، با بردن موس روی هر پایه نقش آن در زیر تصویر نوشته میشود.

2-pdf viewer: با انتخاب این گزینه شما میتوانید مشخصات میکرو مورد استفاده را ببینید (شما ابتدا باید آن ها را دانلود کنید ، با کلیک روی گزینه نحوه دانلود به شما گفته میشود ، شما همچنین به برنامه Adobe Acrobat برای باز کردن pdf ها نیاز دارید.)

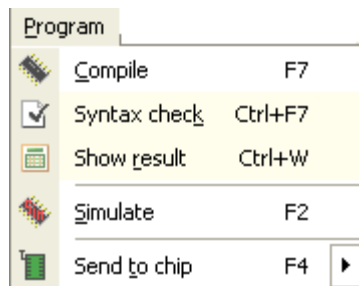
PEYMAN KHODDAMI

3-error panel: با انتخاب این گزینه پنجره ی خطاها باز میشود ، این پنجره بعد از کامپایل کردن برنامه ، در صورت وجود خطا خودکار باز میشود.

4-tip: با زدن این گزینه پنجره ای باز میشود که حاوی نکاتی برای بهتر کار کردن با بسکام است.

<< منوی Program :

که دارای گزینه های زیر است:



1-Compile: با انتخاب این گزینه برنامه نوشته شده به زبان ماشین ترجمه میشود و فایل های از قبیل هگز و گزارش و...ساخته میشود. اگر در این مرحله برنامه دارای خطا باشد پنجره ای باز میشود که در آن خطاها نمایش داده می شوند ؛ با کلیک کردن روی هر خطا ، خط مربوط که دارای خطا است قرمز میشود .

2-Syntax check: با انتخاب این گزینه برنامه از نظر غلط املايي چك میشود (با زدن گزینه Compile دیگر نیازی به زدن این گزینه نمی باشد).

3-Show result: با انتخاب این گزینه پنجره های باز میشود که در آن گزارش کلی از برنامه وجود دارد.

4-Simulate: با انتخاب این گزینه پنجره شبیه سازی باز میشود و شما در این پنجره که دارای lcd و کیبورد و مبدل انالوگ به دیجیتال و...میباشد می توانید برنامه خود را شبیه سازی کنید (برای دریافت اطلاعات بیشتر به ضمائم مراجعه کنید).

5-send to chip: با انتخاب این گزینه وارد محیط پروگرام کردن میکرو می شوید که در قسمت ضمائم مفصلا توضیح داده شده است .

<< منوی های tools و options : با این دو منو در دیگر فصل ها آشنا میشویم.

تا اینجا شما یاد گرفتید که یک برنامه را در بسکام بنویسید و آن را به کد هگز تبدیل کنید. اکنون به ضمیمه 3 بروید و نحوه کار با پروتوس را ببینید، سپس ادامه را بخوانید.

<< در زیر به بیان برخی از عباراتی که از این به بعد به کار میرود می پردازیم .

1-vcc و: gnd منظور از این دو کلمه پایه های تغذیه می باشد که معمولا vcc، 5ولت می باشد و 0 gnd ولت است

شما می توانید این تغذیه را از پورت usb کامپیوتر خود بگیرید .

PEYMAN KHODDAMI

2- پورت : هر میکرو دارای چندین پورت می باشد و هر پورت دارای چندین پایه است (معمولا 8 پایه) برای مثال میکرو مگا 32 (atmega32) دارای 4 پورت a,b,c,d می باشد که هر پورت 8 پایه دارد و پورت a (porta) از پایه 33 تا 40 میکرو می باشد و پایه شماره 40 بین a.0 (pina.0) و پایه 39 بین a.1 (pina.1)... و پایه 33 بین a.7 (pina.7) می باشد (برای دیدن سایر اطلاعات میکروها به دیتاشیت آنها مراجعه کنید)

3- پین (pin): به هر پایه های میکرو پین نیز گفته میشود . مثلا پین a.1 ، یعنی پایه شماره یک پورت a ، که در میکرو مگا 16 پایه شماره 39 میباشد.

4- کریستال: میکرو برای تنظیم زمان برای انجام کارها به یک نوسان ساز نیاز دارد که به این نوسان ساز کریستال گفته می شود حداکثر کریستال مورد استفاده برای avr ، 16 مگاهرتز می باشد. همچنین میکرو های avr دارای نوسان ساز داخلی می باشد ، که در صورت نیاز می توانید از آن استفاده کنید. کریستال به دو پایه 1 xtal و 2 xtal متصل می شود ، این پایه ها برای میکرو مگا 16 پایه های 12 و 13 می باشد

5- پین های avr دارای چندین نقش میباشد ، مثلا پین a.0 (pina.0) (پایه شماره 40) در مگا 16 علاوه بر اینکه میتواند به عنوان ورودی یا خروجی استفاده شود میتواند به عنوان ورودی مبدل آنالوگ به دیجیتال استفاده گردد. شما نمیتوانید در یک زمان از دو نقش یک پین یا پورت استفاده کنید . برای فهمیدن نقش دیگر پین ها به دیتاشیت آنها مراجعه نمایید.

6- رجیستر حافظه : مکان های از حافظه ی میکرو میباشد که اعداد و متغیر ها در آنجا ذخیره میشوند.

7- ادرس حافظه : برای ساده گی کار با حافظه ها ، آن ها را به بخش های کوچکی تقسیم میکنند ، شما میتوانید این خانه ها را ادرس دهی کنید و متغیر یا ثابت های مورد نظر را ذخیره کنید.

PEYMAN KHODDAMI

(معرفی میکروی تاینی دوازده ، Attiny 12)	Attiny12.dat
(معرفی میکروی تاینی پانزده ، Attiny 15)	Attiny15.dat
(معرفی میکروی تاینی بیست و دو ، Attiny 22)	Attiny22.dat
(معرفی میکروی Attiny 24)	Attiny24.dat
(معرفی میکروی Attiny 25)	Attiny25.dat
(معرفی میکروی Attiny44)	Attiny44.dat
(معرفی میکروی Attiny 45)	Attiny45.dat
(معرفی میکروی Attiny 461)	Attiny461.dat
(معرفی میکروی Attiny 84)	Attiny84.dat
(معرفی میکروی Attiny85)	Attiny85.dat
(معرفی میکروی Attiny861)	Attiny861.dat
	سری at90s
(معرفی میکروی AT90S1200)	1200def.dat
(معرفی میکروی AT90S2323)	2323def.dat
(معرفی میکروی AT90S2333)	2333def.dat
(معرفی میکروی AT90S2343)	2343def.dat
(معرفی میکروی AT90S4414)	4414def.dat
(معرفی میکروی AT90S4433)	4433def.dat
(معرفی میکروی AT90S4434)	4434def.dat
(معرفی میکروی AT90S8515)	8515def.dat
(معرفی میکروی AT90S8535)	8535def.dat
	سری atmega
(معرفی میکروی ATMEGA8535)	M8535.dat
(معرفی میکروی ATMEGA8515)	M8515.dat
(معرفی میکروی ATMEGA8)	M8def.dat

PEYMAN KHODDAMI

(معرفی میکروی ATMEGA103)	M103def.dat
(معرفی میکروی ATMEGA16)	M16def.dat
(معرفی میکروی ATMEGA163)	M163def.dat
(معرفی میکروی ATMEGA162)	M162def.dat
(معرفی میکروی ATMEGA164)	M164pdef.dat
(معرفی میکروی ATMEGA8535)	M165def.dat
(معرفی میکروی ATMEGA168)	M168def.dat
(معرفی میکروی ATMEGA169)	M169def.dat
(معرفی میکروی ATMEGA161)	M161def.dat
(معرفی میکروی ATMEGA32)	M32def.dat
(معرفی میکروی ATMEGA8535)	M323def.dat
(معرفی میکروی ATMEGA64)	M64def.dat
(معرفی میکروی ATMEGA128)	M128def.dat
(معرفی میکروی ATMEGA128103)	M128103def.dat
(معرفی میکروی ATMEGA1281)	M1281can.dat
(معرفی میکروی ATMEGA2560)	M2560def.dat
(معرفی میکروی ATMEGA2561)	M2561def.dat
(معرفی میکروی ATMEGA324)	M324def.dat
(معرفی میکروی ATMEGA325)	M325def.dat
(معرفی میکروی ATMEGA329)	M329def.dat
(معرفی میکروی ATMEGA406)	M406def.dat
(معرفی میکروی ATMEGA48)	M48def.dat
(معرفی میکروی ATMEGA603)	M603def.dat
(معرفی میکروی ATMEGA644)	M644def.dat
(معرفی میکروی ATMEGA649)	M649def.dat

PEYMAN KHODDAMI

(معرفی میکرو ATMEGA88)

M88def.dat

خط بعدی معرفی کریستال می باشد :

```
$crystal=x
```

که x کریستال مورد استفاده بر حسب هرتز است مانند:

$\$crystal=8000000$ (در اینجا کریستال 8 مگا هرتز است) .

شما همچنین میتوانید مقدار کرسنال را بر یک عدد تقسیم کنید (این کار برای مواردی که به یک کریستال دسترسی ندارید است)

```
CONFIG CLOCKDIV = constant
```

که constant برابر این مقادیر است: 1, 2, 4, 8, 16, 32, 64, 128, 256.

```
$crystal=8000000
```

```
CONFIG CLOCKDIV = 8
```

از این به بعد میکرو با فرکانس 1 مگا هرتز کار میکند

بعد از معرفی کریستال نوبت به معرفی امکانات می باشد (امکانات شامل تایمر ها و adc (مبدل آنالوگ به دیجیتال) و ورودی یا خروجی قرار دادن پورت ها و... می باشد) .

معرفی امکانات با دستور زیر شروع میشود می باشد :

```
Config
```

مانند

```
Config lcd =16*2
```

بعد از معرفی یا پیکر بندی امکانات جانبی نوبت به استفاده از آنها می باشد معمولا برای استفاده از امکانات باید ان را درون یک حلقه قرار میدهند .

و در نهایت برنامه با end به پایان می رسد.

همیشه چارچوب یک برنامه به زبان بیسیک برای avrمانند بالا میباشد.

PEYMAN KHODDAMI

<<دستورات مربوط به ورودی و خروجی ها: (انجام عملیات روی پایه ها)

در بسکام برای استفاده از یک پورت باید آن را به صورت ورودی یا خروجی قرار داد:

یک پورت هنگامی به عنوان خروجی تعریف میشود که بخواهیم از آن ولتاژ بگیریم و یک پورت هنگامی به عنوان ورودی قرار میگیرد که بخواهیم به آن ولتاژ بدهیم. (دادن ولتاژ را به منزله کلیدی که یک سر آن به 5 ولت است و گرفتن ولتاژ را به منزله led که یک سر آن به گراند متصل است در نظر بگیرید)

برای قرار دادن یک پورت به عنوان ورودی از دستور زیر استفاده میکنیم:

```
Config portx = input
```

که پورت x یکی از پورت های میکرو می باشد.

و برای قرار دادن یک پورت به عنوان خروجی از دستور زیر استفاده میکنیم:

```
Config portx = output
```

که پورت x یکی از پورت های میکرو می باشد. مانند:

Config porta = output پورت a به عنوان خروجی تعریف شده است.

Config portb=input پورت b به عنوان ورودی تعریف شده است.

همچنین شما می توانید یکی از پایه های پورت را به عنوان ورودی یا خروجی معرفی کنید. مانند:

Config pina.1=input پایه شماره 1 از پورت a به عنوان ورودی تعریف شده است.

Config pinb.7=input پایه شماره 7 از پورت b به عنوان ورودی تعریف شده است.

Config pinc.5=output پایه شماره 5 از پورت c به عنوان خروجی تعریف شده است.

دستورات مربوط به پورت ها:

دستور toggle:

این دستور یک بایت را برعکس میکند، بایت میتواند یک پورت یا هر چیز دیگر باشد.

دستور set:

این دستور یک بیت را یک میکند، بیت میتواند یک پین از پورت یا هر چیز دیگر باشد.

PEYMAN KHODDAMI

دستور reset :

این دستور یک بیت را صفر میکند، بیت میتواند یک پین از پورت یا هر چیز دیگر باشد.

<دستور ALIAS :

از این دستور برای تغییر نام متغیر استفاده می شود . مانند:

```
DIM Q AS BIT
```

```
Q ALIAS PORTB.1
```

حال شما می توانید در برنامه، بجای PORTB.1 از متغیر Q استفاده نمایید . (متغیرها در بخش سوم معرفی میشود)

```
SET Q 'is equal with SET PORTB.1
```

دستور BITWAIT:

با این دستور cpu میکرو مدام یک پین را چک میکند ، هنگامی که پایه 1 یا 0 شد (صفر یا یک بودن در برنامه مشخص میشود)

دستورات زیر خط برنامه اجرا میشوند، این دستور به فرم کلی زیر است:

```
BITWAIT x , SET/RESET
```

X: نام پایه است که قرار است چک شود، مثل porta.0 یا portb.7 یا...

SET/RESET: در صورت گذاشتن set ، هنگامی که پایه 1 شد برنامه های زیر دستور اجرا میشود و در صورت گذاشتن reset هنگامی که پایه 0 شد دستورات زیر خط اجرا میشوند.مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Config Portb.7 = Input
```

```
Config Portb.6 = Output
```

```
Bitwait Pinb.7 , Reset
```

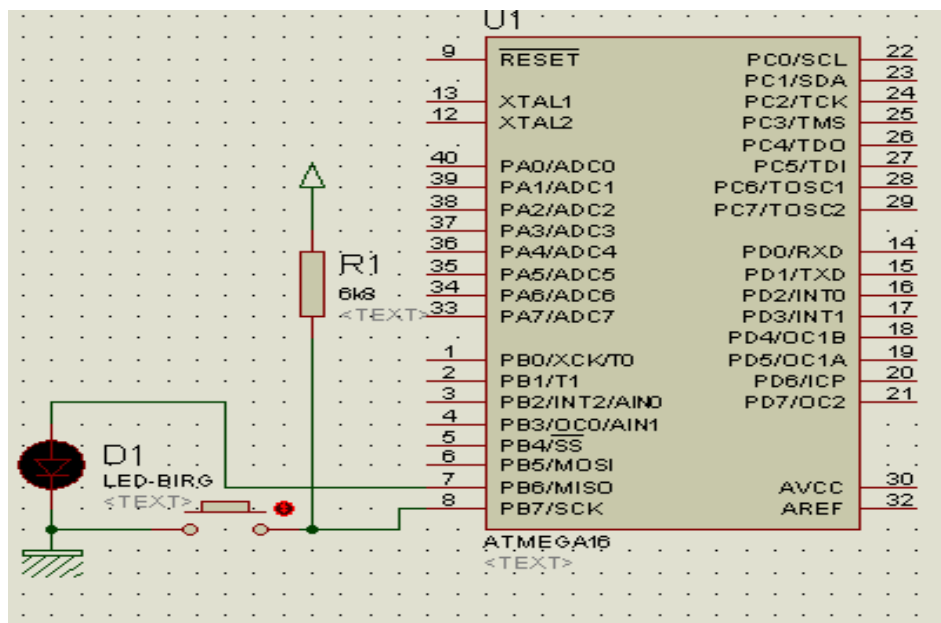
```
Set Portb.6
```

```
Bitwait Pinb.7 , Set
```


PEYMAN KHODDAMI

Reset Portb.6

End



در مثال بالا همانگونه که مشاهده میفرمایید از میکرو مگا 16 ، کلید و led استفاده شده است . دوخط اول برنامه معرفی میکرو و کریستال میباشد ، میکرو مگا 16 و کریستال 8 مگا هرتز است (از کریستال داخلی میکرو استفاده شده است) در خط سوم پورت b.7 (پایه 8 میکرو) به عنوان ورودی تعریف شده است و کلید به آن متصل گردیده ، در خط چهارم پایه b.6 (پایه 7 میکرو) به عنوان خروجی تعریف شده است و led به آن متصل گردیده ، در خط پنجم توسط دستور Bitwait پایه b.7 چک میشود و هنگامی که این پایه ریست شده (به زمین متصل شد ، صفر شد) میکرو به خط بعدی میرود (توجه داشته باشید که cpu روی دستور Bitwait قفل میشود و تا زمانی که شرط (0 یا یک شدن پایه) برقرار نباشد این قفل شدن ادامه دارد) در خط بعدی یا خط ششم ، پایه b.6 یک میشود و led روشن میشود ، در خط بعدی دو باره پایه b.7 چک میشود ، این بار شرط برعکس حالت قبل است ، یعنی اگر پایه 1 شود cpu میکرو به خط بعدی پرش میکند و led را خاموش میکند.و برنامه با دستور end پایان میابد.

وظیفه مقاومت در اینجا بالا نگه داشتن پایه b.7 میباشد (سطح 1 را تصحیح میکند ، مثال را بدون مقاومت اجرا کنید تا..)

PEYMAN KHODDAMI

<< دستورات تاخیر:

برای ایجاد تاخیر در برنامه از دستور wait استفاده میشود.

دستور wait به سه شکل زیر است:

Waitus x این دستور برای ایجاد تاخیر میکرو ثانیه ای می باشد. x مقدار تاخیر میباشد که بین 1 تا 65535 میکرو ثانیه می باشد. مانند

Waitus 500 تاخیر به مدت 500 میکرو ثانیه .

Waitms x این دستور برای ایجاد تاخیر میلی ثانیه ای می باشد. x مقدار تاخیر میباشد که بین 1 تا 65535 میلی ثانیه می باشد. مانند

Waitms 720 تاخیر به مدت 720 میلی ثانیه .

Wait x این دستور برای ایجاد تاخیر میلی ثانیه ای می باشد. x مقدار تاخیر میباشد که عددی بیشتر از یک ثانیه می باشد. مانند

Wait 1000 تاخیر به مدت 1000 ثانیه

دستور DELAY :

این دستور در هر جا که استفاده شود یک تاخیر 1 میلی ثانیه ایجاد می شود

توجه کنید که هر جا دستور wait به کار رود برنامه در آنجا به اندازه زمان مورد نظر متوقف می شود

<< دستورات حلقه و پرش :

گاهی اوقات نیاز است برنامه مدام اجرا شود یا در هنگام اجرای برنامه از یک خط به خط دیگری پرش شود.

برای این کار از حلقه ها و دستورات پرش استفاده میشود، من در اینجا do – loop را معرفی میکنم ، بقیه حلقه ها در بخش بعدی موجود است ،

شروع این حلقه با do و پایان آن با loop است

برای پرش از یک قسمت برنامه به قسمت دیگر می توان از دستور jmp یا goto استفاده کرد (از دستورات فوق به عنوان حلقه نیز می توان استفاده کرد) . مانند:

PEYMAN KHODDAMI

Q :

برنامه نوشته شده

Jump q

مثال دوم:

W :

برنامه نوشته شده

Goto w

حال با توجه به توضیحات بالا برنامه یک مدار چشمک زن را با هم می نویسیم:

(میکرو مورد استفاده مگا 16 (atmega16) و کریستال 8 مگا هرتز است و عدد 8 led با مقاومت 330 اهم به پورت a (porta) متصل است)

مرحله اول معرفی میکرو می باشد

```
$regfile="m16def.dat"
```

مرحله بعد معرفی کریستال می باشد

```
$crystal=8000000
```

معرفی کریستال

(دو مرحله بالا در همه برنامه ها ثابت می باشد(وجود دارد))

مرحله بعد قرار دادن پورت a به عنوان خروجی می باشد (چون ما میخواهیم از ولتاژ بگیریم باید ان را به عنوان خروجی قرار دهیم)

```
Config porta =output
```

مرحله بعد ایجاد یک حلقه می باشد (در اینجا برای ایجاد حلقه می توانید، از تمام موارد گفته شده در بالا استفاده کنید)

Do

مرحله بعد روشن کردن led ها می باشد(ما در اینجا انها را یک در میان روشن میکنیم)

PEYMAN KHODDAMI

set porta.0 (روشن کردن led متصل شده به پایه 40 میکرو (porta.0))

reset porta.1 (خاموش کردن led متصل شده به پایه 39 میکرو (porta.1))

set porta.2 (روشن کردن led متصل شده به پایه 38 میکرو (porta.2))

reset porta.3 (خاموش کردن led متصل شده به پایه 37 میکرو (porta.3))

set porta.4 (روشن کردن led متصل شده به پایه 36 میکرو (porta.4))

reset porta.5 (خاموش کردن led متصل شده به پایه 35 میکرو (porta.5))

set porta.6 (روشن کردن led متصل شده به پایه 34 میکرو (porta.6))

reset porta.7 (خاموش کردن led متصل شده به پایه 33 میکرو (porta.7))

مرحله بعد ایجاد یک تاخیر زمانی است

Waitms 500 (تأخیر به مدت 500 میلی ثانیه (برای اینکه روشن بودن led ها دیده شود))

مرحله بعد برعکس کردن وضعیت پایه های موجود است (برای اینکه led های روشن، خاموش شود و led های خاموش، روشن شود).

Toggle porta (این دستور همانگونه که قبلا گفته شد یک بایت را برعکس میکند که در اینجا بایت مورد نظر 8 پایه پورت a است).

مرحله بعد ایجاد تاخیر زمانی است :

Waitms 500

مرحله بعد نوشتن پایان حلقه می باشد :

Loop (هنگامی که پردازنده میکرو این خط را میخواند به خط do پرش میکند)

و در نهایت برنامه با دستور زیر پایان می یابد:

End (پایان برنامه)

برنامه بالا را به دو شکل زیر نیز میتوان نوشت:

PEYMAN KHODDAMI

```
$regfile="m16def.dat"
```

```
$crystal=8000000
```

```
Config Porta = Output
```

```
Do
```

```
Toggle Porta
```

```
Waitms 500
```

```
Set Porta.0
```

```
Reset Porta.1
```

```
Set Porta.2
```

```
Reset Porta.3
```

```
Set Porta.4
```

```
Reset Porta.5
```

```
Set Porta.6
```

```
Reset Porta.7
```

```
Waitms 500
```

```
Loop
```

```
End
```

```
=====
```

```
$regfile="m16def.dat"
```

```
$crystal=8000000
```

```
Config Porta = Output
```

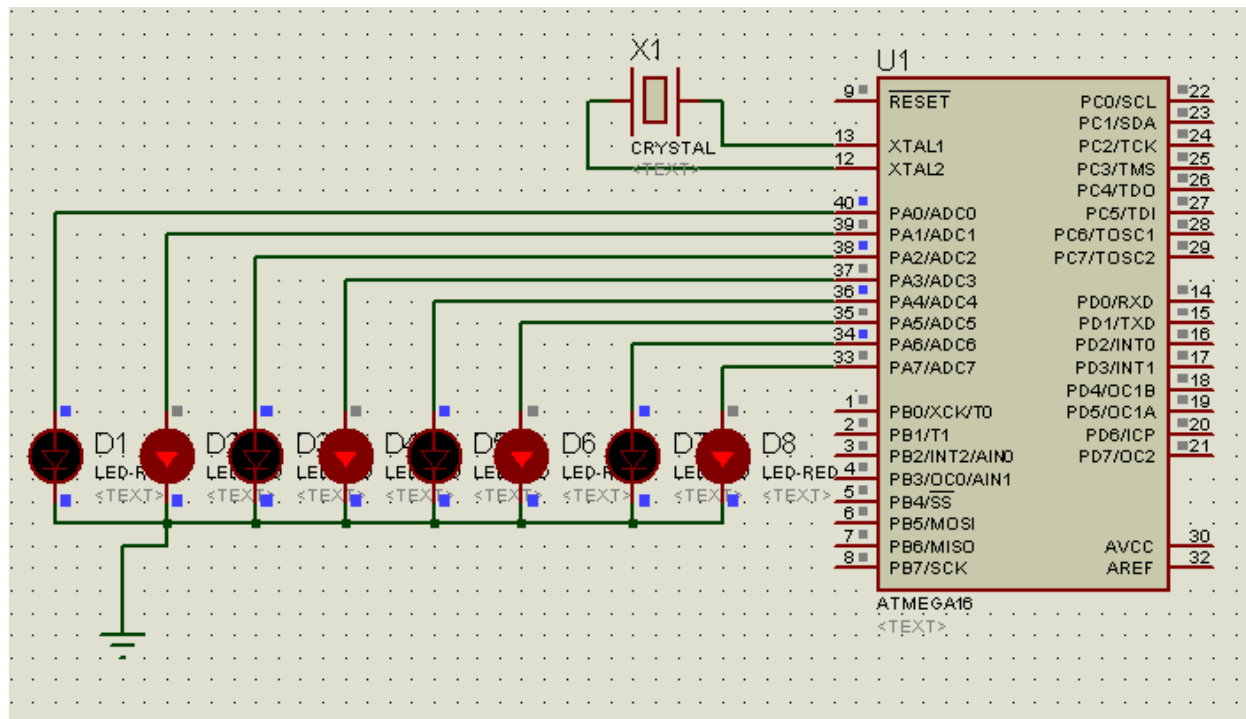
```
Do
```

```
Set Porta.1
```

```
Reset Porta.0
```

PEYMAN KHODDAMI

```
Set Porta.3
Reset Porta.2
Set Porta.5
Reset Porta.4
Set Porta.7
Reset Porta.6
Waitms 500
Set
Porta.0
Reset
Porta.1
Set
Porta.2
Reset
Porta.3
Set
Porta.4
Reset
Porta.5
Set Porta.6
Reset Porta.7
Waitms 500
Loop
End
```



مدار به کار رفته برای این برنامه را در بالا مشاهده میفرمایید.

روش های دیگر هم برای نوشتن برنامه بالا وجود دارد که به عهده شما گذاشته میشود.

PEYMAN KHODDAMI

<< کاراکتری Lcd :

این نوع Lcd در چندین نوع ساخته میشوند و دارای چند سطر و ستون می باشند که نام گذاری آنها بر مبنای سطر و ستون می باشد .

در زیر نام این Lcd آورده شده است:

16*1: که دارای یک سطر و 16 ستون می باشد.

16*1a: که دارای یک سطر و 16 ستون می باشد و میتوان ستون هشتم به بعد آن را ادرس دهی کرد.

16*2: که دارای 2 سطر و 16 ستون می باشد.

16*4: که دارای 4 سطر و 16 ستون می باشد .

20*2: که دارای 2 سطر و 20 ستون می باشد.

20*4: که دارای 4 سطر و 20 ستون می باشد.

40*2: که دارای 2 سطر و 40 ستون می باشد.

40*4: که دارای 4 سطر و 40 ستون می باشد.

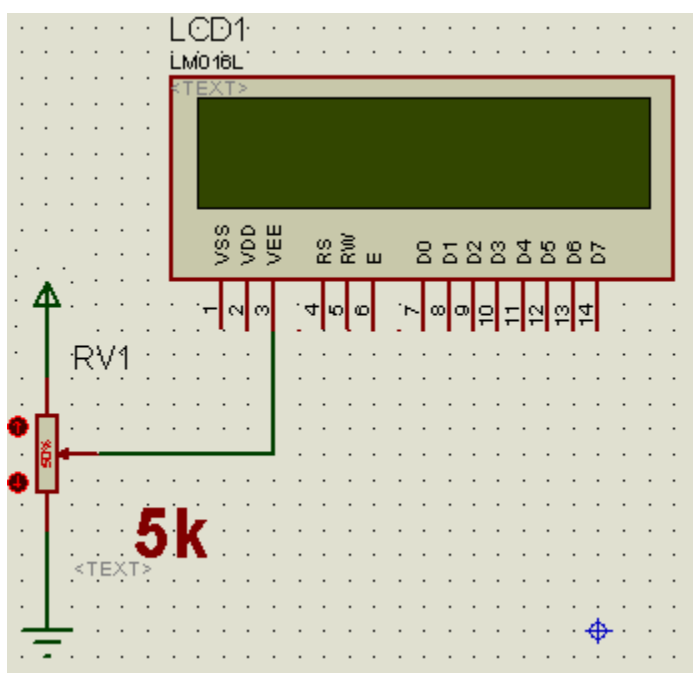
تمام Lcd های کاراکتری دارای 16 پایه می باشد که در زیر آورده شده است :

پایه شماره 1 : VSS ، این پایه ، پایه گراند Lcd است و باید به زمین مدار وصل شود.

پایه شماره 2 : VDD این پایه پایه VCC، LCD است که باید به 5 ولت وصل شود.

پایه شماره 3 : VEE این پایه روشنایی پیکسل های LCD را تعیین میکند و اتصال آن طبق مدار روبرو است:

پایه شماره 4 : RS در Lcd دو رجیستر به نام دستورالعمل داده وجود دارد اگر $rs = 0$ باشد برای Lcd گرفتن دستورالعمل آماده می شود در غیر این صورت برای داده



PEYMAN KHODDAMI

مثلا دستور cls یک دستور العمل است و qwer که باید روی lcd نوشته شود یک داده است .

پایه شماره 5 RW: این پایه دو وضعیت دارد ، rw=1 برای خواندن از lcd و rw=0 برای نوشتن در lcd .

پایه شماره 6 : E با این پایه می توان LCD را انتخاب کرد.

پایه شماره 7 : DB0 این پایه برای گرفتن دیتا (اطلاعات) از LCD میباشد (پایه دیتا ی صفر) .

پایه شماره 8 : DB1 این پایه برای گرفتن دیتا (اطلاعات) از LCD میباشد (پایه دیتا ی یک).

پایه شماره 9 : DB2 این پایه برای گرفتن دیتا (اطلاعات) از LCD میباشد (پایه دیتا ی دو).

پایه شماره 10 : DB3 این پایه برای گرفتن دیتا (اطلاعات) از LCD میباشد (پایه دیتا ی سه).

پایه شماره 11 : DB4 این پایه برای ارسال دیتا (اطلاعات) به LCD میباشد (پایه دیتا ی چهار).

پایه شماره 12 : DB5 این پایه برای ارسال دیتا (اطلاعات) به LCD میباشد (پایه دیتا ی پنج).

پایه شماره 13 : DB6 این پایه برای ارسال دیتا (اطلاعات) به LCD میباشد (پایه دیتا ی شش).

پایه شماره 14 : DB7 این پایه برای ارسال دیتا (اطلاعات) به LCD میباشد (پایه دیتا ی هفت).

پایه شماره 15 : این پایه و پایه شماره 16 تغذیه LED پشت LCD می باشد که به 5ولت متصل میشود.

پایه شماره 16 : این پایه و پایه شماره 15 تغذیه LED پشت LCD می باشد که به 5ولت متصل میشود.

راه اندازی LCD در محیط بسکام:

Lcd میتواند از دو طریق 8سیمه و 4سیمه به میکرو متصل شود.

```
CONFIG LCDBUS = constant
```

Constant میتواند 4 به معنای استفاده از مد 4 سیمه یا 8 به معنای مد هشت سیمه باشد (در صورتی که این دستور نوشته نشود ، مد 4 سیمه در نظر گرفته میشود.)

در مد چهار سیمه فقط میتوان روی lcd نوشت ولی در مد هشت سیمه میتوان اطلاعاتی را که قبلا روی lcd نوشته شده است را خواند و به میکرو ارسال کرد.

PEYMAN KHODDAMI

به طور کلی از خطوط دیتای 0 تا 3 برای خوانده از lcd و از خطوط 4 تا 7 برای نوشتن در lcd نوشته میشود در حالت نوشتن در lcd باید پایه RS پایین نگه داشته شود (صفر شود) و در حالت خواندن از LCD باید پایه RS 1 شود (5 ولت وصل شود).

از آنجا که با وجود حافظه میکرو و راحت شدن کار برنامه نویسی نیازی به خواندن از LCD نمی باشد، از پایه 0 DB تا DB3 استفاده نمی شود و پایه RW نیز به GND (صفر ولت متصل میشود).

دومین مرحله برای راه اندازی LCD معرفی کردن نام آن است:

برای این کار بعد از معرفی میکرو و کریستال با استفاده از دستور زیر می توان LCD را معرفی کرد:

```
CONFIG LCD = LCDNAME
```

که LCDNAME یکی از نام های با لا میباشد. مثلا معرفی LCD 2*16 :

```
Config lcd =16*2
```

مرحله بعد معرفی پایه های از میکرو است که lcd به آنها وصل میشود: برای مد 4 سیمه:

```
Config Lcdpin = Pin , Db4 = Pinx.y , Db5 = Pinx.y , Db6 = Pinx.y , Db7 =  
Pinx.y , Rs = Pinx.y , E = Pinx.y
```

X نام پورت است که یکی از پورت های a یا b یا c یا d ... می باشد و y شماره پایه هست که از 0 تا 7 می باشد برای مثال در زیر lcd کاراکتری 2*16 به پورت c (portc) متصل است .

```
$regfile="m16def.dat"
```

```
$crystal=8000000
```

```
Config Lcd = 16*2
```

```
Config lcdpin=pin,db4=pinc.0,db5=pinc.1,db6=pinc.2_  
,db7=pinc.3,rs=pinc.4,e=pinc.
```

در این مثال که از میکرو مگا16 (atmega16) استفاده شده ، اتصال میکرو و lcd به قرار زیر است:

پایه شماره 1 : VSS ، این پایه باید به زمین مدار وصل شود .

پایه شماره 2 : VDD این پایه باید به 5 ولت وصل شود.

PEYMAN KHODDAMI

پایه شماره 3 : VEE این پایه با یک مقاومت (مقدار مقاومت بستگی به روشنایی مورد نظر شما دارد) به VCC وصل میشود.

پایه شماره 4 : RS این پایه به پورت c پین شماره 4 یا پین c.4 متصل میشود (پایه 26 میکرو مگا16).

پایه شماره 5 : RW این پایه به gnd متصل میشود .

پایه شماره 6 : E این پایه به پورت c پین شماره 5 یا پین c.5 متصل میشود(پایه 27 میکرو مگا16) .

پایه شماره 7 : DB0 این پایه به جایی متصل نمی شود.

پایه شماره 8 : DB1 این پایه به جایی متصل نمی شود.

پایه شماره 9 : DB2 این پایه به جایی متصل نمی شود.

پایه شماره 10 : DB3 این پایه به جایی متصل نمی شود.

پایه شماره 11 : DB4 این پایه به پورت c پین شماره 0 یا پین c.0 متصل میشود(پایه 22 میکرو مگا16).

پایه شماره 12 : DB5 این پایه به پورت c پین شماره 1 یا پین c.1 متصل میشود(پایه 23 میکرو مگا16) .

پایه شماره 13 : DB6 این پایه به پورت c پین شماره 2 یا پین c.2 متصل میشود(پایه 24 میکرو مگا16) .

پایه شماره 14 : DB7 این پایه به پورت c پین شماره 3 یا پین c.3 متصل میشود(پایه 25 میکرو مگا16).

پایه شماره 15 : این پایه و پایه شماره 16 تغذیه LED پشت LCD می باشد که به 5ولت متصل میشود.

پایه شماره 16 : این پایه و پایه شماره 15 تغذیه LED پشت LCD می باشد که به 5ولت متصل میشود.

توجه کنید که اگر پایه VEE به 5ولت متصل شود پیکسل های LCD دارای بیشترین روشنای و اگر به گراند وصل شود دارای کمترین روشنایی می باشند.

اتصال پایه های 15 و 16 اختیاری است .

برای مد 8 سیمه:

Config Lcdpin = Portd , Rs = Pinc.5 , E = Pinc.6 , Rw = Pinc.7

PEYMAN KHODDAMI

در مثال بالا پایه db0 به پورت d.0 و پایه db1 به پورت d.1 و ... پایه db7 به پایه d.7 و پایه rs به پایه c.5 و پایه e به پایه c.6 و پایه rw به پایه c.7 متصل میگردد ، در این حالت یازده پایه از lcd اشغال میشود .

بعد از راه اندازی LCD نوبت کار کردن با آن است .

برای نوشتن روی LCD از دستور زیر استفاده می شود :

```
Lcd "x"
```

که X می تواند هر چیزی باشد (البته در محدوده کارکتر های اسکی).مانند:

```
Lcd "1nafar"
```

یا

```
LCD "+ +"
```

یا

```
Lcd "123@#$qwe"
```

برای پاک کردن lcd از دستور cls استفاده می شود .مانند:

```
lcd "@#$$%^&&*" ("
```

```
Wait 1
```

```
Cls
```

```
Lcd "asdfgfhk"
```

با دستور زیر می توان در سطر ها و ستون های دیگر lcd نوشت.

```
Locate x,y
```

که x ادرس سطر و y ادرس ستون می باشد.

```
Locate 1,2
```

```
Lcd "qwert"
```

```
Locate 2,1
```

```
"Lcd "123456
```

PEYMAN KHODDAMI

Locate 2,8

```
"Lcd "mnbv"
```

توجه کنید برای یک lcd ، مثلا 2*16 حداکثر 2،x و حداکثر y، 16 است .

با دستورات زیر میتوان به سطر های مختلف lcd پرش کرد و در انجا متن را نوشت :

Upperline

با این دستور به خط بالای پرش میشود

Lowerline

با این دستور به خط پایینی پرش میشود

Home

با این دستور به سطر اول ، ستون اول پرش میشود

Thirdline

Fourthline

با دو دستور بالا میتوان به ترتیب به خط سوم و چهارم پرش کرد (این دستور برای lcd های است که 4 سطر دارند)،
مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Config Lcd = 16 * 4
```

```
Config Lcdpin = Pin , Db4 = Pinc.0 , Db5 = Pinc.1 , Db6 = Pinc.2 , Db7 =  
Pinc.3 , Rs = Pinc.4 , E = Pinc.5
```

Lowerline

```
Lcd "qwert"
```

Thirdline

```
Lcd "vcxz"
```

Fourthline

PEYMAN KHODDAMI

Lcd "1236"

Upperline

Lcd "erff"

Wait 2

Home

Lcd "123654"

End

با دستور زیر میتوان تعداد فضای خالی دلخواه را بر روی lcd ایجاد کرد:

LCD SPC (x)

تعداد x ستون خالی میماند ، وبعد از ستون x نوشتن ادامه میابد.

با استفاده از دستور زیر میتوانید lcd را روشن یا خاموش کنید:

DISPLAY ON / OFF

lcd : On روشن میشود / lcd : off خاموش میشود.

با استفاده از دستور زیر میتوانید کنتراست (میزان نور (کم رنگی و پر رنگی) متن) متن را تغییر دهید:

Lcdcontrast x

x میزان کنتراست است که میتوانید بین 0 تا 3 باشد ، به ازای 0 کمترین کنتراست و به ازای 3 بیشترین کنتراست مشاهده میشود.

Lcd دارای یک مکان نما می باشد که با دستور زیر میتوان ان را روشن یا خاموش یا چشمک زن یا ثابت قرارداد .

Cursor On با این دستور مکان نما روشن می شود (در حالت عادی مکان نما روشن است).

Cursor off با این دستور مکان نما خاموش می شود.

Cursor blink با این دستور مکان نما چشمک می زند .

Cursor noblink با این دستور مکان نما دیگر چشمک نمی زند.

PEYMAN KHODDAMI

با دستور زیر می توانید کاراکتر های روی lcd را به چپ یا راست شیفت دهید.

shiftlcd left این دستور کارکترها را به اندازه یک ستون به چپ منتقل میکند.

shiftlcd right این دستور کارکترها را به اندازه یک ستون به راست منتقل میکند .

با دستور زیر می‌توانید مکان نما را به راست یا چپ منتقل کنید:

```
SHIFTCURSOR LEFT | RIGHT
```

left این دستور مکان نما را به اندازه یک ستون به چپ منتقل میکند.

right این دستور مکان نما را به اندازه یک ستون به راست منتقل میکند .

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Config Lcd = 16 * 4
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.3 , Rs = Pind.0 , E = Pind.1
```

```
Lcdcontrast 1
```

```
Locate 2 , 1
```

```
Lcd "12356"
```

```
Shiftcursor Right
```

```
Wait 1
```

```
Display Off
```

```
Wait 1
```

```
Display On
```

```
Lcdcontrast 2
```

```
Locate 4 , 1
```

```
Lcd Spc(5(
```

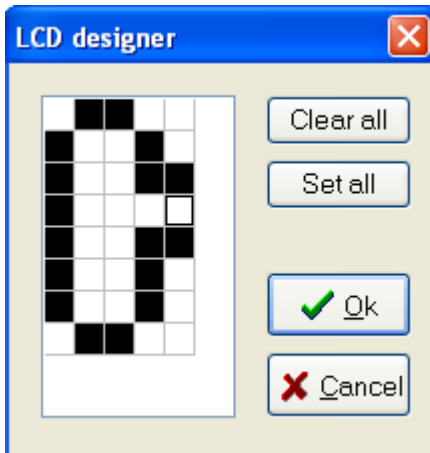
PEYMAN KHODDAMI

Lcd "qwer"

Shiftcursor Left

End

<نوشتن فارسی روی lcd :



Lcd های کاراکتری دارای یک حافظه دائم می باشد که درون آن فقط کد، کارکترهای اسکی وجود دارد (کد کارکترهای فارسی در آن وجود ندارد). در lcd حافظه موقتی وجود دارد که در آن می توان تا 8 کارا کتر دلخواه را قرار داد .

برای ساخت کاراکتر دلخواه مراحل زیر را دنبال کنید:

از منوی tools گزینه lcd designer را انتخاب کنید، پنجره جدیدی باز می شود که شما می توانید در آن کاراکتر دلخواه خود را ایجاد کنید.

بعد از ایجاد کاراکتر دلخواه روی ok کلیک کنید ،پنجره بسته می شود و یک خط به برنامه شما اضافه می شود. مانند زیر:

```
Deflcdchar ?,1 ,4 ,4 ,4 ,4,31,20,4
```

به جای علامت سوال باید یکی از اعداد 0 تا 7 می باشد گذاشته شود .

بعد از ساخت کاراکتر جدید بادرستور زیر می توانید آن را روی lcd نشان دهید:

```
Lcd chr (?)
```

به جای علامت سوال باید شماره کاراکتر که یکی از اعداد 0 تا 7 می باشد گذاشته شود . مانند:

```
$regfile="m16def.dat"
```

```
$crystal=8000000
```

```
Config Lcd = 16*2
```

```
Config lcdpin=pin ,
```

```
db4=pinc.0,db5=pinc.1,db6=pinc.2,db7=pinc.3,rs=pinc.4,e=pinc.5
```

```
Deflcdchar 0,1 ,4 ,4 ,4 ,4,31,20,4
```

```
Locate 1,1
```


PEYMAN KHODDAMI

<< اعداد و متغیرها در بسکام :

< دستور زیر بعد یک متغیر را نشان میدهد . با این دستور می توانید متغیرهایی که در برنامه به کار برده می شوند تعریف کنید .

DIM X AS data type

X نام متغیری است، که در برنامه بکار برده میشود و Data type نوع داده است که می تواند طبق موارد STRING یا WORD یا LONG یا INTEGER یا BYTE یا BIT یا SINGLE یا DOUBLE باشد . (همچنین میتواند XRAM یا SRAM یا ERAM یا OVERLAY یا location که همگی متغیرهای از انواع حافظه ها هستند باشد ، که در بخشهای بعدی توضیح داده میشود.)

در صورت استفاده از متغیر STRING , بیشترین طول آن نیز باید نوشته شود .

BIT : این متغیر میتواند صفر یا یک باشد .

BYTE : این متغیر میتواند از 0 تا 255 تغییر کند و فقط شامل اعداد صحیح مثبت می شود .

WORD : این متغیر میتواند از 0 تا 65535 تغییر کند و فقط شامل اعداد صحیح مثبت می شود.

INTEGER : این متغیر میتواند از -32767 تا +32767 تغییر کند و فقط شامل اعداد صحیح مثبت و منفی می شود.

LONG : این متغیر میتواند از -214783648 تا +214783647 تغییر کند و فقط شامل اعداد صحیح مثبت و منفی می شود.

SINGLE : این متغیر میتواند از -1.5×10^{45} تا 3.4×10^{38} تغییر کند و فقط شامل اعداد صحیح و اعشاری مثبت و منفی می شود .

Double : این متغیر میتواند از 5.0×10^{-324} تا 1.7×10^{308} تغییر کند و فقط شامل اعداد صحیح و اعشاری مثبت و منفی می شود.

STRING : این متغیر میتواند از 0 تا 245 بایت تغییر کند تغییر کند و باری حروف و علائم استفاده می شود. در صورت استفاده از متغیر STRING , بیشترین طول آن نیز باید نوشته شود . مثال:

DIM B AS BIT 'BIT can be 0 or 1

DIM A AS BYTE 'BYTE range from 0 - 255

DIM K AS INTEGER

PEYMAN KHODDAMI

DIM MICRO AS WORD

DIM HASAN AS LONG

شما همچنین می‌توانید یک متغیر ارایه ای (با یک نام چندین متغیر) بسازید مانند:

Dim a(10) as word

در این حالت شما می‌توانید از 10 متغیر a (a(0) تا a(10)) در برنامه استفاده کنید

```
$regfile = "m16def.dat
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A(4) As Byte
```

```
Dim Ali As Word
```

```
Dim Wqew As
```

```
Byte
```

```
A(1) = 10
```

```
A(2) = 11
```

```
Wqew = 5
```

```
Locate 1 , 1
```

```
A(2) = A(3) +
```

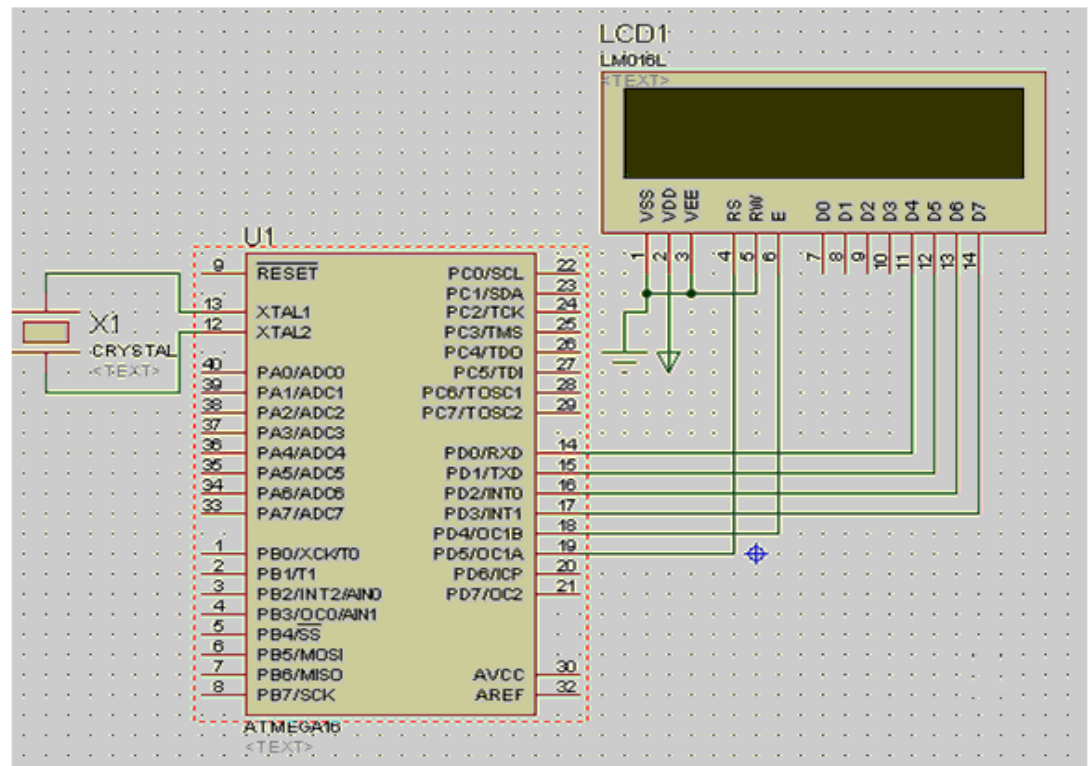
```
(A(4
```

```
Ali = A(2) +
```

```
Wqew
```

```
Lcd Ali
```

```
Locate 2 , 1
```



PEYMAN KHODDAMI

```
Ali = A(1) * Wqew
```

```
Lcd Ali
```

```
End
```

این شکل سخت افزار استفاده شده در کلیه مثال های این فصل است.

نکته : در صورتی که در یک متغیر بیشتر از بعدش مقدار قرار دهید با خطا مواجه میشوید.مانند:

```
DIM A AS BYTE
```

```
A=300
```

مورد بالا غلط می باشد، چون بایت می تواند از 0 تا 255 تغییر کند و مقدار 300 بیشتر از بعد بایت است.

فرم دیگر دستور بالا به شکل زیر است:

DEFBIT x	Define BIT
DEFBYTE x	Define BYTE
DEFINT x	Define INTEGER
DEFWORD x	Define WORD
DEFLNG x	Define LONG
DEFSNG x	Define SINGLE
DEFDBL x	Define DOUBLE

X : نام متغیر است که میتواند یکی از حروف انگلیسی

باشد ، حدود تغییر متغیر مانند مورد قبل میباشد و فقط

شکل نوشتن دستور عوض شده است.

< دستور CONST :

برای تعریف یک ثابت از این دستور استفاده می شود :

```
CONST SYMBOL= NUMCONST
```

```
CONST SYMBOL= STRINGCONST
```

```
CONST SYMBOL= EXPRESSION
```

SYMBOL نام ثابت و NUMCONST مقدار عددی انتساب یافته به SYMBOL و STRINGCONST رشته انتساب یافته به

SYMBOL و EXPRESSION میتواند عبارتی باشد که نتیجه آن به SYMBOL انتساب یابد . مانند:

```
CONST S = "TEST"
```

```
CONST A = 5
```

```
CONST B1 =&B1001
```

```
CONST X = (B1 * 3 ) + 2
```

PEYMAN KHODDAMI

<دستور INCR و دستور DECR:

INCR X

DECR X

دستور INCR یک واحد به متغیر عددی X می افزاید و دستور DECR یک واحد از آن کم میکند .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Byte
```

```
Dim B As Long
```

```
Dim Bp As Byte
```

```
Do
```

```
Incr A
```

```
Decr B
```

```
Locate 1 , 1
```

```
Lcd B
```

```
Locate 2 , 1
```

```
Lcd A
```

```
WAITMS 500
```

```
Loop
```

```
End
```

<دستور SWAP:

```
SWAP var1 , var2
```

PEYMAN KHODDAMI

با اجرای این دستور محتوای متغیر var1 در متغیر var2 و محتوای متغیر var2 در متغیر var1 قرار می‌گیرد .

دو متغیر var1 و var2 بایستی از یک نوع باشند .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte

Dim B As Byte

Cls

A = 10

B = 20

Swap A , B                                'swap them

Locate 1 , 1

Lcd A                                       'A=20

Locate 2 , 1

Lcd B                                       'B=10

End
```

دستور Config Single :

با این دستور میتوان تعداد رقم اعشار سک متغیر از جنس Single را معین کرد ، این دستور به فرم کلی زیر است:

```
Config Single = Scientific , Digits = x
```

X : عددی بین 1 تا 7 است که تعداد رقم اعشار را نشان میدهد ، در صورتی که از این دستور استفاده کنید ، کلیه متغیر

های Single موجود در برنامه تحت پوشش قرار میگیرند ، مثال :

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 8000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Single = Scientific , Digits = 1

Cls

Dim A As Single

A = 10 : A = A / 9

Locate 1 , 1 : Lcd A

End
```

< دستور format :

این دستور یک متغیر عددی را شکل دهی می کند .

```
x = Format (var , "form")
```

var رشته ای است که شکل دهی شود و نتایج در x قرار می گیرد. form نوع شکل دهی است .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim S As String *10, I As Integer

S = " 123"

S = Format(s,"      ")

Locate 1 , 1                                '5 Space

Lcd S                                         'Two Space First ,
Then 123
```

PEYMAN KHODDAMI

```
S = "12345"  
  
S = Format(s , "000.000")  
  
Locate 1 , 8  
  
Lcd S                                     's = "012.345"  
  
S = Format(s,"+")  
  
Locate 2 , 1  
  
Lcd S                                     's = "+12345"  
  
End
```

<دستور fusing :

از این دستور برای روند کردن یک متغیر عددی استفاده می شود .

```
target = Fusing (source , "mask")
```

source رشته موردنظر برای شکل دهی و نتایج در target قرار می گیرد. mask نوع شکل دهی است . عمل mask حتما باید با علامت # شروع شود و حداقل باید یکی از علامات # یا & را بعد از ممیز داشته باشد. با استفاده از # عدد روند می شود و در صورت استفاده از & روندی صورت نمی گیرد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As Single
```

```
Dim A As Byte
```

```
Cls
```

```
S = 10
```

```
A = 3
```

PEYMAN KHODDAMI

```
S = S / A  
  
Locate 1 , 1  
  
Lcd S 'lcd "3.3333333333333333"  
  
Locate 2 , 1  
  
Lcd Fusing(s , "#.##") 'lcd "3.33  
  
Locate 2 , 8  
  
Lcd Fusing(s , "#.####") 'lcd "3.3333  
  
End
```

دستور SHIFT var :

با این دستور میتوان تمام بیت ها را یک بیت به سمت راست یا چپ منتقل کرد ، این دستور به فرم کلی زیر است :

```
SHIFT var , LEFT/RIGHT
```

Var : نام متغیر یا عدد ثابتی است که میخواهیم ان را شیفت دهیم (منتقل کنیم)

LEFT/RIGHT : جهت شیفت را مشخص میکند ، که میتواند راست یا چپ باشد مانند:

```
$regfile = "m16def.dat"  
  
$crystal = 12000000  
  
Config Porta = Output  
  
Porta = &B10000000  
  
Do  
  
Shift Porta , Right  
  
Wait 1  
  
Loop  
  
End
```

مثال :

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte

A = 5

Locate 1 , 1

lcd a

Shift A , Left

Locate 2 , 1

Lcd A

End
```

در مثال بالا مقدار اولیه 5 (0101 باینری) برای a در نظر گرفته شده است ، این مقدار بر روی سطر و ستون اول lcd به نمایش در میاید ، سپس با دستور Shift A , Left تمام بیت های متغیر a به سمت چپ منتقل میشوند ، پس مقدار a برابر با 10 (1010 باینری) میشود .

دستور ROTATE:

این دستور تقریباً مانند دستور shift میباشد و تمام بیت های یک متغیر را به سمت راست یا چپ جابجا میکند ، شما همچنین با این دستور میتوانید بین های روشن یک پورت را جابجا کنید و به فرم کلی زیر است:

```
ROTATE var , LEFT/RIGHT
```

Var : نام متغیر یا عدد ثابتی یا پورتی است که میخواهیم آن را شیفت دهیم (منتقل کنیم)

LEFT/RIGHT : جهت انتقال را مشخص میکند ، که میتواند راست یا چپ باشد مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

PEYMAN KHODDAMI

```
Config Porta = Output
Config Portb = Output
Dim B As Byte
Portb = &B00000001
B = 1
Do
Rotate B , Right
Rotate Portb , Right
Wait 1
Porta =B
Loop
End
```

دستور SHIFTIN:

با این دستور میتوان تعداد بیت را درون یک متغیر شیفت داد ، بیت ها بصورت سریال به یکی از پایه ای میکرو اعمال میشوند . این دستور به فرم کلی زیر است:

```
SHIFTIN pin , pclock , var , option [, bits , delay ]
```

Pin : نام پایه ای است که اطلاعات سریال به آن وارد میشوند.

Pclock : نام پایه ای است که خط کلاک دستگاه دیگر به آن متصل میشود.

Var: نام متغیری است که اطلاعات در آن ذخیره میشوند.

PEYMAN KHODDAMI

Option: نوع شیفت دادن و کلاک را معین میکند و یکی از اعداد زیر است:

- 0- هنگامی که فرکانس کلاک کم باشد ابتدا msb (بیت باز شتر) شیفت داده میشود.
- 1- هنگامی که فرکانس کلاک زیاد باشد ابتدا msb (بیت باز شتر) شیفت داده میشود.
- 2- هنگامی که فرکانس کلاک کم باشد ابتدا lsb (بیت کم ارزش) شیفت داده میشود.
- 3- هنگامی که فرکانس کلاک زیاد باشد ابتدا lsb (بیت کم ارزش) شیفت داده میشود.

Bits : مشخص کننده تعداد بیت است که وارد متغیر میشود و نهایتا میتواند 255 باشد(این گزینه اختیاری است)

Delay: تاخیر زمانی برحسب میکرو ثانیه میباشد که در بین دریافت هر بیت رخ میدهد(استفاده از این گزینه اختیاری است) (در صورت بالا بودن کلاک از این گزینه استفاده نکنید).مانند:

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Porta =input

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Word                                     '200

Shiftin Pina.0 , Porta.1 , A , 0

End
```

در مثال بالا پایه کلاک پین a.1 و پایه دیتا پایه a.0 میباشد ، که از میکرو دیگر که برنامه ان را در دستور بعدی مشاهده میفرماید ، گرفته شده است .

دستور SHIFTOUT :

با این دستور میتوان یک متغیر را بصورت سریال از یک پایه به بیرون داد . این دستور به فرم کلی زیر است:

```
SHIFTOUT pin , pclock , var , option [, bits , delay ]
```

Pin : نام پایه ای است که اطلاعات سریال از ان خارج میشوند.

Pclock : نام پایه ای است که خط کلاک دستگاه دیگر به ان متصل میشود.(خروجی کلاک است)

PEYMAN KHODDAMI

Var: نام متغیری است که اطلاعات در آن وجود دارد و باید ارسال شود.

Option: نوع شیفت دادن و کلاک را معین میکند و یکی از اعداد زیر است:

- 0- هنگامی که فرکانس کلاک کم باشد ابتدا msb (بیت باز شتر) شیفت داده میشود.
- 1- هنگامی که فرکانس کلاک زیاد باشد ابتدا msb (بیت باز شتر) شیفت داده میشود.
- 2- هنگامی که فرکانس کلاک کم باشد ابتدا lsb (بیت کم ارزش) شیفت داده میشود.
- 3- هنگامی که فرکانس کلاک زیاد باشد ابتدا lsb (بیت کم ارزش) شیفت داده میشود.

Bits : مشخص کننده تعداد بیت از متغیر است که به بیرون شیفت داده میشود و نهایتا میتواند 255 باشد(این گزینه اختیاری است)

Delay: تاخیر زمانی برحسب میکرو ثانیه میباشد که در بین ارسال هر بیت رخ میدهد(استفاده از این گزینه اختیاری است)
(در صورت بالا بودن کلاک از این گزینه استفاده نکنید).مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
Dim A As Word
```

```
A = &B11001000 '200
```

```
Shiftout Pina.0 , Porta.1 , A , 0
```

```
End
```

<<< نکته : برای نشان دادن اعداد به فرم باینری از **b&** و برای نشان دادن اعداد به فرم هگز از **h&** استفاده می شود.

مانند:

```
b&0110010
```

```
&h01ff
```

در اینجا تمامی دستورات مر بوط به اعداد و متغیر ها که در زبان بیسیک برای میکرو avr است گفته شد .

در درسهای بعدی با این دستورات به صورت کاربردی آشنا می شوید.(مثال ها را در بسکام کپی کنید و طرز کار دستورات را ببینید)

PEYMAN KHODDAMI

<< دستورات مربوط به کار با رشته ها

< دستور ASC

```
Var = ASC (string)
```

این دستور اولین کاراکتر یک متغیر از نوع داده STRING را به مقدار اسکی آن تبدیل می کند .

برای دیدن کارکترهای اسکی و کد متناظر با آنها به ضمیمه ها مراجعه کنید.

< دستور INSTR:

این دستور محل و موقعیت یک زیر رشته را در رشته دیگر مشخص می کند .

```
Var =Instr (start , String ,Subset)
```

```
Var =Instr (String ,Subset)
```

Var عددی است که مشخص کننده محل SUBSTR در رشته اصلی STRING می باشد و زمانیکه زیر رشته مشخص شده در رشته اصلی نباشد صفر برگردانده می شود. START نیز عددی دلخواه است که مکان شروع جستجو زیر رشته در رشته اصلی را مشخص می کند . در صورتیکه START قید نشود تمام رشته از ابتدا جستجو می شود . رشته اصلی تنها باید از نوع رشته باشد ولی زیر رشته (SUBSTR) می تواند رشته و عدد ثابت هم باشد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 15
```

```
Dim Z As String * 5
```

```
Dim Bp As Byte
```

```
Cls
```

PEYMAN KHODDAMI

```
S = "This is a test"
```

```
Z = "is"
```

```
Bp = Instr(s , Z)
```

```
Lcd Bp
```

```
Bp = Instr(4 , S , Z)
```

```
Lcd Bp
```

```
End
```

<دستور :CHECKSUM

این دستور مجموع کد دسیمال اسکی رشته X را برمی گرداند که البته اگر مجموع کد اسکی رشته از عدد 255 بیشتر شود مقدار 256 از مجموع کم می شود .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 10 ' Dim Variable
```

```
S = "test"
```

```
Locate 1 , 1 ' Assign Variable
```

```
Lcd Checksum(s) ' print value
```

```
( 192
```

```
S = "testNext"
```

```
Locate 2 , 1 ' assign variable
```

```
Lcd Checksum(s) ' lcd value 127
```

```
(127=383 - 256)
```

```
End
```

PEYMAN KHODDAMI

< دستورات HIGH و LOW و Highw :

LOW این دستور (LSB (least significant byte) یک متغیر را برمی گرداند . (lsb باید کمتر از 8 بیت باشد)

HIGH این دستور (MSB (most significant byte) یک متغیر را برمی گرداند . (msb باید کمتر از 8 بیت باشد)

HIGH این دستور (MSB (most significant byte) یک متغیر را برمی گرداند . (msb باید کمتر از 16 بیت باشد)

```
Var = HIGH (s)
```

MSB متغیر S در Var قرار می گیرد . (var باید از جنس byte باشد)

```
Var = LOW (s)
```

LSB متغیر S در Var قرار می گیرد . (var باید از جنس byte باشد)

```
X = Highw(s)
```

MSB متغیر S در Var قرار می گیرد . (var باید از جنس word باشد)

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim I As Integer
```

```
Dim Z As Byte
```

```
Dim Q As Byte
```

```
Cls
```

```
I = &h1001
```

PEYMAN KHODDAMI

```
Z = Low(i)                                ' is 1
Locate 1 , 1
Lcd Z
Q = High(i)                                'IS 16
Locate 2 , 1
Lcd Q
End
```

مثال

```
$regfile = "m16def.dat"
$crystal = 12000000
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5
Dim X As Word , L As Long
L = &H12345678
X = Highw(L)
'4660(des)=1238(hex(
Locate 2 , 1
Lcd X
End
```

<دستور LCASE و دستور UCASE :

دستور LCASE : این دستور تمام حروف رشته مورد نظر را تبدیل به حروف کوچک می کند .

```
Target = Lcase (source)
```

تمام حروف رشته source کوچک شده و در رشته target جای داده می شود .

PEYMAN KHODDAMI

دستور UCASE: این دستور تمام حروف رشته مورد نظر را تبدیل به حروف بزرگ می کند .

```
Target = Ucase (source)
```

تمام حروف رشته source بزرگ شده و در رشته target جای داده می شود .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 12
```

```
Dim Z As String * 12
```

```
Dim Q As String * 12
```

```
S = "Hello World"
```

```
Q = "QWERTGFDD"
```

```
Z = Ucase(s )
```

```
'Z = HELLO WORLD
```

```
Locate 1 , 1
```

```
Lcd Z
```

```
Z = Lcase(q)
```

```
Locate 2 , 1
```

```
Lcd Z
```

```
End
```

<دستور RIGHT و دستور LEFT :

دستور RIGHT: با این دستور قسمتی از یک رشته را جدا می کنیم .

```
Var = RIGHT (var1 , n)
```

از سمت راست رشته var1 به تعداد کاراکتر n , رشته ای جدا شده و در رشته var قرار می گیرد .

PEYMAN KHODDAMI

دستور LEFT: با این دستور کاراکترهای سمت چپ یک رشته را به تعداد تعیین شده جدا می کند .

```
Var = LEFT(var1 , n)
```

از سمت چپ رشته var1 به تعداد کاراکتر n , رشته ای جدا شده و در رشته var قرار می گیرد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 15 , Z As String * 15
```

```
Cls
```

```
S = "abcdefg"
```

```
Z = Left(s , 5)
```

```
'Z = abcde
```

```
Locate 1 , 1
```

```
Lcd Z
```

```
Z = Left(s , 1)
```

```
'Z = a
```

```
Locate 1 , 8
```

```
Lcd Z
```

```
Z = Right(s , 5)
```

```
'Z = CDEFG
```

```
Locate 2 , 1
```

```
Lcd Z
```

```
Z = Right(s , 2)
```

```
'Z = FG
```

```
Locate 2 , 8
```

```
Lcd Z
```

```
End
```

PEYMAN KHODDAMI

<دستور LEN :

این دستور طول، یا بعبارتی تعداد کاراکترهای یک رشته را برمیگرداند .

```
Var = Len(string)
```

طول رشته string در متغیر عددی VAR قرار می گیرد . رشته string نهایتاً می تواند 255 بایت طول داشته باشد .
توجه داشته باشید که فضای خالی (SPACE BAR) خود یک کاراکتر به حساب می آید .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 12
```

```
Dim A As Byte
```

```
Cls
```

```
S = "test"
```

```
A = Len(s)
```

```
Locate 1 , 1 ' 4
```

```
Lcd Len(s)
```

```
S = "test"
```

```
A = Len(s)
```

```
Locate 2 , 1
```

```
Lcd A '6
```

```
End
```

<دستور LTRIM :

این دستور فضای خالی یک رشته را حذف می کند .

PEYMAN KHODDAMI

```
Var = LTRIM( Q)
```

فضای خالی رشته Q برداشته می شود و رشته بدون فضای خالی در متغیر رشته ای var قرار می گیرد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim S As String * 10
```

```
Dim A As String * 10
```

```
Cls
```

```
S = "Q Q 1"
```

```
Locate 1 , 1
```

```
A = Ltrim(s)
```

```
Lcd A 'QQ1
```

```
S = "Q Q Q"
```

```
Locate 2 , 1
```

```
Lcd Ltrim(s) 'QQQ
```

```
End
```

<دستور MID :

با این دستور می توان قسمتی از یک رشته را برداشت و یا قسمتی از یک رشته را با قسمتی از یک رشته دیگر عوض کرد .

```
VAR=MID (VAR1, ST [, L])
```

1- قسمتی از رشته var1 با شروع از کاراکتر stام و طول L برداشته شده و در متغیر var قرار می گیرد.

```
MID (VAR, ST [, L])=VAR1
```

PEYMAN KHODDAMI

2- رشته var1 در رشته var با شروع از کاراکتر St ام و طول L قرار می گیرد .

در صورت قید نکردن گزینه اختیاری L, بیشترین طول در نظر گرفته می شود .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim S As String * 10

Dim Z As String * 10

Cls

S = "adswer"

Z = Mid(s , 2 , 3)

Locate 1 , 1

"Lcd Z                                     'lcd "dsw"

Z = "5685"

Mid(s , 2 , 3) = Z

Locate 2 , 1

"Lcd S                                     'lcd "a568er"

End
```

< دستور space :

برای ایجاد فضای خالی در میان یک رشته ، از این دستور استفاده می شود .

```
Var = SPACE (x)
```

X تعداد فضای خالیست که بعنوان رشته در متغیر رشته ای var جای می گیرد .

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim S As String * 10

Dim Z As String * 10

Cls

S = Space(5)

Z = "qwer"

Locate 1 , 1

Lcd "(" ; S ; Z ; ")"

End
```

با این دستورات در بخش های بعدی بیشتر آشنا میشویم.

<<دیگر دستورات حلقه و پرش و شرط:

PEYMAN KHODDAMI

گاهی نیاز است که یک قسمت از برنامه چندین بار اجرا شود یا در حین اجرای برنامه در یک خط به خط دیگری رجوع شود، برای این کار از دستورات حلقه و پرش که چندین نوع است ، استفاده میشود .

انواع دستورات حلقه و پرش:

< دستور goto :

label:

برنامه

Goto label

با این دستورات می توان به برجسب label پرش کرد . برجسب label باید با علامت : (colon) پایان یابد و می تواند تا 32 کارکتر طول داشته باشد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
Q :
```

```
Set Porta.0
```

```
Waitms 600
```

```
Reset Porta.0
```

```
Waitms 600
```

```
Goto q
```

```
End
```

مثال

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
Q :
```

PEYMAN KHODDAMI

```
Set Porta.0  
  
Waitms 600  
  
Reset Porta.0  
  
Waitms 600  
  
Goto Q  
  
End
```

دستور GOSUB :

با این دستور میتوان به یک برپسب پرش کرد ، باز گشت ار پرچسب با دستور return انجام میشود ، این دستور به فرم کلی زیر است:

```
GOSUB label
```

```
Label:
```

برنامه

Return

مثال :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
E
```

```
Do
```

```
Gosub Q
```

```
Set Porta.0
```

```
Waitms 600
```

```
Loop
```

```
End
```

```
Q:
```


PEYMAN KHODDAMI

```
Reset Porta.0
```

```
Waitms 600
```

```
Return
```

دستور ON var :

این دستور به فرم کلی زیر است:

```
ON var [GOTO] [GOSUB] label1 [, label2 ] [,CHECK]
```

در این دستور به ازای متغیر var به برجسب nم پرش میشود ، مثلا اگر var=2 باشد به برجسب سوم پرش میشود (تعداد برجسب ها نامحدود است) بستگی به حافظه میکرو دارد . در صورتی که از دستور gosub استفاده کنی باز گشت از زیر برنامه (برجسب) با دستور return انجام میشود و در صورت استفاده از goto باید به حلقه اصلی پرش کرد که شما میتوانید از دستور goto یا دیگر دستورات استفاده کنید . مثال با دستور gosub :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
Dim S As Byte
```

```
Do
```

```
On S Gosub Q , W , E , R , T , Y , U , I
```

```
Incr S
```

```
Wait 1
```

```
Loop
```

```
End
```

```
Q:
```

```
Set Porta.0
```

```
Return
```

```
W:
```

```
Set Porta.1
```

```
Return
```

```
E:
```

PEYMAN KHODDAMI

Set Porta.2

Return

R:

Set Porta.3

Return

T:

Set Porta.4

Return

Y:

Set Porta.5

Return

U:

Set Porta.6

Return

I

Set Porta.7

Return

مثال با دستور goto:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Porta = Output
```

```
Dim S As Byte
```

```
A:
```

```
Wait 1
```

```
On S Goto Q , W
```

```
End
```

PEYMAN KHODDAMI

Q:

```
Set Porta.0
```

```
Incr S
```

```
Goto A
```

W:

```
Set Porta.1
```

```
Goto A
```

در مثال های بالا ، در اول کار مقدار s صفر است پس به پرچسب اول پرش شده و پورت a.0 یک میشود ، بعد از این عمل دوباره به حلقه اصلی پرش میشود و به s یک واحد افزوده میشود ، s برابر با 1 میشود ، پس به پرچسب دوم پرش میشود در انجا پورت...

<دستور do-loop :

فرم کلی دستورات DO ... LOOP بصورت زیر می باشد .

