آموزش میکروکنترلر AVR
به زبان بیسیک

این آموزش بر اساس برد آموزشی

حرفه ای AVR نوشته است

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

مقدمه

الکترونیک دیجیتال کار خود را با به وجود آوردن منطق صفر و یک، اختراع میکروپروسسورها و طراحی کامپیوترها آغاز کرد. میکروپروسسورهایی نظیر 8086 از شرکت اینتل و Z80 از شرکت زایلوگ، شروع کار بودند.

با ورود خانواده میکروکنترلر 8051 از شرکت اینتل، تحولی عظیم در این صنعت رخ داد و میکروکنترلرها تنها یک پردازشگر نبودند و عملیات محاسبه و منطق تنها بخشی از این تراشه بود. امکاناتی نظیر حافظه‌ها، تایمرها و ارتباطات به این تراشه افزوده شد و این تراشه مانند یک کامپیوتر کوچک به بازار عرضه شد. طولی نکشید که شرکت‌هایی نظیر Micro Chip و Atmel سری جدیدتری از میکروکنترلرها را عرضه کردند.

میکروکنترلرهای ۸ بیتی AVR ساخت شرکت Atmel از پرکاربردترین نوع میکروکنترلرهای موجود در دنیا می‌باشند و دلیل آن وجود امکانات متمایز از سایر میکروکنترلرها است.

تاریخچه میکروکنترلرهای AVR

اولین میکروکنترلر در سال ۱۹۷۱ توسط شرکت نام آشنای Intel ساخته شد و این شرکت اولین میکروکنترلر کاربردی خود را در سال ۱۹۸۰ با نام 8080 روانه بازار کرد.

کلمه میکروکنترلر از دو عبارت میکرو و کنترل تشکیل شده است که اولی واحدی یونانی به معنای یک میلیونم و دومی به معنای تحت نظارت داشتن کاری است.

با توجه به حرکت جوامع بشری به سوی هر چه کوچک‌تر کردن وسایل کاربردی، طراحان الکترونیک به تبعیت از این قانون، سعی در کوچک کردن مدار کنترلی یک پروسه و کاهش هزینه های مربوط نمودند که این امر موجب پیدایش میکروکنترلرها به عنوان وسیله‌ای که دارای حافظه، CPU، پورت‌های ورودی و خروجی و ... در یک چیپ گردید.

ما امروزه شاهد معماری‌های مختلفی از میکروکنترلرها هستیم که مهم‌ترین آن‌ها عبارتند از:

۱- AVR

۲- PIC

۳- 8051

اما تفاوت میکروکنترلرهای ۳ خانواده مذبور علاوه بر تکنولوژی ساختشان، در برنامه نویسی مورد نیاز و نحوه پروگرام کردن آن‌ها می‌باشد.

میکروکنترلرهای AVR

AVR ها میکروکنترلرهایی ۸ بیتی از نوع Cmos با توان مصرفی پایین هستند که بر اساس ساختار پیشرفته RISC با معماری Harvard ساخته شده‌اند.

RISC مخفف (Reduced Instruction Set Computer) به معنی مجموعه دستورالعمل‌های کاهش یافته و Harvard به نوعی معماری گفته می‌شود که در آن حافظه ذخیره برنامه و حافظه ذخیره داده، از هم جدا می‌باشند.

در میکروکنترلرهای AVR دستورات تنها در یک پالس ساعت اجرا می‌شوند و به این ترتیب به ازای هر یک مگاهرتز می‌تواند یک مگا دستور را در ثانیه اجرا کند، در نتیجه برنامه از لحاظ سرعت پردازش و مصرف توان بهینه می‌شود.

این میکروکنترلرها دارای ۳۲ رجیستر همه منظوره و مجموعه دستورات قدرتمندی هستند که تمام این ۳۲ رجیستر مستقیما به ALU (بخش پردازش) متصل شده‌اند، بنابراین دسترسی به دو رجیستر در یک سیکل ساعت هم امکان پذیر بوده که باعث می‌شود سرعت این میکروها نسبت به میکروکنترلرهای CISC تا ۱۰ برابر افزایش یابد.

انواع میکروکنترلرهای AVR

میکروکنترلرهای AVR با دو معماری ۸ بیتی و ۱۶ بیتی ساخته می‌شوند که در اینجا به شرح کارکرد مدل ۸ بیتی می‌پردازیم.

میکروکنترلرهای ۸ بیتی AVR به سه دسته تقسیم می‌شوند:

۱. Tiny AVR
۲. Mega AVR
۳. Xmega AVR

تفاوت بین این سه نوع به امکانات موجود در آن‌ها مربوط می‌شود. Tiny AVR ها غالبا تراشه‌هایی با تعداد پین و مجموعه دستورات کمتری نسبت به Mega AVR می‌باشند و به عبارتی از لحاظ پیچیدگی حداقل امکانات را دارند و Xmega AVRها حداکثر امکانات را داشته و Mega AVRها در بین این دو نوع هستند.

امکانات کلی ATMEGA 32

- ✓ ۳۲ رجیستر همه منظوره.
- ✓ دارای سه نوع حافظه شامل: Flash, EEprom, Sram
- ✓ توانایی برنامه ریزی تراشه در داخل مدار بدون احتیاج به پروگرامر (In System Programing).

- ✓ حفاظت از کدهای برنامه در مقابل خواندن (با قفل فیوزبیت های آن).
- ✓ قابلیت تنظیم نوسانگر برای کار توسط کریستال خارجی و داخلی و نوسانگر RC داخلی و خارجی.
- ✓ مجهز به پروتکل JTAG برای انجام عمل دیباگ، تست و اسکن وسایل جانبی تراشه و ...
- ✓ شمارنده و تایمر ۸ بیتی و ۱۶ بیتی.
- ✓ RTC با نوسانگر جداگانه.
- ✓ کانال‌های PWM با استفاده از تایمرها به صورت ۸ و ۱۶ بیتی.
- ✓ ADC های ۱۰ بیتی.
- ✓ ارتباط سریال USART با قابلیت برنامه ریزی.
- ✓ تایمر watch dog با قابلیت برنامه ریزی با نوسانگر مجزا (WTD).
- ✓ مقایسه کننده آنالوگ با امکان تعریف وقفه برای آن.
- ✓ منابع وقفه داخلی و خارجی.
- ✓ دارای حدود ۱۳۰ دستور که اکثر آن‌ها در یک سیکل ساعت اجرا می‌شوند.

تشریح پایه های ATMEGA 32

در تراشه‌های AVR پایه‌های آن‌ها علاوه بر استفاده به عنوان I\O برای یک یا چند خصوصیت دیگر نیز مورد استفاده قرار می‌گیرند که در زیر به تشریح آن‌ها می‌پردازیم:

- ❖ پایه OC1A:

خروجی مد مقایسه تایمر– کانتر ۱ و نیز خروجی موج PWM1.

- ❖ پایه OC1B:

خروجی مد مقایسه تایمر– کانتر ۱ و نیز خروجی موج PWM2.

- ❖ پایه SCK:

به عنوان کلاک ورودی و خروجی Master و Slave در ارتباط SPI استفاده می‌شود.

- ❖ پایه MISO:

به عنوان ورودی داده میکرو Master و خروجی داده میکرو Slave استفاده می‌شود.

- ❖ پایه MOSI:

به عنوان خروجی داده میکرو Master و ورودی داده میکرو Slave استفاده می‌شود.

❖ پایه AIN0:

به عنوان ورودی پایه مثبت مقایسه کننده آنالوگ استفاده می‌شود.

❖ پایه AIN1:

به عنوان ورودی پایه منفی مقایسه کننده آنالوگ استفاده می‌شود.

❖ پایه OC0:

در خروجی مد مقایسه‌ای تایمر– کانتر صفر مورد استفاده قرار می‌گیرد.

❖ پایه T0:

در ورودی کلاک برای کانتر صفر استفاده می‌شود.

❖ پایه T1:

در ورودی کلاک برای کانتر یک استفاده می‌شود.

❖ پایه TOSC1:

در زمان استفاده از RTC به این پایه کریستال ۳۲۷۶۸ هرتز وصل می‌شود.

❖ پایه TOSC2:

در زمان استفاده از RTC به این پایه کریستال ۳۲۷۶۸ هرتز وصل می‌شود.

❖ پایه TDI:

ورودی داده سریال در ارتباط JTAG می‌باشد.

❖ پایه TDO:

خروجی داده سریال در ارتباط JTAG می‌باشد.

❖ پایه TMS:

به عنوان ارتباط JTAG استفاده می‌شود.

❖ پایه TCK:

به عنوان ارتباط JTAG استفاده می‌شود.

❖ پایه SDA:

به عنوان خط داده در ارتباط دو سیمه (I2C) استفاده می‌شود.

❖ پایه SCL:

به عنوان خط کلاک در ارتباط دو سیمه (I2C) استفاده می‌شود.

❖ پایه OC2:

مد مقایسه‌ای تایمر– کانتر ۲ و به عنوان خروجی موج PWM2 استفاده می‌شود.

❖ پایه ICP:

به عنوان ورودی Capture تایمر– کانتر ۱ استفاده می‌شود.

❖ پایه RXD:

به عنوان ارسال کننده داده در ارتباط سریال USART استفاده می‌شود.

❖ پایه TXD:

به عنوان دریافت کننده داده در ارتباط سریال USART استفاده می‌شود.

❖ پایه AREF و AVCC:

پایه‌های تعیین کننده ولتاژ مرجع برای مبدل آنالوگ به دیجیتال می‌باشند.

❖ پایه SS:

با فعال شدن در ارتباط SPI میکروکنترلر را به میکروی SLAVE تبدیل می‌کند.

❖ پایه XCK:

به عنوان کلاک خروجی در ارتباط UART در زمان مد آسنکرون استفاده می‌شود.

❖ پایه Reset:

به عنوان پایه‌ای برای ریست کردن میکرو به کار می‌رود.

❖ پایه‌های Xtal1 و Xtal2:

پایه‌هایی جهت اتصال کریستال خارجی به میکرو می‌باشند.

❖ پایه‌های ADC0 تا ADC7:

پایه‌های ورودی مبدل آنالوگ به دیجیتال می‌باشند.

❖ پایه‌های INT0 تا INT7:

پایه‌های ورودی وقفه خارجی می‌باشند.

شروع کار با میکروکنترلرهای AVR به زبان بیسیک

در این قسمت قبل از بستن مدارات میکروکنترلری ابتدا دستورات مربوطه توضیح داده می‌شود و سپس مدارات عملی و تمریناتی در پایان آن برای درک بیشتر قید می‌شود.

فرم کلی نوشتن یک برنامه در کامپایلر بسکام:

۱- میکروکنترلر مورد استفاده خود را انتخاب کنید.

۲- فرکانس کار میکرو را انتخاب کنید.

۳- امکانات میکروکنترلری را که می‌خواهید در برنامه از آن‌ها استفاده کنید را پیکربندی کنید.

۴- متغیرهای مورد نیاز خود را نام گذاری کنید.

۵- برنامه اصلی خود را پیاده سازی کنید.

تشریح پورت ها و پین های میکروکنترلر

در میکروکنترلر ATMEGA 32 چهار پورت به نام‌های A، B، C و D وجود دارد که هر کدام خود دارای ۸ پین می‌باشند. هر پورت دارای ۳ رجیستر به نام‌های PORT، PIN و DDR می‌باشد که در زبان بیسیک به رجیستر DDR دسترسی نداریم(منظور از نداشتن دسترسی به رجیستر DDR غیر قابل استفاده بودن آن نیست بلکه منظور وجود دستورات پیکربندی(config portx) در بسکام به جای مقداردهی مستقیم رجیسترهاست) بنابراین رجیسترهای PORT و PIN باقی می‌ماند که PORT رجیستر خروجی و PIN رجیستر ورودی می‌باشد یعنی اگر پورت را به عنوان ورودی استفاده کنید باید از PIN و اگر از آن به عنوان خروجی استفاده کنید باید از رجیستر PORT استفاده کنید.

دستورات مربوط به پیکربندی وسایل I/O و تنظیمات کامپایلر

معرفی میکروکنترلر به کامپایلر ..... $regfile = "MxDef.Dat"

که نام میکروی مورد استفاده به جای x نوشته می‌شود. مثلا برای Atmega32 داریم:

```
$regfile = "M32Def.Dat"
```

انتخاب فرکانس کاری میکرو..... $crystal = X Hz

توسط این دستور سرعت پردازش اطلاعات به کامپایلر معرفی می شودکه مقدار آن از صفر تا ۱۶ مگاهرتز می‌باشد که تا فرکانس ۸ مگاهرتز توسط اسیلاتور داخلی تامین می‌شود و برای استفاده از فرکانس بالاتر از ۸ مگاهرتز باید از کریستال خارجی استفاده کرد.

انتخاب فرکانس ۸ مگاهرتز برای میکرو $crystal = 8000000

نکته: تعیین این فرکانس فقط برای کامپایلر است و در عمل باید فیوزبیت کلاک را برای فرکانس مورد نظر چه در هنگام استفاده از اسیلاتور داخلی و چه کریستال خارجی تنظیم کنیم، چون مسیر حافظه فلاش و فیوزبیت‌ها در میکروکنترلرهای AVR از هم جدا بوده و این یکی از معایب میکروکنترلرهای AVR محسوب می‌شود.

فراخوانی کتابخانه های موردنیاز

پیکربندی پورت‌ها به عنوان ورودی و خروجی ..... Config portX=Input/Output

پورت مورد نظر جایگزین X می‌شود و همچنین می‌توانید پین‌های مورد نظر را تک تک نیز به عنوان ورودی یا خروجی تعیین کنید.

انتخاب پورت C به عنوان خروجی Config portc = output

انتخاب پین C.0 از پورت C به عنوان خروجی Config portc.0 = output

مقدار دهی پین و پورت ..... portx.y=constant

با این دستور می‌توان مقدار دلخواه ۸ بیتی را به پورت‌ها داد و همچنین می‌توان پین‌ها را نیز مقداردهی نمود.

