به نام خدا

اموزش میکرو کنترلر های 8051 به زبان بیسک (قسمت اول)

1nafar

www.ir-man.com

winter 1387

. . . . . .

فهرست : ---------- شماره صفحه

مقدمه

اگر چه امروزه با وجود میکرو های قدرتمند AVR و pic و ARM جای بحثی برای 8051 باقی نمی ماند ، اما هنوز در مدارات الکترونیکی از این ای سی استفاده می شود ، و در بسیاری از دانشگاه های کشور این میکرو به زبان اسمبلی تدریس میشود.

گاهی اوقات برای تعمیر یک دستگاه که در ان این نمونه از میکرو به کار رفته است ، نیاز به نوشتن برنامه است (برای ای سی جدید)و.... و یا شما حوصله یاد گیری زبان اسمبلی را ندارید ، این زبان تقریبا، با زبان بیسیک برای avr یکی است ، شما با یادگیری این زبان دیگر نیازی به استفاده از زبان اسمبلی ندارید ، فقط کافی است برنامه خود را به زبان بیسیک بنویسید و ان را کامپایل کنید ، سپس کد هگز موجود را با دی اسمبلر ، به اسمبلی تبدیل کنید ، برای مقایسه دو زبان بیسیک و اسمبلی من در زیر دو برنامه را نوشته ام ، هر دو برنامه عبارت "qwertyu" را روی lcd نمایش میدهد

برنامه به زبان بیسیک:

```
$regfile = "8052.DAT"
Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E = P3.5 , Rs = P3.6
Config Lcd = 16 * 2
Lcd "qwertyu"
End
```

برنامه به زبان اسمبلی:

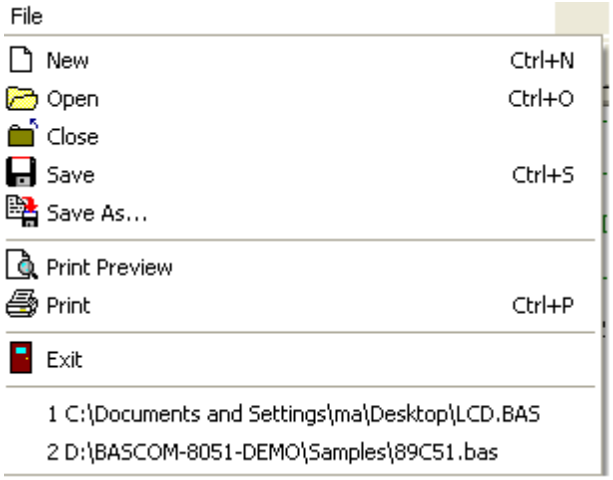
```
ORG 00H
MOV  A,#38H
call  I
MOV  A,#0EH
call  I
MOV  A,#01H
call  I
MOV  A,#06H
call  I
MOV  A,#80H
call  I
MOV  A,#'q'
call  Display
MOV  A,#'w'
call  Display
MOV  A,#'e'
call  Display
MOV  A,#'r'
call  Display
```

```
MOV   A,#'t'
call  Display
MOV   A,#'y'
call  Display
MOV   A,#'u'
call  Display
sjmp$
I:
CALL  DELAY
MOV   P1,A
CLR   P2.0;RS
CLR   P2.1;R/W
SETB  P2.2;E
CLR   P2.2
RET
Display:
call  DELAY
MOV   P1,A
SETB  P2.0;RS
CLR   P2.1;R/W
SETB  P2.2;E
CLR   P2.2
RET
DELAY:
    MOV   R0,#50
LOOP2: MOV   R1,#255
LOOP1: DJNZ  R1,LOOP1
    DJNZ  R0,LOOP2
    RET
    END
```

همانطور که مشاهده میفرمایید برنامه ی که به زبان بیسیک 5 خط شد ، به زبان اسمبلی 49 خط میشود . پس چرا بیخودی وقت خود را هدر دهید؟

در حال حاضر تنها مزیت زبان اسمبلی ، یاد گیری طرز کار سخت افزار است ، مثلا در برنامه بیسیک (برنامه بالا) شما طریقه ی ارسال اطلاعات برای lcd را درک نمیکنید ، نمیتوانید بفهمید که میکرو چگونه کار میکند ، اما در زبان اسمبلی چنین نیست . من در اینده در کتابی دیگر طرز کار کلیه سخت افزار های موجود را شرح میدهم .

اشنايي مختصر با محيط بسكام 8051



>>منوی file:

New-1 با زدن اين گزينه يك صفحه جديد براي نوشتن برنامه جديد باز ميشود ؛ اين صفحه مجهز به ويرايشگر دستورات مي باشد ؛ يعني در صورتي كه دستوري درست وارد شود به رنگ ابي در ميايد ولي اگر دستور اشتباه باشد به رنگ معمولي (مشكي)است (اين مورد براي تعداد كمي از دستورات اجرا نمي شود) .

open-2 با زدن اين گزينه مي توانيد برنامه اي را كه از قبل ذخيره كرده ايد باز كنيد.

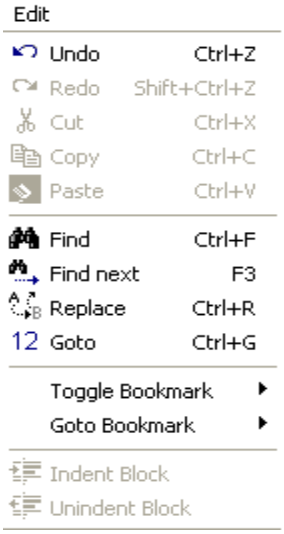
Close-3 با انتخاب اين گزينه صفحه اي كه براي نوشتن برنامه باز شده ؛ بسته ميشود.

Save-4 و Save as اين دو گزينه براي ذخيره كردن پروژه به كار ميروند.

Print Preview-5 وPrint اين دو گزينه براي چاپ كردن برنامه استفاده ميشوند با زدن گزينه Print Preview مي توانيد نسخه قابل چاپ را قبل از چاپ مشاهده كنيد.

Exit-6 با زدن اين گزينه برنامه بسكام به طور كامل بسته مي شود ؛ اما اگر برنامه شما ذخيره نشده باشد ؛در مورد ذخيره برنامه از شما پرسيده ميشود.

7- در زير گزينه Exit چند گزينه ديگر وجود دارد كه اين گزينه ها براي دسترسي سريع به اخرين فايل هاي كه باز بوده اند مي باشد.



>> منوي Edit:

Undo-1وRedo اين دو گزينه براي دست يابي به اخرين تغييرات انجام شده مي باشد .

Copy-2وCutوPaste اين سه گزينه براي برداشتن يا كپي كردن قسمتي از متن به جاي ديگر ميباشد.

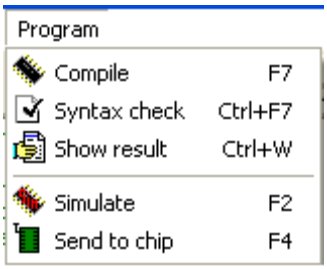
Findnext -3 وFind اين دو گزينه براي پيدا كردن قسمتي از متن در برنامه مي باشد. نحوه كار به اين صورت است كه بعد از انتخاب گزينه Find پن جره جديدي باز مي شود كه بايد در

قسمتText to find متن مورد نظر را تايپ كنيد بعد رويok كليك كنيد تا متن مورد نظر در برنامه انتخاب شود Findnext. متن هاي كه در خط هاي بعدي برنامه وجود دارد پيدا ميكند.