```
DO
```

```
برنامه
```

```
LOOP
```

این حلقه یک حلقه بینهایت است ، که با EXIT DO می توان از درون حلقه خارج شد و اجرای برنامه در خط بعد از loop ادامه یابد.

همچنین با دستور زیر میتوان تعداد دفعات اجرای آن را معین کرد :

```
do
```

```
برنامه
```

```
Loop Until A = x
```

که A یک متغیر از جنس دلخواه و x تعداد دفعات تکرار است .

در مثال زیر در هر بار تکرار حلقه یک واحد به A اضافه می گردد و هرگاه مقدار A به 10 رسید خط بعد از حلقه اجرا می گردد.

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"  
  
$crystal = 12000000  
  
Config Porta = Output  
  
Dim A As Byte  
  
Do  
  
Incr A  
  
Set Porta.0  
  
Waitms 600  
  
Reset Porta.0  
  
Waitms 600  
  
Loop Until A = 10  
  
Toggle Porta  
  
End
```

< دستور FOR-NEXT :

فرم کلی دستورات FOR .. NEXT بصورت زیر می باشد .

```
]FOR var = start TO end [STEP VALUE  
  
برنامه  
  
NEXT var
```

Var بعنوان یک کانتر عمل می کند که start مقدار اولیه آن و end مقدار پایانی است و هر دو می توانند یک ثابت عددی یا متغیر عددی باشند . Value مقدار عددی step را نشان می دهد که می تواند مثبت یا منفی باشد . وجود نام var بعد از NEXT الزامی نیست .

```
$regfile = "m16def.dat"  
  
$crystal = 12000000  
  
Config Lcd = 16 * 2
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Byte
```

```
Dim B As Byte
```

```
Dim C As Integer
```

```
For A = 1 To 10 Step 2
```

```
Locate 1 , 1
```

```
Lcd A
```

```
Next A
```

```
For C = 10 To -5 Step -1
```

```
Locate 1 , 6
```

```
Lcd C
```

```
Next
```

```
For B = 1 To 10
```

```
Locate 2 , 1
```

```
Lcd B
```

```
Next
```

```
End
```

<دستور WHILE-WEND :

```
WHILE condition
```

```
statements
```

```
WEND
```

دستور العمل While-Wend تشکیل یک حلقه تکرار می دهد که تکرار این حلقه تا زمانی ادامه می یابد که عبارت بکاربرده شده نتیجه را FALSE کند و یا مقدار صفر بگیرد . دستور العمل while بصورت ورود به حلقه به شرط می باشد , بنابراین While ممکن است در حالتیابی اصلا اجرا نشود .

PEYMAN KHODDAMI

بخش statement تا وقتی که حاصل condition صفر یا FALSE نشده است تکرار خواهد شد .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte

A = 1

While A <10

Locate 1 , 1

Lcd A

Incr A

Waitms 600

Wend

End
```

<دستور if :

فرم کلی این دستور به شکل زیر است:

```
IF a = 1 THEN
...
ELSEIF a = 2 THEN
..
ELSEIF b1 > a THEN
...
ELSE
```

PEYMAN KHODDAMI

...

END IF

در صورتی که شرط اول برقرار باشد (a=1) دستورات زیر IF a = 1 THEN تا ELSEIF a = 2 THEN اجرا میشود، در صورتی که شرط اول برقرار نباشد و شرط دوم برقرار باشد (a=2) دستورات بین ELSEIF a = 2 THEN تا ELSEIF b1 > a THEN اجرا میشود، در صورتی که و اگر هیچ کدام از شرط ها برقرار نباشد دستورات بین ELSE تا END IF اجرا میشود. این دستور میتواند هر پیزی را چک کند (متغیر ها، واحد های حافظه، پین ها، پورت ها و...)، مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.2 , Db5 = Portd.3 , Db6 = Portd.4 , Db7 =  
Portd.5 , E = Portd.1 , Rs = Portd.0
```

```
Dim A As Byte
```

```
Do
```

```
Incr A : Wait 1
```

```
IF a = 1 THEN
```

```
Locate 1 , 1 : Lcd "CLAUSE1 true" : Locate 2 , 1 : Lcd "a=1"
```

```
Elseif A = 2 Then
```

```
Locate 1 , 1 : Lcd "CLAUSE2 true" : Locate 2 , 1 : Lcd "a=2"
```

```
Elseif A = 3 Then
```

```
Locate 1 , 1
```

```
Lcd "CLAUSE3 true"
```

```
Locate 2 , 1 : Lcd "a=3"
```

```
Elseif A = 4 Then
```

```
Locate 1 , 1
```

```
Lcd "CLAUSE4 true"
```

```
Locate 2 , 1 : Lcd "a=4"
```

```
Elseif A = 5 Then
```

PEYMAN KHODDAMI

```
Locate 1 , 1 : Lcd "CLAUSE5 true"

Locate 2 , 1 : Lcd "a=5"

Elseif A = 6 Then

Locate 1 , 1 : Lcd "CLAUSE6 true" : Locate 2 , 1 : Lcd "a=6"

Else

Locate 1 , 1 : Lcd "CLAUSE4 false" : Locate 2 , 1 : Lcd "a>6"

END IF

Loop

End
```

در مثال بالا ، هر یک ثانیه یک واحد به متغیر a افزوده میشود ، هنگامیکه مقدار آن 1 است روی سطر اول lcd عبارت "CLAUSE1 true" و روی سطر دوم lcd مقدار a نوشته میشود ، هنگامیکه مقدار آن 2 است روی سطر اول lcd عبارت "CLAUSE2 true" و روی سطر دوم lcd مقدار a نوشته میشود ،... هنگامی مقدار آن بزرگتر از 6 شد ، چون مقدار a در هیچ کدام از شرط ها صدق نمیکند ، دستور بین else و end if اجرا میشود . شما میتوانید از این دستور به فرم های مختلف ، یک شرطی یا چند شرطی استفاده کنید با این دستور در ادامه بیشتر آشنا میشویم.

<دستور #if :

این دستور مانند دستور if باست با این تفاوت که در این دستور میتوان شروط مربوط به کارکترها را اجرا کرد ، این دستور به فرم کلی زیر است:

```
#if Condition

case 1

#else

Case 2

#ENDIF
```

در صورتی که شرط درست باشد case 1 اجرا میشود و در صورت نادرست بودن case 2 اجرا میشود.مانند:

```
$regfile = "M16DEF.DAT"

$crystal = 8000000

#if Varexist( "S("

Dim A As Byte
```


PEYMAN KHODDAMI

```
#else
```

```
Dim S As Byte
```

```
#endif
```

```
End
```

<دستور : SELECT-CASE-END SELECT

این دستور تقریباً مانند دستور if است و به فرم زیر میباشد :

```
SELECT CASE var
```

```
    CASE test1 : statements1
```

```
[CASE test2 : statements2 ]
```

```
.
```

```
.
```

```
.
```

```
CASE ELSE : statements3
```

```
END SELECT
```

در این دستور یک متغیر چک میشود ، در صورتی که مقدار آن با شرط test1 برابر باشد statements1 اجرا میشود، در صورتیکه مقدار آن با test2 برابر باشد statements2 اجرا میشود و در صورتی که مقدار آن با هیچ یک از شروط برابر نباشد statements3 اجرا میشود . مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.2 , Db5 = Portd.3 , Db6 = Portd.4 , Db7 =  
Portd.5 , E = Portd.1 , Rs = Portd.0
```

```
Dim A As Byte
```

```
Do
```

```
Incr A : Wait 1
```

```
Select Case A
```

```
Case 1 : Locate 1 , 1
```

PEYMAN KHODDAMI

```
Lcd "CLAUSE true" : Locate 2 , 1 : Lcd "a=1"
```

Case 2:

```
Locate 1 , 1 : Lcd "CLAUSE true" : Locate 2 , 1 : Lcd "a=2"
```

Case 3 To 5:

```
Locate 1 , 1 : Lcd "CLAUSE true" : Locate 2 , 1 : Lcd "a=3-5"
```

```
Case Else : Locate 1 , 1
```

```
Lcd "CLAUSE false" : Locate 2 , 1 : Lcd "a>6"
```

```
End Select
```

```
Loop
```

```
End
```

در مثال بالا هر یک ثانیه یک واحد به متغیر a افزوده میشود، در صورتی که مقدار آن یک باشد شرط اول اجرا شده و بر روی lcd عبارات "CLAUSE true" و "a=1" نمایش داده میشود، در صورتی که مقدار آن دو باشد عبارت های "CLAUSE true" و "a=2" بر روی lcd نمایش داده میشود، در صورتی که مقدار آن بین 3 تا 5 (3 و 4 و 5) باشد عبارت های "CLAUSE true" و "a=3-5" بر روی lcd نمایش داده میشود،... در صورتی که هیچ کدام از شرط ها بر قرار نباشد دستورات بین Case Else و End Select اجرا میشود.

< خروج از حلقه :

با دستور exet میتوان از تمام حلقه های بالا خارج شد، فرم دستور برای هر حلقه در زیر آورده شده است:

```
EXIT FOR
```

```
EXIT DO
```

```
EXIT WHILE
```

```
EXIT SUB
```

```
EXIT FUNCTION
```

و مورد سوم برای خروج do-loop و مورد دوم برای خروج از حلقه ی FOR-NEXT مورد اول برای خروج از حلقه ی و مورد چهارم برای خروج از زیر برنامه و مورد اخر برای خروج از تابع است (دو مورد While-Wend از حلقه ی اخر در بخش های بعدی مورد بررسی قرار میگیرند). مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Portd.2 , Db5 = Portd.3 , Db6 = Portd.4 , Db7 =  
Portd.5 , E = Portd.1 , Rs = Portd.0
```

```
Dim A As Byte
```

```
For A = 1 To 100
```

```
Locate 1 , 1 : Lcd "Exit FOR-NEXT" : Locate 2 , 1 : Lcd "when A=4 a is " ; A
```

```
If A = 4 Then
```

```
Exit For
```

```
End If
```

```
Wait 1
```

```
Next
```

```
A = 1
```

```
Do
```

```
Locate 1 , 1 : Lcd "Exit do-loop" : Locate 2 , 1 : Lcd "when A=5 a is " ; A
```

```
Wait 1
```

```
If A = 5 Then
```

```
Exit Do
```

```
End If
```

```
Incr A
```

```
Loop
```

```
A = 1
```

```
While A < 6
```

```
Locate 1 , 1 : Lcd "Exit While-Wend" : Locate 2 , 1 : Lcd "when A=6 a is " ;  
A
```

```
Incr A
```

```
Wait 1
```

```
Wend
```

```
End
```

PEYMAN KHODDAMI

4-دستورات اجرایی :

برای اینکه بتوانید با کامپایلر بسکام راحت تر کار کنید تعدادی دستور به دستورات بیسیک برای avr اضافه شده است ، این دستورات کدی را برای ریختن روی میکرو تولید نمیکند و فقط جایگزین یک دستور میشود یا ان را خلاصه میکند ، استفاده از این دستورات کاملا اختیاری است ، مثلا شما میتونید با دستور \$PROG که در زیر توضیح داده میشود ، فیوز بیت دلخواه خود را پروگرام کنید ، میتونید این کار را از طریق پروگرامر انجام دهید و... این دستورات با علامت \$ شروع میشوند در زیر انها را توضیح میدهم: (بعضی از دستورات مانند دستور \$BAUD یا .. در بخش مربوطه گفته میشود)

دستورات : (دونقطه) (شيفت + ک)، (کما)(شيفت +و) ; (ويرگول)(حرف ک) _ (خط فاصله) (شيفت + منفي) (نقل قول تکی) (شيفت +گ) (البته در حالت کيبرد انگيسی)

: (دونقطه) (شيفت + ک): با استفاده از این دستور شما میتونید چندین دستور را پشت سر هم بنویسید. مانند:

```
Incr A : Wait 1 : Locate 1 , 1 : Lcd A : Locate 2 , 1 : Lcd B : Wait 2 : Cls  
: Incr B
```

, (کما)(شيفت +و): با این دستور میتونید چندین متغیر را در یک خط معرفی کنید (dim فقط برای متغیر اول آورده میشود) مانند:

```
Dim A As Byte , B As Integer , C As Word , Q As Bit
```

; (ويرگول)(حرف ک) : با این دستور میتونید چندین متغیر یا کارکتر را در یک بنویسید ، کارکتری بعدی از اولین فضای خالی بعد از کارکتر قبلی نمایش داده میشود مانند:

```
Lcd "a" ; A ; "b" ; B
```

_ (خط فاصله) (شيفت + منفي) با این دستور میتوان یک خط طولانی را نصف کرد. مانند:

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2_  
, Db7 = Portd.3 , E = Portd.4 , Rs = Portd.5
```

PEYMAN KHODDAMI

(نقل قول تکی) (شيفت +گ): با گذاشتن این دستور در برنامه ، کلیه حروف واعدادی که بعد از این دستور میآیند ، کامپایل نمیشوند ، شما میتوانید بعد از این دستور توضیحاتی را به برنامه اضافه کنید ، شما همچنین میتوانید به جای این علامت کلمه rem را به کار ببرید مانند: (یادستها به رنگ شبز در میاید.)

```
Start Adc 'roshan shodan adc
```

برای درک مطالب مثال زیر را شبیه سازی کنید:

```
$regfile = "m16def.dat" : $crystal = 8000000 : Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2_
, Db7 = Portd.3 , E = Portd.4 , Rs = Portd.5 ' rah andazi lcd
Dim A As Byte , B As Integer , C As Word , D As Byte
A = 1 : B = 2 : C = 5 : D = 7
Locate 1 , 1 : Lcd A ; " " ; B ; " " ; C ; " " ; D
Locate 2 , 1 : Lcd "a" ; A ; "b" ; B
End rem payan brname
```

دستور \$ASM :

با این دستور شما میتوانید در بین دستورات بیسیک ، دستورات اسمبلی به کار ببرید دستورات اسمبلی بین \$ASM و end Asm\$ قرار میگیرند . (برای دیدن دستورات اسمبلی که برای avr مورد استفاده قرار میگیرد در help بسکام گزینه Assembler mnemonics را جستجو کنید). مانند:

```
$regfile = "m16def.dat"
$crystal = 8000000
$asm
Ldi R24,1 ; load register R24 with the constant 1
St X,R24 ; store 1 into variable c
$end Asm
End
```

PEYMAN KHODDAMI

دستور \$DBG :

با نوشتن این دستور و کامپایلر پروگرام کردن آن در میکرو ، شما میتوانید توسط پروگرامر سریال برنامه داخل میکرو را اشکال زدایی کنید ، تنها کافی است این دستور را در برنامه خود به کار ببرید.

```
$DBG
```

دستور DEBUG :

این دستور به فرم کلی زیر است:

```
DEBUG ON | OFF | var
```

با این دستور میتوان تغییرات متغیر var را مشاهده کرد ، هنگامی که از این دستور استفاده میکنید ، متغیر var توسط پروگرامر سریال به پورا کالم فرستاده میشود و شما میتوانید آن را در محیط شبیه ساز مشاهده کنید، on و off برای شروع و پایان ارسال متغیر هستند. مانند:

```
$regfile = "m16def.dat" : $crystal = 8000000 : Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Byte
```

```
Do
```

```
Incr A : Wait 1 : Locate 1 , 1 : Lcd A
```

```
Debug On A
```

```
Loop
```

```
End
```

در مثال با لا متغیر a بهد از هر باز افزوده شدن به پورت کام فرستاده میشود .

دستور \$DEFAULT:

با این دستور میتوانید متغیر های تعریف شده را در حافظه های مختلف به صورت خود کار ذخیره کنید(ذخیره سازی ار اولین خانه خالی حافظه شروع میشود) ، این دستور به فرم کلی زیر است:

```
$DEFAULT = var
```

Var : میتواند یکی از گزینه های SRAM, XRAM, ERAM باشد ، مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

PEYMAN KHODDAMI

```
$default Eram
```

```
Dim A As Byte , B As Byte
```

```
$default Sram
```

```
Dim D As Byte
```

```
A = 1 : B = 1
```

```
D = A * B
```

```
End
```

در مثال بالا متغیرهای `ea` و `b` در حافظه `EEPROM` و متغیر `d` در حافظه `sram` ذخیره میشود (توجه داشته باشید که میکرو شما باید دارای حافظه‌های انتخاب شده باشد)

دستور `EXTERNAL`:

با این دستور شما میتوانید از لایبریهایی اسمبلی در بسکام استفاده کنید، با دستور `asm` شما به تعدادی از دستورات دسترسی دارید، با این دستور میتوانید لایبری‌های زبان اسمبلی را در پوشه `lib` موجود در محل نصب بسکام کپی کرده و در برنامه فراخوانی کنید. مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
$external Test
```

```
Declare Sub Test (byval X As Byte , Y As Byte)
```

```
Dim Z As Byte
```

```
Call Test(1 , Z)
```

```
End
```

دستور `MAP`:

با نوشتن این دستور در برنامه، گزارشی به نام `map` به انتهای فایل `report` افزوده میشود، این متن شماره خط‌های که در آنها کد تولید شده است را نمایش میدهد (برای دیدن فایل گزارش به آدرس: `Program>show result` بروید)

دستور `INCLUDE`:

با این دستور شما میتوانید برنامه دیگری را وارد برنامه خود کنید، برنامه دیگر باید در محل ذخیره برنامه اصلی موجود باشد، این دستور به فرم کلی زیر است:

```
$INCLUDE "file"
```

PEYMAN KHODDAMI

File: نام برنامه ای است که میخواهید آن را وارد کنید ، نام باید کامل وبا پسوند باشد مانند:

```
$regfile = "m16def.dat"  
  
$crystal = 12000000  
  
Config Lcd = 16 * 2  
  
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5  
  
$include "123.bas"  
  
End
```

برنامه ای که به نام 123 ذخیره شده در زیر آورده شده است:

```
Locate 1 , 1  
  
Lcd "12365"  
  
Locate 2 , 1  
  
Lcd "qwert"
```

دستور \$PROGRAMMER :

با این دستور شما میتوانید نوع پروگرامر مورد استفاده را ، در برنامه تعیین کنید ، این دستور به فرم کلی زیر است:

```
$PROGRAMMER = number
```

Value	Programmer
0	AVR-ISP programmer(old AN 910)
1	STK200/STK300
2	PG302
3	External programmer
4	Sample Electronics
5	Eddie Mc Mullen
6	KITSRUS K122
7	STK500
8	Universal MCS Interface
9	STK500 extended
10	Lawicel Bootloader
11	MCS USB
12	USB-ISP I
13	MCS Bootloader

Number : یکی از اعداد جدول روبرو است:

شما همچنین میتوانید از مسیر زیر در برنامه بسکام نوع پروگرامر را تغییر دهید :

```
Options> Programmer
```


PEYMAN KHODDAMI

برای اطلاعات بیشتر در مورد پروگرامر ها به قسمت ضمیمه ها مراجعه کنید .

دستور \$SIM :

هنگامی که این دستور را در برنامه به کار ببرید ، کلیه دستورات تاخیر غیر فعال میشوند ، این دستور هنگامی که از شبیه ساز داخلی بسکام استفاده میکنید کار برد دارد ، هنگامی که میخواهید برنامه را روی میکرو بریزید یا آن را با پروتوس شبیه سازی کنید این دستور را پاک کنید.

دستور \$LIB :

با این دستور شما میتونید از دیگر لایبری های که برای بسکام نوشته شده است استفاده کنید (مثلا لایبری Icd گرافیکی) این دستور به فرم کلی زیر است :

```
$LIB "libname"
```

libname1: نام لایبری میباشد که در محل ذخیره برنامه وجود دارد ، شما همچنین میتونید آن را در پوشه LIB موجود در محل نصب بسکام کپی کنید ، در این صورت دیگر به این دستور نیازی نیست ، مانند:

```
$lib"mylib.lbx"
```

دستور \$nocompile:

با نوشتن این دستور در برنامه ، برنامه کامپایل نمیشود (کد هگزی تولید نمیشود ، برنامه چک نمیشود ، برنامه اتوماتیک ذخیره نمیشود):

```
$nocompile
```

دستورات POPALL و PUSHALL:

دستور POPALL باعث میشود که بعد از ریست شدن میکرو همه رجیستر های حافظه به حالت پیشفرض برگردند و استفاده از دستور PUSHALL باعث میشود تا رجیستر ها ذخیره شود . (در حالت عادی بعد از ریست شدن میکرو کلیه رجیستر ها به حالت پیش فرض برمیگردند.) برای استفاده از این دستورات آن ها را در ب برنامه خود تایپ کنید.

دستورات دیگری نیز وجود دارد که با آنها در بخش راه اندازی امکانات جانبی آشنا میشویم.

<< زیر برنامه ها و فراخوانی توابع

PEYMAN KHODDAMI

< معرفی تابع DECLARE FUNCTION

از این دستور برای معرفی تابع در ابتدای برنامه استفاده می شود . زمانی که بخواهیم تابعی را معرفی کنیم بایستی تابع معرفی شده باشد . در صورت استفاده از تابع می بایستی یک داده برگردانده شود .

```
DECLARE FUNCTION TEST([ [BYREF/BYVAL]var as type1]) As type2
```

TEST نام تابع موردنظر است . انتقال داده بصورت BYVAL باعث می شود که یک کپی از متغیر به تابع فرستاده شود و در محتوای آن هیچ تغییری ایجاد نشود . ولی در حالت BYREF آدرس متغیر ارسال و تغییرات در آن اثر می گذارد و داده برگشتی در صورت انجام عملیات بر روی آن با مقدار اولیه خود برابر خواهد بود . در صورت عدم استفاده از گزینه [BYREF/BYVAL] بصورت پیش فرض داده بصورت BYREF فرستاده می شود . Type1 نوع داده ارسال شده و type2 نوع داده برگشتی است . که هر دو می توانند داده نوع BYTE , INTEGER , WORD , LONG ,STRING باشند .

مثال :

در مثال زیر I بصورت BYVAL فرستاده شده است بنابراین یک کپی از مقدار I به زیر تابع فرستاده می شود و هیچ تغییری در محتوای آن ایجاد نمی شود . S بصورت BYREF فرستاده می شود و تغیر در آن صورت می گیرد . فراخوانی تابع MYFUNCTION با K و Z از نوع داده INTEGER و STRING است و مقدار برگشتی از نوع INTEGER است که در متغیر T قرار می گیرد . شما می توانید در محدوده تابع یک متغیر محلی تعریف کنید .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Declare Function Myfunction (Byval I As Integer , S As String )As Integer

Dim K As Integer , Z As String*10, T As Integer

K = 5 : Z = "123 " : T = Myfunction(k , Z)

Locate 1 , 1

Lcd T

Locate 1 , 7
```

PEYMAN KHODDAMI

```
Lcd Z                                     'Bascom

Locate 2 , 1

Lcd K                                     '5

End

Function Myfunction (Byval I As Integer , S As String )As Integer

local P As Integer

:Functions

P = I * 5

I = 5

S = "Bascom"

T = P

Myfunction = T

End Function
```

< معرفی زیر برنامه DECLARE SUB :

از این دستور برای معرفی زیر برنامه استفاده می شود . زیر برنامه ای که قصد فراخوانی آن را داریم بایستی در ابتدای برنامه یا حداقل قبل از فراخوانی آن معرفی شده باشد .

```
DECLARE SUB TEST[( [BYREF/BYVAL] var as type)]
```

زیر برنامه برخلاف تابع مقداری بر نمی گرداند . در زمان ارسال داده بصورت BYREF آدرس داده به زیر برنامه فرستاده می شود و در محتوای آن تغییر ایجاد می شود . ولی در حالت BYVAL یک کپی از داده فرستاده می شود و به هیچ وجه در محتوای آن تغییری ایجاد نمی شود . TEST نام زیربرنامه و VAR نام متغیر ارسالی به زیر برنامه و TYPE نوع آن است که می تواند داده نوع BYTE , INTEGER, WORD ,STRING باشند .

برای نوشتن زیر برنامه ابتدا نام آنرا توسط دستور زیر تعریف کرده و سپس شروع به نوشتن زیربرنامه می کنیم .

```
SUB Name [ ( var1) ]
```

NAME نام زیربرنامه که باید توسط دستور Declare معرفی شده باشد و با دستور End Sub پایان می یابد .

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte , B1 As Byte , C As Byte

Declare Sub Test ( A As Byte)

A =1 : B1 = 2 : C = 3

Lcd A ; B1 ; C                                '123 will print

Call Test (B1)

End

Sub Test(a As Byte)

Locate 2 , 1

Lcd A ; B1 ; C                                '123 will print

End Sub
```

<تابع فراخوانی CALL :

توسط این دستور زیر برنامه یا تابعی را فراخوانی می کنیم .

```
CALL TEST( VAR1 , VAR2,....)
```

VAR1 , VAR2 متغیرهایی که به زیر برنامه انتقال می یابند , هستند . می توان زیر برنامه را بصورت زیر نیز فراخوانی کرد .

```
TEST VAR1 , VAR2
```

لازم بتذکر است که نام زیر برنامه قبل از فراخوانی آن , باید توسط دستور Declare فراخوانی شود. اگر خواهیم عدد ثابت را به زیر برنامه انتقال دهیم بایستی حتما با آرگومان BYVAL آن را انتقال دهیم .

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte , B As Byte

Declare Sub Test ( B1 As Byte , Byval B2 As Byte)

A =65

Call Test ( A , 5)

Test A , 5

Locate 1 , 1

lcd A                                     ' will print A = 10

End

Sub Test(b1 As Byte , Byval B2 As Byte)

B1 = 10

B2 = 15

Locate 1 , 8

Lcd B1

Locate 2 , 1

Lcd B2

End Sub
```

<بکارگیری متغیر محلی یا LOCAL :

از این دستور برای تعریف متغیر محلی در زیربرنامه استفاده می کنیم .

```
LOCAL VAR As Type
```

PEYMAN KHODDAMI

VAR نام متغیر و type نوع داده است که می توانند STRING , WORD , INTEGER , BYTE , SINGLE , LONG باشند نوع داده های ERAM , SRAM , XRAM و آرایه ها نمی توانند محلی تعریف شوند.

یک متغیر محلی یک متغیر موقت است که فقط در هنگام فراخوانی زیر برنامه مربوطه برای آن فضا در نظر گرفته می شود و با برگشت از زیر برنامه عمر متغیر (LIFE TIME) به اتمام می رسد .

تذکر متغیرهای بییتی نمی توانند بصورت محلی تعریف شوند .مانند

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Declare Sub Test2

Do

Call test2

Loop

End

Sub Test2

Local A As Byte

Incr A

Lcd A

End Sub
```

PEYMAN KHODDAMI

<< دستورات ریاضی و محاسباتی و تبدیل متغیر های ریاضی

< از عملگر های ریاضی زیر می توان در محیط BASCOM استفاده کرد و عملیات ریاضی را انجام داد.

* علامت ضرب

/ علامت تقسیم

+ علامت جمع

- علامت تفریق

. علامت ممیز

< علامت بزرگتر از

> علامت کوچکتر از

= علامت مساوی

^ علامت توان

=> علامت کوچکتر یا مساوی

=< علامت بزرگتر یا مساوی

<> علامت مخالف

مانند :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

PEYMAN KHODDAMI

```
Dim Q As Byte
```

```
Dim W As Byte
```

```
Dim E As Byte
```

```
Dim R As Byte
```

```
Dim T As Byte
```

```
Q = 2
```

```
W = Q + 3
```

```
E = W * Q
```

```
R = E ^ E
```

```
T = E - r
```

```
Lcd T 't=10
```

```
End
```

دستور SQR:

```
var = SQR( source )
```

این دستور جذر (ریشه دوم) متغیر یا عدد ثابت source را محاسبه کرده و نتیجه را در متغیر VAR قرار میدهد. مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single , A As Byte
```

```
Cls
```

```
X = Sqr(9(
```

```
Locate 1 , 1 'x =3
```


PEYMAN KHODDAMI

```
Lcd X
```

```
X = X * 27
```

```
A = Sqr(x(
```

```
Locate 2 , 1
```

```
Lcd A
```

```
' A =9
```

```
End
```

دستور SGN :

```
var = SGN( x )
```

این دستور علامت یک متغیر شناور را نشان میدهد ، در صورتی که متغیر منفی باشد ($X < 0$) مقدار VAR برابر با -1 و در صورت مثبت بودن متغیر ($X > 1$) مقدار VAR برابر با 1 میشود مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single , A As Single
```

```
Cls
```

```
X = -9
```

```
X = Sgn(x(
```

```
Locate 1 , 1
```

```
'x =0
```

```
Lcd X
```

```
X = 27
```

```
A = Sgn(x(
```

```
Locate 2 , 1
```

PEYMAN KHODDAMI

```
Lcd A                                     ' A =1
End                                         'End
$sim
```

دستور POWER:

```
var = POWER( source, raise )
```

این دستور فرم دیگر دستور $^$ (توان) است و حاصل source به توان raise را محاسبه کرده و حاصل را در متغیر var قرار میدهد مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single , A As Single , B As Single
```

```
Cls
```

```
A = 2.3
```

```
X = 10
```

```
B = A ^ X
```

```
Locate 1 , 1
```

```
Lcd B
```

```
B = Power(a , X(
```

```
Locate 2 , 1
```

```
Lcd B
```

```
End
```

< دستور ABS :

PEYMAN KHODDAMI

```
VAR =Abs (VAR2)
```

این دستور به معنای ریاضی $VAR = |VAR2|$ (قدرمطلق) است (اعداد منفی را به مثبت تبدیل میکند). مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Integer , C As Integer
```

```
A = -100
```

```
C = Abs(a) 'c=|a|
```

```
Lcd C 'C= 100
```

```
End
```

<دستور EXP :

```
Target = Exp (source)
```

Target برابر با e بتوان source است . Target متغیری از نوع داده SINGLE است . مانند :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single
```

```
X= Exp( 1.1)
```

```
Locate 1 , 1
```

```
lcd X 'x = 3.004166124
```

PEYMAN KHODDAMI

```
X = 1.1
```

```
X = Exp(x)
```

```
Locate 2 , 1
```

```
Lcd X 'x = 3.004166124
```

```
End
```

<دستور LOG: این دستور لگاریتم طبیعی یک داده از نوع SINGLE را برمی گرداند .

```
Target = Log (source)
```

لگاریتم متغیر یا ثابت source از نوع داده single گرفته می شود. ودر متغیر target قرار می گیرد . مانند :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single
```

```
X = Log(100)
```

```
Locate 1 , 1
```

```
Lcd X 'x = 4.6051
```

```
X = Log(1000)
```

```
Locate 2 , 1
```

```
Lcd X 'x = 6.9077
```

```
End
```

: دستور LOG10

```
Target = LOG10(source)
```

PEYMAN KHODDAMI

این دستور لگاریتم source که از نوع single یا double میباشد را در مینای 10 محاسبه کرده و نتیجه را در متغیر Target که از نوع جنس single یا double میباشد قرار میدهد. مانند:

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim X As Single

X = Log10(100)

Locate 1 , 1

Lcd X                                     'x = 2

X = Log10(1000)

Locate 2 , 1

Lcd X                                     'x = 3

End

$sim
```

<دستور RND :

این دستور یک عدد را تصادفی برمی گرداند .

```
VAR= RND (limit)
```

عدد تصادفی بین 0 و limit بدست آمده و در متغیر var قرار می گیرد . با هر بار استفاده از این دستور عدد مثبت تصادفی دیگری بدست خواهد آمد . مانند :

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Byte
```

```
Do
```

```
X = Rnd(100)
```

```
Lcd X
```

```
Waitms 500
```

```
Loop
```

```
End
```

دستور :Frac

```
var = FRAC( single )
```

این دستور مقدار اعشاری یک متغیر از جنس single را جدا میکند و در یک متغیر single دیگر (VAR) میریزد مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single , A As Single
```

```
X = 57.35456
```

```
Locate 1 , 1
```

```
Lcd Frac(x)
```

```
'x = 35456
```

```
X = X * 3
```

```
A = Frac(x(
```

```
Locate 2 , 1
```

PEYMAN KHODDAMI

```
Lcd A                                     ' A = 0636749
End                                         'End
$sim
```

دستور INT:

```
var = INT( source )
```

این دستور مقدار صحیح یک متغیر از جنس single یا double را محاسبه میکند و حاصل را در متغیر VAR قرار میدهد (محدوده متغیر VAR باید با محدوده جواب مناسب باشد)مانند:

```
$regfile = "m16def.dat"
$crystal = 12000000
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5
Dim X As Single , A As Byte
Cls
X = 57.35456
Locate 1 , 1
Lcd Int(x)                                 'x =57
X = X * 3
A = Int(x(
Locate 2 , 1
Lcd A                                     ' A = 172
End                                         'End
```

<< دستورات محاسبه نسبت های مثلثاتی:

PEYMAN KHODDAMI

```
var=sin\cos\tan(source)
```

این دستور سینوس یا کسینوس یا تانژانت ثابت یا متغیر source را در متغیر var از نوع SINGLE قرار می دهد . تمام دستورات مثلثاتی با رادیان کار می کنند و ورودی این دستور بایستی رادیان باشد .

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single
```

```
X = Cos(90)
```

```
Locate 1 , 1
```

```
Lcd X
```

```
X = Sin(90)
```

```
Locate 1 , 8
```

```
Lcd X
```

```
X = Tan(90)
```

```
Locate 2 , 1
```

```
Lcd X
```

```
End
```

< دستورات محاسبه نسبت‌های مثلثاتی هایپربولیک :

```
Var = COSH\sinh\tAnh( source)
```

این دستور کسینوس یا سینوس یا تانژانت هایپربولیک ثابت یا متغیر source را در متغیر var از نوع SINGLE قرار می دهد . تمام دستورات مثلثاتی با رادیان کار می کنند و ورودی این دستور بایستی رادیان باشد . مانند :

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim X As Single

Dim Q As Single

Q= .512

X = Cosh(q)

Locate 1 , 1

Lcd X

X = Sinh(q)

Locate 1 , 8

Lcd X

X = Tanh(q)

Locate 2 , 1

Lcd X

End
```

<معکوس توابع مثلثاتی(Arc)

```
Var = A sin\Acos\ATN( source)
```

این دستور آرک سینوس یا کسینوس یا تانژانت ثابت یا متغیر source را در متغیر var از نوع SINGLE قرار می دهد .
تمام دستورات مثلثاتی با رادیان کار می کنند و ورودی این دستور بایستی رادیان باشد . مانند :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

PEYMAN KHODDAMI

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Single
```

```
Dim Q As Single
```

```
Q= .512
```

```
X = Acos(q)
```

```
Locate 1 , 1
```

```
Lcd X
```

```
X = Asin(q)
```

```
Locate 1 , 8
```

```
Lcd X
```

```
X = Atn(q)
```

```
Locate 2 , 1: Lcd X
```

```
End
```

<تابع RAD2DEG

```
Var =RAD2DEG( single)
```

برای تبدیل رادیان به درجه از این دستور استفاده می شود .

رادیان single به درجه تبدیل می شود و در متغیر VAR از نوع داده SINGLE قرار می گیرد

<تابع DEG2RAD

```
Var =DEG2RAD( single)
```

برای تبدیل درجه به رادیان از این دستور استفاده می شود .

زاویه single به رادیان تبدیل می شود و در متغیر VAR از نوع داده SINGLE قرار می گیرد مانند:

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim X As Single

Dim Q As Single

Q = 180

(X = Deg2rad(q

Locate 1 , 1

Lcd X

Q = Rad2deg(x )

Locate 2 , 1

Lcd Q

End
```

<تابع ROUND

```
Var =ROUND( x)
```

متغیر یا داده X از نوع SINGLE روند شده و در متغیر VAR از نوع داده SINGLE قرار می گیرد .

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim X As Single
```

PEYMAN KHODDAMI

```
Dim Q As Single
```

```
X = 2.8
```

```
Q = 5.2
```

```
Locate 1 , 1
```

```
Lcd Round(x)
```

```
Locate 2 , 1
```

```
Lcd Round(q)
```

```
End
```

دستورات min و max:

با این دو دستور میتوان کمترین و بیشتر مقدار یک متغیر ارایه ای از جنس word یا BYTE را بدست آورد ، این دو دستور هبفرم کلی زیر هستند:

```
var1 = MIN(var2)
```

```
var1 = MAX(var2)
```

(var2): نام متغیر ارایه است و var یک متغیر دیگری است که نتیجه در آن ریخته میشود. مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Max(a(1((
```

```
B(1) = Min(b(2((
```

```
Locate 2 , 1
```

PEYMAN KHODDAMI

```
Lcd B(1(
```

```
Loop
```

```
End
```

دستور FIX :

با این دستور کوچکترین عدد صحیح نزدیک به داده اعشاری از نوع بدست میاید (جزء صحیح). این دستور به فرم کلی زیر است:

```
var = FIX( x )
```

x میتواند یک عدد اعشاری یا یک داده از نوع single باشد ، با اجرای دستور کوچکترین عدد نزدیک به داده در متغیر var ریخته میشود ، مثلا کوچکترین عدد صحیح نزدیک به 2.2 ، 2 است . مثال:

```
$regfile= "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Single , S As Single
```

```
Cls
```

```
A = -45.56
```

```
S = Fix(25.9(
```

```
A = Fix(a(
```

```
Locate 1 , 1: Lcd S
```

```
Locate 2 , 1: Lcd A
```

```
End
```

```
$sim
```

PEYMAN KHODDAMI

<<توابع تبدیل کدها و متغیر ها به یکدیگر

<دستور HEX

Var = Hex (x)

این دستور یک داده از نوع BYTE,INTEGER , WORD , LONG را به مقدار هگزادسیمال تبدیل می کند .

مقدار HEX متغیر یا ثابت X در متغیر VAR جای می گیرد .

<دستور Bin

Var = Bin(x)

این دستور یک داده از نوع BYTE,INTEGER , WORD , LONG را به مقدار باینری تبدیل می کند .

مقدار باینری متغیر یا ثابت X در متغیر VAR جای می گیرد .

<دستور STR

Var = STR (X)

با این دستور می توان یک متغیر عددی (X) را به رشته (VAR) تبدیل کرد .

<دستور VAL

Var = VAL (S)

با این دستور می توان یک رشته (S) را به متغیر عددی (VAR) تبدیل کرد .

<دستور HEXVAL

Var = HexVal (x)

این دستور یک داده هگزادسیمال را به مقدار عددی تبدیل می کند .

مقدار عددی داده هگزادسیمال X که می تواند LONG , WORD , INTEGER , BYTE باشد در متغیر VAR جای می گیرد

<دستور Binval

PEYMAN KHODDAMI

```
var = Binval (x)
```

این دستور یک داده باینری را به مقدار عددی تبدیل می کند .

مقدار عددی داده باینری X که می تواند LONG , WORD , INTEGER , BYTE باشد در متغیر VAR جای می گیرد

<دستور BCD و MAKEBCD :

```
Var1 = MAKEBCD (Var2)
```

این دستور متغیر یا ثابت var2 را تبدیل به مقدار BCD اش می کند و در متغیر var1 جای می دهد .

```
PRINT BCD( var )
```

```
LCD BCD( var)
```

با دو دستور بالا میتوان مقدار bcd یک متغیر را مستقیماً روی lcd نمایش داد یا آن را به پورت سریال فرستاد (کار پورت سریال در بخش های بعدی آمده است)

<دستور MAKEDEC :

```
Var1 = MAKEDEC (Var2)
```

برای تبدیل یک داده BCD نوع INTEGER , WORD , BYTE به مقدار DECIMAL از این دستور استفاده می شود . مقدار دسیمال متغیر یا ثابت var2 در متغیر var1 جای می گیرد .

<دستور MAKEINT

```
Varn = MAKEINT (LSB , MSB)
```

این دستور دو بایت را به هم متصل می کند و یک داده نوع WORD یا INTEGER می سازد که LSB بایت کم ارزش و MSB بایت پر ارزش متغیر دو بایتی Varn را تشکیل می دهد .

```
Varn = (256*MSB )+LSB
```

<دستور STRING

```
Var = STRING (m , n)
```

این دستور کد اسکی m را با تعداد تکرار n تبدیل به رشته کرده و در متغیر var قرار می دهد . در صورت قرار دادن m = 0 یک رشته بطول 255 کاراکتر تولید می شود و قرار دادن n = 0 قابل قبول نیست .

PEYMAN KHODDAMI

<دستور BIN2GREY

```
Var1 = BIN2GREY (Var2)
```

متغیر var2 که می تواند داده ای از نوع WORD , INTEGER , BYTE , LONG باشد به کد گری تبدیل شده و در متغیر VAR1 قرار می گیرد .

<دستور GREY2BIN

```
Var1 = grey2bin (Var2)
```

کد گری var2 به مقدار باینری تبدیل شده و در متغیر var1 که می تواند داده ای از نوع WORD , INTEGER , BYTE , LONG باشد قرار می گیرد . مثال برای موارد بالا :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim A As Byte
```

```
Dim S As String * 10
```

```
S = "ABC "
```

```
A = ASC(s)
```

```
Locate 1 , 1
```

```
Lcd A
```

```
S = Hex(a)
```

```
Locate 1 , 8
```

```
Lcd S
```

```
A = Hexval(a (
```

```
Locate 2 , 1
```


PEYMAN KHODDAMI

```
Lcd S  
A = 50  
A = Makebcd (A (  
Locate 2 , 8  
Lcd A  
End
```

8- راه اندازی امکانات جانبی

دستور :debounce

فرم کلی این دستور به شکل زیر است:

```
DEBOUNCE Px.y , state , label [ , SUB]
```

توسط این دستور پین x.y چک میشود و هنگامی که مقدار آن برابر با state شد cpu میکرو به label پرش میکند ،
گزینه SUB اختیاری است ، شما میتوانید از این گزینه برای پرش به یک زیر برنامه استفاده کنید.

به جای گزینه state میتوانید 0 یا 1 قرار دهید ، در صورتی که state صفر باشد ، هنگامی که پایه مورد نظر به گراند
متصل شد ، به برجسب مورد نظر پرش میشود و هنگامی که state یک باشد هنگامی که پایه به ولتاژ 5 ولت متصل شد
پرش صورت میگیرد(پینی که کلید به آن متصل است باید به عنوان ورودی تعریف شود). مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Portc = Input
```

```
W:
```

```
Locate 1 , 1
```

```
Lcd "pinc.0 ro 1 kon"
```

PEYMAN KHODDAMI

```
Debounce Pinc.0 , 1 , Q
```

```
Goto W
```

```
Q:
```

```
Locate 1 , 1
```

```
Lcd "pinc.1 ro 1 kon"
```

```
Debounce Pinc.1 , 1 , W
```

```
Goto Q
```

```
End
```

دستور PULSEOUT:

فرم کلی این دستور به شکل زیر است:

```
PULSEOUT PORT , PIN , PERIOD
```

با این دستور میتوان یک پالس بر روی پورت PORT و پایه PIN دلخواه با زمان تناوب PERIOD دلخواه بر حسب میکرو ثانیه ایجاد کرد (پینی که پالس بر روی آن ایجاد میشود باید به عنوان خروجی تعریف شود) (در صورتی که از حلقه استفاده نکنید ، دستور فقط یک بار اجرا میشود ، این پالس مربعی است (در واقع و وضعیت پایه از صفر به یک یا بالعکس تغییر میکند)).مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Portc.0 = Output
```

```
W:
```

```
Pulseout Portc , 0 , 60000
```

```
Goto W
```

```
End
```

PEYMAN KHODDAMI

دستور PULSEIN:

توسط این دستور میتوان زمان تناوب یک پالس مربعی را اندازه گرفت ، فرم کلی این دستور به شکل زیر است:

```
PULSEIN var , PINX , PIN , STATE
```

این دستور زمان تناوب پالس اعمال شده به پورت PINX و پین PIN دلخواه را در متغیر var که باید از جنس word باشد قرار میدهد.

شما میتوانید مشخص کنید که زمان بین از صفر به یک رفتن یا از یک به صفر رفتن پالس اندازه گرفته شود ، برای حالت اول به جای STATE صفر و برای حالت دوم به جای STATE یک قرار دهید مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Dim X As Word
```

```
Config Portc.0 = Input
```

```
Do
```

```
Pulsein X , Pinc , 0 , 1
```

```
Locate 1 , 1
```

```
Waitms 500
```

```
Lcd X
```

```
Loop
```

```
End
```

در مثال بالا زمان تناوب پالسی که به پین c.0 اعمال شده در هر 500 میلی ثانیه اندازه گرفته میشود و سپس بر روی lcd نمایش داده میشود.(زمان تناوب نباید از 65.535 میلی ثانیه بیشتر باشد ، این دستور از تایمر های میکرو استفاده نمیکند)

دستور SOUND :

PEYMAN KHODDAMI

توسط این دستور میتوان پالسی را بر روی یکی از پایه های میکرو ظاهر کرد ، فرم کلی دستور به شکل زیر است:

```
SOUND pin, duration, pulses
```

Pin نام پایه دلخواهی است که پالس از آن خارج میشود ، duration مشخص کننده تعداد پالس های خروجی است و pulses زمان تناوب پالس برحسب میکرو ثانیه است که حداکثر مقدار آن 65535 است.مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Portc.0 = Output
```

```
Do
```

```
Sound Pinc.0 , 10 , 60000
```

```
Loop
```

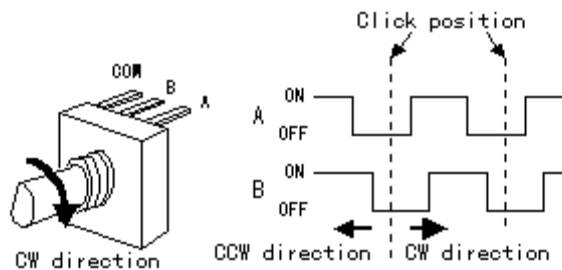
```
End
```

در مثال بالا پالس از پین c.0 خارج میشود ، زمان تناوب آن 60 میلی ثانیه است و 10 بار تکرار میشود . از این دستور معمولا برای راه اندازی بازر استفاده میشود.

دستور ENCODER :



ENCODER نوعی کلید دوطرفه میباشد که تصویر آن را در شکل روبرو مشاهده میکنید:



PEYMAN KHODDAMI

از این قطعه در کبردها و لوازم صوتی تصویری و ... برای کم و زیاد کردن صدا و نور و ... استفاده میشود. در واقع این قطعه از دوکلید تشکیل شده است ، هنگامی که شما سری را به سمت راست میچرخانید ، کلید سمت راست (که یک پایه آن به پایه وسط و پایه دیگر به پایه سمت راست متصل است) قطع و وصل میشود ، و هنگامی که سری را به سمت چپ میچرخانید کلید سمت چپ (که یک پایه آن به سر وسط و پایه دیگر به پین سمت چپ متصل است) قطع و وصل میشود ، با استفاده از دستور زیر میتوان عملیات مناسب با جهت چرخش (قطع و وصل شدن هر کلید) را انجام داد:

```
Var = ENCODER( pin1, pin2, LeftLabel, RightLabel , wait)
```

Var : یک متغیر از جنس دلخواه میباشد که به ازای پالس های فرد مقدار آن صفر و به ازای پالس های زوج مقدار آن یک است (مقدار آن حول صفر و یک تغییر میکند)

pin1 : نشان دهنده پایه ای است که پین 1 (چپ یا راست) انکدر به آن متصل میشود.

pin2 : نشان دهنده پایه ای است که پین 2 (چپ یا راست) انکدر به آن متصل میشود.

LeftLabel : نام برجسی است که در هنگام به چپ چرخیدن انکودر به آن پرش میشود. باز گشت از برجسب با دستور return آن جام میشود.

RightLabel : نام برجسی است که در هنگام به راست چرخیدن انکودر به آن پرش میشود. باز گشت از برجسب با دستور return آن جام میشود.

Wait : در صورتی که شما به جای این کلمه 1 بگذارید cpu میکرو بر روی این دستور متوقف میشود ، هنگامی که انکودر چرخید cpu به زیر برنامه میرود و بعد از انجام دستورات موجود در زیر برنامه دوباره روی این دستور متوقف شده و منتظر میماند تا پالسی اعمال شود و در صورتی که 0 قرار دهید cpu مدام در حلقه گردش میکند و اگر دستور انکودر در حلقه باشد آن را چک میکند و اگر نباشد که هیچ مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Portb = Input
```

```
Dim A As Byte , B As Byte , C As Word : Cls
```

PEYMAN KHODDAMI

```
Do
A = Encoder(pinb.0 , Pinb.1 , Q , W , 0(
Locate 1 , 1
Lcd A
Loop
End
Q:
Incr B
Locate 1 , 5
Lcd B
Return
W:
Incr C
Locate 1 , 11
Lcd C
Return
```

در مثال بالا پایه مشترک انکدر به گراند و پایه چپ ان به پین b.0 و پایه راست ان به پین b.1 متصل شده است .

دستور DTMFOUT :

توسط این دستور میتوانید پالس مناسب با یک عدد را تولید کرده و ان را روی خط تلفن سوار کنید. فرم کلی این دستور به شکل زیر است : این دستور در دو فرم استفاده میشود، فرم اول:

```
DTMFOUT number, duration
```

با این دستور شما میتوانید فقط یک شماره را به خط تلفن ارسال کنید ، متغیری که رقم دلخواه در ان به جای number گذاشته میشود و duration تاخیر زمانی بین ارسال این رقم و ارقام بعدی میباشد ، شما میتوانید به جای رقم یک متغیر قرار دهید ، اما متغیر باید بین 0 تا 15 باشد.مانند

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 4000000
```

```
Dim A As Byte
```

```
Do
```

```
A = 2
```

```
Dtmfout A , 50
```

```
A = 6
```

```
Dtmfout A , 50
```

```
A = 8
```

```
Dtmfout A , 50
```

```
A = 9
```

```
Dtmfout A , 50
```

```
A = 6
```

```
Dtmfout A , 50
```

```
A = 5
```

```
Dtmfout A , 50
```

```
A = 8
```

```
Dtmfout A , 50
```

```
A = 0
```

```
Dtmfout A , 50
```

```
Loop
```

```
End
```

فرم دوم:

در این حالت شما میتوانید شماره خود را در یک متغییر از جنس `x * string` که `x` تعداد شماره است و حداکثر آن 15 است، قرار دهید، با دستور زیر شماره های موجود در متغییر با تاخیر زمانی `duration` پشت سرهم گرفته میشوند

```
DTMFOUT string , duration
```

مثال:

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 4000000  
  
Dim A As String * 15  
  
A = "2696580"  
  
Do  
  
Dtmfout A , 500  
  
Loop  
  
End
```

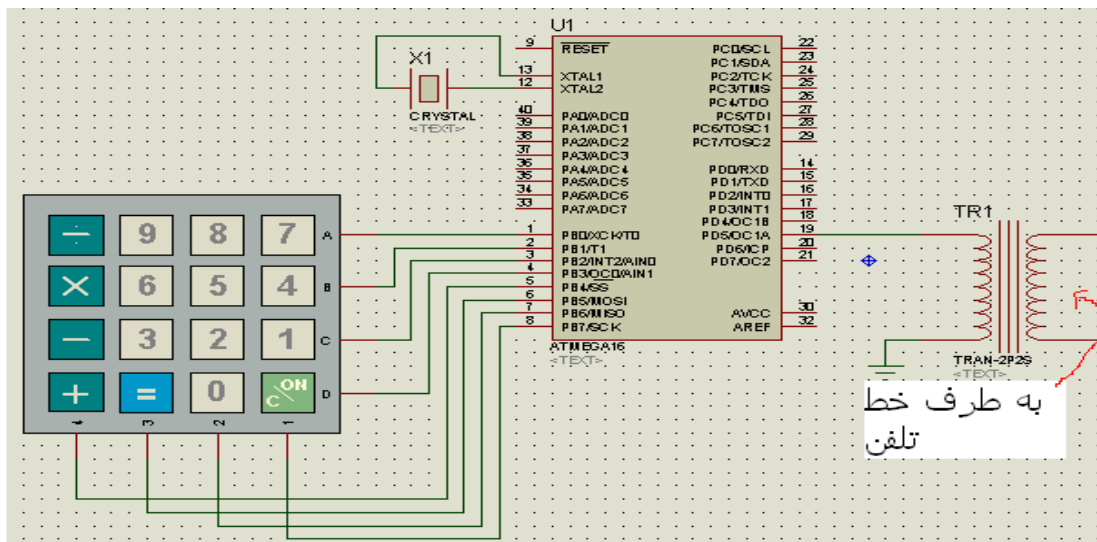
در این مثال شماره 2695680 مدام به خط تلفن ارسال میشود. مثال :

```
$regfile = "M16DEF.DAT"  
  
$crystal = 8000000  
  
Config Kbd = Portb  
  
Dim A As Byte  
  
Q:  
  
A = Getkbd()  
  
If A > 15 Then  
  
Goto Q  
  
End If  
  
Dtmfout A , 50  
  
Goto Q  
  
End
```

در این مثال عدد گرفته شده از کیبورد به پالس dtmf تبدیل شده و به خط تلفن ارسال میشود ، در صورتی که در بین فشردن کلید ها تاخیر زیادی رخ دهد خط تلفن اشغال میشود.

PEYMAN KHODDAMI

نکته: این دستور فقط با استفاده از کریستال های 4 تا 10 مگا هرتز جواب میدهد ، خروجی پالس dtmf پایه oc1a و گراند است (پایه شماره 19 مگا 16) ، در هنگام کار با این دستور مراقب ولتاژ خط تلفن باشید.(بهتر است از اپتوکوپلر یا ترانس ایزوله استفاده کنید تا میکرو آسیب نبیند)(ترانس ایزوله نوعی ترانس است که ورودی را از



کار با magnetic card (کارت های مغناطیسی):

ایا تاکنون کارت مغناطیسی دیده اید ؟ انها را با کارت های اعتباری و کارت های تلفن که روی خود یک تراشه دارند اشتباه نگیرید اشتباه نگیرید ، این کارت ها بدون تماس کار خود را انجام میدهند ، فقط کافی است شما ان را از شکاف یا روی دستگاه مرکزی (کارت خوان(شکل) رد کنید ، دستگاه کارت خوان با میدانی که ایجاد میکند ، کارت را تغذیه کرده و کارت کد موجود در حافظه ی خود را برای دستگاه می فرستد.ویژگی اصلی این کارت ها دارا یودن یک نوار سیاه رنگ است.کارت خوان این دستگاه که بانام ریدپر کارت مغناطیسی(magnetic card reader) در بازار موجود است . دارای 5 سیم به شرح زیر است :

- 1- قرمز = تغذیه 5 ولت کارت ریدر
- 2- مشکی = تغذیه گراند کارت
- 3- زرد = انتخاب تراشه
- 4- سبز = کلاک کارت ریدر
- 5- ابی = خروجی دیتا برای میکرو

با دستورات زیر میتوان کارت ریدر را برای میکرو معرفی کرد:

_mport Alias Piny

_mdata Alias X

PEYMAN KHODDAMI

_mcs Alias X

_mclock Alias X

Config Porty = Input

Porty = 255

READMAGCARD var , count , coding

_mport Alias Piny: با این دستور پورتی که کارت ریدر به آن متصل است مشخص میشود مانند: _mport Alias PinB

_mdata Alias X: نشان میدهد که پایه داده به کدام یک از پایه های میکرو متصل شده است ، X شماره پایه است که میتواند بین 0 تا 7 باشد (سیم ابی)

_mcs Alias X: نشان میدهد که پایه انتخاب به کدام یک از پایه های میکرو متصل شده است ، X شماره پایه است که میتواند بین 0 تا 7 باشد (سیم زرد)

_mclock Alias X: نشان میدهد که پایه کلاک به کدام یک از پایه های میکرو متصل شده است ، X شماره پایه است که میتواند بین 0 تا 7 باشد (سیم سبز)

Config Porty = Input: پورتی که کارت ریدر به آن متصل است حتما باید به عنوان ورودی تعریف شود

Porty = 255: تمام وردی های پورت را یک میکند (تا به فقط به صفر شدن واکنش نشان دهند)

با دستور زیر میتوان خروجی کارت ریدر را خواند:

READMAGCARD var , count , coding

Returned number	ISO characterT
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	hardware control
11	start byte
12	hardware control
13	separator
14	hardware control
15	stop byte

Var: در این متغیر بایت خوانده شده قرار میگیرد

Count: در این متغیر شماره بیت خوانده شده قرار میگیرد، شماره بیت طبق جدول روبرو است.

داده های خوانده شده در کنار هم قرار میگیرند و یک

رقم را به وجود میاورد ، این رقم ها میتواند رمز عبور، کد شناسایی ... باشد

PEYMAN KHODDAMI

Coding: بر روی کارت های مغناطیسی دارای سه خط وجود دارد ، در صورتی که شما از خط اول استفاده کنید باید به جای Coding رقم 7 را قرار دهید و در صورتی که از خط دوم یا سوم استفاده کنید باید به جای Coding رقم 5 را قرار دهید.مانند:

```
$regfile = "m16def.dat" : $crystal = 1000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Pina.0 , Db5 = Pina.1 , Db6 = Pina.2 , Db7 =
Pina.3 , Rs = Pina.4 , E = Pina.5

Dim A As Byte , B As Byte

_mport Alias Pinb

_mdata Alias 0

_mcs Alias 1

_mclock Alias 2

Config Portb = Input

Portb = 255 : Do

Reset Porta.0 : Locate 1 , 1 : Lcd "Insert your card"

Readmagcard A , B , 5

Select Case A

Case 12

Locate 1 , 1 : Lcd "welcome mr a" : Wait 2 : Cls

Case 13:

Locate 1 , 1 : Lcd "welcome mr b" : Set Porta.0 : Wait 5 : Cls

Case 14:

Locate 1 , 1 : Lcd "welcome mr c" : Set Porta.0 : Wait 5 : Cls

Case 15:

Locate 1 , 1 : Lcd "welcome mr d" : Set Porta.0 : Wait 5 : Cls
```

PEYMAN KHODDAMI

Case Else:

```
Locate 1 , 1 : Lcd "card not right" : Reset Porta.0 : Wait 5 : Cls
```

```
End Select : Loop
```

```
End
```

برنامه بالا مربوط به یک سیستم نگهبانی است ، فقط افرادی که دارای کارت میباشند و مقداریر حافظه کارت آنها در دستگاه ثبت شده میتوانند واردشود ،

<<LCD گرافیکی

قبلا در مورد lcd کارکنتری بحث شد، حال در مورد lcd گرافیکی بحث میشود.

lcd های گرافیکی در نمونه های مختلف در بازار وجود دارد ، این lcd ها دارای پایه های زیر میباشد.

1 - vss: پایه تغذیه lcd که به 0 ولت متصل میشود.

2 - vdd: پایه تغذیه lcd که به 5 ولت متصل میشود.

3 - d0 تا d7 دیتا پورت (dataport) این 8 پایه مربوط به دیتای lcd میباشد (lcd اطلاعات را از طریق این 8 پایه رد و بدل میکند) که به یکی از پورت های میکرو که در برنامه مشخص می شود متصل میشود .

4 - controlport: که شامل پایه های زیر است و به یکی از پورت های میکرو که در برنامه مشخص میشود متصل میشود. این پایه ها برای کنترل lcd به کار میروند .

- rst: پایه ریست (باز نشانی) lcd ، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

- ce: این پایه برای فعال کردن چیپ lcd است ، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

- cd: این پایه مشخص کننده ارسال کد یا دیتا است (بدین صورت که اگر این پایه 1 باشد lcd کد را میگیرد و اگر 0 باشد lcd دیتا را میگیرد (دیتا فرمانها می باشد و کد متن ها و اشکال است))، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

- wr: این پایه برای نوشتن در lcd است ، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

- rd: این پایه برای خوانده از lcd است ، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

PEYMAN KHODDAMI

- fs : این پایه برای مشخص کردن فونت lcd است ، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد .

5 - con یا vo : پایه کنترل کنتراست lcd است که با توجه به نوع lcd به vcc یا -vcc هر ولتاژدیگر متصل میشود.

با توجه به مطالب بالا پیکر بندی lcd گرافیکی در بسکام به صورت زیر است :

```
Config GRAPHLCD = type , DATAPORT = port, CONTROLPORT=port , CE = pin , CD = pin , WR = pin, RD=pin, RESET= pin, FS=pin, MODE = mode
```

type : نام lcd است که میتواند یکی از موارد زیر باشد :

.....128 * 240 , 48 * 160 , 64 * 128 , 128 * 128 , 64 * 240

```
Config GRAPHLCD =64*240
```

port : یکی از پورت های دلخواه میکرو است مانند :

```
DATAPORT = portd, CONTROLPORT=portb
```

pin : یکی از پایه های دلخواه پورتی است که در قسمت CONTROLPORT مشخص شده است: مثال:

```
Config Graphlcd = 240 * 128 , Dataport = Porta , Controlport = Portc , Ce = 2 , Cd = 3 , Wr = 0 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8
```

mode : مشخص کننده تعداد ستون متنی lcd است که میتواند 6 یا 8 باشد

شرح پایه ها در مثال با لا مطابق زیر است:

```
شماره پایه*****عمل اتصال*****نام پایه بر روی lcd
GND ***** GND*****1
GND *****GND*****2
V*****+5V5*****3
V- *****-9V potmeter9*****4
WR*****PORTC.0 *****5
RD*****PO RTC.1*****6
CE***** **PORTC.2***** 7
C/D*****PORTC.3***** 8
NC ***** not conneted*****9
```

PEYMAN KHODDAMI

```
RESET*****PORTC.4*****10
D0-D7*****porta *****18-11
FS *****PORTC.5*****19
NC *****not connected ***** 20
```

دستورات مربوط به lcd گرافیکی:

<دستور lcd :

با این دستور میتوان متن یا کارکتری را بر روی lcd نمایش داد مانند:

```
Lcd "MCS Electronics "
```

```
Lcd "Mdgdgsdsscs "
```

<دستور locate :

با این دستور میتوان متن یا کاراکتری را در مکان دلخواه بر روی lcd گرافیکی نمایش داد ، مانند:

```
Locate 16 , 1
```

```
Lcd "write this to the lower line "
```

```
Locate 16 , 5
```

```
Lcd "fgghfhghfhgjhz "
```

<دستور cls :

با این دستور تمام lcd پاک میشود . با استفاده از دستور cls Text می توان قسمت متنی lcd را پاک کرد و با دستور cls

graph می توان قسمت گرافیکی را پاک کرد ، مانند :

```
$regfile = "m16def.dat "
```

```
$crystal = 8000000
```

```
Config Graphlcd = 240 * 128 , Dataport = Porta , Controlport = Portc , Ce = 2
, Cd = 3 , Wr = 0 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8
```

```
Cls
```

```
Wait 1
```

PEYMAN KHODDAMI

```
Locate 1 , 1

Lcd "1nafar "

Locate 2 , 1

Lcd "/*-+234#$$% "()**&^^

Locate 3 , 1

Lcd "1234567890123456789012345678901234567890 "

Locate 16 , 1

Lcd "qwertyuiop "

Wait 2

Lcd "jklfjgfhfdh "

Locate 2 , 20

Lcd "546g5h574gh "

Locate 3 , 13

Lcd "hgf547g56jn4h57nj4gf45jh74fg8jm "

Locate 30 , 1

Lcd "qwertyuiop "

Wait 2

Cls Text

End
```

< دستور pset X , Y, value :

این دستور یک پیکسل را در مختصات x,y به ازای value = 255 روشن و به ازای value = 0 خاموش میکند ، مانند:

```
Pset 10 , 20 , 255

Pset 5, 127 , 255

Pset 10 , 20 , 0
```

PEYMAN KHODDAMI

Pset 5, 127 , 0

حداکثر مقدار x,y بستگی به تعداد پیکسل lcd گرافیکی دارد برای مثال در lcd 240*128 حداکثر مقدار $x=239$, $y=127$ است .

<دستور : CURSOR ON / OFF BLINK / NOBLINK

Lcd گرافیکی مانند lcd کاراکتری دارای یک مکان نما می باشد که با دستور زیر میتوان آن را روشن یا خاموش یا چشمک زن یا ثابت قرارداد :

Cursor On با این دستور مکان نما روشن می شود (در حالت عادی مکان نما روشن است).

Cursor off با این دستور مکان نما خاموش می شود.

Cursor blink با این دستور مکان نما چشمک می زند .

Cursor noblink با این دستور مکان نما دیگر چشمک نمی زند .

<دستور LINE(x0,y0) – (x1,y1), color

با این دستور میتوان در lcd یک خط کشید ، که $(x0,y0)$ پیکسل شروع خط و $(x1,y1)$ پیکسل پایان خط است و $color=255$ خط با رنگ مشکی و $color=0$ خطی با رنگ سفید رسم خواهد کرد .مانند:

```
$regfile = "m165def.dat "
```

```
$crystal = 8000000
```

```
Config Graphlcd = 240 * 128 , Dataport = Porta , Controlport = Portc , Ce = 2  
, Cd = 3 , Wr = 0 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8
```

```
Cls
```

```
Cursor Blink
```

```
Wait 1
```

```
Cursor On
```

```
Wait 1
```

```
Cursor Off
```


PEYMAN KHODDAMI

```
Locate 1 , 1
Lcd "MCS Electronics "
Locate 2 , 1 : Lcd "T6963c support "
Locate 3 , 1 : Lcd "1234567890123456789012345678901234567890 "
Locate 16 , 1 : Lcd "write this to the lower line "
Wait 2
Cls Text
Line(0 , 0) -(239 , 127) , 255
Line(0 , 127) -(239 , 0) , 255
Line(0 , 0) -(240 , 0) , 255
Line(0 , 127) -(239 , 127) , 255
Line(0 , 0) -(0 , 127) , 255
Line(239 , 0) -(239 , 127) , 255
Wait 3
Cls Graph
End
```

<دستور `CIRCLE(x0,y0) , radius, color` :

این دستور یک دایره بر روی lcd میکشد، $(x0,y0)$ مرکز دایره و `radius` شعاع دایره می باشد و `color=255` دایره با رنگ مشکی و `color=0` دایره با رنگ سفید (دایره را پاک میکند) رسم خواهد کرد .

<دستور `SHOWPIC x, y, label` :

این دستور یک عکس را بر روی lcd گرافیکی نمایش میدهد .

دیگر دستورات مانند lcd کارکنتری است ...

<<مراحل نمایش عکس بر روی lcd گرافیکی:

PEYMAN KHODDAMI

اگر عکس مورد نظر رنگی است ان را به محیط فتوشاپ برده و در انجا ان را به عکس سیاه و سفید تبدیل کنید سپس ان را با برنامه point و با پسوند BMP. و در اندازه استاندارد ذخیره کنید (اندازه صفحه نمایش LCD).

سپس از منوی TOOLS گزینه Graphic Converter را انتخاب کنید ، در پنجره باز شده گزینه load را بزنید و در پنجره باز شده عکس مورد نظر که با پسوند BMP ذخیره کردید ،باز کنید.

بعد گزینه save را بزنید و فایل را با نام دلخواه وبا پسوند BGF در کنار برنامه ذخیره کنید .

با استفاده از دستور SHOWPICE x, y , label عکس را در مختصات x, y نمایش دهید . label نام برجسیبی است که عکس مورد نظر در ان قرار میگیرد . برجسب "\$mcs.bgf" اشاره به عکس مورد نظر که در کنار برنامه اصلی قرار گرفته . مانند:

، پیکر بندی شده است ، نام تصویری که روی T6963c با چیپ راه انداز گرافیکی 240 * 128lcd در مثال زیر یک و محل اتصال انها در lcd است که در محل ذخیره برنامه ذخیره شده است در زیر نام پایهای qwe نمایش داده میشود lcd میکرو را مشاهده میفرمایید:

```
1- Gnd - - - - - - - - - - - - - - - - - - - - Gnd
2- Gnd - - - - - - - - - - - - - - - - - - - - Gnd
3- 5v - - - - - - - - - - - - - - - - - - - - + 5v
4 -9v - - - - - - - - - - - - - - - - - - - - -9v Potmeter
5- / Wr - - - - - - - - - - - - - - - - - - - - -portc.0
6- / Rd - - - - - - - - - - - - - - - - - - - - -portc.1
7- / Ce - - - - - - - - - - - - - - - - - - - - Portc.2
8- C / D - - - - - - - - - - - - - - - - - - - - Portc.3
9- Nc - - - - - - - - - - - - - - - - - - - - -not
10- Reset - - - - - - - - - - - - - - - - - - - -portc.4
11 -18 --d0 -d7----- Pa
19- Fs - - - - - - - - - - - - - - - - - - - - -portc.5
20- Nc -----Not Connected
```

برنامه :

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

PEYMAN KHODDAMI

```
Config Graphlcd = 240 * 128 , Dataport = Porta , Controlport = Portc , Ce = 2  
, Cd = 3 , Wr = 0 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8
```

```
Dim X As Byte , Y As Byte
```

```
Cls
```

```
Cursor Off
```

```
Wait 1
```

```
Locate 1 , 1
```

```
Lcd "qwertyuiokjjgf"
```

```
Locate 2 , 1 : Lcd "T6963c support"
```

```
Locate 3 , 1 : Lcd "123456789"
```

```
Wait 2
```

```
Cls Text
```

```
For X =0 To 140
```

```
Pset X , 20 , 255 ' set the pixel
```

```
Next
```

```
Wait 2
```

```
Showpic 0 , 0 , Plaatje
```

```
Wait 2
```

```
Cls Text ' clear the text
```

```
End
```

```
Plaatje:
```

```
$bgf "qwe.bgf"
```

در پوشه ضمیمه چندین مثال وجود دارد که میتوانید آنها را ببینید.

<< استفاده از کلید و کیبرد و کی پد و ... برای avr در محیط بسکام:

استفاده از کلید :

برای اتصال کلید به avr در محیط بسکام روش های مختلفی وجود دارد که در زیر به بیان هر یک میپردازیم .

نکته ، پینی که کلید به آن متصل میشود باید به عنوان ورودی تعریف شود .

<استفاده از دستور شرطی .if:

PEYMAN KHODDAMI

با دستور IF قبلا آشنا شدید اکنون یکی از قابلیت های را بررسی میکنیم :

کلید ها دارای دو پایه می باشد که یک پایه ان به یکی از پایه های میکرو و دیگری به vcc یا gnd متصل میشود با استفاده از دستور شرطی if میتوان فشرده شدن کلید را چک کرد ، هنگامی که کلید فشرده میشود پایه ای که کلید به ان متصل است صفر(به گراند متصل میشود) یا یک (به 5 ولت مثل میشود) میشود.

مثال : در این مثال یک سرکلید به vcc و سر دیگر به پورت C پایه 4متصل شده است

```
If Portc.4 = 1 Then  
  
Set Porta.0  
  
end IF
```

در مثال بالا ، اگر کلید فشرده شود portc.4 به vcc متصل میگردد(1میشود). شرط به این قرار است که اگر portc.4 یک شد porta.0 نیز یک شود در غیر این صورت porta.0 صفر بماند
روش دیگر استفاده از دستور Debounce است ، که قبلا گفته شده.