Portc.2= 0

مقداری که به پورت اختصاص می‌یابد می‌تواند باینری، هگز و یا محتویات یک متغیر باشد.

برای مقادیر باینری از &B و برای مقادیر هگزادسیمال از &h استفاده می‌شود.

```
PORTD= &B00001100
PORTD= &H7f
```

روشن کردن پین و پورت ..... set pin/port

با این دستور می‌توان یک پورت یا پین را ۱ کرد.

```
Set Pind.5
Set PORTD
```

خاموش کردن پین و پورت ..... Reset pin/port

با این دستور می‌توان یک پورت یا پین را 0 کرد.

```
Reset Pind.5
Reset PORT
```

دستورات مربوط به کار با متغیرها

معرفی متغیرها:

متغیر چیست؟ متغیر نامی برای کلمات حافظه است که داده‌ها در آن قرار می‌گیرند و ممکن است در طول اجرای برنامه تغییر کنند. برای دسترسی به متغیرها از نامشان استفاده می‌شود.

متغیرها دارای نوع می‌باشند که هنگام نام گذاری باید آن را مشخص کرد. در جدول زیر انواع متغیرها نمایش داده شده است.

Data Type	Store AS	Value Range
Bit	1 bit	0 OR 1
Byte	Unsigned 8 bits	0 TO 255
Integer	Signed 16 bits	-32767 TO 32767
Word	Unsigned 16 bits	0 TO 65535
Long	Signed 32 bits	-214783648 TO 214783648
Single	Signed 16 bits	1.5*10^-45 TO 3.4*10^38
String	0 - 245	_____

تعریف متغیر ..... DIM var AS Data Type

var نام متغیر و Data Type نوع یا جنس متغیر را تعیین می‌کند که می‌توان آن را بسته به نیاز از روی جدول بالا انتخاب کرد. مانند :

Dim A AS Word

حالت دیگری هم برای تعریف متغیر وجود دارد که به صورت زیر است :

DEF Data Type Var، که Data Type نوع متغیر و Var نام متغیر می‌باشد.

افزایش یک واحدی ..... Incr var

با این دستور یک واحد به متغیر عددی اضافه می‌شود.

کاهش یک واحدی ..... Decr var

با این دستور یک واحد از متغیر عددی کم می‌شود.

استخراج داده از جدول داده‌ها..... جدول Lookup

فرم کلی دستور : var = Lookup(Value,Lable)

Var: متغیری که مقدار استخراج شده در آن قرار می‌گیرد.

Value: اندیس (شماره) داده دلخواه است به طور مثال اگر Value=0 باشد، اولین داده جدول در متغیر قرار می‌گیرد.

Lable: برچسب جدول است که معمولا پس از دستور End و در پایان برنامه نوشته می‌شود.

نکات مربوط به دستور:

- حداکثر مقدار Value (تعداد اندیس‌ها) ۲۵۵ می‌باشد.
- حداکثر مقدار داده برگشتی ۶۵۵۳۵ (Integer,Word) می‌باشد.
- در داده های دو بایتی (Integer,Word) هر داده بایستی به علامت % ختم شود.

مانند :

```
Dim B As Integer
B = Lookup( 1 , Dta )
lcd B
End
Dta:
Data 1000% , 2000%
```

تغییر نام متغیر و پورت‌ها..... NewName ALIAS OldNmae

NewNmae: نام دلخواه

OldName: متغیر یا پورت و یا پایه مورد نظر برای تغییر نام

کاربرد: در زمان کار با پورت‌ها برای مثال یک پایه از پورت C را به یک بلندگو وصل کرده‌ایم، حال به جای آنکه در طی برنامه شماره پایه مورد نظر را حفظ کنید با این دستور می‌توان نام آن را به Speaker تبدیل کرده و از آن استفاده کنید.

معکوس کردن بیت ..... Toggle pin/var

با این دستور می‌توان یک بیت از متغیر یا یک پین از پورت‌ها را معکوس کرد. Toggle PORTd.0

ایجاد تاخیر در برنامه ..... wait X

با این دستور می‌توان در برنامه به مقدار مورد نظر X تاخیر ایجاد کرد که مقدار تاخیر می‌تواند میکرو ثانیه، میلی ثانیه و ثانیه باشد.

ایجاد تاخیر ۱۰ ثانیه wait 10

ایجاد تاخیر ۵۰ میکرو ثانیه waitus 50

نکته : X می‌تواند مقدار یک متغیر هم باشد.

تعریف آرایه DIM Name(X) AS Data Type .....

آرایه مجموعه‌ای از عناصر هم نوع است. هر آرایه دارای نامی است که مشابه متغیرهای معمولی نام گذاری می‌شود. برای دسترسی به عناصر آرایه از اندیس آرایه استفاده می‌شود و از صفر شروع می‌شود.

X: تعداد مورد نیاز متغیر است.

Data Type: نوع یا جنس متغیر می‌باشد.

مثال : DIM A(10) AS Byte

دستورات مربوط به حلقه

ایجاد حلقه تکرار ..... Do ..Loop

فرم کلی دستور به شکل زیر است :

Do Statements(دستورات)

Loop [Until expression:شرط تا زمان درستی]

تا زمانی که شرط درست باشد این حلقه تکرار می‌شود و چون شرط حلقه در پایان آن است پس حداقل یک بار اجرا می‌شود.

نکته: از این حلقه در اکثر موارد به عنوان حلقه بی‌نهایت و بدون شرط استفاده می‌شود و همچنین با دستور Exit DO می‌توان از حلقه خارج شد.

حلقه تکرار محدود ..... For … Next

شکل کلی حلقه : For Var=Start To End [Step Value] Next var

Var: مانند یک شمارنده عمل می‌کند.

Start: مقدار اولیه (ثابت یا متغیر عددی)

End: مقدار نهایی (ثابت یا متغیر عددی)

Step Value: مقدار گام حلقه می‌باشد که می‌تواند مثبت یا منفی باشد و اگر نوشته نشود مقدار ۱ در نظر گرفته می‌شود.

استفاده از عملگرها و توابع در برنامه نویسی

نرم افزار بسکام این امکان را فراهم کرده که بتوان در برنامه نویسی از عملگرها و توابع ریاضی مانند جمع، ضرب، سینوس، کسینوس و ... و یا توابعی آماده غیر از توابع ریاضی استفاده کرد. در جداول زیر لیست تمام عملگرها و یک سری از توابع ریاضی آورده شده است.

عملگر	نماد عملگر
ضرب	*
جمع	+
تفریق	-
تقسیم	/
ممیز	.
تساوی	=

بزرگ‌تر از	>
کوچک‌تر از	<
بزرگ‌تر مساوی با	>=
کوچک‌تر مساوی با	<=
مخالف	<>
توان	^

نام تابع	نحوه استفاده
سینوس	Var=SIN(X)
کسینوس	Var=COS(X)
تانژانت	Var=TAN(X)
سینوس معکوس	Var=ASIN(X)
کسینوس معکوس	Var=ACOS(X)
تانژانت معکوس	Var=ATN(X)
قدر مطلق	Var=ABS(X)
روند کردن	Var=ROUND(X)
لگاریتم بر مبنای ۱۰	Var=LOG10(X)
لگاریتم طبیعی	Var=LOG(X)

نکات:

- نوع متغیرهای Var و X باید از نوع Single باشد.
- تمامی توابع مثلثاتی بر حسب رادیان هستند.

انتخاب عدد تصادفی Var=RND(x)

با استفاده از این تابع می‌توان یک عدد تصادفی بین صفر تا مقدار x مثبت را برگزیده و در متغیر Var قرار دهد.

دستورات مربوط به اتصال کلید به میکروکنترلر

اتصال کلید به پین ..... Debounce

شکل کلی دستور : Debounce Px.y , state , label

Px.y: X نام پورت و Y نام پایه‌ای از پورت است که به عنوان ورودی تعریف شده است.

State: وضعیت کلید که ۰ یا ۱ می‌باشد. پردازنده با بررسی وضعیت پایه Y و مقایسه آن با State، در صورت برابری به برچسب Lable پرش می‌کند و در غیر این صورت برنامه از خط بعد از دستور Debounce ادامه پیدا می‌کند.

Lable: نام یک تابع است که خود شامل یک سری دستورات است و در انتهای آن باید از دستور Return استفاده کرد تا به برنامه اصلی بعد از Debounce برگردد.

نکته: در اتصال کلیدها به میکرو باید از مقاومت‌های بالاکش (Pull Up) و یا پایین‌کش (Pull Down) مطابق با برنامه خود استفاده کنید در غیر این صورت میکرو دچار مشکل می‌شود.
مناسب‌ترین مقاومت‌ها برای Pull Down و یا Pull Up کردن کلیدها، مقاومت‌های ۳.۳ کیلو تا ۱۰ کیلواهم می‌باشند.
نکته: بر روی برد آموزشی از مقاومت‌های Pullup استفاده نشده و باید از Pullup داخلی استفاده کنید.

انتظار برای بیت ..... BITWAIT

شکل کلی دستور : BITWAIT Pinx.y,Set/Reset

Px.y: X نام پورت و Y نام پایه‌ای از پورت است که به عنوان ورودی تعریف شده است.

این دستور تا زمانی که پین مورد نظر به دلخواه SET یا Reset نشود، برنامه را در همان خط متوقف می‌کند.

انجام عملیات شرطی ..... IF THEN

دستور IF به صورت‌های مختلفی مورد استفاده قرار می‌گیرد که در زیر ۳ نمونه از آن نشان داده می‌شود:

شکل ساده دستور: IF comp or(comps) Then Statement

Comp: شرط مورد نظر است. در صورتی که بیش از یک شرط داشته باشیم، شروط را در داخل پرانتز قرار داده و با توابع منطقی مانند OR و AND از هم جدا می‌کنیم.

Statment: دستوری که در صورت درستی شرط اجرا می‌شود و در صورت عدم درستی شرط برنامه به خط بعد از شرط IF منتقل می‌شود.

اگر بیش از یک دستور Statment داشته باشیم دستور IF به شکل زیر استفاده می‌شود:

```
IF comp or( comps ) Then
Statement
End IF
```

در صورتی که هم تعداد شروط و هم تعداد Statment ها بیش از یکی باشد از فرم زیر استفاده می‌شود:

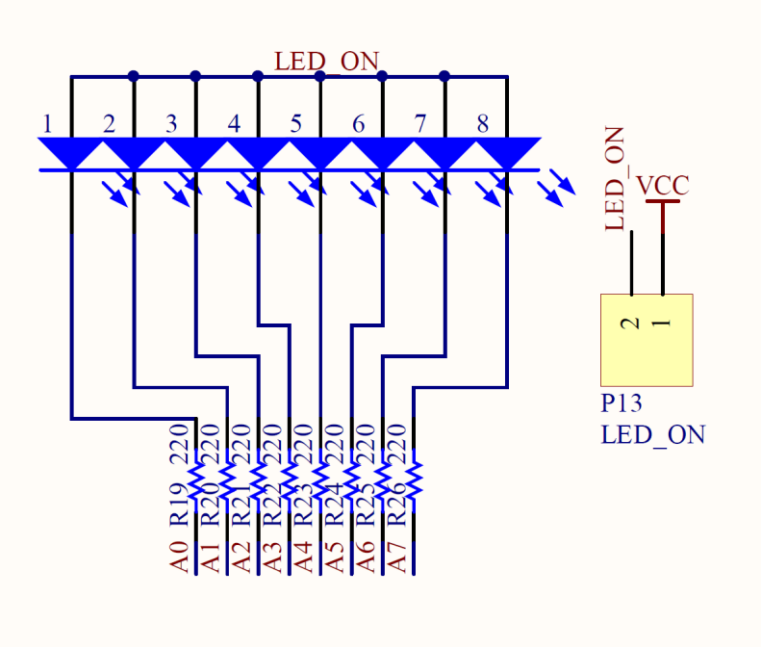
```
IF comp or( comps ) Then
Statement 1
Elseif comp or( comps ) Then
Statement 2
Elseif comp or( comps ) Then
Statement 3
.
.
.
Else
Statement n
End IF
```

مدارات عملی

۱. مدار یک رقص نور ساده را با ۸ LED طراحی کرده و برنامه مربوطه را بنویسید؟

شماتیک مربوط به اتصال LEDها:

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹



تنظیمات اعمال شده روی برد آموزشی:

جامپر مربوط به قسمت تغذیه در مد ۵ ولت قرار داده شود و جامپر LED نیز متصل گردد.

