

به نام خدا

راه اندازی *lcd* با *bascom*

کلمات کلیدی

Lcd گرافیکی و کاراکتری، *AVR*، زبان بیسیک، *bascom* کامپایلر

چکیده

آموزش مختصری از *lcd* های کاراکتری و گرافیکی توسط *AVR*

T S G

WWW.TAAKTEEK.COM

برای بیشتر شما پیش اومده که برای پروژه ی خود نیاز به یک نمایشگر داشتید ، که حالا این نمایشگر میتواند *7segment* یا *lcd* باشد. *Lcd* ها نسبت به *78segment* ها قابل انعطاف ترند شما براحتی می توانید تمام حروف،اعداد و علائم را بر روی *lcd* نمایش دهید در صورتی که اگر بخواهید این کار را با *7segment* ها انجام دهید به مشکلات زیادی روبرو میشوید.

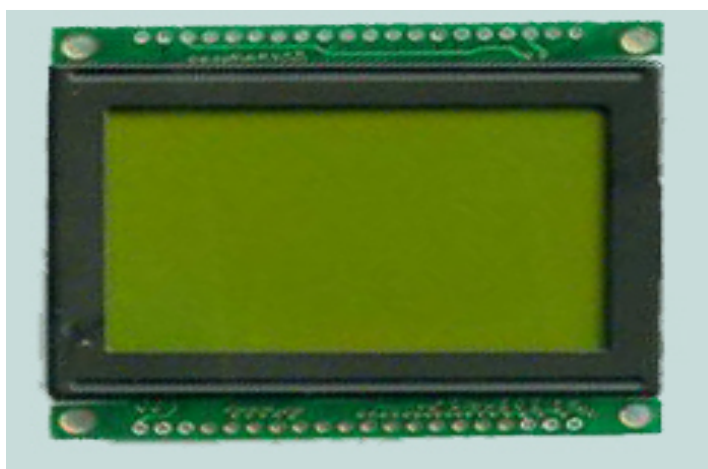
Lcd ها در کل به دو نوع تقسیم میشوند:

- کاراکتری
- گرافیکی

که ما در این مقاله قصد آموزش و کار کردن با *lcd* را به زبان بیسیک با کامپایلر *bascom* رو داریم.



Lcd کاراکتری



Lcd گرافیکی

برای برقراری ارتباط با *lcd* میتوان به دو صورت عمل کرد 1- پارالل 2- سریال ، که البته این دو نوع در همه *lcd* ها موجود نمیشود. بعضی ها پارالل و بعضی ها سریال را پشتیبانی میکنند ولی ناگفته نماند که مدل های مختلفی از *lcd* های گرافیکی هستند که دو نوع ارتباط را پشتیبانی میکنند . سعی میکنیم تا جایی که امکان دارد هم در مورد *lcd* گرافیکی ، کاراکتری و نحوه کار با آنها را برای شما توضیح دهیم .

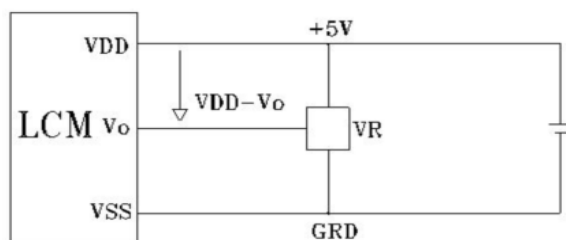
۱. Lcd کاراکتری

در حال حاضر در بازار الکترونیک ایران بیشترین و پر مصرف ترین *lcd* کاراکتری سایز 16×2 میباشد. که ما هم در این مقاله از این نوع *lcd* استفاده میکنیم. پایه های این نوع *lcd* ها تقریباً به صورت یک استاندارد یا قرارداد ساخته میشود و ترکیب قرار گیری پایه ها به یک صورت می باشد.

توضیح	<i>i/o</i>	سمبول	پایه
اتصال به زمین	—	<i>VSS</i>	1
اتصال به +5	—	<i>VDD</i>	2
تنظیم کنتراست	—	<i>VO</i>	3
رجیستر کنترلی اگر $RS=0$ رجیستر دستورالعمل فعال میشود. اگر $RS=1$ رجیستر داده فعال میشود.	<i>I</i>	<i>RS</i>	4
انتخاب نوشتن یا خواندن $R/W=0$ برای نوشتن ، $R/W=1$ برای خواندن اطلاعات	<i>I</i>	<i>R/W</i>	5
فعال ساز	<i>I</i>	<i>E</i>	6
باس 8 تایی اطلاعات و دستورالعمل	<i>I/O</i>	<i>DB0-DB7</i>	7/14

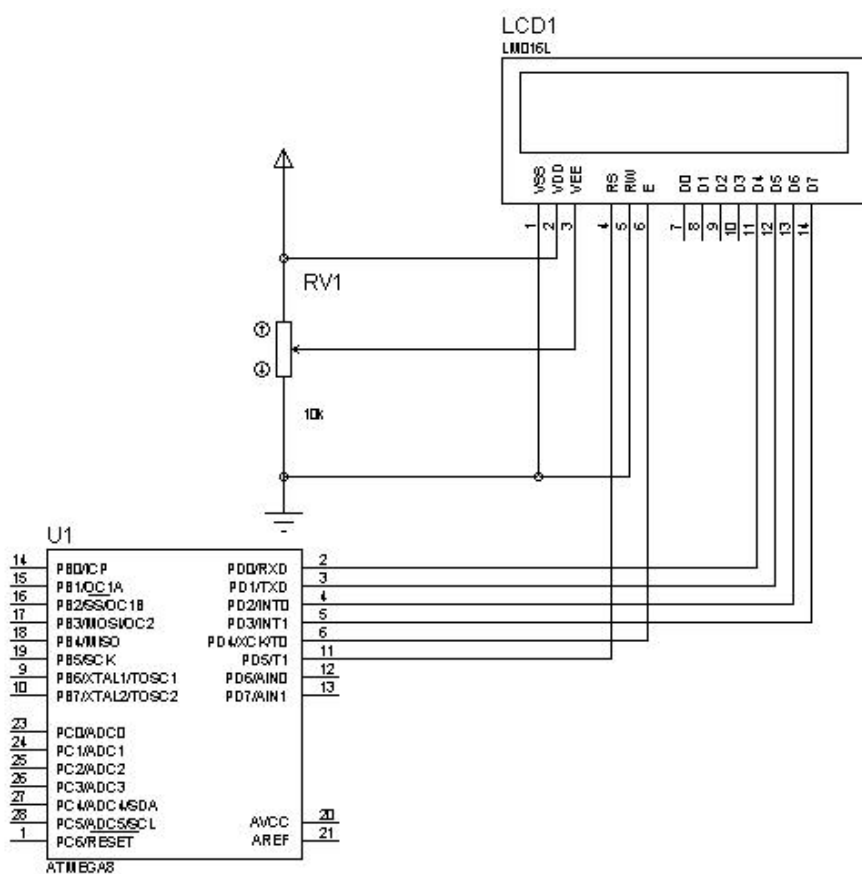
15	LED “+”		آند پس زمینه
/			
16	LED “-”	—	کاتد پس زمینه

نحوه اتصالات پایه 3 (vo) (vo)



شکل ۱

نحوه اتصال LCD به میکروکنترلر:



شکل ۲

مقدار مقاومت متغیر از 10 تا 20 کیلو اهم میباشد.

۱-۱. پیکربندی lcd کاراکتری

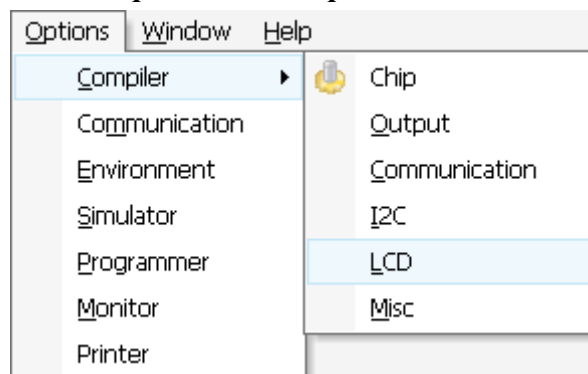
برای این پیکربندی و شروع به کار با *lcd* شما 2 راه دارید :