```
$regfile = "m16def.dat"  
  
$crystal = 12000000  
  
Config Lcd = 16 * 2  
  
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5  
  
Config Portc = Input  
  
Debounce Pinc.0 , 1 , A  
  
If Portc.1 = 1 Then  
  
Lcd "qwer "  
  
End If  
  
End  
  
A :  
  
Lcd "12345 "
```

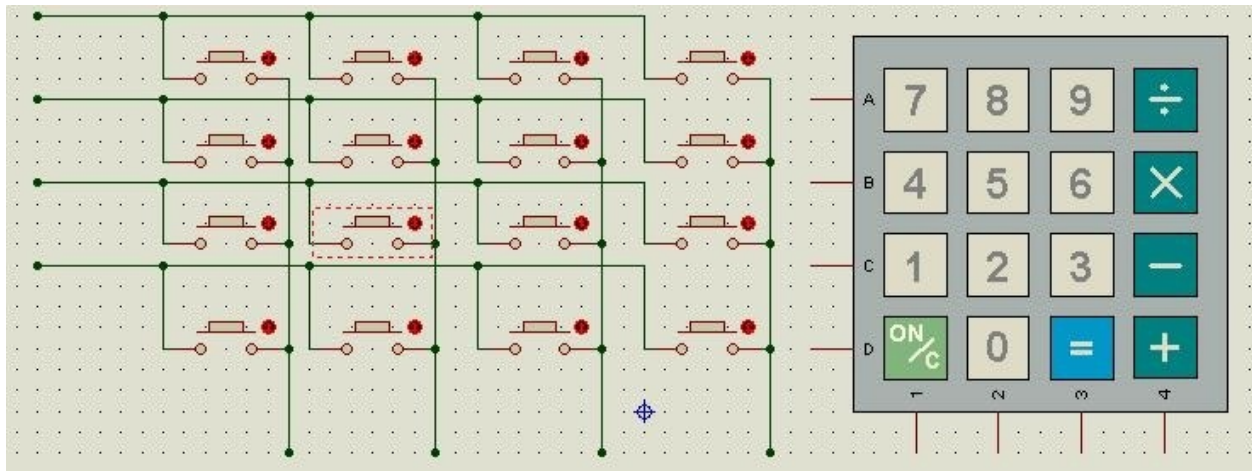
PEYMAN KHODDAMI

Return

در برنامه بالا اگر پین C.0 یک شود روی LCD عبارت 12345 نشان داده میشود و اگر پین C.1 یک شود روی LCD عبارت QWER نشان داده میشود.

< اتصال کی پد به avr

ابتدا در مورد چگونگی کار کی پد بحث کنیم . در زیر شکل این وسیله را مشاهده میکنید



گاهی نیاز از که چندین کلید را به میکرو متصل کنیم ، برای این کار دو راه وجود دارد که روش اول اتصال هر کلید به یک پایه و روش دوم استفاده از صفحه کلید ماتریسی است .

روش اول، چون تعداد زیادی از پایه های میکرو اشغال میشود، کمتر مورد استفاده قرار میگیرد .دستور بسکام برای استفاده از صفحه کلید ماتریسی که از این به بعد به آن kbd میگوییم به شرح زیر است :

```
CONFIG KBD = PORTx , DEBOUNCE= value
```

که portx ، پورتهی است که kbd به آن متصل میشود و DEBOUNCE تاخیر کلید است که بین 20 تا 255 میلی ثانیه است (هنگامی که کلید فشرده میشود ،برای لرزش دست چندین بار دو کنتاکت آن به هم برخورد میکنند و در نهایت ثابت میشوند اگر از دستور DEBOUNCE استفاده نشوند لرزش به منزله فشردن کلید است)

بعد از تعریف kbd در برنامه نوبت به استفاده از آن است برای این کار از دستور زیر استفاده می شود

```
A = Getkbd()
```

PEYMAN KHODDAMI

که a یک متغیر از جنس بایت است که عدد گرفته شده از kbd در آن گذاشته میشود و در صورتی که کلیدی فشرده نشود عدد 16 در داخل آن گذاشته می شود ، برای حذف عدد 16 می توان از دستور شرطی if استفاده کرد. مانند:

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Kbd = Portb,DEBOUNCE = 50

Dim A As Byte

:Q

()A = Getkbd

If A > 15 Then

Goto Q

End If

Locate 1 , 1

Lcd A

Goto Q

End
```

تحلیل برنامه :

دو خط اول برنامه مثل همیشه معرفی میکرو و کریستال است (که در اینجا از میکرو مگا 16 و کریستال 8 مگاهرتز استفاده شده است) .

در خط سوم و چهارم و پنجم lcd راه اندازی شده است (که در اینجا از lcd 16*2 استفاده شده و lcd به پورت d متصل است) .

در خط ششم kbd معرفی شده(که در اینجا kbd به پورت b متصل شده و DEBOUNCE پنجاه میلی ثانیه گرفته شده است) .

PEYMAN KHODDAMI

در خط هفتم یک متغیر از جنس بایت معرفی گردیده .

در خط هشتم یک برجسب به نام q قرار داده شده است .

در خط نهم عدد گرفته شده از kbd در متغیر a قرار میگیرد (در صورتی که هیچ کلیدی فشرده نشود مقدار 16 (a=16) در a ریخته میشود) .

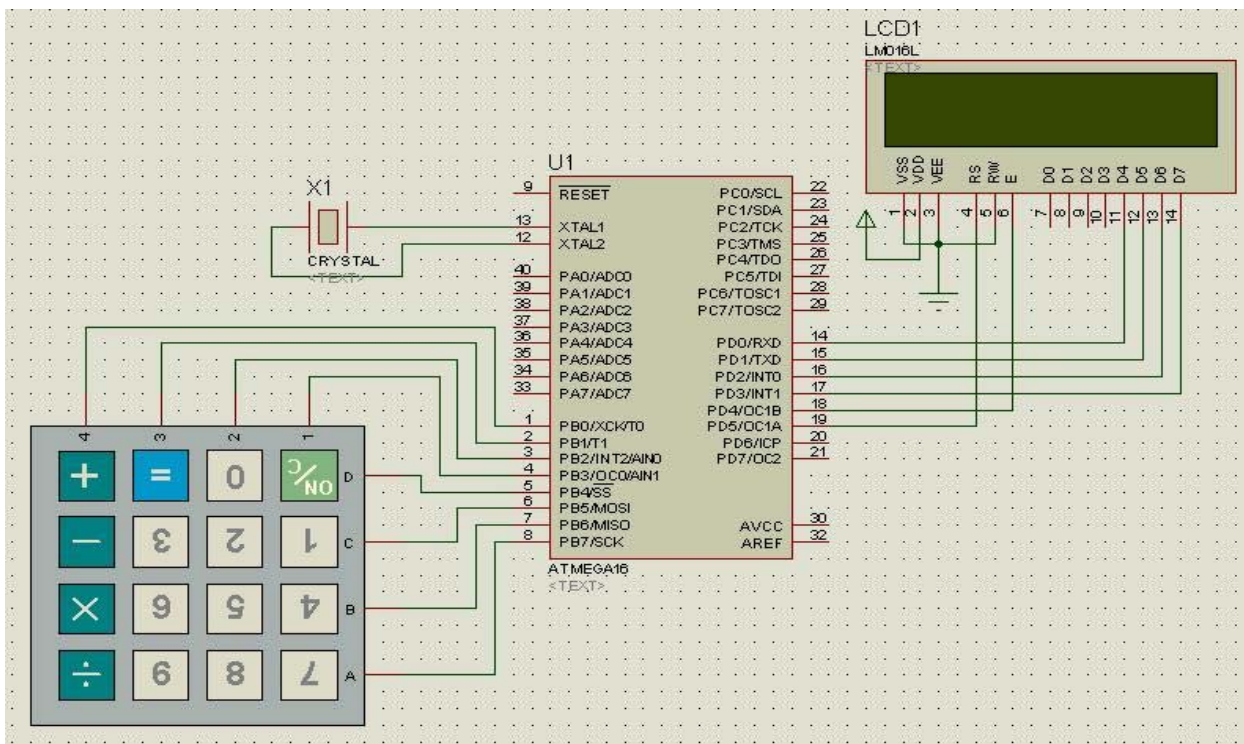
در خط ده و یازده و دوازده یک دستور شرطی قرار گرفته که این دستور شرطی میگوید: اگر a بزرگتر از 15 شد به برجسب q برش کن (در صورتی که شرط درست باشد دستورات بین if و endif اجرا میشود و اگر شرط درست نباشد برنامه از خط بعد از endif ادامه مییابد) .

در خط سیزده و چهارده مقدار a در سطر اول و ستون اول lcd نمایش داده میشود .

و در خط پانزده برنامه به برجسب q پرش میکند و مراحل قبل دوباره تکرار میگردد .

خط آخر برنامه همیشه end است .

مدار مورد استفاده :



PEYMAN KHODDAMI

در این مثال به ازای هر کلید در lcd یک نام نوشته میشود :

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Kbd = Portb , Debounce = 20

Dim A As Byte

:Q

()A = Getkbd

If A > 15 Then : Goto Q : End If

If A = 0 Then : Locate 1 , 1 : Lcd "qwer" : End If

If A = 1 Then : Locate 1 , 1 : Lcd "asdf" : End If

If A = 3 Then : Locate 1 , 1 : Lcd "FDFD" : End If

If A = 4 Then : Locate 1 , 1 : Lcd "aFDf" : End If

If A = 5 Then : Locate 1 , 1 : Lcd "SAKr" : End If

If A = 6 Then : Locate 1 , 1 : Lcd "1234" : End If

If A = 7 Then : Locate 1 , 1 : Lcd "5687" : End If

If A = 8 Then : Locate 1 , 1 : Lcd "7541" : End If

If A = 9 Then : Locate 1 , 1 : Lcd "IO12" : End If

If A = 10 Then : Locate 1 , 1 : Lcd "wqsf" : End If

If A = 11 Then : Locate 1 , 1 : Lcd "112k" : End If

If A = 12 Then : Locate 1 , 1 : Lcd "3,m5" : End If

If A = 13 Then : Locate 1 , 1 : Lcd "(" * "(" : End If
```


PEYMAN KHODDAMI

```
If A = 14 Then : Locate 1 , 1 : Lcd "*****" : End If
```

```
If A = 15 Then : Locate 1 , 1 : Lcd "++++" : End If
```

```
Locate 2 , 1 : Lcd A : Goto Q
```

```
End
```

در برنامه بالا برای اینکه حجم کمتری اشغال شود دستورات به صورت سطری نوشته شده‌اند ، شما می‌توانید با قرار دادن علامت دونقطه (:) در بین دو دستور آنها را در یک خط بنویسید ، همچنین با قرار دادن علامت ویرگول (;) چندین علامت را روی lcd در یک خط نمایش دهید ، همچنین با دستور کما (,) چندین متغیر را در یک خط معرفی کنید

برنامه بالا را با استفاده از جدول lookupstr مینویسیم:

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Kbd = Portb , Debounce = 20
```

```
Dim A As Byte
```

```
Dim B As String * 5
```

```
:Q
```

```
( )A = Getkbd
```

```
If A > 15 Then
```

```
Goto Q
```

```
End If
```

```
(B = Lookupstr(a , W
```

```
Locate 1 , 1: Lcd B
```

```
Locate 2 , 1: Lcd A
```

```
Goto Q
```

PEYMAN KHODDAMI

End

:W

```
Data "qwer" , "asdf" , "gdfh" , "jllh" , "hkn," , "yryh" , "bmn4" , "2452" ,  
"jkym" , "jytj" , "njhf" , "uyjn" , "dyr4" , "e5yh" , "9768" , "hnbh"
```

جدول lookupstr نیز مانند جدول lookup است اما جدول lookup برای بازگردانی اعداد و جدول lookupstr برای بازگردانی حروف به کار می‌رود .

شما جای پایه های kbd که به پایه میکرو متصل است را تغییر دهید و نتیجه را ببینید. همیشه نیاز نیست که ورودی یک کلید باشد بعضی وقت ها می توان صفر یا یک شدن یک پین را هم چک کرد.

بعضی وقت ها به کیبورد با تعداد کلید بیشتر نیاز است شما با دستور زیر 2 سطر دیگر به تعداد سطر های کیبورد اضافه کنید (در مجموع 24 کلید)

```
Config Kbd = Portx , Debounce = Value , Rows = 6 , Row5 = Pina.b, Row6 =  
Pina.b
```

X نام پورتی است که کیبورد به آن متصل شده است (4 سطر و 4 ستون اصلی).

Value مقدار تاخیر در فشردن کلید برای گرفتن لرزش است.

a.b نام پورت و پینی است که توسط دیگر به آن متصل شده اند.

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Kbd = Portb , Debounce = 20 , Rows = 6 , Row5 = Pina.0 , Row6 = Pina.1
```

```
Dim A As Byte
```

```
Q:
```

```
A = Getkbd()
```

```
If A > 16 Then : Goto Q : End If
```



PEYMAN KHODDAMI

Locate 1 , 1 : Lcd A : Goto Q

End

< اتصال کیبرد کامپیوتر به avr :

Table 1. AT Keyboard Connector Pin Assignments

AT Computer		
Signals	DIN41524, Female at Computer سوکت ۵ پین	6-pin Mini DIN PS2 Style سوکت ۶ پین
Clock	1	5
Data	2	1
nc	3	2,6
GND	4	3
+5V	5	4

اتصال کیبرد به avr در بسکام کار ساده ای میباشد ، چون تمامی توابع تعریف شده هستند و نیاز به نوشتن برنامه اضافه نمی باشد.

کیبرد کامپیوتر دارای 4 سیم میباشد که دو تا از آنها مربوط به تغذیه کیبرد و یکی دیتا کیبرد و دیگری کلاک (پالس همزمانی) میباشد

پیکربندی کیبرد به صورت زیر است :

```
CONFIG KEYBOARD = PINX.y , DATA = PINX.y , KEYDATA = table
```

که PINX.y یکی از پایه های دلخواه میکرو می باشد و table نام جدول کد های کیبرد است (از آنجا که کدهای گرفته شده از کیبرد هگز میباشد ، باید به وسیله یک جدول آنها را به کد اسکی برای نمایش تبدیل کرد).

مقدار گرفته شده از کیبرد بعد از تبدیل به کد اسکی باید در یک متغییر (b) ریخته شود که این کار با دستور زیر انجام میشود .

```
B = Getatkbd()
```

اگر کلیدی فشرده نشود مقدار صفر در b ریخته میشود .

به همین سبب در برنامه یک دستور شرط (if) قرار داده می شود تا موقعی که کلید فشرده نشود مقدار صفر در b قرار نگیرد.

PEYMAN KHODDAMI

برای درک بیشتر موضوع به مثال زیر توجه کنید

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Keyboard = Pind.2 , Data = Pind.4 , Keydata = Keydata

Dim B As Byte

Do

  ()B = Getatkbd

  If B > 0 Then

    (Lcd String(1 , B

  End If

Loop

End

:Keydata

normal keys lower case'

Data 0 , 0 , 0 , 0 , 0 , 200 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , &H5E , 0

Data 0 , 0 , 0 , 0 , 0 , 113 , 49 , 0 , 0 , 0 , 122 , 115 , 97 , 119 , 50 , 0

Data 0 , 99 , 120 , 100 , 101 , 52 , 51 , 0 , 0 , 32 , 118 , 102 , 116 , 114 , 53 , 0

Data 0 , 110 , 98 , 104 , 103 , 121 , 54 , 7 , 8 , 44 , 109 , 106 , 117 , 55 , 56 , 0

Data 0 , 44 , 107 , 105 , 111 , 48 , 57 , 0 , 0 , 46 , 45 , 108 , 48 , 112 , 43 , 0

Data 0 , 0 , 0 , 0 , 0 , 92 , 0 , 0 , 0 , 0 , 13 , 0 , 0 , 92 , 0 , 0

Data 0 , 60 , 0 , 0 , 0 , 0 , 8 , 0 , 0 , 49 , 0 , 52 , 55 , 0 , 0 , 0

Data 48 , 44 , 50 , 53 , 54 , 56 , 0 , 0 , 0 , 43 , 51 , 45 , 42 , 57 , 0 , 0
```

PEYMAN KHODDAMI

shifted keys UPPER case'

Data 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0

Data 0 , 0 , 0 , 0 , 0 , 81 , 33 , 0 , 0 , 0 , 90 , 83 , 65 , 87 , 34 , 0

Data 0 , 67 , 88 , 68 , 69 , 0 , 35 , 0 , 0 , 32 , 86 , 70 , 84 , 82 , 37 , 0

Data 0 , 78 , 66 , 72 , 71 , 89 , 38 , 0 , 0 , 76 , 77 , 74 , 85 , 47 , 40 , 0

Data 0 , 59 , 75 , 73 , 79 , 61 , 41 , 0 , 0 , 58 , 95 , 76 , 48 , 80 , 63 , 0

Data 0 , 0 , 0 , 0 , 0 , 96 , 0 , 0 , 0 , 0 , 13 , 94 , 0 , 42 , 0 , 0

Data 0 , 62 , 0 , 0 , 0 , 8 , 0 , 0 , 49 , 0 , 52 , 55 , 0 , 0 , 0 , 0

Data 48 , 44 , 50 , 53 , 54 , 56 , 0 , 0 , 0 , 43 , 51 , 45 , 42 , 57 , 0 , 0

PEYMAN KHODDAMI

<< اتصال avr به عنوان کیبرد به کامپیوتر:

"نکته مهم: اتصال avr به عنوان کیبرد به کامپیوتر، که در بسکام به نام Atemu نام گذاری شده رایگان نبوده و کامپایلر در هنگام کامپایل کردن برنامه های مربوطه (که در زیر دو مورد ان موجود است) از شما خطای عدم وجود لایبری را میگیرد، شما باید در help بسکام گزینه ی AT_EMULATOR را جستجو کنید، در صفحه پیدا شده طریقه تهیه لایبری آمده است."

شما میتوانید بر عکس حالات با لا را با چند دستور ساده زیر انجام دهید، یعنی یک میکرو avr را به پورت کیبرد که در پشت کامپیوتر قرار دارد، متصل کنید و اطلاعات مورد نیاز را برای کامپیوتر ارسال کنید، راه اندازی این مورد به صورت زیر است:

```
CONFIG Atemu = int , DATA = data, CLOCK=clock
```

Int: شما میتوانید از وقفه صفر یا یک برای این مورد استفاده کنید (int0 یا int1)

Data نام پایه ای از میکرو است که سیم دیتای پورت کیبرد کامپیوتر (که شکل ان را در مبحث قبل مشاهده فرمودید) به ان متصل میشود. (سیم دیتای سوکت کیبرد باید به پایه ورودی وقفه متصل شود) شما مجازید از دوپایه int0 و int1 (پایه 16 و 17 میکرو مگا 16) استفاده کنید))

Clock: نام پایه ای از میکرو است که سیم کلاک پورت کیبرد کامپیوتر به ان متصل میشود. با دستور زیر میتوانید کد های دلخواه را به کیبرد را به کامپیوتر ارسال کنید

```
SENDSCANKBD label | var
```

Label نام برجسیبی است که کد در ان قرار دارد (شما میتوانید چند کد را مانند مثال پشت سر هم نوشته و انها را به کامپیوتر بفرستید)

PEYMAN KHODDAMI

Var : شما همچنین می‌توانید یک متغیر را به کامپیوتر بفرستید. در زیر نام کلید های صفحه کلید و کدی که هر یک می‌سازند را مشاهده می‌فرمایید:

KEY	MAKE	BREAK	KEY	MAKE	BREAK	KEY	MAKE	BREAK
A	1C	F0,1C	9	46	F0,46	[54	F0,54
B	32	F0,32	^	0E	F0,0E	INSERT	E0,70	E0,F0,70
C	21	F0,21	-	4E	F0,4E	HOME	E0,6C	E0,F0,6C
D	23	F0,23	=	55	F0,55	PG UP	E0,7D	E0,F0,7D
E	24	F0,24	\	5D	F0,5D	DELETE	E0,71	E0,F0,71
F	2B	F0,2B	BKSP	66	F0,66	END	E0,69	E0,F0,69
G	34	F0,34	SPACE	29	F0,29	PG DN	E0,7A	E0,F0,7A
H	33	F0,33	TAB	0D	F0,0D	U ARROW	E0,75	E0,F0,75
I	43	F0,43	CAPS	58	F0,58	L ARROW	E0,6B	E0,F0,6B
J	3B	F0,3B	L SHFT	12	F0,12	D ARROW	E0,72	E0,F0,72
K	42	F0,42	L CTRL	14	F0,14	R ARROW	E0,74	E0,F0,74
L	4B	F0,4B	L GUI	E0,1F	E0,F0,1F	NUM	77	F0,77
M	3A	F0,3A	L ALT	11	F0,11	KP /	E0,4A	E0,F0,4A
N	31	F0,31	R SHFT	59	F0,59	KP *	7C	F0,7C
O	44	F0,44	R CTRL	E0,14	E0,F0,14	KP -	7B	F0,7B
P	4D	F0,4D	R GUI	E0,27	E0,F0,27	KP +	79	F0,79
Q	15	F0,15	R ALT	E0,11	E0,F0,11	KP EN	E0,5A	E0,F0,5A
R	2D	F0,2D	APPS	E0,2F	E0,F0,2F	KP .	71	F0,71
S	1B	F0,1B	ENTER	5A	F0,5A	KP 0	70	F0,70
T	2C	F0,2C	ESC	76	F0,76	KP 1	69	F0,69
U	3C	F0,3C	F1	05	F0,05	KP 2	72	F0,72
V	2A	F0,2A	F2	06	F0,06	KP 3	7A	F0,7A
W	1D	F0,1D	F3	04	F0,04	KP 4	6B	F0,6B
X	22	F0,22	F4	0C	F0,0C	KP 5	73	F0,73
Y	35	F0,35	F5	03	F0,03	KP 6	74	F0,74
Z	1A	F0,1A	F6	0B	F0,0B	KP 7	6C	F0,6C
0	45	F0,45	F7	83	F0,83	KP 8	75	F0,75
1	16	F0,16	F8	0A	F0,0A	KP 9	7D	F0,7D
2	1E	F0,1E	F9	01	F0,01]	5B	F0,5B
3	26	F0,26	F10	09	F0,09	;	4C	F0,4C
4	25	F0,25	F11	78	F0,78	'	52	F0,52
5	2E	F0,2E	F12	07	F0,07	,	41	F0,41
6	36	F0,36	PRNT	E0,12,	E0,F0,	.	49	F0,49
			SCRN	E0,7C	7C,E0,			
					F0,12			
7	3D	F0,3D	SCROLL	7E	F0,7E	/	4A	F0,4A
8	3E	F0,3E	PAUSE	E1,14,77,	-NONE-			
				E1,F0,14,				
				F0,77				

در مثال زیر می‌خواهیم عبارت "1nafar" را به کامپیوتر بفرستیم ، بعد از ساخت سخت افزار و اتصال آن به کامپیوتر در صورتی که برنامه Notepad یا word را باز کنید ، عبارت مذکور در آن نوشته می‌شود.

مانند تمامی برنامه ها ابتدا میکرو و کریستال را معرفی می‌کنیم:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

PEYMAN KHODDAMI

قدمی بعدی فعال سازی وقفه سراسری (هنگامی که می‌خواهیم از وقفه استفاده کنیم باید آن را فعال کنیم ، با دستور که نام برده میشود تمامی وقفه ها فعال میشوند) است:

```
Enable Interrupts
```

مرحله بعدی معرفی پایه های میکرو است که باید به پورت کیبرد موجود در پشت کامپیوتر متصل شود:

```
Config Atemu = Int1 , Data = Pind.3 , Clock = Pinb.0
```

در حالت بالا پایه کلاک پورت کیبرد به پورت b.0 و پایه دیتا آن به ورودی وقفه 1 (پورت d.3) متصل است ،(در صوتی که از وقفه 0 استفاده شود ، پایه دیتا باید به پورت d.2 متصل شود)

در مرحله بعد یک حلقه ایجاد میکنیم تا میکرو مدام عبارت را به کامپیوتر ارسال کند:

```
Do
```

و در نهایت با دستور زیر برجسیبی که در آن عبارت "1nafar" وجود دارد به کامپیوتر ارسال میشود:

```
Sendscankbd data1
```

یک تاخیر زمانی ، برای اینکه صفحه مدام پر نشود:

```
Wait 2
```

پایان حلقه و پایان برنامه:

```
Loop
```

```
End
```

و در نهایت ایجاد برجسب:

```
Data1:
```

```
Data 18 , &H16 , &HF0 , &H16 , &H31 , &HF0 , &H31 , &H1C , &HF0 , &H1C , &H2B  
, &HF0 , &H2B , &H1C , &HF0 , &H1C , &H2D , &HF0 , &H2D
```

```
'data tedad ersal ,1 ,n , a ,  
f , a r
```

اولین داده جدول تعداد بایت ارسالی است ، در اینجا تمامی بایت ها (که تعداد آنها 18 تاست) با هم ارسال شده اند (منظور پشت سر هم است) ، شما میتوانید مانند جدول lookup ، جدول را ادرس دهی کنید و داده های مورد نظر را بفرستید ،

PEYMAN KHODDAMI

توجه داشته باشید که تعداد باید نباید از 3 کمتر باشد ، در این صورت داده کامل منتقل نمیشود .داده های بعدی ، اعداد و حروف میباشند ، مثلا برای عدد 1 باید ، مطابق جدول بالا (نام کلید های صفحه کلید و کدی که هر یک میسازند) کد

ارسال H16 , &HF0 , &H16 & ارسال شود و برای کلید DELETE باید کد &H71 , &HF0 , &HE0 , &H71 & ارسال شود و در زیر کد مربوط به کلید های مدیاپلیر و دیگر کلید های موجود بر روی صفحه کلید را مشاهده میفرمایید:

Key	Make Code	Break Code
Power	E0, 37	E0, F0, 37
Sleep	E0, 3F	E0, F0, 3F
Wake	E0, 5E	E0, F0, 5E

Key	Make Code	Break Code
Next Track	E0, 4D	E0, F0, 4D
Previous Track	E0, 15	E0, F0, 15
Stop	E0, 3B	E0, F0, 3B
Play/Pause	E0, 34	E0, F0, 34
Mute	E0, 23	E0, F0, 23
Volume Up	E0, 32	E0, F0, 32
Volume Down	E0, 21	E0, F0, 21
Media Select	E0, 50	E0, F0, 50
E-Mail	E0, 48	E0, F0, 48
Calculator	E0, 2B	E0, F0, 2B
My Computer	E0, 40	E0, F0, 40
WWW Search	E0, 10	E0, F0, 10
WWW Home	E0, 3A	E0, F0, 3A
WWW Back	E0, 38	E0, F0, 38
WWW Forward	E0, 30	E0, F0, 30
WWW Stop	E0, 28	E0, F0, 28
WWW Refresh	E0, 20	E0, F0, 20
WWW Favorites	E0, 18	E0, F0, 18

میخواهیم با یک کلید صدا را کم و با کلید دیگر صدا را زیاد کنیم برای این کار چند روش وجود دارد که یکی از آنها در زیر آمده است:

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 8000000

Enable Interrupts

Config Atemu = Int1 , Data = Pind.3 , Clock = Pinb.0

Waitms 500

Config Porta = Input

Q:

Debounce Pina.0 , 1 , Vup

Debounce Pina.0 , 1 , Vdown

Goto Q

Vup:

Sendscankbd Data1

Waitms 500

Goto Q

Vdown:

Sendscankbd Data2

Waitms 500

Goto Q

End

Data 5 , &HE0 , &H32 , &HE0 , &HF0 , &H32

Data2:

Data 5 , &HE0 , &H21 , &HE0 , &HF0 , &H21
```

PEYMAN KHODDAMI

<< اتصال avr به عنوان موس به کامپیوتر:

"نکته مهم: اتصال avr به عنوان موس به کامپیوتر، که در بسکام به نام PS2EMU نام گذاری شده رایگان نبوده و کامپایلر در هنگام کامپایل کردن برنامه های مربوطه (که در زیر دو مورد آن موجود است) از شما خطای عدم وجود لایبری را میگیرد، شما باید در help بسکام گزینه ی AT_EMULATOR را جستجو کنید، در صفحه پیدا شده طریقه تهیه لایبری آمده است."

<p>Male</p> <p>(Plug)</p> <p>فیش نری</p>	<p>Female</p> <p>(Socket)</p> <p>سوکت مادگی</p>	<p>موسهای AT/XT</p> <p>5-pin DIN (AT/XT):</p> <p>1 - Clock کلاک 2 - Data داده 3 - Not اتصال بدون Implementec 4 - Ground گراند 5 - +5v تغذیه +</p>	<p>Male</p> <p>(Plug)</p> <p>فیش نری</p>	<p>Female</p> <p>(Socket)</p> <p>سوکت مادگی</p>	<p>موس های PS/2</p> <p>6-pin Mini-DIN (PS/2):</p> <p>1 - Data داده 2 - Not اتصال بدون Implemented 3 - Ground گراند 4 - +5v تغذیه + 5 - Clock کلاک 6 - Not Implementec</p>
--	---	---	--	---	---

راه اندازی این مورد با دستور زیر انجام میشود:

```
CONFIG PS2EMU= int , DATA = data, CLOCK=clock
```

Int : شما میتوانید از وقفه صفر یا یک برای این مورد استفاده کنید (int0 یا int1)

Data : نام پایه ای از میکرو است که سیم دیتای پورت موس کامپیوتر (که شکل آن را در بالا مشاهده فرمودید) به آن متصل میشود. (سیم دیتای سوکت موس باید به پایه ورودی وقفه متصل شود) شما مجازید از دوپایه int0 و int1 (پایه 16 و 17 میکرو مگا 16) استفاده کنید))

Clock : نام پایه ای از میکرو است که سیم کلاک پورت موس کامپیوتر به آن متصل میشود.

با دستور زیر میتوانید کد های دلخواه را به کامپیوتر ارسال کنید

```
PS2MOUSEXY X , Y, button
```

X و Y مختصات مکان توقف اشاره گر موس است که از -255 تا 255 میباشد.

Button : عدد معادل کلید فشرده شده میباشد (در روی موس 3 کلید اصلی وجود دارد : کلید راست ، کلید چپ، کلید وسط) که رقم معادل هر کلید در زیر آمده است

0 – no buttons pressed

صفر : هیچ کلیدی فشرده نشده است

1- left button pressed

یک : کلید سمت چپ فشرده شده است

PEYMAN KHODDAMI

2- right button pressed

دو: کلید سمت راست فشرده شده است

4- middle button pressed

چهار: کلید وسط فشرده شده است

با دستور زیر نیز میتوانید کد های مربوط به موس را از جدول بخوانید و به کامپیوتر ارسال کنید:

Sendscan lable

Lable : نام برجسیبی است که کد ها در آن وجود دارد.مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Enable Interrupts
```

```
Config Ps2emu = Int0 , Data = Pind.2 , Clock = Pind.1
```

```
Config Porta = Input
```

```
Dim X As Integer , Y As Integer , Button As Byte
```

```
Q:
```

```
Debounce Pina.0 , 1 , Mright
```

```
Debounce Pina.1 , 1 , Mleft
```

```
Debounce Pina.2 , 1 , Mup
```

```
Debounce Pina.3 , 1 , Mdown
```

```
Debounce Pina.4 , 1 , Rpessed
```

```
Debounce Pina.5 , 1 , Lpressed
```

```
Debounce Pina.6 , 1 , Mpressed
```

```
Ps2mousexy X , Y , Button
```

```
Goto Q
```

```
Mright:
```

```
Incr X : Waitms 500 : Goto Q
```

```
Mleft:
```

PEYMAN KHODDAMI

```
Decr X : Waitms 500 : Goto Q
```

```
Mup:
```

```
Incr Y : Waitms 500 : Goto Q
```

```
Mdown:
```

```
Decr Y : Waitms 500 : Goto Q
```

```
Rpressed:
```

```
If Button = 2 Then : Button = 0 : Else : Button = 2 : End If
```

```
Waitms 500 : Goto Q
```

```
Lpressed:
```

```
If Button = 1 Then : Button = 0 : Else : Button = 1 : End If
```

```
Waitms 500 : Goto Q
```

```
Mpressed:
```

```
If Button = 4 Then : Button = 0 : Else : Button = 4 : End If
```

```
Waitms 500 : Goto Q
```

```
End
```

مثال:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Enable Interrupts
```

```
Config Ps2emu = Int0 , Data = Pind.2 , Clock = Pind.1
```

```
Dim A As Byte
```

```
Q:
```

```
Incr A
```

```
Wait 1
```

PEYMAN KHODDAMI

Select Case A

```
Case 1 : Ps2mousexy 0 , 10 , 0          ' up
Case 2 : Ps2mousexy 0 , -10 , 0         ' down
Case 3 : Ps2mousexy -10 , 0 , 0        ' left
Case 4 : Ps2mousexy 10 , 0 , 0         ' right
Case 5 : Ps2mousexy 0 , 0 , 1          ' left button
pressed
Ps2mousexy 0 , 0 , 0                    ' left button
released
Case 6 : Sendscan W

End Select

Goto Q
```

W:

```
Data 3 , &H08 , &H00 , &H01
```

نکته: شما مجازید فقط یک موس یا کیبورد به کامپیوتر خود متصل کنید ، اتصال سخت افزار بیشتر ممکن است به کامپیوتر شما آسیب بزند.

<< مبدل آنالوگ به دیجیتال(adc):

گاهی نیاز است که یک کمیت بیرونی (مانند دما و شدت صدا و شدت نور...) اندازه گیری شود ، برای اینکار از وسیله ای به نام سنسور استفاده میشود. سنسور ها مقدار یک کمیت آنالوگ را به ولتاژ یا جریان تبدیل میکند، سپس این ولتاژ آنالوگ به مبدل آنالوگ به دیجیتال میکرو داده میشود و مبدل آنالوگ به دیجیتال مقدار ولتاژ را به کمیت دیجیتال متناظر تبدیل میکند ، سپس این مقدار دیجیتال با اعمال ریاضی به مقدار عددی متناظر تبدیل میشود و روی Icd یا 7سگمنت نمایش داده میشود .

حداکثر ولتاژی که مبدل آنالوگ به دیجیتال ، که از این به بعد به آن adc میگوییم میتواند اندازه بگیرد برابر با vcc است و اگر ولتاژ اعمالی از vcc بیتر شود مبدل آنالوگ به دیجیتال آسیب ببیند(معمولا بیشترین ولتاژ ورودی که به adc اعمال میکنند 4.5 ولت است) و کمترین ولتاژ اعمالی برابر با gnd است . adc به ازای ولتاژ 5 ولت عدد 1023 و به ازای صفر ولت عدد صفر را در متغیر مربوطه قرار می دهد .

PEYMAN KHODDAMI

Adc با دستور زیر راه اندازی میشود :

```
Config adc = single/free, PRESCALER = AUTO, REFERENCE = opt
```

گزینه های single/free : در حالتی که single انتخاب شود مقدار دیجیتال سیگنال آنالوگ توسط دستور getadc در یک متغیر از جنس word ریخته میشود در حالتی که free انتخاب شود مقدار دیجیتال سیگنال آنالوگ در رجیستر مربوط به adc ریخته میشود.

PRESCALER : این گزینه کلاک adc را مخص میکند و در حالتی که AUTO انتخاب شود کامپایر با توجه به کریستال انتخاب شده بهترین کلاک را در نظر میگیرد ، موارد دیگر برای کلاک عبارتند از 2 و 4 و 8 و 15 و 32 و 64 یا 128 است که به جای گزینه AUTO نوشته میشود .

REFERENCE: در صورتی که بخواهید از یک ولتاژ مرجع استفاده کنید این گزینه را بنویسید(در صورت عدم استفاده از این دستور ولتاژ مرجع زمین است و نیازی به نوشتن این دستور نیست) .

(این امکان فقط در بعضی از میکرو ها وجود دارد) opt میتواند یکی از موارد زیر باشد:

Off : در این حالت ولتاژ مرجع داخلی خاموش شده و از ولتاژ مرجع بر روی پایه aref استفاده میشود.

Avcc: در این حالت ولتاژ پایه avcc به عنوان ولتاژ مرجع در نظر گرفته میشود .

Internal: در این حالت از ولتاژ مرجع داخلی 2.65 ولت استفاده میشود .

بعد از راه اندازی adc نوبت به استفاده از آن است برای اینکار با دستور start adc روشن شده و شروع به نمونه برداری از سیگنال آنالوگ موجود بر روی پایه اش میکند و آن را به مقدار دیجیتال تغییر میدهد ، این مقدار دیجیتال با دستور زیر در یک متغیر از جنس word ریخته میشود

```
var = GETADC(channel)
```

var یک متغیر از جنس word میباشد

Channel : شماره : شماره adc است که سیگنال آنالوگ به آن اعمال شده .مانند:

```
$regfile = "m8def.dat "
```

```
$crystal = 8000000
```

```
Config Lcd = 16 * 2
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , Rs = Pinb.0 , E = Pinb.1
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word
```

```
Start adc
```

```
Q :
```

```
A = Getadc(1 (
```

```
Locate 1 , 1
```

```
Lcd A
```

```
Goto Q
```

```
End
```

در خط های اول lcd و adc پیکر بندی شده است در خط دهم مقدار آنالوگ داده شده به پایه 24 میکرو (portc.1) بعد از تبدیل به مقدار دیجیتال در متغیر a ریخته میشود و سپس این متغیر در سطر اول و ستون اول lcd به نمایش در می آید

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 4
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word , B As Word , C As Word , D As Word , E As Word , F As Word , G  
As Word , H As Word
```

```
Cls
```

```
Q:
```

```
A = Getadc(0) : Locate 1 , 1 : Lcd A
```

```
B = Getadc(1) : Locate 1 , 8 : Lcd B
```


PEYMAN KHODDAMI

```
C = Getadc(2) : Locate 2 , 1 : Lcd C
D = Getadc(3) : Locate 2 , 8 : Lcd D
E = Getadc(4) : Locate 3 , 1 : Lcd E
F = Getadc(5) : Locate 3 , 8 : Lcd F
G = Getadc(6) : Locate 4 , 1 : Lcd G
H = Getadc(7) : Locate 4 , 8 : Lcd H

Goto Q

End
```

در این مثال از میکرو مگا 16 و $16*4$ lcd استفاده شده است ، میکرو مگا 16 دارای 8 کانال adc میباشد ، که در این مثال از همه adc های این میکرو استفاده شده است .

ممکن است این سوال برای شما پیش بیاید که: adc نمی تواند بیشتر از 5 ولت را اندازه گیری کند ، ما ولتاژ های بالا تر را چگونه اندازه بگیریم؟ برای اندازه گیری ولتاژ های زیاد شما باید آن را با مقاومت کم کنید ، فرض کنید یک ولتاژ از 0 تا 200 ولت متغییر دارید و میخواهید آن را با میکرو اندازه بگیرید ، شما باید این ولتاژ متغییر را به 0 تا 5 ولت تبدیل کنید... این موارد در ضمیمه ها آورده شده است.

<< سروو موتور:

سروو ها نوعی موتور بسیار پر قدرت است که میتواند حول یک زاویه خاص با دقت بالا بچرخد ، از سروو ها برای بازوی ربات و باز و بسته کردن درب و دریچه و ... استفاده میشود، سروو ها دارای سه سیم میباشد که دوتای آنها تغذیه و دیگری برای کنترل درجه چرخش مورد استفاده قرار میگیرد (در صورتی که سروو شما دارای 5 سیم است ، دو تا از آنها برای تغذیه مدار داخلی سروو و دوتای دیگر برای تغذیه خود سروو و سیم اخر برای کنترل میباشد ، بهتر است دیتا شیت سروو را از فروشنده دریافت کنید) راه اندازی سروو در بسکام با دستور زیر انجام میشود:

```
Config Servos = X , Servo1 = Portx.y , Servo2 = Portx.y , Servon = Portx.y ,
Reload = R1
```

Config Servos = X : نشان دهنده تعداد سروو های استفاده شده میباشد که بیشترین تعداد میتواند 14 باشد.(به جای x تعداد گذاشته میشود مانند : Config Servos = 3)

PEYMAN KHODDAMI

Servo1 = Portx.y : پایه کنترل سروو به یکی از بین های میکرو که با Portx.y مشخص شده متصل میشود.

Reload : نشان دهنده زمانی است که میکرو دوباره اطلاعات مربوط به سروو ها را روی پین مورد نظر میفرستد(این زمان برحسب میکروثانیه است)

نکته: این دستور از تایمر صفر برای راه اندازی (تولید زمان روشن بودن سروو) استفاده میکند و هنگامی که سروو را پیکربندی کردید دیگر نمیتوانید ا تایمر صفر استفاده کنید.

بعد از پیکر بندی سروو نوبت به راه اندازی ان است برای این کار از دستور زیر استفاده میشود:

```
Servo(x) = y
```

X شماره سروو است که میتواند از 1 تا 14 باشد و y ضرب در 10 زمان روشن بودن سروو را نشان میدهد، که میتواند یک متغیر یا عدد صحیح باشد.مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Servos = 14 , Servo1 = Portd.0 , Servo2 = Portd.1 , Servo3 = Portd.2_  
, Servo4 = Portd.3 , Servo5 = Portd.4 , Servo6 = Portd.5 , Servo7 = Portd.6_  
, Servo8 = Portd.7 , Servo9 = Portc.7 , Servo10 = Portc.6 , Servo11 =  
Portc.5_
```

```
, Servo12 = Portc.4 , Servo13 = Portc.3 , Servo14 = Portc.2 , Reload = 100
```

```
Config Portd = Output , Portc = Output
```

```
Enable Interrupts
```

```
Dim A As Word
```

```
A=10
```

```
Do
```

```
Servo(1) = 1 : Wait 1
```

```
Servo(2) = 5 : Wait 1
```

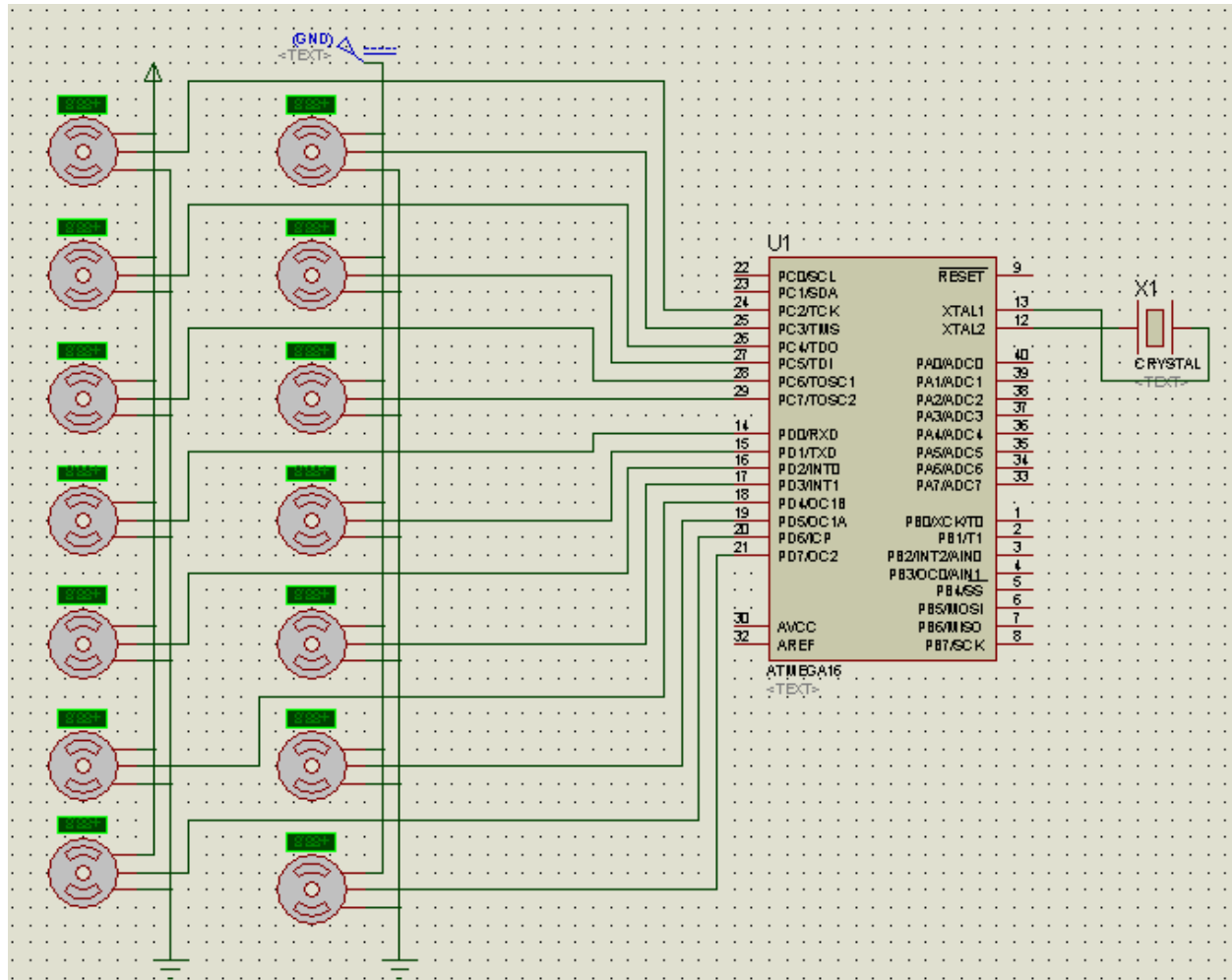
```
Servo(3) = a : Wait 1
```

PEYMAN KHODDAMI

```
Servo(4) = 15 : Wait 1  
Servo(5) = 20 : Wait 1  
Servo(6) = 25 : Wait 1  
Servo(7) = 30 : Wait 1  
Servo(8) = 35 : Wait 1  
Servo(9) = 40 : Wait 1  
Servo(10) = 45 : Wait 1  
Servo(11) = 50 : Wait 1  
Servo(12) = 55 : Wait 1  
Servo(13) = 60 : Wait 1  
Servo(14) = 65 : Wait 1  
  
Loop  
  
End
```

در مثال بالا تعداد 14 عدد سروو به میکرو مگا 16 متصل شده است ، مدت زمان روشن بودن سروو ها به ترتیب از سروو 1 به این شرح است: 10 و 50 و 100 و 150 و 200 و 250 و 300 و 350 و 400 و 450 و 500 و 550 و 600 و 650 میکرو ثانیه است. (یک سروو فقط در یک زاویه خاص میچرخد ، مثلا از 0 تا 270 درجه یا از 0 تا 360 درجه، زمان که در بالا گفته شد ، بیان میکند که چقدر طول میکشد تا سروو به یک زاویه خاص برسد ، برای فهمیدن این زمان باید به دیتاشیت سروو مراجعه کنید ، مثلا نوعی سروو در هر 10 میکرو ثانیه 1 درجه حرکت میکند و زاویه چرخش آن از 0 تا 10 است، بنابراین برای رسیدن به زاویه 120 درجه باید به جای y عدد 120 را قرار دهیم، در صورتی که به جای y عدد بیشتر از 180 قرار دهیم سروو روی 180 قفل میشود.بیشتر مقداری که به جای y میتوانید قرار دهید 255 است.مدار مثال بالا:

PEYMAN KHODDAMI



<< راه اندازی WATCHDOG:

WATCHDOG یکی از تایمر های میکرو است که میتواند تا یک زمان خاص بشمارد و میکرو را ریست کند ، این تایمر میتواند تا 8 زمان 16، 32، 64 ، 128 ، 256 ، 512 ، 1024 و 2048 و در بعضی از میکروها 8192، 4096 میلی ثانیه بشمارد ، بعد از سپری شدن زمان میکرو ریست میشود و برنامه دوباره از ابتدا اجرا میشود ، راه اندازی WATCHDOG به فرم زیر است :

```
CONFIG WATCHDOG = time
```

Time : یکی از زمان های گفته شده در بالا میباشد (16، 32، 64 ، 128 ، 256 ، 512 ، 1024 و 2048 و در بعضی از میکروها 8192، 4096 میلی ثانیه)

با دستور Start Watchdog تایمر شروع به شمارش میکند و پس از سپری شدن زمان میکرو ریست میشود. مانند:

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 8000000  
  
Config Porta.0 = Input  
  
Config Porta.1 = Output  
  
Config Watchdog = 1024  
  
Do  
  
If Pina.0 = 0 Then  
  
Set Porta.1  
  
Else  
  
Start Watchdog  
  
End If  
  
Loop  
  
End
```

در مثال بالا پایه a.0 (که در حالت عادی 1 است) مدام چک میشود و در صورتی که پایه 0 شود میکرو بعد از 1024 میلی ثانیه ریست میشود.

<<راه اندازی وقفه های خارجی :

با استفاده از دستور زیر میتوان منابع وقفه خارجی را راه اندازی کرد :

```
CONFIG INTx = state
```

INTx : نام پایه وقفه است که در اکثر میکرو ها به نام INT0 و INT1 و در بعضی از میکرو ها مانند مگا 64 به نام INT0 تا INT7 موجود است .

State : نوع پالس اعمالی برای فعال شده وقفه را معین میکند ، State میتواند یکی از موارد زیر باشد:

Falling : با اعمال یک پالس پایین روند (یک به صفر) به پایه مورد نظر وقفه فعال میشود

RISING : با اعمال یک پالس بالا رونده (صفر به یک) به پایه مورد نظر وقفه فعال میشود.

LOW LEVEL : با اعمال سطح صفر به پایه مورد نظر وقفه فعال میشود.

PEYMAN KHODDAMI

سپس با دستور `Enable Interrupts` وقفه سراسری و با دستور `Enable Intx` وقفه پیکر بندی شده فعال میشود.

و در نهایت شما میتوانید با دستور `On Intx lable` ، به هنگام اعمال پالس به `lable` مورد نظر پرش کنید .

با فعال شدن وقفه ، پایه های ورودی وقفه در همه حالت ها چک میشود و نیازی به آوردن دستورات در حلقه اصلی و دیگر حلقه ها نیست. باز گشت از زیر بزنامه با دستور `Return` انجام میشود. مانند:

```
$regfile = "m16def.dat" : $crystal = 8000000

Config Porta = Output

Config Int0 = Falling

Config Int1 = Rising

Enable Int0

Enable Int1

Enable Interrupts

On Int0 Q

On Int1 W

Do

Set Porta.2 : Waitms 500 : Reset Porta.2 : Waitms 500

Loop

End

Q:

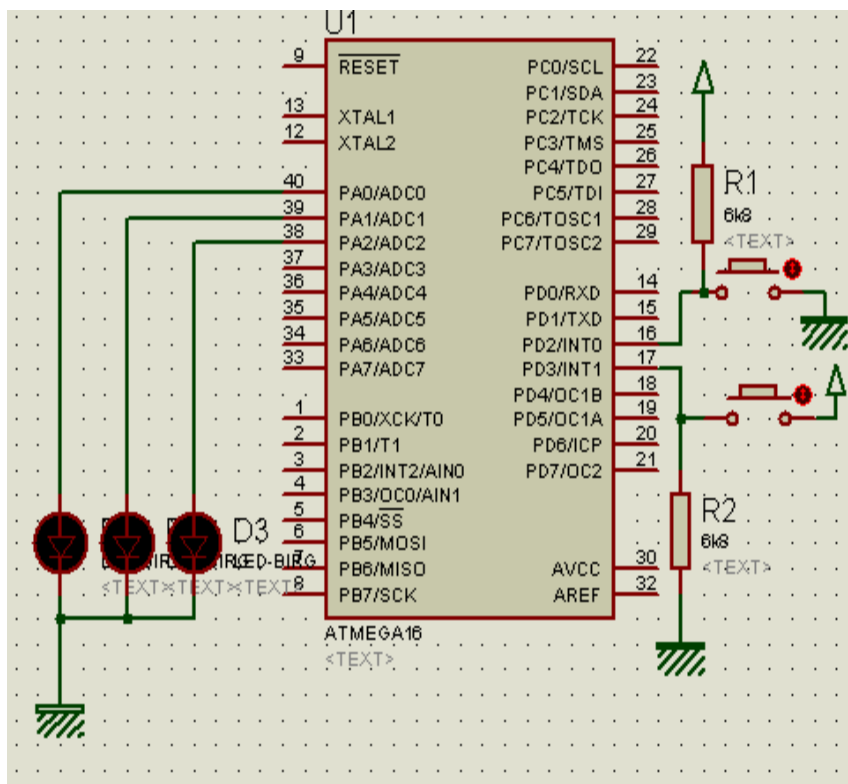
If Porta.0 = 0 Then : Set Porta.0 : Else : Reset Porta.0 : End If

Return

W:

If Porta.1 = 0 Then : Set Porta.1 : Else : Reset Porta.1 : End If
```

Return

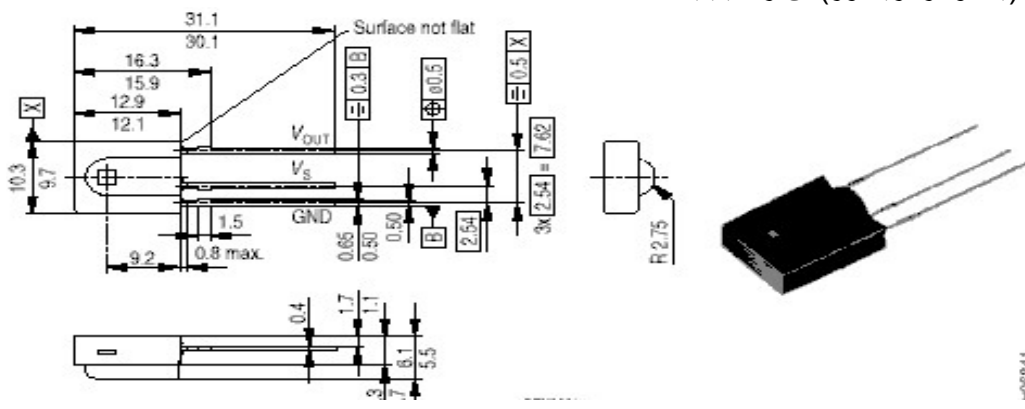


در برنامه بالا از دو منبع وقفه $int0$ و $int1$ استفاده شده است ، همانگونه که میبینید به یکی از پایه ها یک پالس بالا رودنده و به دیگری پالس پایین رونده اعمال میشود.

در حلقه ی اصلی مدام یکی از پایه ها خاموش و روشن میشود ، درو منبع وقفه مدام چک میشوند ، هنگامی که پالس مشخص شده به پایه وقفه عمل شد به زیر برنامه تعریف شده (q برای منبع وقفه صفر و w برای منبع وقفه 1) پرش میشود و وضعیت دو led تغییر میکند.مدار مورد استفاده را مشاهده میفرمایید.(این میکرو دارای منبع وقفه خارجی سوم نیز میباشد که در اینجا استفاده نشده ($int2$ پایه b.3)

<< راه اندازی گیرنده rc5:

امروزه گیرنده و فرستنده های مادون قرمز که انها را با نام تجاری گیرنده و فرستنده های rc5 میشناسند رواج فوق العاده ای پیدا کرده اند ، از انها در کنترل از راه دور ، سنسور های شمارنده ، ربات ها و استفاده فراوانی میشود.در زیر شکل گیرنده(یا فتو ترانزیستور) ان را میبینید:



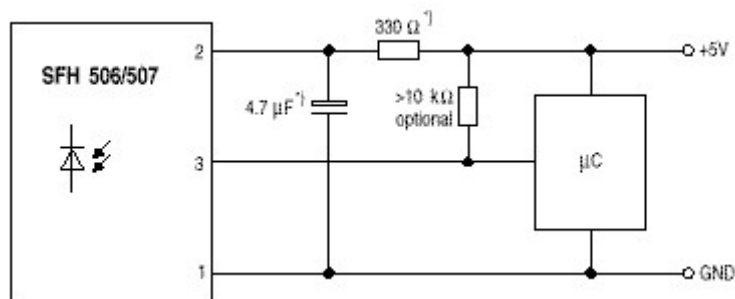
(یا ir-

فرستنده

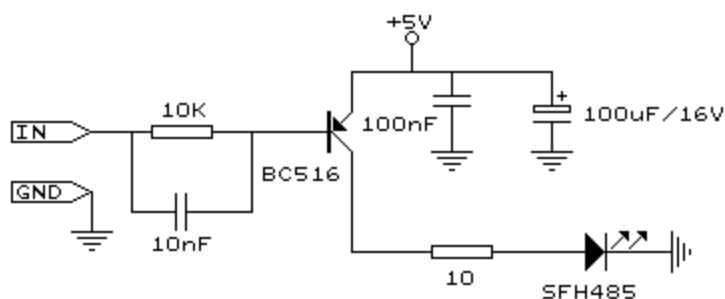
(led) دقیقاً مانند یک led میباشد که رنگ ان سفید است ، اتصال گیرنده به میکرو مطابق شکل زیر است:(نوع 2 پایه این سنسور نیز موجود میباشد که به نام گیرنده مادون قرمز معروف است ، اتصال نوع دوپایه بدون VCC صورت میگیرد

PEYMAN KHODDAMI

، نوع دویاپه باعث ایجاد خطای غیر قابل چشم پوشی میشود ، به طوری که از ان فقط در موارد تشخیص مانع استفاده میگردد)



اتصال فرستنده به میکرو مانند شکل زیر است :



ترانزیستور نقش تقویت پالسها و دیگر قطعات وظیفه حذف نویز را به عهده دارند. در زیر ابتدا به پیکر بندی فرستنده میپردازیم:

فرستنده مادون قرمز با دستور زیر راه اندازی میشود:

RC5SEND togglebit, address, command

Togglebit : به جای این واژه شما باید 0 یا 32 قرار دهید ، این اعداد نشان دهنده سطح شروع ارسال پالس میباشد.

Address : نشان دهنده ادرس دستور است که میتواند به فرم باینری هگز یا دسیمال باشد. (در گیرنده و فرستنده باید ادرس دسترو فرستاده شده و دستور گرفته شده یکی باشد ، تا دستور اجرا شود)

Command : نشان دهنده فرمان است که میتواند به فرم باینری ، هگز یا دسیمال باشد. (مثلا شما در گیرنده دستور زیر را ارسال میکنید : Rc5send 0 , 20 , 12 ، در گیرنده با دستوراتی که بعدا گفته میشود ، این کد را دریافت میکنید ، سپس با یک دستور if میتوانید هر کاری که خواستید انجام دهید ، اگر دستور 12 و ادرس 20 بود بین b.0 را یک کن)

PEYMAN KHODDAMI

وقتی rc 5 پیکر بندی میشود ، پایه OC1 (A) به عنوان خروجی داده قرار میگیرد و فرستنده باید به این پایه متصل شود ، در این حالت دیگر نمیتوان از وقفه های تایمر 1 استفاده کرد. در مثال زیر با نحوه کار این فرستنده بیشتر آشنا میشوید:

```
$regfile = "2313def.dat"
```

```
$crystal = 4000000
```

```
Config Portd = Input
```

```
Do
```

```
Debounce Pind.0 , 0 , Q
```

```
Debounce Pind.1 , 0 , W
```

```
Loop
```

```
Q:
```

```
Rc5send 32 , 0 , 12
```

```
Wait 1
```

```
Return
```

```
W:
```

```
Rc5send 32 , 0 , 13
```

```
Wait 1
```

```
Return
```

```
End
```

در حالت قبل کد ها نوشته شده بدون هیچ تغییری به خروجی ارسال میشد ، حالتی وجود دارد که شما میتوانید کد ارسالی را با یک کد باینری ترکیب کنید (کد را به صورت رمز در اورید) برای این کار از دستور زیر استفاده میشود:

```
RC5SENDEXT togglebit, address, command
```

همه چیز مانند حالت قبل است فقط به جای togglebit میتوانید هر عددی که دلتان میخواهد قرار دهید تا با دستور ترکیب شده و ارسال شود (در گیرنده باید عدد گذاشته شده را بردارید که در ادامه توضیح داده میشود)مانند:

PEYMAN KHODDAMI

```
$regfile = "2313def.dat"

$crystal = 4000000

Config Portd = Input

Do

Debounce Pind.0 , 0 , Q

Debounce Pind.1 , 0 , W

Loop

Q:

Rc5sendext &B11000000 , 0 , 26

Wait 1

Return

W:

Rc5sendext 9 , 0 , 30

Wait 1

Return

End
```

<راه اندازی گیرنده rc5

گیرنده rc5 که شکل و طریقه اتصال آن به میکرو را در بالا مشاهده کردید با دستور زیر راه اندازی میشود:

```
CONFIG RC5 = pin
```

که oin نام پایه دلخواه میکرو میباشد که پایه خروجی گیرنده سه پایه rc5 به آن متصل میشود. با دستور زیر میتوان اطلاعات دریافتی توسط گیرنده rc5 را اشکار کرد:

```
GETRC5( address, command )
```

PEYMAN KHODDAMI

Address و command اطلاعات مربوط به دستور و ادرسی میباشند که توسط فرستنده ارسال شده و توسط گیرنده دریافت میشود، این اطلاعات باید در متغیرهای مناسب ریخته شوند و مورد استفاده قرار گیرند، متغیر میتواند از جنس bayt یا word باشد. مانند(این برنامه برای گیرنده مثال میباشد)

```
$regfile = "2313def.dat"

$crystal = 4000000

Config Portd = Output

Config Rc5 = Pind.7

Enable Interrupts

Dim Address As Byte , Command As Byte

Do

Getrc5(address , Command(

If Command = 12 Then

Set Portd.0

Reset Portd.1

End If

If Command = 13 Then

Set Portd.1

Reset Portd.0

End If

Loop

End
```

<< ساخت کنترل تلویزیون و سیدی sony:

توسط دستور زیر میتوان دستورات مخصوص کنترل تلوزیون و cd سونی را به این دستگاه ارسال کرد.

```
SONYSEND address
```

PEYMAN KHODDAMI

address: آدرس نام دستور است که با ارسال آن به تلوزیون کار مخصوص آن انجام میشود ، در زیر جدول کدهای مخصوص تلوزیون های sony آمده است:

SONY CD Infrared Remote Control codes (RM-DX55)

Function	Hex	Bin
Power	A91	1010 1001 0001
Play	4D1	0100 1101 0001
Stop	1D1	0001 1101 0001
Pause	9D1	1001 1101 0001
Continue	B91	1011 1001 0001
Shuffle	AD1	1010 1101 0001
Program	F91	1111 1001 0001
Disc	531	0101 0011 0001
1	011	0000 0001 0001
2	811	1000 0001 0001
3	411	0100 0001 0001
4	C11	1100 0001 0001
5	211	0010 0001 0001
6	A11	1010 0001 0001
7	611	0110 0001 0001
8	E11	1110 0001 0001
9	111	0001 0001 0001
0	051	0000 0101 0001
>10	E51	1110 0101 0001
enter	D11	1101 0001 0001
clear	F11	1111 0001 0001
repeat	351	0011 0101 0001
disc -	BD1	1011 1101 0001
disc +	H7D1	0111 1101 0001
<<	0D1	0000 1101 0001
>>	8D1	1000 1101 0001
<<	CD1	1100 1101 0001
>>	2D1	0010 1101 0001
SONY Cassette	RM-J901)	
Deck A		
stop	1C1	0001 1100 0001
play >	4C1	0100 1100 0001
play <	EC1	1110 1100 0001
>>	2C1	0010 1100 0001
<<	CC1	1100 1100 0001
record	6C1	0110 1100 0001
pause	9C1	1001 1100 0001
Dec B		
stop	18E	0001 1000 1110
play >	58E	0101 1000 1110
play <	04E	0000 0100 1110
>>	38E	0011 1000 1110
<<	D8E	1101 1000 1110
record	78E	0111 1000 1110
pause	98E	1001 1000 1110

PEYMAN KHODDAMI

---[SONY TV Infrared Remote Control codes (RM-694)]-----		
نام کلید	کد هگز	کد باینری
program +	&H090	0000 1001 0000
program -	&H890	1000 1001 0000
volume +	&H490	0100 1001 0000
volume -	&HC90	1100 1001 0000
Power	&HA90	1010 1001 0000
sound on/off	&H290	0010 1001 0000
1	&H010	0000 0001 0000
2	&H810	1000 0001 0000
3	&H410	0100 0001 0000
4	&HC10	1100 0001 0000
5	&H210	0010 0001 0000
6	&HA10	1010 0001 0000
7	&H610	0110 0001 0000
8	&HE10	1110 0001 0000
9	&H110	0001 0001 0000
0	&H910	1001 0001 0000
-/--	&HB90	1011 1001 0000

مثال

```
$regfile = "m8def.dat"
```

```
$crystal = 4000000
```

```
Config Kbd = Portd
```

```
Dim A As Byte
```

```
Dim B As Byte
```

```
Q:
```

```
A = Getkbd()
```

```
If A > 15 Then
```

```
Goto Q
```

```
End If
```

```
B = Lookup(a , Dat(
```

```
Sonysend B
```

PEYMAN KHODDAMI

Goto Q

End

Dat:

Data &H090 , &H890 , &H490 , &HC90 , &HA90 , &H290 , &H010 , &H810 , &H410 ,
&HC10 , &H210 , &HA10 , &H610 , &HE10 , &H110 , &H910

'program +,program -,volume +,volume -,power,sound on/off,1,2,3,4,5,6,7,8,9,0

در برنامه بالا با استفاده از میکرو مگا 8 و فرستنده rc5 یک کنترل مخصوص تلوزیون سونی ساخته شده است ، فرستنده rc5 مطابق مداری که در بالا معرفی شد به پایه oc1a میکرو (پایه 15) متصل میشود شما میتوانید از میکرو مگاشت استفاده کنید و این مدار را با 3 ولت راه اندازی کنید ، کد مربوط به دیگر دستگاههای شرکت SONY را میتوانید از

ادرس زیر بدست آورید:<http://www.fet.uni-hannover.de/purnhage/>

<< فرستندهای RC6:

این پروتکل ، برخلاف RC5 در اکثر دستگاههای صوتی تصویری جدید(تمامی دستگاههای CD چینی را پشتیبانی میکند) استفاده میشود ، تمامی جزییات مانند RC5 است ، فقط نحوه ارسال فرق دارد ، با دستور زیر میتوان دیتا را توسط این رابط به گیرنده ارسال کرد:

RC6SEND togglebit, address, command

Togglebit : نشان دهنده وضعیت پایه بعد از ارسال دستور است که میتواند صفر یا یک باشد .

Address : این مورد میتواند یکی از موارد جدول زیر باشد:

Device	Address
TV	0
VCR	5
SAT	8
DVD	4

Command : این گزینه مشخص کننده دستور ارسالی است (دستور مورد نظر به جای این کلمه نوشته میشود) دستورات مخصوص هر کنترل را میتوانید از سایت سازنده دریاف کنید در زیر کدهای مخصوص نوعی VCR(ویدئو سیدی رادیو دار) که در HELP بسکام موجود بود را مشاهده میفرمایید:

PEYMAN KHODDAMI

Command	Value	Command	Value
Key 0	0	Balance right	26
Key 1	1	Balance left	27
Key 2-9	2-9	Channel search+	30
Previous program	10	Channel search -	31
Standby	12	Next	32
Mute/un-mute	13	Previous	33
Personal preference	14	External 1	56
Display	15	External 2	57
Volume up	16	TXT submode	60
Volume down	17	Standby	61
Brightness up	18	Menu on	84
Brightness down	19	Menu off	85
Saturation up	20	Help	129
Saturation down	21	Zoom -	246
Bass up	22	Zoom +	247
Bass down	23		
Treble up	24		
Treble down	25		

مثال

```
$regfile = "m8def.dat"
```

```
$crystal = 4000000
```

```
Config Kbd = Portd
```

```
Dim A As Byte
```

```
Dim B As Byte
```

```
Q:
```

```
A = Getkbd()
```

```
If A > 15 Then
```

```
Goto Q
```

```
End If
```

PEYMAN KHODDAMI

```
B = Lookup(a , Dat (
```

```
Rc6send 0 , 5 , B
```

```
Goto Q
```

```
End
```

```
Dat:
```

```
Data 30 , 31 , 16 , 17 , 61 , 12 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 0
```

```
'program +,program -,volume +,volume -,power,sound on/off,1,2,3,4,5,6,7,8,9,0
```

<<اندازه گیری یک خازن یا مقاومت:

شما با استفاده از دستور زیر میتوانید مقدار ثابت زمانی مقاومت و خازنی که به پایه دلخواه میکرو avr متصل شده است را بدست آورید :

```
var = GETRC( pin , number )
```

var: یک متغیر از جنس word میباشد که مقدار ثابت زمانی در آن ریخته میشود.

Pin: نام پورتهی است که خازن و مقاومت به آن متصل است (مانند porta یا portd).

Number: شماره پایه ای است که مقاومت و خازن به آن متصل شده است (مانند 1 یا 2) (این مقدار نمیتواند از 7 بیشتر شود)).

در مدارات مقاومت یا خازن که به اختصار به آن rc میگویند ، خازن بعد از 5 ثابت زمانی شارژ میشود (بعد از 5t)

مقدار دقیق این ثابت زمانی به مقدار خازن و مقدار مقاومت بستگی دارد و فرمول آن به شکل $t = r * C$ است ، میکرو

مقدار ثابت زمانی را اندازه میگیرد ، شما با داشتن مقدار یکی از المانها میتوانید مقدار دیگر را بدست آورید ، مانند:

```
$regfile = "M16DEF.DAT"
```

```
$crystal = 8000000
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =  
Pinc.4 , E = Pind.2 , Rs = Pind.3
```

```
Config Lcd = 16 * 2
```

```
Config Porta = Output
```


PEYMAN KHODDAMI

Dim W As Word

Do

```
W = Getrc(pina , 7) : W = W / 1000 : Locate 1 , 1 : Lcd W : Wait 2
```

Loop

end

در مثال بالا مقدار یک خازن اندازه گرفته شده است ، خازن مجهول با یک مقاومت 1 کیلو اهم سری شده و porta.7 متصل شده است ،(شما همچنین میتوانید مقدار مقاومت را نیز اندازه گیری کنید ، به شرطی که مقدار خازن را بدانید) مدار مثال بالا را در زیر مشاهده میکنید:



<< مقایسه کننده آنالوگ :

مقایسه کننده آنالوگ مقادیر ولتاژ آنالوگ موجود بر روی دو پایه خود را (پایه مثبت (AIN0) و پایه منفی (AIN1)) با هم مقایسه می کند. (مانند op-amp). زمانی که ولتاژ موجود در ورودی مثبت بیشتر از ولتاژ موجود در ورودی منفی باشد ، خروجی مقایسه کننده (ACO) یک می شود.مقایسه کننده دارای یک پرچم وقفه مجزا است.خروجی مقایسه کننده می تواند به عنوان تریگر ورودی CAPTURE تایمر/ کانتر یک نیز استفاده شود.

پیکره بندی مقایسه کننده آنالوگ در محیط BASCOM

دستور پیکره بندی مقایسه کننده آنالوگ در محیط BASCOM

```
CONFIG ACI =ON/OFF, COMPARE = ON/OFF, TRIGGLE=TOGGLE,RISING/FALLING
```

CONFIG ACI = ON/OFF : در زمان استفاده از مقایسه کننده باید یک باشد.(در صورت استفاده از صفر مقایسه کننده کار نخواهد کرد).

COMPARE = ON/OFF: در صورت انتخاب ON ، ACO مستقیماً به ورودی CAPTURE تایمر/کانتر یک وصل می شود.

TRIGGLE=TOGGLE,RISING/FALLING: نحوه روی دادن وقفه مقایسه کننده را نشان می دهد.

PEYMAN KHODDAMI

FALLING : یک لبه پایین رونده در خروجی مقایسه کننده باعث یک شدن پرچم وقفه مقایسه کننده و اجرا شدن برنامه وقفه خواهد شد.

RISING : یک لبه بالا رونده در خروجی مقایسه کننده باعث یک شدن پرچم وقفه مقایسه کننده و اجرا شدن برنامه وقفه خواهد شد.

TOGGLE : یک به صفر یا یک صفر به یک در خروجی مقایسه کننده باعث یک شدن پرچم وقفه مقایسه کننده و اجرا شدن برنامه وقفه خواهد شد.

<< تایمر / کانتر

میکرو های AVR نهایتاً دارای سه تایمر / کانتر هستند (به جز MEGA128, MEGA64, MEGA162) و تعداد محدود دیگری که 4 تایمر دارند). این تایمر / کانتر ها به نام های تایمر کانتر 0 و تایمر کانتر 1 و تایمر کانتر 2 و تایمر کانتر سه ، نام گذاری میشوند . کار تایمر ها شمردن تا یک عدد خاص و کار کانتر ها شمردن یک پالس ، که به پایه مخصوص اعمال میشود است ، از تایمر و کانتر استفاده های دیگری نیز میشود ، مانند ساخت پالس PWM و ... که در ادامه انها را معرفی میکنیم.