4-دو گزينه بعدي براي گذاشتن علامت در خطوط مختلف و پرش به انها مي باشد كه اهميت چنداني در برنامه نويسي ندارد.

5-IndentBlock وUnindentBlock اين دو گزينه متن انتخاب شده را به اندازه يك tab به چپ يا راست منتقل ميكند.

>>منوي Program:



1-Compile با انتخاب اين گزينه فايل هاي از قبيل هگز و گزارش و...ساخته ميشود .اگر در اين مرحله برنامه داراي خطا باشد پنجره اي باز ميشود كه در ان خطاها نمايش داده مي شوند ؛ با كليك كردن روي هر خطا خط مربوط كه داراي خطا است قرمز ميشود .

2-Syntax check با انتخاب اين گزينه برنامه از نظر غلط املايي چك ميشود (با زدن گزينه Compile ديگر نيازي به زدن اين گزينه نمي باشد).

3-Show result با انتخاب اين گزينه پنجرهاي باز ميشود كه در ان گزارش كلي از برنامه وجود دارد.

4-Simulate با انتخاب اين گزينه پنجره شبيه سازي باز ميشود و شما در اين پنجره كه داراي lcd و كيبرد و مبدل انالوگ به ديجتال و...ميباشدمي توانيد برنامه خود را شبيه سازي كنيد (به علت استفاده از پروتوس نيازي به اين محيط نمي باشد).

5-send to chip با انتخاب اين گزينه وارد محيط پروگرام كردن ميكرو مي شويد كه در قسمت هاي بعدي مفصلا توضيح داده مي شوند.

----------منوي بعدي منوي tools وoptions است كه داراي امكانات پر كار بردي مي باشد كه در مكان مورد نياز توضيح داده ميشود

برخی اصطلاحات و عناوینی که از این به بعد میبینیم:

1-vcc و:gnd :

منظور از این دو کلمه پایه های تغذیه می باشد که معمولا vcc، 5ولت می باشد و 0 gndولت است که برای میکرو های 8051 پایه 40 vcc و پایه 20 gnd می باشد.

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

2-پورت :

هرمیکرو دارای چندین پورت می باشد و هر پورت دارای چندین پایه است (معمولا 8 پایه)برای مثال میکرو 89C51 دارای 4 پورت 0,1,2,3 می باشد که هر پورت 8 پایه دارد وپورت 0 (port0) از پایه 32 تا 39 میکرو می باشد.

3- 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 8051 ،24مگاهرتز می باشد همچنین میکرو های 8051 دارای نوسان ساز داخلی نمی باشد ،و در صورتی که از کریستال استفاده نکنید میکرو کار نمیکند . کریستال به دو پایه xtal 1 و xtal2 متصل می شود >این پایه ها برای میکرو 8051 پایه های18 و 19 می باشد.

4-کامپایلر:

نرم افزار های هستند که برای راحت تر شدن امر برنامه نویسی بوجود امدند ، این نرم افزارها کدهای قرار دادی که از قبل تعیین شده اند و توسط کاربر انتخاب میشوند را به زبان قابل فهم برای ماشین (0 و1)ترجمه میکنند .

5- پروگرامر:

برای ریختن کد هگز روی میکرو به واسطی نیاز است تا اطلاعات را از پورت کامپیوتر به میکرو منتقل کند ، به این دستگاه پروگرامر میگویند(بعد از اشنایی با دستورات به ضمائم مراجعه کنید ، در انجا توضیحات بیشتری موجود است).

6-کد هگز :

میکرو کنترلر ها فقط 0 و 1 را میفهمند ، شما باید صفر و یک را بر روی انها بریزید ، از انجا که کار با صفر ویک دشورا میباشد ، زبان های برنامه نویسی بوجود امدند ، شما برنامه مورد نیاز خود را در این نرم افزار به زبانی نزدیک به گفتار مینویسید و نرم افزار کد هگز منتاظر را برای شما بوجود میاورد .

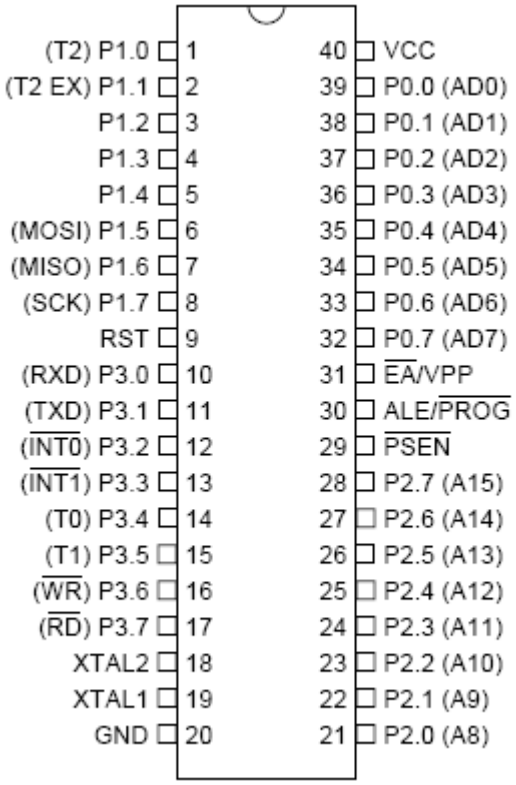
بررسی یک نمونه میکرو mC51(نام دیگر خانواده 8051)

در ایران از دو خانواده از میکرو های 8051 استفاده میشد.

سری s و سری c ، که سری s دارای مزیت های بیشتری نسبت به سری c است که در ادامه با انها اشنا می شویم .

این ایسی 40 پایه دارد و دارای امکانات زیر است :

1- 1 تا 3 عدد تایمر 16 بیتی
2- پورت سریال
3- کنترل کننده وقفه
4- چهار پورت ورودی و خروجی
5- حافظه rom (مقدار ان بسته به نوع میکرو متفاوت است)
6- حافظه ram (مقدار ان بسته به نوع میکرو متفاوت است)
7- ...



نقش پایه ها:

پورت 0 :

این پورت دارای 8 پایه میباشد ، این 8 پایه عاوه بر i/o عمومی میتواند به عنوان خطوط ادرس برای حافظه خارجی به کار رود

پورت 1 :

این پورت دارای 8 پایه میباشد ، این 8 پایه به عنوان i/o عمومی عمل میکنند ، بعضی از پای های این پورت نقش دومی دارند که در زیر اورده شده است :

پایه شماره 1 ، p1.0 ،(t2): این پایه علاوه بر نقش ورودی و خروجی عادی میتواند در حالتی که ،تایمر /کانتر 2 در حالت کانتر پیکربندی میشود ، به عنوان وردی پالس کانتر استفاده شود (کانتر پالس های موجود بر روی این پایه را میشمارد)

پایه شماره 2 ، p1.1 ،(t2 ex): این پایه علاوه بر نقش ورودی و خروجی عادی میتواند در حالتی که ،تایمر /کانتر 2 در حالت تایمر پیکربندی میشود ، به عنوان وردی پالس تریگر تایمر استفاده شود (در این مد ، هنگامی که به این پایه پالسی اعمال شود (نوع پالس در پیکربندی مشخص میشود) تایمر شروع به شمارش میکند).

پایه شماره 6 ، p1.5 ، (mosi): این پایه علاوه بر نقش ورودی و خروجی عادی میتواند در مد ارتباط سریال spi به عنوان خروجی داده میکرو مستر عمل کند .