نکته: چون LED ها به پورت A متصل شده‌اند، پورت A را به عنوان خروجی در نظر بگیرید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
Config Porta = Output
Porta = &HFF
'-----------------------
Dim I As Byte
'-----------------------
Do
For I = 1 To 128 Step I * 2
Porta = Not I
Waitms 250
Next I
Loop
End
```

۲. برنامه مدار یک رقص نور پیشرفته را با استفاده از حلقه For Next با ۸ LED را بنویسید؟

```
$regfile = "m32def.dat"
$crystal = 8000000
```

```
'--------------------------
Config Porta = Output
Dim Count As Byte
Dim X As Byte
'--------------------------
Do
For Count = 0 To 7
X = Lookup(count , Dta(
Porta = Not X
Waitms 250
Next Count
Loop
End
Dta:
Data &H00 , &H81 , &HC3 , &HE7 , &HFF , &HE7_ ,
&Hc3 , &H81 , &H00 , &HFF , &H00 , &HFF , &H00 , &HFF
```

نکته :

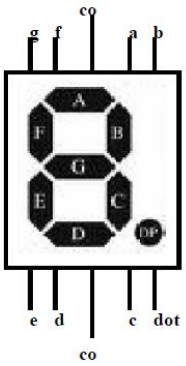
- با استفاده از تک کتیشن (') می‌توانید توضیحات خود را جلوی دستور یا هر قسمت از برنامه که احتیاج به توضیح دارد، استفاده کنید. چون علایم و توضیحات بعد از تک کتیشن توسط کامپایلر نادیده گرفته می‌شود.
- اگر خطی از برنامه طولانی شد می‌توانید از خط Under Line (شیفت + منها) در پایان خط اول استفاده کرده و بقیه دستور را در خط بعد بنویسید.

اتصال سون سگمنت به میکروکنترلر

سون سگمنت از ۸ LED تشکیل شده است که از ۷ عدد آن برای نمایش اعداد و حروف A تا F و از Led هشتم برای نمایش ممیز (Dot) استفاده می‌شود.

هر سون سگمنت تک رقمی دارای ۱۰ پایه به شرح زیر است:

- ۷ پایه که با حروف a تا g نام گذاری شده‌اند.
- ۱ پایه که با Dot نام گذاری شده است.
- ۲ پایه که پایه‌های مشترک بوده و در داخل IC به هم متصل می‌باشند.



سون سگمنت ها به دو دسته تقسیم می‌شوند:

۱- آند مشترک: پایه آند هر ۸ LED در داخل به هم وصل است و پایه کاتد آن‌ها آزاد می‌باشد.

۲- کاتد مشترک: پایه کاتد هر ۸ LED در داخل به هم وصل است و پایه آند آن‌ها آزاد می‌باشد.

اتصال سون سگمنت به میکرو و نمایش عدد بر روی آن

برای اتصال سون سگمنت به میکرو دو راه وجود دارد :

- اتصال پایه های a تا g مستقیم به یکی از پورت‌ها.
- استفاده از ای سی‌های دیکودر مانند ۷۴۴۷ و ۷۴۴۸.

نمایش اعداد تک رقمی روی سون سگمنت به روش معمولی

در این روش برای نمایش هر رقم یا حرف روی سون سگمنت ابتدا بایستی کد هگزادسیمال معادل آن را بدست آوریم.

برای این کار بایستی به دو نکته توجه داشته باشیم:

۱- آند یا کاتد مشترک بودن سون سگمنت.

۲- در نمایش هر رقم یا حرف کدام LEDها روشن و کدام LEDها خاموش خواهند شد.

با توجه به دو نکته بالا کد هگزادسیمال را برای ارقام ۰ تا ۹ جهت نمایش توسط سون سگمنت کاتد مشترک را بدست می‌آوریم:

رقم	HEX	A	B	c	d	e	f	g	dot
شماره پین		PA5	PA4	PA6	PA7	PA0	PA2	PA1	
۰	F5	1	1	1	1	1	1	0	0
۱	50	0	1	1	0	0	0	0	0
۲	B3	1	1	0	1	1	0	1	0
۳	F2	1	1	1	1	0	0	1	0
۴	56	0	1	1	0	0	1	1	0
۵	E6	1	0	1	1	0	1	1	0
۶	C7	1	0	1	1	1	1	1	0
۷	70	1	1	1	0	0	0	0	0
۸	F7	1	1	1	1	1	1	1	0
۹	76	1	1	1	1	0	1	1	0

نمایش اعداد تک رقمی روی سون سگمنت با استفاده از دیکودرها

برای انجام این روش به دو نکته بایستی توجه داشت:

- انتخاب دیکدر مناسب با توجه به نوع سون سگمنت.
- وصل خروجی‌های دیکدر(QA…..QG) نظیر به نظیر به ورودی‌های سون سگمنت (a…..g).

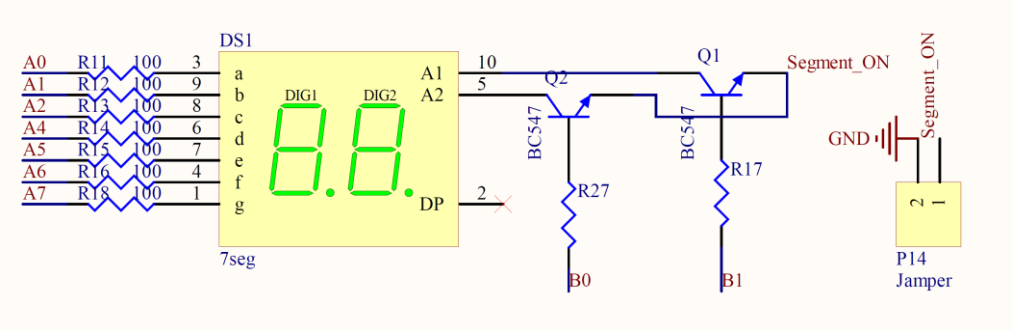
برای راه اندازی سون سگمنت آند مشترک از دیکدر ۷۴۴۷ و برای راه اندازی سون سگمنت کاتد مشترک از دیکدر ۷۴۴۸ استفاده می‌شود.

در برنامه نویسی برای نمایش عدد روی سون سگمنت با استفاده از دیکدر مستقیم از معادل دسیمال اعداد در برنامه استفاده می‌شود.

مدارات عملی :

۱- برنامه‌ی یک شمارنده ۰ تا ۹ با نمایش روی سون سگمنت را بنویسید؟

شماتیک:



تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه را در حالت ۵ ولت قرار داده و همچنین جامپر 7seg را وصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
Config Porta = Output
Config Portb = Output
Porta = &H00
Portb = &HFF
'-----------------------
Dim I As Byte
Dim Segment(10) As Byte
Segment(1) = &HF5
Segment(2) = &H50
Segment(3) = &HB3
Segment(4) = &HF2
```

```
Segment(5) = &H56
Segment(6) = &HE6
Segment(7) = &HC7
Segment(8) = &H70
Segment(9) = &HF7
Segment(10) = &H76
'----------------------
Do
  For I = 1 To 10
    Porta = Segment(i)
    Waitms 500
  Next I
Loop
End
```

نمایش اعداد چند رقمی روی سون سگمنت

برای نمایش اعداد چند رقمی روی سون سگمنت یک راه این است که به ازای هر رقم از یک پورت استفاده کنیم اما چون تعداد پورت‌های یک میکرو محدود است این روش، روش مناسبی نیست. همانطور که می‌دانیم چشم انسان در صورتی که ۲۵ تصویر یا بیشتر از یک شی پشت سر هم در یک ثانیه پخش شود آن را پیوسته می‌بیند و می‌توان از این خطای چشم استفاده کرد. که به روش مالتی پلکس کردن معروف است.

در این روش خطوط دیتا یعنی پایه های a تا g را به یکی از پورت‌ها وصل کرده و پایه های دیگر میکرو برای کنترل پایه مشترک سون سگمنت ها مورد استفاده قرار می‌گیرد. در این روش در هر لحظه فقط یک سون سگمنت روشن است و بقیه‌ی سون سگمنت ها خاموش می‌باشند ولی چون این عمل با سرعت بالا انجام می‌گیرد ما احساس می‌کنیم که همه‌ی آن‌ها روشن هستند.

مدارات عملی :

۱) برنامه ای برای نمایش عدد ۲۳ بر روی سون سگمنت های موجود را بنویسید؟

```
  $regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
Config Porta = Output
Config Portb = Output
```

```
Porta = &H00
Portb = &H00
'----------------------
Dim I As Byte
Dim Segment(10) As Byte
Segment(1) = &HF5
Segment(2) = &H50
Segment(3) = &HB3
Segment(4) = &HF2
Segment(5) = &H56
Segment(6) = &HE6
Segment(7) = &HC7
Segment(8) = &H70
Segment(9) = &HF7
Segment(10) = &H76
'----------------------
Do
  Porta = Segment(3)
  Portb = &H02
  Waitms 10
  Portb = 0
  Porta = Segment(4)
  Portb = &H01
  Waitms 10
  Portb = 0
Loop
End
```

نکات مهم در مورد کار با سون سگمنت ها:

- اگر یکی از سون سگمنت ها نسبت به بقیه نور کمی داشت زمان تاخیر یا همان Wait برنامه (روشن و خاموش بودن سون سگمنت ها) را درست انتخاب نکرده‌اید.
- برای کنترل سون سگمنت ها باید طبق مدارات بالا عمل کنید یعنی با استفاده از یک ترانزیستور آن‌ها را راه اندازی کنید چون با اتصال مستقیم آن‌ها به پایه‌های میکرو جریان زیادی را از پایه کشیده و باعث سوختن پایه مورد استفاده می شود.

- در طراحی شمارنده‌های خودکار که به صورت اتوماتیک کاهش یا افزایش می‌یابند باید تاخیری که در بین برنامه قرار می‌گیرد خیلی کم باشد در غیر اینصورت نمی‌توان روی سون سگمنت ها عدد مورد نظر را مشاهده کرد.

LCDهای کاراکتری و راه اندازی آن‌ها توسط میکروکنترلرهای AVR

LCDهای کاراکتری نمایشگرهایی با سطر و ستون مشخص برای نمایش داده‌ها می‌باشند.

در تمام این LCD تعداد پایه‌ها ثابت و برابر ۱۶ پایه بوده که نحوه اتصال آن‌ها به شرح زیر است:

نحوه اتصال پایه	سمبول	شماره پایه
اتصال به زمین	Vss	۱
اتصال به +5V	Vdd	۲
تنظیم کنتراست LCD	Vo یا VEE	۳
کنترل رجیستر	RS	۴
انتخاب مد خواندن یا نوشتن	RW	۵
فعال سازی LCD	E	۶
گذرگاه 8 تایی اطلاعات و دستورالعمل	D0 - D7	۷ – ۱۴
آند و کاتد LED پس زمینه	کاتد LED، آند LED	۱۵ ،۱۶

در این LCD می‌توان آن را به دو صورت به میکرو وصل کرد:

۱- مد ۴ سیمه: از چهار پایه گذرگاه برای اتصال LCD به میکرو استفاده می‌شود.

۲- مد ۸ سیمه : از هشت پایه گذرگاه برای اتصال LCD به میکرو استفاده می‌شود.

معایب و محاسن:

در مد هشت سیمه سرعت انتقال داده بیشتر است اما پایه بیشتری از میکرو را اشغال می‌کند ولی در مد چهار سیمه سرعت انتقال داده کمتر است اما پایه کمتری از میکرو را اشغال کرده و شلوغی سیم کشی مدار نیز نسبت به مد هشت سیمه کمتر است.

تقریبا در اکثر موارد از مد چهار سیمه استفاده می‌شود چون سرعت انتقال داده‌ی آن برای کارهایی که ما با میکرو انجام می‌دهیم مناسب است.

نحوه سیم بندی ارتباط LCD با میکرو در مد چهار سیمه

۱- خطوط دیتای Db4 الی Db7 جهت دریافت داده‌ها به ۴ پین از یک پورت میکرو که به صورت خروجی تعریف شده متصل می‌گردند.

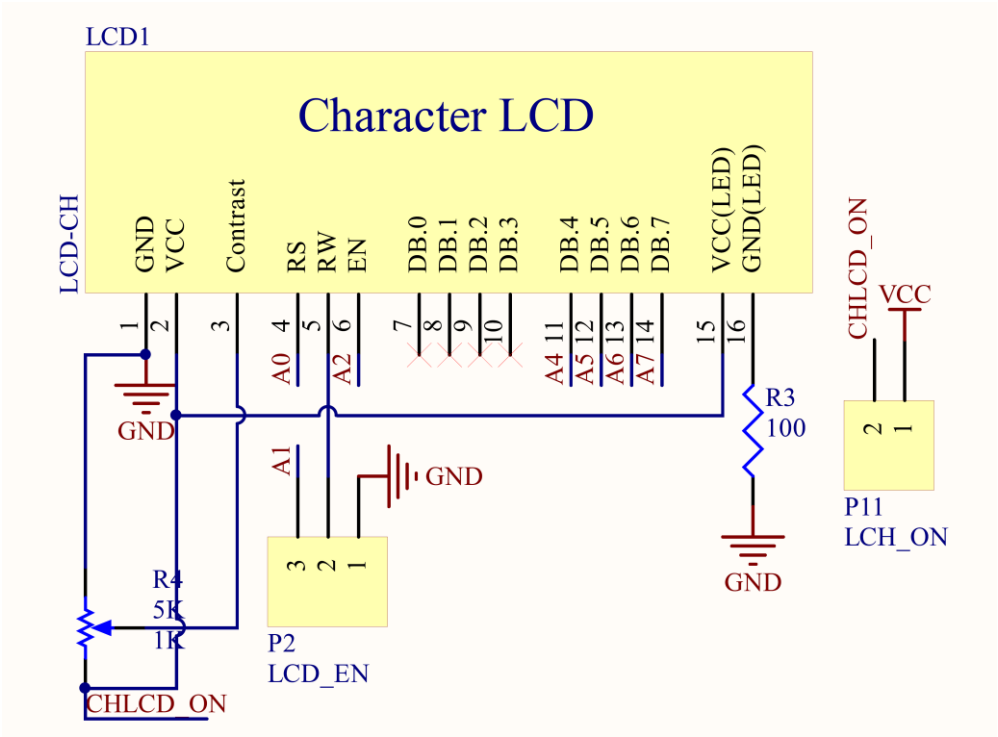
۲- خط RS برای دریافت دستورالعمل به یک پایه از پورت متصل می‌شود.

۳- خط E برای فعال سازی LCD به یک پایه از پورت متصل می‌شود.

۴- خط RW در مد چهار سیمه استفاده نمی‌شود در نتیجه به GND وصل می‌شود.

۵- خطوط Vss و Vdd به ترتیب به GND و +5V جهت تغذیه LCD متصل می‌شوند.

۶- خط Vo جهت تنظیم کنتراست به یک پتانسیومتر ۵ تا ۱۰ کیلو اهم متصل می‌شود.



نکات مهم:

- جهت تنظیم کنتراست هیچ گاه آن را مستقیم به Vcc یا GND نزنید.
- برای جلوگیری از اثر نویز روی مدار بهتر است پایه‌های Db0 تا Db3 را به همراه RW به GND وصل کنید.
- پایه‌های ۱۵ و ۱۶ مربوط به LED پس زمینه را مستقیم به Vcc و GND وصل نکنید چون جریان زیادی از مدار می‌کشند. در مسیر Vcc یا GND یک مقاومت در حد ۱۰۰ اهم قرار دهید.