اولین تایمر/کانتری که معرفی میشود تایمر صفر است ، در زیر مشخصات این مورد را مشاهده میفرمایید:

- 1- این تایمر/ کانتر 8 بیتی است ، و نهایتاً میتواند تا 2^8 (255) بشمارد.
- 2- کلاک این تایمر میتواند توسط نوسان ساز داخلی یا نوسان ساز خارجی تامین شود (مقدار فرکانس نوسان ساز بر عدد PRESCALE تقسیم میشود).
- 3- این تایمر دارای چندین منبع وقفه میباشد که شما میتوانید انها را در هر قسمت از برنامه فعال یا غیر فعال کنید. از این منابع وقفه میتوان ، وقفه سرریزی را نام برد.
- 4- این تایمر/کانتر میتواند در مد تایمر و کانتر راه اندازی شود.
- 5- ورودی کانتر پایه T0 (در میکرو مگا 16 پورت B.0 ، پایه شماره 1) میباشد < راه اندازی تایمر صفر در محیط بسکام:

تایمر صفر با دستور زیر پیکر بندی میشود:

```
CONFIG TIMER0 = TIMER ,PRESCALE =1|8|64|256|1024
```

عدد PRESCALE فرکانس (دقت) تایمر را معین میکند . فرکانس و زمانی که تایمر میشمارد از فرمولهای زیر محاسبه میشوند:

PEYMAN KHODDAMI

$$\text{زمان} = \frac{\text{PREACALE} * \text{تایمر بیت}}{\text{مقدار کریستال}}$$

$$\text{فرکانس} = \frac{\text{مقدار کریستال}}{\text{PREACALE}}$$

بعد از دستور بالا، تایمر با دستور START TIMER شروع به شمارش میکند و با دستور STOP TIMER متوقف می شود.

تایمر پس از شمردن تا 255 سر ریز میشود ، شما با استفاده از دستور ENABLE OVFO یا ENABLE INTRRUPTS میتوانید وقفه سر ریزی تایمر را راه اندازی کنید . در صورتی که وقفه سر ریزی تایمر فعال باشد ، تایمر پس از سر ریزی به برجسیبی که با یکی از دستورات ON OVFO LABLE و یا ON TIMER0 LABLE مشخص شده پرش میکند ، باز گشت از وقفه با دستور RETURN انجام میشود.

شما همچنین میتوانید با استفاده از دستور VAR = TIMER0 ، مقدار تایمر را در یک متغیر از جنس بایت قرار دهید و یا با دستور TIMER0 = VALUE ، مقدار اولیه ای در تایمر قرار داد تا تایمر از آن شروع به شمارش کند.

در این مثال میخواهیم توسط تایمر صفر زمان 1 ثانیه بسازیم ، طبق فرمول اگر شما از کریستال 1 مگا هرتز (که کمترین مقدار کریستال مورد استفاده برای AVR میباشد (به جز کریستال ساعت)) و PREACALE ، 1024 استفاده نمایید ، بیشترین زمانی که میتوانید ایجاد کنید { $265 \text{ ms} = \frac{256 * 1024}{1000000} = \text{زمان}$ }، است، بنابراین ما باید از وقفه سر ریزی تایمر استفاده کنیم ، و در هر بار وقفه به یک متغیر عددی یک واحد بیفزاییم ، هنگامی که متغیر برابر 4 شد ، متغیر صفر شود و یک واحد به متغیر اصلی افزوده شود. بنابراین با ضرب کردن زمان بدست آمده در 4 زمان اصلی بدت میاید که برابر با 1048 میلی ثانیه است.

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Dim B As Byte , C As Byte
```

```
Config Timer0 = Timer , Prescale = 1024
```

```
Enable Interrupts
```

PEYMAN KHODDAMI

```
Enable Timer0  
  
On Timer0 P  
  
Start Timer0  
  
Do  
  
Loop  
  
End  
  
:P  
  
Locate 1 , 8 : Lcd B  
  
Incr B  
  
If B > 3 Then : Incr C : Locate 1 , 1 : Lcd C : B = 0 : End If  
  
return
```

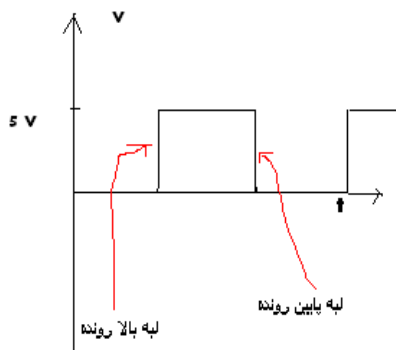
در این برنامه ، هنگامی که تایمر سرریز میشود (تا 255 می‌شمارد) به زیر برنامه p پرش میشود و یک واحد به متغیر b افزوده میشود ، شرط if میگوید که اگر مقدار متغیر b برابر با 4 شد یک واحد به متغیر c اضافه کن و آن را در سطر اول ، ستون اول lcd نمایش بده ، بعد متغیر b را صفر کن ، از آنجا که مقدار b به 4 نرسیده (در اولین رجوع مقدار b 1 میشود) این شرط اجرا نمیشود .

با دستور return به برنامه تایمر برمیگردد . مقدار تایمر صفر شده و دوباره تا 255 شمرده میشود ... و این عمل مدام تکرار میشود. با چند دستور ساده میتوان از برنامه بالا یک ساعت ساخت (برنامه ساعت را در پروژه ها ببینید)

< راه اندازی کانتر صفر در محیط بسکام:

کانتر صفر در بسکام با دستور زیر پیکربندی میشود:

```
CONFIG TIMER0 = COUNTER , EDGE = RISING / FALLING
```



با انتخاب EDGE = RISING کانتر نسبت به لبه ی بالا رونده حساس است

با انتخاب EDGE = FALLING کانتر نسبت لبه ی پایین رونده حساس است

PEYMAN KHODDAMI

(لبه های بالا رونده یا پایین رونده را می‌شمارد (بالا رونده <صفر به یک و پایین رونده >یک به صفر))

شما همچنین می‌توانید با استفاده از دستور `VAR = COUNTER 0` ، مقدار کانتر را در یک متغیر از جنس بایت قرار دهید و یا با دستور `COUNTER 0 = VALUE` ، مقدار اولیه ای در کانتر قرار دهید تا کانتر از آن شروع به شمارش کند.

کانتر نیز مانند تایمر پس از شمردن تا 255 سر ریز می‌شود ، شما با استفاده از دستور `ENABLE OVFO` می‌توانید وقفه سر ریزی کانتر را راه اندازی کنید . در صورتی که وقفه سر ریزی کانتر فعال باشد ، کانتر پس از سر ریزی به برچسبی که با یکی از دستورات `ON OVFO LABLE` و یا `ON TIMERO LABLE` مشخص شده پرش میکند ، باز گشت از وقفه با دستور `RETURN` انجام می‌شود.

در این مثال کانتر صفر تعداد پالسهای اعمالی به پایه t0 (پایه شماره 1 مگا 16) را می‌شمارد و پس از شمردن تا مقدار 255 ، به زیر برنامه وقفه می‌رود ، در انجا یک واحد به متغیر b افزوده می‌شود ، متغیر b تعداد دفعات سر ریزی کانتر را نشان می‌دهد که ما آن را روی سطر دوم lcd به نمایش می‌گذاریم ، برنامه با دستور `return` از زیر برنامه خارج می‌شود و به حلقه اصلی برمی‌گردد.

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Dim B As Byte
```

```
Config Timer0 = Counter , Edge = Rising
```

```
Enable Interrupts
```

```
On Counter0 P
```

```
Do
```

```
Locate 1 , 1 : Lcd Counter0
```

```
Loop
```

```
End
```

```
:P
```

PEYMAN KHODDAMI

Incr B : Locate 2 , 1 : Lcd "tedad dafeat sarrizi" ; B

Return

نکته : عملکرد تایمر و کانتر از عملکرد سایر قسمت ها جدا بوده و دیگر اعمال (مانند دستورات تاخیر مثل wait و...) بر عملکرد ان اثری ندارد.

<<تایمر کانتری یک:

در زیر مشخات این مورد را مشاهده میفرمایید:

- 1- این تایمر کانتر 16 بیتی است ، و نهایتاً میتواند تا 2^{16} (65535) بشمارد.
- 2- کلاک این تایمر میتواند توسط نوسان ساز داخلی یا نوسان ساز خارجی یا از پایه t1 (در مگا 16 پایه شماره 2 (portb.1) توسط پالس خارجی، تامین شود (مقدار فرکانس نوسان ساز بر عدد PRESCALE تقسیم میشود).
- 3- تایمر / کانتر یک دارای دو خروجی مقایسه ای است که دو رجیستر OCR1A و OCR1B 3مقدار مقایسه ای را در خود جای می دهند و با محتوای تایمر/کانتر مقایسه می شوند .
- 4- در زمان تساوی محتوای رجیستر مقایسه و محتوای تایمر/کانتر ، وضعیت پایه های خروجی مد مقایسه ای OC1A و OC1B می تواند تغییر کند.
- 5- تایمر /کانتر در مد CAPTURE نیز می تواند به کار رود . با تحریک پایه ICP می توان محتوای تایمر / کانتر را در رجیستر ورودی CAPTURE(ICR1) قرار داد.
- 6- خروجی مقایسه کننده آنالوگ نیز می تواند به عنوان تریگر ورودی CAPTURE قرار گیرد.
- 7- این تایمر دارای چندین منبع وقفه میباشد که شما میتوانید انها را در هر قسمت از برنامه فعال یا غیر فعال کنید. از این منابع و وقفه میتوان ،وقفه سرریزی را نام برد.
- 8- این تایمر کانتر میتواند در مد تایمر و کانتر و pwm راه اندازی شود.
- 9- ورودی کانتر پایه T1 (در میکرو مگا 16 پورت B.0 ، پایه شماره 1)میباشد و خروجی پالس pwm پایه های oc1a و oc1b (در میکرو مگا 16 به ترتیب پایه های 18 و 19 (portd.4 و d.5) است همچنین د پایه ی فوق میتوانند به عنوان خروجی مد مقایسه ای تایمر مورد استفاده قرار بگیرند.

< راه اندازی تایمر یک در محیط بسکام:

تایمر 1 با دستور زیر بیکربندی میشود:

Config Timer1 = Timer , PRESCALE = 1 | 8 | 64 | 256 | 1024

PEYMAN KHODDAMI

عدد PRESCALE فرکانس (دقت) تایمر را معین میکند. فرکانس و زمانی که تایمر می‌شمارد از فرمولهای زیر محاسبه میشوند:

$$\text{زمان} = \frac{\text{PRESCALE} * \text{تایمر بیت}}{\text{مقدار کریستال}}$$

$$\text{فرکانس} = \frac{\text{مقدار کریستال}}{\text{PRESCALE}}$$

بعد از دستور بالا، تایمر با دستور START TIMER شروع به شمارش میکند و با دستور STOP TIMER متوقف می‌شود. تایمر پس از شمردن تا 2^{16} 65536 سر ریز میشود، شما با استفاده از دستور ENABLE OVF1 یا ENABLE INTERRUPTS میتوانید وقفه سر ریزی تایمر را راه اندازی کنید. در صورتی که وقفه سر ریزی تایمر فعال باشد، تایمر پس از سر ریزی به برجسبی که با یکی از دستورات ON OVF1 LABEL و یا ON TIMER1 LABEL مشخص شده پرش میکند، باز گشت از وقفه با دستور RETURN انجام میشود.

شما همچنین میتوانید با استفاده از دستور VAR = TIMER1، مقدار تایمر را در یک متغیر از جنس word قرار دهید و یا با دستور TIMER1 = VALUE، مقدار اولیه ای در تایمر قرار دهید تا تایمر از آن شروع به شمارش کند.

در این مثال میخواهیم توسط تایمر 1 یک ساعت بسازیم، برای این کار ابتدا باید یک زمان 1 ثانیه ای ایجاد کنیم طبق فرمول اگر شما از کریستال 4 مگا هرتز و PRESCALE 64، استفاده نمایید، میتوانید زمانی برابر با 1.04 ثانیه ایجاد کنید، با مقادیر دیگر نیز میتوانید زمان های دقیق تر بسازید. (بیشترین زمانی که با این تایمر میتوانید بسازید برابر با 67.108864 ثانیه میباشد)

```
$regfile = "m16def.dat"
```

```
$crystal = 12000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 ,  
Db7 = Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Dim San As Byte , Da As Byte , Saat As Byte
```

```
San = 0 : Da = 0 : Saat = 1
```

```
Config Timer1 = Timer , Prescale = 64
```

```
Enable Interrupts
```

PEYMAN KHODDAMI

```
Enable Timer1

On Timer1 P

Start Timer1

Do

Locate 1 , 1 : Lcd Saat ; ":" ; Da ; ":" ; San

Loop

End

:P

Incr San

If San > 59 Then : Incr Da : San = 0 : End If

If Da > 59 Then : Incr Saat : Da = 0 : End If

If Saat > 12 Then : San = 1 : End If

Return
```

در برنامه بالا هنگامی که تایمر سرریز میشود به زیر برنامه p پرش میشود در آنجا متغیر ثانیه 1 واحد افزایش می یابد
'

<راه اندازی تایمر 1 در مد مقایسه ای (Compare):

در این مد شما میتوانید مقدار تایمر 1 را با دو عدد دلخواه مقایسه کنید و در صورت برابری یا نابرابری مقدار تایمر با مقدار دلخواه وضعیت پایه های oc1a و oc1b را تغییر دهید. راه اندازی تایمر یک در مد مقایسه ای با دستورات زیر انجام میشود:

```
CONFIG TIMER1= TIMER,COMPARE A = CLEAR
|SET|TOGGLE|DISCONNECT,COMPARE B = CLEAR
|SET|TOGGLE|DISCONNECT, PRESCALE=1|8|64|256|1024,CLEAR TIMER
=1|0
```


PEYMAN KHODDAMI

COMPARE A = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط تایمر 1 با مقدار

COMPARE A ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC1A می تواند SET(یک), (صفر) CLEAR ,
(برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc1a به یک ورودی خروجی عادی تبدیل شود).

COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط تایمر 1 با مقدار

COMPARE B ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC1B می تواند SET(یک), (صفر) CLEAR ,
(برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc1b به یک ورودی خروجی عادی تبدیل
شود).

PRESCALE=1|8|64|256|1024 : عدد PRESCALE فرکانس (دقت) تایمر را معین میکند.

CLEAR TIMER = 1|0 : با انتخاب گزینه 1 ، محتوای تایمر/کانتر در زمان تطابق مقایسه ای (\$0000) می
شود و در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی را که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Compare1a = x
```

```
Compare1b =x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد ، هنگامی که عدد شمرده شده توسط تایمر یک با اعداد گذاشته شده
برابر شد میکرو وضعیت پایه های مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد.

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Timer , Compare A = Set , Compare B = Toggle , Prescale =  
1024 , Clear Timer = 0
```

```
Compare1a = 1000
```

```
Compare1b = 5000
```

```
Do
```

PEYMAN KHODDAMI

Locate 1 , 1

Lcd Timer1

Loop

End

هنگامی که تایمر 1 تا 1000 شمرد پایه oc1a یک میشود و هنگامی که مقدار شمرده شده به 5000 رسید پایه oc1b یک میشود (oc1a پایه 19 و oc1b پایه 18 مگا 16 میباشد).

< استفاده از وقفه مد مقایسه ای تایمر 1 :

مد مقایسه ای دارای 2 منبع وقفه میباشد که با دستورات زیر فعال میشوند:

Enable Interrupts

Enable Oc1a

Enable Oc1b

با دستور فعال سازی وقفه مقایسه ، هنگامی که مقدار شمرده شده توسط میکرو با Compare1a برابر شد cpu میکرو با دستور on oc1a lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد، هنگامی که مقدار شمرده شده توسط میکرو با Compare1b برابر شد cpu میکرو با دستور on oc1b lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد. در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی برش میکند.مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Timer , Compare A = Set , Compare B = Toggle , Prescale =  
1024 , Clear Timer = 0
```

```
Compare1a = 2120
```

```
Compare1b = 63000
```

PEYMAN KHODDAMI

```
Enable Interrupts
```

```
Enable Oc1a
```

```
Enable Oc1b
```

```
On Oc1a Q
```

```
On Oc1b W
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Timer1
```

```
Loop
```

```
End
```

```
Q:
```

```
Locate 2 , 1
```

```
Lcd Timer1
```

```
Return
```

```
W:
```

```
Locate 2 , 8
```

```
Lcd Timer1
```

```
Return
```

در مثال بالا هنگامی که رقم شمرده شده توسط تایمر با مقدار Compare1a (2120) برابر شد cpu به برچسب q پرش میکند و در آنجا مقدار شمرده شده توسط تایمر را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد ، هنگامی که رقم شمرده شده توسط تایمر با مقدار Compare1b (63000) برابر شد cpu به برچسب w پرش میکند و در آنجا مقدار شمرده شده توسط تایمر را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد . همچنین در مورد اول پایه oc1a یک میشود و در مورد دوم وضعیت پایه oc1b تغییر میکند (در صورت 0 بودن یک میشود و بلعکس).

<راه اندازی تایمر 1 در مد CAPTURE :

PEYMAN KHODDAMI

در صورتی که تایمر یک را در این مد پیکربندی کنید ، با اعمال یک پالس بالا رونده یا پایین رونده (که نوع آن در هنگام پیکر بندی مشخص میشود) به پایه icp (پایه 20 مگا 16) ، در همان لحظه مقدار شمرده شده توسط تایمر 1 در رجیستر CAPTURE قرار میگیرد ، محتوای رجیستر CAPTURE را می توان با دستور VAR = CAPTURE در یک متغیر از جنس word قرار داد . راه اندازی تایمر یک در مد CAPTURE با دستورات زیر انجام میشود:

```
Config Timer1 = Timer, Capture Edge= Falling | Rising ,Noise Cancel=1|0 ,  
Prescale =1|8|64|256|1024
```

Capture Edge= Falling | Rising : این گزینه مشخص میکند ، پالس اعمالی به پایه icp بالا رونده (Falling) است یا پایین رونده (Rising). (CAPTURE به کدام پالس حساسیت نشان میدهد)

Noise Cancel=1|0 : در صورت انتخاب یک از نویز های موجود بر روی پایه icp چشم پوشی میشود و هر پالس با لبه ی تعیین شده میتواند CAPTURE را راه اندازی کند ، در صورت انتخاب صفر فقط پالس های با دامنه ی 5 ولت قادر به راه اندازی CAPTURE خواهند بود.

Prescale =1|8|64|256|1024: دقت تایمر را نشان میدهد (با استفاده از این مقدار و مقدار کریستال میتوانید زمان شمرده شده توسط تایمر تا هنگام سرریزی را با فرمولی که در بالا گفته شد محاسبه کنید)
با دستور Enable Icp1 این مد فعال میشود و پایه icp آماده دریافت پالس میگردد . مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Timer , Capture Edge = Falling , Noise Cancel = 1 , Prescale  
= 1024
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Timer1
```

```
Locate 2 , 1
```

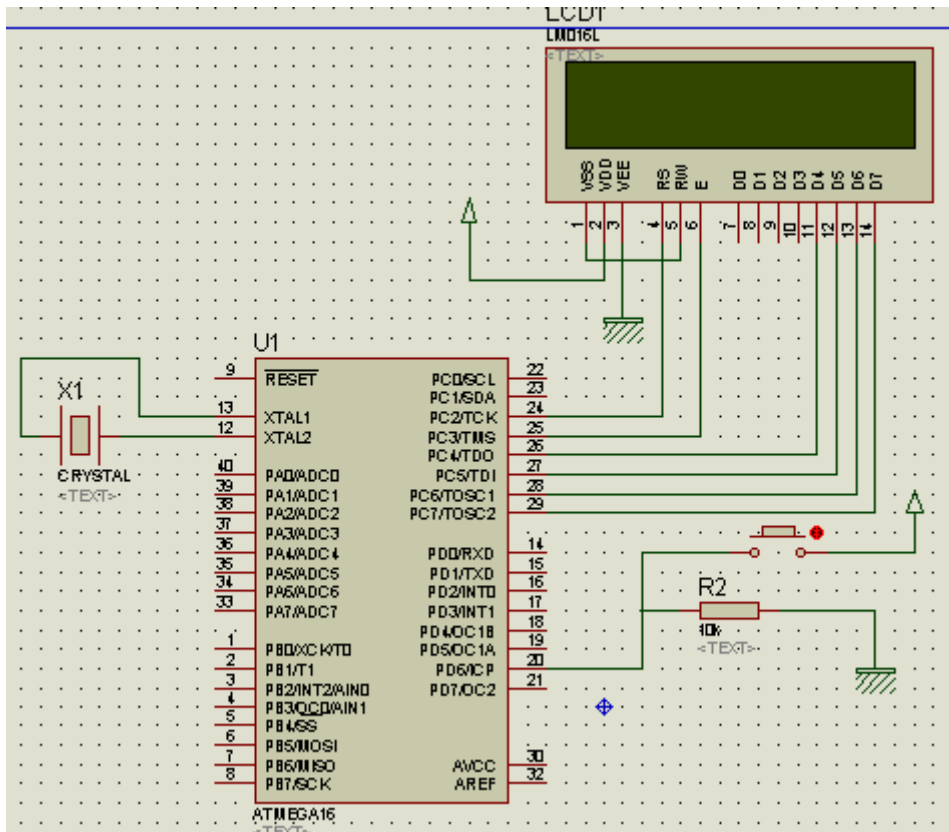
```
Lcd Capture1
```

PEYMAN KHODDAMI

Loop

End

در مثال بالا هنگامی که یک پالس بالا رونده به پایه icp میشود مقدار شمرده شده توسط تایمر 1 درون رجیستر Capture قرار میگیرد و سپس بر روی Lcd نمایش داده میشود. مدار استفاده شده برای مثال بالا و سایر مثال های این بخش را مشاهده میفرمایید:



< استفاده از وقفه مد Capture تایمر 1:

مد Capture تایمر یک نیز مانند دیگر مد ها دارای یک منبع وقفه است که شما میتونید با دستور Enable Icp1 ان را فعال کنید تا با دستور On icp1 lable ، هنگامی که پالسی به پایه icp اعمال شد ، cpu به برجسب مورد نظر پرش کند و در انجا عملیات دلخواه را انجام دهد .مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Timer , Capture Edge = Falling , Noise Cancel = 1 , Prescale  
= 1024
```

```
Enable Interrupts
```

```
Enable Icp1
```

```
On Icp1 Q
```

```
Dim A As Byte
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Timer1
```

```
Loop
```

```
Q:
```

```
Locate 2 , A
```

```
Lcd Capture1
```

```
A = A + 4
```

```
Return
```

```
End
```

نر مثال بالا ، هنگامی که پالسی به پایه icp اعمال شود مقدار شمرده شده توسط تایمر 1 در رجیستر Capture1 ریخته میشود و cpu به برجسب q برش میکند و در انجا مقدار Capture1 را در ستون های مختلف lcd نمایش میدهد.

< راه اندازی کانتر 1 در محیط بسکام: (راه اندازی تایمر/کانتر1 در مد کانتر در محیط بسکام)

کانتر 1 در بسکام با دستور زیر پیکربندی میشود:

```
CONFIG TIMER1 = COUNTER , EDGE = RISING / FALLING
```

با انتخاب EDGE = RISING کانتر نصبت به لبه ی بالا رونده حساس است

PEYMAN KHODDAMI

با انتخاب EDGE = FALLING کانتر نسبت لبه ی پایین رونده حساس است

(لبه های بالا رونده یا پایین رونده را می‌شمارد (بالا رونده < صفر به یک و پایین رونده > یک به صفر))

شما همچنین می‌توانید با استفاده از دستور VAR = COUNTER1 ، مقدار کانتر را در یک متغیر از جنس word قرار دهید و یا با دستور COUNTER1 = VALUE ، مقدار اولیه ای را در کانتر قرار دهید تا کانتر از آن شروع به شمارش کند. کانتر نیز مانند تایمر پس از شمردن تا 65536 سر ریز میشود ، شما با استفاده از دستور ENABLE OVF1 می‌توانید وقفه سر ریزی کانتر را راه اندازی کنید . در صورتی که وقفه سر ریزی کانتر فعال باشد ، کانتر پس از سر ریزی به برجسی که با یکی از دستورات ON OVF1 LABEL مشخص شده پرش میکند ، باز گشت از وقفه با دستور RETURN انجام میشود.

در این مثال کانتر صفر تعداد پالسهای اعمالی به پایه t1 (پایه شماره 2 مگا 16) را می‌شمارد.

```
$regfile = "m16def.dat"

$crystal = 12000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Timer1 = Counter , Edge = Falling

Do

Locate 1 , 1 : Lcd Counter1

Loop

End
```

کانتر یک را نیز میتوان مانند تایمر در دو مد مقایسه ای و Capture راه اندازی کرد که طریقه راه اندازی در زیر آورده شده است:

< راه اندازی کانتر 1 در مد مقایسه ای (Compare):

در این مد شما می‌توانید مقدار کانتر 1 را با دو عدد دلخواه مقایسه کنید و در صورت برابری یا نابرابری مقدار شمرده شده توسط کانتر با مقدار دلخواه وضعیت پایه های oc1a و oc1b را تغییر دهید. راه اندازی کانتر یک در مد مقایسه ای با دستورات زیر انجام میشود:

PEYMAN KHODDAMI

```
CONFIG TIMER1 = COUNTER , EDGE = RISING / FALLING, COMPARE A = CLEAR  
|SET|TOGGLE| DISCONNECT, COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT, CLEAR TIMER  
= 1|0
```

EDGE = RISING / FALLING : با انتخاب EDGE = RISING کانتر نسبت به لبه ی بالا رونده حساس است و با انتخاب
EDGE = FALLING کانتر نسبت به لبه ی پایین رونده حساس است.

COMPARE A = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط کانتر 1 با مقدار
COMPARE A ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC1A می تواند SET (یک) ، (صفر) CLEAR
(برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc1a به یک ورودی خروجی عادی تبدیل شود).

COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط کانتر 1 با مقدار
COMPARE B ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC1B می تواند SET (یک) ، (صفر) CLEAR
(برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc1b به یک ورودی خروجی عادی تبدیل
شود).

CLEAR TIMER = 1|0 : با انتخاب گزینه 1 ، محتوای کانتر 1 در زمان تطابق مقایسه ای RESET (\$0000) می شود و
در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Compare1a = x
```

```
Compare1b =x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد ، هنگامی که عدد شمرده شده توسط کانتر یک با اعداد گذاشته شده
برابر شد میکرو وضعیت پایه های مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد.

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Counter , Edge = Falling , Compare A = Set , Compare B = Set  
, Clear Timer = 0
```


PEYMAN KHODDAMI

```
Compare1a = 300
```

```
Compare1b = 400
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter1
```

```
Loop
```

```
End
```

هنگامی که پالس های شمرده شده توسط کانتر 1 به 300 رسید ، پایه oc1a یک میشود و هنگامی که مقدار شمرده شده به 400 رسید پایه oc1b یک میشود (oc1a پایه 19 و oc1b پایه 18 مگا 16 میباشد). (ورودی کانتر 1 پایه t1 پایه 2 مگا 16 است)

< استفاده از وقفه مد مقایسه ای کانتر 1 :

مد مقایسه ای دارای 2 منبع وقفه میباشد که با دستورات زیر فعال میشوند:

```
Enable Interrupts
```

```
Enable Oc1a
```

```
Enable Oc1b
```

با دستور فعال سازی وقفه مقایسه ، هنگامی که تعداد پالس شمرده شده توسط کانتر با Compare1a برابر شد cpu میکرو با دستور on oc1a lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد، هنگامی که تعداد پالس شمرده شده توسط کانتر با Compare1b برابر شد cpu میکرو با دستور on oc1b lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد. در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی برش میکند.مانند

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

PEYMAN KHODDAMI

```
Config Timer1 = Counter , Edge = Falling , Compare A = Set , Compare B = Set  
, Clear Timer = 0
```

```
Compare1a = 30
```

```
Compare1b = 40
```

```
Enable Interrupts
```

```
Enable Ocl1a
```

```
Enable Ocl1b
```

```
On Ocl1a Q
```

```
On Ocl1b W
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter1
```

```
Loop
```

```
End
```

```
Q:
```

```
Cls
```

```
Locate 2 , 1
```

```
Lcd " Counter1=30"
```

```
Return
```

```
W:
```

```
Cls
```

```
Locate 2 , 1
```

```
Lcd " Counter1=40"
```

```
Return
```

PEYMAN KHODDAMI

در مثال بالا هنگامی که تعداد پالس شمرده شده توسط کانتر با مقدار Compare1a (30) برابر شد cpu به برچسب q پرش میکند و در انجا عبارت "Counter1=30" را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد ، هنگامی که تعداد پالس شمرده شده توسط کانتر با مقدار Compare1b (40) برابر شد cpu به برچسب w پرش میکند و در انجا عبارت "Counter1=40" را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد . همچنین در مورد اول پایه oc1a یک میشود و در مورد دوم وضعیت پایه oc1b یک میشود .

<راه اندازی کانتر 1 در مد CAPTURE :

در صورتی که کانتر یک را در این مد پیکربندی کنید ، با اعمال یک پالس بالا رونده یا پایین رونده (که نوع ان در هنگام پیکر بندی مشخص میشود) به پایه icp (پایه 20 مگا 16) ، در همان لحظه تعداد پالس شکرده شده توسط کانتر 1 در رجیستر CAPTURE قرار میگیرد ، محتوای رجیستر CAPTURE را می توان با دستور VAR = CAPTURE در یک متغیر از جنس word قرار داد . راه اندازی کانتر یک در مد CAPTURE با دستورات زیر انجام میشود:

```
Config Timer1 = Counter , Edge= Falling|Rising , Capture Edge=_ Falling |  
Rising , Noise Cancel=1|0 ,
```

Edge= Falling|Rising : با انتخاب EDGE = RISING کانتر نصبت به لبه ی بالا رونده حساس است و با انتخاب
EDGE = FALLING کانتر نصبت لبه ی پایین رونده حساس است.

Capture Edge= Falling | Rising : این گزینه مشخص میکند ، پالس اعمالی به پایه icp بالا رونده (Falling) است
یا پایین رونده (Rising). (CAPTURE) به کدام پالس حساسیت نشان میدهد).

Noise Cancel=1|0 : در صورت انتخاب یک از نویز های موجود بر روی پایه icp چشم پوشی میشود و هر پالس با
لبه ی تعیین شده میتواند CAPTURE را راه اندازی کند ، در صورت انتخاب صفر فقط پالس های با دامنه ی 5 ولت قادر به
راه اندازی CAPTURE خواهند بود. مانند:

```
$regfile = "m16def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Counter , Edge = Falling , Capture Edge = Rising , Noise  
Cancel = 0
```

PEYMAN KHODDAMI

Do

Locate 1 , 1

Lcd Counter1

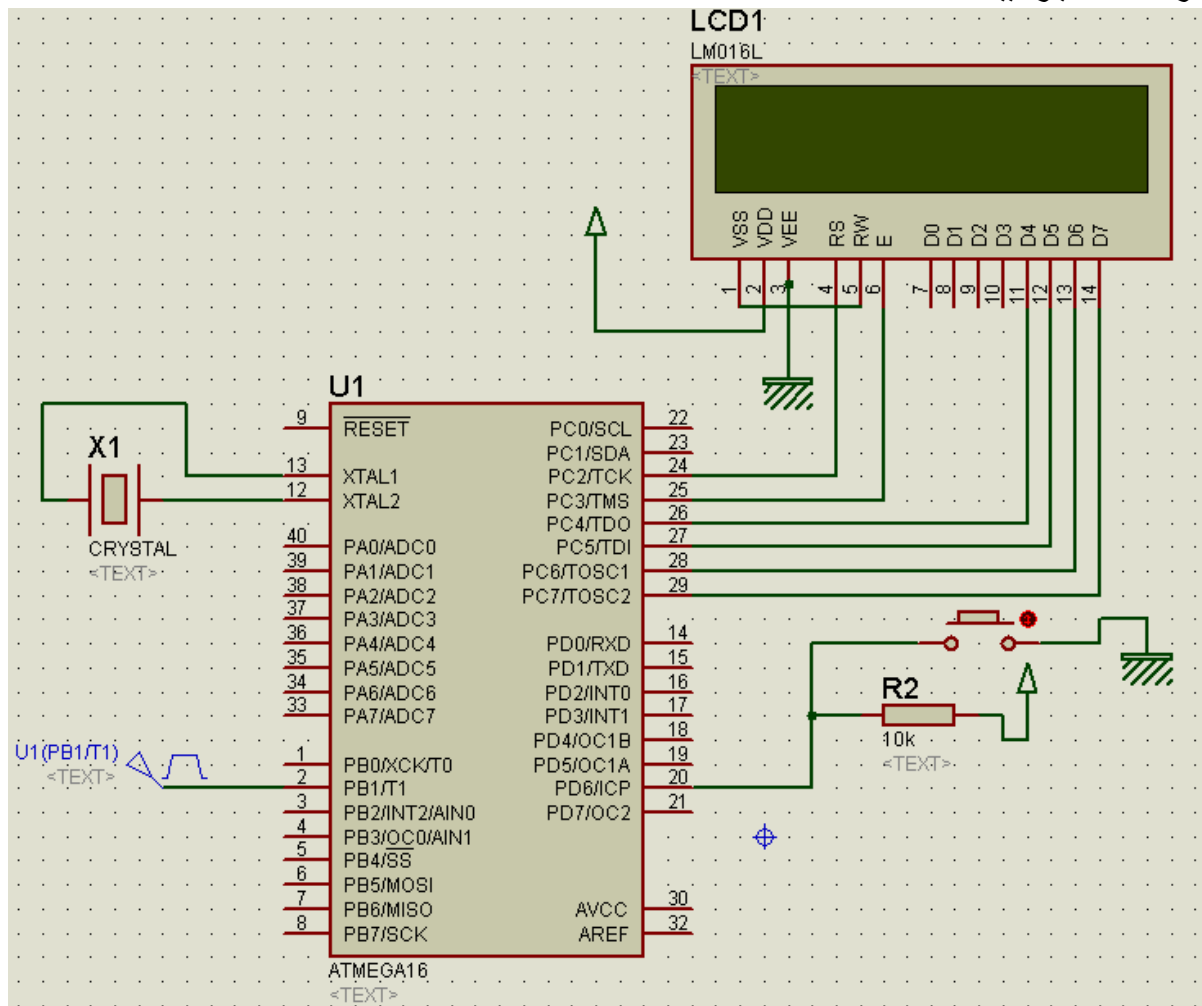
Locate 2 , 1

Lcd Capture1

Loop

End

در مثال بالا هنگامی که یک پالس بالا رونده به پایه icp اعمال میشود مقدار شمرده شده توسط تایمر 1 درون رجیستر Capture قرار میگیرد و سپس بر روی lcd نمایش داده میشود. مدار استفاده شده برای مثال بالا و سایر مثال های این بخش را مشاهده میفرمایید:



PEYMAN KHODDAMI

< استفاده از وقفه مد Capture کانتر 1:

مد Capture کانتر یک نیز مانند دیگر مد ها دارای یک منبع وقفه است که شما میتوانید با دستور Enable Icp1 ان را فعال کنید تا با دستور On icp1 lable ، هنگامی که پالسی به پایه icp اعمال شد ، cpu به برچسب مورد نظر پرش کند و در انجا عملیات دلخواه را انجام دهد .مانند:

```
$regfile = "m16def.dat"

$crystal = 1000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =
Portc.7 , E = Portc.3 , Rs = Portc.2

Config Timer1 = Counter , Edge = Falling , Capture Edge = Rising , Noise
Cancel = 0

Enable Interrupts

Enable Icp1

On Icp1 Q

Do

Locate 1 , 1

Lcd Counter1

Loop

End

Q:

Locate 2 , 1

Lcd Capture1

Return
```

PEYMAN KHODDAMI

در مثال بالا ، هنگامی که پالسی به پایه icp اعمال شود ، تعداد پالس شمرده شده توسط کانتر 1 در ریجیستر Capture1 ریخته میشود و cpu به برچسب q برش میکند و در انجا مقدار Capture1 را بر روی lcd نمایش میدهد.

< پیکر بندی تایمر/کانتر یک در مد PWM :

pwm یا مدولاسین پهنای پالس یکی از امکانات پرکاربرد در avr میباشد ، از این موج برای کنترل دور موتور ساخت پالس مربعی و دیگر پالس ها و ... استفاده میشود ، در این نوع مدولاسیون دامنه پالس ثابت است و نسبت زمان صفر به یک ان تغییر میکند برای درک بیشتر موضوع بهتر است کلیه مثالها را اجرا کنید. راه اندازی تایمر/کانتر 1 در مد pwm با دستورات زیر انجام میشود :

```
Config Timer1 = Pwm,Pwm = 8|9|10 , Compare A Pwm=Clear Up |Clear Down  
|Disconnect ,Compare B Pwm =Clear Up |Clear Down |Disconnect ,  
Prescale=1|8|64|256|1024
```

Pwm = 8|9|10 :pwm میتواند 8 یا 9 یا 10 بیتی باشد که مقدار بیت هرچه بیشتر باشد دقت موج بیشتر است (تعداد پله بیشتر است) pwm 8 بیتی تا 256 سرریز میشود (شما میتوانید 256 واحد ان را کم یا زیاد کنید) pwm 9 بیتی تا 512 و pwm 10 بیتی تا 1024 سرریز میشود.

Compare A Pwm=Clear Up |Clear Down |Disconnect : در صورت استفاده از گزینه Clear Up ، موج pwm از سطح 1 شروع میشود و در صورت انتخاب Clear Down ، موج pwm از سطح صفر شروع میشود و در صورت انتخاب Disconnect ، هنگامی که مقدار pwm با pwm1a که در برنامه مشخص میشود برابر شد ، ارتباط پالس با پایه ی oc1a قطع میشود .

Compare b Pwm=Clear Up |Clear Down |Disconnect : در صورت استفاده از گزینه Clear Up ، موج pwm از سطح 1 شروع میشود و در صورت انتخاب Clear Down ، موج pwm از سطح صفر شروع میشود و در صورت انتخاب Disconnect ، هنگامی که مقدار pwm با pwm1b که در برنامه مشخص میشود برابر شد ، ارتباط پالس با پایه ی oc1b قطع میشود .

Prescale : این گزینه و مقدار کریستال در تعیین فرکانس pwm نقش دارند . برای تولید PWM با فرکانس های متفاوت از این گزینه ها استفاده می شود.

با استفاده از دو دستور زیر میتوان یک عدد ثابت یا متغییر را در ریجیستر pwm قرار داد تا مقدار pwm با انها مقایسه شود:

Pwm1a=x

PEYMAN KHODDAMI

Pwm1b=x

یا

COMPARE1A = x

COMPARE1B = x

مثال:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =  
Portc.7 , E = Portc.3 , Rs = Portc.2
```

```
Config Timer1 = Pwm , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm =  
Clear Down , Prescale = 1
```

```
Dim A As Byte
```

```
Dim B As Byte
```

```
Config portd = output
```

```
Do
```

```
Pwm1a = A
```

```
Pwm1b = B
```

```
Incr A
```

```
Incr B
```

```
Waitms 500
```

```
Loop
```

```
End
```

در مثال بالا مقدار دو متغیر a و b در رجیستر pwm قرار داده شده اند ، مقدار آنها هر یک میلی ثانیه افزایش میابد .

PEYMAN KHODDAMI

زمان تناوب پالس pwm از رابطه ی زیر بدست میاید:

$$\text{ایمر بیت} = \frac{\text{زمان}}{\text{فرکانس}}$$

فرکانس = 1 / زمان

<< راه اندازی تایمر/کانتر دو در محیط بسکام:

- 1- این تایمر /کانتر 8 بیتی است ، و نهایتا میتواند تا 2^8 (255) بشمارد.
- 2- کلاک این تایمر میتواند توسط نوسان ساز داخلی یا نوسان ساز خارجی یا از دو پایه toc1 و toc2 (در مگا 16 پایه شماره 28 و 29 (portc.6 و portc.7)) توسط کریستال 32768 هرتز، تامین شود (مقدار فرکانس نوسان ساز بر عدد PRESCALE تقسیم میشود).
- 3- تایمر / کانتر صفر دارای یک خروجی مقایسه ای است که جیستر ocr2 مقدارمقایسه ای را در خود جای می دهد و با محتوای تایمر/کانتر مقایسه می کند .
- 4- در زمان تساوی محتوای رجیستر مقایسه و محتوای تایمر/کانتر ، وضعیت پایه های خروجی مد مقایسه ای ocr2 همان گونه که در برنامه مشخص میشود ، می تواند تغییر کند.
- 5- این تایمر دارای چندین منبع وقفه میباشد که شما میتوانید انها را در هر قسمت از برنامه فعال یا غیر فعال کنید. از این منابع و قفه میتوان ،وقفه سرریزی را نام برد.
- 6- این تایمر/ کانتر میتواند در مد تایمر و کانتر و pwm راه اندازی شود.(در بعضی از میکروها مانند 8535 و مگا 32 و... مد کانتر وجود ندارد ، برای کسب اطلاعات بیشتر در مورد امکانات avr به دیتاشیت انها مراجعه کنید(پایه t2 وجود ندارد))
- 7- ورودی کانتر پایه T2 (در میکرو مگا 103 پورت d.7 ، پایه شماره 32)میباشد و خروجی پالس pwm پایه oc2 (در میکرو مگا 16 پایه 21 (portd.7))است همچنین پایه ی فوق میتواند به عنوان خروجی مد مقایسه ای تایمر مورد استفاده قرار گیرد.

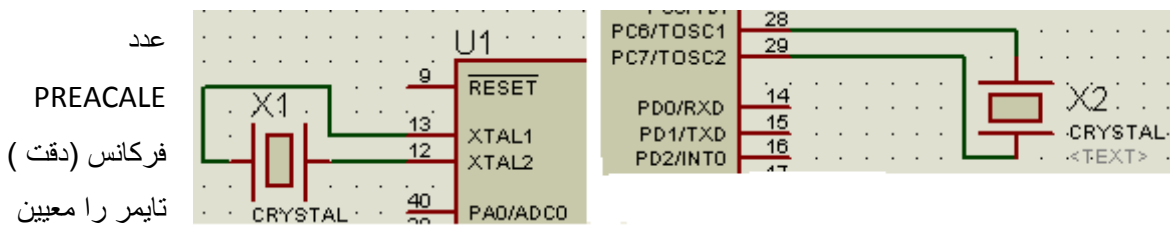
PEYMAN KHODDAMI

< راه اندازی تایمر دو در محیط بسکام:

تایمر دو با دستور زیر پیکر بندی میشود:

```
CONFIG TIMER2 = TIMER , ASYNC = ON|OFF , PREACALE=1|8|32|64|128|256|1024
```

ASYNC = ON|OFF : زمانی که ON انتخاب شود ، تایمر کلاک مورد نیاز را از پایه های TOSC1 و TOSC2 با کریستال 32768HZ در یافت می کند . در این حالت با PRESCALE=128 دقیقاً تایمر بعد از یک ثانیه سرریز میشود ، شما با استفاده از وقفه ها میتوانید به یک برجسب پرش کرده و یک متغییر را بیافزاید و یک ساعت دقیق بسازید ، در صورت انتخاب off تایمر کلاک خود را از کریستال داخلی یا کریستال متصل به دو پایه xtal1 و xtal2 تامین میکند.



میکند .فرکانس و زمانی که تایمر می شمارد، از فرمولهای زیر محاسبه میشوند: (ممکن است در بعضی از میکرو ها اعداد 32 و 128 موجود نباشد)

$$\text{زمان} = \frac{\text{PREACALE} * \text{تایمر بیت}}{\text{مقدار کریستال}}$$

$$\text{فرکانس} = \frac{\text{مقدار کریستال}}{\text{PREACALE}}$$

مثالا با استفاده از کریستال 32768HZ و PRESCALE=128 زمانی که تایمر می شمارد برابر است با:

$$\text{زمان} = \frac{2^8 * 128}{32768} = 1s$$

بعد از دستور بالا، تایمر با دستور START TIMER شروع به شمارش میکند و با دستور STOP TIMER متوقف می

شود. تایمر پس از شمردن تا 255 سر ریز میشود ، شما با استفاده از دستور ENABLE OVF2 و ENABLE

INTRRUPTS میتوانید وقفه سر ریزی تایمر را راه اندازی کنید . در صورتی که وقفه سرریزی تایمر فعال باشد ، تایمر

پس از سرریزی به برجسبی که با یکی از دستورات ON OVF2 LABLE و یا ON TIMER2 LABLE مشخص شده

پرش میکند ، باز گشت از وقفه با دستور RETURN انجام میشود.

شما همچنین میتوانید با استفاده از دستور VAR = TIMER2 ، مقدار تایمر را در یک متغییر از جنس بایت قرار دهید و یا

با دستور TIMER2 = VALUE ، مقدار اولیه ای در تایمر قرار داد تا تایمر از ان شروع به شمارش کند.مانند:

PEYMAN KHODDAMI

```
$regfile = "M16DEF.DAT"

$crystal = 1000000

Config Lcdpin = Pin , Db4 = Pinc.2 , Db5 = Pinc.3 , Db6 = Pinc.4 , Db7 =
Pinc.5 , E = Pinc.1 , Rs = Pinc.0

Config Lcd = 16 * 2

Dim A As Byte , B As Byte

Config Timer2 = Timer , Prescale = 1024

Enable Interrupts

Enable Ovf2

On Ovf2 R

Timer2 = 12

Start Timer2

Do

Loop

End

R:

Incr A

If A > 3 Then : Incr B : Locate 1 , 1 : Lcd B : A = 0 : End If

Return
```

در مثال بالا توسط تایمر 2 زمان 0.999424 ثانیه ایجاد شده است (مقدار اولیه 12 در تایمر قرار داده شده ، پس تایمر از 12 تا 256 می‌شمارد ، یعنی 244 ، پس در فرمول به جای بیت تایمر (256) عدد 244 قرار می‌گیرد و زمان فوق بدست می‌آید ، شما می‌توانید با این روش زمان های دیگری را نیز بدست آورید.)

<راه اندازی تایمر 2 در مد مقایسه ای (Compare):

PEYMAN KHODDAMI

تایمر دو با دستور زیر در مد مقایسه ای پیکر بندی میشود (در این مد مقدار شمرده شده توسط تایمر با عددی که شما در برنامه تعیین میکنید ، مقایسه میشود و در صورت اختلاف مقدار شمرده شده با عدد شما وضعیت پایه oc2 (پایه 21 مگا 16) مطابق آنچه که در برنامه تعیین کردید تغییر میکند)

```
CONFIG TIMER2 = TIMER , COMPARE = CLEAR | SET | TOGGLE | DISCONNECT , PRESCALE = 1 | 8  
| 32 | 64 | 256 | 1024 , CLEAR TIMER = _ 1 | 0
```

COMPARE = CLEAR | SET | TOGGLE | DISCONNECT : زمانی که مقدار شمرده شده توسط تایمر 2 با مقدار COMPARE ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی oc2 می تواند SET (یک) ، (صفر) ، CLEAR (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc2 به یک ورودی خروجی عادی تبدیل شود).

PRESCALE = 1 | 8 | 32 | 64 | 256 | 1024 : عدد PRESCALE فرکانس (دقت) تایمر را معین میکند. (ممکن است در بعضی از میکرو ها اعداد 32 و 128 موجود نباشد)

CLEAR TIMER = 1 | 0 : با انتخاب گزینه 1 ، محتوای تایمر/کانتر در زمان تطابق مقایسه ای RESET (\$0000) می شود و در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی را که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Compare1 = x
```

یا

```
Ocr2=x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد ، هنگامی که عدد شمرده شده توسط تایمر یک با اعداد گذاشته شده برابر شد میکرو وضعیت پایه های مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد. مثال:

```
$regfile = "M16DEF.DAT"
```

```
$crystal = 1000000
```

```
Config Lcdpin = Pin , Db4 = Pinc.2 , Db5 = Pinc.3 , Db6 = Pinc.4 , Db7 =  
Pinc.5 , E = Pinc.1 , Rs = Pinc.0
```

```
Config Lcd = 16 * 2
```

```
Config Timer2 = Timer , Compare = Toggle , Prescale = 64 , Clear Timer = 0
```

```
Ocr2 = 100
```

PEYMAN KHODDAMI

```
Start Timer2
```

```
Config Portd = Output
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Timer2
```

```
Loop
```

```
End
```

در مثال بالا هنگامی که مقدار شمرده شده با مقدار Compare (100) برابر شد وضعیت پایه oc2 تغییر میکند.

<استفاده از وقفه مد مقایسه ای تایمر 2 :

مد مقایسه ای تایمر دو دارای 1 منبع وقفه میباشد که با دستورات زیر فعال میشوند:

```
Enable Interrupts
```

```
Enable Oc2
```

با دستور فعال سازی وقفه مقایسه ، هنگامی که مقدار شمرده شده توسط میکرو با Compare (یا ocr2) برابر شد cpu میکرو با دستور on oc2 lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد، در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی پرش میکند.مانند

```
$regfile = "M16DEF.DAT" : $crystal = 4000000
```

```
Config Lcdpin = Pin , Db4 = Pinc.2 , Db5 = Pinc.3 , Db6 = Pinc.4 , Db7 =  
Pinc.5 , E = Pinc.1 , Rs = Pinc.0
```

```
Config Lcd = 16 * 2
```

```
Config Timer2 = Timer , Compare = Toggle , Prescale = 128 , Clear Timer = 1
```

```
Ocr2 = 125 : Cursor Off
```

```
Dim A As Byte , B As Byte , C As Byte , D As Byte
```

```
Enable Interrupts
```

```
Enable Oc2
```

PEYMAN KHODDAMI

```
On Oc2 Q
Start Timer2
Config Portd = Output : D = 1 : C = 0 : B = 0
Do
Locate 1 , 1
Lcd D ; " " ; C ; " " ; B" " ;
Loop
End
Q:
Incr A
If A > 249 Then : Incr B : A = 0 : End If
If B > 59 Then : Incr C : B = 0 : End If
If C > 49 Then : Incr D : C = 0 : End If
If D > 12 Then : D = 1 : End If
Return
```

در مثال بالا با استفاده از مد مقایسه ای برنامه یک ساعت نوشته شده است ، این ساعت دقیق است و زمان ساخته شده توسط تایمر دقیقا یک ثانیه می باشد . در برنامه بالا هنگامی که تایمر تا 125 می شمارد (مقدار شمرده شده (بیت تایمر = 125) مقدار آن با مقدار مقایسه ای (Ocr2) برابر میشود ، بنابراین cpu میکرو به برچسب q پرش کرده و در آنجا یک واحد به متغیر a افزوده میشود ، هنگامی که a برابر با 250 شد، یک واحد به متغیر ثانیه افزوده میشود و مقدار a صفر شده و موارد گفته شده تکرار می گردد ، هنگامی که b=60 شد (ثانیه 60 شد) به متغیر دقیقه یک واحد افزوده میشود و مقدار b صفر میشود ، این کار برای متغیر ساعت نیز تکرار می گردد ...

مقدار کریستال/ (مقداری که توسط تایمر شمرده میشود (بیت تایمر * تعداد حلقه)*پریسکلیر)=زمان

$$T=(128 (250*125))/4000000=1s$$

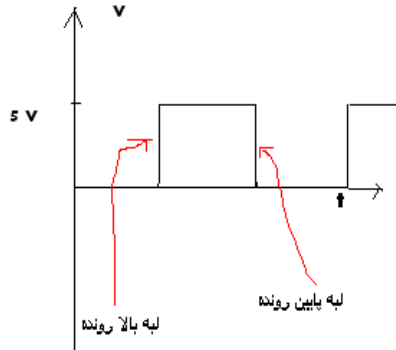
شما میتوانید زمانهای گوناگون را با روش های مختلف بسازید .

PEYMAN KHODDAMI

< راه اندازی کانتر دو در محیط بسکام:

کانتر دو در بسکام با دستور زیر پیکربندی میشود:

```
CONFIG TIMER2 = COUNTER , EDGE = RISING / FALLING
```



با انتخاب $EDGE = RISING$ کانتر نسبت به لبه ی بالا رونده حساس است

با انتخاب $EDGE = FALLING$ کانتر نسبت لبه ی پایین رونده حساس است

(لبه های بالا رونده یا پایین رونده را می‌شمارد (بالا رونده < صفر به یک و پایین رونده > یک به صفر))

شما همچنین میتوانید با استفاده از دستور $VAR = COUNTER2$ ، مقدار کانتر را

در یک متغیر از جنس بایت قرار دهید و یا با دستور $COUNTER2 = VALUE$ ، مقدار اولیه ای در کانتر قرار دهید تا کانتر از آن شروع به شمارش کند.

کانتر نیز مانند تایمر پس از شمردن تا 255 سر ریز میشود ، شما با استفاده از دستور $ENABLE OVF2$ و $Enable$ Interrupts میتوانید وقفه سر ریزی کانتر را راه اندازی کنید . در صورتی که وقفه سر ریزی کانتر فعال باشد ، کانتر پس از سر ریزی به برجسبی که با یکی از دستورات $ON OVF2 LABLE$ و یا $ON counter2 LABLE$ مشخص شده پرش میکند ، باز گشت از وقفه با دستور $RETURN$ انجام میشود.(این حالت در مواردی که رقم شمرده شده توسط کانتر از 255 بیشتر میشود ، کاربرد دارد)

در این مثال کانتر صفر تعداد پالسهای اعمالی به پایه t2 (32 میکرو مگا 103) را می‌شمارد و بر روی lcd نمایش میدهد:

```
$regfile = "M103DEF.DAT" : $crystal = 4000000
```

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , E = Pinb.1 , Rs = Pinb.0
```

```
Config Lcd = 16 * 2
```

```
Config Timer2 = Counter , Edge = Rising
```

```
ENABLE TIMER2
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter2
```

PEYMAN KHODDAMI

Loop

End

مثال:

```
$regfile = "M103DEF.DAT" : $crystal = 4000000
```

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , E = Pinb.1 , Rs = Pinb.0
```

```
Config Lcd = 16 * 2
```

```
Dim A As Word
```

```
Config Timer2 = Counter , Edge = Rising
```

```
Enable Timer2
```

```
Enable Interrupts
```

```
Enable Ovf2
```

```
On Counter2 Q
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter2
```

```
Loop
```

```
End
```

```
Q:
```

```
A = A + Counter2 : Locate 2 , 1 : Lcd A
```

```
Return
```

در مثال بالا از وقفه کانتر استفاده شده است ، هنگامی که کانتر سرریز شد (255 پالس اعمای را شمرد) به زیر برنامه q پرش میشود و در آنجا مقدار شمرده شده توسط کانتر با متغیر a جمع میشود و متغیر a در سطر اول ستون دوم lcd نمایش داده میشود و ... ، بدین ترتیب کانتر میتواند تا 65536 پالس را بشمارد.

<راه اندازی کانتر 2 در مد مقایسه ای (Compare):

PEYMAN KHODDAMI

در این مد شما میتوانید مقدار کانتر 2 را با یک عدد دلخواه مقایسه کنید و در صورت برابری یا نابرابری مقدار شمرده شده توسط کانتر با مقدار دلخواه وضعیت پایه OC2 را تغییر دهید. راه اندازی کانتر دو در مد مقایسه ای با دستورات زیر انجام میشود:

```
CONFIG TIMER2 = COUNTER , EDGE = RISING / FALLING, COMPARE = CLEAR  
|SET|TOGGLE| DISCONNECT, CLEAR TIMER = 1|0
```

EDGE = RISING / FALLING : با انتخاب EDGE = RISING کانتر نصبت به لبه ی بالا رونده حساس است و با انتخاب EDGE = FALLING کانتر نصبت لبه ی پایین رونده حساس است.

COMPARE = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط کانتر دو با مقدار COMPARE ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC2 می تواند SET (یک) ، (صفر) ، CLEAR (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه OC2 به یک ورودی خروجی عادی تبدیل شود).

CLEAR TIMER = 1|0 : با انتخاب گزینه 1 ، محتوای کانتر دو در زمان تطابق مقایسه ای RESET (\$0000) می شود و در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Ocr2=x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد ، هنگامی که عدد شمرده شده توسط کانتر دو با اعداد گذاشته شده برابر شد میکرو وضعیت پایه مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد. مثال:

```
$regfile = "M103DEF.DAT" : $crystal = 4000000
```

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , E = Pinb.1 , Rs = Pinb.0
```

```
Config Lcd = 16 * 2
```

```
Dim A As Word
```

```
Config Timer2 = Counter , Edge = Rising , Compare = Toggle , Clear Timer = 1
```

```
Enable Oc2
```

```
Ocr2 = 100
```

```
Config Portb.7 = Output
```


PEYMAN KHODDAMI

Do

Locate 1 , 1 : Lcd Counter2

Loop

End

هنگامی که پالس های شمرده شده توسط کانتر 1 به 100 رسید ، پایه oc2 تغییر وضعیت میدهد (اگر صفر باشد یک میشود و بلعکس)

<استفاده از وقفه مد مقایسه ای کانتر دو :

مد مقایسه ای دارای یک منبع وقفه میباشد که با دستور زیر فعال میشوند:

Enable Interrupts

Enable Oc2

با دستور فعال سازی وقفه مقایسه ، هنگامی که تعداد پالس شمرده شده توسط کانتر با Compare برابر شد cpu میکرو با دستور on oc2 lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد. در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی پرش میکند.مانند:

```
$regfile = "M103DEF.DAT" : $crystal = 4000000
```

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , E = Pinb.1 , Rs = Pinb.0
```

```
Config Lcd = 16 * 2
```

```
Dim A As Word
```

```
Config Timer2 = Counter , Edge = Rising , Compare = Toggle , Clear Timer = 1
```

```
Enable Interrupts
```

```
Enable Oc2
```

```
Ocr2 = 100
```

```
On Oc2 B
```

```
Config Portb.7 = Output
```

PEYMAN KHODDAMI

```
Config Portb.6 = Output
```

```
Do
```

```
Locate 1 , 1 : Lcd Counter2
```

```
Loop
```

```
End
```

```
B:
```

```
Toggle Portb.6
```

```
return
```

در مثال بالا هنگامی که تعداد پالس شمرده شده توسط کانتر با مقدار Compare (100) برابر شد cpu به برچسب b پرش میکند و در آنجا پایه b.6 تغییر وضعیت میدهد و با دستور return به حلقه ی اصلی برمیگردد.

<راه اندازی تایمر/ کانتر دو در مد PWM :

همانگونه که اشاره شد تایمر کانتر دو دارای یک خروجی PWM 8 بیتی میباشد ، این خروجی پایه OC2 میکرو است در زیر نحوه راه اندازی تایمر کانتر دو در مد مدولاسیون عرض پالس آورده شده است:

```
Config Timer2 = Pwm ,Pwm =ON|OFF , Compare Pwm=Clear Up|Clear Down|Disconnect  
, Prescale=1|8|32|64|128|256|1024
```

PWM=ON|OFF : این گزینه برای روشن یا خاموش کردن PWM است ، برای راه اندازی PWM از ON استفاده میکنیم.

Compare A Pwm=Clear Up | Clear Down | Disconnect : در صورت استفاده از گزینه Clear Up ، موج pwm از سطح 1 شروع میشود و در صورت انتخاب Clear Down ، موج pwm از سطح صفر شروع میشود و در صورت انتخاب Disconnect ، هنگامی که مقدار pwm با pwm1a که در برنامه مشخص میشود برابر شد ، ارتباط پالس با پایه ی OC2 قطع میشود. (OC2 به ورودی خروجی عادی تبدیل میشود)

Prescale : این گزینه و مقدار کریستال در تعیین فرکانس pwm نقش دارند . برای تولید PWM با فرکانس های متفاوت از این گزینه ها استفاده می شود.

با استفاده از دستور زیر میتوان یک عدد ثابت یا متغییر را در ریجیستر pwm قرار داد تا مقدار pwm با انها مقایسه شود:

PEYMAN KHODDAMI

OCR2=X

مثال:

```
$regfile = "M8DEF.DAT" : $crystal = 8000000

Config Portb.3 = Output

Config Timer2 = Pwm , Pwm = On , Compare Pwm = Clear Up , Prescale = 64

Dim A As Byte

Do

Ocr2 = A

Waitms 500 : Incr A

Loop

End
```

<< راه اندازی تایمر/کانتر سه در محیط بسکام:

چهارمین تایمر کانتری که معرفی میشود تایمر سه است ، در زیر مشخصات این مورد را مشاهده میفرمایید:

- 1- این تایمر کانتر 16 بیتی است ، و نهایتاً میتواند تا 2^{16} (65535) بشمارد.
- 2- کلاک این تایمر میتواند توسط نوسان ساز داخلی یا نوسان ساز خارجی یا از پایه t3 توسط پالس خارجی، تامین شود (مقدار فرکانس نوسان ساز بر عدد PRESCALE تقسیم میشود).
- 3- تایمر / کانتر یک دارای دو خروجی مقایسه ای است که دو رجیستر OC3A و OC3B مقدار مقایسه ای را در خود جای می دهند و با محتوای تایمر/کانتر مقایسه می شوند .
- 4- در زمان تساوی محتوای رجیستر مقایسه و محتوای تایمر/کانتر ، وضعیت پایه های خروجی مد مقایسه ای OC3A و OC3B می تواند تغییر کند.
- 5- تایمر / کانتر در مد CAPTURE نیز می تواند به کار رود . با تحریک پایه ICp3 می توان محتوای تایمر/کانتر را در رجیستر ورودی CAPTURE قرار داد.
- 6- این تایمر دارای چندین منبع وقفه میباشد که شما میتوانید آنها را در هر قسمت از برنامه فعال یا غیر فعال کنید. از این منابع و وقفه میتوان ،وقفه سرریزی را نام برد.
- 7- این تایمر کانتر میتواند در مد تایمر و کانتر و pwm راه اندازی شود.

PEYMAN KHODDAMI

- 8- ورودی کانتر پایه T3 می باشد و خروجی پالس pwm پایه های oc3a و oc3b است همچنین دوپایه ی فوق میتوانند به عنوان خروجی مد مقایسه ای تایمر مورد استفاده قرار بگیرند.
- 9- این تایمر/ کانتر فقط در میکرو های مگا 64 و 128 و 162 موجود است.

< راه اندازی تایمر سه در محیط بسکام:

تایمر 3 با دستور زیر پیکربندی میشود:

```
Config Timer3 = Timer , PRESCALE = 1 | 8 | 64 | 256 | 1024
```

عدد PRESCALE فرکانس (دقت) تایمر را معین میکند. فرکانس و زمانی که تایمر می شمارد از فرمولهای زیر محاسبه میشوند:

$$\text{زمان} = \frac{\text{PRESCALE} * \text{تایمر بیت}}{\text{مقدار کریستال}}$$

$$\text{فرکانس} = \frac{\text{مقدار کریستال}}{\text{PRESCALE}}$$

بعد از دستور بالا، تایمر با دستور START TIMER شروع به شمارش میکند و با دستور STOP TIMER متوقف می شود. تایمر پس از شمردن تا 65536 (2^{16}) سر ریز میشود ، شما با استفاده از دستور ENABLE OVF3 و ENABLE INTERRUPTS میتوانید وقفه سر ریزی تایمر را راه اندازی کنید . در صورتی که وقفه سر ریزی تایمر فعال باشد ، تایمر پس از سر ریزی به برجسبی که با یکی از دستورات ON OVF3 LABLE و یا ON TIMER3 LABLE مشخص شده پرش میکند ، باز گشت از وقفه با دستور RETURN انجام میشود.

شما همچنین میتوانید با استفاده از دستور VAR = TIMER3 ، مقدار تایمر را در یک متغیر از جنس word قرار دهید و یا با دستور TIMER3 = VALUE ، مقدار اولیه ای در تایمر قرار دهید تا تایمر از ان شروع به شمارش کند.

یک شمارنده معکوس بسازیم ، برای این کار ابتدا باید یک زمان 1 میلی ثانیه ای 3 در این مثال میخواهیم توسط تایمر ، 64 استفاده نمایید ، میتوانید زمانی برابر با PRESCALE ایجاد کنیم طبق فرمول اگر شما از کریستال 4 مگا هرتز و 1.04 ثانیه ایجاد کنید ، با مقادیر دیگر نیز میتوانید زمان های دقیق تر بسازید . (بیشترین زمانی که با این تایمر میتوانید بسازید برابر با 67.108864 ثانیه میباشد)، شما ابتدا باید یک مقدار را تعیین کنید ، میکرو بعد از گذشت 1 ثانیه از مقدار تعیین شده 1 واحد کم میکند

```
$regfile = "m64def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 4000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =
Portf.5 , E = Portf.1 , Rs = Portf.0

Config Timer3 = Timer , Prescale = 64

Dim A As Byte : A = 15

Enable Timer3

Start Timer3

Do

If Timer3 > 1 Then : A = A -1 : End If

Locate 1 , 1 : Lcd A

Loop

End
```

<راه اندازی تایمر 3 در مد مقایسه ای (Compare):

در این مد شما میتوانید مقدار تایمر 3 را با دو عدد دلخواه مقایسه کنید و در صورت برابری یا نابرابری مقدار تایمر با مقدار دلخواه وضعیت پایه های oc3a و oc3b (پایه های 5 و 6 میکرو مگا 64) را تغییر دهید. راه اندازی تایمر یک در مد مقایسه ای با دستورات زیر انجام میشود:

```
CONFIG TIMER3= TIMER,COMPARE A = CLEAR |SET|TOGGLE|DISCONNECT,COMPARE B =
CLEAR |SET|TOGGLE|DISCONNECT, PRESCALE=1|8|64|256|1024,CLEAR TIMER =1|0
```

COMPARE A = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط تایمر 3 با مقدار COMPARE A ، که بعدا معرفی میشود ، برابر شد ،پایه خروجی OC3A می تواند SET(یک), (صفر) CLEAR, (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc3a به یک ورودی خروجی عادی تبدیل شود).

COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط تایمر 3 با مقدار COMPARE B ، که بعدا معرفی میشود ، برابر شد ،پایه خروجی OC3B می تواند SET(یک), (صفر) CLEAR, (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc3b به یک ورودی خروجی عادی تبدیل شود).

PEYMAN KHODDAMI

PRESCALE=1|8|64|256|1024 : عدد PRESCALE فرکانس (دقت) تایمر را معین میکند.

CLEAR TIMER = 1|0 : با انتخاب گزینه 1، محتوای تایمر/کانتر در زمان تطابق مقایسه ای (RESET) (\$0000) می شود و در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی را که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Compare3a = x
```

```
Compare3b =x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد، هنگامی که عدد شمرده شده توسط تایمر یک با اعداد گذاشته شده برابر شد میکرو وضعیت پایه های مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد.مثال:

```
$regfile = "m64def.dat"
```

```
$crystal = 4000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Timer , Prescale = 64
```

```
Compare3a = 1000
```

```
Compare3b = 5000
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Timer3
```

```
Loop
```

```
End
```

هنگامی که تایمر 3 تا 1000 شمرده پایه oc3a یک میشود و هنگامی که مقدار شمرده شده به 5000 رسید پایه oc3b یک میشود (oc3a پایه 5 و oc3b پایه 6 مگا 64 میباشد).

استفاده از وقفه مد مقایسه ای تایمر 3 :

PEYMAN KHODDAMI

مد مقایسه ای دارای 2 منبع وقفه میباشد که با دستورات زیر فعال میشوند:

```
Enable Interrupts
```

```
Enable Oc3a
```

```
Enable Oc3b
```

با دستور فعال سازی وقفه مقایسه ، هنگامی که مقدار شمرده شده توسط میکرو با Compare3a برابر شد cpu میکرو با دستور on oc3a lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد، هنگامی که مقدار شمرده شده توسط میکرو با Compare3b برابر شد cpu میکرو با دستور on oc3b lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد. در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی برش میکند.مانند:

```
$regfile = "m64def.dat"
```

```
$crystal = 4000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Timer , Prescale = 64
```

```
Compare3a = 2120
```

```
Compare3b = 63000
```

```
Enable Interrupts
```

```
Enable Oc3a
```

```
Enable Oc3b
```

```
On Oc3a Q
```

```
On Oc3b W
```

```
Do
```

```
Locate 1 , 1 : Lcd Timer3
```

```
Loop
```

PEYMAN KHODDAMI

End

Q:

Locate 2 , 1 : Lcd Timer3

Return

W:

Locate 2 , 8 : Lcd Timer3

Return

در مثال بالا هنگامی که رقم شمرده شده توسط تایمر با مقدار Compare3a (2120) برابر شد cpu به برچسب q پرش میکند و در آنجا مقدار شمرده شده توسط تایمر را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد ، هنگامی که رقم شمرده شده توسط تایمر با مقدار Compare3b (63000) برابر شد cpu به برچسب w پرش میکند و در آنجا مقدار شمرده شده توسط تایمر را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد . همچنین در مورد اول پایه oc3a یک میشود و در مورد دوم وضعیت پایه oc3b تغییر میکند (در صورت 0 بودن یک میشود و بلعکس).

راه اندازی تایمر 3 در مد CAPTURE :

در صورتی که تایمر 3 را در این مد پیکربندی کنید ، با اعمال یک پالس بالا رونده یا پایین رونده (که نوع ان در هنگام پیکر بندی مشخص میشود) به پایه ic3 (پایه 9 مگا 64) ، در همان لحظه مقدار شمرده شده توسط تایمر 3 در رجیستر CAPTURE قرار میگیرد ، محتوای رجیستر CAPTURE را می توان با دستور VAR = CAPTURE در یک متغییر از جنس word قرار داد . راه اندازی تایمر 3 در مد CAPTURE با دستورات زیر انجام میشود:

```
Config Timer3 = Timer, Capture Edge= Falling | Rising ,Noise Cancel=1|0 ,  
Prescale =1|8|64|256|1024
```

Capture Edge= Falling | Rising : این گزینه مشخص میکند ، پالس اعمالی به پایه icp بالا رونده (Falling) است یا پایین رونده (Rising). (CAPTURE) به کدام پالس حساسیت نشان میدهد)

Noise Cancel=1|0 : در صورت انتخاب یک از نویز های موجود بر روی پایه icp چشم پوشی میشود و هر پالس با لبه ی تعیین شده میتواند CAPTURE را راه اندازی کند ،در صورت انتخاب صفر فقط پالس های با دامنه ی 5 ولت قادر به راه اندازی CAPTURE خواهند بود.

PEYMAN KHODDAMI

Prescale = 1|8|64|256|1024: دقت تایمر را نشان میدهد (با استفاده از این مقدار و مقدار کریستال میتوانید زمان شمرده شده توسط تایمر تا هنگام سرریزی را با فرمولی که در بالا گفته شد محاسبه کنید)
با دستور Enable Icp3 این مد فعال میشود و پایه icp آماده دریافت پالس میگردد . مانند:

```
$regfile = "m64def.dat"

$crystal = 4000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =
Portf.5 , E = Portf.1 , Rs = Portf.0

Config Timer3 = Timer , Capture Edge = Falling , Noise Cancel = 1 , Prescale
= 1024

Do

Locate 1 , 1

Lcd Timer3

Locate 2 , 1

Lcd Capture3

Loop

End
```

در مثال بالا هنگامی که یک پالس بالا رونده به پایه icp اعمال میشود مقدار شمرده شده توسط تایمر 3 درون رجیستر Capture قرار میگیرد و سپس بر روی Icd نمایش داده میشود.