پیکربندی داخل *options* کامپایلر

پیکربندی داخل محیط برنامه نویسی کامپایلر

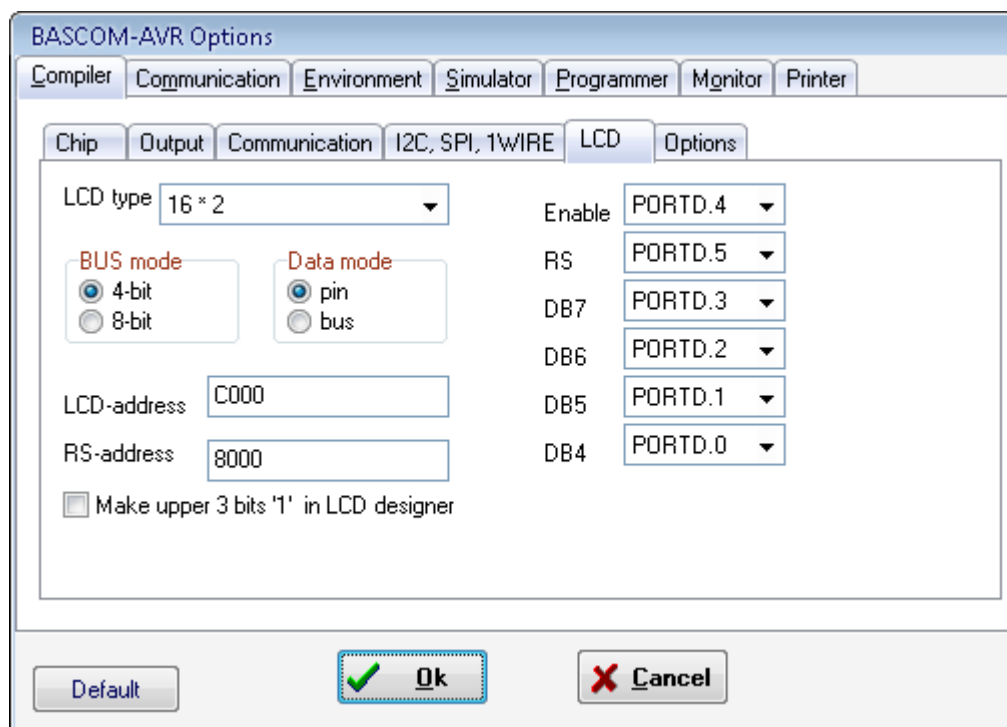
راه اول: برای این کار شما باید مراحل زیر را طی کنید:

Options => Compiler => Lcd



شکل ۳

اگر مسیر بالا را درست طی کنید باید صفحه زیر برای شما باز شود.



شکل ۴

که باید تنظیمات را مانند بالا تغییر دهید. (طبق سخت افزار شکل 2)
Lcd type مشخص کننده ساینز *lcd* میباشد.

Bus mode مشخص کننده نوع باس ارتباط دهی میباشد که اگر بخواهیم به صورت 8 پایه راه اندازی کنیم باید *8-bit* را انتخاب کنیم.

ما در اینجا از باس *4-bit* استفاده کرده ایم.

اگر قصد راه اندازی *lcd* در محیط های پر نویز و صنعتی را دارید پیشنهاد میشود از باس *8-bit* استفاده کنید.

راه دوم: برای پیکربندی در داخل برنامه باید دستور زیر را در اول برنامه قرار دهید و طبق سخت افزار مشخصات را وارد کنید.

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

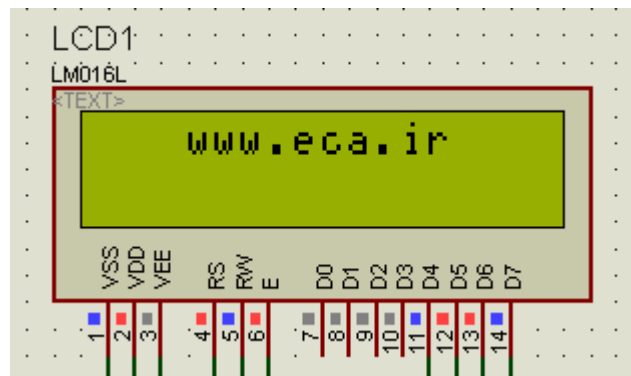
وقتی به این صورت *lcd* را پیکربندی میکنید باید توسط دستور زیر ساینز *lcd* را هم مشخص کنید.

*Config Lcd = 16 * 2*

۲-۱. برنامه نویسی

در اولین مرحله قصد نوشتن یک کاراکتر ثابت را داریم. برای این کار کافیه پیکربندی *lcd* را انجام دهید و کار را شروع کنید. (در اینجا ما پیکربندی را در *options* انجام داده ایم.)

```
"regfile = "M8def.dat$  
crystal = 8000000$  
Cursor Off  
Do  
Locate 1 , 4  
"Lcd "www.eca.ir  
Loop  
End
```



دستور *cursor* ۴ حالت دارد:

- ۱- *cursor on* که مکان نما را روشن میکند.
- ۲- *cursor off* مکان نما را خاموش میکند.
- ۳- *cursor blink* مکان نما را چشمک زن میکند.
- ۴- *cursor noblink* مکان نما را بدون چشمک زن میکند.

دستور *locate x,y* محل مکان نما را مشخص میکند، که x بیانگر سطر میباشد که در این نوع *lcd* ۱ یا ۲ میتواند باشد و y بیانگر ستون که در این نوع *lcd* از ۱ تا ۱۶ میتواند باشد.
دستور *lcd"eca"* برای نمایش یک کاراکتر ثابت میباشد.

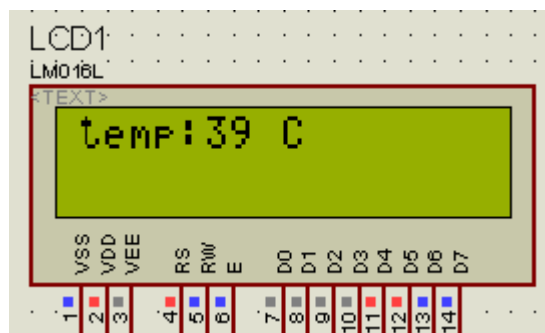
اگر بخواهیم یک متغیر را بر روی *lcd* نمایش دهیم باید به صورت زیر عمل کنیم:

```
"regfile = "M8def.dat$
crystal = 8000000$
Dim Milad As Byte
Cursor Off
Do
Locate 1 , 1
Lcd Milad
Wait 1
Incr Milad
Loop
End
```

برای نمایش چندین متغیر یا کاراکتر پشت سر هم باید از علامت *(semicolon)* استفاده کنیم:

Lcd "temp:" ; milad ; "C"

در مثال بالا مقدار دما در متغیر *milad* قرار داده شده است که بر روی *lcd* به صورت زیر نمایش داده میشود.

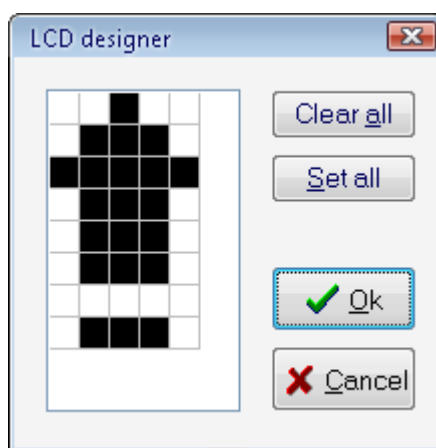


۳-۱. طریقه ساخت یک کاراکتر

برای ساخت یک کاراکتر باید از خود کامپایلر استفاده کنید برای این کار باید مراحل زیر را طی کنید:

Tools => Lcd designer

و یا میتوانید از کلید میانبر **ctrl+l** استفاده کنید.



اگر دقت کنید هر یک از کاراکترهای *lcd* تشکیل شده از یک ماتریس 8×5 که هر کدام از این 40 پیکسل قابلیت روشن و خاموش شدن را دارند.

برای مثال من در محیط *lcd designer* یک فلش طراحی کردم که توسط برنامه زیر میتوانیم این کاراکتر را بر روی *lcd* نمایش دهیم.

```
"regfile = "M8def.dat$
```

```
crystal = 8000000$
```

Deflcdchar 0 , 4 , 14 , 31 , 14 , 14 , 14 , 14 , 14

Cursor Off

Cls

)Lcd Chr(0

Do

Loop

End



توجه باید قبل از دستور *lcd chr* تمام *lcd* را پاک کنیم که برای این کار از دستور *cls* استفاده کنیم. دستورات *lcd* کاراکتری:

توضیح	دستور
برای پاک کردن کل صفحه <i>lcd</i>	<i>cls</i>
<i>Display on</i> صفحه نمایش را روشن، <i>Display off</i> صفحه نمایش را خاموش میکند.	<i>Display on/off</i>
<i>Cursor on</i> مکان نما را روشن، <i>Cursor off</i> مکان نما را خاموش، <i>Cursor blink</i> مکان نما را چشمک زن و <i>Cursor noblink</i> مکان نما را بدون چشمک زن میکند.	<i>cursor</i>
<i>Home upper</i> مکان نما را به اول سطر یک میبرد. <i>Home lower</i> مکان نما را به اول سطر دو میبرد.	<i>home</i>
<i>X</i> بیانگر سطر و <i>y</i> بیانگر ستون بوده که مکان نما را به آن	<i>Locate x,y</i>

	مکان منتقل میکند.
<i>Shift cursor</i>	<i>Shift cursor right</i> مکان نما را یک واحد به راست ، <i>shift cursor left</i> مکان نما را یک واحد به چپ میبرد.
<i>Shift lcd</i>	<i>Shift lcd right</i> کل صفحه نمایش را یک واحد به راست ، <i>shift lcd left</i> کل صفحه نمایش را یک واحد به چپ منتقل میکند.
<i>Upperline</i>	مکان نما را به خط بالا منتقل میکند.
<i>lowerline</i>	مکان نما را به خط پایین منتقل میکند.
<i>Lcd chr(0)</i>	کاراکتر شماره 0 را که توسط <i>lcd designer</i> درست شده است را نمایش میدهد.