پایه شماره 7 ، p1.6 ، (miso): این پایه علاوه بر نقش ورودی و خروجی عادی میتواند در مد ارتباط سریال spi به عنوان ورودی داده میکرو مستر عمل کند .

پایه شماره 8 ، p1.7 ، (sck): این پایه علاوه بر نقش ورودی و خروجی عادی میتواند در مد ارتباط سریال spi به عنوان پایه خروجی کلاک میکرو مستر یا ورودی کلاک برای میکرو پیرو عمل کند .

پورت 2 :

این پورت دارای 8 پایه میباشد ، این 8 پایه عاوه بر i/o عمومی میتواند به عنوان خطوط ادرس برای حافظه خارجی به کار رود

پورت 3 :

این پورت دارای 8 پایه میباشد ، این 8 پایه به عنوان i/o عمومی عمل میکنند ، بعضی از پای های این پورت نقش دومی دارند که در زیر اورده شده است :

پایه شماره 10 ، p3.0 ، (rxd) در هنگام راه اندازی ارتباط سریال ، این پایه به عنوان خروجی داده عمل میکند

پایه شماره 11 ، p3.1 ، (txd) در هنگام راه اندازی ارتباط سریال ، این پایه به عنوان ورودی داده عمل میکند

پایه شماره 12 ، p3.2 ، (int0) در هنگام راه اندازی وقفه (interrupt) پایه فعال کننده منبع وقفه 0 میباشد

پایه شماره 13، p3.3 ، (int1) در هنگام راه اندازی وقفه (interrupt) پایه فعال کننده منبع وقفه 1میباشد

هنگامی که وقفه راه اندازی شود ، با تحریک دو پایه نام برده ، میکرو برنامه در دست اجرا را رها میکند و به مکانی که در برنامه تعین شده میرود ، در این مورد در ادامه بیشتر توضیح داده میشود

پایه شماره 14، p3.4، (t0) در حالتی که ،تایمر /کانتر 0 در حالت کانتر پیکربندی میشود ، به عنوان وردی پالس کانتر استفاده شود (کانتر پالس های موجود بر روی این پایه را میشمارد)

پایه شماره 15، p3.5 ، (t1) در حالتی که ،تایمر /کانتر 1 در حالت کانتر پیکربندی میشود ، به عنوان وردی پالس کانتر استفاده شود (کانتر پالس های موجود بر روی این پایه را میشمارد)

پایه شماره 16، p3.6 ، (wr) عملیات نوشتن یا خواندن حافظه خارجی توسط این پایه مشخص میشود

پایه شماره 17، p3.7 ، (rd) عملیات ارسلا داده یا فرمان به حافظه خارجی توسط این پایه مشخص میشود

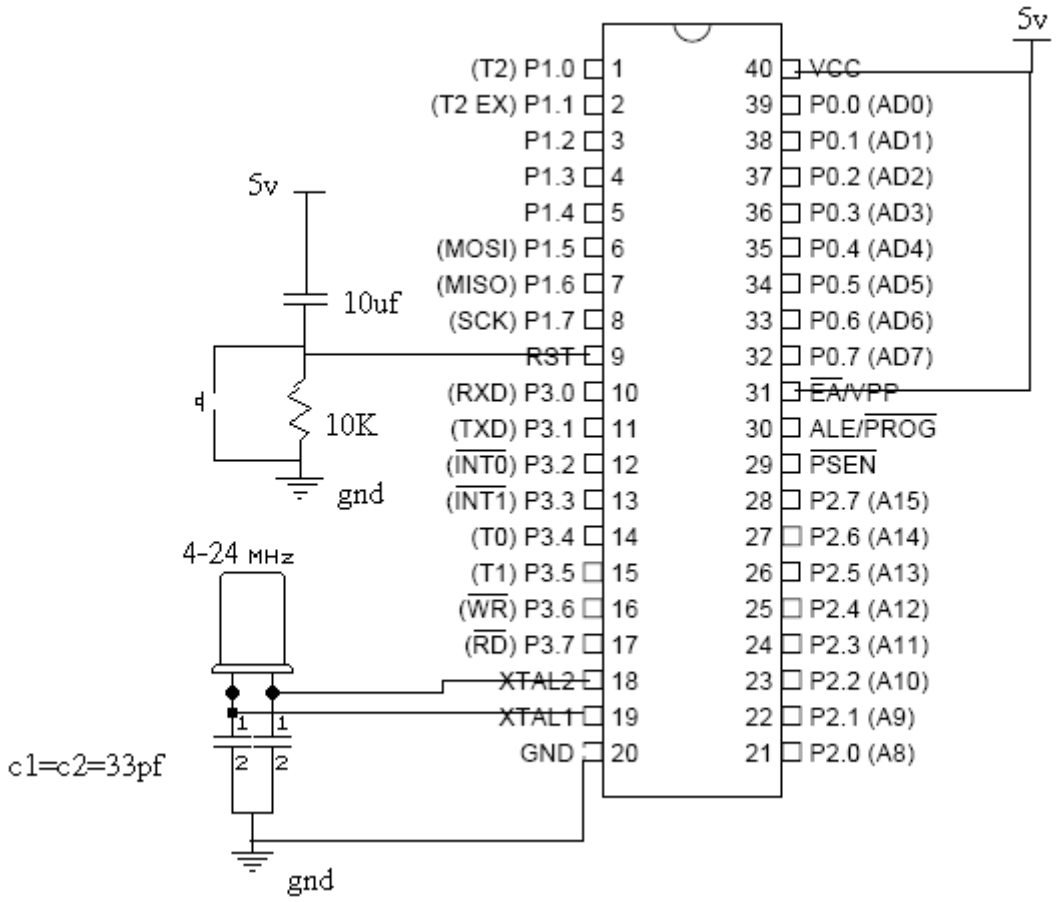
دیگر پایه ها :

پایه شماره 9 ،rst : این پایه ، پایه ریست میکرو است ، با اعمال یک پالس یک به صفر به این پایه ، میکرو کلیه المال ر حال اجرا را رها میکند و به خط ول حافظه میرود و کر را از اول شروع میکند.

پایه شماره 18 و 19 ، xtal : به این دوپایه یک کریستال متصل میشود ، از کریستال بر همزمانی بین اجزا پردازنده استفاده میشود .

پایه شماره 20 ، gnd :گراند میکرو میباشد که باید به گراند مدار متصل شود

پایه شماره 40 ، vcc : پایه تغذیه مثبت میکرو میباشد ک باید به 5 ولت متصل شود .ساده ترین مداری که برای کار با 8051 لازم است در زیر اورده شده است :



مراحل نوشتن یک برنامه جدید:

بعد از باز کردن برنامه بسکام گزینه new را از منوی file انتخاب کنید صفحه جدیدی که باز می شود محل نوشتن برنامه می باشد.

در زبان بیسیک همیشه اولین خط برنامه مر بوط به معرفی میکرو می باشد:

```
$regfile="micro name"
```

که گزینه micro neme یکی از گزینه های زیر است:

89c1051.dat

89c2051.dat

89c4051.dat

8052.dat

...

خط بعدی معرفی کریستال می باشد:

```
$crystal=x
```

که x کریستال مورد استفاده بر حسب هرتز است مانند:

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

بعد از معرفی کریستال نوبت به معرفی امکانات می باشد (امکانات شامل تایمر و ورودی یا خرو جی قرار دادن پورت ها و....می باشد) ، معرفی امکانات با دستور زیر می باشد:

```
Config
```

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

ودر نهایت برنامه با end به پایان می رسد.