تعیین نوع ارتباط ..... Config LcdBus

از این دستور برای تعیین نحوه سیم بندی LCD استفاده می‌شود که با انتخاب ۴ از مد چهار سیمه و با انتخاب ۸ از مد هشت سیمه استفاده می‌شود.

Config LcdBus = 4 OR 8

تعیین سایز یا نوع LCD ..... Config LCD

در این دستور سایز LCD مورد نظر را به کامپایلر معرفی می‌کنیم. منظور از سایز LCD تعداد سطرها و ستون‌های آن است.

که در اینجا LCD دارای ۲ سطر و ۱۶ ستون می‌باشد. Config LCD = 16*2

نحوه اتصال پایه‌های LCD به میکرو ..... Config Lcdpin

با این دستور پایه‌های LCD متصل شده به میکرو برای کامپایلر مشخص می‌شود که شکل کلی آن به صورت زیر است:

Config Lcdpin = pin , Rs = Portx.y , E = portx.y , Db4 = portx.y , Db5 = portx.y , Db6 = portx.y , Db7 = portx.y

X = نام پورت مورد نظر.

Y = شماره پایه مورد نظر از آن پورت.

نکته: برای پیکربندی پایه‌های LCd کاراکتری می‌توان از طریق خود نرم افزار هم عمل کرد که در این صورت به مسیر زیر در داخل نرم افزار رفته و تنظیمات مربوطه را لحاظ می‌کنیم.

Option⇒Compiler ⇒ Lcd

تا این قسمت تمام دستورات مربوط به پیکربندی LCd بحث شد از این به بعد به شرح دستورات مربوط به نوشتن علایم و پیغام‌ها روی LCD بحث می‌کنیم:

نمایش کاراکتر یا مقدار متغیر روی LCd ..... LCD X

با این دستور می‌توان مقدار متغیر مورد نظر را که به جای X قرار می‌گیرد را روی LCD نمایش داد.

نکته: اگر حرف یا حروف مربوطه داخل جفت کتیشن قرار گیرند خود آن‌ها نمایش داده خواهد شد. به مثال زیر دقت کنید:

نمایش محتویات متغیر A: LCD A

نمایش خود حرف A روی نمایشگر: LCD "A"

نکته: برای نمایش رشته کاراکترهای متفاوت در یک سطر بین آن‌ها از علامت «;» می‌توان استفاده کرد. Lcd " ECA " ; " AVR "

پاک کردن صفحه نمایش ..... Cls

توسط این دستور صفحه LCD کاملا پاک شده و آماده‌ی نوشتن مجدد می‌شود.

انتخاب شروع محل نوشتن ..... Locate Y,X

با این دستور می‌توان محل مورد نظر، جهت نمایش نوشته را انتخاب کرد.

Y = شماره ستون مورد نظر.

X = شماره سطر مورد نظر که نوشتن از این سطر آغاز می‌شود.

تنظیمات مکان نما ..... Cursor

برای مکان نما چهار حالت وجود دارد که به شرح زیر می‌باشند:

۱- روشن کردن مکان نما ..... Cursor on

۲- خاموش کردن مکان نما ..... Cursor off

۳- مکان نما به صورت چشمک زن ..... Cursor Blink

۴- مکان نما به صورت غیر چشمک زن ..... Cursor Noblink

انتقال مکان نما به سطر و ستون اول ..... Home

با این دستور مکان نما به سطر و ستون اول LCD پرش کرده و شروع به نوشتن اطلاعات جدید می‌کند.
تفاوت این دستور با دستور Cls در این است که دستور Cls کل محتویات صفحه را پاک می‌کند سپس اطلاعات جدید را می‌نویسد ولی با این دستور مکان نما به اولین قسمت LCD آمده و بدون پاک کردن اطلاعات قبل، روی آن‌ها اطلاعات جدید را می‌نویسد.

انتقال مکان نما به سطر دلخواه ..... Line

توسط دستورات زیر می‌توان مکان نما را مستقیم به سطر مورد نظرانتقال داد:

- Upperline: پرش مکان نما به یک سطر بالاتر.
- Lowerline: پرش مکان نما به یک سطر پایین‌تر.
- Thirdline: پرش به سطر سوم در صورت داشتن بیش از دوسطر.
- Fourthline: پرش به سطر چهارم در صورت داشتن حداقل چهار سطر.

نکته: دستورات مربوط به انتقال مکان نما به سطر سوم و چهارم برای LCD هایی مورد استفاده قرار می‌گیرد که دارای بیشتر از دو سطر باشند.

شیفت متن و مکان نما ..... Shift

با این دستور می‌توان متن و مکان نما را به سمت چپ یا راست به اندازه یک واحد شیفت داد. که شکل کلی آن به صورت زیر است:

شیفت کل متن به اندازه یک ستون به چپ یا راست: Shiftlcd Left / Right

شیفت مکان نما به اندازه یک ستون به چپ یا راست: Shiftcursor Left / Right

استخراج داده از جدول رشته ..... LOOKUPSTR

شکل کلی دستور : Var = Lookupstr(Value,Lable)

Var: متغیری که مقدار استخراج شده در آن قرار می‌گیرد.

Value: اندیس (شماره) داده دلخواه است، به طور مثال اگر Value=0 باشد اولین داده جدول در متغیر قرار می‌گیرد.

Lable: برچسب جدول است که معمولا پس از دستور End و در پایان برنامه نوشته می‌شود.

نکته: تعداد رشته‌های جدول تنها می‌تواند تا مقدار ۲۵۵ داشته باشد.

نکات:

- در صورتی که بخواهیم Shift به دفعات معین انجام شود بایستی از دستورات حلقه مانند حلقه For…Next استفاده کنید.

توجه داشته باشید موقع شبیه سازی این مدارات در پروتئوس متن و مکان نما برعکس شیفت می‌خورند یعنی وقتی نوشته‌اید که به راست شیفت بخورد به چپ حرکت می‌کند ولی در عمل چنین نیست.

خاموش و روشن کردن صفحه ..... Display On/Off LCD

توسط این دستور می‌توان صفحه نمایش را خاموش و روشن کرد.

نکته: توجه کنید هیچ گاه در آخرین خط برنامه از دستور Display Off استفاده نکنید چون صفحه نمایش خاموش شده و اگر بعد از آن قصد نمایش اطلاعات روی صفحه را داشته باشید قادر به مشاهده آن‌ها نخواهید بود مگر اینکه مجددا صفحه را روشن کنید.

مدارات عملی :

برنامه‌ای بنویسید که بتوان در دو سطر یک LCD ۲*۱۶ پیغامی را چاپ کند؟

تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه را در حالت ۵ ولت قرار داده، جامپر CHLCD را متصل کنید و نیز جامپر LCD_EN را به زمین متصل کنید.

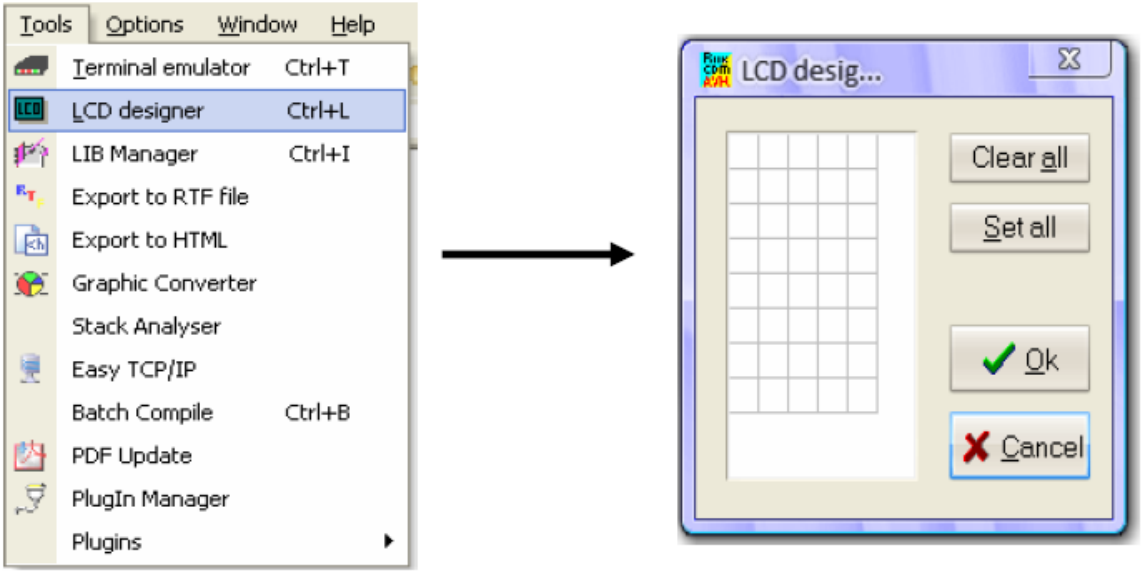
```
$regfile = "m32def.dat"
$crystal = 8000000
'----------------------------
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
'----------------------------
Dim I As Byte
'----------------------------
Do
  Cls
  Locate 1 , ۶ : Lcd "   www.ECA.ir"
  Locate 2 , ۵ : Lcd "  AVR Training "
Loop
End
```

ایجاد کاراکترهای ویژه و نمایش آن‌ها روی LCD

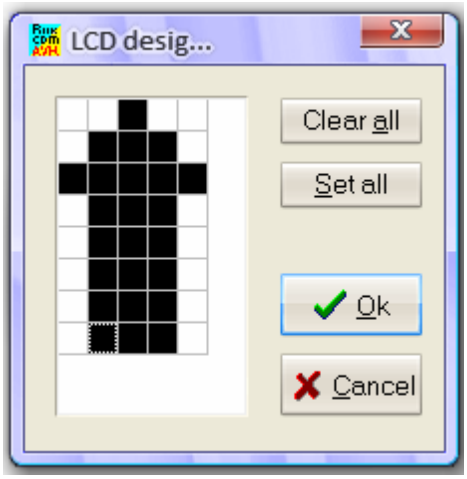
LCDهای کاراکتری دارای یک حافظه دائم هستند که درون آن کد کاراکترهای اسکی حروف انگلیسی وجود دارد و همچنین دارای یک حافظه موقت هستند که در آن می‌توان در هر لحظه حداکثر ۸ کاراکتر دلخواه را قرار داد.

برای ساخت کاراکتر دلخواه در برنامه بسکام، نرم افزار مورد نیاز تعبیه شده که در این محیط هر کاراکتر از یک ماتریس ۵*۸ تشکیل شده است، که با روشن و یا خاموش کردن هر پیکسل، می‌توان کاراکتر مورد نظر را طراحی کرد.

برای ساخت کاراکتر به مسیر Tools⇒ LCDdesigner رفته و مراحل زیر را دنبال می‌کنیم:



برای مثال می‌خواهیم کاراکتر فلش را طراحی کنیم:



بعد از روشن کردن پیکسل‌های مورد نظر Ok را می‌زنیم، با این کار یک خط به صورت زیر به برنامه اضافه می‌شود:

Deflcdchar ?,32,17,21,17,17,31,32,32

که به جای علامت سوال باید یکی از ارقام ۰ تا ۷ را قراردهید.

بعد از ساخت کاراکتر جدید با دستور زیر می‌توان آن را روی LCD نمایش داد.

نمایش کاراکتر ویژه ..... LCD CHR

با این دستور که شکل کلی آن به صورت زیر است می‌توان کاراکتر ساخته شده را روی LCD نمایش داد:

Lcd chr (?)

که به جای علامت سوال شماره کاراکتر مورد نظر که بین ۰ تا ۷ بود را وارد می‌کنیم.

نکته: قبل از دستور Lcd chr صفحه نمایش توسط دستور Cls پاک شود.

نکته : با استفاده از علامت دو نقطه (:) می‌توان چندین دستور را در یک خط نوشت.