نکته: اگر از دستور *shift lcd* برای شیفت دادن یک متن استفاده میکنید باید آن را داخل یک حلقه قرار دهید زیرا این دستور فقط یک واحد شیفت میدهد.
به برنامه زیر توجه کنید:

```
"regfile = "M8def.dat$
crystal = 8000000$
Dim Milad As Byte
Milad = 0
Cursor Off
Cls
Locate 1 , 14
"Lcd "eca
Do
Incr Milad
Shiftlcd Right
Waitms 500
Loop Until Milad = 16
End
```

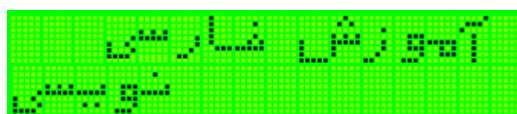
برای درست کردن حلقه میتوانید از برنامه زیر هم استفاده کنید:

```
For Milad = 0 To 16
Shiftlcd Right
Waitms 500
```

۴-۱. فارسی نویسی در lcd کاراکتری

بعضی مواقع برای شما پیش آمده که بخواید روی *lcd* فارسی نویسی کنید ولی کمی که فکر میکنید می بینید نشدنی.

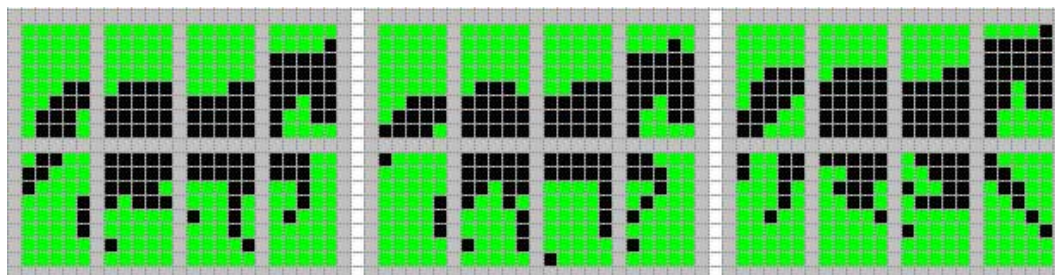
این کار شدنی ولی مشکلات خودشو داره، برای این کار کافیه شما تمام حروف فارسی را در محیط *lcd* *dedigner* طراحی کنید و آنها را در در برچسب های مختلف به کار ببرید و هر موقع که به حرفی نیاز داشتید آن برچسب را فراخوانی کنید و کاراکتر مورد نظر را نمایش دهید.



نکته این روش را برای زمانی استفاده بکنید که در هر لحظه فقط بر روی *lcd* کمتر از ۸ کاراکتر بخواید نمایش دهید.

برای نمایش فایل متحرک هم میتوانید چند کاراکتر را درست کنید و آنها را با تاخیر معینی نمایش دهید.

به برنامه زیر توجه کنید:



```
$regfile = "M8def.dat"  
$crystal = 8000000  
Dim Milad As Byte  
Cursor Off  
Cls
```

Do

Gosub Horse1

Gosub Lcd_pictur

Gosub Horse2

Gosub Lcd_pictur

Gosub Horse3

Gosub Lcd_pictur

Loop

End

Lcd_pictur:

Locate 1 , 1

Lcd Chr(0) ; Chr(1) ; Chr(2) ; Chr(3)

Locate 2 , 1

Lcd Chr(4) ; Chr(5) ; Chr(6) ; Chr(7)

Waitms 300

Return

Horse1:

Deflcdchar 0 , 32 , 32 , 32 , 32 , 3 , 7 , 14 , 14

Deflcdchar 1 , 32 , 32 , 32 , 32 , 15 , 31 , 31 , 31

Deflcdchar 2 , 32 , 32 , 32 , 32 , 3 , 31 , 31 , 31

Deflcdchar 3 , 32 , 1 , 31 , 31 , 31 , 27 , 19 , 16

Deflcdchar 4 , 12 , 24 , 16 , 1 , 1 , 1 , 32 , 32

Deflcdchar 5 , 31 , 31 , 30 , 23 , 32 , 32 , 16 , 32

Deflcdchar 6 , 31 , 31 , 3 , 2 , 18 , 2 , 1 , 32

Deflcdchar 7 , 28 , 28 , 4 , 4 , 8 , 32 , 32 , 32

Return

Horse2:

Deflcdchar 0 , 32 , 32 , 32 , 32 , 32 , 7 , 15 , 30

Deflcdchar 1 , 32 , 32 , 32 , 32 , 14 , 31 , 31 , 31

Deflcdchar 2 , 32 , 32 , 32 , 32 , 7 , 31 , 31 , 31

Deflcdchar 3 , 32 , 2 , 31 , 31 , 31 , 27 , 19 , 16

Deflcdchar 4 , 16 , 32 , 32 , 1 , 1 , 1 , 32 , 32

Deflcdchar 5 , 31 , 31 , 27 , 18 , 1 , 1 , 17 , 32

Deflcdchar 6 , 31 , 31 , 1 , 1 , 1 , 1 , 32 , 32

Deflcdchar 7, 24, 28, 4, 8, 16, 32, 16, 32

Return

Horse3:

Deflcdchar 0, 32, 32, 32, 7, 15, 14, 28, 24

Deflcdchar 1, 32, 32, 32, 15, 31, 31, 31, 31

Deflcdchar 2, 32, 32, 32, 3, 31, 31, 31, 31

Deflcdchar 3, 1, 31, 31, 31, 27, 27, 16, 16

Deflcdchar 4, 19, 19, 2, 2, 4, 32, 32, 32

Deflcdchar 5, 31, 7, 14, 6, 1, 32, 32, 32

Deflcdchar 6, 15, 7, 19, 15, 32, 16, 32, 32

Deflcdchar 7, 16, 24, 12, 4, 2, 1, 32, 32

Return

در برنامه بالا برای شبیه سازی حرکت اسب 3 تصویر را در نظر گرفته که این 3 تصویر را با تاخیر 300 میلی ثانیه پشت سر هم نمایش میدهد. برای ساختن هر تصویر هم از 8 کاراکتر استفاده کرده است.

۲. Lcd گرافیکی

یک نوع دیگر از *lcd* ها ، گرافیکی ها هستند که امروزه طرفداران خاص خود را دارد. این *lcd* ها در سایز های مختلف تولید میشوند که بیشتر این سایز ها هم در بازار الکترونیک ایران به راحتی یافت میشود.