از انجا که یاد گیری با مثال بهتر است در زیر ابتدا به معرفی lcd میپردازیم و مطالب را با مثالهای متنوع ادامه میدهیم.

Lcd کاراکتری

این نوع lcd از دارای چند سطر وستون می باشد که نام گذاری انها برمبنای سطر وستون می باشد

در زیر چند نوع lcd نام برده شده است

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

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 و40 ستون می باشد

تمام lcd های کاراکتری دارای 16 پایه می باشد که در زیر اورده شده است

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

پایه شماره 2 : VDDاین پایه پایه VCC LCD است که باید به 5ولت وصل شود

پایه شماره 3 : VEE این پایه روشنایی پیکسل های LCD را تعیین میکند

پایه شماره 4 : RS در lcd دو رجیستر به نام دستورالعمل وداده وجود دارد اگر rs =0 باشد lcd برای گرفتن دستورالعمل اماده می شود در غیر این صورت برای داده .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 (اطلا عات) از LCD میباشد (پایه دیتا ی صفر)

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

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

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

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

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

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

پایه شماره 14 : DB7 این پایه برای ارسال دیتا (اطلا عات) 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 ازدو طریق 8سیمه و 4سیمه به میکرو متصل شود

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

به طور کلی از خطوط دیتای 0تا 3 برای خوانده از lcd و از خطوط4تا7 برای نوشتن در lcd نوشته میشود در حالت نوشتن در lcd باید پایه RS پایین نگه داشته شود (صفر شود) ودر حالت خواندن از LCD باید پایه RS 1 شود)5ولت وصل شود)

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

اولین مرحله برای راه اندازی LCD معرفی کردن نام ان است:

برای این کار بعد از معرفی میکرو وکریستال با استفاده از دستور زیر می توان LCD ارمعرفی 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 انها وصل میشود

```
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 یd ... می باشد و y شماره پایه هست که از 0 تا 7 می باشد

و در نهایت با دستور زیر میتوان کاراکتری را روی lcd نمایش داد

```
lcd " karakter"
```

کارکتر میتواند یکی از کارکتر های اسکی یا اعداد باشد

مانند:

```
LCD "*1NAFAr"
```

یـا

```
LCD "+ qwetty"
```

یـا

```
Lcd "1@#$q#$%^&*()_+1"
```

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

```
Lcd"@#$%$%^&&*)("
Wait 1
Cls
Lcd "asdfgfghk"
```

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"
Locate 2,8
Lcd "mnbv"
```

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

روی lcd نمیتوان فارسی نوشت (فقط میتوان کاراکتر های اسکی ،که شامل حروف انگلیسی وعلامت های استاندارد و اعداد میشود را روی ان نشان داد).

Lcd دارای یک مکان نما می باشد که با دستور زیر میتوان ان را روش ن 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

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 tab به چپ منتقل میکند

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

نوشتن فارسی روی lcd :

Lcd های کاراکتری دارای یک حافظه می باشد که درون ان کدهای کارکتر های اسکی وجود دارد اما فارسی در ان وجود ندارد، اما در lcd حافظه موقتی وجود دارد که در ان 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 ان کاراکتر دلخوا ه 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 0تا 7 می باشد گذاشته شود .

بعد از ساخت کاراکتر جدید بادستور زیر می توانید ان را روی lcd نشان دهید

Lcd chr(?)

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

```
regfile =$ "80552.dat"

crystal=$8000000

Config Lcd = 16*2

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

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

Locate 1,1

Lcd chr(0

End
```

شما میتوانید تا نهایتا تا 8 کاراکتر را روی lcd نمایش دهید.

در اینجا دستورات به 8 قسمت تقسیم شده که در زیر ارده شده است :

1- بدنه یک برنامه در محیط بسکام

2- اعداد و متغییر ها در بسکام

3-دستور ات حلقه و پرش

4-دستورات ایجاد تاخیر

5-زیر برنامه ها و فراخوانی توابع

6-توابع ریاضی و محاسباتی

7- توابع تبدیل کدها و متغییر ها به یکدیگر

8- دستورات اجرایی

بدنه یک برنامه در محیط بسکام

```
$REGFILE = micro name
```

برای شروع یک برنامه در محیط BASCOM ابتدا باید میکرو مورد نظر تعریف گردد . microname نام میکرو مورد استفاده است که می تواند یکی از موارد زیر باشد .

```
89c1051.dat

89c2051.dat

89c4051.dat

.8052dat
```

....

خط بعد معرفی کریستال میباشد.

برای مشخص کردن فرکانس کریستال استفاده شده برحسب هرتز، از دستور زیر استفاده می نماییم .

```
$CRYSTAL = X
```

X فرکانس کریستال استفاده شده بر حسب هرتز است .

مـثـال

```
$ CRYSTAL = 14000000 ’14MHZ external osc

$CRYSTAL = 8000000 ’8MHZ external osc

$CRYSTAL = 1000000 ’1MHZ internal osc
```

دستور END:

این دستور دراخرین خط برنامه قرار می گیرد و هنگامی که برنامه به این خط می رسد،تمام وقفه ها غیر فعال شده و یک حلقه بی نهایت تولید و برنامه خاتمه می یابد .

در محیط بسکام همیشه اولین کار معرفی میکرو و کریستال است در مرحله بعد امکانات (lcd و kbd ...) معرفی میشوند و در مرحله اخر از امکانات استفاده میشود و در نهایت برنامه با دستور end پایان میابد

اعداد و متغییر ها در بسکام

در زبان بیسیک منظور از متغییر یک ارایه است که میتواند حول دو عدد تغییر کند مثلا هنگامی که میگوییم dim a as byte ، ما یک م متغییر به نام a از جنس بایت تعریف کرده ایم که این متغییر میتواند بین 0 تا 255 تغییر کند(مثلا میتوانیم بگوییم a=10 یا a=226 یاa=a+6)

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 که در برنامه بکار برده میشود . Data type نوع داده است که می تواند طبق موارد زیر STRING ,BIT, BYTE ,INTEGER ,LONG ,WORD یا SINGLE باشد .

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

BIT :این متغییر میتواند صفر یا یک باشد.

BYTE :این متغییر میتواند از0 تا 255 تغییر کند و فقط شامل اعداد صحیح مثبت می شود.

WORD :این متغییر میتواند از0 تا65535 تغییر کند و فقط شامل اعداد صحیح مثبت می شود .

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

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

SINGLE :این متغییر میتواند از-1.5*10^45 تا 3.4* 10^38تغییر کند و فقط شامل اعداد صحیح واعشاری مثبت و منفی می شود.

STRING :این متغییر میتواند از0 تا 245 بایت تغییر کند تغییر کند و برای حروف وعلائم استفاده می 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

DIM MICRO AS WORD

DIM HASAN AS LONG
```

مثال :

```
"regfile = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Dim Ali As Word

Dim Wqew As Byte

A = 10

Wqew = 5

Locate 1 , 1

Ali = A + Wqew 'ali=15

Lcd Ali

Locate 2 , 1

Ali = A * Wqew 'ali=50

Lcd Ali

End
```

نکته: در صورتی که در یک متغییر بیشتر از بعدش مقدار قرار دهید، با خطا مواجه میشوید

مثال

```
DIM A AS BYTE

A=300
```

مورد بالا غلط می باشد چون بایت می تواند از 0 تا 255 تغییر کند ومقدار 300 بیشتر از بعد بایت 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 , STRINGCONST رشته انتساب یافته به SYMBOL و EXPRESSION میتواند عبارتی باشد که نتیجه آن به SYMBOL انتساب یابد .

مثال

```
CONST S = "TEST "

CONST A = 5

CONST B1 =&B1001

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

دستور ALIAS:

از این دستور برای تغیر نام متغیر استفاده می شود .