استفاده از وقفه مد Capture تایمر 3:

مد Capture تایمر 3 نیز مانند دیگر مد ها دارای یک منبع وقفه است که شما میتوانید با دستور Enable Icp3 ان را فعال کنید تا با دستور On icp3 lable ، هنگامی که پالسی به پایه icp اعمال شد ، cpu به برجسب مورد نظر پرش کند و در انجا عملیات دلخواه را انجام دهد .

< راه اندازی کانتر 3 در محیط بسکام: (راه اندازی تایمر/کانتر 3 در مد کانتر در محیط بسکام)

PEYMAN KHODDAMI

کانتر 3 در بسکام با دستور زیر پیکربندی میشود:

```
CONFIG TIMER3 = COUNTER , EDGE = RISING / FALLING
```

با انتخاب EDGE = RISING کانتر نصبیت به لبه ی بالا رونده حساس است

با انتخاب EDGE = FALLING کانتر نصبیت لبه ی پایین رونده حساس است

(لبه های بالا رونده یا پایین رونده را می‌شمارد (بالا رونده <صفر به یک و پایین رونده >یک به صفر))

شما همچنین میتوانید با استفاده از دستور VAR = COUNTER3 ، مقدار کانتر را در یک متغیر از جنس word قرار دهید و یا با دستور COUNTER3 = VALUE ، مقدار اولیه ای را در کانتر قرار دهید تا کانتر از آن شروع به شمارش کند. کانتر نیز مانند تایمر پس از شمردن تا 65536 سر ریز میشود ، شما با استفاده از دستور ENABLE OV3F3 می‌توانید وقفه سر ریزی کانتر را راه اندازی کنید . در صورتی که وقفه سر ریزی کانتر فعال باشد ، کانتر پس از سر ریزی به برجسبی که با یکی از دستورات ON OV3F3 LABEL مشخص شده پرش میکند ، باز گشت از وقفه با دستور RETURN انجام میشود. در این مثال کانتر سه تعداد پالسهای اعمالی به پایه 3t (پایه شماره 8 مگا 64) را می‌شمارد.

```
" $regfile = "m64def.dat"
```

```
crystal = 4000000$
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =  
Portd.3 , E = Portd.4 , Rs = Portd.5
```

```
Config Timer3 = Counter , Edge = Falling
```

```
Do
```

```
Locate 1 , 1 : Lcd Counter3
```

```
Loop
```

```
End
```

کانتر سه را نیز میتوان مانند تایمر در دو مد مقایسه ای و Capture راه اندازی کرد که طریقه راه اندازی در زیر آورده شده است:

PEYMAN KHODDAMI

راه اندازی کانتر 3 در مد مقایسه ای (Compare):

در این مد شما میتوانید مقدار کانتر 3 را با دو عدد دلخواه مقایسه کنید و در صورت برابری یا نابرابری مقدار شمرده شده توسط کانتر با مقدار دلخواه وضعیت پایه های oc3a و oc3b را تغییر دهید. راه اندازی کانتر 3 در مد مقایسه ای با دستورات زیر انجام میشود:

```
CONFIG TIMER3 = COUNTER , EDGE = RISING / FALLING, COMPARE A = CLEAR  
|SET|TOGGLE| DISCONNECT, COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT, CLEAR TIMER  
= 1|0
```

EDGE = RISING / FALLING : با انتخاب EDGE = RISING کانتر نصبت به لبه ی بالا رونده حساس است و با انتخاب EDGE = FALLING کانتر نصبت لبه ی پایین رونده حساس است.

COMPARE A = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط کانتر 3 با مقدار COMPARE A ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC3A می تواند SET (یک)، CLEAR (صفر) ، (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc3a به یک ورودی خروجی عادی تبدیل شود).

COMPARE B = CLEAR |SET|TOGGLE|DISCONNECT : زمانی که مقدار شمرده شده توسط کانتر 3 با مقدار COMPARE B ، که بعدا معرفی میشود ، برابر شد ، پایه خروجی OC3B می تواند SET (یک)، CLEAR (صفر) ، (برعکس) TOGGLE و یا ارتباط پایه با مد مقایسه ای قطع شود (پایه oc3b به یک ورودی خروجی عادی تبدیل شود).

CLEAR TIMER = 1|0 : با انتخاب گزینه 1 ، محتوای کانتر 3 در زمان تطابق مقایسه ای RESET (\$0000) می شود و در صورت انتخاب 0 مقدار شمرده شده تغییری نمیکند.

با دستورات زیر میتوان عددی که محتوای تایمر باید با آن مقایسه شود را تعیین کرد

```
Compare3a = x
```

```
Compare3b =x
```

به جای x یک عدد ثابت یا یک متغیر قرار میگیرد ، هنگامی که عدد شمرده شده توسط کانتر 3 با اعداد گذاشته شده برابر شد میکرو وضعیت پایه های مربوطه را همانگونه که در پیکربندی مشخص کردید تغییر میدهد.

```
$regfile = "m64def.dat"
```

```
$crystal = 4000000
```

PEYMAN KHODDAMI

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Counter , Edge = Falling , Compare A = Set , Compare B = Set  
, Clear Timer = 0
```

```
Compare3a = 300
```

```
Compare3b = 400
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter3
```

```
Loop
```

```
End
```

هنگامی که پالس های شمرده شده توسط کانتر 3 به 300 رسید ، پایه oc3a یک میشود و هنگامی که مقدار شمرده شده به 400 رسید پایه oc3b یک میشود (oc1a پایه 15 و oc1b پایه 16 مگا 64 میباشد).

استفاده از وقفه مد مقایسه ای کانتر 3 :

مد مقایسه ای دارای 2 منبع وقفه میباشد که با دستورات زیر فعال میشوند:

```
Enable Interrupts
```

```
Enable Oc3a
```

```
Enable Oc3b
```

با دستور فعال سازی وقفه مقایسه ، هنگامی که تعداد پالس شمرده شده توسط کانتر با Compare3a برابر شد cpu میکرو با دستور on oc3a lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد، هنگامی که تعداد پالس شمرده شده توسط کانتر با Compare3b برابر شد cpu میکرو با دستور on oc3b lable به برچسب مورد نظر برش میکند و در انجا عملیات دلخواه را انجام میدهد. در صورتی که در پایان برچسب دستور return گذاشته شود cpu به حلقه ی اصلی برش میکند.مانند

```
:$regfile = "m64def.dat" : $crystal = 4000000
```

PEYMAN KHODDAMI

```
Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =
Portf.5 , E = Portf.1 , Rs = Portf.0

Config Timer3 = Counter , Edge = Falling , Compare A = Set , Compare B = Set
, Clear Timer = 0

Compare3a = 30

Compare3b = 40

Enable Interrupts

Enable Oc3a

Enable Oc3b

On Oc3a Q

On Oc3b W

Do

Locate 1 , 1 : Lcd Counter3

Loop

End

Q:

Cls : Locate 2 , 1 : Lcd " Counter3=30"

Return

W:

Cls : Locate 2 , 1 : Lcd " Counter3=40"

Return
```

در مثال بالا هنگامی که تعداد پالس شمرده شده توسط کانتر با مقدار Compare3a (30) برابر شد cpu به برچسب q پرش میکند و در آنجا عبارت "Counter3=30" را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد ، هنگامی که تعداد پالس شمرده شده توسط کانتر با مقدار Compare3b (40) برابر شد cpu به برچسب w

PEYMAN KHODDAMI

پرش میکند و در انجا عبارت "Counter3=40" را بر روی lcd نمایش داده و با دستور return به حلقه ی اصلی برمیگردد. همچنین در مورد اول پایه oc3a یک میشود و در مورد دوم وضعیت پایه oc3b یک میشود .

راه اندازی کانتر 3 در مد CAPTURE :

در صورتی که کانتر 3 را در این مد پیکربندی کنید ، با اعمال یک پالس بالا رونده یا پایین رونده (که نوع ان در هنگام پیکر بندی مشخص میشود) به پایه ic3 (پایه 9 مگا 64) ، در همان لحظه تعداد پالس شکرده شده توسط کانتر 3 در رجیستر CAPTURE قرار میگیرد ، محتوای رجیستر CAPTURE را می توان با دستور VAR = CAPTURE در یک متغیر از جنس word قرار داد . راه اندازی کانتر یک در مد CAPTURE با دستورات زیر انجام میشود:

```
Config Timer3 = Counter , Edge= Falling|Rising , Capture Edge=_ Falling |  
Rising , Noise Cancel=1|0 ,
```

Edge= Falling|Rising : با انتخاب EDGE = RISING کانتر نصبت به لبه ی بالا رونده حساس است و با انتخاب EDGE = FALLING کانتر نصبت لبه ی پایین رونده حساس است.

Capture Edge= Falling | Rising : این گزینه مشخص میکند ، پالس اعمالی به پایه ic3 بالا رونده (Falling) است یا پایین رونده (Rising). (CAPTURE) به کدام پالس حساسیت نشان میدهد)

Noise Cancel=1|0 : در صورت انتخاب یک از نویز های موجود بر روی پایه ic3 چشم پوشی میشود و هر پالس با لبه ی تعیین شده میتواند CAPTURE را راه اندازی کند ، در صورت انتخاب صفر فقط پالس های با دامنه ی 5 ولت قادر به راه اندازی CAPTURE خواهند بود. مانند

```
:$regfile = "m64def.dat" : $crystal = 4000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Counter , Edge = Falling , Capture Edge = Rising , Noise  
Cancel = 0
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter3
```

PEYMAN KHODDAMI

Locate 2 , 1

Lcd Capture3

Loop

End

در مثال بالا هنگامی که یک پالس بالا رونده به پایه ic3 اعمال میشود مقدار شمرده شده توسط تایمر 3 درون رجیستر Capture قرار میگیرد و سپس بر روی lcd نمایش داده میشود.

استفاده از وقفه مد Capture کانتر 3 :

مد Capture کانتر 3 نیز مانند دیگر مد ها دارای یک منبع وقفه است که شما میتوانید با دستور Enable Icp3 ان را فعال کنید تا با دستور On icp3 lable ، هنگامی که پالسی به پایه icp اعمال شد ، cpu به برچسب مورد نظر پرش کند و در انجا عملیات دلخواه را انجام دهد .مانند:

```
$regfile = "m64def.dat" : $crystal = 4000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Counter , Edge = Falling , Capture Edge = Rising , Noise  
Cancel = 0
```

```
Enable Interrupts
```

```
Enable Icp3
```

```
On Icp3 Q
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Counter3
```

```
Loop
```

```
End
```

```
Q:
```

PEYMAN KHODDAMI

Locate 2 , 1

Lcd Capture3

Return

در مثال بالا ، هنگامی که پالسی به پایه icp اعمال شود ، تعداد پالس شمرده شده توسط کانتر 3 در ریجیستر Capture3 ریخته میشود و cpu به برچسب q برش میکند و در انجا مقدار Capture3 را بر روی lcd نمایش میدهد.

<< پیکر بندی تایمر/کانتر 3 در مد PWM

راه اندازی تایمر/کانتر 3 در مد pwm با دستورات زیر انجام میشود :

```
Config Timer3 = Pwm,Pwm = 8|9|10 , Compare A Pwm=Clear Up | Clear Down | Disconnect  
,Compare B Pwm =Clear Up | Clear Down | Disconnect , Prescale=1|8|64|256|1024
```

Pwm = 8|9|10 :pwm میتواند 8 یا 9 یا 10 بیتی باشد که مقدار بیت هرچه بیشتر باشد دقت موج بیشتر است (تعداد پله بیشتر است) pwm 8 بیتی تا 256 سرریز میشود (شما میتوانید 256 واحد ان را کم یا زیاد کنید) pwm 9 بیتی تا 512 و pwm 10 بیتی تا 1024 سرریز میشود.

Compare A Pwm=Clear Up | Clear Down | Disconnect : در صورت استفاده از گزینه Clear Up ، موج pwm از سطح 1 شروع میشود و در صورت انتخاب Clear Down ، موج pwm از سطح صفر شروع میشود و در صورت انتخاب Disconnect ، هنگامی که مقدار pwm با pwm1a که در برنامه مشخص میشود برابر شد ، ارتباط پالس با پایه ی oc1a قطع میشود .

Compare b Pwm=Clear Up | Clear Down | Disconnect : در صورت استفاده از گزینه Clear Up ، موج pwm از سطح 1 شروع میشود و در صورت انتخاب Clear Down ، موج pwm از سطح صفر شروع میشود و در صورت انتخاب Disconnect ، هنگامی که مقدار pwm با pwm1b که در برنامه مشخص میشود برابر شد ، ارتباط پالس با پایه ی oc1b قطع میشود .

Prescale : این گزینه و مقدار کریستال در تعیین فرکانس pwm نقش دارند . برای تولید PWM با فرکانس های متفاوت از این گزینه ها استفاده می شود.

با استفاده از دو دستور زیر میتوان یک عدد ثابت یا متغیر را در ریجیستر pwm قرار داد تا مقدار pwm با انها مقایسه شود:

Pwm3a=x

PEYMAN KHODDAMI

Pwm3b=x

یا

COMPARE3A = x

COMPARE3B = x

مثال:

```
$regfile = "m64def.dat" : $crystal = 4000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Portf.2 , Db5 = Portf.3 , Db6 = Portf.4 , Db7 =  
Portf.5 , E = Portf.1 , Rs = Portf.0
```

```
Config Timer3 = Pwm , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm =  
Clear Down , Prescale = 1
```

```
Dim A As Word
```

```
Dim B As Word
```

```
Config Portb = Output
```

```
Do
```

```
Pwm3a = A
```

```
Pwm3b = B
```

```
Incr A
```

```
Incr B
```

```
Waitms 500
```

```
Loop
```

```
End
```

در مثال بالا مقدار دو متغیر a و b در رجیستر pwm قرار داده شده اند ، مقدار آنها هر 500 میلی ثانیه افزایش میابد .

زمان تناوب پالس pwm از رابطه ی زیر بدست میاید:

PEYMAN KHODDAMI

تایمر بیت = زمان

زمان / 1 = فرکانس

<<rtc (Real Time Counter) (شمارش گر زمان واقعی):

در بعضی از میکرو ها تایمر /کانتر دو یا صفر میتواند به صورت اسنکرون (در حالت اسنکرون کلاک تایمر توسط کریستال ساعت Hz32768 از دو پایه toc1,toc2 تامین میشود) کار کند و زمان و تاریخ را بشمارد ، این شمارش در همه حالتها حتی زمانی که میکرو در حالت power-save است نیز وجود دارد. همچنین ایسی های نیز وجود دارد که زمان و تاریخ را میشمارد و حتی قابلیت شمارش و ذخیره زمان و تاریخ شمرده شده را نیز دارد در زیر هر دو مورد را بررسی میکنیم:

Rtc داخلی میکرو :

این قابلیت در بعضی از میکروها وجود دارد ، که شما میتوانید با مراجعه به دیتا شیت میکرو از وجود یا عدم وجود آن مطلع شوید.rtc با دستور زیر پیکربندی میشود:

```
CONFIG CLOCK = soft | USER [, GOSUB = SECTIC]
```

Soft : هنگامی استفاده میشود که میخواهید از rtc داخلی میکرو استفاده کنید.

USER : هنگامی استفاده میشود که بخواهید از rtc خارجی (مانند Ds1307 یا...) (این ایسی از پروتکل I2C استفاده میکند ، کار با این ایسی در بخش "روشهای ارتباطی در avr" گفته شده است)) استفاده کنید.

GOSUB = SECTIC : این گزینه اختیاری است ، زمانی که تایمر سرریز شد (تا یک ثانیه شمرد) به برجسب SECTIC پرش میشود ، بازگشت از برجسب با دستور return صورت میگیرد

با دستورات زیر مقدار اولیه زمان و تاریخ مشخص می شوند(روش اول) :

```
DATE$ = "mm/dd/yy"
```

```
TIME$ = "hh:mm:ss"
```

به جای حروف اعداد اولیه قرار داده میشوند مانند:

```
$regfile = "M16DEF.DAT"
```

```
$crystal = 8000000
```

PEYMAN KHODDAMI

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.5 , E = Pind.1 , Rs = Pind.0
```

```
Config Lcd = 16 * 2
```

```
Enable Interrupts
```

```
Config Clock = Soft
```

```
Date$ = "11/11/00"
```

```
Time$ = "02:20:00"
```

```
Do
```

```
Locate 1 , 1
```

```
Lcd Date$
```

```
Locate 2 , 1
```

```
Lcd Time$
```

```
Loop
```

```
End
```

شما میتوانید با دستورات زیر مقدار اولیه ای را برای زمان و تاریخ مشخص کنید:

`_sec = X` x بین 0 تا 59 است

`_min = X` x بین 0 تا 59 است

`_hour = X` x بین 0 تا 23 است

`_day = X` x بین 1 تا 31 است

`_month = X` x بین 1 تا 12 است

`_year = x` x بین 0 تا 99 است

شما میتوانید کلیه اعمال جمع و تفریق (کاستن و افزودن) را مستقیم روی متغیرهای بالا انجام دهید. مثال:

```
$regfile = "M16DEF.DAT"
```

PEYMAN KHODDAMI

```
$crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 = Pind.5 , E =
Pind.1 , Rs = Pind.0

Config Lcd = 16 * 2

Config Portb = Input

Enable Interrupts

Config Date = YMD , Separator.=

Config Clock = Soft , Gosub = Sctic

_sec = 57 : _min = 59 : _hour = 23 : _day = 32 : _month = 11 : _year = 99

Goto W

Set_ok:

Locate 1 , 10 : Lcd "set ok"

Wait 2 : Locate 1 , 10 : Lcd "          "

W:

Do

Debounce Pinb.5 , 0 , Incr_sec

Loop

Incr_sec:

Do

Locate 1 , 10 : Lcd "set sec"

Debounce Pinb.5 , 0 , Incr_min : Waitms 100

If Pinb.6 = 0 Then : _sec = 0 : Waitms 400 : End If

If Pinb.7 = 0 Then : _sec = 30 : Waitms 400 : End If

Loop

Incr_min:

Do
```

PEYMAN KHODDAMI

```
Locate 1 , 10 : Lcd "set min" : Waitms 100

Debounce Pinb.5 , 0 , Incr_hour

If Pinb.6 = 0 Then : Incr _min: Waitms 400 : End If

If Pinb.7 = 0 Then : Decr _min : Waitms 400 : End If

Loop

Incr_hour:

Do

Locate 1 , 10 : Lcd "set hou"

Debounce Pinb.5 , 0 , Incr_day:Waitms 100

If Pinb.6 = 0 Then : Incr _hour : Waitms 400 : End If

If Pinb.7 = 0 Then : Decr _hour: Waitms 400 : End If

Loop

Incr_day:

Do

Locate 1 , 10 : Lcd "set day"

Debounce Pinb.5 , 0 , Incr_month:Waitms 100

If Pinb.6 = 0 Then : Incr _day : Waitms 400 : End If

If Pinb.7 = 0 Then : Decr _day : Waitms 400 : End If

Loop

Incr_month:

Do

Locate 1 , 10 : Lcd "set mon"

Debounce Pinb.5 , 0 , Incr_year :Waitms 100

If Pinb.6 = 0 Then : Incr _month : Waitms 400 : End If

If Pinb.7 = 0 Then : Decr _month : Waitms 400 : End If

Loop
```

PEYMAN KHODDAMI

```
Incr_year:

Do

Locate 1 , 10 : Lcd "set yea"

Debounce Pinb.5 , 0 , Set_ok : Waitms 100

If Pinb.6 = 0 Then : Incr _year : Waitms 400 : End If

If Pinb.7 = 0 Then : Decr _year : Waitms 400 : End If

Loop

End

Sectic:

Locate 1 , 1 : Lcd Date$ : Locate 2 , 1 : Lcd Time$

Return

:مثال

$regfile = "M16DEF.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 = Pind.5 , E =
Pind.1 , Rs = Pind.0

Config Lcd = 16 * 2

Enable Interrupts

Cursor Off

Config Clock = Soft , Gosub = Sectic

Config Adc = Single , Prescaler = Auto

Dim Q(2) As Word

Date$ = "11/11/00"

Time$ = "02:20:00"

Start Adc

Do
```

PEYMAN KHODDAMI

```
Q(1) = Getadc(0 (
Q(2) = Getadc(1 (
Q(1) = Q(1) / 2
Q(2) = Q(2) / 2
Locate 1 , 11
Lcd "t1:" ; Q(1 (
Locate 2 , 11
Lcd "t2:" ; Q(2 (
Loop
End
Sectic:
Locate 1 , 1
Lcd Date$
Locate 2 , 1
Lcd Time$
Return
```

برنامه بالا علاوه بر نمایش ساعت و تقویم دمای دو نقطه ی دلخواه را نیز نمایش میدهد ، همانگونه که مشاهده میفرمایید برنامه اسکن کانال های adc در حلقه ی do-loop قرار داده شده ، هنگامی که تایمر سرریز میشود به پرچسب Sectic پرش میشود و زمان و تاریخ بر روی lcd نمایش داده شده و دوباره با دستور return به حلقه ی do-loop رجوع میشود و این کار مدام تکرار میگردد. شما میتوانید در حلقه از هر دستوری استفاده کنید .

شما همچنین میتوانید با دستور زیر فرمت نمایش تاریخ را معین کنید (در هر کشوری تاریخ به شکل خاص نشان داده میشود ، مثلا در ایران به صورت : yy/mm/dd و در ایالات متحده امریکا به صورت: mm-dd-yy است)

```
CONFIG DATE = DMY/MDY/YMD , Separator = char
```

DMY/MDY/YMD : نشان دهنده مکان نمایش روز ، ماه و سال است ، d نمایشگر روز (Day) ، m نمایشگر ماه (month) و y نمایشگر سال (year) است .

Char : علامت بین روز و ماه و سال میباشد که میتواند نقطه "." ، ممیز "/" یا خط فاصله "-" باشد.

PEYMAN KHODDAMI

جدول زیر اطلاعات بیشتری را در اختیار شما میگذارد:

Country	Format	Statement
American	mm/dd/yy	Config Date = MDY, Separator = /
ANSI	yy.mm.dd	Config Date = YMD, Separator = .
British/French	dd/mm/yy	Config Date = DMY, Separator = /
German	dd.mm.yy	Config Date = DMY, Separator = .
Italian	dd-mm-yy	Config Date = DMY, Separator = -
Japan/Taiwan	yy/mm/dd	Config Date = YMD, Separator = /
USA	mm-dd-yy	Config Date = MDY, Separator = -

مثال

```
$regfile = "M16DEF.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =
Pind.5 , E = Pind.1 , Rs = Pind.0

Config Lcd = 16 * 2

Enable Interrupts

Config Date = Dmy , Separator. =

Config Clock = Soft

Date$ = "10/11/02" : Time$ = "02:20:00"

Do

Locate 1 , 1 : Lcd Date$ : Locate 2 , 1 : Lcd Time$

Loop

End
```

مثال:

```
$regfile = "M16DEF.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =
Pind.5 , E = Pind.1 , Rs = Pind.0

Config Lcd = 16 * 2
```


PEYMAN KHODDAMI

Target = DayOfWeek() : با این دستور ، به ازای هر روزی که از هفته میگذرد یک واحد به متغیر Target افزوده میشود . این متغیر از صفر تا شش تغییر میکند ، شما با استفاده از جدول Lookupstr میتونید زور های هفته را به راحتی جایگزین اعداد کنید.مانند:

```
$regfile = "M16DEF.DAT" : $crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =
Pind.5 , E = Pind.1 , Rs = Pind.0

Config Lcd = 16 * 2 : Cursor Off : Enable Interrupts

Dim Bweekday As Byte , Strweekday As String * 10

Config Date = YMD , Separator.=

Config Clock = Soft

_sec = 59 : _min = 59 : _hour = 23 : _day = 26 : _month = 11 : _year = 2

Do

Locate 1 , 1 : Lcd Date$ : Locate 2 , 1 : Lcd Time$

Bweekday = Dayofweek()

Strweekday = Lookupstr(bweekday , Weekdays (

Locate 1 , 10 : Lcd Bweekday

Locate 2 , 10 : Lcd Strweekday

Loop

End

Weekdays:

Data "Monday" , "Tuesday" , "Wednesday" , "Thursday" , "Friday" , "Saturday"
, "Sunday"
```

دستورات دیگری نیز مانند دستور DayOfWeek وجود دارد که با آنها میتوان روزهای گذشته از ماه ، سال ، ثانیه گذشته از ماه ، سال ، دقیقه گذشته از ماه سال و.. را مشخص کرد . شما میتونید این دستورات را در help بسکام ببابید ، استفاده از آنها مانند دستور DayOfWeek است.

<< روشهای ارتباطی در avr

PEYMAN KHODDAMI

دستورات این پروتکل در بسکام به شرح زیر است:

تعیین نرخ انتقال دیتا:

```
$BAUD=VAR
```

این دستور میزان انتقال دیتا در ثانیه را مشخص میکند و باید در هر دو وسیله ای که به هم متصل میشوند یکی باشد (در غیر این صورت ارتباط کار نمیکند) بهتر است نرخ انتقال دیتا در مناطق دارای نویز کمتر انتخاب شود.

دستور PRINT :

```
PRINT VAR
```

توسط این دستور میتوان داده یا متغیری را به پورت سریال ارسال کرد. VAR یک متغیر از جنس متغیرهای گفته شده میباشد.

دستور PRINTBIN :

```
PRINTBIN VAR
```

توسط این دستور متغیر VAR به باینر تبدیل شده سپس به پورت سریال ارسال میشود.

دستور WAITKEY :

```
VAR=WAITKEY ()
```

این دستور تا زمانی که متغیر توسط دستگاه دیگر به پورت سریال ارسال شود منتظر میماند و پس از دریافت متغیر برنامه از خط بعد ادامه می یابد.

دستور INKEY :

```
VAR=INKEY ()
```

این دستور مقدار اسکی کاراکتر دریافت شده از پورت سریال را برمیگرداند.

دستور INPUTBIN :

```
INPUTBIN VAR
```

این دستور داده باینری را از پورت سریال میگیرد و در متغیر VAR قرار میدهد.

دستور INPUTHEX :

PEYMAN KHODDAMI

INPUTHEX VAR

این دستور داده هگز را از پورت سریال دریافت میکند و در متغیر VAR قرار میدهد. مانند:

در مثال زیر با استفاده از ارتباط سریال ، یک ارتباط دوطرفه بین دو میکرو برقرار کرده ایم ، در زیر برنامه مدار مربوطه و توضیحات برنامه را مشاهده میفرمایید.

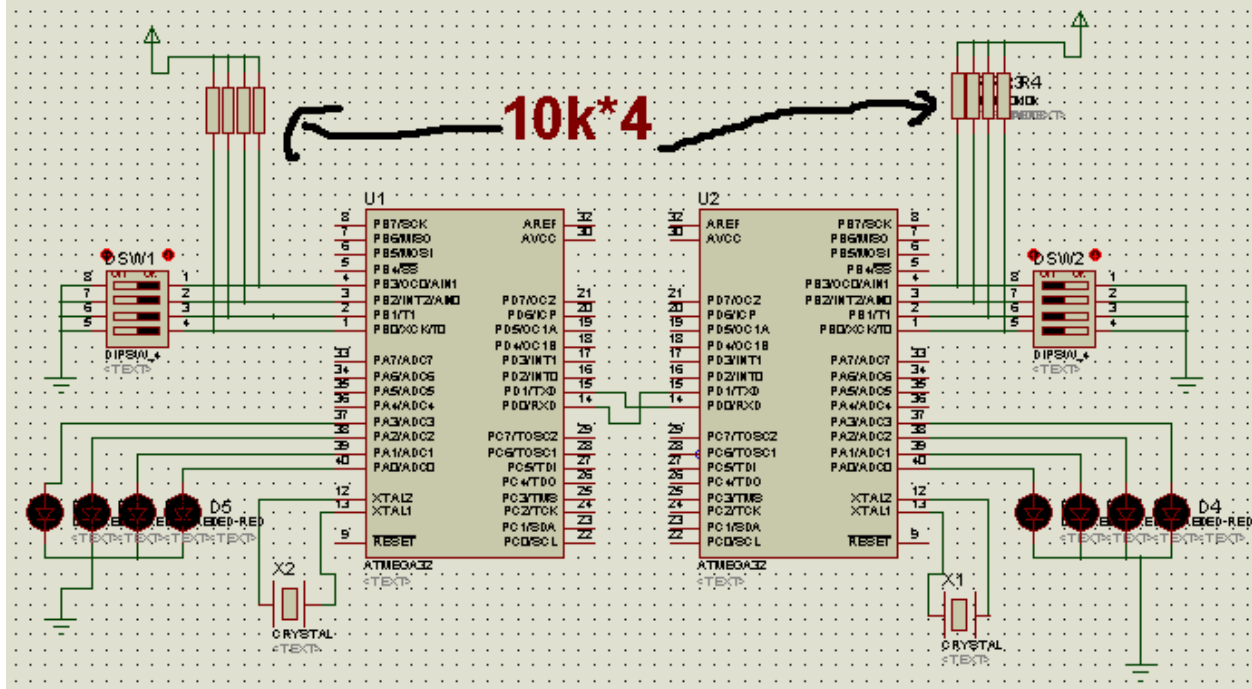
```
$regfile = "m32def.dat" : $crystal = 1000000
$baud = 9600
Config Portb = Input : Config Porta = Output
Dim A As Byte , Q As Byte
W:
Q = Pinb : Printbin Q
A = Inkey() : Porta = A
Goto W
End
```

میکرو 2:

```
$regfile = "m32def.dat" : $crystal = 1000000
$baud = 9600
Config Portb = Input : Config Porta = Output
Dim A As Byte : Dim Q As Byte
W:
Q = Pinb : Printbin Q
A = Inkey() : Porta = A
Goto W
End
```

مدار مربوطه:

PEYMAN KHODDAMI



از انجا که ارتباط دو طرفه میباشد (هر دو میکرو دقیقا مانند هم هستند) برنامه دو میکرو مشابه است ، در خط اول میکرو و کریستال معرفی شده است که میکرو 32 مگا و کریستال مورد استفاده 10 مگا هرتز میباشد .

در خط دوم نرخ انتقال دیتا مشخص گردیده است ، مقدار آن 9600 است . نرخ انتقال دیتا باید در هر دو میکرو یکسان باشد .

در خط سوم پورت b به عنوان ورودی (برای اتصال کلید) و پورت a به عنوان خروجی (برای اتصال led) معرفی شده اند .

در خط چهارم دو متغیر از جنس بایت برای ذخیره مقادیر معرفی شده است .

در خط پنجم شروع یک حلقه میباشد (شما میتواندی از دیگر حلقه ها نیز استفاده کنید).

در خط ششم مقدار موجود بر روی پورت b در متغیر q ریخته میشود و سپس با دستور Printbin Q به پورت سریال فرستاده میشود.

در خط هفتم مقدار گرفته شده از پورت سریال در متغیر a ریخته میشود و بعد متغیر a بر روی پورت a ریخته میشود.

خط هفتم پایان حلقه میباشد ، هنگامی که cpu میکرو به این خط رسید به برجسب w پرش میکند.

خط هشتم پایان برنامه است.

PEYMAN KHODDAMI

در حالتی که هیچ یک از کلید ها یک نشده اند ، مقدار q صفر دسیمال و &b00000000 باینری است ، حال اگر هر یک از کلید ها فشرده شود مقدار q تغییر میکند.

دو پایه txd و rxd میکرو نقش دریافت و ارسال داده را در حالت پیشفرض برعهده دارند ، با دستور زیر شما میتوانید این دو پایه را به پایه های دلخواه تغییر دهید:

```
Open "comx.y:$baud,8,n,1" For Output/input As #q
```

comx.y : نام پورت و پایه ای است که باید به عنوان txd یا rxd جدید عمل کند.

\$baud : نرخ داده عبوری از پایه را نشان میدهد ، این مقدار باید با نرخ انتقال دیتا ی اصلی برابر باشد.

Output/input : پایه میتواند وردی داده (rxd) یا خروجی داده (txd) باشد.

Q : شماره کانال را مشخص میکند. مانند:

```
Open "comd.1:19200,8,n,1" For Output As #1
```

```
Open "comd.0:19200,8,n,1" For Input As #2
```

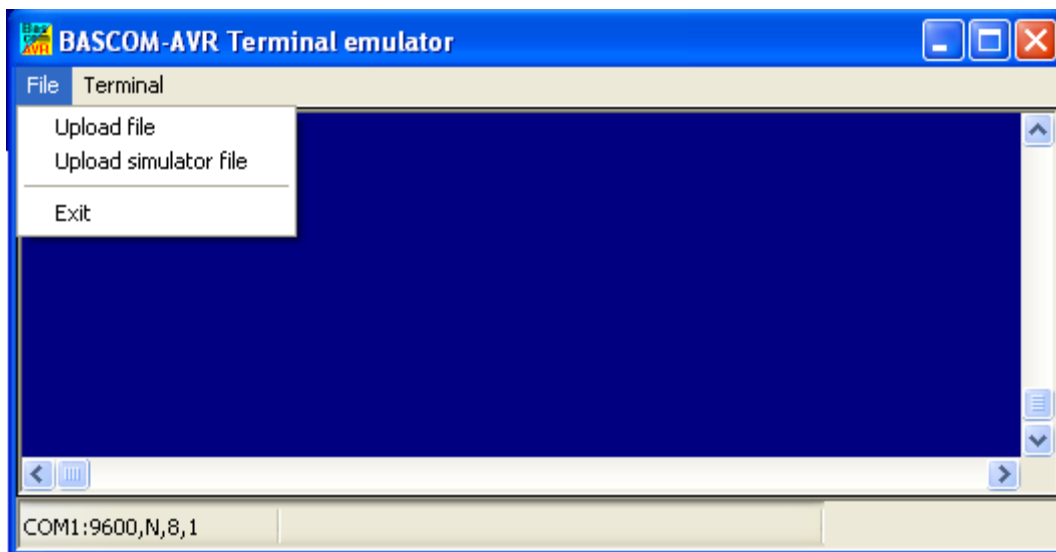
در مورد بالا portd.1 به عنوان txd و portd.0 به عنوان rxd در نظر گرفته شده است ، همچنین نرخ انتقال داده برابر با 19200 است.

بررسی محیط TERMINAL EMULATOR:

از این محیط برای نمایش داده ارسالی و دریافتی در ارتباط سریال RS-232 بین میکرو و کامپیوتر استفاده میشود.

اطلاعاتی که در این محیط تایپ می شود به میکرو ارسال و اطلاعاتی که از پورت سریال کامپیوتر (COM) دریافت می شود در این پنجره نمایش داده می شود . هنگامیکه در برنامه از SERIAL IN و یا SERIAL OUT استفاده می شود , پس از PROGRAM کردن برنامه درون میکرو و اتصال آن به پورت سریال PC , می توان داده های ارسالی توسط UART میکرو را دریافت کرده و نمایش داد و از صحت آنها اطلاع یافت . همچنین اگر از دستوری مانند INKEY در برنامه استفاده شود , میتوان داده خود را از طریق پنجره TERMINAL EMULATOR به میکرو ارسال نمود .

PEYMAN KHODDAMI



FILE UPLOAD : برنامه جاری در فرمت HEX را UPLOAD میکند

FILE ESCAPE : صرفنظر کردن از UPLOAD کردن فایل .

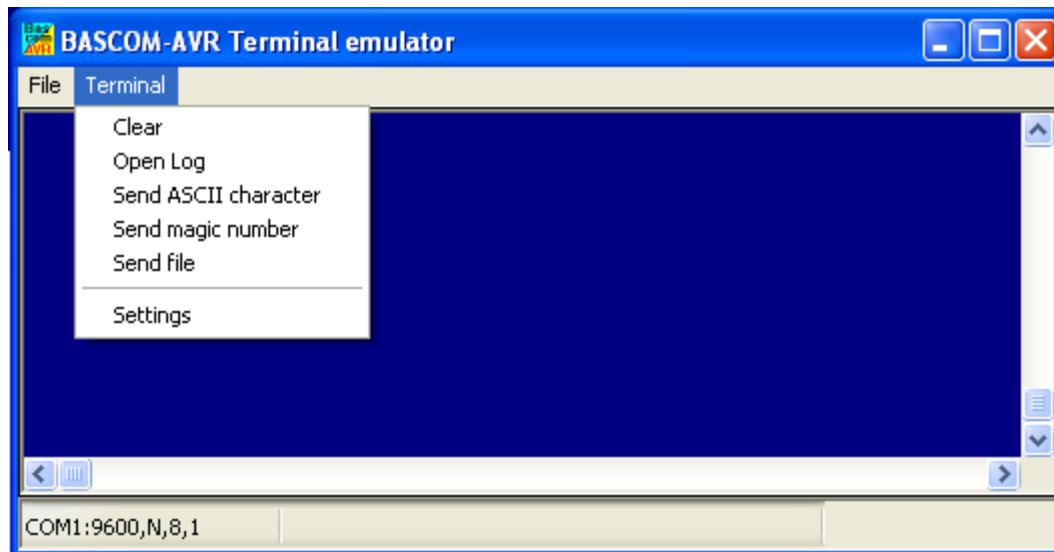
FILE EXIT : خروج از برنامه EMULATOR .

TERMINAL CLEAR : پنجره ترمینال را پاک می کند .

SETTING : تنظیمات پورت COM و دیگر OPTION ها توسط این منو صورت می گیرد .

TERMINAL OPEN LOG : فایل LOG را باز یا بسته می کند . هنگامیکه فایل LOG وجود نداشته باشد درخواست نامی

برای فایل گزارش می کند . تمام اطلاعاتی که در پنجره TERMINAL پرینت می شود داخل فایل LOG ثبت می شود .



PEYMAN KHODDAMI

2-ارتباط سریال SPI :

ویژگی های این ارتباط را در زیر مشاهده میکنید:

- 1- ارتباط سریال سنکرون با سرعت بالا
- 2- از این ارتباط میتوان برای اتصال میکرو های avr به یکدیگر یا اتصال میکرو به هر وسیله ای که این ارتباط را پشتیبانی میکند استفاده کرد
- 3- ارسال و دریافت داده هم زمان
- 4- استفاده از 4 سیم برای انتقال اطلاعات
- 5- قابلیت تنظیم سرعت انتقال دیتا
- 6- دارای منبع وقفه پایان ارسال
- 7- حداکثر طول کابل ارتباطی بدون سیم شیلد در یک منطقه با نویز متوسط پنجاه سانتی متر است
- 8- ارتباط به صورت های مستر یا اسلیو
- 9- نیاز به سیم گراند برای ارتباط

در این ارتباط از چهار پایه mosi و miso و sck و ss استفاده میشود که در میکرو مگا 16 به ترتیب پایه 5 تا 8 میباشد (از portb.4 تا portb.7).

در زیر نحوه پیکر بندی spi در بسکام آمده است:

دستورپیکره بندی SPI در محیط BASCOM:

```
CONFIG SPI = HARD, INTERRUPT=ON/OFF, DATA ORDER = LSB/MSB, MASTER =  
YES/NO, POLARITY=HIGH/LOW, PHASE=0/1, CLOCK RATE=4/16/64/128, NOSS=0/1
```

INTERRUPT=ON/OFF : در صورت استفاده از وقفه در ارتباط سریال از گزینه ON استفاده میشود ، در این حالت تنها زمانی که داده ای میخواهد منتقل شود میکرو کار میکند .

DATA ORDER = LSB/MSB: در صورت انتخاب LSB ، ابتدا LSB (بیت کم ارزش) و سپس MSB(بیت پر ارزش) داده ارسال می شود و بالعکس.

MASTER = YES/NO: این گزینه مشخص میکن میکرو master (فرمانده) است یا slave (فرمانبردار) گه گزینه yes مشخص کننده میکرو مستر و گزینه no مشخص کننده میکرو اسلیو است.

POLARITY=HIGH/LOW : این گزینه و وضعیت پایه کلاک را در زمان بیکاری میکرو مشخص میکن که میتواند صفر (low) یا یک (high) باشد

PEYMAN KHODDAMI

CLOCK RATE=4/16/64/128: مشخص کننده فرکانس کلاک SPI است .

NOSS=0/1: زمانی که در حالت MASTER نمی خواهید سیگنال /SS ایجاد شود، یک انتخاب می شود و در این حالت کاربر به صورت نرم افزاری باید پایه SLAVE مورد نظر را پایین نگه دارد.

در حالت بالا از پایه های پیشفرض برای انتقال دیتا استفاده میشود ، در صورتی که میخواهید آنها را به پایه های دیگر تغییر دهید باید از دستور زیر استفاده کنید:

```
CONFIG SPI=SOFT, DIN=PIN, DOUT = PIN , SS = PIN|NONE, CLOCK = PIN
```

DIN=PIN: نشانگر پایه MISO است و بین نام پایه دل خواه میباشد.

DOUT = PIN : نشانگر پایه MOSI است و بین نام پایه دل خواه میباشد.

SS = PIN|NONE: نشانگر پایه SS است و بین نام پایه دلخواه میباشد. (در صورتی از گزینه NONE استفاده کنید پایه تغییر نمیکند

CLOCK = PIN : نشانگر پایه SCK است و بین نام پایه دلخواه میباشد.مانند:

```
CONFIG SPI=SOFT, DIN=PIND.0, DOUT = PIND.1 , SS = PIND.2, CLOCK = PIND.3
```

دیگر دستورات مربوط به SPI :

دستور SPIINIT :

```
SPIINIT
```

توسط این دستور پایه های که برای SPI (mosi و miso و sck و ss (در میکرو مگا 16 به ترتیب پایه 5 تا 8 میباشد (از portb.4 تا portb.7)) استفاده میشوند ، برای این مد فعال میگردند و دیگر نمیتوان از آنها به عنوان ورودی یا خروجی استفاده کرد.

دستور SPIIN :

```
SPIIN VAR, BAYT
```

توسط این دستور به تعداد BAYT از درگاه SPI اطلاعات دریافت میشود و در متغیر VAR قرار میگیرد ، در صورتی که متغیر شما از جنس WORD یا دیگر متغیر ها است ، شما باید تعداد بایت متغیر را به جای BAYT بنویسید مانند:

```
'MASTER
```

PEYMAN KHODDAMI

```
$regfile = "m16def.dat"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =
Pind.5 , Rs = Pind.0 , E = Pind.1

Config Lcd = 16 * 2

Dim A As Byte

Config Spi = Hard , Interrupt = On , Data Order = Lsb , Master = Yes ,
Polarity = High , Phase = 0 , Clockrate = 128

Spiinit

Do

Spiin A , 1

Locate 1 , 1

Lcd A

Loop

End
```

دستور SPIOUT :

SPIOUT VAR , BAYT

توسط این دستور به تعداد BAYT، داده VAR به درگاه SPI اطلاعات ارسال میشود، در صورتی که متغیر شما از جنس WORD یا دیگر متغیرها است، شما باید تعداد بایت متغیر را به جای BAYT بنویسید مانند:

```
'SLAVE

$regfile = "m16def.dat"

$crystal = 8000000

Dim A As Word

Config Spi = Hard , Interrupt = On , Data Order = Lsb , Master = No ,
Polarity = High , Phase = 0 , Clockrate = 128
```

PEYMAN KHODDAMI

Spiinit

Do

Incr A

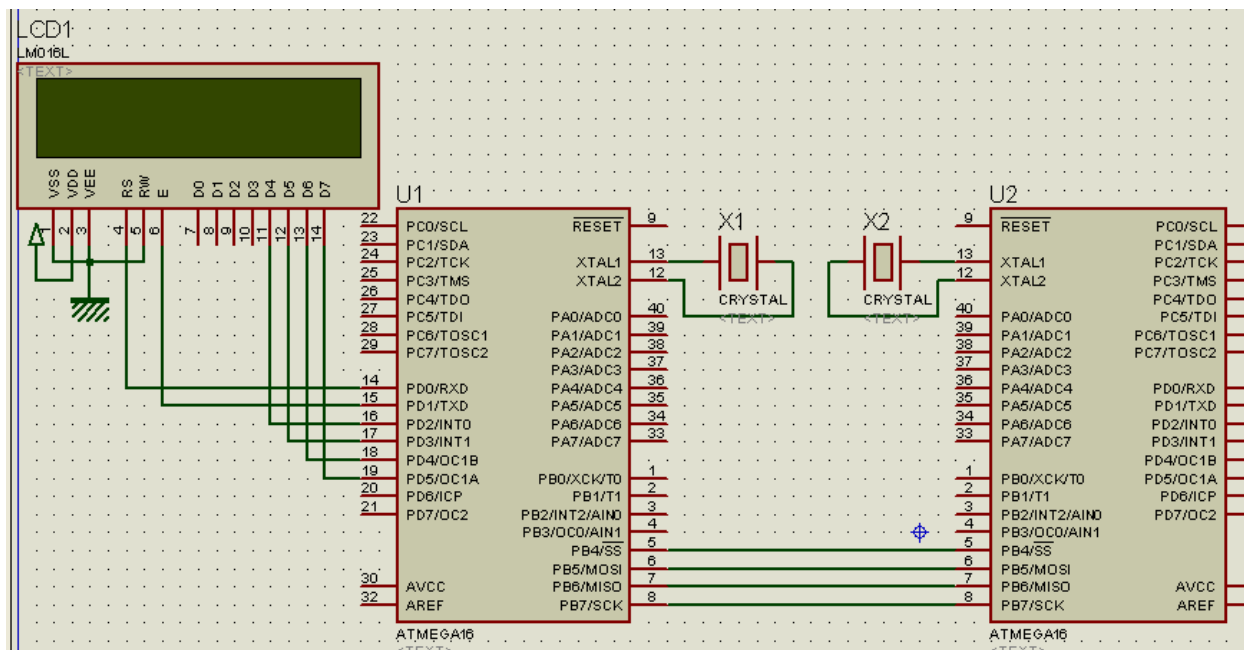
Waitms 300

Spiout A , 1

Loop

End

هر دو مثال بالا مربوط به ارتباط SPI بین دو میکرو مگا 16 است که مدار ان را در زیر مشاهده میفرمایید:



دستور SPIMOVE :

VAR=SPIMOVE (BAYT)

PEYMAN KHODDAMI

از این دستور در زمان ارتباط دوطرفه استفاده میشود ، توسط این دستور متغیر BAYT به باس SPI ارسال شده و همزمان داده دریافت شده از باس در متغیر VAR قرار میگیرد.مثال:

برنامه SLAVE :

```
$regfile = "m16def.dat":$ crystal = 8000000
```

```
Dim A As Byte,B As Byte
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.5 , Rs = Pind.0 , E = Pind.1
```

```
Config Lcd = 16 * 2
```

```
Config Spi = Hard , Interrupt = On , Data Order = Lsb , Master = No ,  
Polarity = High , Phase = 0 , Clockrate = 128
```

```
Spiinit
```

```
Do
```

```
B = Spimove(a(
```

```
A = A + 2
```

```
Locate 1 , 1 :Lcd "Spiout:"; A
```

```
Locate 2 , 1: Lcd "SPIIN:" ; B
```

```
Waitms 300
```

```
Loop
```

```
End
```

برنامه MASTER :

```
$regfile = "m16def.dat":$ crystal = 8000000
```

```
Dim A As Byte,B As Byte
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.5 , Rs = Pind.0 , E = Pind.1
```

PEYMAN KHODDAMI

```
Config Lcd = 16 * 2
```

```
Config Spi = Hard , Interrupt = On , Data Order = Lsb , Master = Yes ,  
Polarity = High , Phase = 0 , Clockrate = 128
```

```
Spiinit
```

```
Do
```

```
B = Spimove(a(
```

```
Incr A
```

```
Locate 1 , 1: Lcd "Spiout:"; A
```

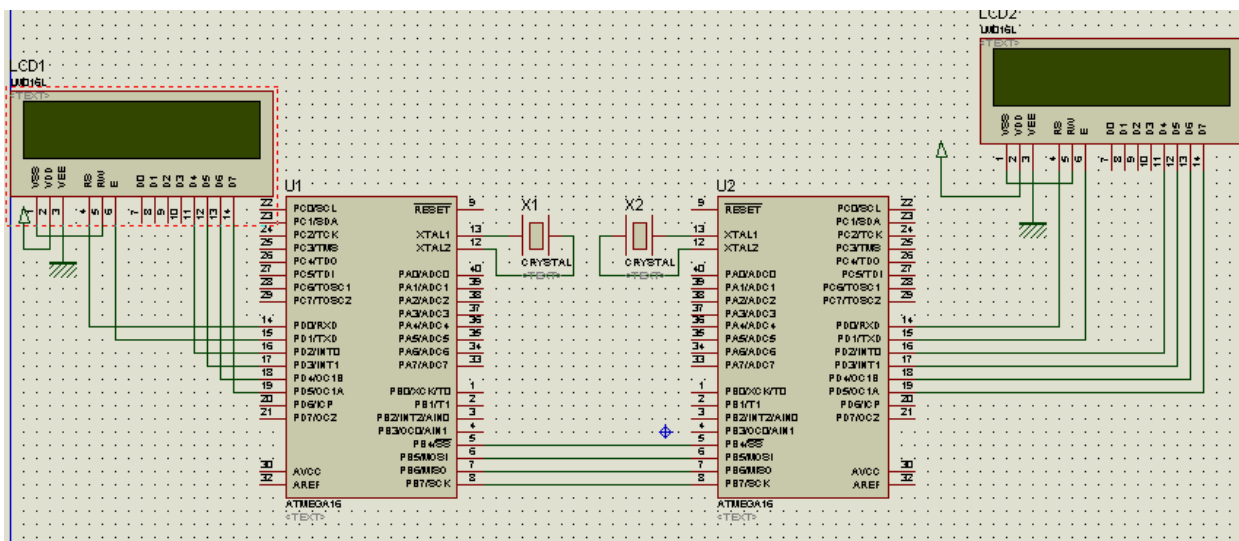
```
Locate 2 , 1 :Lcd "SPIIN:" ; B
```

```
Waitms 300
```

```
Loop
```

```
End
```

مدار مورد استفاده:



مثال:

در این مثال پایه های مربوط به ارتباط SPI تغییر کرده اند ، شما میتونید انها را به حالت پیش فرض برگردانید:

برنامه SLAVE :

```
$regfile = "m16def.dat":$crystal = 8000000
```

PEYMAN KHODDAMI

```
Dim A As Byte,B As Byte
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.5 , Rs = Pind.0 , E = Pind.1
```

```
Config Lcd = 16 * 2
```

```
Config Spi = Soft , Din = Pina.0 , Dout = Pina.1 , Ss = Pina.2 , Clock =  
Pina.3
```

```
Spiinit
```

```
Do
```

```
B = Spimove(a(
```

```
A = A + 2
```

```
Locate 1 , 1: Lcd "Spiout:"; A
```

```
Locate 2 , 1: Lcd "SPIIN:" ; B
```

```
Waitms 300
```

```
Loop
```

```
End
```

: MASTER برنامه

```
$regfile = "m16def.dat":$crystal = 8000000
```

```
Dim A As Byte,B As Byte
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 =  
Pind.5 , Rs = Pind.0 , E = Pind.1
```

```
Config Lcd = 16 * 2
```

```
Config Spi = Soft , Din = Pina.0 , Dout = Pina.1 , Ss = Pina.2 , Clock =  
Pina.3
```

```
Spiinit
```

```
Do
```

```
B = Spimove(a(
```

PEYMAN KHODDAMI

Incr A

Locate 1 , 1: Lcd "Spiout:"; A

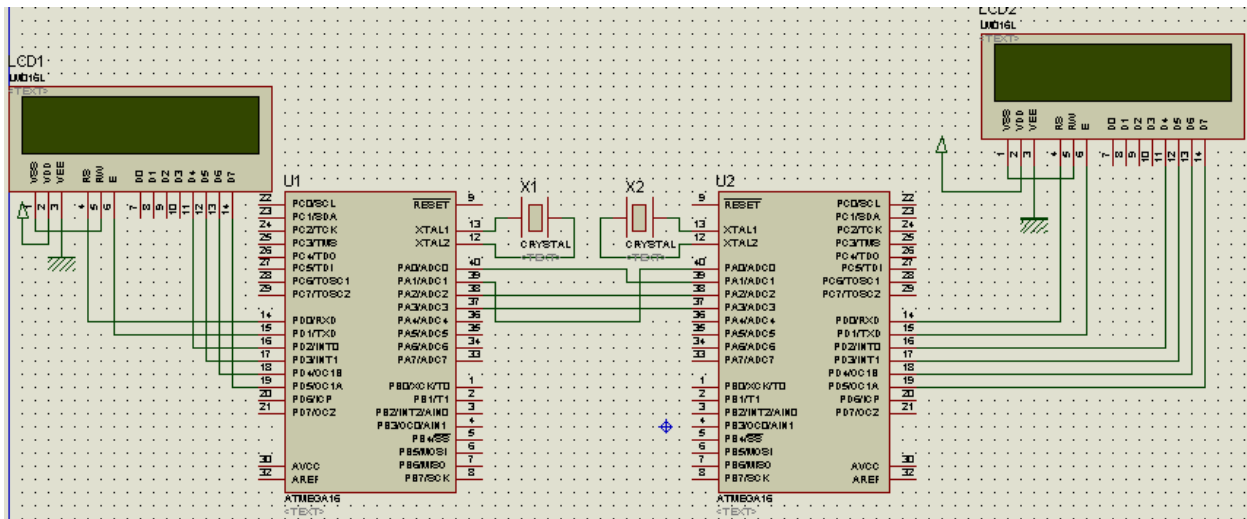
Locate 2 , 1: Lcd "SPIIN:" ; B

Waitms 300

Loop

End

مدار مورد استفاده:



3-ارتباط سریال I2C یا 2-wire

ویژگی های این ارتباط به شرح زیر است:

- 1- در این ارتباط از دوسیم همرا با گراند و VCC ، که درمجموع چهار سیم میشود ، برای انتقال دیتا استفاده میشود
- 2- بالا ترین فرکانس کلاک 400 کیلو هرتز است
- 3- در این ارتباط میتوان تعداد نامحدود وسیله جانبی با ادرس سخت افزاری مختلف را به هم متصل کرد
- 4- حداکثر طول کابل ارتباطی باسیم شیلد 80 سانتی متر است
- 5- کلاک ارتباط I2C به شدت به کلاک سیستم (فرکانس کریستال) وابسته است

برای ارتباط I2C از دو پایه SCL و SDA (PORTC.0 و PORTC.1 میگو 16 مگا) استفاده میشود که پایه SDA پایه داده و پایه SCL پایه کلاک میباشد.(دو پایه مذکور پایه های پیش فرض میباشند ، شما میتونید با دستوراتی که گفته میشود آنها را به پایهای دلخواه خود تغییر دهید)

PEYMAN KHODDAMI

دستورات مربوط به راه اندازی I2C در محیط بسکام:

تعیین کلاک I2C :

```
Config I2cdelay = X
```

X میتواند از 1 تا 255 باشد ، رابطه ای بین کلاک و عدد وجود ندارد ، مثلا برای عدد 10 کلاک 100 کیلو و برای عدد 5 کلاک 200 کیلو و برای عدد 1 کلاک 400 کیلو هرتز است (کلاک I2C به فرکانس کریستال وابسته است ، در این ارتباط باید کریستال نوشته شده در برنامه با کریستال استفاده شده یکی باشد ، همچنین کلاک هر دو دستگاه باید مساوی باشد)

تعیین پایه های داده و کلاک I2C :

با دستور زیر پایه SCL (پایه کلاک) تعیین میشود :

```
CONFIG SCL = pin
```

Pin نام یکی از پایه های دلخواه میکرو میباشد .

با دستور زیر پایه SDA (پایه داده) تعیین میشود :

```
CONFIG SDA = pin
```

Pin نام یکی از پایه های دلخواه میکرو میباشد

بعد از آنکه I2C پیکر بندی شد با استفاده از دستور زیر میتوان ارتباط را آغاز کرد

```
I2CSTART
```

با این دستور ارسال و دریافت داده شروع میشود شما همچنین میتوانید با دستور زیر به ارسال و دریافت داده خاتمه دهید:

```
I2CSTOP
```

با استفاده از دستور زیر میتوان داده ای را به باس I2C فرستاد:

```
I2CSEND slave, var
```

```
I2CSEND slave, var , bytes
```

Slave: ادرس گیرنده اطلاعات است که میتواند به فرم یک عدد ثابت یا متغییر باشد

PEYMAN KHODDAMI

Var : عدد ثابت یا متغیری است که میخواهیم آن را ارسال کنیم

bytes: با این گزینه شما میتوانید تعداد بایت دلخواه را به باس ارسال کنید (این گزینه اختیاری است)

فرم خلاصه شده این دستور به شکل زیر است:

```
I2CWBYTE val
```

Val : عدد ثابت یا متغیری است که کد دریافت شده در آن قرار میگیرد. مثال:

برنامه فرستنده:

```
$regfile = "m32def.dat"
```

```
$crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinb.2 , Db5 = Pinb.3 , Db6 = Pinb.4 , Db7 =  
Pinb.5 , Rs = Pinb.0 , E = Pinb.1
```

```
Config Kbd = Porta
```

```
Config I2cdelay = 5
```

```
Config Sda = Portc.1
```

```
Config Scl = Portc.0
```

```
Dim A As Byte
```

```
I2cstart
```

```
W:
```

```
A = Getkbd()
```

```
If A > 15 Then
```

```
Goto W
```

```
End If
```

```
I2csend &H40 , A
```

```
Locate 2 , 1
```

PEYMAN KHODDAMI

Lcd A

Goto W

End

در مثال با کلاک I2C ، 200 کیلو هرتز انتخاب شده است همچنین پایه SCL (پایه کلاک) به پورت c.0 (پایه 22 مگا) و پایه SDA (پایه داده) به پین c.1 (پایه 23 مگا) متصل شده است. دستور I2cstart پروتکل i2c شروع به کار کرده و عدد گرفته شده از کیبورد را بخ باس i2c میفرستد ، برای گیرنده ادرس &h40 در نظر گرفته شده است ، برای درک بیشتر موضوع متغیر ارسالی بر روی یک lcd به نمایش در میاید.

با استفاده از دستور زیر میتوان داده ای را از باس I2C دریافت کرد:

```
I2CRECEIVE slave, var
```

```
I2CRECEIVE slave, var , b2W, b2R
```

Slave: ادرس فرستنده اطلاعات است که میتواند به فرم یک عدد ثابت یا متغیر باشد.

Var : عدد ثابت یا متغیری است که کد دریافت شده در آن قرار میگیرد

bytes: با این گزینه شما میتوانید تعداد بایت دلخواه را از باس دریافت کنید (این گزینه اختیاری است). (توجه داشته باشید که تعداد بایت دریافتی و ارسالی باید با هم برابر باشند در غیر اینصورت اطلاعات دریافتی ناقص خواهد بود).

فرم خلاصه شده این دستور به شکل زیر است:

```
I2CRBYTE var, ack/nack
```

Var : عدد ثابت یا متغیری است که کد دریافت شده در آن قرار میگیرد

ack/nack: زمانی که بخواهیم بیشتر از یک بایت را از باس بخوانیم باید از ack استفاده کنیم و زمانی که می خواهیم آخرین بایت را از باس بخوانیم از nack استفاده میکنیم.مثال:

این برنامه مربوط به گیرنده مداری است که برنامه ان را در بالا مشاهده فرمودید:

```
$regfile = "m32def.dat" : $crystal = 1000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pina.0 , Db5 = Pina.1 , Db6 = Pina.2 , Db7 =  
Pina.3 , Rs = Pina.4 , E = Pina.5
```

PEYMAN KHODDAMI

```
Config I2cdelay = 5
```

```
Config Sda = Portc.1
```

```
Config Scl = Portc.0
```

```
Dim A As Byte
```

```
I2cstart
```

```
W:
```

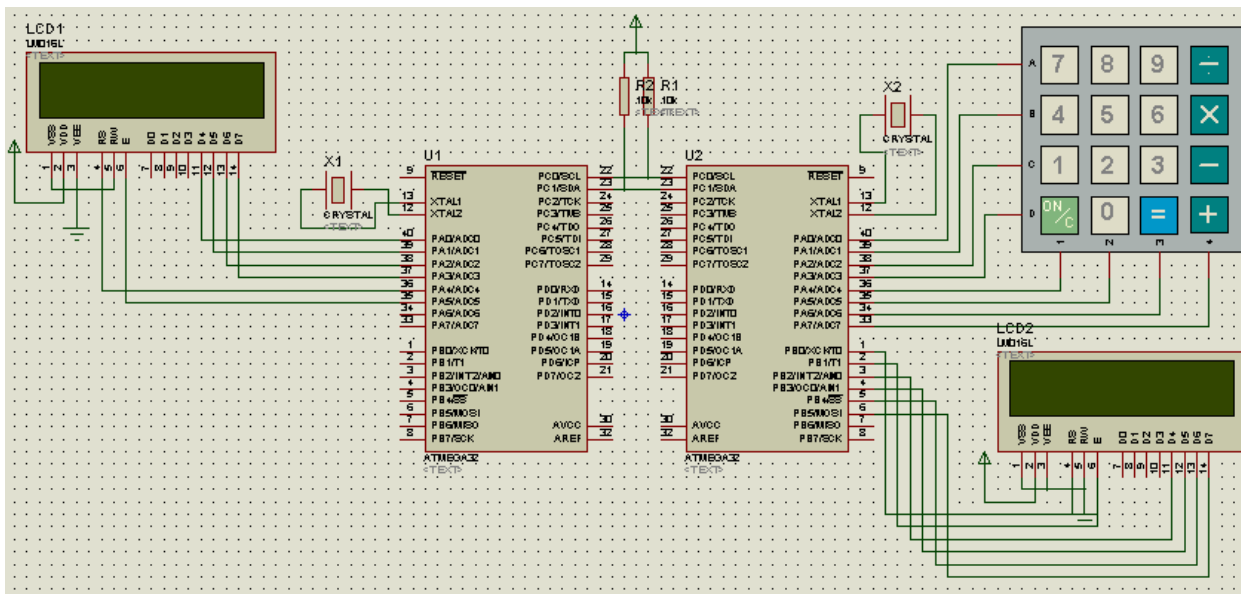
```
I2creceive &H40 , A
```

```
If A < 16 Then : Locate 1 , 1 : Lcd A : End If
```

```
Goto W
```

```
End
```

در مثال بالا کلاک I2C ، 200 کیلو هرتز (Config I2cdelay = 5) انتخاب شده است همچنین پایه SCL (پایه کلاک) به



پورت c.0 (پایه 22 مگا) و پایه SDA (پایه داده) به پین c.1 (پایه 23 مگا) متصل شده است. دستور I2cstart و I2cstart شروع به کار کرده و عدد گرفته شده از باس را بر روی lcd نمایش میدهد ، برای فرستنده ادرس &h40 در نظر گرفته شده است .مثال:

در قسمت های قبل گفته شده بود که ایسی به شماره ds1307 وجود دارد که میتواند زمان را دقیق بشمارد .ویژگی های این ایسی به شرح زیر است:

PEYMAN KHODDAMI

- 1- ای سی DS1307 یک RTC می باشد rtc مخفف (Real Time Counter) (شمارش گر زمان واقعی) است.
- 2- این ای سی با یک باتری بک آپ 3 ولتی می تواند تا 10 سال زمان را در خود بشمارد. (در صورتی که باتری دوام آورد ، منظور کم مصرفی ایسی است).
- 3- نمایش ساعت (شامل ثانیه - دقیقه وساعت) به صورت 24 ساعت.
- 4- نمایش تاریخ(شامل روز-ماه - سال) به صورت میلادی
- 5- شمارش روز هفته (شنبه - 1 شنبه تا جمعه)
- 6- شمارش روز های طی شده از اول سال (به صورت میلادی)
- 7- دارای پایه جدا برای اتصال باتری بک آپ
- 8- دارای دو پایه برای اتصال نوسان ساز مستقل
- 9- عملکرد کاملاً ثابت در تمامی محیط ها
- 10- قیمت و پشتیبانی خوب
- 11-.....

ما در اینجا با استفاده از این ایسی زمان را به صورت 24 ساعته نمایش میدهیم (بهتر است ابتدا دیتا شیت ایسی را مطالعه کنید تا با نحوه خوانده و نوشتن در ان آشنا شوید ،دیتا شیت را میتوانید از سایت www.datasheet4u.com تهیه کنید).

```
$regfile = "m16def.dat":$crystal = 1000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Rs = Pind.0 , E = Pind.1 , Db4 = Pind.2 , Db5 =
Pind.3 , Db6 = Pind.4 , Db7 = Pind.5

Config Clock = User:Config Sda = Portc.1:Config Scl = Portc.0

Const Ds1307w = &HD0:Const Ds1307r = &HD1:Cursor Off

Do

Locate 1 , 1:Lcd Time$

Loop

End

Getdatetime:

I2cstart
```

PEYMAN KHODDAMI

```
I2cwrite Ds1307w
```

```
I2cwrite 0
```

```
I2cstart
```

```
I2cwrite Ds1307r
```

```
I2cwrite _sec , Ack
```

```
I2cwrite _min , Ack
```

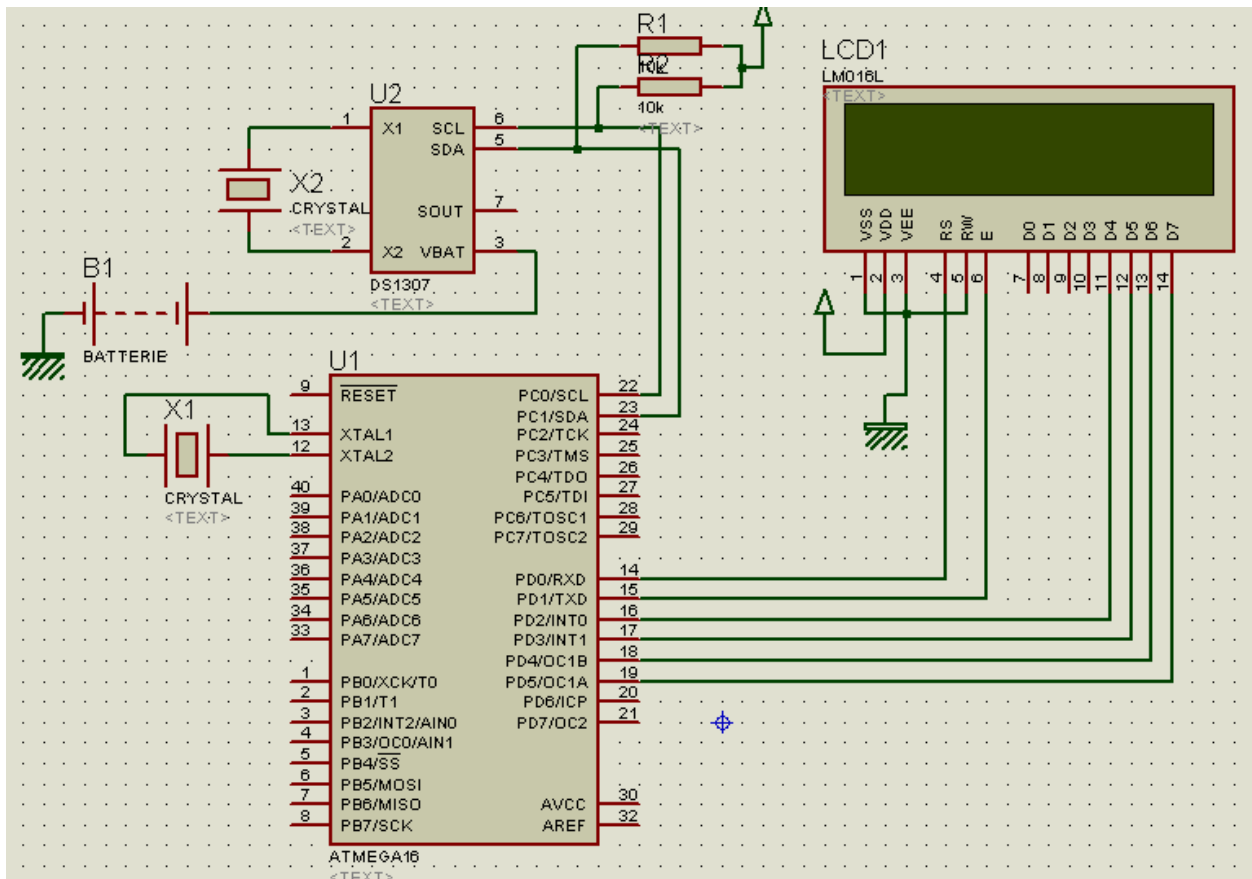
```
I2cwrite _hour , Ack
```

```
I2cstop
```

```
_sec = Makedec(_sec) : _min = Makedec(_min) : _hour = Makedec(_hour)
```

```
Return
```

مدار برنامه بالا را در زیر مشاهده میکنید:



PEYMAN KHODDAMI

ارتباط سریال WIRE 1 :

ویژگی های این پروتکل به شرح زیر است:

- 1- در این ارتباط از یک همرا با گراند و VCC ، که در مجموع سه سیم میشود ، برای انتقال دیتا استفاده میشود.
 - 2- بالا ترین فرکانس کلاک 2 کیلو هرتز است.
 - 3- در این ارتباط میتوان تعداد دو وسیله ی اصلی و تعداد نامحدود وسیله جانبی را به هم متصل کرد.
 - 4- کلیه خطوط باید با مقاومت 4.7 کیلو به VCC متصل شوند.
- با دستور زیر باس WIRE 1 مشخص میشود:

```
CONFIG 1WIRE = pin
```

Pin : نام پایه ی دلخواه میکرو است که به عنوان وردی و خروجی داده WIRE 1 استفاده میشود.

(این پایه باید با پایه ای که در قسمت compiler setting بسکام مشخص شده یکی باشد)(پایه ای که در بالا مشخص میشود ، باس اصلی میباشد که به دستگاه دیگر متصل است)

با دستور زیر میتوان داده را از باس WIRE 1 خواند:

```
var2 = 1WREAD( [ bytes] )
```

```
var2 = 1WREAD( bytes , port , pin)
```

دستور اول داده را از باس اصلی و دستور دوم داده را از دیگر دستگاه های جانبی میخواند

var2 : یک متغیر است که داده خوانده شده از باس در آن ریخته میشود ، شما همچنین میتوانید معین کنید چند بایت از باس خوانده شود ([bytes]).

Port : نام پورتهی است که دستگاه جانبی به آن متصل است .

Pin : نام پایه ی پورتهی است که دستگاه به آن متصل شده است مانند :

```
A = 1wread(8 , Pinb , 2)
```

با دستور زیر میتوان داده را در باس WIRE 1 نوشت:

```
1WWRITE var1, bytes
```

```
1WWRITE var1 , bytes , port , pin
```

دستور اول داده را در باس اصلی و دستور دوم داده را در باس دیگر دستگاه های جانبی میریزد.

PEYMAN KHODDAMI

Var1 : متغیر یا عدد ثابتی است که در باس نوشته میشود ، شما همچنین میتوانید معین کنید چند بایت در باس نوشته شود ([bytes]). (تعداد بایت خوانده شده و نوشته در گذر گاه باید با هم برابر باشد ، در غیر اینصورت خطا بوجود میاید)

Port : نام پورتهی است که دستگاه جانبی به آن متصل است .

Pin : نام پایه ی پورتهی است که دستگاه به آن متصل شده است مانند :

```
lwwrite b , 8,pinb,2
```

با دستور زیر باس 1 WIRE ریست میشود (داده های موجود در آن پاک میشود):

```
1WRESET
```

```
1WRESET , PORT , PIN
```

دستور اول باس اصلی و دستور دوم دیگر باس ها را ریست میکند

Port : نام پورتهی است که دستگاه جانبی به آن متصل است .