128*64
128*128
240*64
240*128
240*240

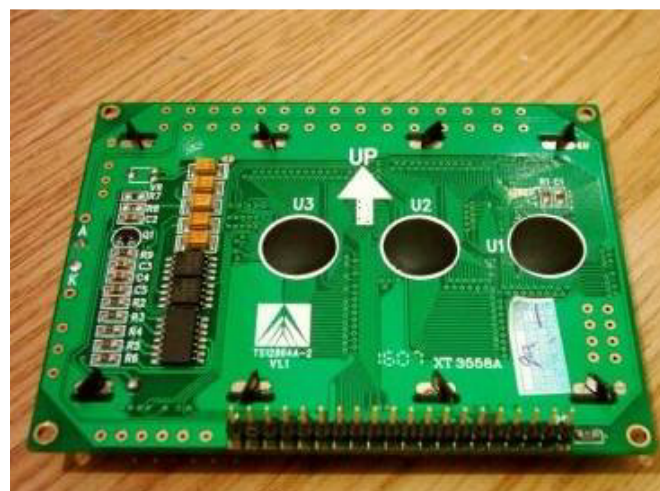
سایز های موجود در بازار

این نوع *lcd* ها به 2 نوع تقسیم میشوند:

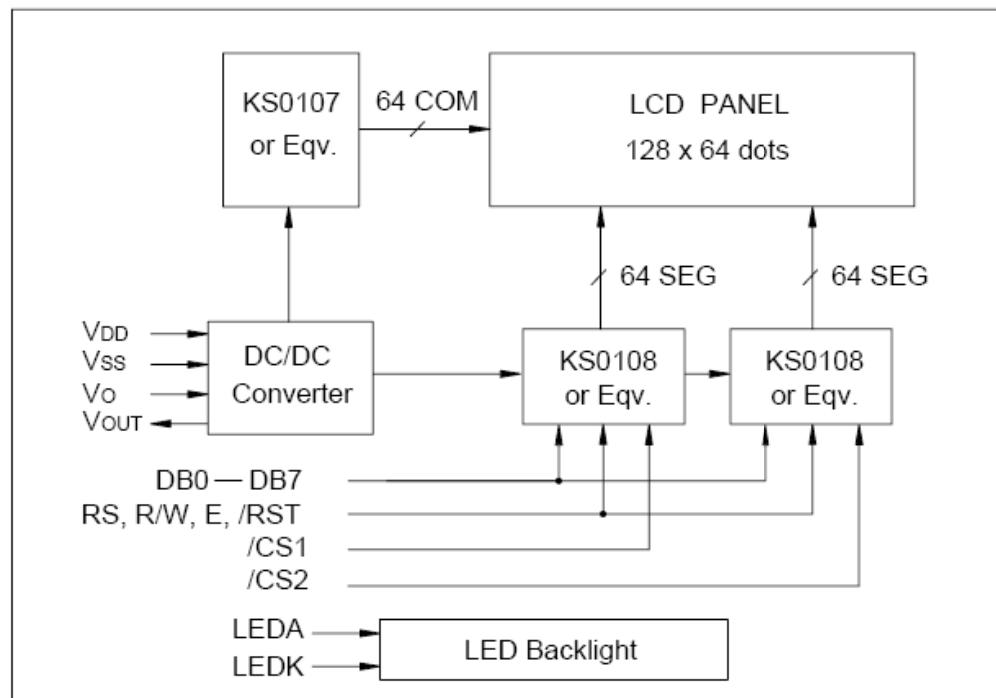
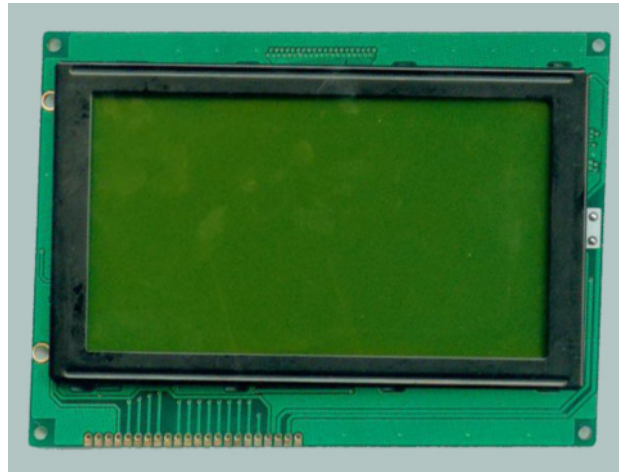
lcd های *sed* با چیپ *ks108*

lcd های *Toshiba* با چیپ *t6963*

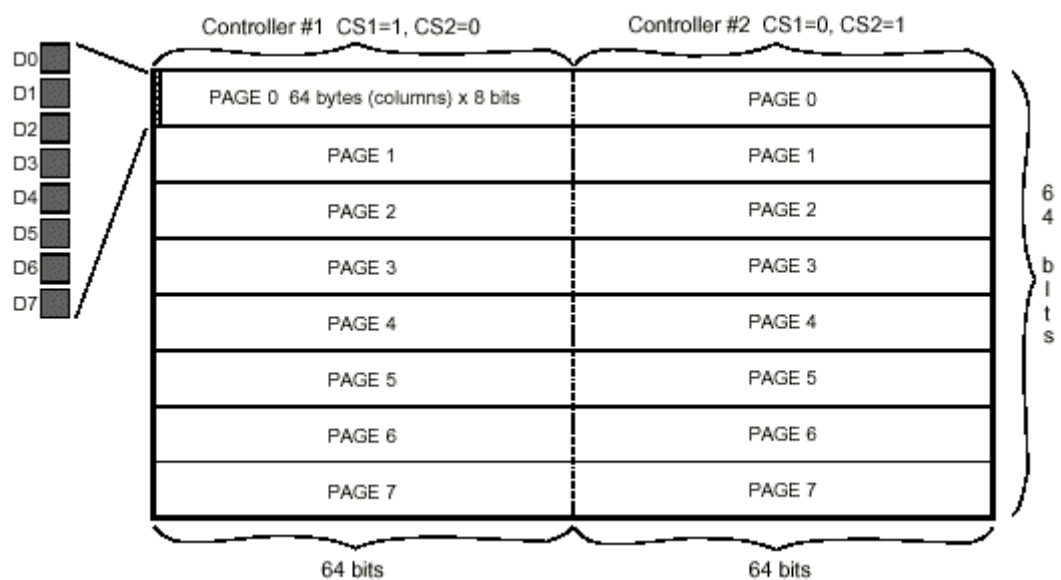
از لحاظ ظاهری هیچ فرقی با هم نمیکنند ولی از لحاظ قیمت *lcd* های *sed* ارزان تر و از لحاظ سرعت *lcd* های *Toshiba* دارای سرعت بیشتری هستند.



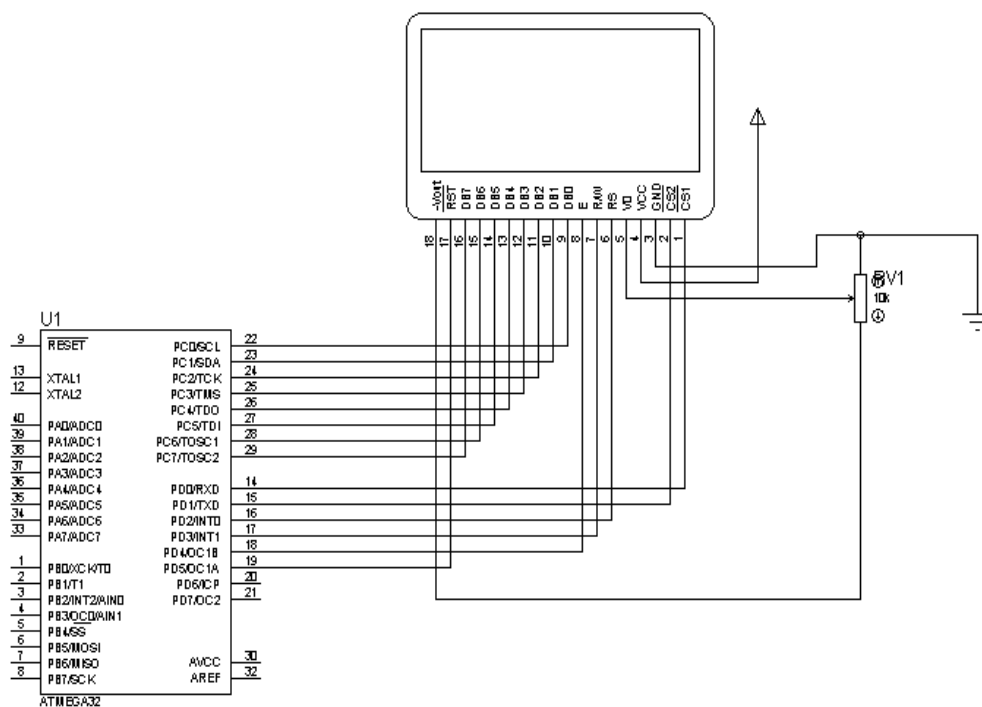
نمایی از پشت یک *sed lcd*



تصویر بالا بلوک دیاگرام *lcd* های *Sed* را نمایش میدهد، که این برای سایز 128×64 میباشد. همانطور که مشاهده میکنید در واقع ما دو *lcd* گرافیکی با سایز 64×64 داریم که در کنار هم قرار گرفته شده است که این *lcd* ها 2 پایه *ce* دارد که *ce 0* برای *lcd* سمت چپ و *ce 1* برای *lcd* سمت راست.