مثال

```
DIM Q AS BIT
```

```
Q ALIAS P3.1
```

حال شما می توانید بجای P3.1از متغیر Q استفاده نمایید .

```
SET Q 'is equal with SET P3.1
```

دستور INSTR:

این دستورمحل 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 = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim S As String * 15

Dim Z As String * 5

Dim Bp As Byte

Cls

S = "This is a test"

Z = "is"

Bp = Instr(s , Z(

Locate 1 , 1

Lcd Bp

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

```
Locate 2 , 1

Lcd Bp

End
```

دستور INCR و دستور DECR:

```
INCR X

DECR X
```

دستور INCR یک واحد به متغییر عددی X می افزاید و دستور DECR یک واحد از ان کم میکند

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Dim C As Long

Do

Incr A

Decr C

Locate 1 , 1

Lcd C

Locate 2 , 1

Lcd A

WAITMS 500

Loop

End
```

(از دستور do -loop برای ایجاد یک حلقه استفاده میشود که در بخش های بعد شرح داده میشود)

دستورات HIGH و LOW:

LOW این دستور (least significant byte) LSB یک متغیر را برمی گرداند .

HIGH این دستور(most significant byte) MSB یک متغیر را برمی گرداند .

(Var = HIGH (s

MSB متغیر S در Var قرار می گیرد .

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

LSB متغیر S در Var قرار می گیرد

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim I As Integer

Dim Z As Byte

Dim Q As Byte

Cls

I = &h1001

Z = Low(i) ' is 1

Locate 1 , 1

Lcd Z

Q = High(i) 'IS 16

Locate 2 , 1

Lcd Q
```

```
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 .

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 = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7
= P3.4 , E = P3.5 , Rs = P3.6

Dim S As String * 12

Dim Z As String * 12

"S = "Hello WORLD

Z = Ucase(s ) 'Z = HELLO WORLD

Locate 1 , 1

Lcd Z

(Z = Lcase(s

Locate 2 , 1

Lcd Z
```

End

دستور RIGHT و دستور LEFT

دستور RIGHT:با این دستورقسمتی 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 .

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 = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7
= P3.4 , E = P3.5 , Rs = P3.6

Dim S As String * 10

Dim Z As String * 10

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
```

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 نهایتا می تواند 255 بایت طول داشته باشد . توجه داشته باشید که فضای خالی (SPACE BAR) خود یک کاراکتر به حساب می آید .

```
"regfile = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

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 SWAP:

```
SWAP var1 , var2
```

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

دو متغیر var1 وvar2 بایستی از یک نوع باشند .

```
"regfile = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Dim A As Byte

Dim C As Byte

Cls

A = 10

C = 20

Swap A , C 'swap them

Locate 1 , 1

Lcd A 'A=20

Locate 2 , 1

Lcd C 'B=10

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
```

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 = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

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 ازاین دستور استفاده می شود .

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

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

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6
```

```
Dim S As String * 10

Dim Z As String * 10

Cls

S = Space(5(

Z = "qwer"

Locate 1 , 1

Lcd "(" ; S ; Z ; ")" 'lcd( qwer)

End
```

دستور fusing:

از این دستور برای روند کردن رشته های 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 = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Dim S As Single

Dim A As Byte

Dim C As String * 10

Cls

S = 10

A = 3

S = S / A

Locate 1 , 1
```

```
Lcd S 'lcd "3.33333333333333"
Locate 2 , 1
C = Fusing(s(##.# ,
Lcd C 'lcd 3.33
Locate 2 , 8
C = Fusing(s , 000(####.
Lcd C 'lcd 003.3333
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

```
&0110010b
&h01ff
```

در اینجا تمامی دستورات مر بوط به اعداد ومتغییر ها که در زبان بیسیک برای میکرو 8051 است گفته شد

در درسهای بعدی با این دستورات به صورت کاربردی اشنا می شوید.

دستور ات حلقه و پرش

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:

دستور goto:

```
:LABLE

برنامه

GOTO LABLE
```

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

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E = P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Dim C As Long

Q:

Incr A

Decr C

Locate 1 , 1

Lcd C

Locate 2 , 1

Lcd A

Waitms 500

Goto Q

End
```

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 حلقه ادامه یابد .

همچنین با دستور زیر میتوان تعداد دفعات اجرای ان را معین کرد

```
do
  برنامه
Loop Until A = x
```

که A یک متغییر 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

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Do

Incr A

Wait 1

Locate 1 , 1

Lcd A

Loop Until A = 10
```

```
Lcd "END"

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 مقداراولیه 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 = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Dim D As Byte

Dim C As Integer

For A = 1 To 10 Step 2

Locate 1 , 1

Lcd A

Waitms 500

Next A

For C = 10 To -5 Step -1

Locate 1 , 6

Lcd C

WAITMS 500
```

```
Next

For D = 1 To 10

Locate 2 , 1

Lcd D

WAITMS 500

Next

End
```

دستور WHILE-WEND

```
WHILE condition

statements

WEND
```

دستورالعمل 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 . دستورالعمل while بصورت ورود به حلقه به شرط می باشد , بنابراین While ممکن است در حالتهایی اصلا اجرا نشود .

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

مانند

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7
= P3.4 , E = P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

A = 1
```

```
While A <10
Locate 1 , 1
Lcd A
Incr A
Waitms 600
Wend
Lcd "END"
End
```

دستورات دیگری نیز برای ایجاد حلقه وجود دارد که خود شما باید به انها دست یابید.

دستورات ایجاد تاخیر

گاهی نیاز است برنامه در یک قسمت متوقف شود برای اینکار از دستورات تاخیر که در زیر بیان میشود استفاده میشود، این دستورات در هر جا که استفاده شوند میکرو CPU میکرو در انجا متوقف میشود و هیچ کار دیگری انجام نمیدهد ،در صورتی که میخواهید در ان واحد یک تاخیر ایجاد کنید وکاری را نیز انجام دهید از تایمر ها استفاده کنید (تایمر ها در بخش های بعدی گفته میشود.).