مدارات عملی :

۱) برنامه‌ای بنویسید که روی LCD کلمه «سلام» را نشان دهد؟

کد اسکی حروف فارسی در حافظه LCD های کاراکتری وجود ندارد و برای نمایش کلمات فارسی باید آن‌ها را به روش گفته شده طراحی کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'----------------------

Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 =
Porta.6 , Db7 = Porta.7
Cursor Off
_____'

Deflcdchar 0 , 32 , 32 , 32 , 32 , 21 , 21 , 31 , 32
Deflcdchar 1 , 32 , 20 , 20 , 20 , 20 , 20 , 31 , 32
Deflcdchar 2 , 32 , 32 , 32 , 3 , 31 , 16 , 16 , 16
_____'

Cls
Locate 1 , 13
Lcd Chr(2) ; Chr(1) ; Chr(0(

End
```

سوالات متداول در مورد کار با LCDهای کاراکتری

۱- آیا می‌توان فقط یک سطر از LCD را شیفت داد و بقیه‌ی سطرها را ثابت نگه داشت؟ در LCDهای کاراکتری همچین عملی مقدور نیست، هرچند که می‌توان با تغییر Locateهای متن مورد نظر فقط یک سطر را شیفت داد ولی چون LCD چشمک می‌زند مورد استفاده قرار نمی‌گیرد.

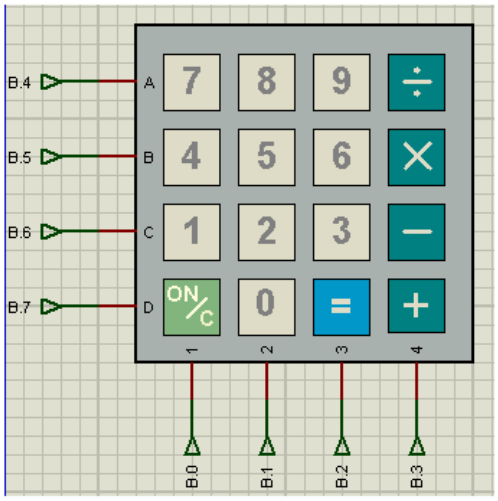
۲- چرا در ساخت کاراکتر ویژه، اعدادی که به جای علامت سوال قرار می‌گیرند باید یکی از اعداد ۰ تا ۷ باشند؟ در پاسخ این سوال باید گفت همانطور که در بالا توضیح داده شد حافظه موقت این نمایشگرها فقط می‌تواند تا ۸ کاراکتر را در خود جای دهد و ۰ تا ۷ هم ۸ کاراکتر می‌شود.

اتصال کی‌پد به میکروکنترلر

همانطور که در بالا توضیح داده شد برای اتصال وسایل I/O خارجی به میکروکنترلر ابتدا باید آن را پیکربندی نمود.

کی‌پدها شامل کلیدهایی هستند که به صورت ماتریسی به هم متصل شده‌اند که باعث شده تعداد پایه‌های کلیدها کمتر شده و پورت کمتری را در میکروکنترلرها اشغال کند.

کی‌پدها در انواع گوناگونی در بازار یافت می‌شوند که متداول‌ترین آن‌ها کی‌پدهای ۴*۳ و ۴*۴ می‌باشد. در زیر نحوه اتصال کی‌پد ۴*۴ و یک نمونه از کی‌پدهای 4*4 موجود در بازار، نمایش داده شده است.



بر روی بردی که در اختیار دارید یک کی‌پد ۴*۳ قرار داده شده است که به پورت D متصل می‌باشد و به راحتی می‌توان از آن استفاده کرد. بنابراین دیگر نیاز به تهیه کی‌پد خارجی نخواهد بود.

پیکربندی کی پد ..... ConfigKBD

با این دستور کی‌پد پیکربندی می‌شود که شکل کلی آن به صورت زیر است:

Config Kbd = PortX , Debounce= value

که X نام پورت مورد نظر است که کی‌پد به آن وصل شده است و Debounce مربوط به مدت زمانی است که میکرو کلید را چک می‌کند و همچنین مدت زمان سرکشی یا تاخیر کلید هم نامیده می‌شود و مقدار value می‌تواند از 20ms تا 255ms تغییر کند.

نکته: هنگامی که یک کلید فشار داده می‌شود، براثر لرزش دست، دوکنتاکت آن چندین بار به هم برخورد می‌کند و در نهایت ثابت می‌شود. درصورتی که از دستور Debounce استفاده نشود هر لرزشی به منزله فشارداده شدن یک کلید محسوب می‌شود.

اسکن کی‌پد ..... GetKbd

جهت استفاده از کی‌پد و گرفتن (خواندن) اعداد وارد شده توسط کی پد از این دستور استفاده می‌شود و شکل کلی آن به صورت زیر است:

Var = Getkbd ( ) درآن قرار می‌گیرد. Var = متغیری از قبل تعریف شده که کلید فشارداده شده

توجه:

در برد آموزشی با توجه با سایز کیپد نمی توان از توان اماده بسکام استفاده کرد بنابراین بایستی کاربر با استفاده از صفر و یک کردن پین های متصل شده و چک کردن آن ها کیپد را اسکن کند. این اسکن دقیقا همانند دستور بالا عمل می کند.

پرش به برچسب ..... GOTO

شکل کلی دستور : Goto Lable

برای پرش به برچسب Lable مورد استفاده قرار می‌گیرد. این پرش یک پرش بدون بازگشت است.

پرش به برچسب با بازگشت ..... GOSUB

شکل کلی دستور : GOSUB Lable

از این دستور برای پرش به برچسب Lable استفاده می‌شود و در صورت نیاز می‌توان با دستور Return به خط بعد از دستور پرش بازگشت.

انتخاب اجرای کدامین دستور ..... SELECT CASE

اجرای یک سری دستورات باتوجه به مقادیر یک متغیر.

شکل کلی دستور:

```
Dim A As Byte
Select Case A
Case test1 : Statements 1
Case test2 : Statements 2
Case test3 : Statements 3
.
.
Case Else : Statements n
End Select
```

در صورتی که مقدار test1 برابر با مقدار متغیر A باشد، دستورات اول (Statements 1) اجرا شده، سپس اجرای برنامه بعد از خط End Select ادامه پیدا می‌کند.

در صورتی که مقدار test2 برابر با مقدار متغیر A باشد، دستورات دوم (Statements 2) اجرا شده، سپس اجرای برنامه بعد از خط End Select ادامه پیدا می‌کند.

در صورتی که مقدار متغیر A با هیچکدام از testها برابر نباشد، دستورات مربوط به Case Else اجرا شده، سپس اجرای برنامه بعد از خط End Select ادامه پیدا می‌کند.

نکته: می‌توان چندین Case را که دستوراتشان از یک نوع باشد به صورت یک Case کلی نوشت:

```
Dim A As Byte
```

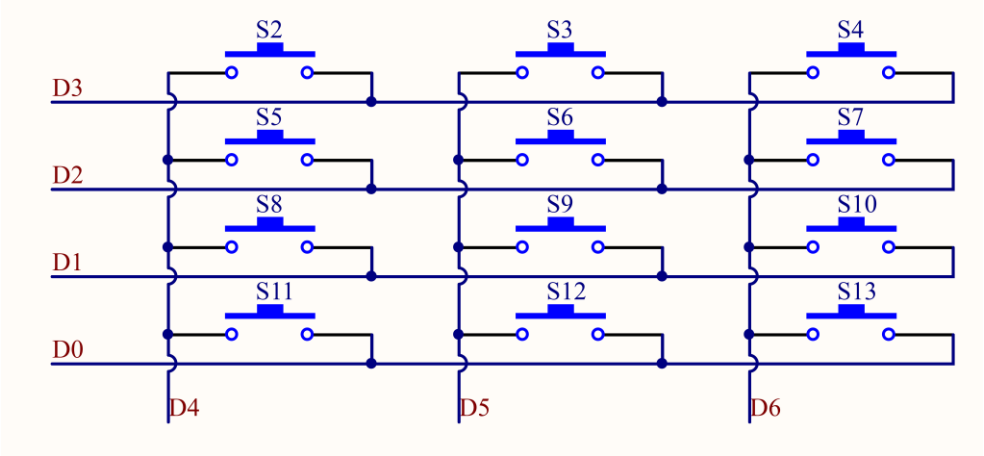

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

```
Select Case A
Case 0 - 2 : Lcd A
Case 3 : Lcd " Micro AVR " : Goto Main
Case Else : Cls
End Select
```

مدارات عملی :

۱) برنامه‌ای بنویسید که اعداد وارد شده توسط یک کی‌پد ۳*۴ را روی LCD نمایش دهد؟

شماتیک اتصال کیپد به میکروکنترلر:



تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه را بر روی ۵ ولت قرار داده و جهت فعال سازی LCD کاراکتری جامپرهای CHLCD و LCD_EN را متصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'---------------------------
Ddrd = &H0F
Portd = &HFF
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
'---------------------------
Declare Function Getkey As Byte
Dim I As Byte , Key As Byte
```

```
'---------------------------
Cls
Do
    Gosub Getkey
    If Key <> 12 Then
       Cls
       Lcd Key
    End If
Loop
End
'**********************
Getkey:
Key = 12
Portd = &HFF
Portd.0 = 0
Waitms 20
If Pind.4 = 0 Then Key = 10
If Pind.5 = 0 Then Key = 0
If Pind.6 = 0 Then Key = 11
Portd = &HFF
Portd.1 = 0
Waitms 20
If Pind.4 = 0 Then Key = 7
If Pind.5 = 0 Then Key = 8
If Pind.6 = 0 Then Key = 9
Portd = &HFF
Portd.2 = 0
Waitms 20
If Pind.4 = 0 Then Key = 4
If Pind.5 = 0 Then Key = 5
If Pind.6 = 0 Then Key = 6
Portd = &HFF
Portd.3 = 0
Waitms 20
If Pind.4 = 0 Then Key = 1
If Pind.5 = 0 Then Key = 2
If Pind.6 = 0 Then Key = 3
```

```
Return
```

۲) برنامه‌ای بنویسید که با زدن هر کلید از کی پد به جای آن یک ستاره روی LCD نمایش داده شود؟

```
$regfile = "m32def.dat"
$crystal = 8000000
----------------------'
Config Portd = Input
----------------------'
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 =
Porta.6 , Db7 = Porta.7
Cursor Off
----------------------'
Declare Function Getkey As Byte
Dim I As Byte , Key As Byte
----------------------'


Do
Gosub Getkey
If Key <> 12 Then Gosub Gotkey
Loop
Gotkey:
Incr I
Waitms 500
Locate 1 , I : Lcd"*"
Return
End
Getkey:
Key = 12
Portd = &HFF
Portd.0 = 0
Waitms 20
If Pind.4 = 0 Then Key = 10
If Pind.5 = 0 Then Key = 0
```

```
If Pind.6 = 0 Then Key = 11
Portd = &HFF
Portd.1 = 0
Waitms 20
If Pind.4 = 0 Then Key = 7
If Pind.5 = 0 Then Key = 8
If Pind.6 = 0 Then Key = 9
Portd = &HFF
Portd.2 = 0
Waitms 20
If Pind.4 = 0 Then Key = 4
If Pind.5 = 0 Then Key = 5
If Pind.6 = 0 Then Key = 6
Portd = &HFF
Portd.3 = 0
Waitms 20
If Pind.4 = 0 Then Key = 1
If Pind.5 = 0 Then Key = 2
If Pind.6 = 0 Then Key = 3
Return
```

شروع کار با ADC

گاهی نیاز است که یک کمیت بیرونی مانند دما، شدت نور و ... اندازه گیری شود. جهت این کار از سنسورهای مربوطه استفاده می‌شود، سنسورها کمیت مورد نظر را به ولتاژ یا جریان تبدیل می‌کنند. ولتاژ تهیه شده توسط سنسور را می‌توان به یک ADC اعمال کرده و مقدار دیجیتال آن را تولید کنیم و این مقدار دیجیتال را با محاسبات ریاضی به عددی دسیمال جهت نمایش تبدیل کنیم.

ADCهای موجود در میکروکنترلرهای AVR، ۱۰ بیتی بوده و بنابراین می‌توانند اعداد بین ۰ تا ۱۰۲۳ را در خود ذخیره کنند. مثلا برای صفر ولت عدد صفر، برای پنج ولت عدد ۱۰۲۳ و برای 2.5 ولت عدد ۵۱۱ را در خود ذخیره می‌کنند. برای کار با این قابلیت باید ابتدا مثل سایر امکانات پیکربندی آن را انجام داد.

پیکربندی ADC ..... Config ADC

شکل کلی دستور: Config ADC = Single / Free , Prescaler = Auto , Reference = Opt

Single / Free: مربوط به مد انتخاب نمونه برداری می‌باشد که در حالت Single نمونه برداری از کانال دلخواه انتخاب شده و در متغیری از نوع Word ذخیره می‌شود و در حالت Free مقدار نمونه برداری در رجیستر مربوط به ADC ریخته می‌شود.

Prescaler: تعیین کلاک برای ADC می‌باشد که در حالت Auto کامپایلر با توجه به کریستال انتخاب شده بهترین کلاک را برای ADC در نظر می‌گیرد.

Reference: تعیین کننده ولتاژ مرجع است که یکی از حالات زیر می‌باشد:

Off: ولتاژ مرجع داخلی که برابر ۲.۵۶ ولت است خاموش شده و ولتاژ وصل شده به پایه Aref به عنوان ولتاژ مرجع در نظر گرفته می‌شود.

AVcc: ولتاژ پایه AVcc به عنوان ولتاژ مرجع در نظر گرفته می‌شود.

Internal: از ولتاژ مرجع داخلی ۲.۵۶ ولت استفاده می‌شود.

نکته : پایه AVcc تغذیه قسمت ADC می‌باشد و دلیل جدا بودن آن از تغذیه اصلی خود IC جلوگیری از تاثیر نویز بر روی ADC می‌باشد هر چند که در ICهای مثل Atmega8 پایه AVcc و Vcc یعنی تغذیه اصلی قطعه در داخل اتصال کوتاه هستند.

نکته: وقتی که از ADCها استفاده می‌شود دیگر نمی‌توان از پورت‌ها به عنوان I/O استفاده کرد.

فعال سازی ADC ..... Enable ADC

توسط این دستور ADC آماده نمونه برداری می‌شود.

راه اندازی ADC ..... Start ADC

با این دستور ADC شروع به کار می‌کند.

خواندن مقدار آنالوگ ..... GetADC

شکل کلی دستور : Var = Getadc (channel)

توسط این دستور مقدار نمونه برداری شده در مد Single، در متغیر Var از جنس Word ذخیره می‌شود.

Chanel: شماره ADC (پایه‌ای از پورتی که ADC می‌باشد) که سیگنال آنالوگ به آن اعمال می‌شود.

تعیین تعداد ارقام صحیح و اعشار ..... Fusing(X, "#.##")

توسط دستور بالا می‌توان تعداد ارقام صحیح و اعشار را با کم و زیاد کردن (#) در طرفین ممیز به دلخواه برای متغیر X که از جنس Single است را انتخاب کرد.

برای این کار روش دیگری هم وجود دارد که استفاده از فرمول زیر است:

```
Config single =Scientific , Digits = 2
```

که عدد ۲ همان تعداد ارقام اعشار است که می‌توان به دلخواه آن را تغییر داد.