۲-۱. طریقه اتصالات *lcd* با میکروکنترلر



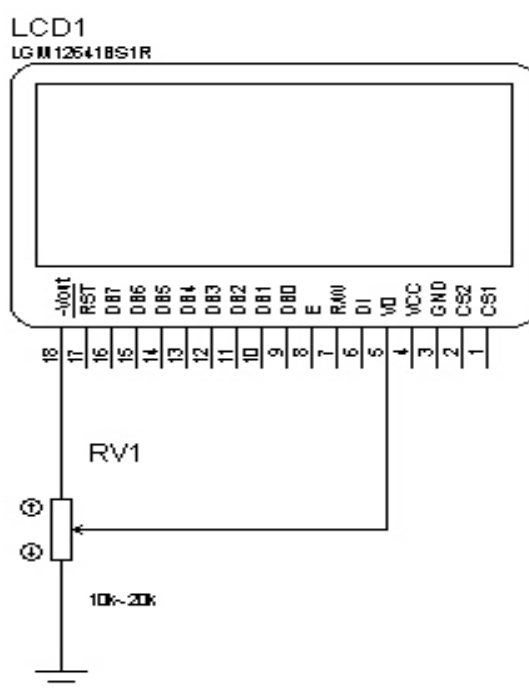
پایه های *lcd* گرافیکی *sed* :

<i>pin-no.</i>	<i>symbol</i>	<i>level</i>	توضیح
۱	<i>Vss</i>	—	تغذیه مدارات داخلی <i>lcd</i> 0 v -----
۲	<i>Vdd</i>	—	تغذیه مدارات داخلی <i>lcd</i> $+5\text{ v}$ -----
۳	<i>vo</i>	—	تنظیم کنتراست
۴	<i>d/i</i>	<i>h/l</i>	اگر <i>H</i> باشد ورودی دیتا را می پذیرد. اگر <i>l</i> باشد ورودی دستورالعمل را می پذیرد.
۵	<i>r/w</i>	<i>h/l</i>	اگر <i>h</i> باشد دیتا را میتوان خواند. اگر <i>l</i> باشد دیتا را میتوان نوشت.
۶	<i>e</i>	<i>H,h=>l</i>	فعال ساز
۷	<i>Db0</i>	<i>h/l</i>	باس دیتا
۸	<i>Db1</i>	<i>h/l</i>	
۹	<i>Db2</i>	<i>h/l</i>	
۱۰	<i>Db3</i>	<i>h/l</i>	
۱۱	<i>Db4</i>	<i>h/l</i>	
۱۲	<i>Db5</i>	<i>h/l</i>	
۱۳	<i>Db6</i>	<i>h/l</i>	
۱۴	<i>Db7</i>	<i>h/l</i>	
۱۵	<i>Ce 1</i>	<i>H</i>	فعال ساز چیپ ۱
۱۶	<i>Ce 2</i>	<i>h</i>	فعال ساز چیپ ۲
۱۷	<i>Res</i>	<i>l</i>	بازنشانی با سطح صفر
۱۸	<i>Vee</i>	—	تولید ولتاژ منفی جهت تنظیم کنتراست
۱۹	<i>K</i>	—	کاتد <i>led</i> نور پس زمینه

۲۰	a	—	آند <i>led</i> نور پس زمینه
----	---	---	-----------------------------

توجه: ترتیب قرار گیری پایه های *lcd* گرافیکی برای هر کمپانی متفاوت است و هر *lcd* ترکیب بندی خاص خود را دارد ، پس همیشه قبل از راه اندازی کاتالوگ *lcd* را مطالعه کنید .

۲-۲. طریقه قرار گیری پتانسیومتر کنتر است:



۲-۳. پیکربندی *lcd* های *sed*:

برای پیکر بندی این نوع *lcd* (ما در اینجا از *lcd* ی *sed 128*64* استفاده کرده ایم) باید دستور زیر را ابتدای برنامه خود بنویسید:

Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd = 2 , Rd = 3 , Reset = 4 , Enable = 5

*Graphlcd = 128 * 64sed* باید در این قسمت ساینز *lcd* خود را بنویسید.

Dataport مشخص کننده پورتی از میکرو میباشد که به پایه های دیتای *lcd* متصل شده است.

Ce این پایه مخصوص چیپ *lcd* ی سمت چپ میباشد. که به پین *b.1* متصل شده است.

Ce1 این پایه مخصوص چیپ **lcd** ی سمت راست میباشد. که به پین **b.0** متصل شده است.

CD این پایه در **lcd** ها با **D/I** مشخص شده که **CD** به معنی **Code/Data** میباشد و **D/I** به معنی **Data/Instruction** که باید به پین **b.2** متصل شود. (این پایه دقیقا مانند پایه **RS** در **lcd** های کاراکتری میباشد).

RD این پایه باید به پین **b.3** متصل شود. این پایه در **lcd** ها با **R/W** نمایش داده شده است.

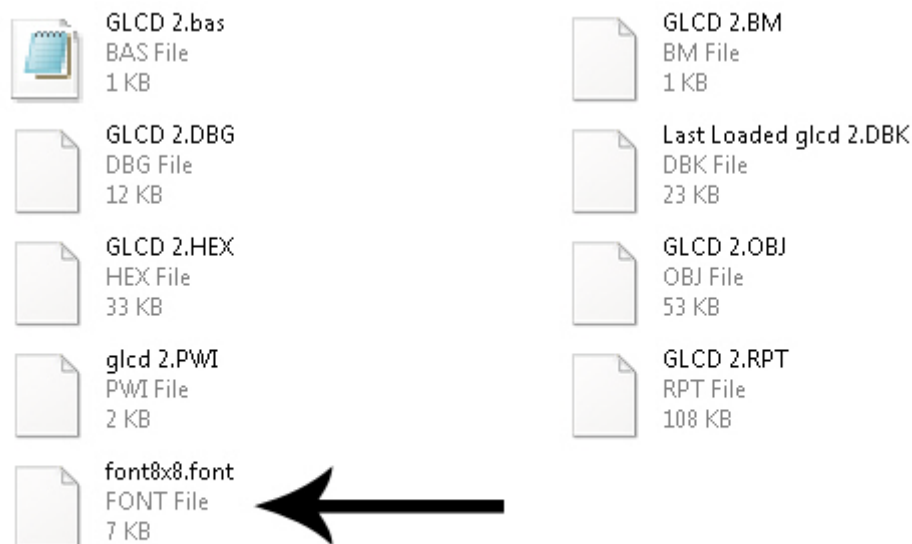
Reset این پایه به پین **b.4** متصل شده است.

Enable این پایه به پین **b.5** متصل شده است. عملکرد این پایه تقریبا همانند پایه **E** در **lcd** های کاراکتری میباشد.

بعد از اینکه **lcd** را پیکربندی کردیم باید فونت و کتابخانه لازم جهت **lcd sed** را در برنامه معرفی کنیم.

به آدرس زیر بروید و فایل **font8x8.font** را پیدا کنید و آن را در پوشه ای که فایل برنامه نویسی خود را در آن ذخیره کرده اید قرار دهید.

C:\Program Files\MCS Electronics\BASCOM-AVR\SAMPLES



بعد از اینکه این فایل را در محل ذخیره برنامه قرار دهید باید این فونت را در برنامه فراخوانی کنید ،

برای این کار باید دستور زیر را ابتدای برنامه خود قرار دهید.

\$include "font8x8.font"

حالا باید کتابخانه *lcd* های *ks108* را در برنامه فراخوانی کنید، برای این منظور باید دستور زیر را ابتدای برنامه خود قرار دهید.

\$lib "glcdKS108.LBX"

برنامه نویسی *lcd* گرافیکی با چیپ *ks108*

خوب بعد از اینکه مراحل قبل را انجام دادید نوبت به شروع برنامه نویسی میشود در ابتدا یک برنامه ساده مینویسیم و تمام نکات را در این برنامه برای شما توضیح میدهم.

\$regfile = "M32def.dat"

\$crystal = 8000000

\$lib "glcdKS108.LBX"

\$include "font8x8.font"

Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd = 2 , Rd = 3 , Reset = 4 , Enable = 5

SetFont Font8x8

Cls

Lcdat 1 , 1 , "www.eca.ir"

Do

Loop

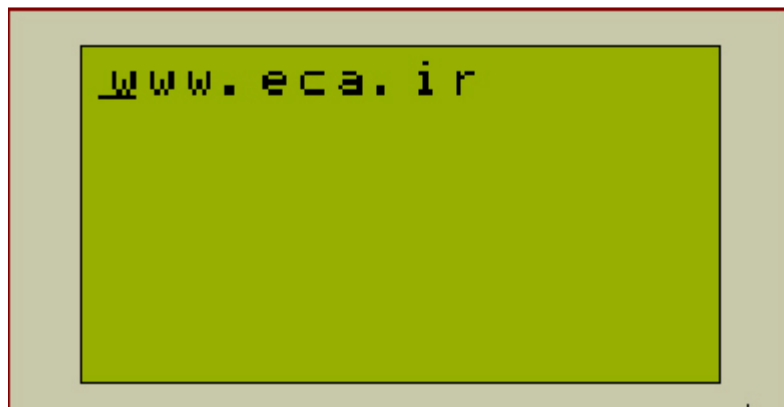
End

نکته : اگر شما کتابخانه *ks108* را در برنامه فراخوانی نکنید برنامه اجرا نخواهد شد.