دستور wait:

برای ایجاد تاخیر در برنامه، از دستور wait استفاده میشود

دستور wait به دو شکل زیر است:

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

Waitms 720 تاخیر به مدت 700 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 1میلی ثانیه ایجاد می شود

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

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Qwer:

Lcd "fjghfgf"
```

```
Waitms 400

Cls

Lcd "123678"

Wait 1

Cls

Lcd "fkjjkb"

Wait 1

Cls

DelAy

Goto Qwer

End
```

در صورتی که کریستال معرفی شده در برنامه با کریستال استفاده شده یکی نباشد دستورات تاخیر به درستی اجرا نمی شود مثلا ممکن است به جای1 ثانیه 3 ثانیه تاخیر ایجاد شود.

راه های دیگری نیز برای ایجاد تاخیر وجود دارد که کشف انها باشما .

زیر برنامه ها و فراخوانی توابع

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 برنمی گرداند .در زمان ارسال داده بصورت BYREF آدرس داده به زیر برنامه فرستاده می شود و در محتوای آن تغیر ایجاد می شود .ولی در حالت BYVAL یک کپی از داده فرستاده می شود و به هیچ وجه در محتوای آن تغیری 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 Subپایان می یابد .

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

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 '223 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 بخواهیم عدد ثابت را به زیر برنامه انتقال دهیم بایستی حتما با آرگومان BYVAL آن را انتقال دهیم .

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte , C As Byte

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

A =65

Call Test

Locate 1 , 1

lcd A ' will print A = 10

End

Sub Test:

C = 10

A = 15

Locate 1 , 8

Lcd C

Locate 2 , 1
```

```
Lcd A

End Sub
```

دستور GOSUB

این دستور به زیربرنامه پرش می کند و اجرای برنامه را از آدرس برچسب ادامه می دهد .

```
GOSUB label
```

LABEL نام برچسبی زیر برنامه است که به آن پرش می شود .توسط دستور RETURN می توان از SUB برگشت کرد و اجرای برنامه بعد از دستور GO SUB ادامه می یابد .

```
"regfile = "8052.DAT$

crystal = 8000000$

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim X As Byte

:Q

Wait 1

Gosub R

Goto Q

End

:R

X = X + 2

Locate 1 , 1

Lcd X

Return
```

دستوراتی که در بالا گفته شد برای فراخوانی زیر برنامه بود ، شما میتوانید با دستور GOTO به مکانهای مختلف برنامه پرش کنید

```
$regfile = "8052.DAT"

$crystal = 8000000
```

```
Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim X As Byte

Dim C As Byte

Goto T

W:

X = C + 4

Locate 2 , 1

LCD X

Goto R

T:

Do

C = C + 1

Locate 1 , 8

Lcd C

Wait 1

Loop Until C = 10

Goto W

R:

X = X * 2

Locate 1 , 1

Lcd X

Goto T

End
```

توابع ریاضی و محاسباتی

از عملگرهای ریاضی زیر می توان در محیط BASCOM استفاده کرد و عملیات ریاضی خود را انجام داد.

سنبل ریاضی	نام عملیات	مثال	نتیجه
+	جمع	A+b	A+b
-	تفریق	a-b	a-b
*	ضرب	A*b	ab
/	تقسیم	a/b	a/b
<	کوچکتر	A<b	A=0,b=1
>	بزرگتر	a>b	A=3,b=1
=	مساوی/انتصاب	A=b	B=2,a=2
=>	بزرگتر مساوی	A=>b	B=1,a=(1-10)
<=	کوچکترمساوی	B<=a	B=1,a=(1-10)
<>	ناتساوی	A<>b	A=1,b=3

مثال :

```
$regfile = "8052.DAT”

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6
```

```
Config Lcd = 16 * 2
Dim Q As Byte
Dim W As Byte
Dim E As Byte
Dim R As Byte
Dim T As Byte
Q = 3
W = Q + 3
E = W * Q
R = E / 3
T = E - r
Locate 1 , 1
Lcd W
Locate 1 , 8
Lcd E
Locate 2 , 1
Lcd R
Locate 2 , 8
Lcd T 't=10
End
```

تابع ABS :

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

این دستور به معنای ریاضی|VAR2| = VAR (قدرمطلق) است .

```
$regfile = "8052.DAT"
$crystal = 8000000
Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6
```

```
Config Lcd = 16 * 2

Dim A As Integer , C As Integer

A = -100

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

Lcd C 'C= 100

End
```

تابع 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 هربار استفاده از این دستور عدد مثبت تصادفی دیگری بدست خواهد آمد .

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim X As Byte

Do

X = Rnd(100(

Locate 1 , 1

Lcd X

Waitms 500

Loop

End
```

توابع تبدیل کدها و متغییر ها به یکدیگر

دستور ASC:

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

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

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 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 .

دستور MAKEDEC:

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

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

```
$regfile = "8052.DAT"

$crystal = 8000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

Dim A As Byte

Dim S As String * 10

S = "ABC"

A = Asc(s(

Locate 1 , 1

Lcd A '65

S = Hex(a(

Locate 1 , 8 '41

Lcd S

A = 50

A = Makebcd (A(

Locate 2 , 8 '80

Lcd A

End
```

دستورات اجرایی

این دستورات کد هگزی را برای ریختن روی میکرو ایجاد نمیکند ، این دستورات فقط برای راحت تر شدن کار شما با کامپایلر بسکام بوجود امده اند و استفاده از انها باعث راحتی کار میشود

دستور $ASM

با این دستور میتوانید در بین دستورات بیسیک از دستورات اسمبلی استفاده کنید ، فرم کلی این دستور به شکل زیر است :

```
$ASM

برنامه اسمبلی

$END ASM
```

گذاشتن دستور $end asm الزامی است ، مانند:

```
$regfile = "8052.DAT"

$crystal=12000000

do

$asm

mov p1,#1

$end asm

wait 1

reset p1.0

wait 1

loop

End
```

دستور $CRYSTAL:

با این دستور مقدار فرکانس نوسان ساز بر حسب هرتز مشخص میشود ، این دستور به فرم زیر است:

```
$CRYSTAL=x
```

Xفرکانس نوسانساز بر حسب هرتز است ، مانند:

```
$regfile = "8052.DAT"

$crystal=12000000

do

set p1.0

wait 1

reset p1.0

wait 1

loop

End
```

دستور $DEFAULT XRAM :

با نوشتن دستور $DEFAULT XRAMدر برنامه کلیه متغیر های موجود ، در حافظه ی xram ذخیره میشوند ،مانند:

```
$regfile = "8052.DAT"

$crystal=12000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

$DEFAULT XRAM

dim a as byte

do

incr a:wait 1

locate 1,1:lcd a

loop

End
```

دستور $INCLUDE:

با این دستور که به شکل زیر است میتوان یک فایل دیگر با پسوند bas. را وارد برنامه کرد:

```
$INCLUDE  "file.bas"
```

مانند : در این مثال فایلی به نام 123.bas وارد برنامه اصلی میشود ، این فایل در کنار برنامه اصلی (در محل ذخیره برنامه اصلی) ذخیره شده است:

```
$regfile = "8052.DAT"

$crystal=12000000

$INCLUDE  "123.bas"

do

locate 1,1

lcd "qwetr"

loop

End
```

فایل 123 پیکر بندی lcdمیباشد:

```
Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2
```

دستور $LIB :

با این دستور میتوان از دیگر کتاب خانه های که برای بسکام نوشته شده اند استفاده کرد ، در صورتی که شما کتابخانه را در پوشه LIB در محل نصب بسکام کپی کنید ،نیازی به نوشتن این دستور نمیباشد. این دستور به فرم زیر است:

```
$LIB "mylib.lib"
```

دستور $SIM:

با نوشتن این دستور در برنامه کلیه تاخیر های موجود نادیده گرفته میشود ، از این دستور در شبیه سازی با شبیه ساز داخلی بسکام استفاده میشود، این دستور به فرم زیر است و برای استفاده کافی است ان را در مکان دلخواه برنامه بنویسید:

```
$SIM
```

دستور $WAIT:

با نوشتن این دستور در برنامه ، شروع کار میکرو با یک ثانیه تاخیر همراه خواهد بود ، این دستور به فرم زیر است و استفاده از ان مانند $sim میباشد:

```
$WAIT
```

دستور $ROMSTART:

این دستور به فرم کلی زیر است و مشخص میکند که اجرای برنامه از چه مکانی از حافظه شروع شود:

```
$ROMSTART =  [&H] address
```

[&H] address ادرس حافظه به فرم کد هگز میباشد ، در صورت عدم استفاده از این دستور ، برنامه از خانه 0 حافظه شروع به اجرا میکند مانند:

```
$regfile = "8052.DAT"

$crystal=12000000

Config Lcdpin = Pin , Db4 = P3.1 , Db5 = P3.2 , Db6 = P3.3 , Db7 = P3.4 , E =
P3.5 , Rs = P3.6

Config Lcd = 16 * 2

$ROMSTART = &H4000

do

locate 1,1

lcd "tfdhgh"

loop

End
```

راه اندازی lcd گرافیکی

تا حالا حتما lcd گرافیکی دیدید ، این lcd ها دارای پیکسل بیشتری نسبت به lcd های معمولی میباشند و میتوان روی انها عکس و متن و ... را نشان داد ،

در زیر در مورد چگونگی راه اندازی این lcd با 851 بحث میکنیم:

lcd های گرافیگی در نمونه های مختلف در بازار وجود دارد ،که انها را از نظر تعداد پیکسل نام گذاری میکنند ،مانند 240*64 و...

این lcd ها دارای پایه های زیر میباشد. (ممکن است محل این پایه هابر روی lcd تفاوت داشته باشد ، اما در همه lcd ها یکسان است (بهتر است موقع خرید دیتا شیت lcd را از فروشنده دریافت کنید))

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

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

3 - dataport : (d0 تا d7)(دیتا پورت) این 8 پایه مربوط به دیتای lcd میباشد (lcd اطلاعات را از طریق این 8 پایه رد وبدل 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 چند تا از پایه های دلخواه 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 متصل میگردد.

- ce :این پایه برای فعال کردن چیپ (درایور) 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 باشدlcd کد را میگیرد و اگر 0 باشد lcd دیتا را میگیرد (دیتا فرمانها می باشد و کد متن هاو اشکال است))، که به یکی از پایه های میکرو که در برنامه مشخص میشود متصل میگردد.

- wr :این پایه و پایه rd وضعیت خواندن یا نوشتن دیتا را برای lcd معین میکند بدین صورت که اگر این پایه در وضعیت صفر وپایه rd در وضعیت 1 قرار گیرد ،میکرو دیتا را به lcd ارسال میکند و lcd ان را نمایش میدهد ، اما اگر این پایه در وضعیت 1 قرار گیرد و پایه rd در وضعیت صفر ، lcd ،دیتای را که قبلا بر رویش نوشته شده است را به میکرو ارسال میکند در بسکام 8051 امکان خواندن از lcd وجود ندارد بنابراین این پایه به گراند و پایه rd به 5 ولت متصل میشود.

- rd : در بالا گفته شد.

- fs : این پایه برای مشخص کردن فونت lcd است ، که در اتصال به 8051 ازاد میماند.

- 5 con یا vo :پایه کنترل کنتراست lcd است که با توجه به نوع lcd به vcc یا -vcc هر ولتاژ دیگر متصل میشود

با توجه به مطالب بالا پیکر بندی lcd گرافیکی در بسکام به صورت زیر است

```
Config GRAPHLCD = type , PORT = mode, CE = pin , CD = cd , COLS = 30

Reset Px.y
```

type : مشخص کننده نوع lcd استفاده شده است ،که میتواند یکی از موارد زیر باشد :

.....128 * 240 , 48 * 160 , 64 * 128 ,128 *128, 64 * 240

```
Config GRAPHLCD =64*240
```

mode: یکی از پورت های دلخواه میکرو است ، که برای ارسال دیتا به lcd به کار میرود (پایه های دیتا پورت به این پورت میکرو متصل میشود)(هر پورت از 8 پین (پایه)تشکیل میشود که اولین پین ، پین صفر و اخرین پین ،پین 7 میباشد)

```
PORT = P2
```

pin : یکی از پایه های دلخواه پورتی است که در قسمت که پایه های ce و cd ، lcd به انها متصل میشوند

COLS: نشان دهنده تعداد ستون های استفاده شده برای هر حرف میباشد که حداکثر باید 30 باشد

px.y این پایه یکی ار پایه های دلخواه میکرو است که پایه ریست lcd به ان متصل میشود.

مثال:

```
"regfile = "8052.DAT$

crystal = 1000000$

Config Graphlcd = 240 * 64 , Port = P2 , Ce = P3.1 , Cd = P3.0 , Cols =29

Reset P3.2
```

نام پایه های lcd مورده استفاده در مثال و محل اتصال انها:

نام پایه بر روی lcd	محل اتصال	شماره پایه
GND	GND	1
GND	GND	2
v	5v5	3
-9V potmeter	-9v	4
WR	GND	5
RD	+5v	6
CE	p3.1	7
C/D	p3.0	8
NC	not conneted	9
RESET	p3.2	10
D0-D7	p2	18-11
FS	not connected	19
NC	not connected	20

پایه d0 (دیتا شماره 0 lcd)به پایه p2.0 (پایه 21)متصل میشود،پایه d1 (دیتا شماره1 lcd)به پایه p2.1 (پایه 22)متصل میشود.....پایه d7 (دیتا شماره 7 lcd)به پایه p2.7 (پایه 28)متصل میشود.

دستورات مربوط به lcd گرافیکی:

دستور lcd

با این دستور میتوان متن یا کارکتری را بر روی lcd نمایش داد

```
Lcd "MCS Electronics "
Lcd "Mdgdgsdsscs "
```

دستور locate :

با این دستور میتوان متن یا کاراکتری را در مکان دلخواه بر روی lcd گرافیکی نمایش داد

```
Locate 16 , 1
Lcd "write this to the lower line "
Locate 16 , 5
Lcd "fgghfhghfhgjhj "
```

دستور cls :

با این دستور تمام lcd پاک میشود .

با استفاده از دستورCls Text می توان قسمت متنی lcd را پاک کرد .

و با دستور Cls graph می توان قسمت گرافیکی را پاک کرد .

مثال:

```
$regfile = "8052.DAT"
$crystal = 1000000
Config Graphlcd = 240 * 64 , Port = P2 , Ce = P3.1 , Cd = P3.0 , Cols = 29
Reset P3.2
Cls
Wait 1
Locate 1 , 1
Lcd "1nafar"
```

```
Locate 2 , 1
Lcd "/*-+234#$%"()*&^^
Locate 3 , 1
Lcd "12345678901234567890123456789012345678901234567890"
Locate 16 , 1
Lcd "qwertyuiop"
Wait 2
Lcd "jkfjgfghfdh"
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 .

```
Pset 10 , 20 , 255
Pset 5, 127 , 255
Pset 10 , 20 , 0
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 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 .

مثال:

```
$regfile = "8052.DAT"
$crystal = 1000000
Config Graphlcd = 240 * 64 , Port = P2 , Ce = P3.1 , Cd = P3.0 , Cols = 29
Reset P3.2
Dim X as Byte, Y as Byte
Cls
Cursor Blink
Wait 1
Cursor On
Wait 1
Cursor Off
Locate 1 , 1
Lcd "MCS Electronics"
Locate 2 , 1 : Lcd "fhfnvn"
Locate 3 , 1 : Lcd "12345678901234567890"
Locate 16 , 1 : Lcd "write this to the lower line"
Wait 2
Cls Text
For X = 0 To 10
```