Pin : نام پایه ی پورتهی است که دستگاه به آن متصل شده است و باید ریست شود مانند :

```
lwreset Pinb , 2
```

با دستور زیر شماره دستگاه متصل شده به باس 1 WIRE خوانده میشود :

```
var2 = 1WIRECOUNT()
```

```
var2 = 1WIRECOUNT( port , pin)
```

var2 : یک متغیر از جنس word یا integer است که داده خوانده شده از باس در آن ریخته میشود .

Port : نام پورتهی است که دستگاه جانبی به آن متصل است .

Pin : نام پایه ی پورتهی است که دستگاه به آن متصل شده است مانند :

```
W = 1wirecount(pinb , 2)
```

با دستور زیر میتوان داده را از دستگا های که به صورت سریال به یک باس متصل شده اند را خواند:

```
var2 = 1WSEARCHFIRST()
```

```
var2 = 1WSEARCHFIRST( port , pin)
```


PEYMAN KHODDAMI

```
var2 = 1WSEARCHNEXT()
```

```
var2 = 1WSEARCHNEXT( port , pin)
```

var2 : یک متغیر از جنس long است که داده خوانده شده از باس در آن ریخته میشود .

Port : نام پورتهی است که دستگاه جانبی به آن متصل است .

Pin : نام پایه ی پورتهی است که دستگاه به آن متصل شده است مثال:

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pind.2 , Db5 = Pind.3 , Db6 = Pind.4 , Db7 = Pind.5 , Rs =  
Pind.0 , E = Pind.1
```

```
Config 1wire = Portb.0 'use this pin
```

```
Dim A As Byte , C As Byte
```

```
Wait 1
```

```
1wreset
```

```
1wwrite &H33
```

```
Do
```

```
A = 1wread(8 , Pinb , 0)
```

```
Locate 1 , 1 : Lcd Hex(a)
```

```
Wait 1 : Incr C
```

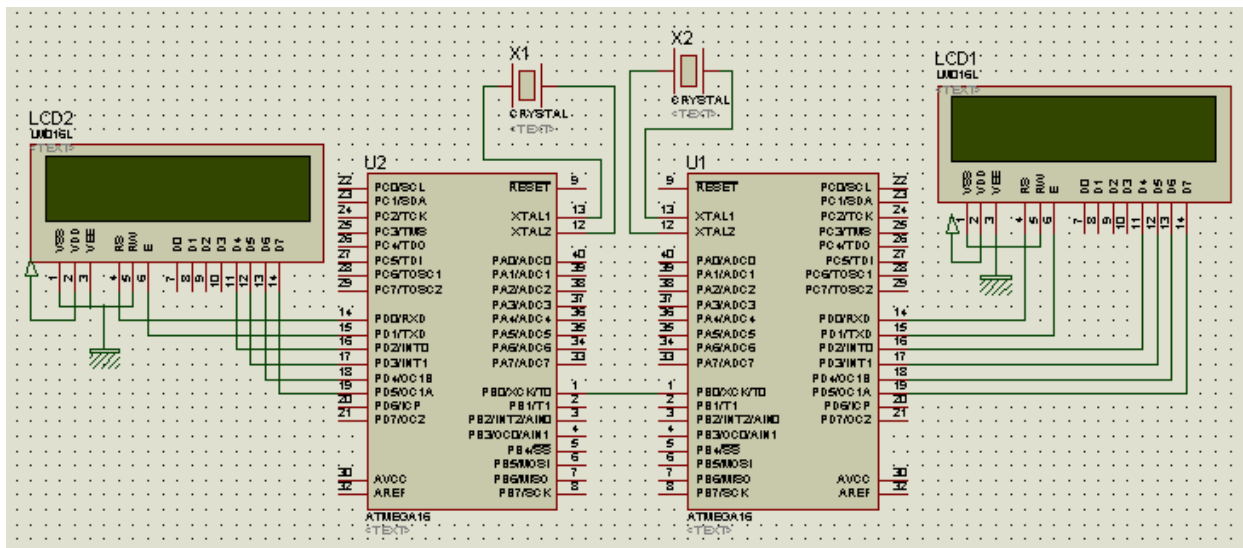
```
1wwrite C , 8 , Pinb , 0
```

```
Locate 2 , 1 : Lcd C
```

```
Loop
```

```
End
```

PEYMAN KHODDAMI



کار با حافظه داخلی میکرو (eprom):

تعداد زیادی از میکرو های avr دارای حافظه داخلی میباشند ، حجم این حافظه بسته به نوع میکرو متفاوت است ، اطلاعات این حافظه بر اثر قطع برق پاک نمیشود و میتواند تا سالهای زیادی محفوظ بماند .

Eeprom داخلی میکرو همیشه آماده به کار است و کافی است شما در آن بنویسید یا از آن بخوانید ، نوشتن در eeprom با دستور زیر انجام میشود :

```
WRITEEEPROM var , address
```

Var: متغیر یا عدد پابندی است که قصد ذخیره آن را دارید.

Address: مکانی از حافظه میباشد که متغیر در آن ذخیره میشود .(در صورتی که ادرسی درج نشود متغیر در اولین مکان خالی ذخیره میشود و پیدا کردن آن با خداست)

ذخیره متغیر اتوماتیک است و آخرین ادرس بستگی به نوع میکرو دارد (به دیتا شیت میکرو مراجعه کنید).

شما همچنین با دستور زیر میتوانید اطلاعات داخل حافظه را بخوانید:

```
READEEPROM var , address
```

Var : یک متغیر متناسب با مقدار اطلاعات میباشد ، که اطلاعات خوانده شده از ادرس درج شده ،در آن ریخته میشود .

Address: ادرسی است که باید اطلاعات از آن خوانده شود.مانند:

```
$regfile = "m16def.dat"
```

PEYMAN KHODDAMI

```
$crystal = 8000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Dim A As Byte , C As Byte

C = 12

Writeeprom C , 1

Wait 1

Readeeprom A , 1

Locate 1 , 1 : Lcd A

End
```

در این مثال مقدار 12 در متغیر c ریخته شده است ، من این متغیر را در ادرس 1 حافظه داخلی ذخیره کردم ، و بعد از گذشت 1 ثانیه خانه 1 حافظه را خواندم و حاصل را در متغیر a ریختم و آن را روی lcd نمایش دادم . همانگونه که مشاهده خواهید کرد عدد خوانده شده برابر 12 میشود . مثال دیگر:

```
$regfile = "m16def.dat"

$crystal = 8000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Portd.0 , Db5 = Portd.1 , Db6 = Portd.2 , Db7 =
Portd.3 , E = Portd.4 , Rs = Portd.5

Config Adc = Single , Prescaler = Auto

Dim A As Word , B As Byte , C As Byte

Start Adc

Do

A = Getadc(0) : A = A / 2

Locate 1 , 1 : Lcd A
```

PEYMAN KHODDAMI

```
Wait 1 : Incr B  
If B > 180 Then  
Writeeprom A , C  
C = C + 1 : B = 0  
End If  
If C > 3 Then  
Readeeprom A , 0 : Locate 1 , 1 : Lcd A  
Readeeprom A , 1 : Locate 1 , 9 : Lcd A  
Readeeprom A , 2 : Locate 2 , 1 : Lcd A  
Readeeprom A , 3 : Locate 2 , 9 : Lcd A  
C = 0 : Else : Loop  
End If  
End
```

در برنامه بالا هر سه ساعت یک بار دمای محیط اندازه گرفته میشود و در حافظه داخلی میکرو ذخیره میشود ، بعد از گذشت 12 ساعت دما های ذخیره شده بر روی lcd به نمایش در میاید .

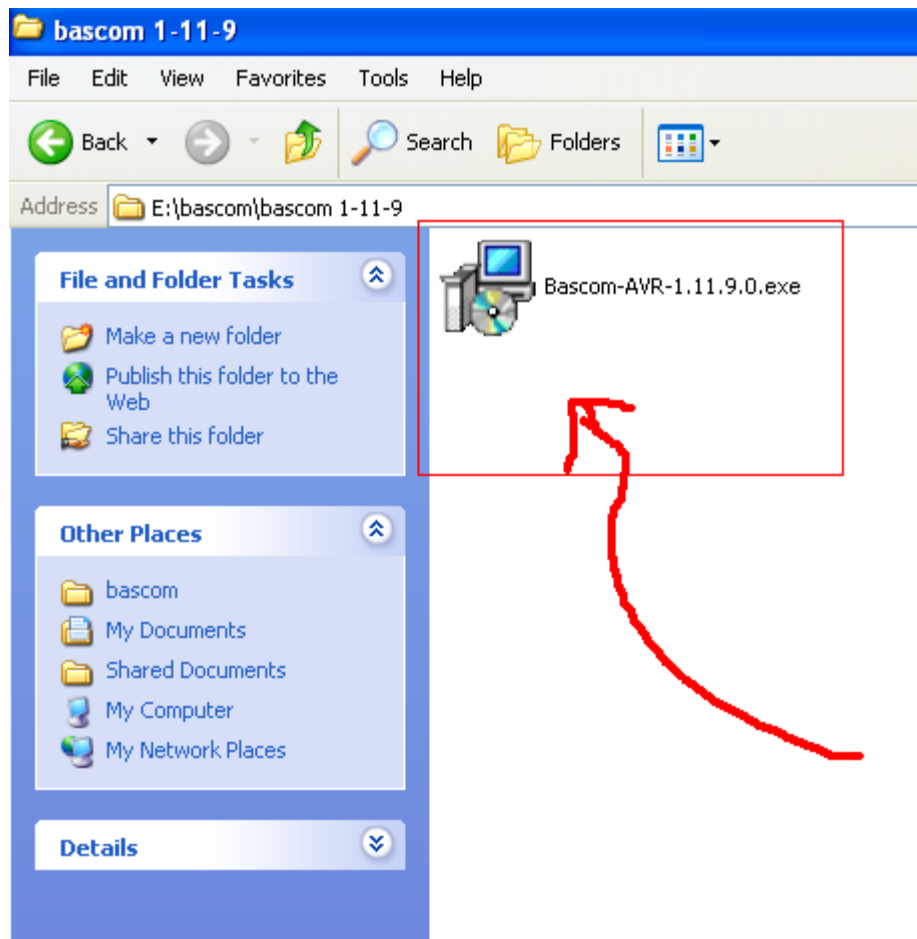
ضمائم:

در این قسمت مطالب و امکاناتی که میتوان با avr راه اندازی کرد آورده شده است :

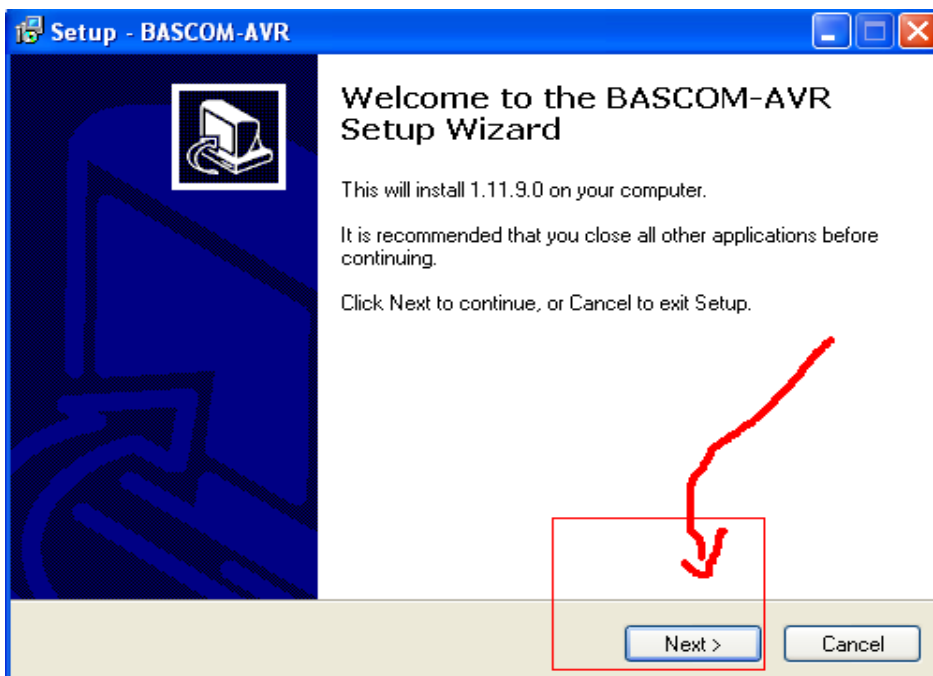
ضمیمه 1 : طریقه ی نصب بسکام

بر ای نصب نرم افزار روی ایکون ان دوبار کلیک راست کنید:

PEYMAN KHODDAMI

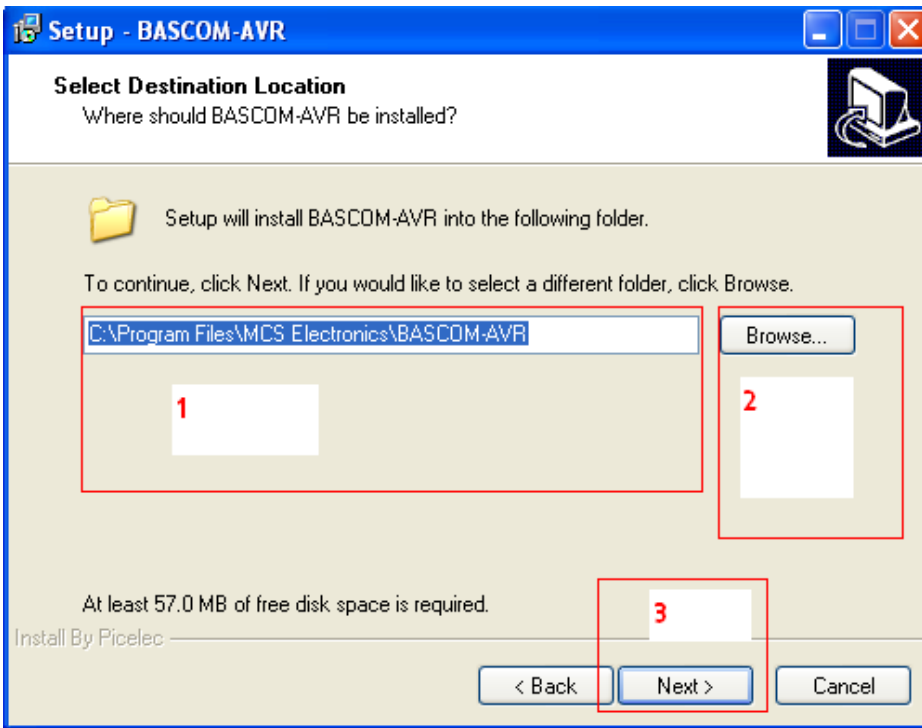


پنجره ی زیر باز میشود ، بر روی دکمه next کلیک کنید:

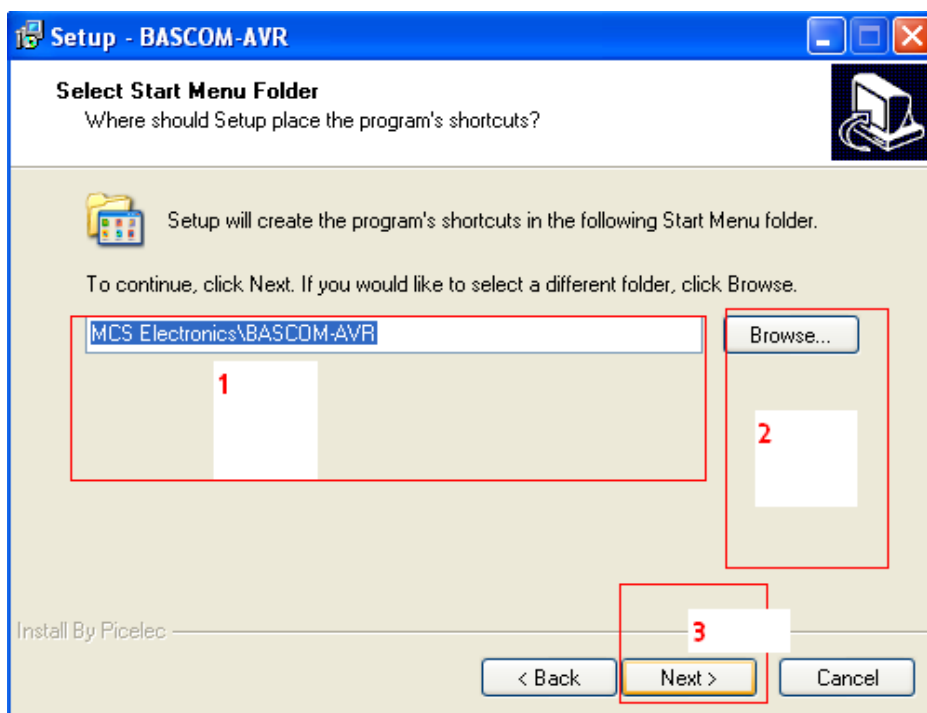


PEYMAN KHODDAMI

پنجره ی زیر باز میشود ، در این پنجره از شما ادرس محل نصب نرم افزار پرسیده میشود ، درصوتی که میخواهید ان را تغییر دهید بازدن کلید browse ... (قسمت 2) ادرس محل نصب را تعیین کنید یا ادرس را در محل 1 تایپ کنید سپس بر روی next (قسمت 1) کلیک کنید:



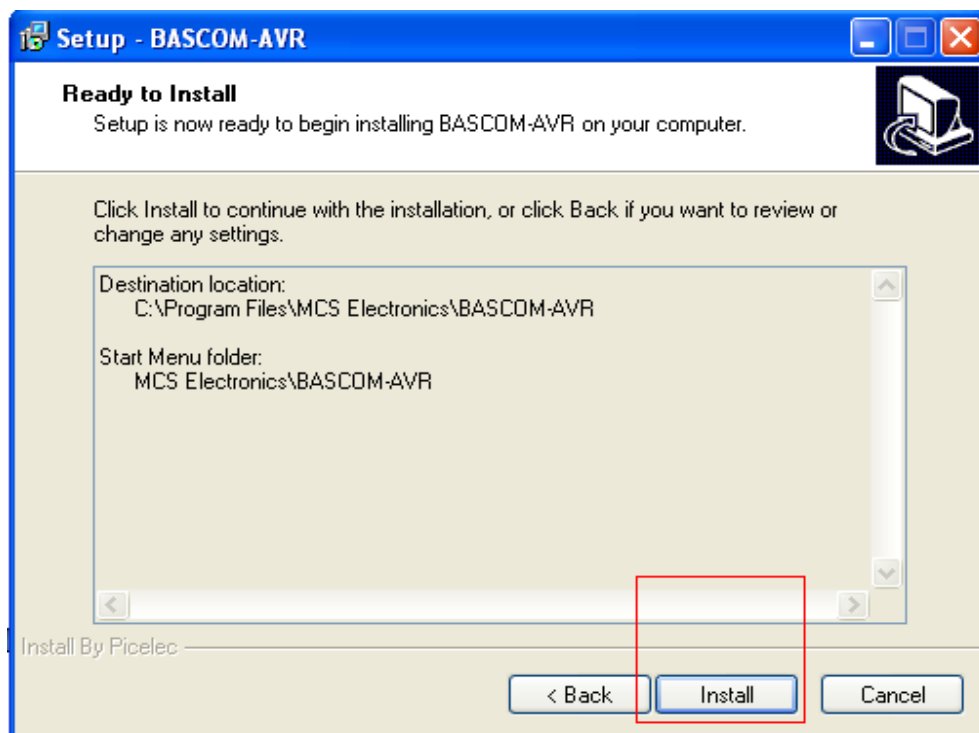
در این پنجره نام پوشه ای که نرم افزار در ان نصب میشود پرسیده میشود که شما میتوانید مانند حالت قبل ان را تغییر



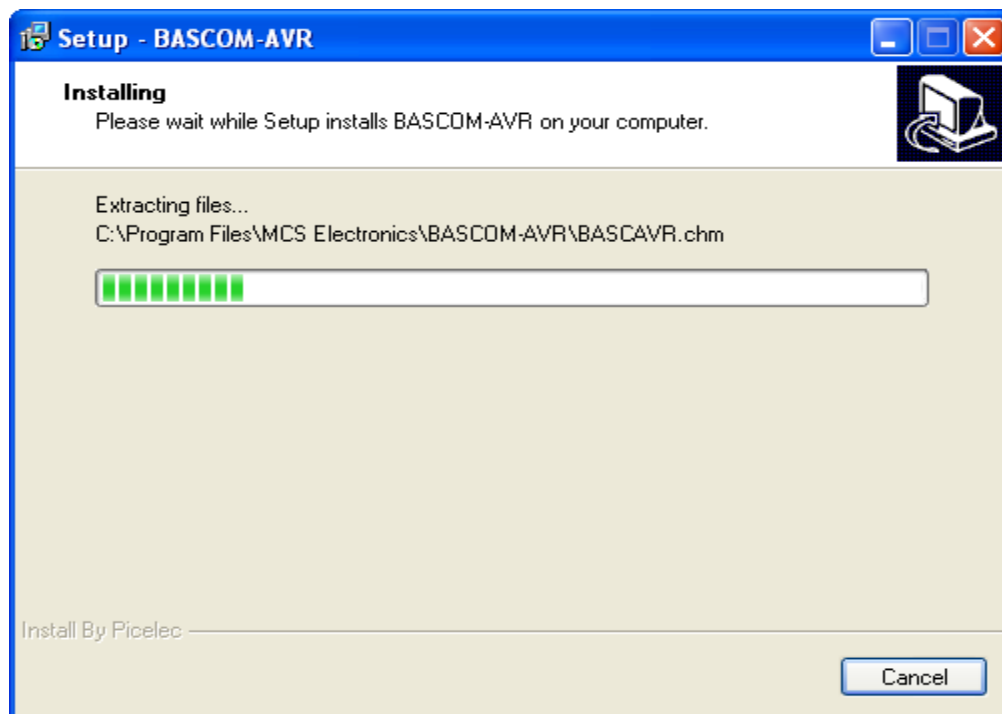
دهید ، بر روی next کلیک کنید:

PEYMAN KHODDAMI

در این پنجره محل نصب و نام پوشه ی نصب یک بار دیگر به نمایش در میاید تا در صورت نیاز ان را اصلاح کنید ،
برای تایید اطلاعات روی Install کلیک کنید:

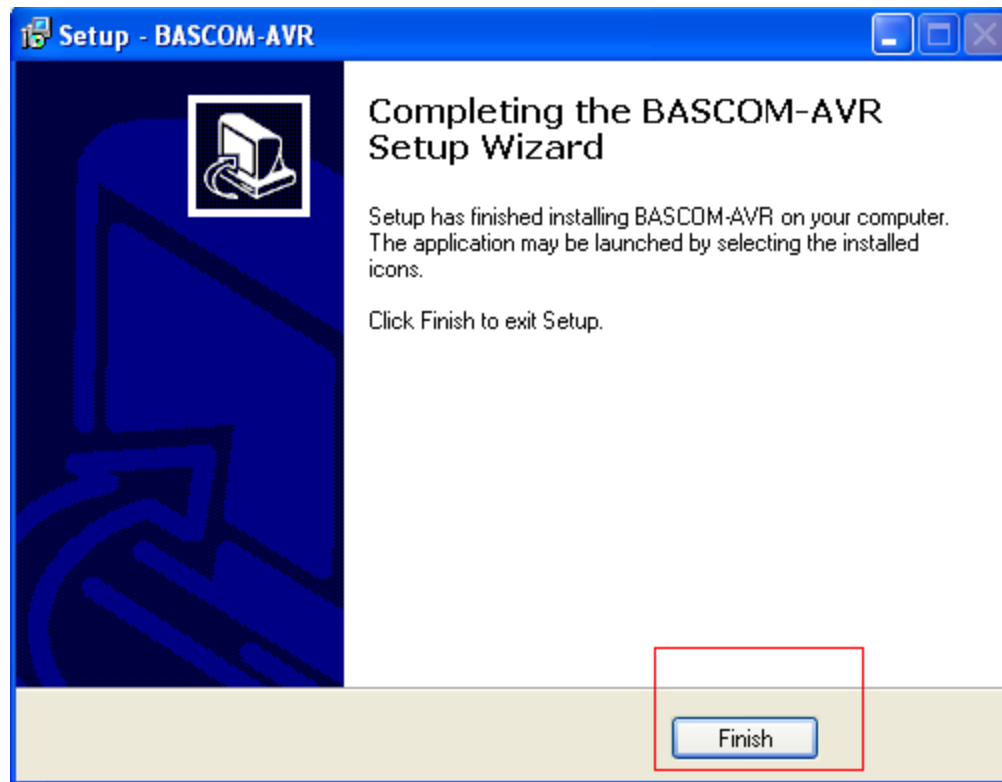


صبر کنید تا نرم افزار نصب شود (اطلاعات کپی شوند):



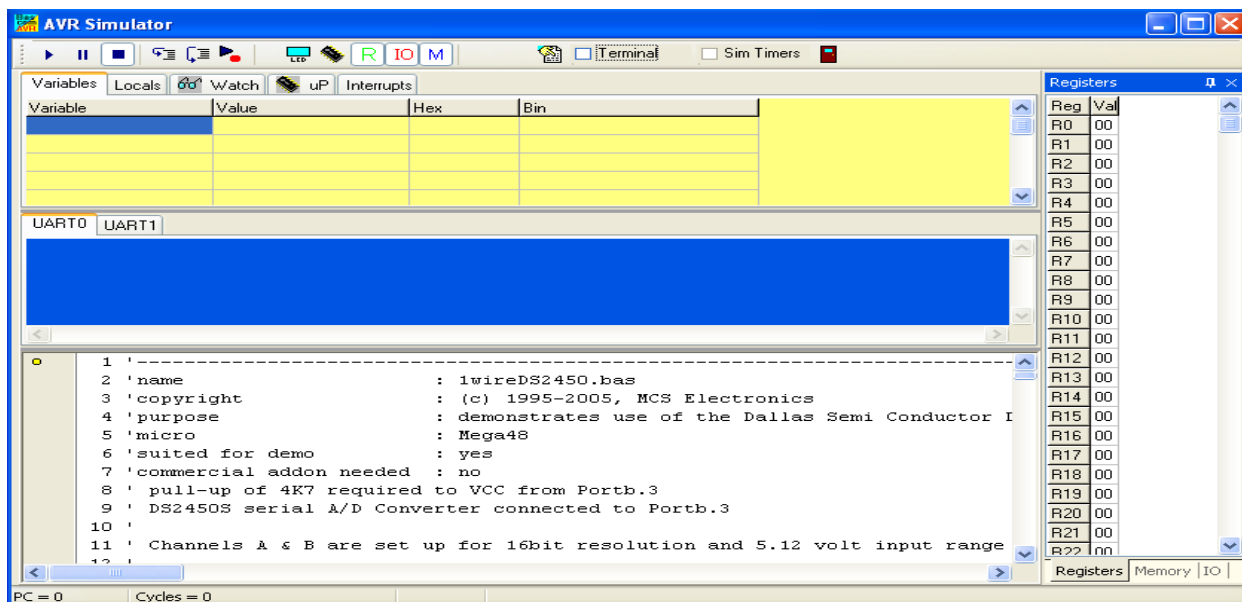
PEYMAN KHODDAMI

و در نهایت بر روی گزینه ی Finish کلیک کنید:



این ورژن تنها ورژن کامل نرم افزار بسکام است و نیازی به کرک ندارد(در زمان تایپ این کتاب).

ضمیمه 2: آشنایی با محیط شبیه سازی بسکام(simulate):

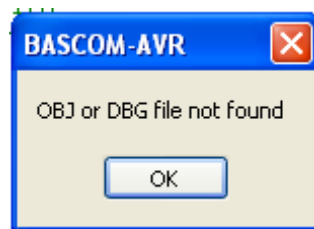


PEYMAN KHODDAMI

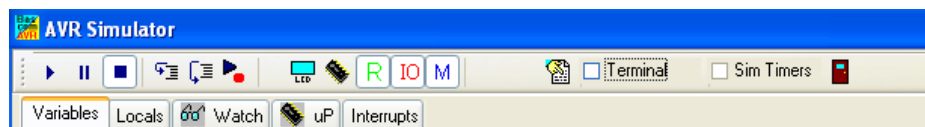
از این محیط برای شبیه سازی برنامه های نوشته شده در بسکام استفاده میشود، شما میتوانید از منوی `Program > simulate`

یا با زدن کلید `f2` به این محیط وارد شوید. (این محیط شبیه سازی در مقایسه با پروتوس خیلی ضعیف است، بهتر است برای شبیه سازی کد با حجم بالا از پروتوس استفاده کنید) این محیط در زیر نمایش داده شده است:

(شبیه ساز برای اجرای برنامه نیاز به فایل `OBJ` و `DBG` دارد که این فایل ها توسط خود برنامه تولید میشود، در صورتی که هنگام شبیه سازی با پیغام زیر شدید، آن را `ok` کنید و کارهای گفته شده را انجام دهید: ابتدا برنامه را کامپایل کنید و دوباره محیط شبیه سازی را فعال کنید، در صورتی که مشکل حل نشد به ادرس `Options > compiler > output` رفته و در آن پنجره گزینه های `Debug file` و `avr Studio Object file` را تیک بزنید، در صورتی که مشکل حل نشد محل ذخیره برنامه را تغییر بدهید و اگر حل نشد کامپایلر را تعویض کنید.)



اولین قسمت تولبار کنترل میباشد:



RUN با فشردن این دکمه شبیه سازی آغاز می شود.

PAUSE باعث توقف موقت شبیه سازی می شود و با فشردن دکمه **RUN** شبیه سازی ادامه پیدا می کند.

STOP باعث توقف کامل شبیه سازی برنامه جاری می شود.

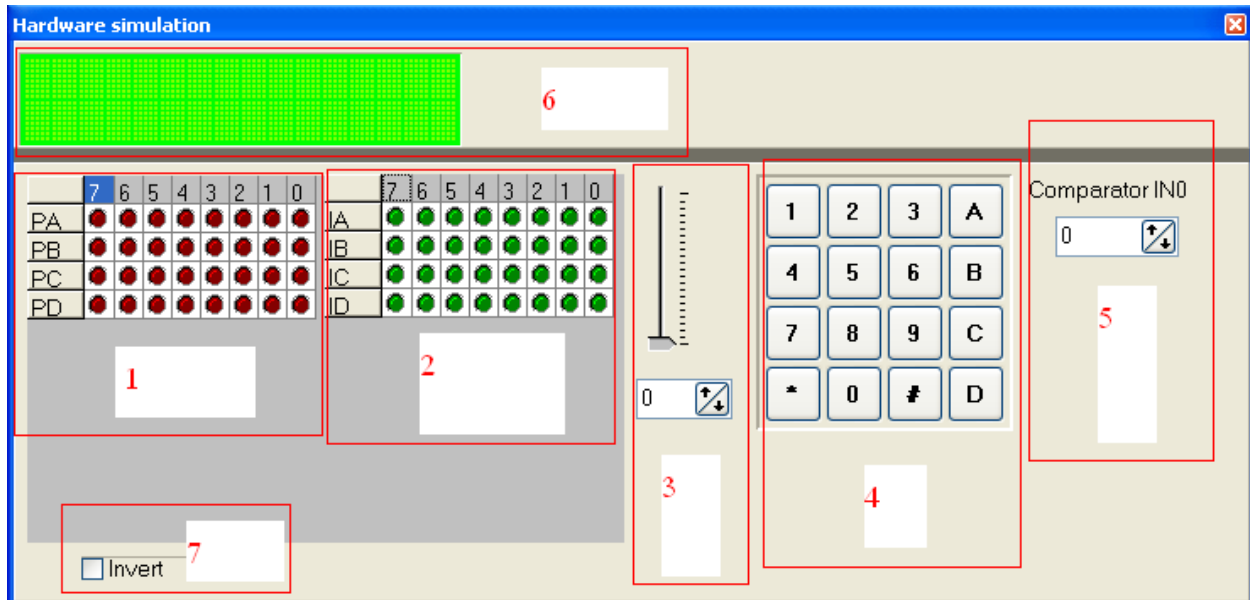
STEP INTO CODE با استفاده از این دکمه می توان برنامه را خط به خط اجرا نمود و هنگام فراخوانی توابع به داخل آنها رفته و مراحل اجرای آنها را بررسی کرد. این کار را با فشردن کلید `F8` نیز می توانید انجام دهید. بعد از هر بار اجرای این دستور شبیه سازی به حالت **PAUSE** می رود.

STEP OVER این دکمه شبیه دکمه قبلی است با این تفاوت که در هنگام فراخوانی توابع به داخل **SUB ROUTINE** نخواهید رفت. این کار را می توانید با فشردن کلید `SHIFT F8` نیز انجام دهید.

PEYMAN KHODDAMI

دکمه RUN TO شبیه سازی را تا خط انتخاب شده انجام میدهد و سپس به حالت PAUSE میرود (خط جاری باید شامل کدهای قابل اجرا باشد) .

Show hardware emulation با کلیک بر روی این گزینه پنجره زیر باز می شود. در این پنجره امکانات سخت افزاری نظیر Lcd ، میدل انالوگ به دیجیتال ، کیبورد 4*4 ، پورت ها و همچنین ورودی مقایسه ای تایمرها موجود میباشد. led های قرمز رنگ نماد پورتهای خروجی و led های سبز رنگ نماد پین های ورودی میباشند ، در صورتی که روی آنها کلیک کنید آنها تغییر رنگ میدهند (مانند این است که پایه را یک led روشن) یا صفر کرده اید .



1- قسمت شماره 1 خروجی پورت ها میباشد و تعداد آن بستگی به تعداد پایه های میکرو دارد (مثلا برای مگا 8 به 3 ردیف کاهش می یابد) هنگامی که شما در برنامه از دستورات set یا ... استفاده میکنید وضعیت این led ها تغییر میکند)

2- این قسمت به عنوان ورودی پورت ها میباشد و تعداد آن مانند حالت قبل به تعداد پایه های میکرو بستگی دارد ، هنگامی که شما در برنامه از دستور debounce یا ... (برای اتصال کلید به میکرو یا ...) استفاده میکنید ، میتوانید با کلیک کردن روی led مورد نظر وضعیت آن را تغییر دهید

3- این قسمت به عنوان ورودی میدل انالوگ به دیجیتال استفاده میشود با تغییر دادن میتوانید به کانال دلخواه بروید و با تغییر دادن میتوانید مقدار ورودی را تغییر دهید.

4- این قسمت مانند یک کیبورد 4*4 است که به یکی از پایه های میکرو متصل شده

5- این قسمت به عنوان یک ورودی برای مقایسه میباشد .

6- این قسمت یک lcd است که متغییر یا اشکالی که در برنامه نوشته اید ("fhhdgd" lcd) را نمایش میدهد ، سطر و ستون آن توسط دستور Config Lcd = x * y تعیین میشود .

7- و در نهایت از این قسمت برای معکوس کردن وضعیت پایه های ورودی و خروجی (led های سبز و قرمز) استفاده میشود.

Enable/disable real hardware emulation با انتخاب این گزینه شما میتوانید میکرو را به پورت سریال وصل کرده و برنامه داخل آن را اجرا کنید (این امکان فقط در بعضی از میکرو ها نظیر 2313 و ... وجود دارد)

PEYMAN KHODDAMI

این دکمه پنجره ثبتها را با مقادیر قبلی نمایش می دهد . مقدارهای نشان داده شده در این پنجره هگزادسیمال می باشد که برای تغییر هر کدام از آنها روی خانه مربوطه کلیک کرده و مقدار جدید را وارد کنید .

برای نمایش ثبتهای ورودی ها و خروجی ها استفاده می شود . که مانند مورد قبلی قابل مقدار دهی است

برای نمایش مکان های مختلف حافظه استفاده میشود .

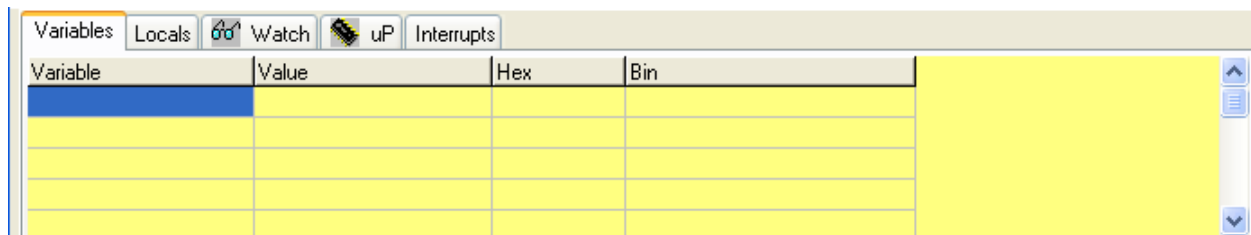
این گزینه برای تازه سازی متغیر ها استفاده میشود (هنگامی که متغیری تغییر کرده باشد ، بازدن این گزینه تغییرات اعمال میشوند .

این گزینه برای استفاده از محیط ترمینال ایمولاتور میباشد (شما میتوانید با فعال کردن این گزینه از محیط ترمینال برای شبیه سازی واقعی استفاده کنید)

هنگامی قصد شبیه سازی تایمر ها را دارید باید این گزینه را تیک برنید.

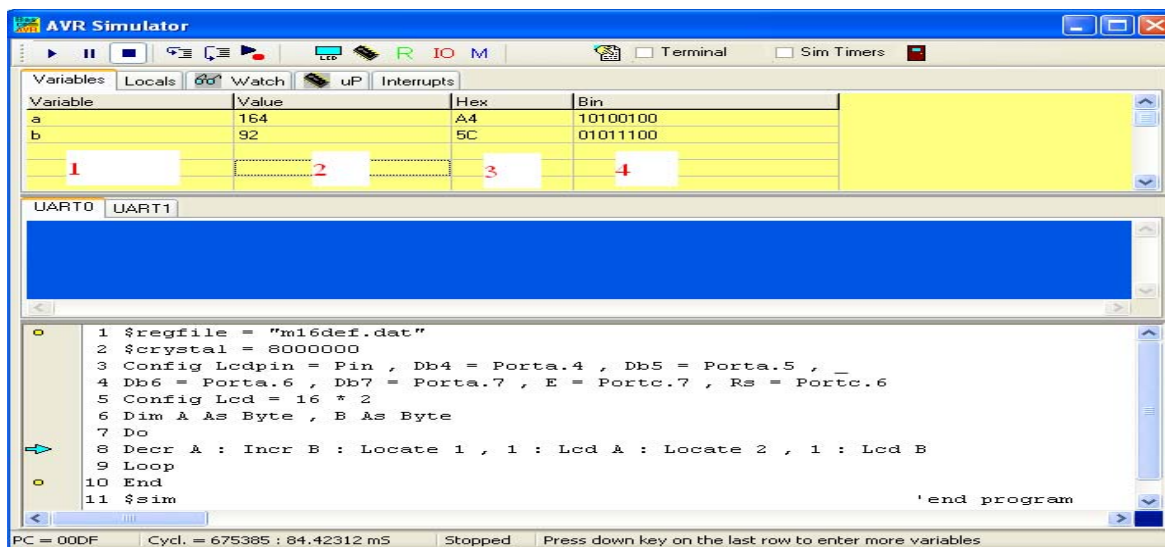
با زدن این گزینه محیط شبیه سازی بسته میشود.

قسمت دوم تولبار کنترل داده ها میباشد (البته اسم کنترل داده رو من روش گذاشتم)



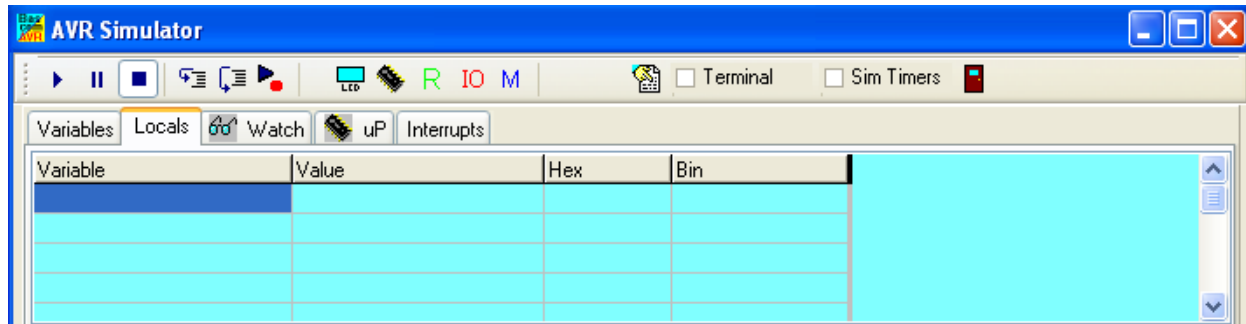
این تولبار از قسمت های زیر تشکیل شده است:

- 1- Variables: شما در قسمت اول نام متغیر موجود در برنامه را مینویسید ، با زدن کلید enter میتوانید مقادیر دسیمال (در قسمت 2) مقدار hex (در قسمت 3) و مقدار باینری را (در قسمت 4) مشاهده فرمایید. همچنین شما میتوانید مقداری را به آن نسبت دهید ، برای این کار مقدار را در قسمت های گفته شده به فرم مورد نیاز بنویسید .

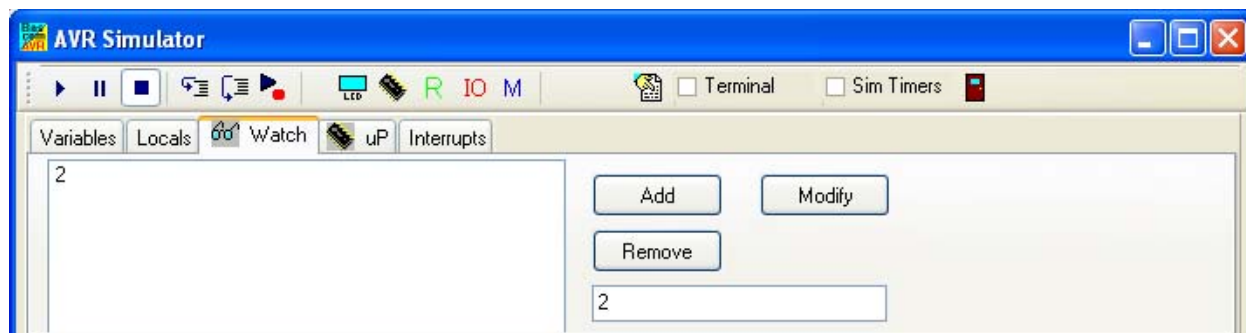


PEYMAN KHODDAMI

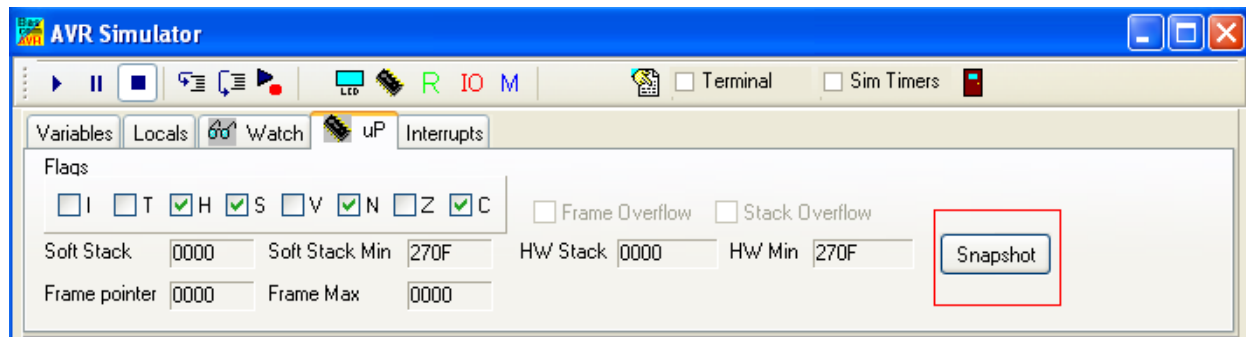
Locals-2: این گزینه مانند حالت قبل است ، تنها تفاوت در این است که مورد قبل متغیر های موجود در حلقه اصلی و این مورد متغیر های موجود در زیر برنامه ها را نشان میدهد:



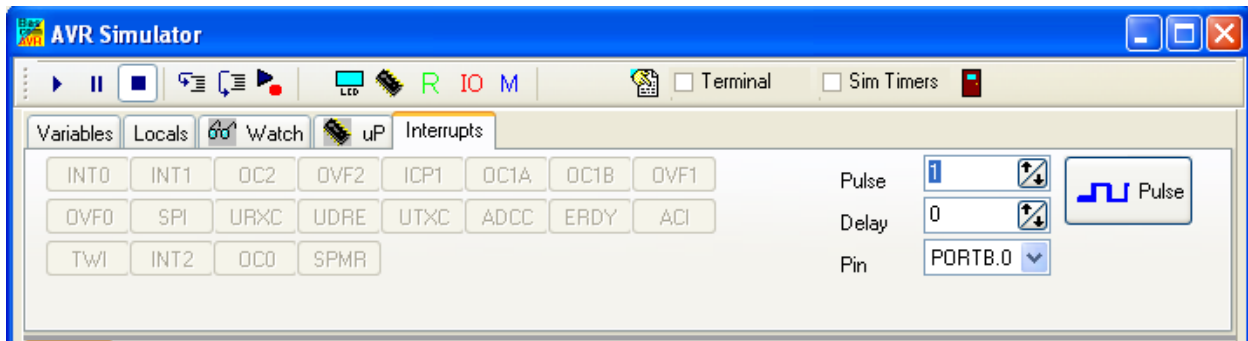
WATCH-3: این گزینه برای وارد کردن وضعیتی که باید ارزیابی شود، مورد استفاده قرار می گیرد و هنگامی که وضعیت مورد نظر صحیح شد شبیه سازی در حالت PAUSE قرار خواهد گرفت . حالت مورد نظر را در مکان مورد نظر تایپ نموده و دکمه ADD را فشار دهید تا وضعیت در کادر نمایش داده شود . هنگامیکه دکمه MODIFY فشار داده شود ، وضعیت مورد نظر مورد بازنگری میشود و عمل متناظر با آن انجام میشود. برای حذف هر وضعیت آنرا در کادر انتخاب کرده انتخاب کرده و دکمه REMOVE را فشار دهید .



up-4 در این قسمت میتوان وضعیت رجیستر های داخلی را مشاهده کرد (رجیستر تایمر ها ، adc و ...). ابتدا بر روی Snapshot کلیک کنید ، (به طور پیش فرض انتخاب شده است) و در زمان دلخواه دوباره روی آن کلیک کنید ، کادری باز میشود و در آن وضعیت رجیستر های انتخاب شده نمایش داده میشود ، برای اجرای دوباره روی کلید stop کلیک کنید (Snapshot به stop تبدیل میشود)



PEYMAN KHODDAMI

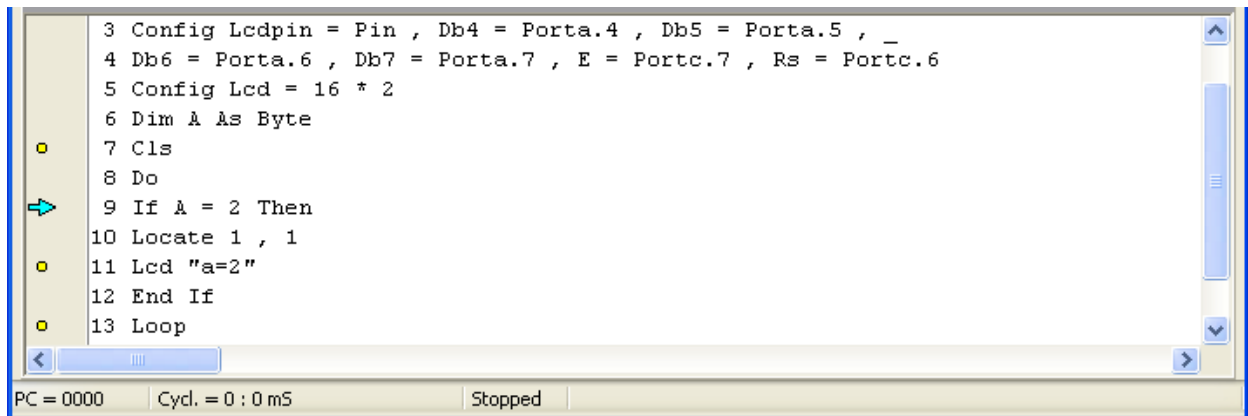


5- Interrupts : در این قسمت منابع وقفه خارجی میکرو مورد استفاده نمایش داده میشود (تنها در صورتی که در برنامه از آنها استفاده کنید فعال میشوند) همچنین شما میتونید به پایه دلخواه یک پالس دلخواه اعمال کنید .

قسمت سوم ترمینال ایمولاتور میباشد که از ان برای نمایش داده دریافتی از میکرو استفاده میشود:



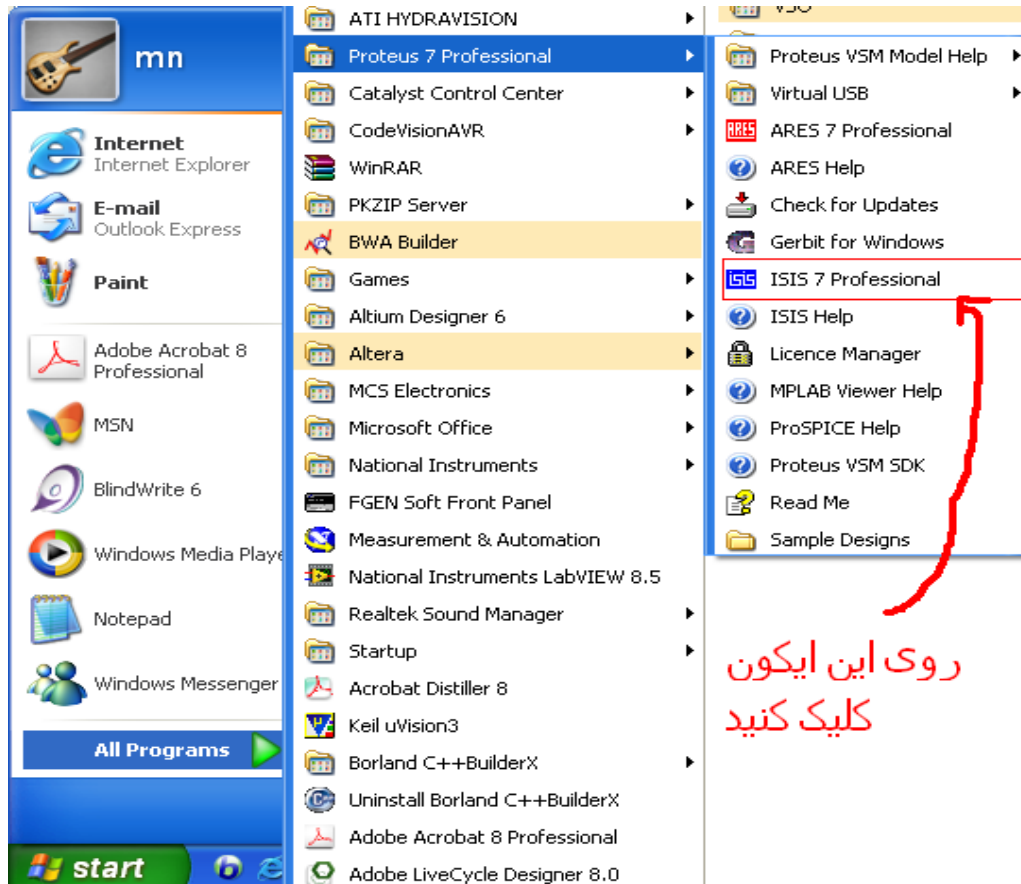
آخرین قسمت محل نمایش برنامه میباشد ، در زیر این قسمت وضعیت cpu میکرو و زمان که از آغاز شبیه سازی گذشته نمایش داده میشود:



PEYMAN KHODDAMI

ضمیمه 3: شبیه سازی میکرو کنترل avr با برنامه پروتوس (آشنایی مقدماتی)

بعد از نصب و فعال کردن نرم افزار پروتوس روی ایکون آن در منوی استارت کلیک کنید تا نرم افزار باز شود:



محیط نرم افزار به شکل زیر است :

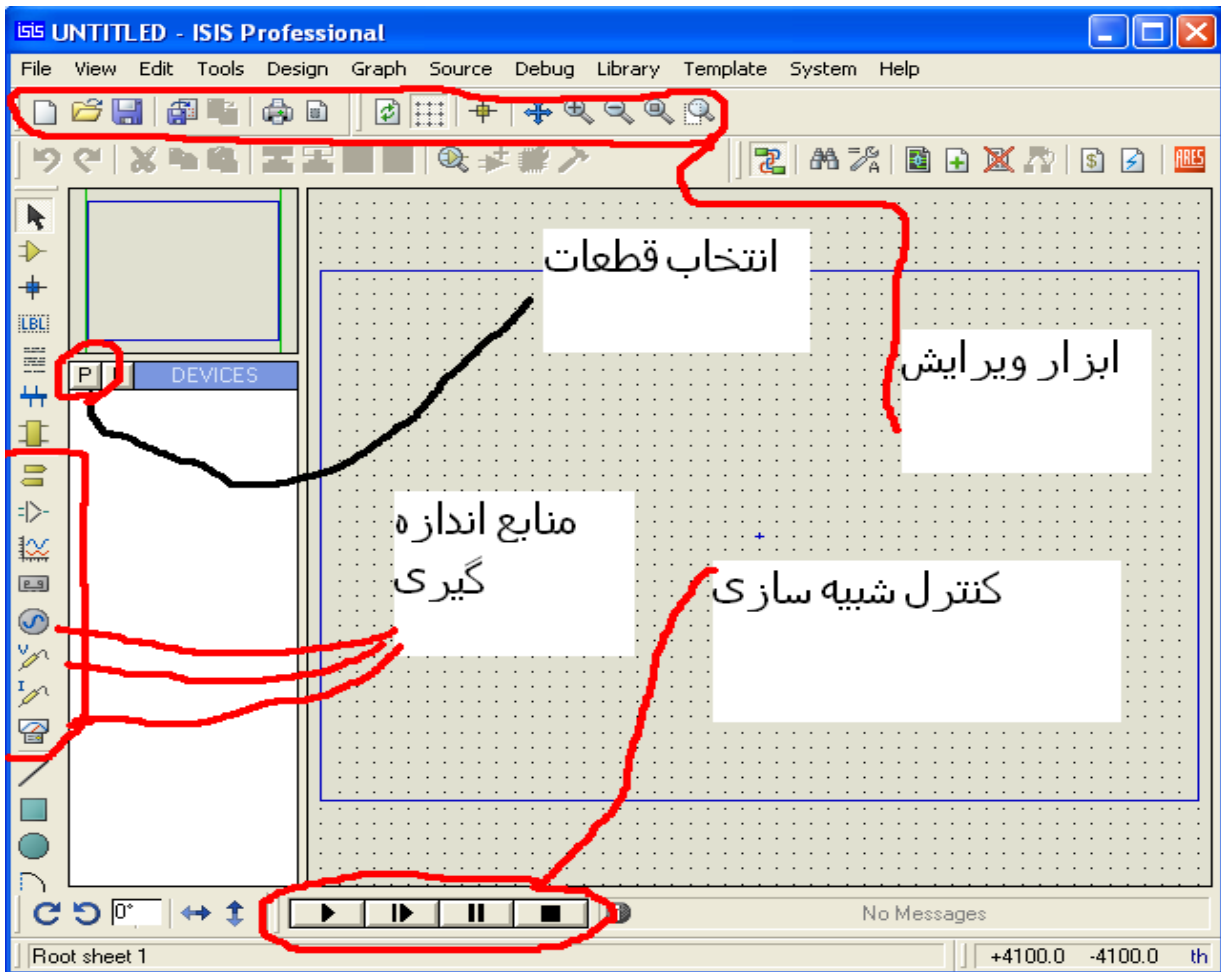
در قسمت منابع اندازه گیری ، مجموعه ای از منابع ولتاژ، جریان، اسلیسکوپ، ولتمتر ، سیگنا ژنراتور ، منابع پالس (ac ، مربعی ، ...) وجود دارد که شما میتوانید آنها را انتخاب کنید و در صفحه بگذارید (ابتدا روی آنها کلیک کنید ، سپس به صفحه آمده و در نقطه مورد نظر دوباره کلیک کنید تا قطعه در آنجا گذاشته شود).

در قسمت ابزار ویرایش ابزاری برای ذخیره و تهیه پرینت و ... و بزرگ و کوچک کردن صفحه و .. وجود دارد که در صورت نیاز میتوانید آنها را به کار ببرید.

در قسمت کنترل شبیه سازی 4 کلیک برای شروع و مکث و توقف و دیباگ کردن برنامه وجود دارد (برای اطلاعات بیشتر به نسخه آموزش پروتوس 7.2 مراجعه کنید)

در قسمت انتخاب قطعه ، شما میتوانید قطعات بکار رفته در مدار مورد نظرتان را انتخاب کنید و سپس آنها را طبق نقشه به هم متصل کرده و شبیه سازی کنید ، در زیر با یک مثال این عمل توضیح داده میشود:

PEYMAN KHODDAMI



مدار مربوط به یک شمارنده میباشد ، که با استفاده از تایمر/کانتر 1 پالس های اعمالی به پایه t1 میکرو مگا 16 را می شمارد و روی یک 16*2 lcd نمایش میدهد:

```
$regfile = "m16def.dat"
```

```
$crystal = 8000000
```

```
Config Lcdpin = Pin , Db4 = Portd.2 , Db5 = Portd.3 , Db6 = Portd.4 , Db7 = Portd.5 , E = Portd.1 ,  
Rs = Portd.0
```