اگر فونت *font8x8.font* در ابتدای برنامه فراخوانی نکنید برنامه اجرا نخواهد شد.

باید در ابتدای برنامه *font8x8.font* را *set* کنید.

اگر دستور *setfont font8X8* را در ابتدای برنامه *Set* نکنید برنامه شما اجرا نمیشود.



۲-۴. نمایش تصویر بر روی lcd گرافیکی:

برای نمایش تصویر بر روی lcd گرافیکی 2 راه وجود دارد:

نمایش فایل *bgr* بر روی *lcd*

نمایش کد عکس مورد نظر (در این آموزش از روش اول استفاده شده)

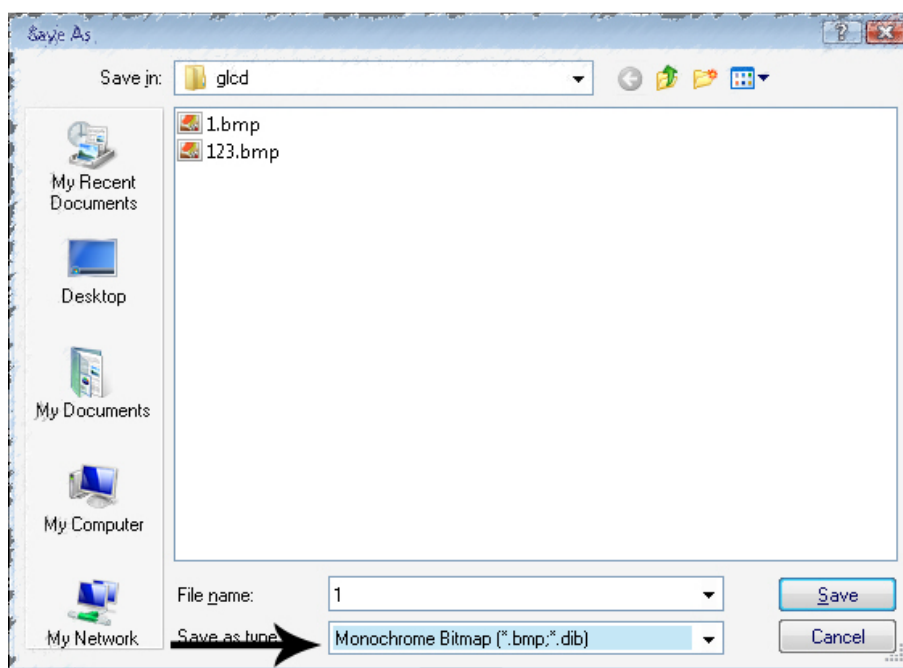
روش اول را با خود کامپایلر میتوان انجام داد و نیاز به نرم افزار جانبی دیگری ندارد. برای نمایش یک

تصویر بر روی *lcd* باید مراحل زیر را انجام دهید:

ابتدا عکس خود را با یک *editor* به اندازه سایز *lcd* خود در آورید.

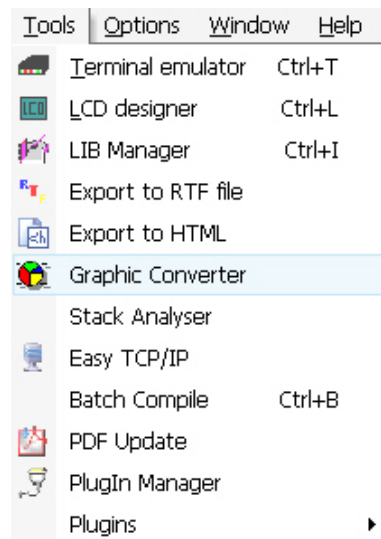
عکس را پسوند *Monochrome Bitmap (*.bmp - *.dib)* ذخیره کنید. (این پسوند در نرم افزار *Paint*

موجود میباشد)



وارد نرم افزار بسکام شوید.

از منوی *Tools* گزینه *Graphic Converter* را انتخاب کنید.



باید برای شما صفحه زیر ظاهر شود.



در قسمت **LCD Type** باید سایز **lcd** خود را انتخاب کنید.

در قسمت **Font** باید اندازه فونت خود را انتخاب کنید. باید روی **8*8** باشد فونت **6*8** برای **lcd** های **Toshiba** میباشد.

چون **lcd** ما از نوع **sed** میباشد باید تیک گزینه **sed** را فعال کنیم.

برای وارد کردن عکس به این محیط باید از گزینه **Loud** استفاده کنیم.

بعد از تنظیم کردن تمام قسمتها باید روی گزینه **SAVE** کلیک کنید و عکس خود را در کنار برنامه خود ذخیره کنید.

در این مرحله باید از دستور **Showpic x, y, lable** برای نمایش عکس استفاده کنیم.

به مثال زیر توجه کنید.

```
$regfile = "M32def.dat"  
$crystal = 8000000  
$lib "glcdKS108.LBX"  
$include "font8x8.font"  
Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd  
= 2 , Rd = 3 , Reset = 4 , Enable = 5  
Cursor off  
SetFont Font8x8  
Cls  
Showpic 0 , 0 , Pic  
Do  
Loop  
End  
Pic:  
$bgf "1.bgf"
```



وقتی از دستور *Showpic x , y , lable* استفاده میکنیم باید با توجه به محل قرار گیری عکس به جای *x,y* مقدار عددی بگذاریم. به جای *lable* هم باید نام یک برجسبی را بنویسیم و در آن برجسب مورد نظر دستور *\$Bgf "1.bgf"* را بنویسیم.

شما میتوانید به این صورت چندین عکس را بر روی *lcd* نمایش دهید.

۲-۵. *Invert* کردن عکس ها

بعضی مواقع پیش می آید که شما مجبور به *invert* کردن یک عکس میشوید برای این منظور شما میتوانید از همان دستور قبلی برای نمایش تصویر خود استفاده کنید ، به مثال زیر توجه کنید:

```
$regfile = "M32def.dat"
$crystal = 8000000
$lib "glcdKS108.LBX"
$include "font8x8.font"
Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd
= 2 , Rd = 3 , Reset = 4 , Enable = 5
Cursor off
SetFont Font8x8
Cls
Showpic 0 , 0 , Pic , 1
Do
Loop
End
Pic:
$bgf "1.bgf"
```



اگر کمی دقت کنید متوجه میشوید که در برنامه بالا یک تغییر جزئی به وجود آمده. ما قبلا از دستور *Showpic 0 , 0 , Pic* استفاده کردیم و تصاویر خودمان را با آن نمایش دادیم و حالا میبینید که در برنامه بالا دستور کمی فرق کرده *Showpic 0 , 0 , Pic , 1* ، با اضافه کردن *1*، به آخر دستور مشاهده میشود که تصویر ما به صورت *invert* شده نمایش داده شده است.

این افکت برای نوشته هم عملی هست فقط کافیه دستور را مانند زیر تغییر دهید:

Lcdat 1 , 1 , "www.eca.ir" , 1

۲-۶. نمایش فایل متحرک

برای نمایش فایل متحرک (gif) بر روی *lcd* باید چندی عکس را با تاخیر مناسب پشت سر هم نمایش دهید. تمام مراحل مانند نمایش یک عکس میباشد.

```
$regfile = "M32def.dat"
$crystal = 8000000
$lib "glcdKS108.LBX"
$include "font8x8.font"
Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd
= 2 , Rd = 3 , Reset = 4 , Enable = 5
Dim X As Byte , Y As Byte
Cursor off
SetFont Font8x8
Cls
Do
Incr A

Showpic 1 , 1 , Pic4
Waitms 100
Cls Graph
Showpic 1 , 1 , Pic5
Waitms 100
Cls Graph
Showpic 1 , 1 , Pic6
Waitms 100
Cls Graph
Showpic 1 , 1 , Pic7
Waitms 100
Cls Graph
Showpic 1 , 1 , Pic8
Waitms 100
Cls Graph
Loop Until A = 30
Do
Loop
End
Pic4:
$bgf "006.bgf"
```

Pic5:

\$bgf "007.bgf"

Pic6:

\$bgf "008.bgf"

Pic7:

\$bgf "009.bgf"

Pic8:

\$bgf "010.bgf"

در این برنامه از 5 عدد عکس استفاده شده که با تاخیر 100 میلی ثانیه نمایش داده میشود.
عکس ها به ترتیب زیر نمایش داده میشوند:



که در آخر یک فایل متحرک را از پرچم ایران برای بیننده نمایان میکند.