مدارات عملی :

برنامه‌ای بنویسید که توسط آن مقادیر آنالوگ ورودی به اعداد دیجیتال تبدیل شده و روی LCD نمایش داده شود؟

تنظیمات اعمال شده روی برد آموزشی:

جامپرهای تغذیه و نمایشگر کاراکتری را متصل کرده همچنین جامپر AIN0/ADC را در حالت ADC قرار دهید.

```
$regfile = "m32def.dat"
$crystal = 8000000
```

```
_____________________________'
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 =
Porta.6 , Db7 = Porta.7
Cursor Off
_____________________________'
Dim I As Word
_____________________________'
Do
I = Getadc(3(
Cls
Lcd I
Waitms 500
Loop
End
```

همانطور که در مثال بالا ملاحظه شد این اعداد بین ۰ تا ۱۰۲۳ هستند. برای اینکه آن‌ها را قابل فهم کنیم باید از محاسبات ریاضی استفاده کنیم. به مثال زیر دقت کنید.

مثال : برنامه‌ی یک ولت متر ۰ تا ۵ ولت را بنویسید؟

```
$regfile = "m32def.dat"
$crystal = 8000000
_________________________'
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 =
Porta.6 , Db7 = Porta.7
Cursor Off
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Enable Adc
Start Adc
_________________________'
Dim A As Word
Dim B As Single
_________________________'
Do
```

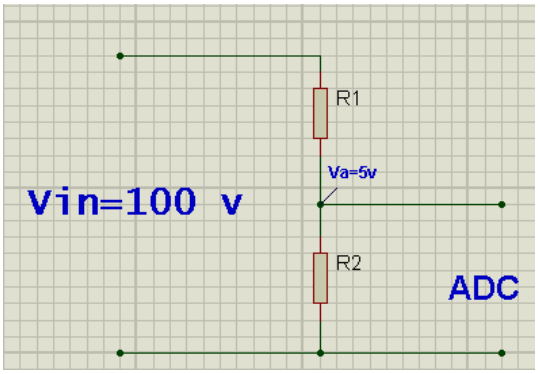
```
A = Getadc(3(
B = A / 204.6
Locate 1 , 2 : Lcd Fusing(b , "#.#") ; " Volt"
Waitms 500
Cls
Loop
End
```

همانطور که در بالا گفته شد ماکزیمم ولتاژ اعمالی به ADCها باید ۵ ولت باشد، خوب حالا اگر بخواهیم یک ولت متر ۰ تا ۱۰۰ ولت بسازیم باید چکار کنیم؟

در این موارد باید ولتاژ اعمالی تا ۵ ولت کاهش پیدا کند که یکی از راه‌های کاهش ولتاژ، استفاده از مدارات مقاومتی است. به مثال زیر که نحوه ساخت یک ولت متر ۰ تا ۱۰۰ و فرمولات مربوطه را نمایش می‌دهد دقت کنید.

مثال : برنامه یک ولت متر ۰ تا ۱۰۰ را بنویسید؟

برای این کار ابتدا مدار مقاومتی کاهش ولتاژ ۱۰۰ را به ۵ ولت طراحی می‌کنیم:



$$R_1 = \frac{R2(Vin-Va)}{Va}$$

$$R_2 = \frac{R1*Va}{(Vin-Va)}$$

در این فرمول‌ها شما مقدار یکی از مقاومت‌ها را قید کرده و مقاومت دیگر با توجه به دو فرمول بالا بدست می‌آید. در این دو فرمول مقدار Va ثابت و برابر ۵ ولت و همچنین Vin را با توجه به نیاز خود تعیین می‌کنید.

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

برای طراحی این مدار مقدار R_2 را برابر ۱۰ کیلواهم انتخاب کرده و با توجه به آن و قید در فرمول R_1، مقدار R_1 برابر ۱۹۰ کیلواهم بدست خواهد آمد که مقاومت استانداردی نیست و باید با سری و موازی کردن مقاومت‌های استاندارد، آن را بدست آورد.

تنها چیزی که باقی مانده است تبدیل ۱۰۲۳ گرفته شده توسط ADC و تبدیل آن به ۱۰۰ ولت به جای ۵ ولت است که برای این کار از فرمول زیر استفاده می‌شود:

$$X = \frac{1023}{Vin}$$

که X در اینجا می‌شود: ۱۰.۲۳

حالا با توجه به مطالب بالا برنامه مدار را می‌نویسیم:

```
$regfile = "m32def.dat"
$crystal = 1000000
'-----------------------
Config Portc = Output
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portc.2 , Db5 = Portc.3 , Db6 = Portc.4 , Db7 = Portc.5 , E =
Portc.1 , Rs = Portc.0
Config Adc = Single , Prescaler = Auto , Reference = Off
Start Adc
Enable Adc
'-------------------------
Dim A As Word , B As Single
Config Single = Scientific , Digits = 2
Cursor Off
'-------------------------
Do
  A = Getadc(0)
  B = A / 10.23
  Locate 1 , 2 : Lcd B ; "Volt"
  Wait 1
  Cls
Loop
End
```

نکات مهم:

- باید ماکزیمم ولتاژ اعمالی به ADC، ۵ ولت باشد در غیر این صورت پین مورد نظر خواهد سوخت.
- در مداراتی که احتمال رسیدن ولتاژ بالا به ADC انتظار می‌رود، باید از مدارات محافظ استفاده شود. علل خصوص در مدار آمپرمتر، طراحی مدارات محافظ خیلی مهم است.
- برای طراحی مداراتی که ولتاژ آن‌ها کمتر از ۱ ولت است می‌توان با استفاده از تقویت کننده‌های Op-Amp ولتاژ مورد نظر را تقویت کرد و به ۵ ولت یا بالاتر که کار با آن‌ها راحتر است تبدیل کرد.
- برای اندازه گیری ولتاژهای منفی ابتدا باید آن را توسط مدارات Op-Amp به ولتاژ مثبت تبدیل کرده، سپس آن را به ADC اعمال کرد، چون ADC فقط ولتاژ بین ۰ تا ۵ ولت را می‌شناسد.

وقفه‌ها

وقفه چیست؟ گاهی اوقات نیاز است که میکرو همزمان دو عمل را انجام دهد مثلا هم اطلاعاتی را در حافظه ثبت کند و هم تعداد پالس‌های اعمال شده به یک پایه را بشمارد. در عمل هیچ پردازشگری در آن واحد نمی‌تواند دو عمل را انجام دهد. برای این منظور از وقفه‌ها استفاده می‌شود.

در میکروکنترلرهای AVR دو دسته وقفه طراحی شده است:

وقفه های داخلی و وقفه های خارجی.

وقفه‌های داخلی

برای اکثر امکانات و خصوصیات یک میکروکنترلر طراحی شده و برای هر یک بیت، پرچمی که به آن بیت وقفه گفته می‌شود تعبیه شده است، تا یک شدن آن بیت، نشان دهنده وقوع وقفه برای خصوصیت مورد نظر باشد.

وقفه‌های خارجی

تعدادی از پایه‌های میکروکنترلرهای AVR را می‌توان به عنوان یک عمل دهنده وقفه پیکربندی نمود. این پایه‌ها در هر میکروکنترلر با کلمه INTx مشخص شده است که در آن x، عدد وقفه خارجی را نشان می‌دهد.

فعال سازی کلی وقفه‌ها ..... Enable Interrups

توسط این دستور وقفه کلی فعال شده و می‌توان با توجه به نیاز از وقفه مورد نظر استفاده کرد.

فعال سازی وقفه مورد نظر ..... Enable Interrupt

بعد از فعال سازی وقفه کلی می‌توان هر وقفه را با ذکر نام آن فعال کرد که در اینجا Interrupt نام وقفه مورد نظر است.

نکته: برای غیر فعال سازی وقفه‌ها از دستور Disable استفاده می‌شود.

ما وقفه را لازم داریم تا در زمان اجرای برنامه اصلی یک برنامه فرعی هم اجرا شود. برای این کار باید در زمان وجود وقفه به یک زیربرنامه پرش کنیم. این کار توسط دستور بالا انجام می‌شود.

Interrupt: نام وقفه مورد نظر.

Lable: نام برچسب زیر برنامه وقفه.

مقایسه کننده آنالوگ

یکی دیگر از امکانات جالب موجود در میکروکنترلرهای AVR واحد مقایسه آنالوگ می باشد که با استفاده از آن می توان دو موج آنالوگ را با هم مقایسه کرد. عملکرد این قسمت مشابه عملکرد آپ امپ در مد مقایسه کننده می باشد و در صورتی که ولتاژ پایه AIN0 از AIN1 بیشتر باشد، خروجی مقایسه کننده ACO یک می شود.

سوالی که شاید به ذهن برخی از افراد خطور کنه این است که با وجود مبدل آنالوگ به دیجیتال دیگر چه نیازی به این بخش می باشد؟

در جواب باید گفت سرعت عملکرد این بخش در مقایسه با مبدل انالوگ به دیجیتال بسیار بیشتر بوده و همین سرعت باعث احساس نیاز به چنین بخشی را فراهم کرده است.

پیکربندی واحد مقایسه آنالوگ در بسکام

فرم کلی دستور :

CONFIG ACI = ON|OFF, COMPARE = ON|OFF, TRIGGER=TOGGLE|RISING|FALLING

ON|OFF : روشن و خاموش کردن مقایسه کننده آنالوگ.

COMPARE = ON|OFF : با انتخاب ON در این قسمت، خروجی مقایسه کننده به ورودی مد Capture تایمر کانتر متصل می شود.

TRIGGER=TOGGLE|RISING|FALLING : با استفاده از این قسمت می توان مشخص کرد در چه حالتی وقفه مقایسه کننده فعال شود.

مدارات عملی :

برنامه ای بنویسید که با استفاده از واحد مقایسه کننده آنالوگ دو ولتاژ آنالوگ را با هم مقایسه کرده و در صورت افزایش ولتاژ پایه مثبت، یک LED از پورت A را روشن کند؟

تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه را بر روی ۵ ولت قرار داده و همچنین جامپر ADC/AIN0 را در حالت AIN0 قرار دهید و در نهایت جامپر AIN1 را جهت اعمال ولتاژ ۳.۳ ولت به عنوان رفرنس متصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'----------------------------
Config Aci = On , Compare = Off , Trigger = Rising
Config Porta = Output
Porta = &HFF
'----------------------------

Do
  If Acsr.5 = 1 Then Porta.0 = 0
  Porta = &HFF
Loop
End
```

شروع کار با تایمر-کانتر ۱

پیکربندی تایمر کانتر یک در مد تایمر:

پیکربندی تایمر کانتر ۱ به عنوان تایمر..... Config Timer1

config TIMER1= TIMER, PRESCALE=1|8|64|256|1024 , Clear Timer = 0|1

توسط دستور بالا، تایمر کانتر یک به عنوان تایمر، پیکربندی می‌شود. این تایمر ۱۶ بیتی بوده و حداکثر تا ۶۵۵۳۵ را می شمارد.

مقادیر 1|8|64|256|1024 یا PRESCALE که همواره باید فقط یکی از آن‌ها انتخاب شود برای تقسیم فرکانس کار میکروکنترلر و بدست آوردن فرکانس تایمر استفاده می‌شود که رابطه آن به شکل زیر است:

PRESCALE / فرکانس کار میکرو = فرکانس تایمر

این فرکانس نشان می‌دهد که برای هر بار شمارش چقدر زمان نیاز است.

Clear Timer = 0|1: اگر ۰ باشد در صورت متوقف شدن و فعال سازی مجدد به ادامه شمارش مقدار قبلی ادامه می‌دهد، اما اگر ۱ باشد بعد از متوقف شدن و فعال سازی مجدد مقدار آن صفر می‌شود.

فعال سازی و روشن کردن تایمر .....START

توسط دو دستور زیر که بهتر است همیشه با هم استفاده شوند، تایمر شروع به کار می‌کند.

Enable TIMER1

Start TIMER1

نکته : با استفاده از دستور Stop TIMER1 می‌توان در جاهای مختلف برنامه تایمر را متوقف کرد.

نوشتن و خواندن تایمر ..... Var=TIMER1

توسط دستورات زیر می‌توان مقدار تایمر را در یک متغیر قرار داد و یا به تایمر مقدار داد تا از آن مقدار شروع به شمارش کند.

TIMER1=Var

Var=TIMER1

متغیر Var باید از جنس Word باشد.

وقفه تایمر ۱ ..... ENABLE OVF1

با رسیدن مقدار تایمر یک به ۶۵۵۳۵ مقدار تایمر باز به صفر برگشته اما وقفه آن با نام OVF1 یک می‌شود و می‌تواند به یک زیربرنامه پرش کند. شکل کلی دستور به صورت زیر است:

On OVF1 Lable

نکته: در موقع کار با این وقفه‌ها حتما باید وقفه سراسری فعال باشد.

مدارات عملی :

۱) برنامه ای بنویسید که بتواند به طور تقریبی مقدار تاخیر ۱ ثانیه را ایجاد کرده و آن را روی LCD نمایش دهد؟

تنظیمات اعمال شده روی برد آموزشی:

فیوزبیت کلاک را بر روی یک مگاهرتز داخلی تنظیم کنید و جامپرهای مربوط به تغذیه و CHLCD را متصل کنید.