```
Config Lcd = 16 * 2
```

```
Config Timer1 = Counter , Edge = Rising
```

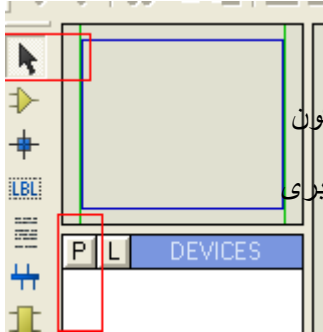
```
Do
```

PEYMAN KHODDAMI

Locate 1 , 1 : Lcd Counter1

Loop

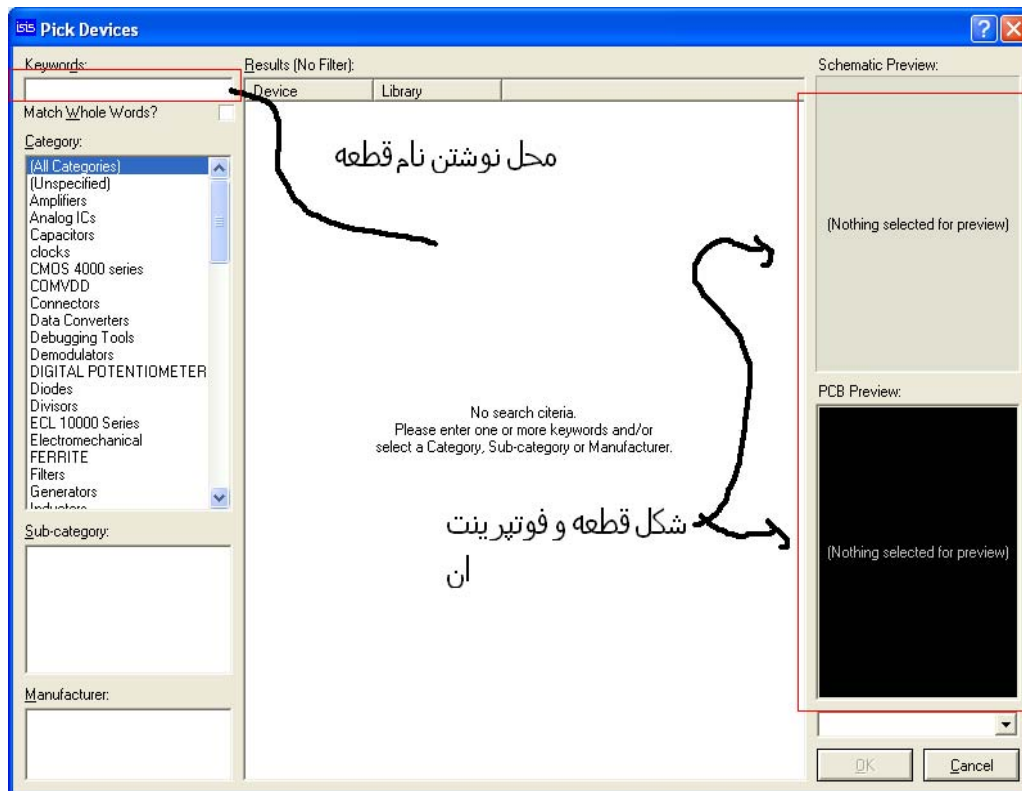
End



اولین قدم آوردن قطعات است برای این کار روی گزینه p کلیک کنید .
توجه داشته باشید که برای کلیک روی ایکون (p) pick from li... باید ابتدا روی ایکون Selection mode (موس مشکی) کلیک کنید. بعد از کلیک روی ایکون p پنجره لایبری باز میشود :

در قسمت " محل نوشتن نام قطعه " اسم قطعات مرود نیاز را تک تک بنویسید و سپس انها را انتخاب کنید تا در محل که بعدا گفته میشود نمایش داده شود، قطعات مورد نیاز:

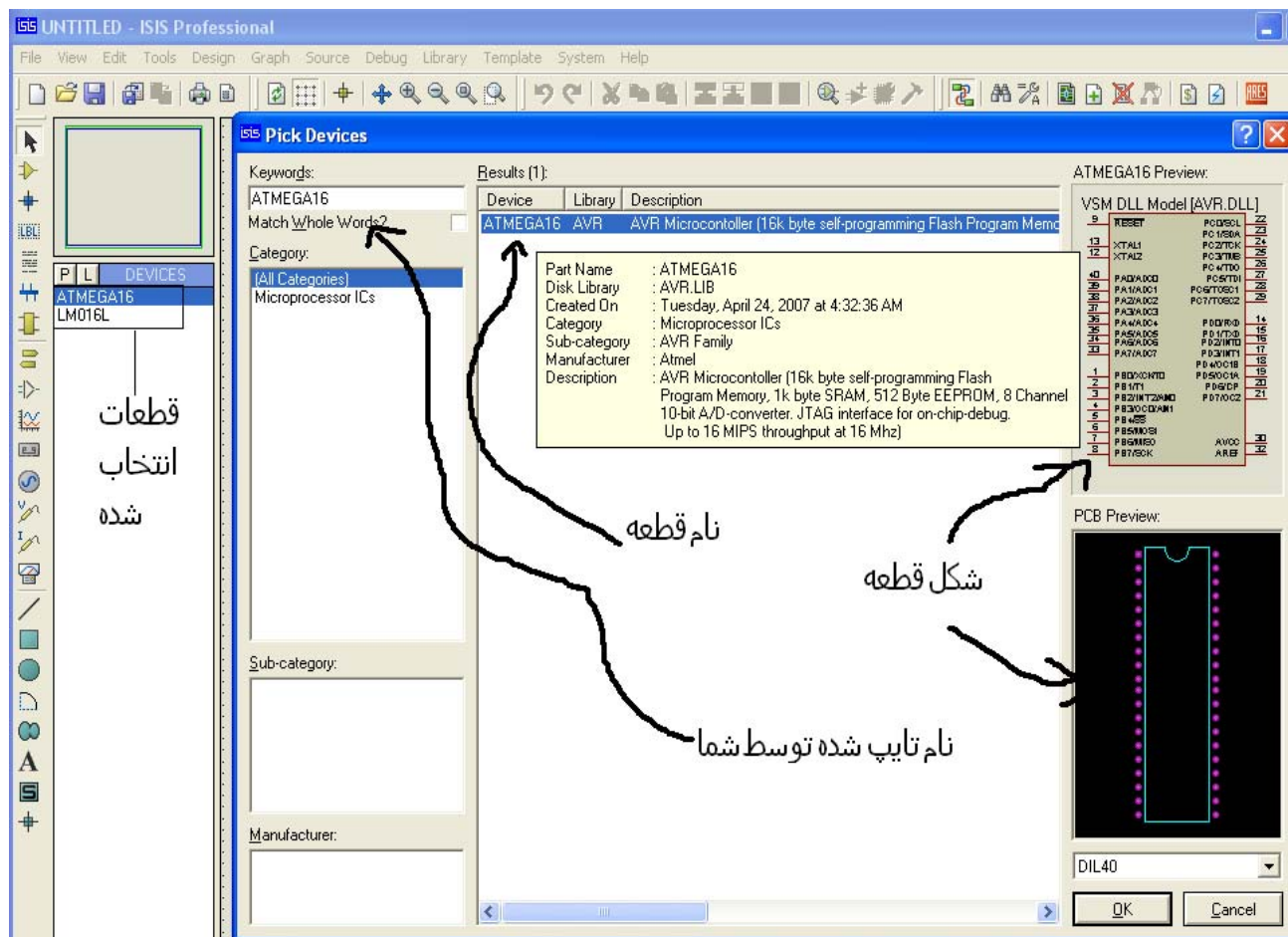
- 1 میکرو مگا 16 : atmega16
- 2 LM016L: Lcd 16*2
- 3 کریستال : CRYSTAL



PEYMAN KHODDAMI

برای نمونه نام میکرو را در قسمت گفته شده تایپ کنید ، نرم افزار قطعه را به شما معرفی میکند (مانند شکل زیر) روی نام آن دوبار کلیک کنید تا نام آن به قطعات انتخاب شده افزوده شود.

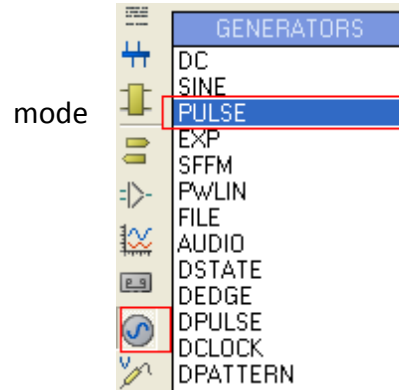
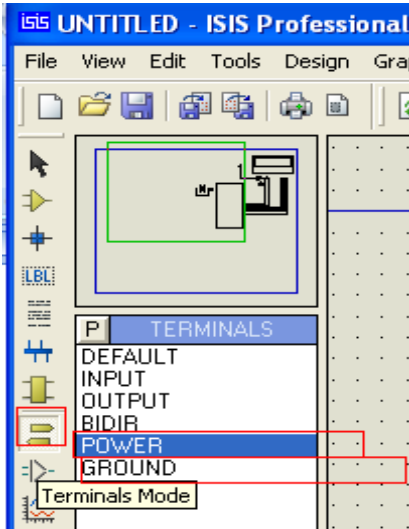
این کار را برای دیگر قطعات نیز انجام دهید و در آخر روی OK کلیک کنید تا کتابخانه بسته شود



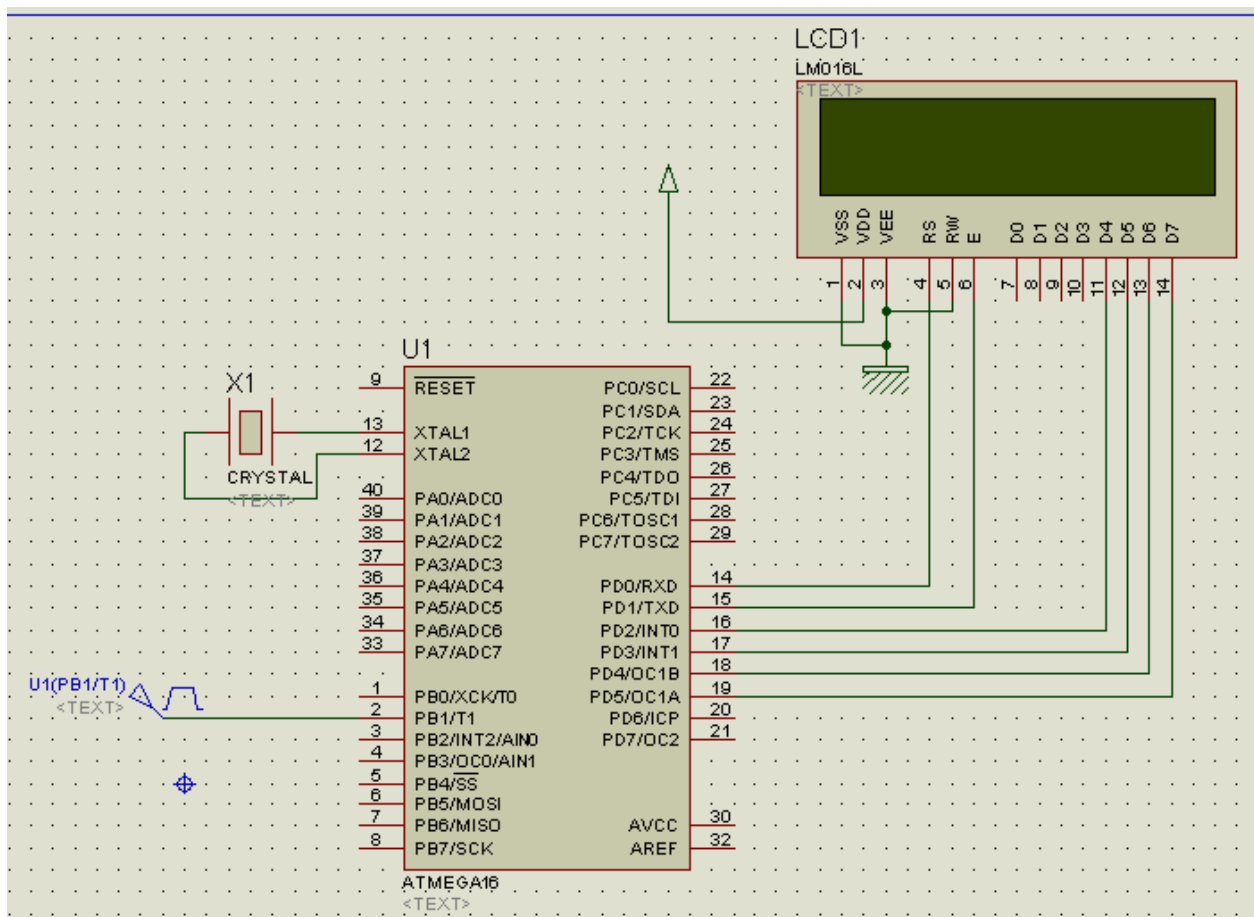
PEYMAN KHODDAMI

مدار شما باید طبق برنامه به شکل زیر باشد :

برای آوردن VCC و گراند ، روی ایکون terminals mode کلیک کنید و در آنجا گزینه ای POWER و GROUND را انتخاب کنید.

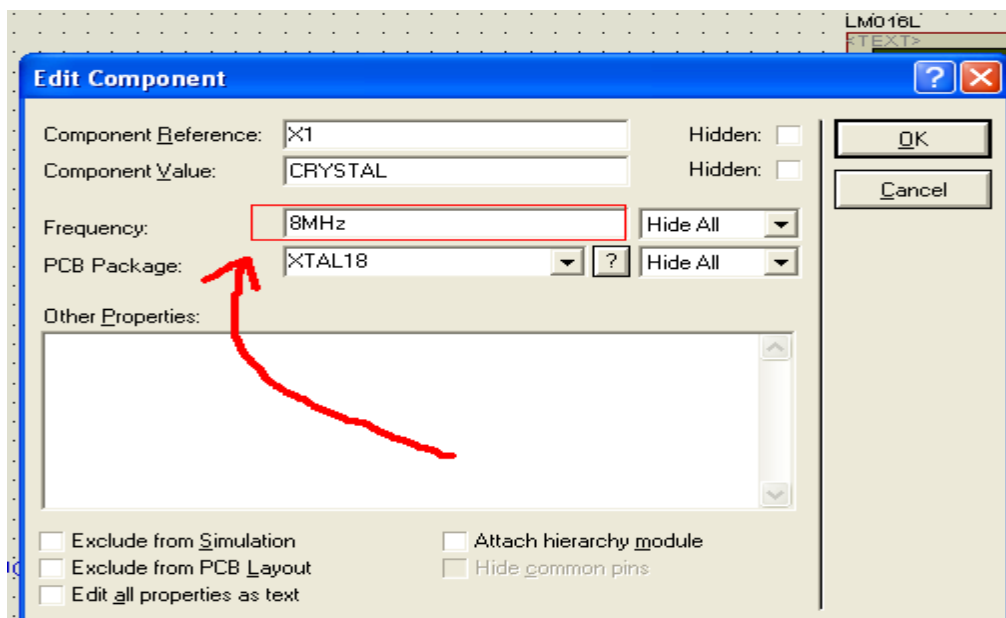
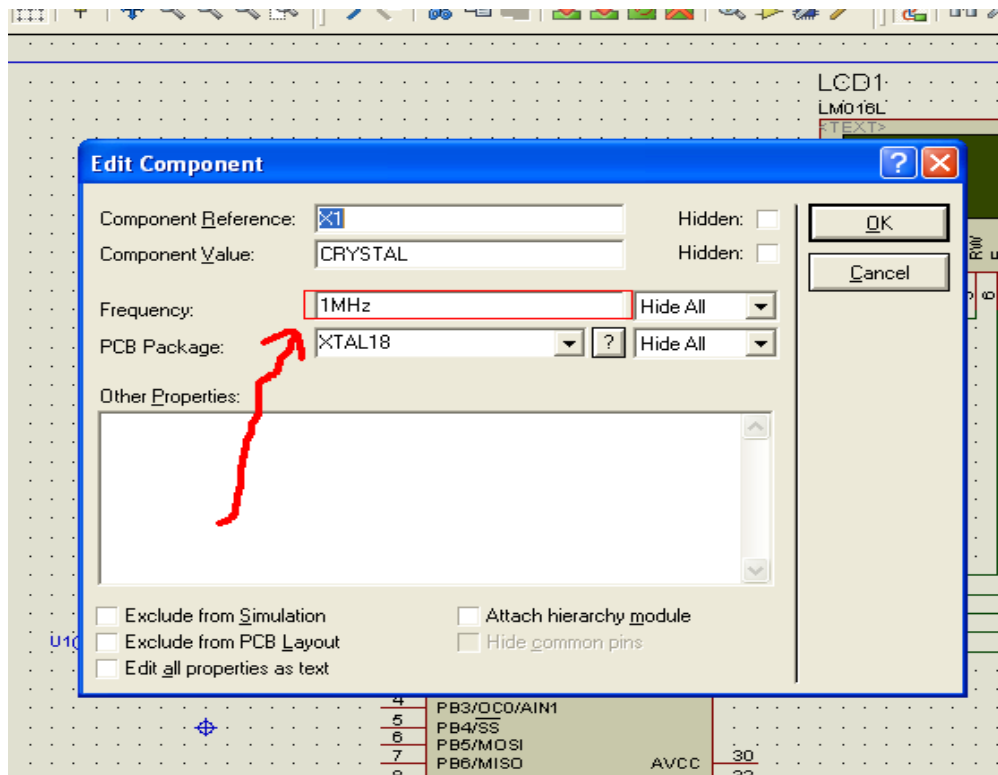


همچنین برای آوردن منبع پالس روی ایکون generator کلیک کنید و در آنجا گزینه PULSE را انتخاب نمایید .



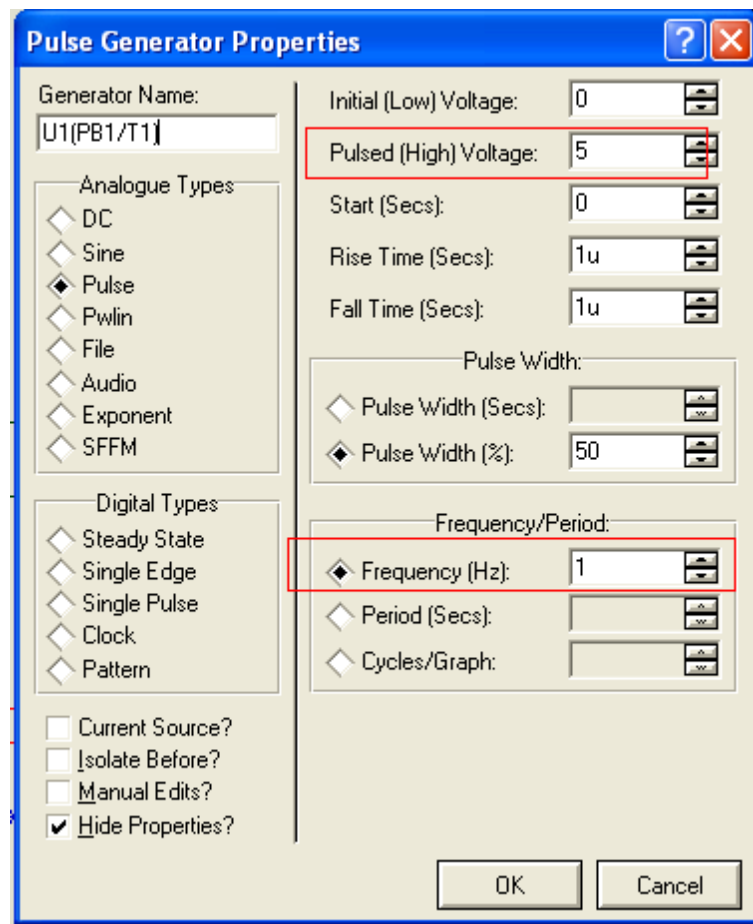
PEYMAN KHODDAMI

کلیه قطعات دارای مقادیر پیش فرض میباشند ، مثلا مقدار کریستال 1 مگا هرتز است ، دامنه پالس ورودی 1 ولت است ، ما باید همه آنها را به مقادیر دلخواه خودمان تغییر دهیم ، برای این کار روی قطعه (در صفحه اصلی) دو بار کلیک کنید ، پنجره ای باز میشود که در آن مشخصات قطعه موجود میباشد ، من بر روی کریستال دو بار کلیک راست کردم و پنجره زیر باز شد ، در این پنجره باید فرکانس کریستال را به مقدار نوشته شده در برنامه تغییر داد ، برای این کار فرکانس جدید را در قسمت نمایش داده شده تایپ کنید:



PEYMAN KHODDAMI

دامنه پالس ورودی را به 5 ولت تغییر میدهیم ، همچنین فرکانس ان را نیز به مقدار دلخواه تغییر میدهیم (برای این کار روی منبع پالس دوبار کلیک کرده م مشخصات ان را به مشخات شکل زیر تغییر دهید:

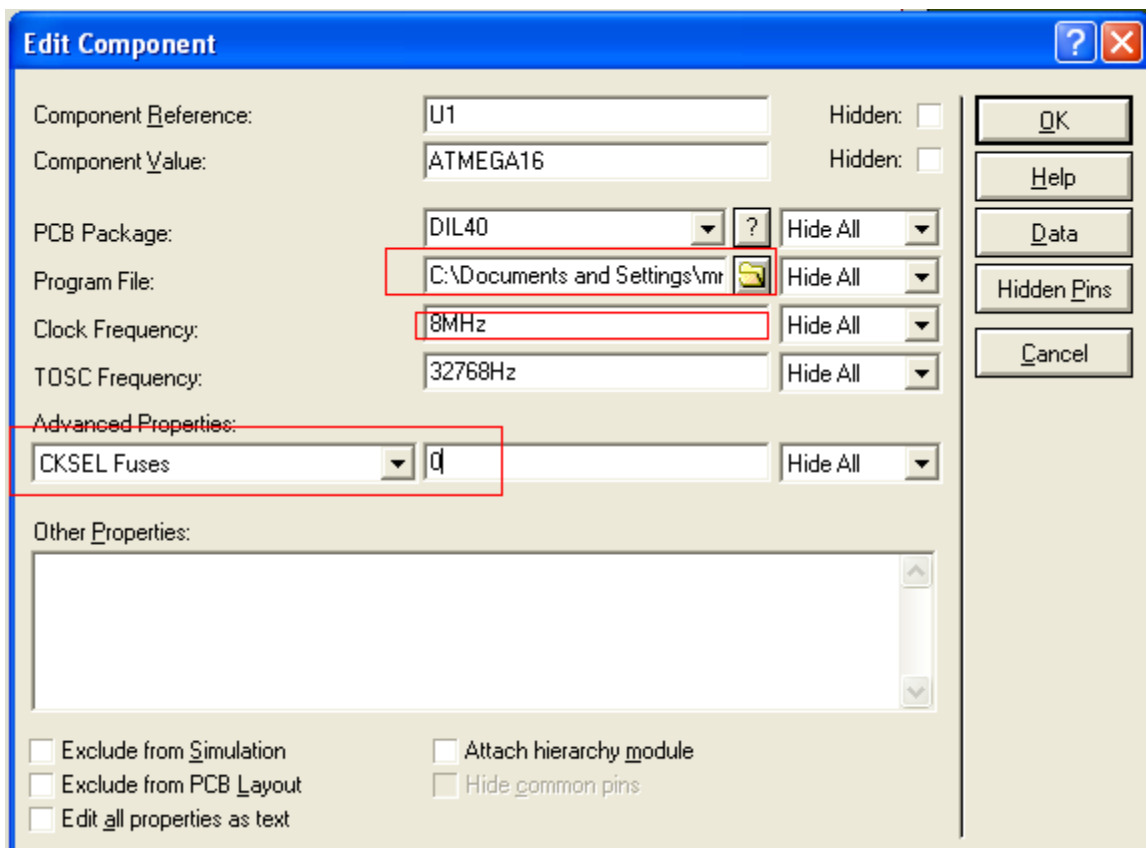


برای ریختن کد هگز روی میکرو ، روی ان دوبار کلیک چپ کنید ، در پنجره باز شده ادرس محل ذخیره کد هگز را بدهید ، همچنین فرکانس کریستال را به 8 مگا هرتز تغییر دهید و فیوز بیت مربوط به کریستال خارجی را نیز برنامه ریز ی کنید:

کد هگز با پسوند .hex ذخیره شده است ، برنامه ای که در بالا موجود است را در محیط بسکام کپی کرده و ان را کامپایل کنید .. کد هگز را وارد میکرو کنید.

PEYMAN KHODDAMI

و در نهایت روی ایکون شروع شبیه سازی کلیک کنید ، مشاهده میکنید که تعدا پالس های خروجی منبع پالس توسط میکرو شمرده شده و روی lcd نمایش داده میشود:



ضمیمه 4 پروگرام کردن میکرو : (معرفی منوی send to chip)

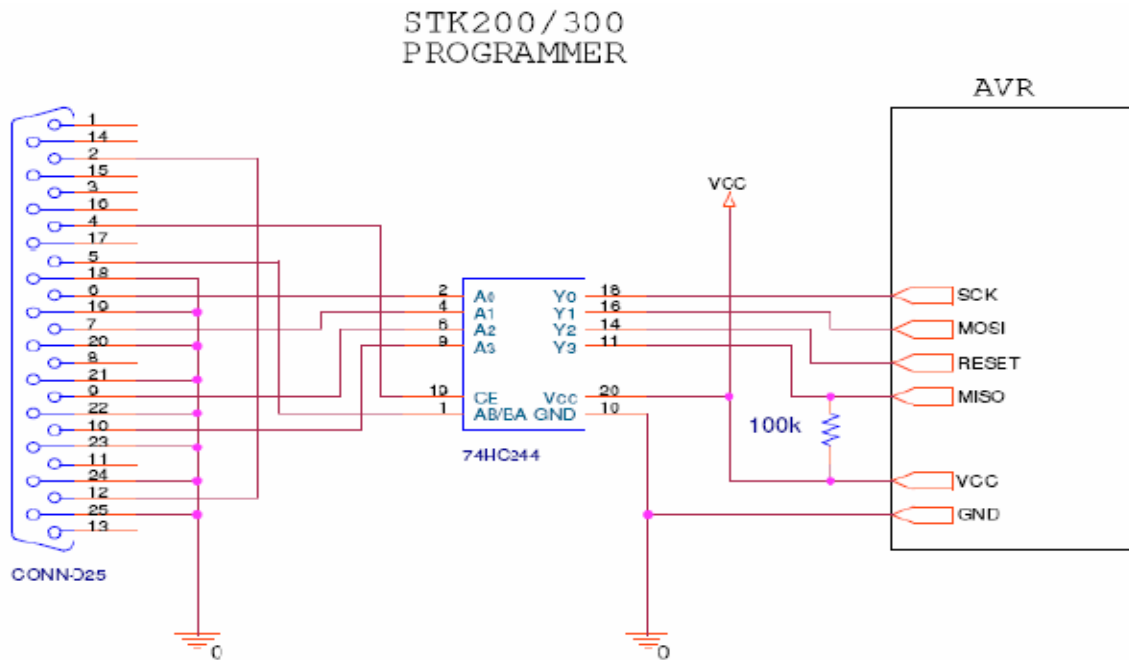
بعد از اینکه برنامه مورد نظرتان را نوشتید و آن را تست کردید ، باید آن را روی میکرو بریزید برای این کار به دستگاهی به نام پروگرامر نیاز دارید ، تا بین کامپیوتر و میکرو قرار گیرد و کد هگز موجود را از کامپیوتر به میکرو انتقال دهد. پروگرامرها در نمونه ها و قیمت های متنوعی ساخته و عرضه شده اند ، شما میتوانید نمونه مناسب با پورت کامپیوتر (پروگرامرها به یکی از پورت های کامپیوتر متصل میشوند ، این پورت ممکن است پورت موازی (Ipt) یا چاپگر) ، پورت سریال (com) یا پورت usb باشد) خود را انتخاب کنید و از آن استفاده کنید .یکی از پروگرامرهای رایگان ارائه شده برای avr ، stk200/300 میباشد ، کامپایلر بسکام و دیگر کامپایلرهای مخصوص avr از آن پشتیبانی میکنند و ساخت آن ساده و بدون دردسر می باشد .

این پروگرامر به پورت موازی متصل میگردد و با 4 سیم + ولتاژ های تغذیه به میکرو متصل میشود ، پروتکل این پروگرامر isp است . مدار این پروگرامر را در زیر مشاهده میکنید:

برای ساخت مدار نکات زیر را رعایت کنید:

PEYMAN KHODDAMI

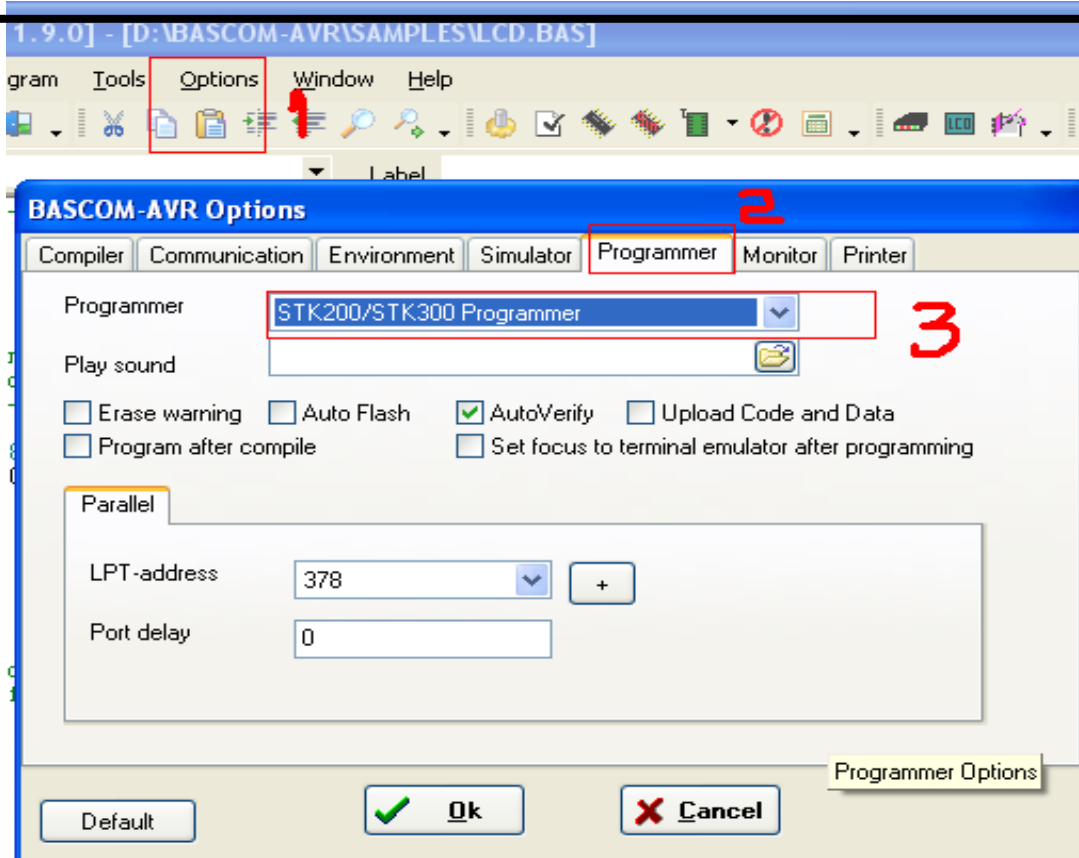
- 1- این ضمیمه را کامل بخوانید.
- 2- طول کابل پروگرامر از 50 نانثی متر بیشتر نباشد(بین کامپیوتر و پروگرامر کمتر از 50 سانتی متر ، بین میکرو و پروگرامر کمتر از 50 سانتی متر).
- 3- برای تغذیه میکرو به ولتاژ صاف و رگوله شده 5 ولت نیاز است ، شما میتوانید این ولتاژ را از پورت usb تامین کنید ، vcc به پایه میکرو و گراند به گراند پروگرامر و گراند میکرو متصل میشود.
- 4- در صورتی که پورت lpt را روبروی خود بگیرید (پورتهای که خریداری میکنید) میتونید شماره پایه های آن را ببینید.



برای راه اندازی این پروگرامر به مسیر زیر بروید و

Options>Programmer و تنظیمات پنجره ی موجود را مانند عکس زیر کنید (محیط بسکام از پروگرامرهای دیگری نیز پشتیبانی میکند ، شما میتونید از آنها استفاده کنید)

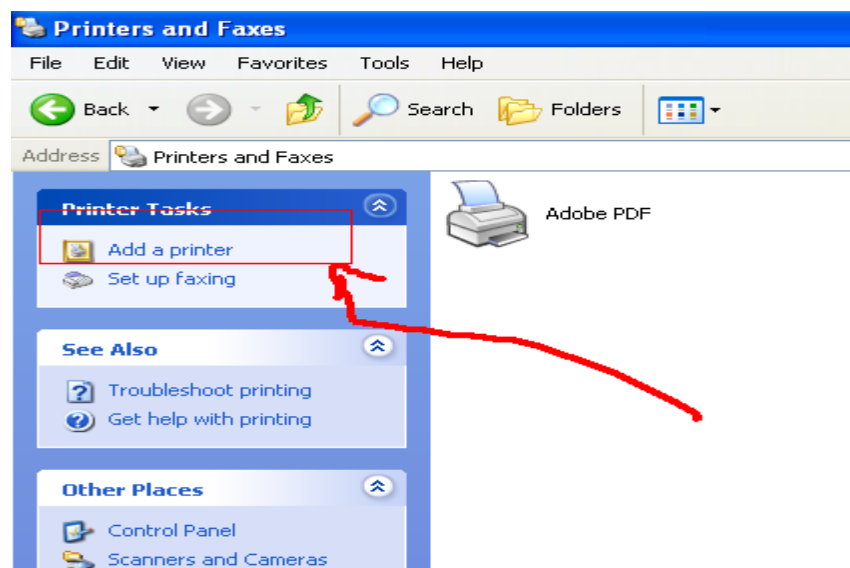
PEYMAN KHODDAMI



همچنین شما باید به کنترل پانل ویندوز بروید و در آنجا یک پرینتر نصب کنید ، برای این کار به مسیر زیر بروید

Start menu>control panel>Printers and Faxes

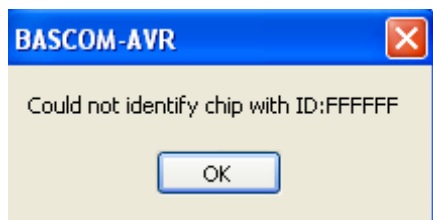
در پنجره باز شده بر روی گزینه add a printer کلیک کنید:



پنجره ای باز میشود ، در همه پنجره ها فقط گزینه next را بزنید، و در نهایت روی finish کلیک کنید.

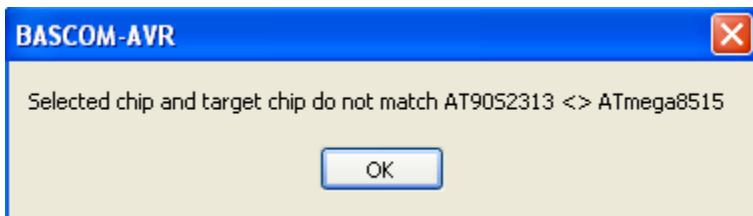
PEYMAN KHODDAMI

سخت افزار پروگرامر آماده است ، (برای اطلاع از موقعیت پایه های میکرو ، که پروگرامر به انها متصل میشود به دیتا شیت میکرو مراجعه کنید. حال باید برنامه را کامپایل کنید و ان را روی میکرو بریزد ، برای دستیابی به نرم افزار پروگرامر ، در محیط بسکام به ادرس `program>send to chip` بروید یا کلید `f4` را فشار دهید ، ممکن است با پیغام های زیر روبرو شوید :



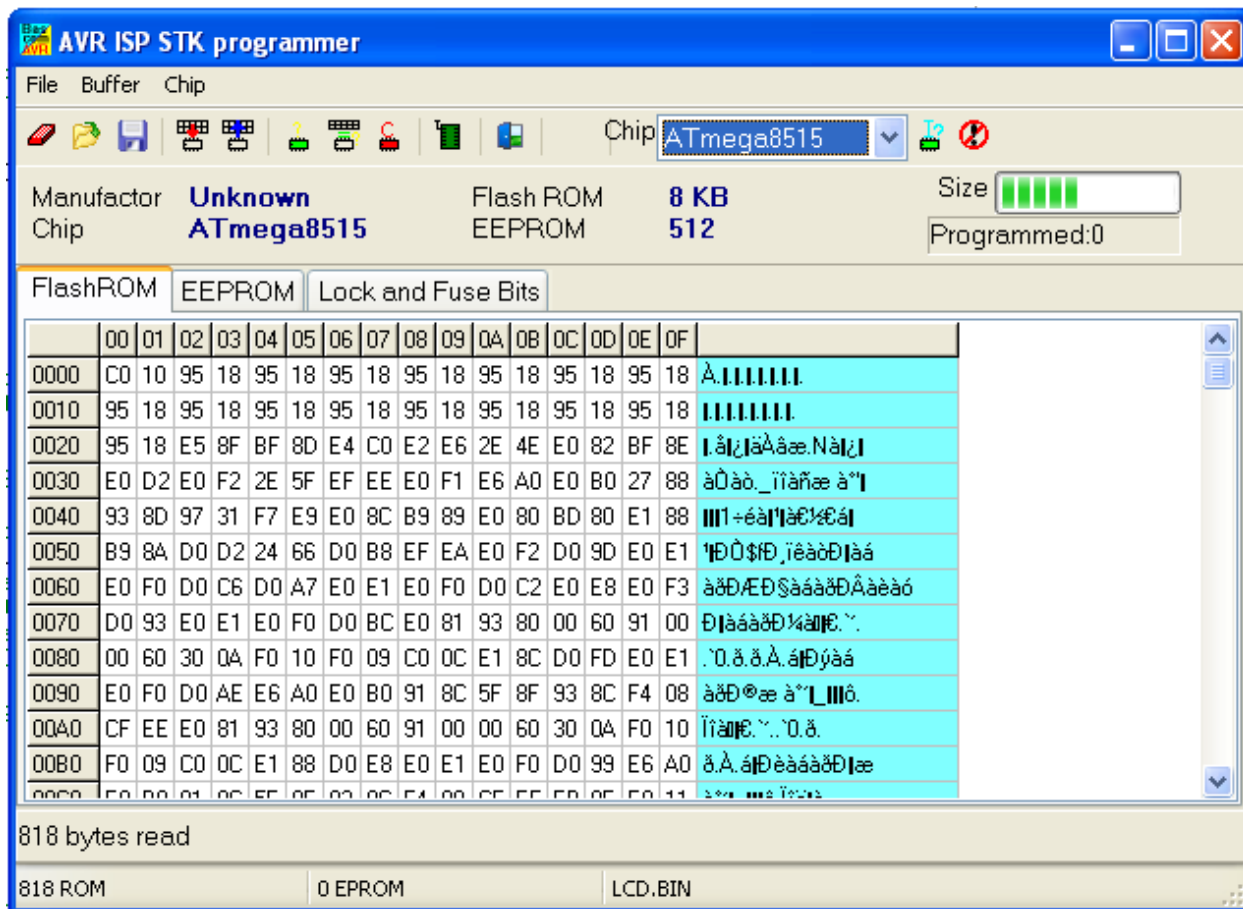
این پیغام ، هنگامی که سخت افزار پروگرامر به درستی به کامپیوتر متصل نشده باشد نمایش داده میشود(ممکن است میکرو به پروگرامر ، یا پروگرامر به کامپیوتر متصل نشده باشد)

این پیغام هنگامی که میکرو نوشته شده در برنامه با میکرو ی متصل شده به



پروگرامر یکی نباشد نمایش داده میشود.

در صورتی که همه چیز درست باشد ، پنجرخ ی زیر بدون هیچ پیغامی نمایش داده میشود:



PEYMAN KHODDAMI



منوی FILE

EXIT : با زدن این گزینه پنجره پروگرامر بسته میشود .

TEST : با زدن این گزینه پایه های پورت 1 lpt میشود و میتوان پورت را امتحان کرد.

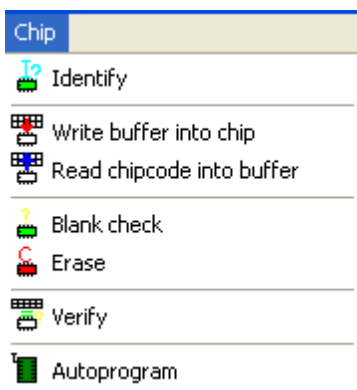


منوی BUFFER

BUFFER CLEAR : با زدن این گزینه اطلاعات موجود بر روی حافظه نرم افزار پروگرامر پاک میشود.

LOAD FROM FILE : با زدن این گزینه شما میتوانید یک فایل هگز یا باینری دیگر را باز کنید و آن را بر روی میکرو پروگرام کنید.

SAVE TO FILE : شما میتوانید محتوای حافظه نرم افزار را در مکان دلخواه با زدن این گزینه ذخیره کنید



منوی CHIP

CHIP IDENTIFY : با زدن این گزینه میکرو متصل به پروگرامر شناسایی میشود.

WRITE BUFFER TO CHIP : با زدن این گزینه محتوای حافظه نرم افزار (کد هگزی که باز کردید

یا بوجود آمده بود) ، در حافظه میکرو ریخته میشود.

READ CLIPCODE INTO BUFFER : با زدن این گزینه کد هگز موجود در حافظه میکرو خوانده میشود و در حافظه نرم افزار قرار میگیرد ، شما میتوانید با زدن گزینه SAVE TO FILE آن را در مکان دلخواه ذخیره کرده و بعداً آن را روی میکرو دیگر بریزید.

BLACK CHECK : با زدن این گزینه پر یا خالی بودن حافظه میکرو مشخص میشود .

ERASE : با زدن این گزینه حافظه میکرو پاک میشود.

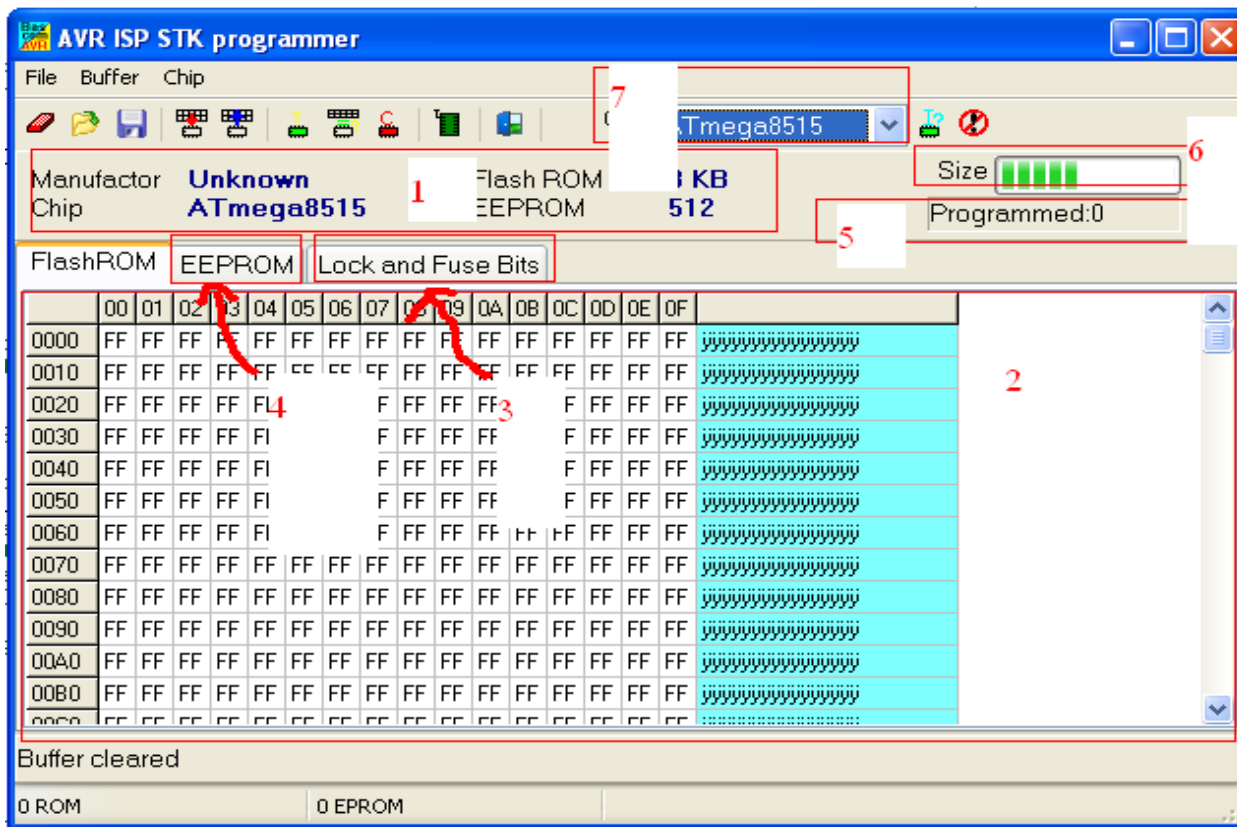
VERIFY : این گزینه کد هگز موجود را با کد ریخته شده در حافظه میکرو مقایسه میکند ، در صورتی که کدها با هم تفاوت داشته باشند پیغامی ظاهر میشود .

AUTO PROGRAM : با زدن این گزینه حافظه میکرو پاک شده و کد هگز در آن ریخته میشود و بعد عمل VERIFY را به صورت خودکار انجام می دهد.

RESET : میکرو متصل به PROGRAMMER را ریست می کند

PEYMAN KHODDAMI

دیگر منوها:



- 1- اطلاعات مربوط به میکرو ، شامل نام و مقدار حافظه ها را نشان میدهد.
 - 2- نمایانگر حافظه نرم افزار و فیوز بیت ها و حافظه میکرو میباشد (با موارد 3 و 4 تغییر میکند)
 - 3- با زدن این گزینه وارد محیط پروگرام کردن فیوز بیت ها میشوید (این محیط در ضمیمه بعدی توضیح داده میشود.
 - 4- با زدن این گزینه محتویات حافظه eeprom نمایش داده میشود.
 - 5- تعداد دفعاتی که با این پروگرامر میکرو را پروگرام کرده اید نمایش میدهد
 - 6- سایز واقعی کد هگز موجود را نشان میدهد ، توجه داشته باشید که سایزی که ویندوز به شما نشان میدهد حجم واقعی نیست .
 - 7- نام میکرو را نشان میدهد ، هنگامی که این پنجره را باز میکنید ، در صورت درست بودن سخت افزار ، نام میکرو به صورت خود کار نمایش داده میشود.
- دیگر گزینه ها در منو ها موجود میباشد ، اگر موس را روی آنها نگه دارید ، نام آنها نمایش داده میشود.

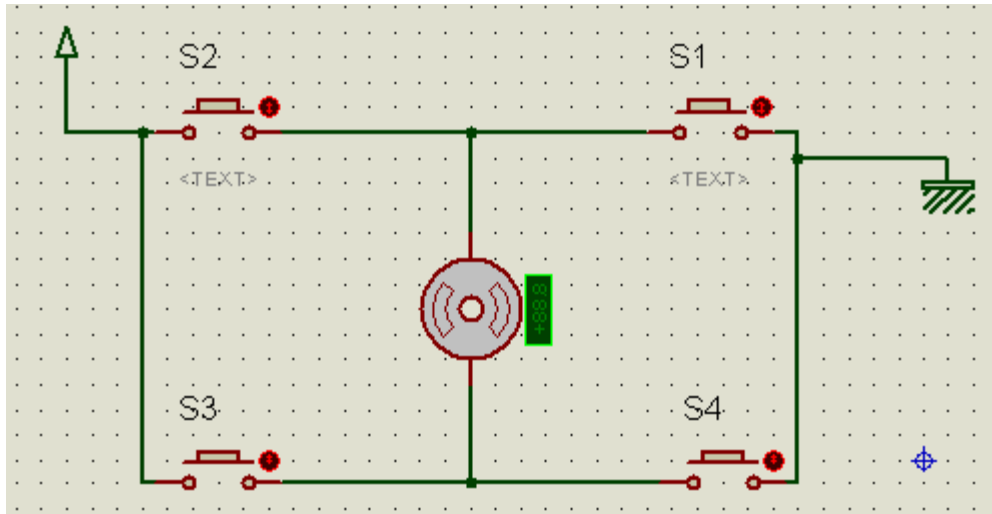
ضمیمه 5 : راه اندازی انواع موتور ها ی dc و پله ای بوسیله ی avr و زبان بیسیک : (برای راه اندازی موتور ها روش های زیادی وجود دارد که در زیر به تعدادی از آنها اشاره میشود)

موتورهای: DC

این نوع موتور از یک آهنربای دائم و چند آهنربای الکتریکی تشکیل شده است ، جریان الکتریکی از طریق دو عدد جاروبک به سیم پیچ ها اعمال میشود و در آنها یک میدان بوجود میآورد ، این میدان باعث گردش موتور میشود. سرعت موتور DC وابسته به ولتاژ و گشتاور آن وابسته به جریان است. معمولاً سرعت توسط ولتاژ متغیر یا عبور جریان و با استفاده از تپ ها (نوعی کلید تغییر دهنده وضعیت سیم پیچ) در سیم پیچی موتور یا با داشتن یک منبع ولتاژ متغیر، کنترل می شود. شما میتوانید جهت گردش ، و سرعت چرخش و مقدار گشتاور موتور را کنترل کنید اما کنترل تعداد دور آن بصورت دقیق کار نشدنی است .

PEYMAN KHODDAMI

کنترل جهت گردش موتور dc :
شما میتوانید با تغییر جهت عبوری از موتور (برعکس کردن قطب ها ، (جای دوسیم موتور را عوض کنید))
جهت گردش آن را تغییر دهید :

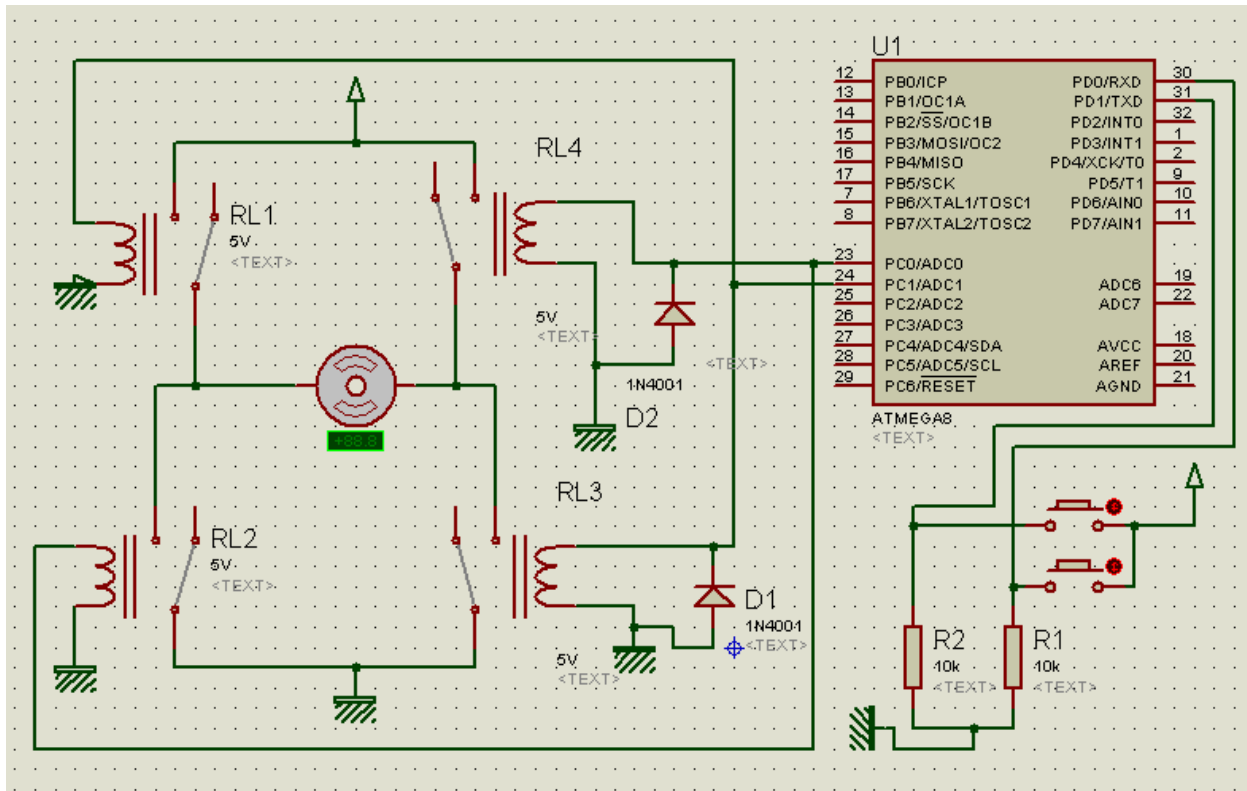


در مدار بالا اگر کلید های s1 و s3 فشرده شوند موتور به سمت راست میچرخد ، و اگر کلید های s2 و s4 فشرده شوند ، موتور به سمت چپ میچرخد. عملیات بالا را توسط میکرو ، و دو کلید انجام میدهیم، برنامه میکرو و مدار را در زیر مشاهده میفرمایید:

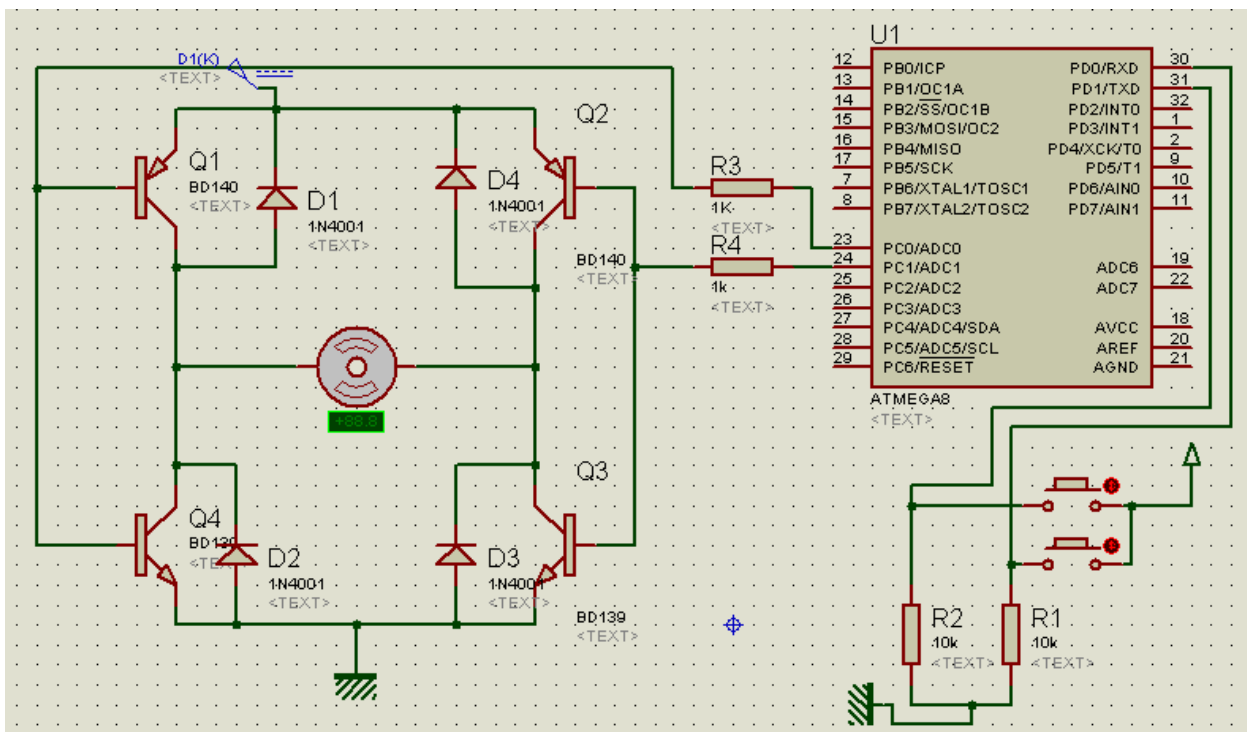
```
$regfile = "M8DEF.DAT" : $crystal = 8000000
Config Portc = Output : Config Portd = Input
Do
Debounce Pind.0 , 1 , Q : Debounce Pind.1 , 1 , W
Loop
End
Q:
Reset Portc.1 : Waitms 100 : Set Portc.0 : Return
W:
Reset Portc.0 : Waitms 100 : Set Portc.1 : Return
```

مدارات سلفی در برابر تغییر جریان عکس العمل نشان میدهند ، این عکس العمل به صورت ولتاژی تا چند برابر تغذیه در مدار ظاهر میشود ، برای اینکه ولتاژ عکس العمل رله ها که به آن ولتاژ القایی میگویند به میکرو آسیب نرساند ، از دو دیود هرزگرد استفاده شده است ، این دو دیود ولتاژ القایی را اتصال کوتاه میکنند . در این حالت اگر ولتاژ تغذیه میکرو از ولتاژ تغذیه ی موتور ورله ها تامین شود ، هنگام وصل شدن رله ها ولتاژ به شدت کم میشود و میکرو ریست میشود ، از روش زیر عملاً چندان استفاده نمیشود مگر در مواردی که موتور مورد استفاده در توان بالا باشد و
در عمل به جای رله از ترانزیستور های bjt یا fet یا ایسی های درایور استفاده میشود که در ادامه به معرفی آنها میپردازیم...

PEYMAN KHODDAMI

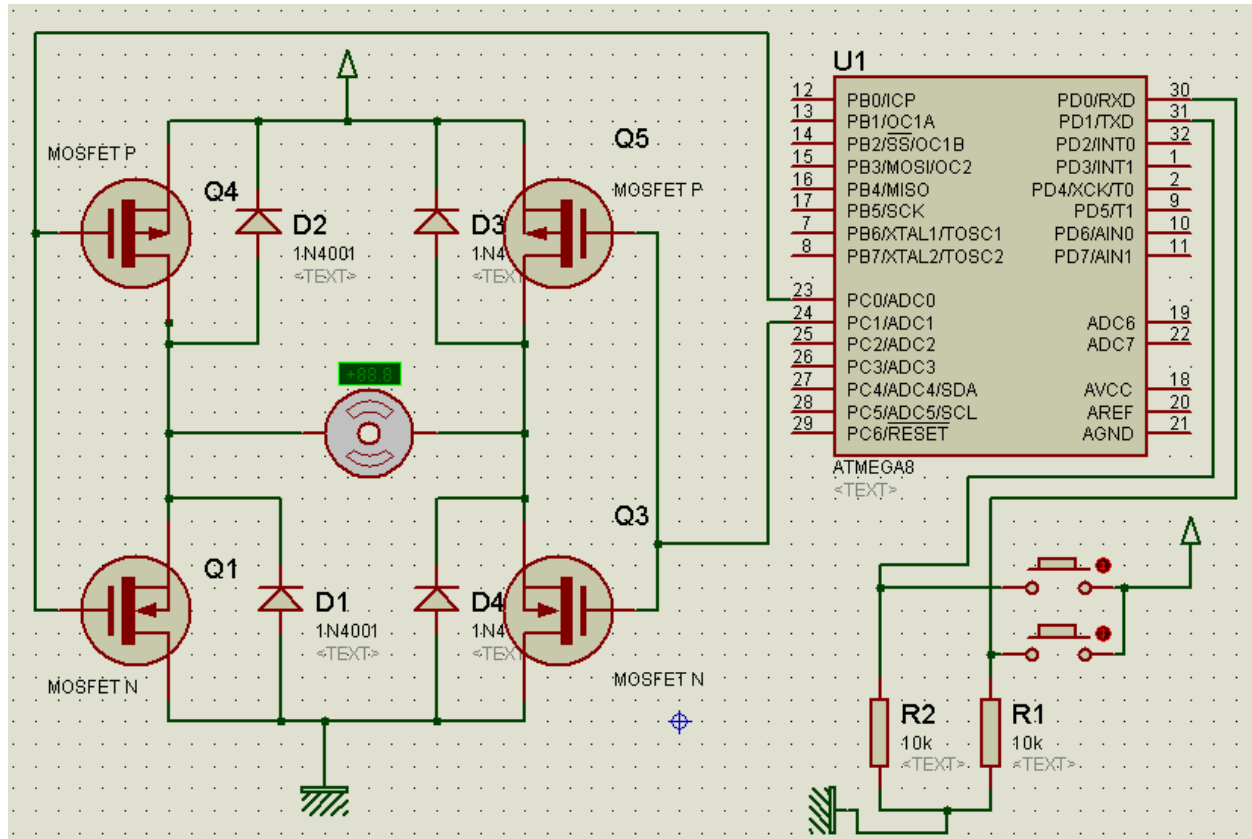


در زیر به جای رله ها از ترانزیستور bjt و fet استفاده کرده ایم ، به این ارایش ترانزیستور پل H گفته میشود:



PEYMAN KHODDAMI

نقش دیود ها حفاظت از ترانزیستور ها در برابر ولتاژ القای موتور میباشد (این ولتاژ القای در حالت قبل هم وجود داشت و بر روی خود موتور افت میکرد) مدار بالا برای راه اندازی موتور های کم قدرت که به جریان کمی نیاز دارند ، مناسب است ، برای راه اندازی موتور های قوی تر شما میتوانید از زوج دارلینگتون مانند 2N3055 و ... استفاده کنید همچنین میتوانید از MOSFET مانند مدار زیر استفاده کنید:



MOSFET ها و ترانزیستورها علاوه بر قیمت کم در توان های گوناگون وجود دارند که شما میتوانید مطابق نیاز خود ان را تهیه کنید . روش دیگر برای راه اندازی موتور های DC استفاده از درایو موتور میباشد ، درایو های مانند L298 و L293 و L293 و ... در زیر توضیحات مربوط به ای سی L298 را مشاهده میفرمایید:

ای ایسی در انواع بسته بندی PowerSo20 و Multiwatt15 تولید و عرضه میشود ، ولتاژ تغذیه ی این ایسی 7 ولت است و میتواند 2 موتور را بصورت هم زمان راه اندازی کند ، بیشترین ولتاژ و جریانی که برای هر موتور فراهم میشود ، به ترتیب 46 و 2 امپر میباشد ، محدوده دمای کار بین -40 تا 150 درجه سانتی گراد است در زیر شماره پایه ها و شکل ایسی را مشاهده میکنید:

PEYMAN KHODDAMI

Multiw att15	PowerSo 20		وظیفه پایه
1;15	2;19	حس کننده a و b	از این دو پایه به عنوان سنسور جریان استفاده میشود ، بین این دو پایه و زمین مقاومت متغیری قرار میگیرد ، بوسیله ی این مقاومت میتوان جریان بار را کنترل کرد (مقدار مقاومت بستگی به دقت کنترل دارد ، در صورتی که کنترل جریان مهم نیست این دو پایه را مطابق شکل با یک مقاومت به زمین متصل کنید.
2;3	4;5	خروجی 1 و 2	این دو پایه خروجی موتور a میباشد ، باید این دو پایه را مستقیماً به موتور a متصل کنید.
4	6	تغذیه موتور	پایه 4 پایه تغذیه موتور ها میباشد ، شما میتونید بسته به نوع موتور ولتاژ بین 1.5 تا 50 ولت را به این پایه اعمال کنید ، برای کاهش نویز بین این پایه و زمین حتماً یک خازن 100nf قرار دهید. (این ولتاژ 1 ولت از تغذیه موتور بیشتر باشد)
5;7	7;9	ورودی 1 و 2	این دو پایه ورودی برای کنترل موتور a میباشد ، این دو پایه از میکرو یا ... گرفته میشود.
6;11	8;14	فعال ساز خروجی a و فعال ساز خروجی b	این دو پایه فعال ساز خروجی های a و b میباشد ، با دادن یک ولتاژ بین 2.3 تا 7 ولت به پایه 6 (8) موتور a فعال میشود ، با صفر کردن پایه موتور غیر فعال میگردد و با دادن یک ولتاژ بین 2.3 تا 7 ولت به پایه 11 (14) موتور b فعال میشود ، با صفر کردن پایه موتور غیر فعال میگردد.
8	1,10,11,20	گراند	پایه های گراند مدار هستند ، گراند تغذیه موتور ها با گراند تغذیه ایسی ، باید به هم متصل شوند.
9	12		این پایه، پایه تغذیه ی ای سی است که 7 ولت میباشد، برای کاهش نویز بین این پایه و زمین حتماً یک خازن 100nf قرار دهید.
10 ; 12	13 ; 15	وردی 3 و 4	این دو پایه ورودی برای کنترل موتور b میباشد ، این دو پایه از میکرو یا ... گرفته میشود.
13;14	16;17	خروجی 3 و 4	این دو پایه خروجی موتور b میباشد ، باید این دو پایه را مستقیماً به موتور b متصل کنید.
-	3;18	بدون اتصال	چیزی متصل نمیشود

PEYMAN KHODDAMI

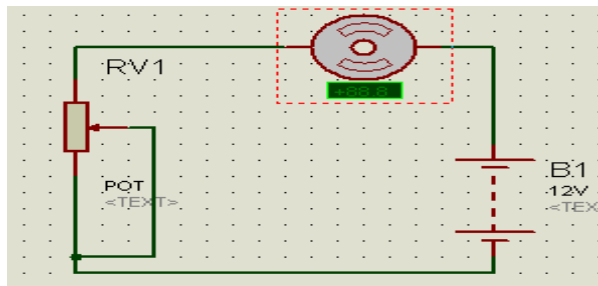
وضعیت پایه ها برای کنترل موتور :

وضعیت پایه 2	وضعیت پایه 3	موتور A	...	وضعیت پایه 1	وضعیت پایه 2	موتور B
0	0	متوقف	...	0	0	متوقف
0	1	راستگرد	...	0	1	راستگرد
1	0	چپگرد	...	1	0	چپگرد
1	1	متوقف	...	1	1	متوقف

منظور از صفر این است که پایه به گرانند متصل شود و منظور از یک وصل شدن پایه به ولتاژی بین 2.3 تا 7 ولت است.

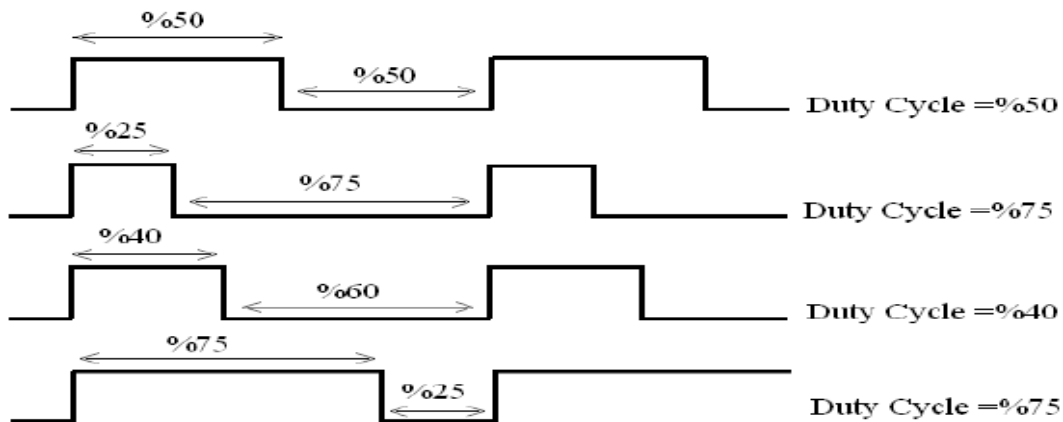
این درایور فاقد دیود های هرزگرد داخلی است ، برای خنثی کردن ولتاژ القایی موتور باید از دیود های خارجی استفاده کنید در غیر اینصورت درایو آسیب میبیند.

کنترل سرعت موتور : ساده ترین راه برای کنترل موتور استفاده از یک مقاومت متغیر میباشد که مانند شکل با موتور سری شده است ، این روش به دلیل داشتن توان خروجی کم کاربرد چندانی ندارد:



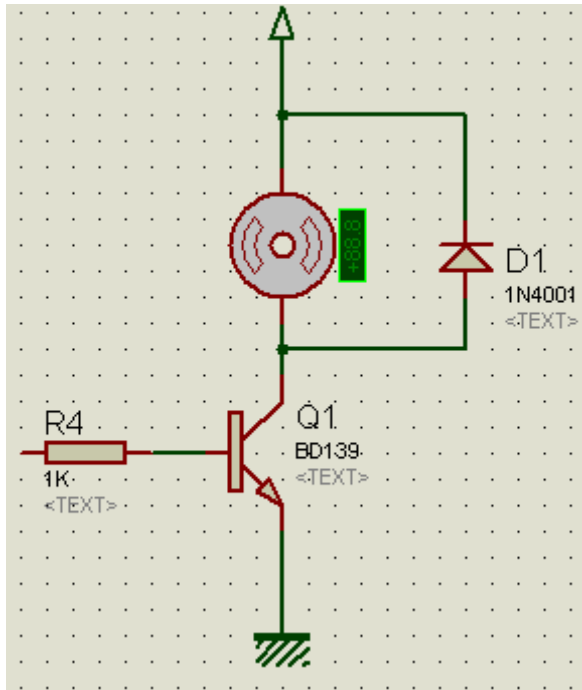
در عمل برای کنترل دور موتور از مدولاسیون عرض پالس (PWM) که در بخش تایمر وکانتر گفته شد استفاده میشود. سرعت موتور تابعی از ولتاژ دو سر آن است ، یعنی هرچه ولتاژ دو سر موتور بیشتر باشد سرعت آن بیشتر است و برعکس.

PWM پالسی است با فرکانس ثابت و پهنای متغیر در زیر نمونه های از پالس PWM را در زیر مشاهده میکنید: شما با تغییر عرض پالس میتوانید توان اعمال شده به بار و سرعت آن را کنترل کنید.



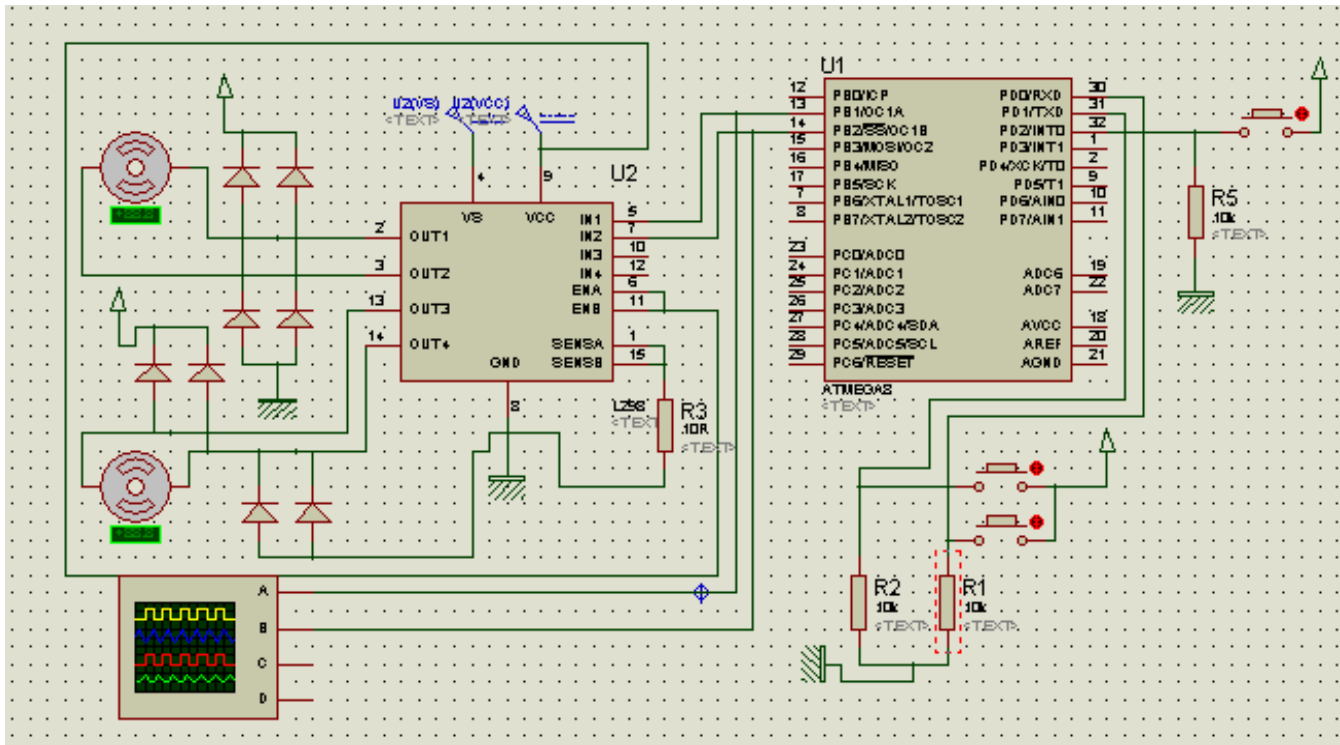
PEYMAN KHODDAMI

در زیر یک مدار ساده برای کنترل سرعت یک موتور DC آورده شده است، نقش ترانزیستور تقویت جریان است :



خروجی پالس PWM از میکرو به مقاومت R4 اعمال میشود. (در اینجا نسبت خاموش به روشن بودن موتور دور را تعیین میکند) شما همچنین میتوانید با استفاده از دو پالس PWM و پل H یا درایو علاوه بر کنترل جهت گردش دور موتور را نیز کنترل کنید. در زیر مدار و برنامه کنترل سرعت و جهت چرخش یک موتور DC را مشاهده میفرمایید: با استفاده از کلید تکی جهت گردش موتور معین میشود ، و با استفاده از دو کلید دیگر ، سرعت ان تغییر میکند.

In1	In2	عملکرد
0	0	ترمز
0	پالس PWM	راستگرد
پالس PWM	0	چپگرد
1	1	ترمز



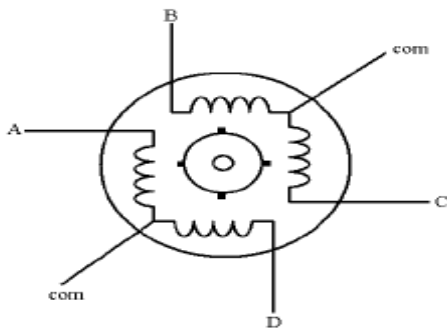
PEYMAN KHODDAMI

برنامه:

```
$regfile = "M8DEF.DAT" : $crystal = 8000000
Config Portb = Output : Config Portd = Input
Config Timer1 = Pwm , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm =
Clear Down , Prescale = 8
Dim A As Word , B As Word
A = 0 : B = 0
Do
If Pind.0 = 1 Then : A=A+10 : B=B+10 : Waitms 100 : End If
If Pind.1 = 1 Then : A = A -10 : B = B -10 : Waitms 100 : End If
If Pind.2 = 1 Then : B = 0 : A = 100 : Else : A = 0 : B = 100 : End If
Pwm1a = A : Pwm1b = B
Loop
End
```

موتورهای پله ای

نوع دیگری از موتورهای الکتریکی موتور پله ای است، که در آن یک روتور درونی، شامل آهنرباهای دائمی توسط یک دسته از آهنرباهای خارجی که به صورت الکترونیکی روشن و خاموش می شوند، کنترل می شود. یک موتور پله ای ترکیبی از یک موتور الکتریکی DC و یک سلونویید است. موتورهای پله ای ساده توسط بخشی از یک سیستم دنده ای در حالت های موقعیتی معینی قرار می گیرند، اما موتورهای پله ای نسبتاً کنترل شده، می توانند بسیار آرام بچرخند. در زیر ساختمان یک موتور پله ای ساده را مشاهده میکنید:



نحوه عمل کرد یک موتور پله ای با موتور DC تفاوت چندانی ندارد، برای راه اندازی این نوع موتور کافی است به ترتیب به سیم پیچ ها ولتاژ دهید، در اکثر موتور ها سیم های گراند از داخل به هم متصل میباشند، جدول زیر شما را در مورد طریقه دادن پالس راه نمایی میکند:

ساختمان یک موتور پله ای ساده

A	B	C	D	جهت موتور	A	B	C	D	جهت موتور
1	0	0	0	در جهت عقربه های ساعت	0	0	0	1	خلاف جهت عقربه های ساعت
0	1	0	0		0	0	1	0	
0	0	1	0	ساعت	0	1	0	0	ساعت
0	0	0	1		1	0	0	0	

در زیر برنامه ای برای راه اندازی یک موتور پله ای 5 سیمه آورده شده است:

```
$regfile = "M8DEF.DAT" : $crystal = 8000000
Config Portb = Output
Do
```

PEYMAN KHODDAMI

```

Portb = &B00000001 : WAITMS 900

Portb = &B00000010 : WAITMS 900

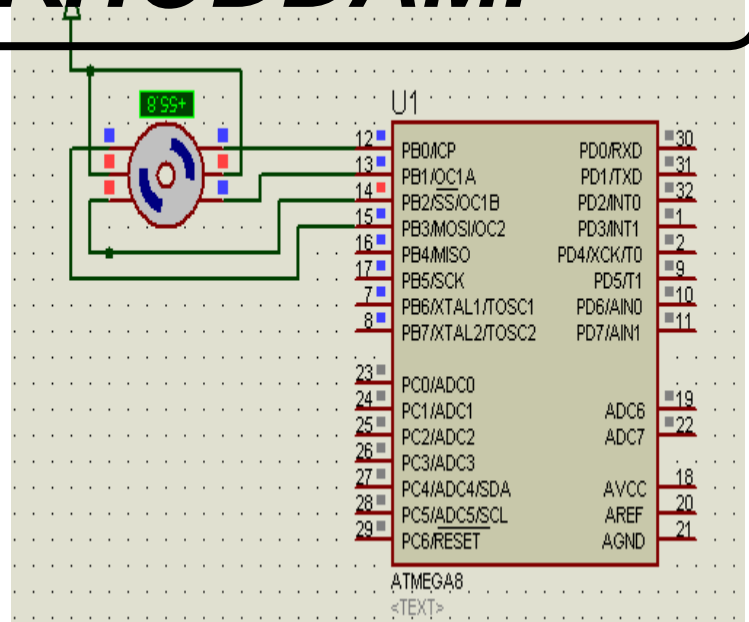
Portb = &B00000100 : WAITMS 900

Portb = &B00001000 : WAITMS 900

Loop

End

```



برای کنترل دقیق تر زاویه حرکت موتور، آن را به صورت نیم پله راه اندازی میکنند، در جدول زیر طریقه پالس دهی را مشاهده میکنید:

A	B	C	D	جهت موتور	A	B	C	D	جهت موتور
0	0	0	1	در خلاف جهت عقربه های ساعت	1	0	0	1	موافق جهت عقربه های ساعت
0	0	1	1		1	0	0	0	
0	0	1	0		1	1	0	0	
0	1	1	0		0	1	0	0	
0	1	0	0		0	1	1	0	
1	1	0	0		0	0	1	0	
1	0	0	0		0	0	1	1	
1	0	0	1		0	0	0	1	

در زیر برنامه برای راه اندازی موتور به صورت نیم پله را مشاهده میکنید:

```

$regfile = "M8DEF.DAT" : $crystal = 8000000

Config Portb = Output

Do

Portb = &B00000001 : Waitms 50

Portb = &B00000011 : Waitms 50

Portb = &B00000010 : Waitms 50

```

PEYMAN KHODDAMI

```

Portb = &B00000110 : Waitms 50

Portb = &B00000100 : Waitms 50

Portb = &B00001100 : Waitms 50

Portb = &B00001000 : Waitms 50

Portb = &B00001001 : Waitms 50

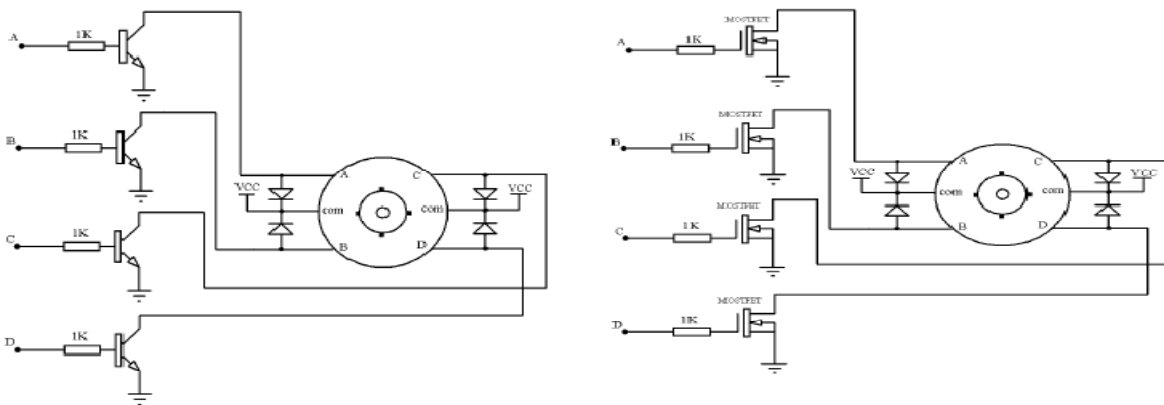
Loop

End

```

تاخیر زمانی سرعت موتور را معین میکند (چقدر طول میکشد تا موتور یک پله حرکت کند) .

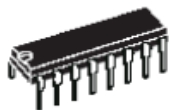
برای راه اندازی موتور های قوی به یک راه انداز نیاز است (حداکثر جریان دهی میکرو 200 میلی امپر است) شما میتوانید از درایو های ULNXXXX یا ترانزیستورهای bjt یا fet استفاده کنید:



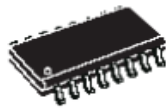
درایوهای ulnxxxx:

این درایوها در ولتاژها و جریان های مختلف تولید میشوند یکی از معروف ترین نوع آنها uln2003 است که در زیر به بررسی آن میپردازیم:

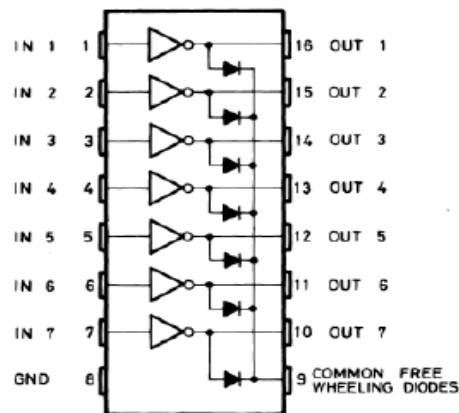
شکل ظاهری:



DIP-16



SO-16
(Narrow)

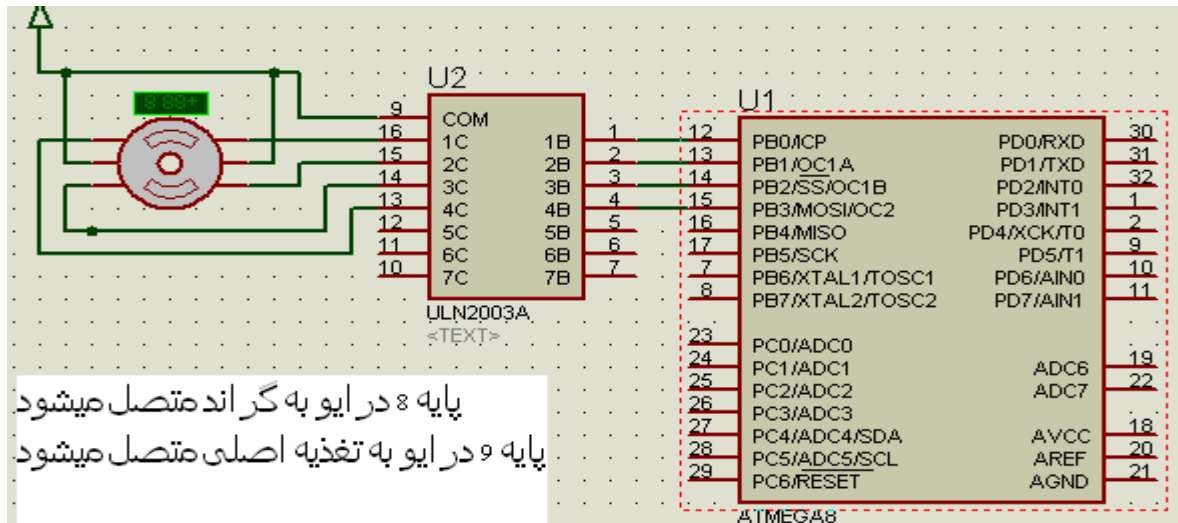


PEYMAN KHODDAMI

مشخصات:

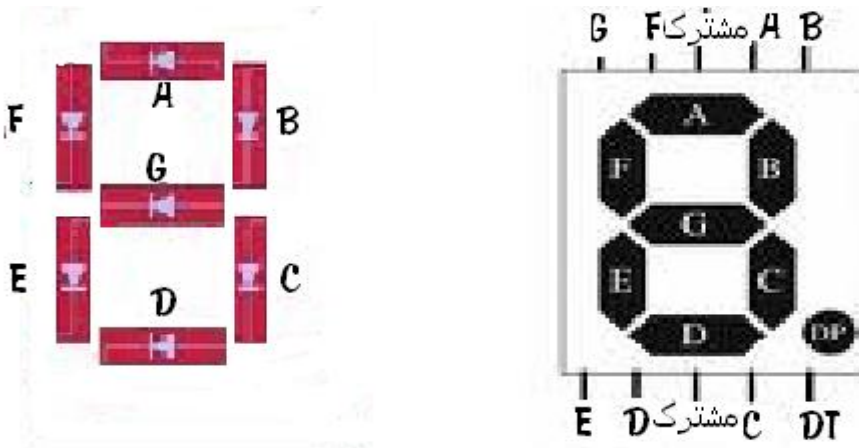
Symbol	Parameter	Value	Unit
V_O	Output voltage	50	V
V_I	Input voltage (for JLN2002A/D - 2003A/D - 2004A/D)	30	V
I_C	Continuous collector current	500	mA
I_B	Continuous base current	25	mA

حداکثر جریان خروجی این درایو 500 میلی امپر میباشد و ولتاژ خروجی آن به میزان ولتاژ تغذیه موجود بین پایه گراند ایسی و سیم مشترک موتور است و میتواند حداکثر 50 ولت باشد ، ولتاژ ورودی آن نیز حداکثر 30 ولت است و ایسی از ورودی جریان 25 میلی امپر را به ازای بیشترین ولتاژ میکشد. در زیر نحوه اتصال میکرو به ایسی و موتور را مشاهده میکنید:



برنامه نیز مانند برنامه قبلی است. (دیگر اطلاعات در مورد درایو ها و موتور را میتانید در دیتاشیت آنها ببینید).

<< 7 سگمت



در روبرو عکس این قطعه را مشاهده میفرمایید :

این قطعات از 8 led تشکیل می شود که 7 تا از آنها نمایشگر هستند و یکی به عنوان ممیز به کار میرود. این led ها از داخل پایه های کاتد یا اند آنها به هم

PEYMAN KHODDAMI

متصل است ، که به نوع اول کاتد مشترک و به نوع دوم اند مشترک گفته میشود .

در زیر می خواهیم عدد 8 را روی 7 سگمنت کاتد نمایش دهیم:

برای نمایش عدد 8 باید همه led ها روشن شوند پس برای 7 سگمنت کاتد مشترک باید آنها را 1 کنیم (5 ولت بدهیم) و برای 7 سگمنت اند مشترک باید آنها را 0 کنیم (زمین) .

مثل همیشه اولین خط برنامه معرفی کریستال و میکرو است :

```
$regfile = "m16def.dat"
```

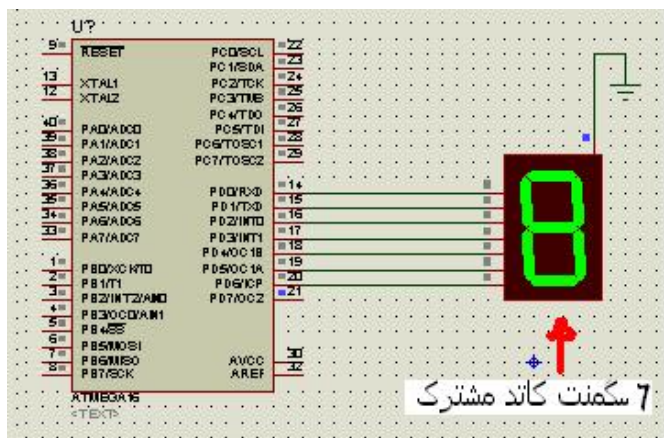
```
$crystal = 8000000
```

در مرحله بعد پورتهی که 7 سگمنت به آن وصل میشود به عنوان خروجی تعریف میگردد(در اینجا 7 سگمنت به پورت d وصل میشود) .