نکات

۱- وقتی از دستور *"text"* , *y* , *Lcdat x* استفاده میکنید مقدار *X* از *I* تا *8* متغییر میباشد، یعنی نوشته شما در 8 سطر قابل نمایش است و مقدار *Y* از 0 تا 127 متغییر است ، یعنی شما باید با مقدار دهی به *Y* محل نمایش نوشته خود را برحسب پیکسل مشخص کنید.

در عمل خواهید دید که **LCD** نوشته شما را واضح نمایش نمیدهد در صورتی که شما تمام مراحل را درست انجام داده اید ، پایه های **CE 1 , CE 2** با آنچه در پشت **lcd** گفته است فرق دارد در واقع شما باید اتصال این پایه ها را برعکس آن چیزی که در برنامه مشخص کردید برقرار کنید .
گذاشتن پتانسیومتر جهت کنتراست الزامی است و اگر شما این کار نکرده باشید **lcd** هیچ چیزی را نمایش نمیدهد .

۲-۷. پاک کردن عکس ها با افکت:

وقتی ما از دستور **showpic** برای نمایش تصاویر خودمان استفاده میکنیم فقط به قادر نمایش و پاک کردن تصویر هستیم و نمیتوانیم تصویر خود را با افکت های مختلف پاک کنیم. برای این کار میتوانیم از دستورات **line , Circle , pset** استفاده کنیم و عکس خود را با افکتهای زیبا پاک کنیم .
برای مثال شما عکسی را بر روی **lcd** نمایش داده اید و بعد از گذشت زمانی میخواهید آن را پاک کنید اگر از دستور **CLS** استفاده کنیم تصویر به سرعت از روی **lcd** محو میشود ولی ما نمی خواهیم به این صورت باشد .

میتوانیم عکس را به آرامی از سمت راست ، چپ ، بالا و پایین به طرف مخالف پاک کنیم .

مثال

پاک شدن عکس از سمت چپ به راست

For x = 0 To 128

Line(x , 0) -(x , 64) , 0

Waitms 10

Next Y

مقدار متغیر **X** از 0 تا 128 متغیر است و دستور **Line** به خط از مختصات

(X , 0) تا **(X , 128)** رسم میکند و بعد از گذشت 10 میلی ثانیه مقدار **X** یکی اضافه میشود .

اگر بجای **Line(x , 0) -(x , 64) , 0** از **Line(x , 0) -(x , 64) , 1** استفاده کنید به جای اینکه عکس را پاک کند ، کل صفحه نمایش را روشن میکند .

پاک شدن عکس از بالا به پایین :

For Y = 0 To 64

```
Line (0 , Y) - (128 , Y) , 0
Waitms 10
next
```

پاک شدن از راست به چپ:

```
X = 128
Do
Line(x , 0) -(x , 64) , 0
Waitms 10
Decr X
Loop Until X = 0
```

پاک کردن از پایین به بالا:

```
Y = 64
Do
Line(0 , Y) -(128 , Y) , 0
Waitms 10
Decr Y
Loop Until Y = 0
```

ساخت این نوع افکت ها بستگی به سلیقه هر شخص دارد، شما میتوانید تعداد زیادی افکت را درست کنید و به عنوان یک برجسب در آورید و در برنامه آن را فراخوانی کنید.

ذخیره فایل *bgf* در *EEPROM* میکروکنترلر:

اگر شما بخواهید از تعداد زیادی عکس استفاده کنید ممکن است که حافظه *FLASH* میکروی شما پر شده و دیگر فضای خالی برای نوشتن برنامه خود ندارید. دراین حالت شما میتوانید عکس های خود را در *EEprom* میکرو ذخیره کنید و بعد آن را نمایش دهید.

به مثال زیر توجه کنید:

```
$regfile = "M32def.dat"
$crystal = 8000000
```

```

$lib "glcdKS108.LBX"
$include "font8x8.font"
Config Graphlcd = 128 * 64sed , Dataport = Portd , Controlport = Portb , Ce = 1 , Ce2 = 0 , Cd
= 2 , Rd = 3 , Reset = 4 , Enable = 5
Cursor off
SetFont Font8x8

```

```

$eeprom
Pic:
$bgf "1.bgf"
$data

```

```

Cls
Showpice 0 , 0 , Pic
End

```

وقتی شما از دستور *\$eeprom* استفاده میکنید حافظه شما از *FLASH* به *EEPROM* تعویض میشود و ما میتوانیم فایل *BGF* خود را در *EEPROM* ذخیره کنیم و بعد از ذخیره سازی باید دستور *\$data* را به کار ببرید تا حافظه به حالت اول خود بازگردد (از *EEPROM* به *Flash* برود). شما میتوانید تا جایی که حافظه *EEPROM* شما ظرفیت داشته باشد عکس های خود را در آن ذخیره کنید.

وقتی شما عکس های خود را در *EEPROM* ذخیره میکنید برای نمایش عکس باید از دستور *Showpice* *0, 0, Pic* استفاده کنید.

۲-۸. دستورات LCD گرافیکی SED:

توضیحات	دستور
پاک کردن کل صفحه نمایش.	<i>CLS</i>
پاک کردن نوشته های روی صفحه نمایش.	<i>CLS TEXT</i>
پاک کردن تصاویر، پیکسل، دایره و خطوط روی صفحه نمایش.	<i>CLS GRAPH</i>
یک پیکسل را در مختصات X, Y روشن و یا خاموش میکند. اگر <i>value</i> برابر 1 باشد روشن و اگر 0 باشد خاموش میکند.	<i>PSET X, Y, value</i>
یک خط از مختصات $(x0, y0)$ تا $(x1, y1)$ رسم میکند. اگر <i>color</i> برابر 1 باشد خط روشن و اگر 0 باشد خط خاموش رسم میکند.	<i>LINE(x0,y0) – (x1,y1), color</i>
یک دایره از مختصات $(x0, y0)$ به شعاع <i>radius</i> و با <i>color</i> صفر یا یک رسم میکند.	<i>CIRCLE(x0,y0) , radius, color</i>
عکسی که در <i>label</i> وجود دارد را در مختصات x, y نمایش میدهد.	<i>SHOWPIC x, y, label</i>
عکسی که در <i>label</i> وجود دارد را در مختصات x, y نمایش میدهد. <i>Label</i> مورد نظر در <i>EEprom</i> قرار دارد.	<i>SHOWPICE x, y, label</i>
برای معرفی فایل <i>BGF</i> به برنامه به کار میرود.	<i>\$BGF "file"</i>
کاراکتر یا متغیری را در مختصات x, y نمایش میدهد. اگر <i>inv</i> برابر 0 باشد نوشته ساده و اگر <i>inv</i> برابر 1 باشد نوشته به صورت <i>INVERT</i> شده نمایش داده میشود.	<i>LCDAT y, x, var [, inv]</i>
فونتی که در اول برنامه معرفی کردیم را <i>SET</i> میکند.	<i>SETFONT font</i>
مکان نما را به مختصات x, y میبرد.	<i>LOCATE y, x</i>
وضعیت <i>CURSOR</i> را مشخص میکند.	<i>CURSOR ON / OFF BLINK / NOBLINK</i>

۲-۹. پیکر بندی lcd های Toshiba با چیپ T6963 :

پیکربندی این نوع lcd ها با lcd های sed فرق آن چنانی ندارد. تفاوت lcd های sed از لحاظ ساختار پایه ها با lcd های Toshiba به قرار زیر است.

<i>sed</i>	<i>Toshiba</i>
<i>D/I</i>	<i>FG</i>
<i>R/W</i>	<i>FS</i>
<i>E</i>	<i>RD</i>
<i>CE 0</i>	<i>WR</i>
<i>CE 1</i>	<i>CE</i>
<i>RST</i>	<i>CD</i>
	<i>RST</i>

LCD های SED ۲ پایه برای CE دارند LCD های Toshiba یک ce دارند.

فرمان *read/writhe* در lcd های sed با یک پایه صادر میشود ولی در lcd های Toshiba ۲ پایه مستقل از هم دارد.

انتخاب سائز فونت در lcd های Toshiba با یک پایه انتخاب میشود ولی در lcd های sed از طریق برنامه تعیین میشود.

برای پیکر بندی باید کد زیر را در ابتدای برنامه بنویسید:

$Config\ Graphlcd = 128 * 64$, $Dataport = Portc$, $Controlport = Portd$, $Ce = 3$, $Cd = 0$, $Wr = 2$, $Rd = 1$, $Reset = 4$, $Fs = 5$, $Mode = 8$

CE به پین B.3 متصل شده است، این پایه برای فعال سازی چیپ داخلی LCD میباشد.