```
$regfile = "m32def.dat"
$crystal = 1000000
'------------------------
Enable Interrupts
Config Porta.1 = Output
Porta.1 = 0
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1
Enable Timer1
Start Timer1
Enable Ovf1
Timer1 = 64535
```

```
Cursor Off
'------------------------
On Ovf1 Shomar
Dim A As Word , B As Word
'------------------------
Initlcd
Do

Loop
End
Shomar:
   Incr B
   Locate 1 , 1 : Lcd B ; " S"
   Timer1 = 64535
   Return
```

۲) توسط تایمر ۱، یک ساعت ساده را طراحی کنید؟

```
$regfile = "m32def.dat"
$crystal = 1000000
'------------------------
Enable Interrupts
Config Porta.1 = Output
Porta.1 = 0
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Config Timer1 = Timer , Prescale = 1024 , Clear Timer = 1
Enable Timer1
Start Timer1
Enable Ovf1
Timer1 = 64535
Cursor Off
'------------------------
On Ovf1 Shomar
Dim S As Byte , D As Byte , Sa As Byte
'------------------------
Initlcd
```

```
Do

Loop
End
Shomar:
   Incr S
   If S = 60 Then
     S = 0
     Incr D
     If D = 60 Then
      D = 0
      Incr Sa
      If Sa = 24 Then
        Sa = 0
      End If
     End If
   End If
   Home
   If Sa < 10 Then
     Lcd "0"
   End If
   Lcd Sa
   Lcd ":"
   If D < 10 Then
     Lcd "0"
   End If
   Lcd D
   Lcd ":"
   If S < 10 Then
     Lcd "0"
   End If
   Lcd S
   Timer1 = 64535
   Return
```

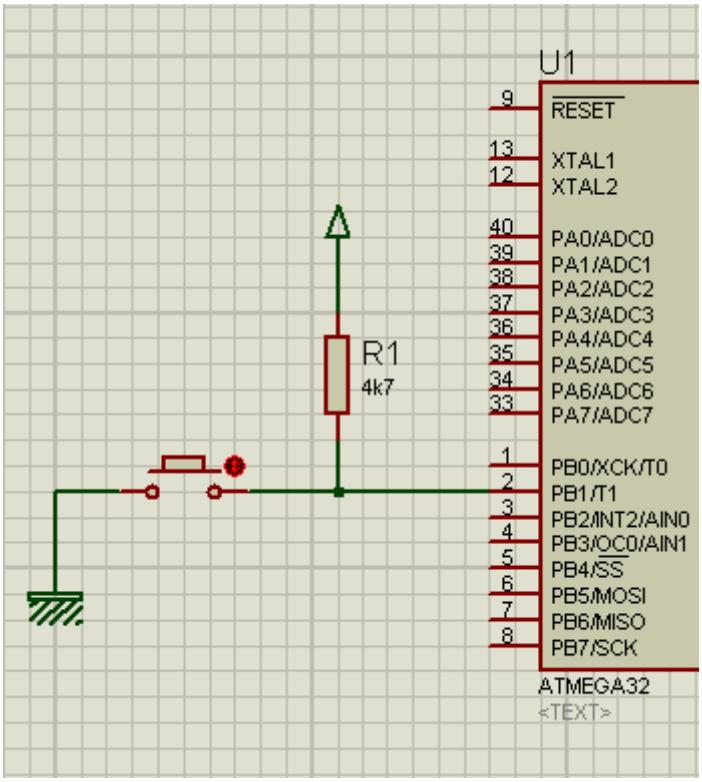
پیکربندی تایمر کانتر ۱ به عنوان کانتر

```
Config Timer1=Counter.....  پیکربندی کانتر ۱
```

شکل کلی دستور: Config Timer1 = Counter , Edge = Rising|falling

EDGE: در حالت Rising پایه به لبه بالارونده و در حالت Falling پایه به لبه پایین رونده حساس است. مثلا اگر کلید را طبق شماتیک زیر وصل کنید در حالت Falling به محض فشار دادن کلید یک واحد به Counter اضافه می‌شود ولی اگر از مد Rising استفاده شود تا انگشت خود را از روی کلید برندارید به Counter اضافه نمی‌شود.

توسط این دستورات کانتر ۱ پیکربندی شده و برای کار با آن باید کلید یا پالس تحریک به پایه T1 اعمال شود.



فعال سازی و روشن کردن کانتر ..... START

توسط دستور روبه رو کانتر فعال و آماده کار می‌شود: Start Counter 1

نکته: با استفاده از دستور Stop Counter 1 می‌توان در جاهای مختلف برنامه کانتر را متوقف کرد.

نوشتن و خواندن کانتر ۱ ..... Var=Counter1

همانند تایمر ۱ می‌توان مقدار خاصی را برای شروع شمارش به کانتر داد یا مقدار آن را در یک متغیر از جنس Word ذخیره کرد.

Var = Counter1 || Counter1= Var

فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

وقفه کانتر ۱ ..... OVF1

وقفه کانتر یک هم عینا مانند وقفه تایمر می‌باشد یعنی با رسیدن مقدار شمارش، وقفه کانتر یک، ۱ شده و مقدار شمارش به صفر برمی گردد.

Enable OVF1 || ON OVF1 Lable

پیکربندی تایمر کانتر ۱ به عنوان مولد موج PWM

PWM یا همان مدولاسیون عرض پالس یکی دیگر از امکانات تایمر کانتر ۱ می‌باشد که کاربرد وسیعی در کنترل دور موتورهای DC دارد.

پیکربندی Config PWM PWM .....

شکل کلی دستور:

Config TIMER=PWM,PWM=8|9|10,Prescale=1|8,….,

COMPARE A|B=CLEAR UP|CLEAR DOWN_|DISCONNECT

تایمر کانتر ۱ دارای دو خروجی PWM با نام‌های OC1A و OC1B می‌باشد. دو رجیستر برای قرارگیری مقادیر PWM خروجی با نام‌های PWM1A و PWM1B وجود دارد که می‌توان در آن‌ها نوشت و یا آن‌ها را خواند.

PWM=8|9|10: برای پیکربندی PWM به صورت مد ۸، ۹ و ۱۰ بیت می‌باشد.

Prescale: همانند دو مد تایمر و کانتر برای تنظیم فرکانس موج استفاده می‌شود. برای تنظیم فرکانس موج در مدهای مختلف از فرمولات زیر استفاده می‌کنیم:

PWM=8 ⟹ Fpwm= فرکانس کار میکرو / (۲۵۶*PRESCALE)

PWM=9 ⟹ Fpwm= فرکانس کار میکرو / (۵۱۲*PRESCALE)

PWM=10 ⟹ Fpwm= فرکانس کار میکرو / (۱۰۲۴*PRESCALE)

COMPARE A|B PWM: این گزینه‌ها نوع تغییرات سیگنال PWM را مشخص می‌کنند که تغییرات به شرح زیر است:

CLEAR UP: موج PWM از سطح یک شروع می‌شود.

CLEAR DOWN: موج PWM از سطح صفر شروع می‌شود.

DISCONNECT: در این حالت خروجی PWM قطع می‌شود.

نکته: در حالت کلی اگر از CLEAR UP استفاده کنید هرچه مقدار عدد داده شده به رجیستر کمتر باشد سرعت بیشتر است و در حالت CLEAR DOWN برعکس.

فعال سازی START PWM .....

توسط دستورات زیر PWM آماده تولید پالس می‌شود:

```
Enable TIMER1
Start TIMER1
```

خواندن و نوشتن رجیسترهای PWM ..... Var=PWM1A

توسط دستورات زیر می‌توان مقادیر PWM را خواند و یا در آن نوشت. در حالت PWM=8 جنس Var می‌تواند از جنس Byte باشد اما در حالت‌های PWM=9|10 جنس متغیر باید Word باشد.

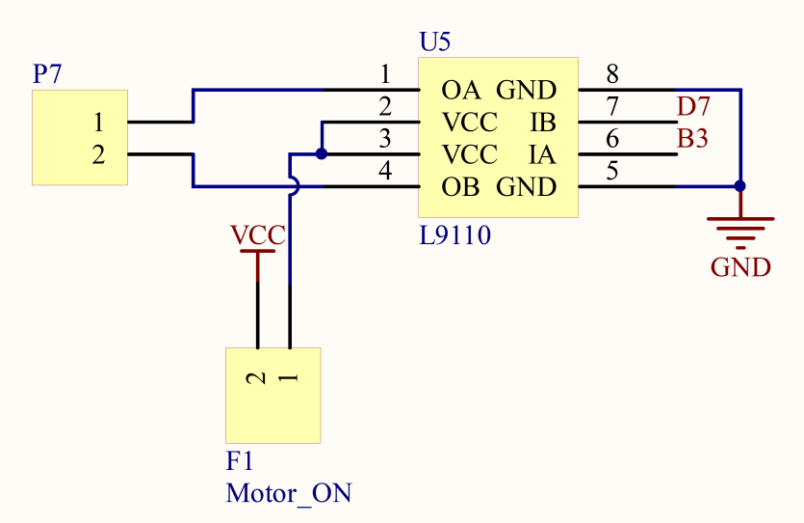
```
PWM1A|B = Var
Var = PWM1A|B
```

مدارات عملی :

۱) با استفاده از تایمر ۰ در مد PWM برنامه ای بنویسید که سرعت یک موتور DC هر ۲۵۰ میلی ثانیه افزایش یابد؟

توجه: توضیحات داده شده در قسمت های بالا مروبط به تایمر ۱۶ بیتی ۱ بود اما در برد آموزشی از PWM تایمرهای ۰ و ۲ استفاده شده است. دقت کنید عملکرد تایمرها شبیه هم بوده و تفاوتی نمی کند.در بسکام تنظیمات مربوط به تایمر ۰ محدودیت هایی دارد که در این قسمت به طور مستقیم از طریق رجیسترها تایمر را تنظیم کرده ایم.

شماتیک اتصال درایور به میکروکنترلر:



تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه و Motor را متصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------------
Config Portb.3 = Output
Config Portd.7 = Output
Tccr0 = &H79
Tcnt0 = &H00
Ocr0 = &H00
Dim I As Byte
'-----------------------------
```

```
Portd.7 = 0
Do
 Ocr0 = Ocr0 + 20
 Waitms 250
Loop
End
```

۲) برنامه‌ای بنویسید که بتوان توسط دو کلید دور یک موتور را کاهش و افزایش داد؟

```
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------------
Config Portd.2 = Input
Config Portd.3 = Input
Config Portb.3 = Output
Config Portd.4 = Output
Config Portd.7 = Output
Portd.2 = 1
Portd.3 = 1
Portd.4 = 0
'-----------------------------
Tccr0 = &H79
Tcnt0 = &H00
Ocr0 = &H00
Config Int0 = Rising
Config Int1 = Rising
```

```
Enable Interrupts
Enable Int0
Enable Int1
On Int0 Afzayesh
On Int1 Kahesh
Dim I As Byte
'-----------------------------
Portd.7 = 0
Do
 Ocr0 = I
Loop
End
Afzayesh:
I = I + 25
Return
Kahesh:
I = I - 25
Return
```


فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

راه اندازی LCDهای گرافیکی سری SED

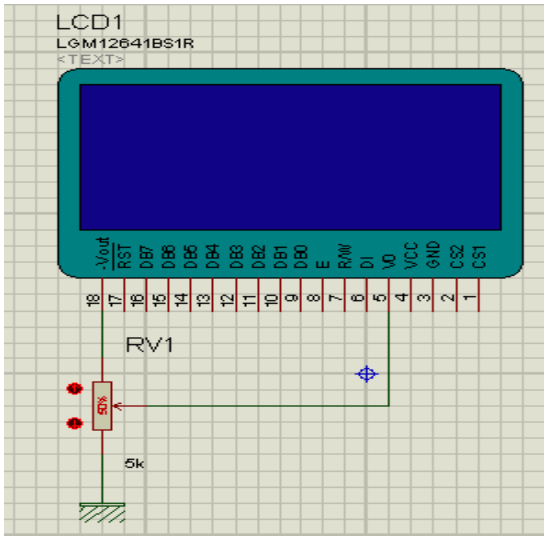
LCDهای کاراکتری محدودیت‌های زیادی دارند، مثلا نمی‌توان کلمات فارسی را به صورت پیوسته دید یا از افکت استفاده کرد، ولی در این LCDها می‌توان به هر پیکسل که نیاز است دسترسی داشت و به همین دلیل محدودیتی برای نمایش بر روی آن وجود ندارد. LCDهای گرافیکی مانند LCDهای کاراکتری در سایزهای مختلف موجود می‌باشند.

این LCD دارای دو نیم صفحه چپ و راست می‌باشد. نحوه سیم بندی در این LCDها با توجه به شرکت سازنده متفاوت بوده و بهتر است قبل از راه اندازی آن دیتاشیت LCD را مشاهده کنید.

شرح پایه‌های یک نمونه موجود در بازار:

شماره پایه	Symbol	توضیحات
۱	GND	پایه تغذیه
۲	Vcc	پایه تغذیه
۳	Vo	تنظیم کنتراست
۴	RS	ورودی دیتا و دستورالعمل
۵	RW	خواندن و نوشتن دیتا
۶	E	فعال سازی ورودی دیتا
۷ – ۱۴	Db0 To Db7	پایه های دو جهته برای خواندن یا نوشتن داده در LCD
۱۵	CS1	فعال ساز چیپ ۱
۱۶	CS2	فعال ساز چیپ ۲
۱۷	RST	بازنشانی یا ریست کردن LCD
۱۸	VEE	تولید ولتاژ منفی جهت تنظیم کنتراست
۱۹	A	آند LED پس زمینه
۲۰	K	کاتد LED پس زمینه

نحوه اتصال پایه‌ها برای تنظیم کنتراست:



در پیکربندی LCDهای گرافیکی باید به چهار نکته دقت کرد:

- معرفی فونت و کتابخانه لازم جهت کار با LCD
- تعیین سایز LCD
- Dataport: تعیین پورتی از میکروکنترلر برای اتصال به پایه‌های Db0 الی Db7
- Controlport: تعیین پورتی از میکروکنترلر برای اتصال به پایه‌های کنترلی میکروکنترلر شامل: CS1, CS2, E, RST, RS, R/W.

دستورات مربوط به پیکربندی و کار با LCDهای گرافیکی

فراخوانی کتابخانه ..... $LIB

توسط این دستور کتابخانه مورد نیاز با توجه به LCD فراخوانی می‌شود. مثلا در اینجا LCD مورد استفاده ما سری SED با چیپ Ks108 می‌باشد. $lib "glcdKS108.LBX"

فراخوانی فونت ..... $Include

فراخوانی فونت مورد استفاده: $include "font8x8.font"

نکته: بایستی فونت مورد نظر را از مسیر نصب برنامه بسکام (C:\Program Files\MCS Electronics\BASCOM-AVR\Samples) پیداکرده ودرمسیر ذخیره فایل برنامه خود کپی کنید و سپس آن را

دربرنامه فراخوانی کنید در غیر این صورت در هنگام کامپایل برنامه خطایی مبتنی بر ناشناس بودن قسمت فونت ظاهر می‌شود.