```
config PORTD= OUTPUT
```

مرحله بعد قراردادن کد 7 سگمنت در پورت d است

در شکل زیر نحوه ی بدست آوردن کد برای عدد 8 وجود دارد



7 سگمنت کاتد مشترک

A	B	C	D	E	F	G
1	1	1	1	1	1	1

&B1111111

7 سگمنت اند مشترک

A	B	C	D	E	F	G
0	0	0	0	0	0	0

&B0000000

از اینجا که 7 سگمنت مورد استفاده کاتد مشترک است ، پس :

```
Portd = &B1111111
```

و آخرین خط برنامه دستور END است

PEYMAN KHODDAMI

END

میخواهیم اعداد 0 تا 2 را با تاخیر یک ثانیه روی 7 سگمنت نمایش دهیم ، مانند برنامه قبلی کریستال و میکرو را معرفی میکنیم و پورت D را به عنوان خروجی قرار میدهیم :

```
$regfile = "m16def.dAt"
```

```
$crystal = 8000000
```

```
Config Portd = Output
```

مرحله بعد قرار دادن کد اعداد برای 7 سگمنت است ، اولین عدد 0 است برای عدد صفر تمام LED ها به جز G روشن هستند پس کد 0 برای 7 سگمنت می شود: &B1111110

```
Portd = &B1111110
```

مرحله بعد قرار دادن تاخیر زمانی است تا عدد صفر دیده شود .قرار دادن تاخیر زمانی به مدت 1 ثانیه :

```
Wait 1
```

عدد بعدی 1 است برای عدد 1، LED ها B و C روشن هستند وبقیه خاموش ، پس کد 1 برای 7 سگمنت می شود:
&B0110000

```
Portd = &B0110000
```

مانند دو مرحله قبل ،قرار دادن تاخیر زمانی به مدت 1 ثانیه:

```
Wait 1
```

عدد بعدی 2 است، برای عدد 2، LED ها A , B , G , E , D روشن هستند و بقیه خاموش ، پس کد 2 برای 7 سگمنت می شود:
&B1101101

```
Portd = &B0110000
```

قرار دادن تاخیر زمانی به مدت 1 ثانیه :

```
Wait 1
```

در نهایت برنامه با دستور END به پایان میرسد :

PEYMAN KHODDAMI

End

این برنامه فقط یک بار تکرار میشود در صورتی که شما میخواهید برنامه مدام تکرار شود کافی است یک حلقه در ابتدای برنامه (بعد از دستور Config Portd = Output دستور DO یا دیگر حلقه ها را بنویسید) و پایان حلقه را در انتهای برنامه (قبل از دستور END دستور LOOP یا دیگر حلقه ها را بنویسید) قرار دهید. شما همچنین میتوانید کد اعداد دیگر را نیز به برنامه اضافه کنید.

در پوشه پیوست یک نرم افزار برای ساخت کد 7 سگمنت و جود دارد که میتوانید از آن استفاده کنید.

در ضمن شما میتونید کد های باینری را به هگز تبدیل کنید و کد هگز را روی پورت قرار بدهید. مانند:

```
&b1010101=&h55
```

```
portd=&H55
```

< سگمنت و جدول lookup

در روش اول برنامه طولانی می شود، اما روشهای دیگری نیز وجود دارد، که علاوه بر کاهش حجم برنامه آن را ساده تر میکند، یکی از این روش ها استفاده از جدول lookup هست .

توسط این جدول می توان مقدار دلخواهی را از جدولی برگرداند.

```
var = LOOKUP(value , label)
```

Label برچسب جدول و value اندیس داده دلخواه است . داده برگشتی از جدول در متغیر var قرار می گیرد . value =0 اولین داده در جدول را برمی گرداند و value =n داده nام را از جدول برمیگرداند . تعداد اندیس ها و مقدار داده برگشتی به ترتیب نهایتاً می تواند 255 و 65535 باشد .

میخواهیم عدد 8 را با استفاده از این جدول روی 7 سگمنت نمایش دهیم :

همانند برنامه های قبلی ابتدا میکرو و کریستال معرفی میگردد و پورتهای که 7 سگمنت به آن متصل میشود به عنوان خروجی تعریف می شود .

```
$regfile = "m16def.dAt"
```

```
$crystal = 8000000
```

```
Config Portd = Output
```

بعد از مراحل اولیه با دستور زیر مقدار جدول را در پورت d قرار میدهیم :

PEYMAN KHODDAMI

```
Portd = Lookup(0 , Q
```

```
End
```

```
:Q
```

```
Data &B1111111
```

برنامه شمارنده 0 تا 9 با جدول lookup و چگونگی کار آن

```
$regfile = "m16def.dat "
```

```
$crystal = 8000000
```

```
Config Portd = Output
```

```
Dim Q As Byte
```

```
Do
```

```
Waitms 1
```

```
(Portd = Lookup(q , W
```

```
Incr Q
```

```
Loop Until Q = 9
```

```
End
```

```
:W
```

```
Data &B1111110 , &B0110000 , &B1101101 , &B1111001 , &B0110011 , &B1011011 ,  
&B1011111 , &B1110000 , &B1111111 , &B1111011
```

سه خط اول پیکربندی امکانات است.

در خط چهارم یک متغیر از جنس بایت معرفی میشود .

خط پنجم شروع حلقه do-loop است.

در خط ششم یک تاخیر یک ثانیه اجاد میشود .

در خط هفتم مقدار qم خوانده شده از جدول w در پورت d قرار میگیرید ،در اینجا چون مقدار q=0 است پس اولین عدد

جدول w در پورت d قرار میگیرد (&b1111110) (که بر روی 7سگمت عدد صفررا نشان میدهد) .

PEYMAN KHODDAMI

در خط هشتم یک واحد به متغیر معرفی شده اضافه میگرددد $(0+1=1) (q=q+1)$.

خط نهم پایان حلقه do-loop میباشد که یک شرط نیز در آن به کار برده شده ، این شرط میگوید هرگاه مقدار $q=9$ شد حلقه تمام شود و برنامه از خط بعد از حلقه شروع شود (در اینجا بعد از حلقه پایان برنامه است) .

اما در حالتی که q مخالف 9 است برنامه به خط do پرش میکند در آنجا یک واحد دیگر به q افزوده میشود ($q=2$) و بعد از تاخیر یک ثانیه دومین عدد از جدول lookup در پورت d گذاشته میشود (&b01100000) (که عدد یک را روی 7 سگمنت نمایش میدهد) .

برنامه تا آنجا اجرا میشود که مقدار $q=9$ شده و برنامه پایان یابد .

خط دهم ، خط پایان برنامه است .

خط های یازده و دوازده جدول lookup هستند که مقادیر در آنها قرار میگیرد.

<استفاده از مبدل BCD به کد 7 سگمنت:

برای راه اندازی 7 سگمنت راه دیگری هم وجود دارد که نسبت به دو روش قبلی ساده تر است و ان استفاده از ایسی 7447 است .

ای سی 7447 یک ایسی مبدل کد bcd به کد 7 سگمنت است . برای مثال اگر به وردی این ایسی کد bcd عدد 5 را بدهید، در خروجی کد 7 سگمنت مربوط به عدد 5 ظاهر میشود(در این روش مقدار عدد مستقیماً روی پورت قرار میگیرد)

```
$regfile = "m16def.dat "
```

```
$crystal = 8000000
```

```
Config Portd = Output
```

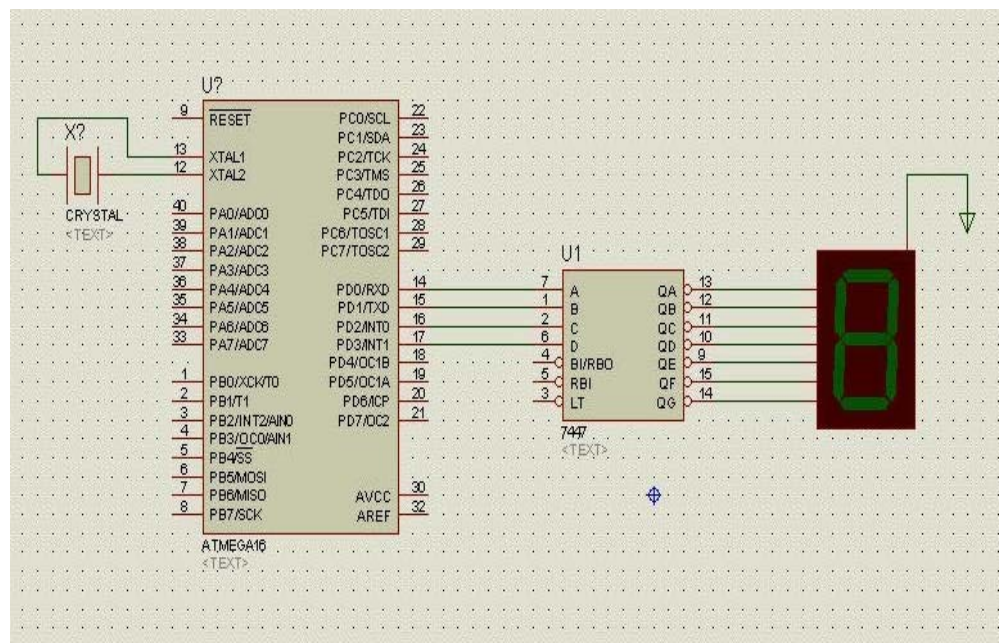
```
Dim Q As Byte
```

```
W :
```

```
Do
```

```
Portd = Q
```

```
Incr Q
```



PEYMAN KHODDAMI

```
Waitms 500  
  
Loop Until Q = 9  
  
Q = 0  
  
jmp w  
  
End
```

نقشه مدار را در بالا ملاحظه میفرمایید:

<<7 سگمنت چند تایی (مولتی پلکس):

در فصل قبلی با روش های راه اندازی 7 سگمنت تکی آشنا شدید ، در این قسمت میخواهیم چند 7 سگمنت را به میکرو متصل کنیم.

فرض کنید میخواهیم عدد 8321 را روی چهار عدد 7 سگمنت نمایش دهیم برای این کار چندین راه وجود دارد:

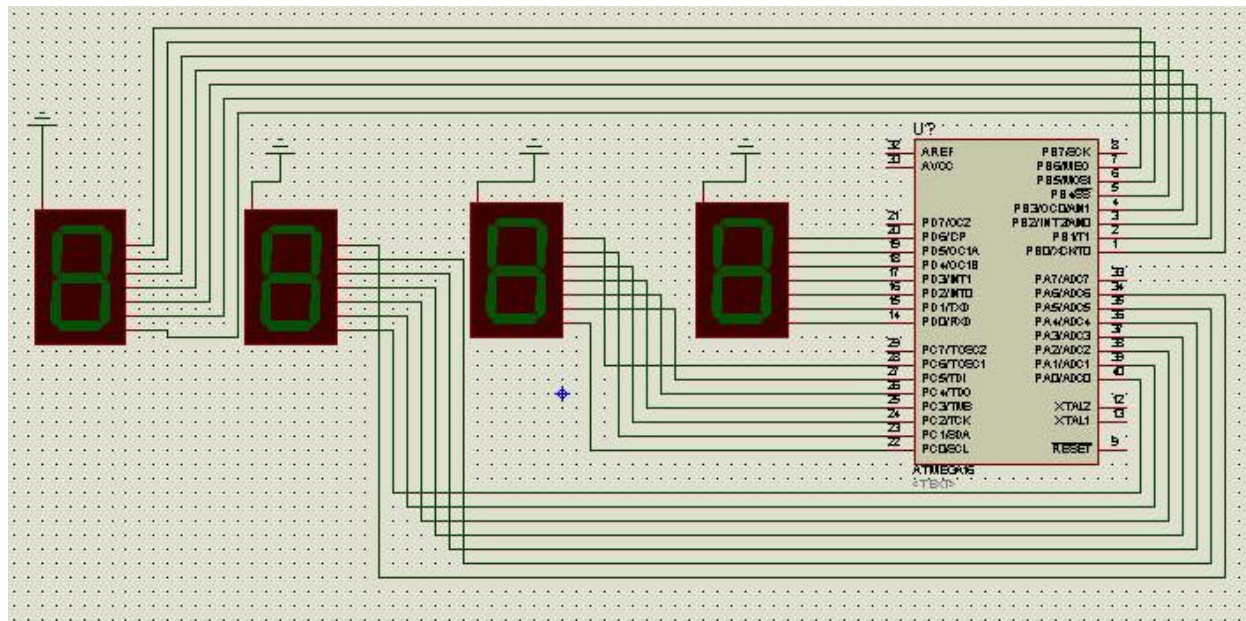
1- چهار 7 سگمنت به طور جداگانه به پورتهای میکرو وصل شود و هر عدد روی هر کدام از آنها نمایش داده شود :

مانند برنامه و مدار زیر :

```
$regfile = "m16def.dat "  
  
$crystal = 8000000  
  
Config Porta = Output  
  
Config Portb = Output  
  
Config Portc = Output  
  
Config Portd = Output  
  
Portd = &B0110000  
  
Portb = &B1111111  
  
Porta = &B1111001  
  
Portc = &B1101101
```

PEYMAN KHODDAMI

End



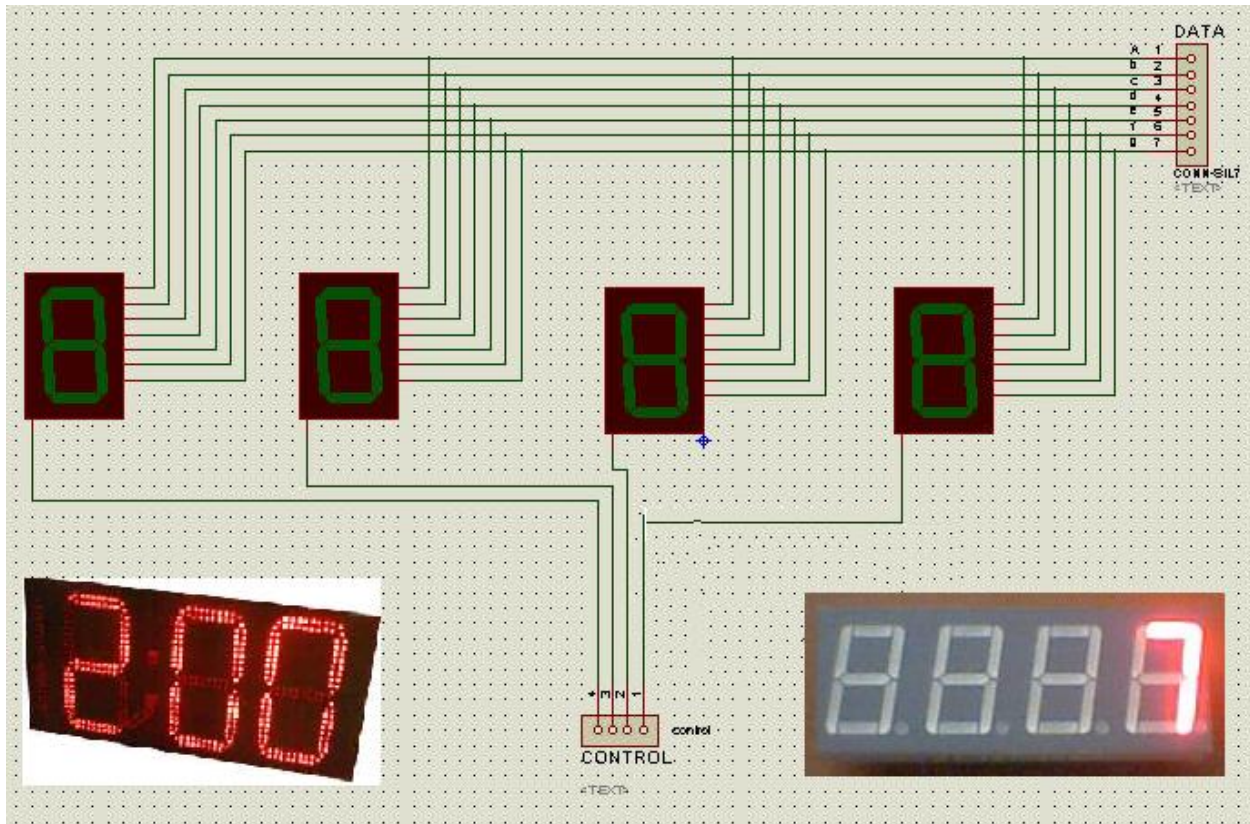
در روش بالا علاوه بر اینکه برای تقویت جریان 7 سگمنت ها نیاز به تقویت کننده میباشد ، کلیه پورت های میکرو اشغال میشود و بیشتر از چهار 7 سگمنت نمیتوان به میکرو متصل کرد ، بنابراین ازان به ندرت استفاده میشود .

<2- روش رفرشی یا تازه سازی:

در این روش خطوط دیتای 7 سگمنتها به هم متصل گردیده و فقط یک پورت برای دیتای 7 سگمنتها(در صورت استفاده از 7447 ، 4 پایه) و n پایه برای کنترل n سون سگمنت مورد استفاده قرار میگیرد .

مانندشکل زیر : (مدار داخلی کلیه ی 7 سگمنت های چند تایی مانند زیر است)

PEYMAN KHODDAMI



این روش براساس خطای چشم انسان کار میکند (در صورتی 25 تصویر(یا بیشتر) پشت سر هم ، در یک ثانیه پخش شود انسان آنها را پیوسته می بیند)، بدین صورت که در هر واحد زمانی فقط یکی از 7سگمنت ها روشن است و کد مربوط به آن ارسال می شود ، این کار با سرعت زیادی انجام می شود ، بطوری که افراد متوجه چشمک زدن اعداد نمیشوند .

برای مثال می خواهیم عدد 8321 رو روی 7 سگمنت مالتی پلکس نشان دهیم :

اول عدد 8 را روی 7سگمنت اول نشان می دهیم ، برای این کار پایه گراند انرا (که به میکرو متصل است)0میکنیم و کد مربوط به عدد 8 برای 7سگمنت را می فرستیم (پایه گراند بقیه 7سگمنتها 1می گردند) .

دوم عدد 3 را روی 7سگمنت دوم نشان می دهیم ، برای این کار پایه گراند انرا (که به میکرو متصل است)0میکنیم و کد مربوط به عدد 3 برای 7سگمنت را می فرستیم (پایه گراند بقیه 7سگمنتها 1می گردند) .

سوم عدد 2 را روی 7سگمنت سوم نشان می دهیم ، برای این کار پایه گراند انرا (که به میکرو متصل است)0میکنیم و کد مربوط به عدد 2 برای 7سگمنت را می فرستیم (پایه گراند بقیه 7سگمنتها 1می گردند).

PEYMAN KHODDAMI

چهارم عدد 1 را روی 7سگمنت چهارم نشان می دهیم ، برای این کار پایه گراند انرا (که به میکرو متصل است) 0میکنیم و کد مربوط به عدد 1 برای 7سگمنت را می فرستیم (پایه گراند بقیه 7سگمنتها 1می گردند) .

و کلیه موارد بالا مدام تکرار می گردند ، مانند برنامه و مدار زیر :

```
"regfile = "m16def.dAt$
```

```
crystal = 8000000$
```

```
Config Portc = Output
```

```
Config Portd = Output
```

```
Do
```

```
Portd = &B0000000
```

```
Waitms 1
```

```
Reset Portc.3
```

```
Reset Portc.2
```

```
Reset Portc.0
```

```
Set Portc.1
```

```
Portd = &B0000110
```

```
Waitms 1
```

```
Reset Portc.0
```

```
Reset Portc.3
```

```
Reset Portc.1
```

```
Set Portc.2
```

```
Portd = &B0010010
```

```
Waitms 1
```

```
Reset Portc.1
```

```
Reset Portc.0
```

PEYMAN KHODDAMI

```
Reset Portc.2
```

```
Set Portc.3
```

```
Portd = &B10011111
```

```
Waitms 1
```

```
Reset Portc.1
```

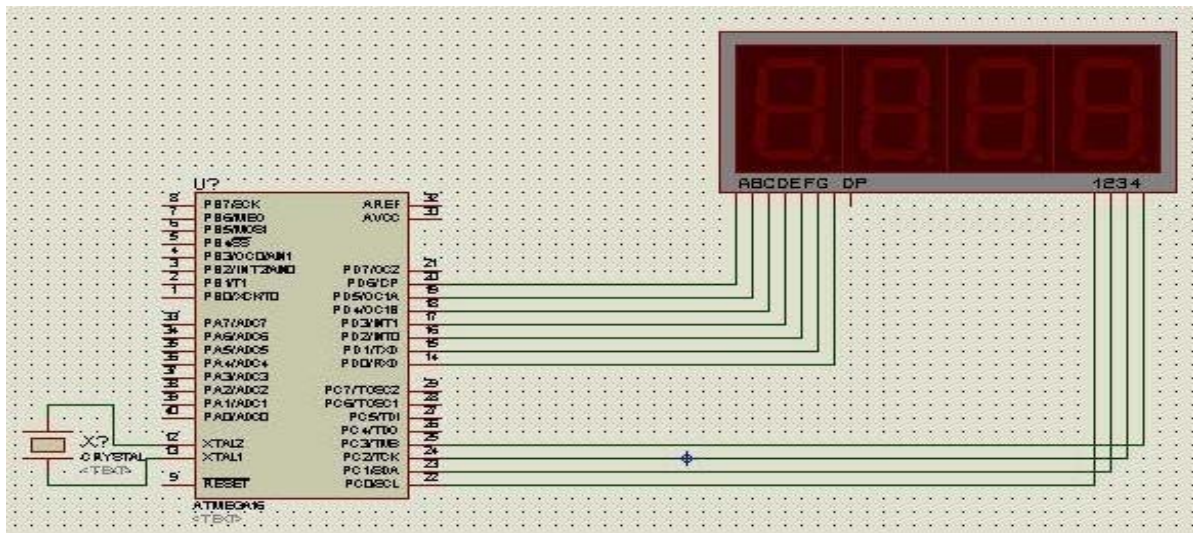
```
Reset Portc.2
```

```
Reset Portc.3
```

```
Set Portc.0
```

```
Loop
```

```
End
```



مانند 7 سگمنت تکی ، میتوانید با استفاده از جدول LOOKUP برنامه را ساده و کمتر کنید:

برای مثال می خواهیم عدد 8321 رو روی 4 عدد 7 سگمنت نشان دهیم ، مانند همه برنامه های خطوط اول معرفی میکرو و کریستال پیکر بندی امکانات و معرفی مغيير ها است

```
$regfile = "m16def.dAt "
```

```
$crystal = 8000000
```

```
Config Portc = Output
```

PEYMAN KHODDAMI

Config Portd = Output

بعد برنامه:

Do

Reset Portc.1

ReSet Portc.2

ReSet Portc.3

Set Portc.0

Portd = Lookup(0 , W (

Waitms 300

ReSet Portc.1

ReSet Portc.3

ReSet Portc.0

Set Portc.1

Portd = Lookup(1 , W (

Waitms 300

ReSet Portc.3

ReSet Portc.2

ReSet Portc.1

Set Portc.2

Portd = Lookup(2 , W (

Waitms 300

ReSet Portc.1

ReSet Portc.3

ReSet Portc.2

PEYMAN KHODDAMI

```
Set Portc.3
```

```
Portd = Lookup(3 , W (
```

```
Waitms 300
```

```
Loop
```

و در اخر پایان برنامه و جدول lookup :

```
End
```

```
:W
```

```
Data &B0000000 , &B0000110 , &B0010010 , &B1001111
```

(مدار این برنامه مانند مدار مثال قبلی میبا شد) .

تحلیل برنامه :

بعد از خط do پین های c.1,c.2,c.3 صفر میشوند و پین c.0 یک می شود تا 7سگمنت اول روشن شود وبقیه خاموش شوند(در انجا از 7سگمنت اند مشترک استفاده شده است) .

بعد از روشن شدن 7 سگمنت اول کد مربوط به عدد هشت (&B0000000) توسط جدول lookup در پورت d گذاشته می شود ،

بعد یک تاخیر زمانی کوتاه (1میلی ثانیه)(در اینجا برای دیده شدن عمل رفرش مقدار تاخیر 300 میلی ثانیه گرفته شده است)

پین های c.1,c.2,c.0 صفر میشوند و پین c.1 یک می شود تا 7سگمنت دوم روشن شود وبقیه خاموش شوند .

بعد از روشن شدن 7 سگمنت دوم کد مربوط به عدد سه (&B0000110)توسط جدول lookup در پورت d گذاشته می شود

و این کار برای اعداد 2 و 1 نیز تکرار میگردد .

وقتی برنامه به خط loop رسید به do پرش میکند این مراحل مدام تکرار میگردد.

<استفاده از ای سی 7447 برای راه انداز 7سگمنت چندتایی :

PEYMAN KHODDAMI

استفاده از ای سی 7447 ساده ترین روش برای راه اندازی 7سگمنتها می باشد، چون علاوه بر این که تعداد پایه های کمتری از میکرو را اشغال میکند، باعث ساده و کمتر شدن برنامه میگردد.

می خواهیم عدد 8321 رو با استفاده از 7447 ری 7سگمنت نمایش بدیم

```
$regfile = "m16def.dAt "
```

```
$crystal = 8000000
```

```
Config Portc = Output
```

```
Config Portd = Output
```

```
Do
```

```
Reset Portc.1
```

```
ReSet Portc.2
```

```
ReSet Portc.3
```

```
Set Portc.0
```

```
Portd = 8
```

```
Waitms 1
```

```
ReSet Portc.1
```

```
ReSet Portc.3
```

```
ReSet Portc.0
```

```
Set Portc.1
```

```
Portd = 3
```

```
Waitms 1
```

```
ReSet Portc.3
```

```
ReSet Portc.2
```

PEYMAN KHODDAMI

ReSet Portc.1

Set Portc.2

Portd = 2

Waitms 1

ReSet

Portc.1

ReSet

Portc.3

ReSet

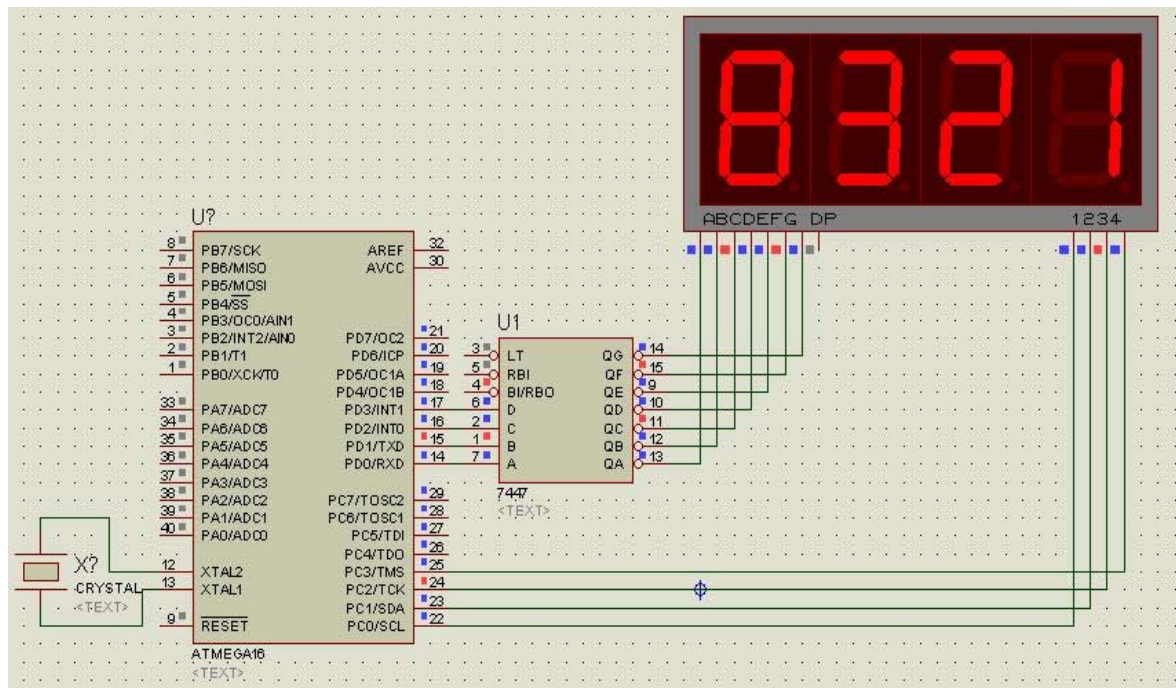
Portc.2

Set Portc.3

Portd = 1

Waitms 1

Loop End



مدار مربوطه را در بالا مشاهده می فرمایید:

همانطور که میبینید استفاده از ایسی 7447 باعث سادگی برنامه شد. شما میتوانید 7 سگمنت های اند مشترک را با ایسی 7448 به سادگی راه اندازی کنید .

ضمیمه 7: خطا های بسکام

در صورتی که برنامه نوشته شده در بسکام دارای خطایی در نگارش یا ... باشد ، پنجره ی زیر باز میشود ، در این پنجره خطا های موجود نمایش داده میشود ، با کلیک بر روی هر خط به خطی که خطا در آن وجود دارد پرش میشود ، در این پنجره خطی که در آن خطا وجود دارد ، شماره خطا و علت آن گفته شده است ، در زیر طریقه رفع خطاها گفته شده است . (بر حسب شماره خطا)

Errors			
Error : 1	Line : 7	Unknown statement [WAITM 50]	In File : C:\DOCUMENTS AND SETTINGS\MN\DESKTOP\NONAME2.BAS
شماره خطا	خطا	علت	محل ذخیره برنامه

PEYMAN KHODDAMI

error	Description
1	Unknown statement
2	Unknown structure EXIT statement
3	WHILE expected
4	No more space for IRAM BIT
5	No more space for BIT
6	. expected in filename
7	IF THEN expected
8	BASIC source file not found
9	Maximum 128 aliases allowed
10	Unknown LCD type
11	INPUT, OUTPUT, 0 or 1 expected
12	Unknown CONFIG parameter
13	CONST already specified
14	Only IRAM bytes supported
15	Wrong data type
16	Unknown Definition
17	9 parameters expected
18	BIT only allowed with IRAM or SRAM
19	STRING length expected (DIM S AS STRING * 12 ,for example)
20	Unknown DATA TYPE
21	Out of IRAM space
22	Out of SRAM space
23	Out of XRAM space
24	Out of EPROM space
25	Variable already dimensioned
26	AS expected
27	parameter expected
28	IF THEN expected
29	SELECT CASE expected
30	BIT's are GLOBAL and can not be erased
31	Invalid data type
32	Variable not dimensioned
33	GLOBAL variable can not be ERASED
34	Invalid number of parameters
35	3 parameters expected
36	THEN expected
37	Invalid comparison operator
38	Operation not possible on BITS
39	FOR expected
40	Variable can not be used with RESET
41	Variable can not be used with SET
42	Numeric parameter expected
43	File not found
44	2 variables expected
45	DO expected
46	Assignment error
47	UNTIL expected
50	Value doesn't fit into INTEGER
51	Value doesn't fit into WORD
52	Value doesn't fit into LONG

PEYMAN KHODDAMI

60	Duplicate label
61	Label not found
62	SUB or FUNCTION expected first
63	Integer or Long expected for ABS()
64	, expected
65	device was not OPEN
66	device already OPENED
68	channel expected
70	BAUD rate not possible
71	Different parameter type passed then declared
72	Getclass error. This is an internal error.
73	Printing this FUNCTION not yet supported
74	3 parameters expected
80	Code does not fit into target chip
81	Use HEX(var) instead of PRINTHEX
82	Use HEX(var) instead of LCDHEX
85	Unknown interrupt source
86	Invalid parameter for TIMER configuration
87	ALIAS already used
88	0 or 1 expected
89	Out of range : must be 1-4
90	Address out of bounds
91	INPUT, OUTPUT, BINARY, or RANDOM expected
92	LEFT or RIGHT expected
93	Variable not dimensioned
94	Too many bits specified
95	Falling or rising expected for edge
96	Prescale value must be 1,8,64,256 or 1024
97	SUB or FUNCTION must be DECLARED first
98	SET or RESET expected
99	TYPE expected
100	No array support for IRAM variables
101	Can't find HW-register
102	Error in internal routine
103	= expected
104	LoadReg error
105	StoreBit error
106	Unknown register
107	LoadnumValue error
108	Unknown directive in device file
109	= expected in include file for .EQU
110	Include file not found
111	SUB or FUNCTION not DECLARED
112	SUB/FUNCTION name expected
113	SUB/FUNCTION already DECLARED
114	LOCAL only allowed in SUB or FUNCTION
115	#channel expected
116	Invalid register file
117	Unknown interrupt
200	.DEF not found
201	Low Pointer register expected

PEYMAN KHODDAMI

202	.EQU not found, probably using functions that are not supported by the selected chip
203	Error in LD or LDD statement
204	Error in ST or STD statement
205	} expected
206	Library file not found
207	Library file already registered
210	Bit definition not found
211	External routine not found
212	LOW LEVEL, RISING or FALLING expected
213	String expected for assignment
214	Size of XRAM string 0
215	Unknown ASM mnemonic
216	CONST not defined
217	No arrays allowed with BIT/BOOLEAN data type
218	Register must be in range from R16-R31
219	INT0-INT3 are always low level triggered in the MEGA
220	Forward jump out of range
221	Backward jump out of range
222	Illegal character
223	* expected
224	Index out of range
225	() may not be used with constants
226	Numeric of string constant expected
227	SRAM start greater than SRAM end
228	DATA line must be placed after the END statement
229	End Sub or End Function expected
230	You can not write to a PIN register
231	TO expected
232	Not supported for the selected micro
233	READ only works for normal DATA lines, not for EPROM data
234) block comment expected first
235	(block comment expected first
236	Value does not fit into byte
238	Variable is not dimensioned as an array
239	Invalid code sequence because of AVR hardware bug
240	END FUNCTION expected
241	END SUB expected
242	Source variable does not match the target variable
243	Bit index out of range for supplied data type
244	Do not use the Y pointer
245	No arrays supported with IRAM variable
246	No more room for .DEF definitions
247	. expected
248	BYVAL should be used in declaration
249	ISR already defined
250	GOSUB expected
251	Label must be named SECTIC
252	Integer or Word expected
253	ERAM variable can not be used
254	Variable expected
255	Z or Z+ expected

PEYMAN KHODDAMI

256	Single expected
257	"" expected
258	SRAM string expected
259	- not allowed for a byte
260	Value larger than string length
261	Array expected
262	ON or OFF expected
263	Array index out of range
264	Use ECHO OFF and ECHO ON instead
265	offset expected in LDD or STD like Z+1
266	TIMER0, TIMER1 or TIMER2 expected
267	Numeric constant expected
268	Param must be in range from 0-3
269	END SELECT expected
270	Address already occupied
322	Data type not supported with statement
323	Label too long
324	Chip not supported by I2C slave library
325	Pre-scale value must be 1,8,32,128,256 or 1024
326	#ENDIF expected
327	Maximum size is 255
328	Not valid for SW UART
329	FileDateTime can only be assigned to a variable
330	Maximum value for OUT is &H3F
332	\$END ASM expected
334) blockcomment end expected
335	Use before DIM statements
336	Could not set specified CLOCK value
999	DEMO/BETA only supports 4096 bytes of code
9999	I hope you do not see this one.

ضمیمه 8 : کلید های میانبر در بسکام : با استفاده از این کلید ها میتوانید اعمال موجود را با استفاده از کلید های کیبرد انجام دهید ، مثلا برای کامپایل کردن برنامه باید به منوی program بروید و گزینه compile را بزنید ، این کار را میتوانید با زدن کلید F7 انجام دهید ، در زیر تمام کلید های میانبر آورده شده است:

Key	Action
LEFT ARROW	One character to the left
RIGHT ARROW	One character to the right
UP ARROW	One line up
DOWN ARROW	One line down
HOME	To the beginning of a line
END	To the end of a line
PAGE UP	Up one window
PAGE DOWN	Down one window
CTRL+LEFT	One word to the left
CTRL+RIGHT	One word to the right
CTRL+HOME	To the start of the text

PEYMAN KHODDAMI

CTRL+END	To the end of the text
CTRL+ Y	Delete current line
INS	Toggles insert/over strike mode
F1	Help (context sensitive)
F2	Run simulator
F3	Find next text
F4	Send to chip (run flash programmer)
F5	Run
F7	Compile File
F8	Step
F9	Set breakpoint
F10	Run to
CTRL+F7	Syntax Check
CTRL+F	Find text
CTRL+G	Go to line
CTRL+K+x	Toggle bookmark. X can be 1-8
CTRL+L	LCD Designer
CTRL+M	File Simulation
CTRL+N	New File
CTRL+O	Load File
CTRL+P	Print File
CTRL+Q+x	Go to Bookmark. X can be 1-8
CTRL+R	Replace text
CTRL+S	Save File
CTRL+T	Terminal emulator
CTRL+P	Compiler Options
CTRL+W	Show result of compilation
CTRL+X	Cut selected text to clipboard
CTRL+Z	Undo last modification
SHIFT+CTRL+Z	Redo last undo
CTRL+INS	Copy selected text to clipboard
SHIFT+INS	Copy text from clipboard to editor
CTRL+SHIFT+J	Indent Block
CTRL+SHIFT+U	Unindent Block
Select text	Hold the SHIFT key down and use the cursor keys to select text. or keep the left mouse key pressed and drag the cursor over the text to select.

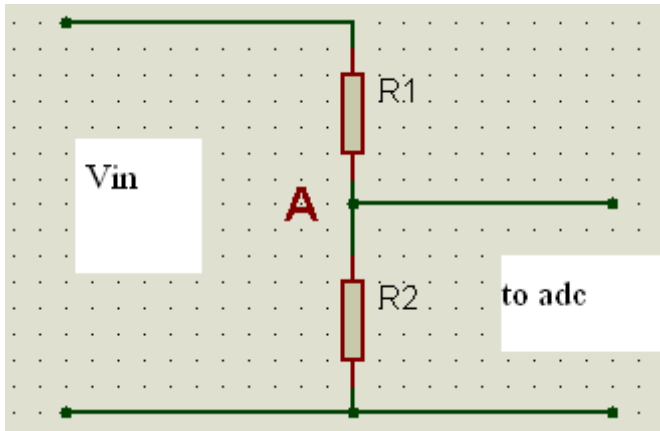
ضمیمه 9 : اندازه گیری ولتاژ های منفی زیاد مثبت و... با ADC (کار با OP-AMP):

مبدل آنالوگ به دیجیتال میکرو توانای اندازه گیری ولتاژ بین 0 تا 5 ولت را دارد ، این مبدل این محدوده ولتاژ را با دقت بالا اندازه میگیرد اما توانایی اندازه گیری ولتاژ های بیشتر یا کمتر از این مقدار را ندارد ، همچنین دقت در اندازه گیری ولتاژ های کمتر از 1 ولت (ولتاژ های اعشاری و خیلی کم) بسیار پایین است ، در زیر روش اندازه گیری هریک گفته میشود.

اندازه گیری ولتاژ های بیشتر از 5 ولت:

PEYMAN KHODDAMI

شما باید ولتاژ مذکور را تضعیف کنید و آن را به سطح 5 ولت برسانید برای این کار می‌توانید از op-amp یا تقسیم ولتاژ مقاومتی استفاده کنید:



در این روش ولتاژ نقطه ی a (سر مشترک دو مقاومت)

از فرمول زیر بدست می‌آید:

در تبدیل آنالوگ به دیجیتال ولتاژ نقطه ی A پنج ولت است و

Vin بیشترن ولتاژ ورودی میباشد ، بنابراین با داشتن یکی

از مقاومت ها مقاومت دیگر به سادگی از فرمول زیر بدست می‌آید :

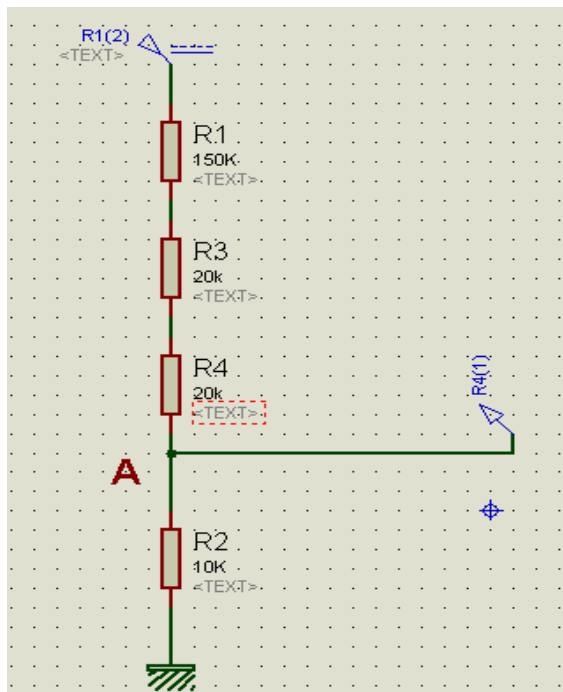
با داشتن R2:

$$R1 = \frac{R2(Vin - Va)}{Va}$$

با داشتن R1:

$$R2 = \frac{Va * R1}{(Vin - Va)}$$

مثال: میخواهیم ولتاژ 100 ولت را با مبدل آنالوگ به دیجیتال اندازه بگیریم ، اولین قدم محاسبه دو مقاومت برای تبدیل 100 ولت به 5 ولت است ، ابتدا با استفاده از دو فرمول بالا مقادیر مقاومت مجهول را بدست می‌آوریم (مقدار یکی از مقاومت ها را در نظر می‌گیریم و دیگری را محاسبه میکنیم ، مثلا مقدار R2 را 10 کیلو در نظر می‌گیریم و مقدار R1 را طبق فرمول بدست می‌آوریم:



$$R1 = \frac{10(100 - 5)}{5} = 190K$$

مدار تقسیم ولتاژ را در زیر مشاهده میکنید:

ولتاژ نقطه ی A نباید از 5 ولت بیشتر شود ، کمتر شدن از 5 ولت را میتوان در محاسبات لحاظ کرد ، ولتاژ بیشتر از 5 ولت

ADC را میسوزاند.

دومین قدم نوشتن برنامه است:

برنامه زیر مقدار آنالوگ موجود بر روی پایه ی adc0 را

PEYMAN KHODDAMI

به دیجیتال تبدیل میکنید ، همانطور که در عمل یا شبیه سازی مشاهده میفرمایید ، به ازای ولتاژ صفر ولت ، بر روی lcd عدد 0 و به ازای ولتاژ 5 ولت عدد 1023 نمایش داده میشود ، (با نوشتن دستور \$sim در انتهای برنامه میتوانی ان را توسط شبیه ساز بسکام شبیه سازی کنید ، نحوه شبیه سازی در ضمیمه های قبلی گفته شد):

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =  
Pinc.4 , Rs = Pind.3 , E = Pind.2
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word
```

```
Do
```

```
A = Getadc(0(
```

```
Locate 1 , 1 : Lcd A
```

```
Loop
```

```
End
```

کار ما یک اشکال دارد ، ما ولتاژ 100 ولت داریم ، و باید به ازای 100 ولت ورودی که ان را با مقاومت به 5 ولت کاهش دادیم ، عدد 100 روی lcd نمایش داده شود ، اما اکنون 1023 نوشته میشود ، شما با استفاده از فرمول زیر میتوانید مقدار مناسب با ولتاژ ورودی را بدست اورید (کلا خاصیت adc این است که به ازای ولتاژ بین صفر تا 5 ولت مقدار 0 تا 1023 را در متغیر مورد نظر (متغیری که با دستور getadc مقدار دیجیتال در ان ریخته میشود) میریزد)

$$Vin = \frac{1023}{x} \Rightarrow x = 1023/Vin$$

اگر عدد 1023 را بر x تقسیم کنیم ، همه چیز حله ، برنامه اصلاح شده را در زیر مشاهده میفرمایید:

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =  
Pinc.4 , Rs = Pind.3 , E = Pind.2
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word
```

```
Do
```

```
A = Getadc(0(
```

PEYMAN KHODDAMI

```
Locate 1 , 1 : Lcd A
```

```
A = A / 10.23
```

```
Locate 2 , 1 : Lcd A
```

```
Loop
```

```
End
```

در برنامه بالا ولتاژ ورودی 100 ولت است ، من 1023 را به 100 تقسیم کردم ، بعد در برنامه 1023 را به عدد بدست آمده تقسیم کردم ($A = A / 10.23$) . موقع شبیه ساری مشاهده میکنید که به ازای ولتاژ 5 ولت بر روی پایه ی **adc** مقدار 102 روی **lcd** نمایش داده میشود . این موزد به این دلیل است که متغیر **a** از جنس **word** است ، **word** اعداد صحیح بین 0 تا 65535 را شامل میشود و فقط میتوان در اعداد صحیح ضرب یا به اعداد صحیح تقسیم شود ، بنابراین در برنامه عدد 10.23 به 10 گرد میشود و سپس متغیر **a** به 10 تقسیم میشود ، در برنامه زیر این مشکل نیز رفع شده است:

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =  
Pinc.4 , Rs = Pind.3 , E = Pind.2
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word , B As Single
```

```
Config Single = Scientific , Digits = 1
```

```
Do
```

```
A = Getadc(0(
```

```
Locate 1 , 1 : Lcd A
```

```
B = A : B = B / 10.23
```

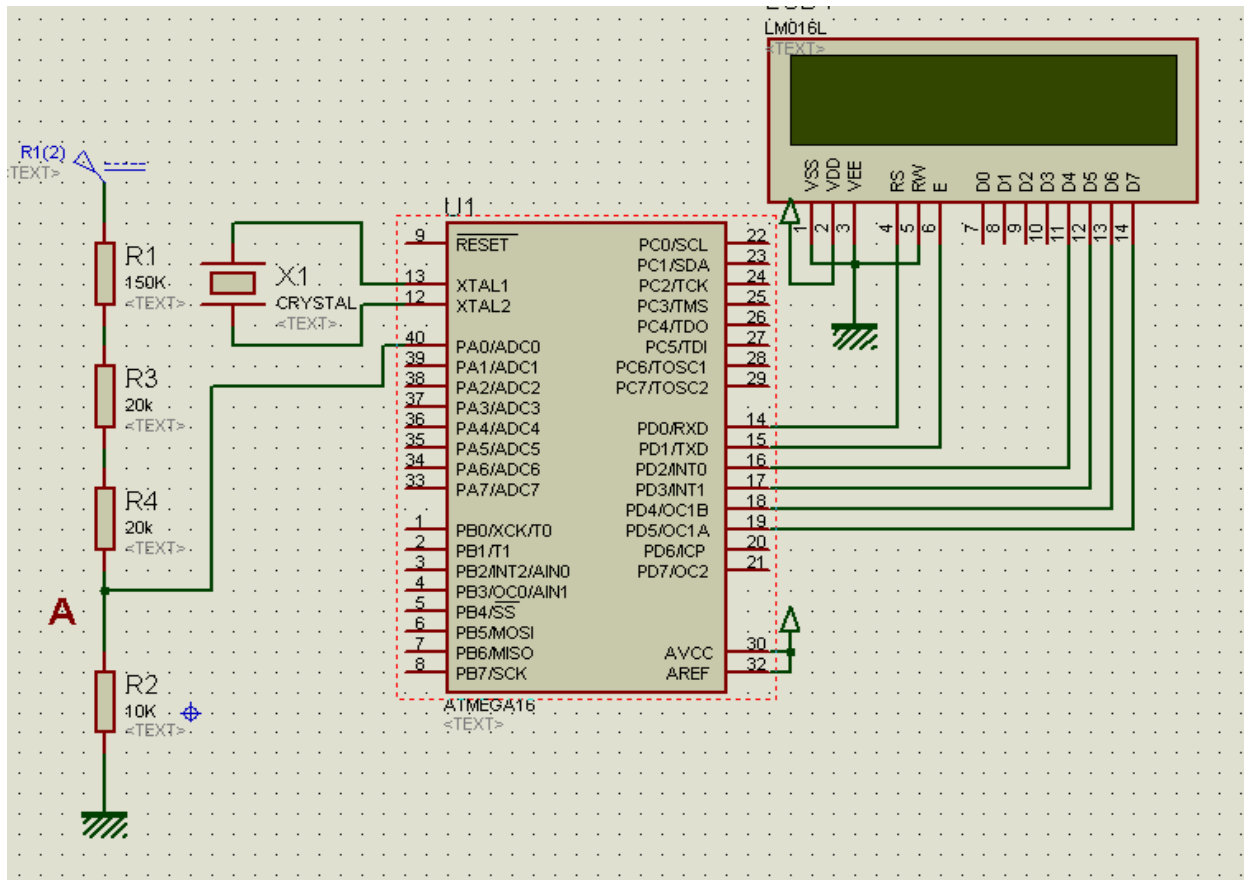
```
Locate 2 , 1 : Lcd B
```

```
Loop
```

```
End
```

همانطور که مشاهده میفرمایید مشکل حل شد. در برنامه بالا مقدار متغیر **a** در متغیر **b** که از جنس **single** است و اعداد اعشاری را نیز شامل میشود ، ریخته شده است، سپس این متغیر بر عدد 10.23 تقسیم شده و جواب روی **lcd** نمایش داده شده است. دستور `Config Single = Scientific , Digits = 1` ، مقدار رقم اعشار متغیر سینگل را تعیین میکند که در اینجا 1 است. مدار مربوط به مثال های بالا را مشاهده میفرمایید:

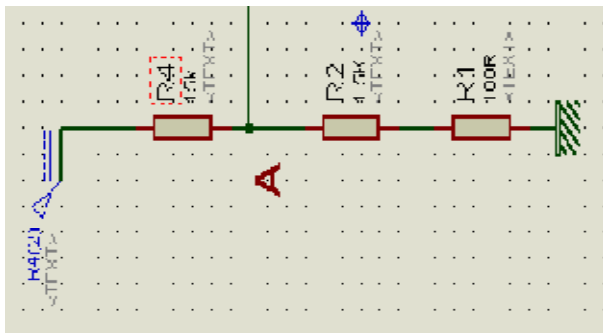
PEYMAN KHODDAMI



مثال : یک منبع تغذیه ی 0 تا 50 ولت داریم و میخوایم ولتاژ خروجی ان را تا دو رقم اعشار اندازه بگیریم ، در زیر ان را طراحی میکنیم:

اولین قدم محاسبه ی R1 و R2 است ، مقدار R1 را 15 کیلو در نظر میگیریم و R2 را محاسبه میکنیم:

$$R2 = \frac{V_a * R1}{(V_{in} - V_a)} \Rightarrow R2 = \frac{5 * 15}{(50 - 5)} = 1.66K$$



بیشترین ولتاژ ورودی 50 ولت است ، بنابراین به جای 1023 باید عدد 50.00 نمایش داده شود ، بنابراین با استفاده از فرمول بالا ، در برنامه عدد

PEYMAN KHODDAMI

1023 باید به 20.46 تقسیم شود ، همچنین برای دو رقم اعشار باید یک متغیر از جنس SINGLE با حداکثر رقم اعشار 2 پیش بینی شود ، برنامه مذکور را در زیر مشاهده میفرمایید:

```
$regfile = "m16def.dat" : $crystal = 8000000

Config Lcd = 16 * 2

Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =
Pinc.4 , Rs = Pind.3 , E = Pind.2

Config Adc = Single , Prescaler = Auto

Dim A As Word , B As Single

Config Single = Scientific , Digits = 2

Do

A = Getadc(0(

Locate 1 , 1 : Lcd A" " ;

B = A : B = B / 20.46

Locate 2 , 1 : Lcd B" " ;

Loop

End
```

تنها مشکلی که حتما به اون پی بردید ، این است که در تقسیم ولتاژ مقاومتی ، هنگام استفاده از مقاومت های استاندارد ، ولتاژ 4.81 به پایه ADC میرسد برای حل این مشکل باید دید که به ازای 4.8 ولت چه مقداری در متغیر A ریخته میشود (5ولت 1024 حالا 4.81 ولت چقدر ؟) این مقدار را میتوانید با رابطه ی ساده زیر بدست آورید:

$$OUT = \frac{1023}{5} * X$$

X ولتاژ خروجی تقسیم ولتاژ است و OUT هم مقدار دیجتالی ریخته شده در متغیر میباشد ، حال بقیه محاسبات را با OUT انجام دهید:

$$Vin = \frac{OUT}{x} \Rightarrow x = OUT/Vin$$

در برنامه بالا اگر به جای 20.46 عدد 19.68252 را قرار دهید مقدار نمایش داده شده روی LCD دقیقاً برابر برآوردی میشود.

مثال : ولتاژ ورودی بین 0 تا 3.5 ولت را با دقت یک رقم اعشار اندازه بگیرید:

چون ولتاژ ورودی کمتر از 5 ولت است نیازی به تقسیم ولتاژ نیست و ان را مستقیم به ADC اعمال میکنیم.

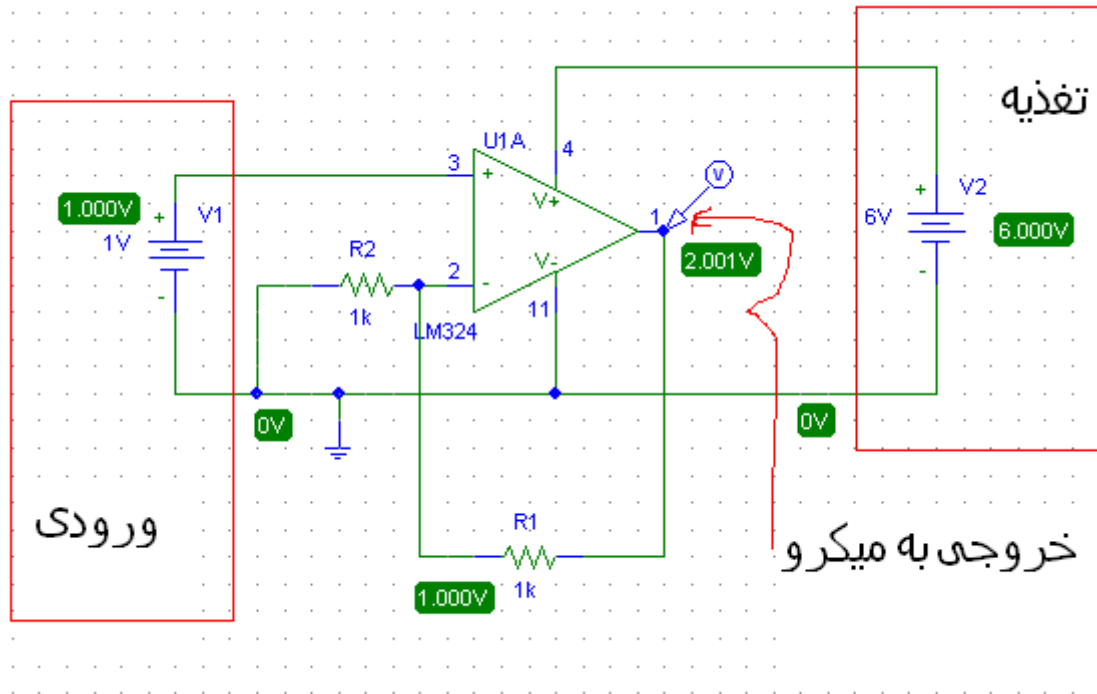
PEYMAN KHODDAMI

در ولتاژ های زیر 5 ولت (هر چقدر بود) بیشترین ولتاژ را 5 ولت در نظر بگیرید و محاسبات را انجام دهید (فرض کنید ولتاژ ورودی 0 تا 5 ولت است و شما ان را تا 3.5 زیاد میکنید)

در برنامه بالا اگر به جای $B = B / 20.46$ مقدار $B = B / 204.6$ را قرار دهید ، برنامه درست میشود...

اندازه گیری ولتاژ های زیر 1 ولت :

برای اندازه گیری این ولتاژ ها ، شما باید ابتدا ان را تقویت کنید و ان را به 5 ولت برسانید ، برای این کار میتوانید از تقویت کننده های تفاضلی (OP-AMP) استفاده کنید:



مدار بالا یک تقویت کننده غیر معکوس است ، که در زیر روابط ان را مشاهده میفرمایید:

$$V_o = R_1 * \frac{V_{in}}{R_2} + V_{in}$$

$$I_{in} = \frac{V_i}{R_2}$$

$$A_v = 1 + \frac{R_1}{R_2}$$

$$A_v = \frac{V_o}{V_{in}}$$

PEYMAN KHODDAMI

ولتاژ خروجی شما باید 5 ولت باشد بنابراین با داشتن ولتاژ ورودی و خروجی ویکی از مقاومت ها میتوانید به سادگی مقاومت دیگر را محاسبه کنید و ولتاژ وردی را به 5 ولت برسانید .

بدست آوردن R1 با معلوم بودن R2:

$$R1 = (AV * R2)$$

مثال ، میخواهیم ولتاژ 0 تا 100 میلی ولت را اندازه بگیریم ، مدار مناسب را طراحی کنید:

با فرض اینکه R2 یک کیلو اهم باشد(ان را 1 کیلو در نظر میگیریم) مقدار R1 را محاسبه میکنیم:

ابتدا بهره ولتاژ تقویت کننده را بدست میاوریم :

$$Av = \frac{5}{.1} = 50$$

$$R1 = (Av * R2) - 1 \Rightarrow R1 = (50 * 1) - 1 = 49K$$

سپس ولتاژ خروجی op-amp را به adc میکرو اعمال میکنیم ، همانطور که مشاهده میفرمایید ، به ازای ولتاژ ورودی که 5 ولت است ، روی lcd عدد 1023 نمایش داده میشود، شما باید 1023 را به 100 میلی تبدیل کنید (روی lcd عدد 100 نمایش داده شود ، همچنین جلوی آن عبارت mv نیز به نشانه 100mv باشد، از طریق فرمول زیر که مانند حالت قبل است به سادگی میتوانید عدد 1023 را به 100 تبدیل کنید:

$$OUT = \frac{1023}{5} * X$$

X ولتاژ خروجی تقسیم ولتاژ است و OUT هم مقدار دیجتالی ریخته شده در متغیر میباشد ، حال بقیه محاسبات را با OUT انجام دهید:

$$Vin = \frac{OUT}{x} \Rightarrow x = OUT / Vin$$

در صورتی که vin را 100 میلی ولت در نظر بگیرید ، عدد اعشاری 1 روی lcd نمایش داده میشود که واحد آن ولت است اما اگر vin را 100 در نظر بگیرید ، باید واحد mv را به آن بیافزاید تا نمایش داده شود.(کلمه کم اوردم)

و در نهایت برنامه:

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 = Pinc.4 , Rs = Pind.3 , E = Pind.2
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word , B As Single
```

PEYMAN KHODDAMI

```
Config Single = Scientific , Digits = 1
```

```
Do
```

```
A = Getadc(0(
```

```
Locate 1 , 1 : Lcd A" " ;
```

```
B = A : B = B / 10.23
```

```
Locate 2 , 1 : Lcd B ; "mv"
```

```
Loop
```

```
End
```

مثال:مداری طراحی کنید که ولتاژ 5 میلی ولت را اندازه بگیرد ؟

ابتدا مدار تقویت کننده را طراحی میکنیم ، مقدار مقاومت R2 را 6.8 کیلو در نظر میگیریم و R2 را بدست میاوریم:

$$Av = \frac{V_O}{V_{in}} \Rightarrow Av = \frac{5}{5m} = 1000$$

$$R1 = (Av * R2) - 1 \Rightarrow R1 = (1000 * 6.8) - 1 = 6799K$$

زیاد شد ، مقدار 1 کیلو اهم را انتخاب میکنیم:

$$R1 = (Av * R2) - 1 \Rightarrow R1 = (1000 * 1) - 1 = 999K$$

در اینجا به علت زیاد بودن بهره مقدار یکی از مقاومت ها زیاد بدست میاید ، در این حالت شما میتوانید از دو تقویت کننده پشت سر هم استفاده کنید ...

حالا به ازای ولتاژ 5 میلی ولت که به 5 ولت تبدیل شد و به adc اعمال گردید ، روی lcd عدد 1023 نمایش داده میشود . ما باید 1023 را به 5 میلی ولت تبدیل کنیم (به جای 1023 که نمایش داده میشود ، 5 میلی ولت را نمایش دهیم) ، طبق فرمول:

$$OUT = \frac{1023}{5} * X$$

X ولتاژ خروجی تقسیم ولتاژ است و OUT هم مقدار دیجتالی ریخته شده در متغیر میباشد ، حال بقیه محاسبات را با OUT انجام دهید:

$$V_{in} = \frac{OUT}{x} \Rightarrow x = OUT / V_{in}$$

در صورتی که vin را 5 میلی ولت در نظر بگیرید ، عدد اعشاری 0.005 روی lcd نمایش داده میشود که واحد آن ولت است اما اگر vin را 5 در نظر بگیرید ، باید واحد mv را به آن بیافزاید تا نمایش داده شود.

PEYMAN KHODDAMI

در زیر برنامه و مدار مربوطه را مشاهده میفرمایید:

```
$regfile = "m16def.dat" : $crystal = 8000000
```

```
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Db4 = Pinc.1 , Db5 = Pinc.2 , Db6 = Pinc.3 , Db7 =  
Pinc.4 , Rs = Pind.3 , E = Pind.2
```

```
Config Adc = Single , Prescaler = Auto
```

```
Dim A As Word , B As Single
```

```
Config Single = Scientific , Digits = 5
```

```
Do
```

```
A = Getadc(0)
```

```
B = A : B = B / 204600
```

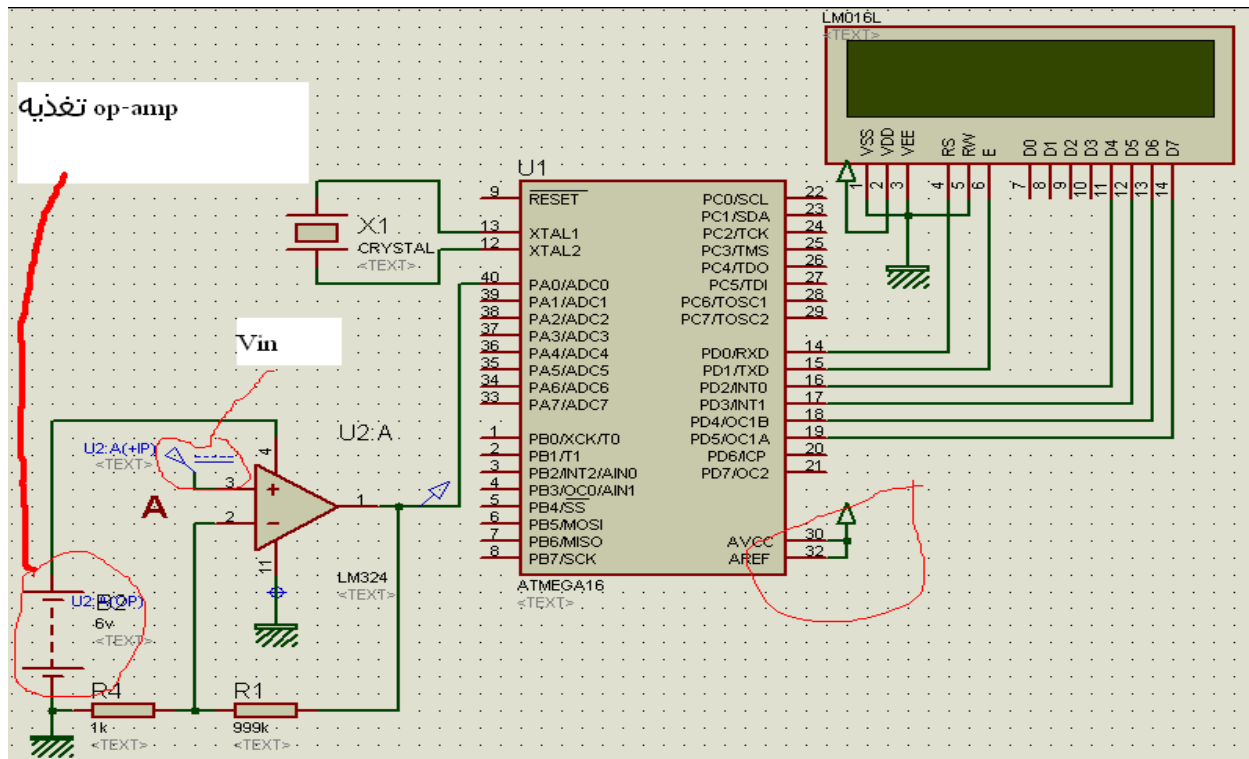
```
Locate 2 , 1 : Lcd B ; "v"
```

```
B = A : B = B / 204.600
```

```
Locate 1 , 1 : Lcd B ; "mv"
```

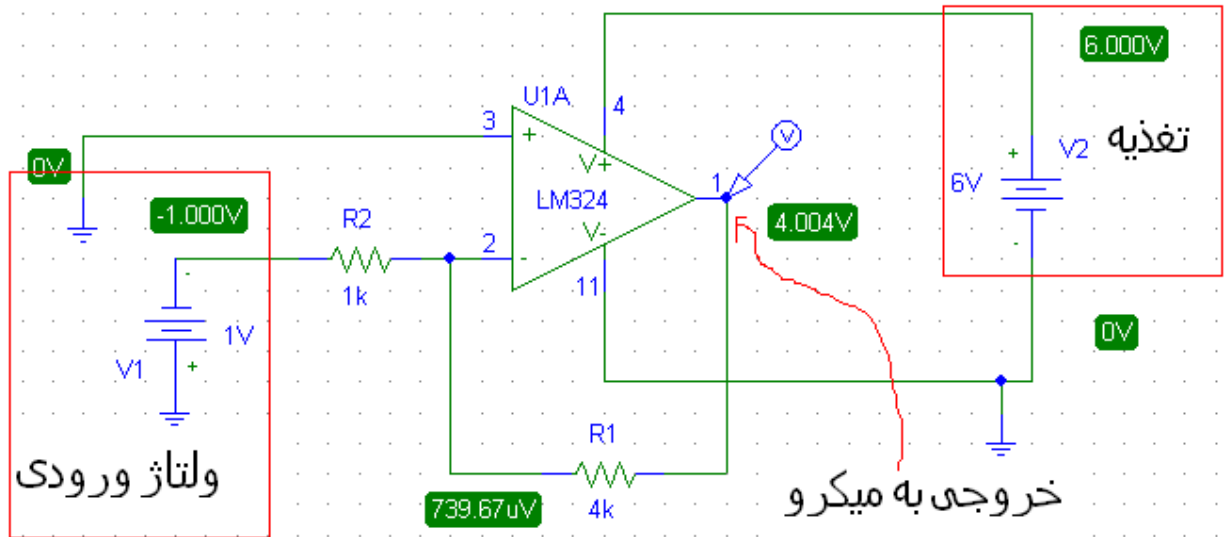
```
Loop
```

```
End
```



PEYMAN KHODDAMI

اندازه گیری ولتاژ های منفی ، برای اندازه گیری ولتاژ منفی شما باید ان را به مقدار متناظر مثبت تبدیل کنید و سپس ان را با یکی از روش های گفته شده به 0 تا 5 ولت تبدیل کرده و اندازه گیری کنید ، یکی از راه های تبدیل ولتاژ منفی به مثبت استفاده از تقویت کننده وارن ساز (معکوس کننده) است ، در زیر شکل و محاسبات مربوطه را مشاهده میفرمایید:



$$A_v = \frac{-R_2}{R_1}$$

ورودی این تقویت کننده فقط ولتاژ منفی است

در صورتی که ولتاژ منفی شما از 5 ولت بیشتر است ابتدا ان را با مقاومت (در بالا توضیح داده شد) تضعیف کنید و بعد از مثبت کردن به میکرو اعمال کنید.

چند نکته ی ضروری:

ولتاژ ورودی adc میتواند نهایتا 5.3 باشد (بیشتر از این ولتاژ سوختن adc حتمی است) این ولتاژ باید کمتر از 5 ولت باشد.

در هنگام کار اتصال پایه avcc به تغذیه فراموش نشود.

برای آشنایی با راه اندازی adc به صفحه ی 147 همین کتاب مراجعه کنید.