```
For Y = 0 To 10

Pset X , Y , 1 'make a nice block

Next

Next

Wait 3

Cls Graph

End
```

دستور SHOWPIC x, y , label

این دستور یک عکس را بر روی lcd گرافیکی نمایش میدهد .

دیگر دستورات مانند lcd کارکتری است ...

مراحل نمایش عکس بر روی lcd گرافیکی:

اگر عکس مورد نظر رنگی است ان را به محیط فتوشاپ برده و در انجا ان را به عکس سیاه وسفید 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 bmp 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 , labelعکس را در مختصاتx, y نمایش دهید .

label نام برچسبی است که عکی مورد نظر در ان قرار میگیرد .

برچسب $bgf "mcs.bgf" اشاره به عکس مورد نظر که در کنار برنامه اصلی قرار گرفته است.

راه اندازی سروو موتور

سروو موتور ها موتور های پرقدرتی هستند که میتواند حول یک زاویه خاص گردش کنند ، از این نوع موتور ها در بازوی ربات ها ، باز بسته کردن درب و پنجره و... استفاده میشود ، این موتور ها دارای 3 سیم میباشند که دوتا از انها تغذیه و سیم دیگر برای کنترل موقعیت به کار میرود ، در واقع زمان یک بودن این پایه زاویه ای که سررو روی ان قفل میشود معین میکند ، برای دریافت اطلاعات بیشتر به دیتا شیت سروو خریداری شده مراجعه کنید.

پیکربندی سروو با دستور زیر انجام میشود:

```
Config SERVOS = number , SERVO1 = Px.y , SERVO2 = Px.y , SERVO3 = Px.y ,
SERVO4 = Px.y , RELOAD = value
```

Number: تعداد سروو های که به میکرو متصل است و باید راه اندازی شود را نمایش میدهد ، شما میتوانید از 1 تا 16 سروو را راه اندازی کنید (به جای number تعدا سروو نوشته میشود).

Px.y: نشان دهنده پورت و پایه ای است که پایه کنترل سروو به ان متصل است

Value : زمان مورد نظر برای بار گذاری مجدد اطلاعات میباشد که پیش فرض 100 میکرو ثانیه است .

با دستورات زیر میتوانید زاویه ای را که هر سروو باید بچرخد را مشخص کنید:

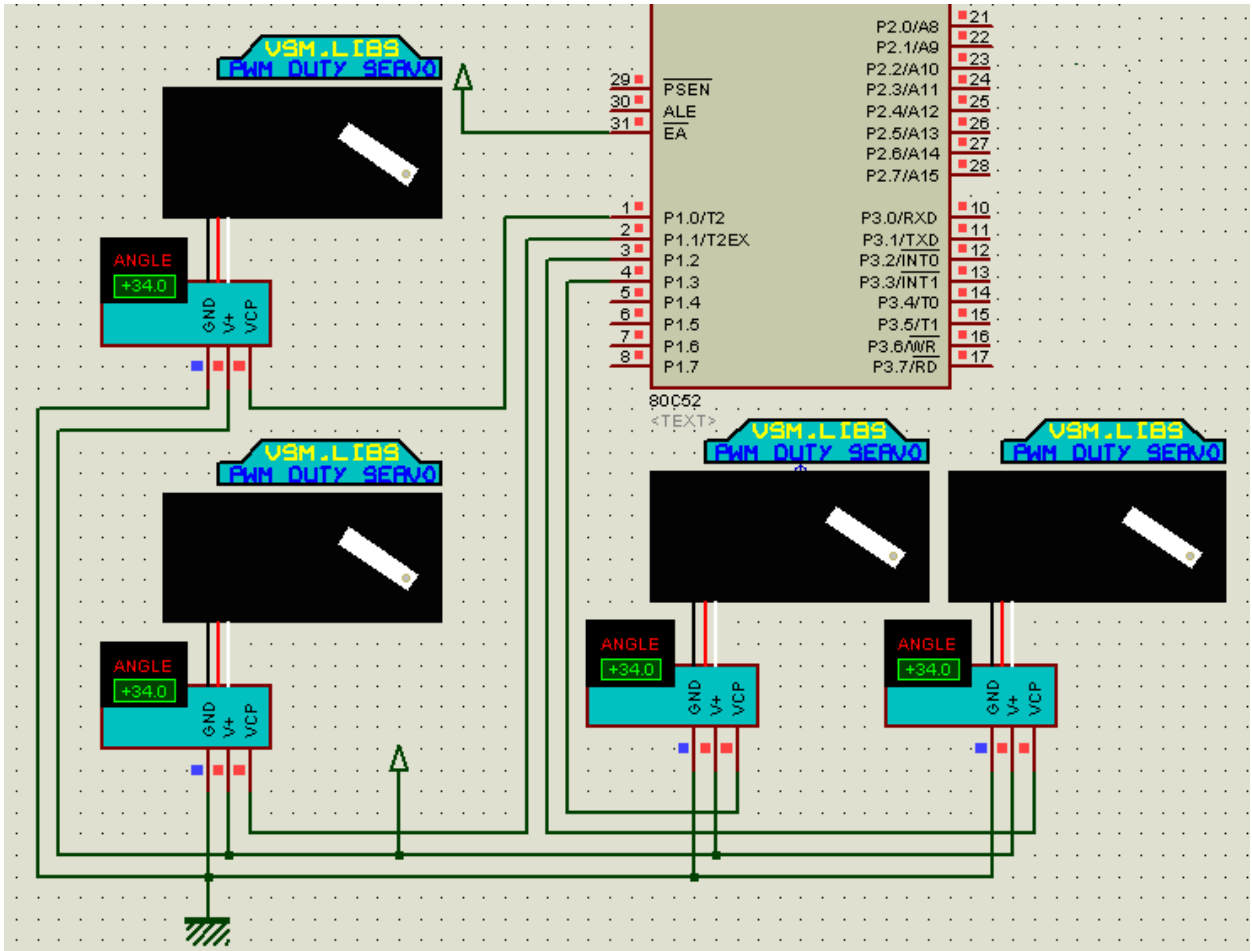
```
SERVOy = x
```

X مدت زمان یک بودن پایه را مشخص میکند و y شماره سروو است (1تا16)

مثال:

```
$regfile = "8052.DAT"

$crystal=12000000

Config SERVOS = 4 , SERVO1 = P1.0 , SERVO2 = P1.1 , SERVO3 = P1.2 , SERVO4 =
P1.3 , RELOAD = 100

SERVO1 = 8 '800 uS pulse

SERVO2 = 10 '1000 uS duration

SERVO1 = 4 '400 uS pulse
```

```
SERVO2 = 20 '2000 uS duration

End
```



اندازه گیری مقدار یک مقاومت یا خازن

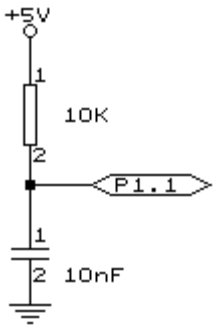
شما با استفاده از دستور زیر میتوانید مقدار ثابت زمانی مقاومت و خازنی که به پایه دلخواه میکرو avr متصل شده است را بدست اورید :

```
var = GETRC( pin , number )
```

var: یک متغییر از جنس word میباشد که مقدار ثابت زمانی در ان ریخته میشود.

Pin : نام پورتی است که خازن و مقاومت به ان متصل است (مانند porta یا portd).

Number :شماره پایه ای است که مقاومت و خازن به ان متصل شده است (مانند 1 یا 2 (این مقدار نمیتواند از 7 بیشتر شود)).



در مدارات مقاومت یا خازن که به اختصار به ان 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 ان 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 اورید ، مانند:

:

.