CD به پین B.0 متصل شده است، این پایه به معنی Code/Data می باشد. که برای انتخاب ورود دیتا یا دستورالعمل می باشد.

WR به پین B.2 متصل شده است، این پایه برای فعال سازی نوشتن در LCD می باشد.

RD به پین B.1 متصل شده است، این پایه برای فعال سازی خواندن از LCD می باشد.

RESET به پین B.4 متصل شده است، که برای بازنشانی LCD میباشد.

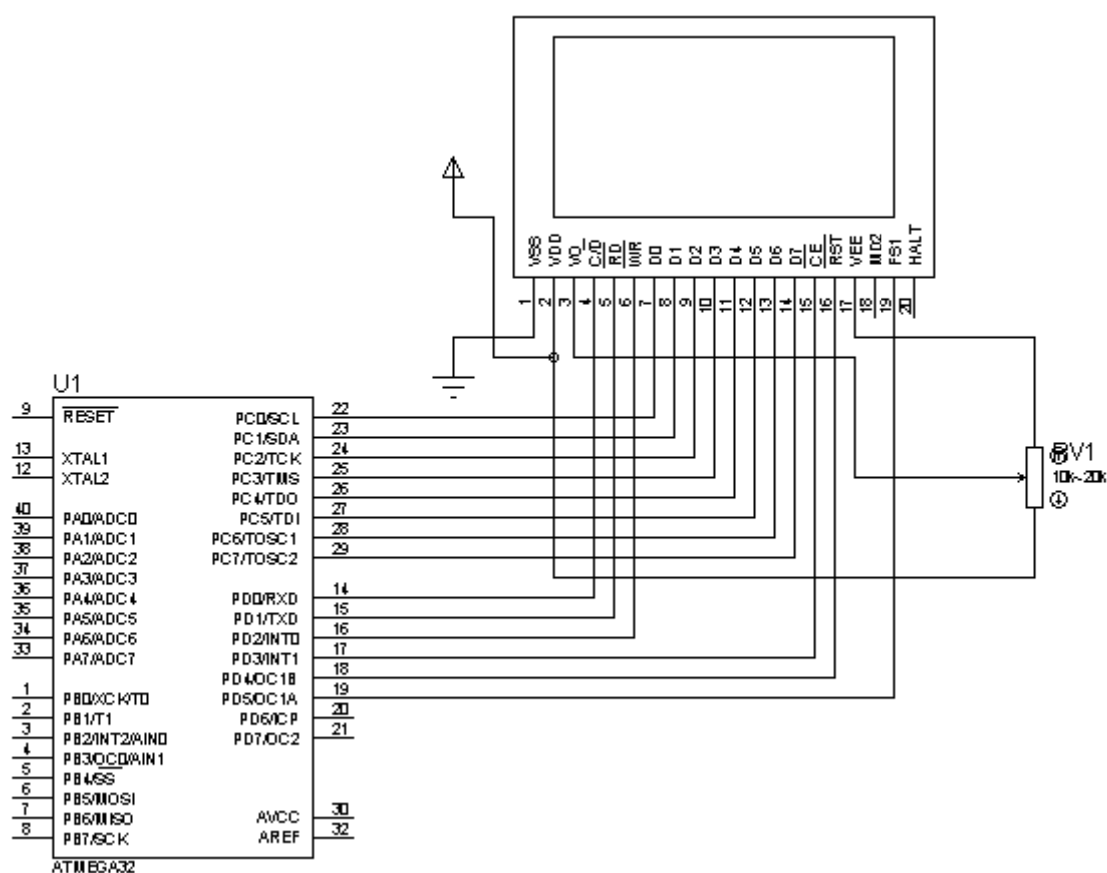
FS به پین *B.5* متصل شده است، که برای انتخاب نوع فونت *LCD* میباشد، اگر این پایه سطح منطقی *1* باشد فونت *LCD ۸*۸* و اگر در سطح منطقی *0* باشد فونت *LCD ۸*۸* انتخاب میشود.

MODE برای مشخص کردن فونت *LCD* میباشد که 2 حالت دارد:

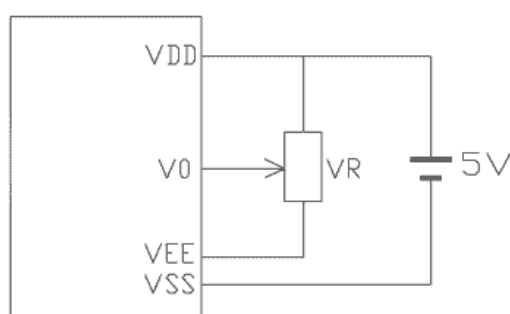
MODE=8 که فونت *8*8* را انتخاب میکند.

MODE=6 که فونت *6*8* را انتخاب میکند.

طریقه اتصال پایه های *LCD* به میکروکنترلر:



به طریقه اتصال پتانسیومتر جهت کنتراست دقت کنید.



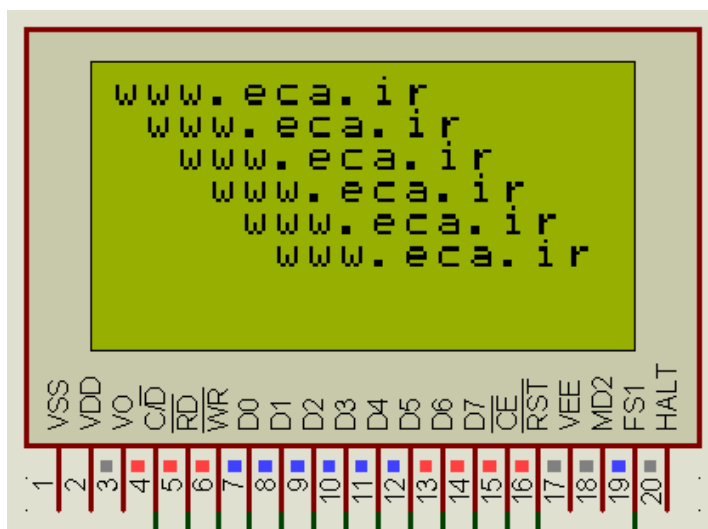
البته این برای تمام *LCD* ها به این صورت نمیباشد.
همیشه قبل از راه اندازی کاتالوگ *LCD* را مطالعه کنید.

تمام دستورات *lcd* های *Toshiba* همانند دستورات *lcd* های *sed* میباشد. تنها تفاوت آنها در دستور *LCDAT y, x, var* میباشد. برای *lcd* های *Toshiba* این دستور همانند *lcd* های کاراکتری میباشد.

مثال

```
$regfile = "m32def.dat"  
$crystal = 8000000  
Config Graphlcd = 128 * 64 , Dataport = Portc , Controlport = Portd , Ce = 3 , Cd = 0 , Wr =  
2 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8  
Dim X As Byte , Y As Byte
```

```
Cursor Off  
Cls  
Do  
  Locate X , Y  
  Lcd "www.eca.ir"  
  Waitms 500  
  Incr X  
  Incr Y  
Loop  
End
```



برای نمایش نوشته و یا یک متغیر باید از دستور *LCD var* استفاده کنید.

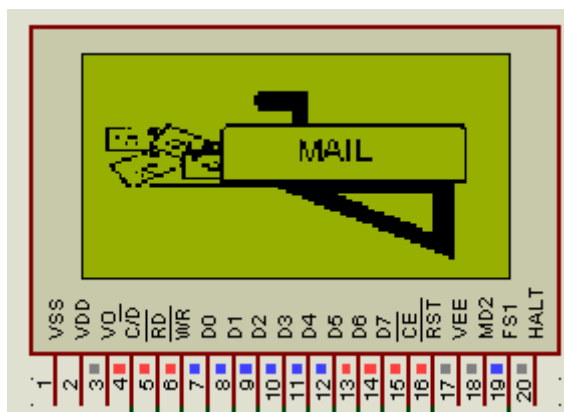
برای نمایش تصویر هم به همان صورتی که برای *lcd* های *Sed* عمل میکردیم ، عمل میکنیم.
مثال:

```
$regfile = "m32def.dat"  
$crystal = 8000000  
Config Graphlcd = 128 * 64 , Dataport = Portc , Controlport = Portd , Ce = 3 , Cd = 0 , Wr =  
2 , Rd = 1 , Reset = 4 , Fs = 5 , Mode = 8  
Dim X As Byte , Y As Byte
```

```
Cursor Off  
Cls  
Showpic 0 , 0 , Pic
```

```
End
```

```
Pic:  
$bgf "1.bgf"
```



T S G

WWW.TAAKTEEK.COM