پیکربندی پایه های LCD ..... Config GRAPHLCD

شکل کلی دستور:

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

تعیین سایز LCD، تعیین Dataport، تعیین Controlport و نحوه اتصال پایه‌ها.

Ce: همان پایه CS1 است.

Ce2: همان پایه CS2 است.

Cd: همان پایه RS است.

Rd: همان پایه RW است.

نکته: قبل از نمایش متن مورد نظر بایستی فونت مربوطه را Set کنید: Setfont Font8x8

پاک کردن صفحه نمایش ..... CLS

این دستور برای پاک کردن صفحه مورد استفاده قرار می‌گیرد که به سه صورت کاربرد دارد:

- CLS: پاک کردن کل صفحه نمایش.
- CLS Text: پاک کردن متون روی صفحه نمایش.
- CLS Graph: پاک کردن اشکال گرافیکی روی صفحه نمایش که هیشه قبل از نمایش لازم است.

نمایش متغیر و رشته کاراکتر ثابت ..... LCDAT

برای نمایش مقدار یک متغیر یا یک رشته کاراکتر ثابت در مکان دلخواه مورد استفاده قرار می‌گیرد که شکل کلی آن به صورت زیر است:

Lcdat x , y , var , inv

X = سطر مورد نظر که می‌تواند اعداد ۱ الی ۸ باشد.

Y = ستون مورد نظر که می‌تواند ۰ تا ۱۲۷ باشد.

Var = متغیری که مقدار آن نمایش داده می‌شود. برای نمایش رشته ثابت باید آن را در بین جفت کتیشن قرار داد.

Inv = اگر مقدار آن صفر باشد متن سیاه و در صورتی که ۱ باشد متن سفید نمایش داده می‌شود.

روشن یا خاموش کردن پیکسل ..... PSET

روشن یا خاموش کردن یک پیکسل درمختصات تعیین شده.

شکل کلی دستور:

Pset x , y , value

X,Y = مختصات نقطه مورد نظر.

Value = وقتی ۰ باشد پیکسل مورد نظر خاموش و وقتی 1 باشد پیکسل مورد نظر روشن می‌شود.

رسم خط LINE .....

توسط این دستور می‌توان در صفحه نمایش با مختصات تعیین شده خط راست رسم کرد.

شکل کلی دستور: Line (X0 , Y0) – (X1 , Y1) , Color

(X0 , Y0): مختصات نقطه شروع.

(X1 , Y): مختصات نقطه انتها.

Color: رنگ خط است و اگر ۰ باشد خط ساده و اگر ۱ باشد خط با رنگ معکوس نسبت به زمینه رسم می‌شود.

رسم دایره ..... CIRCLE

رسم دایره درصفحه نمایش با مشخصات تعیین شده.

شکل کلی دستور:

Circle (X0 , Y0) , Radius ,Color

(X0 , Y0): مختصات مرکز دایره.

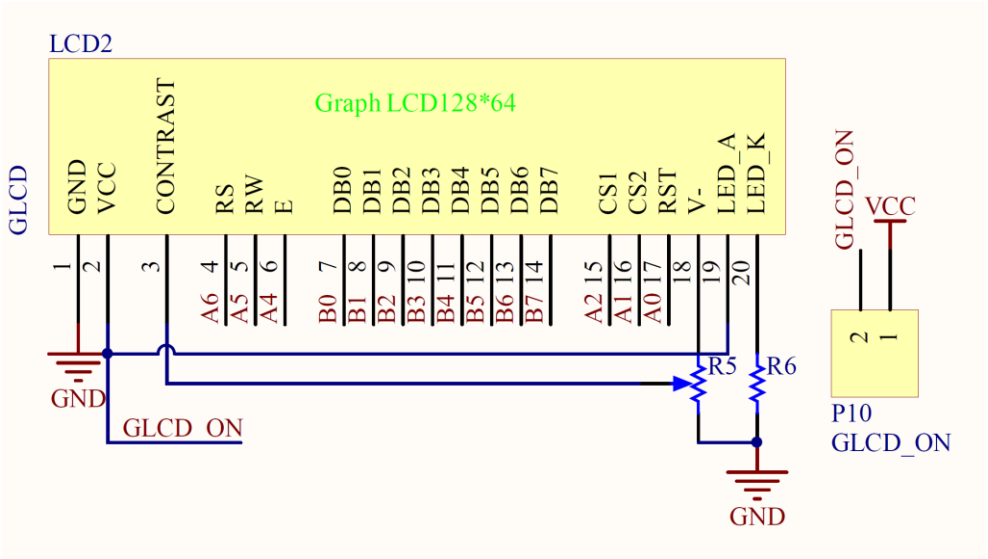
Radius: شعاع دایره.

Color: رنگ دایره است و اگر ۰ باشد دایره ساده و اگر ۱ باشد دایره با رنگ معکوس نسبت به زمینه رسم می‌شود.

مدارات عملی :

۱- برنامه ای بنویسید که بتواند عبارت " www.ECA.ir " را روی در دو حالت بر روی LCD چاپ کند؟

شماتیک مربوط به اتصال نمایشگر گرافیکی:



تنظیمات روی برد:

جامپر تغذیه را در حالت ۵ ولت قرار داده و جامپر GLCD را نیز وصل کنید.

```
$regfile = "m32def.dat"

$crystal = 8000000

------------------------'

$lib "glcdKS108.LBX"
```

```
$include "font8x8.font"

Config Graphlcd = 128 * 64sed , Dataport = Portb , Controlport = Porta , Ce = 1 , Ce2 = 2 , Cd = 6 , Rd = 5 , Enable = 4 , Reset = 0

'-----------------------

Cls Graph

Setfont Font8x8

Lcdat 2 , 20 , "www.ECA.ir" , 0

Lcdat 4 , 20 , "www.ECA.ir" , 1

End
```

۲- برنامه‌ای بنویسید که چهار دایره متحد المرکز را رسم کند؟

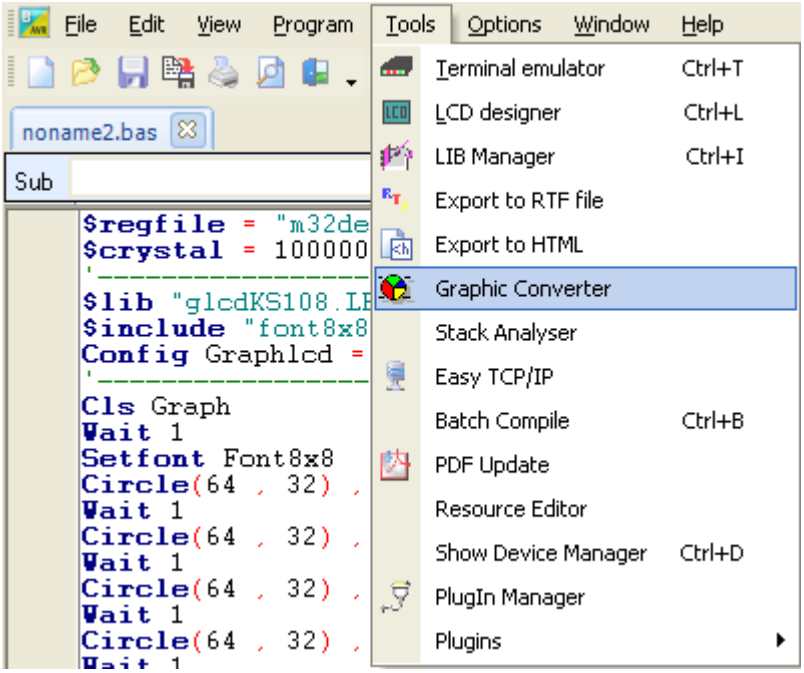
```
$regfile = "m32def.dat"
$crystal = 8000000
'-----------------------
$lib "glcdKS108.LBX"
Config Graphlcd = 128 * 64sed , Dataport = Portb , Controlport = Porta , Ce = 1 , Ce2 = 2 , Cd = 6 , Rd = 5 , Enable = 4 , Reset = 0
'-----------------------
Cls Graph
Waitms 500
Circle(64 , 32) , 5 , 1
Wait 1
Circle(64 , 32) , 10 , 1
Wait 1
Circle(64 , 32) , 15 , 1
Wait 1
Circle(64 , 32) , 20 , 1
Wait 1
Circle(64 , 32) , 25 , 1
End
```

نکات :

- در موقع کار با LCD ها اگر تصویر شما از وسط دو نیم شده، نمایش داده شود یا ۱۸۰ درجه اختلاف داشته باشد در این صورت پایه‌های CS1 و CS2 را معکوس متصل کرده‌اید.
- پایه‌ی RESET در کامپایلر بسکام معرفی می‌شود ولی در اکثر موارد مستقیم به Vcc اتصال پیدا می‌کند.

نمایش تصویر بر روی LCDهای گرافیکی

برای این کار ابتدا باید عکس مورد نظر را به اندازه LCD مورد استفاده در آورده و سپس آن را با پسوند BMP یا DIP ذخیره کنیم. سپس در برنامه بسکام به مسیر زیر رفته و تنظیمات زیر را انجام می‌دهیم:



۱- LCD Type: تعیین سایز LCD مورد استفاده.

۲- Font: اندازه فونت که برای LCDهای SED ۸*۸ انتخاب می‌شود.

۳- فعال کردن گزینه Sed Series.

پس از انجام این تنظیمات تصویر مورد نظر را توسط دکمه Load فراخوانی می‌کنیم و سپس با زدن دکمه Save تصویر را در آدرس برنامه خود با پسوند BGF ذخیره می‌کنیم.

دستورات کامپایلر برای نمایش تصویر

نمایش تصویر ..... SHOWPIC

توسط این دستور تصویر ذخیره شده با پسوند BGF آماده نمایش می‌شود.

شکل کلی دستور: Showpic x , y , Lable , inv

X و Y: مختصات نقطه شروع رسم.

Lable: برچسب قسمت فراخوانی تصویر مورد نظر.

Inv: ۰ جهت نمایش ساده و ۱ جهت نمایش به صورت Invert شده.

نکته: در صورت استفاده از دستور CLS تصویر مورد نظر به سرعت از روی صفحه نمایش پاک می‌شود اما اگر بخواهیم پاک شدن تصویر همراه با افکت‌های زیبا باشد می‌توان از دستورات Circle و Line همراه با حلقه For Next افکت‌های زیبایی را ایجاد کرد.

مدارات عملی :

۱) برنامه‌ای بنویسید که بتوان یک عکس ساده را روی LCD گرافیکی نمایش دهد؟

```
$regfile = "m32def.dat"
$crystal = 8000000
'------------------------
$lib "glcdKS108.LBX"
Config Graphlcd = 128 * 64sed , Dataport = Portb , Controlport = Porta , Ce = 1 , Ce2 = 2 , Cd = 6 , Rd = 5 , Enable = 4 , Reset = 0
'------------------------
Dim I As Byte
'------------------------
Cls Graph
```

```
Do
    Showpic 0 , 0 , Pic , 1
    For I = 64 To 0 Step -1
       Line(0 , I) -(128 , I) , 0
       Waitms 100
       Next I
    Showpic 0 , 0 , Pic , 0
    For I = 0 To 128
       Line(i , 0) -(i , 64) , 0
       Waitms 100
    Next I
Loop
End
Pic:
$bgf "eca.bgf"
```

۲) برنامه‌ای بنویسید که تصویر پاک شده از روی صفحه نمایش همراه با افکت باشد؟

```
$regfile = "m32def.dat"
$crystal = 8000000
------------------------'
$lib "glcdKS108.LBX"
Config Graphlcd = 128 * 64sed , Dataport = Portb , Controlport = Porta , Ce = 1 , Ce2 = 2 ,
Cd = 6 , Rd = 5 , Enable = 4 , Reset = 0
------------------------'
Dim I As Byte
------------------------'
```

```
Cls Graph

Do
Showpic 0 , 0 , Pic , 1
For I = 64 To 0 Step -1
Line(0 , I) -(128 , I) , 0
Waitms 100
Next I
Showpic 0 , 0 , Pic , 0
For I = 0 To 128
Line(i , 0) -(i , 64) , 0
Waitms 100
Next I
Loop
End
```

Pic:

```
$bgf "eca.bgf"
```

شروع کار با Step Motor

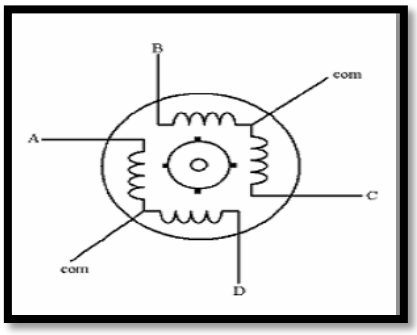
موتورهای پله‌ای وسایل الکترومکانیکی هستند که پالس‌های دیجیتالی را به یک جابجایی یا چرخش معین تبدیل می‌کنند. در کاربردهایی مانند راه اندازی دیسک سخت، چاپگرهای مغناطیسی، رباتیک و کنترل دقیق ماشین ابزارها، از موتورهای پله ای استفاده می‌شود.

انواع موتورهای پله‌ای

موتورهای پله‌ای در دو نوع پنج سیمه و شش سیمه وجود دارند که متداول‌ترین موتورهای پله‌ای شش سیمه می‌باشند.
این موتورها به موتورهای پله‌ای چهار فاز یا چهار قطبی نیز معروف هستند.
در این موتورها ۴ سیم پیچ استاتور وجد دارد که دو به دو با سیم سر وسط جفت شده‌اند.
سر یا سرهای وسط با توجه به برنامه راه اندازی موتور به Vcc یا GND وصل می‌شوند.

تشخیص پایه‌های موتور پله‌ای

با استفاده از یک اهم متر می‌توان پایه‌های موتور را تشخیص داد، به این استدلال که پایه‌های مشترک هیچ ارتباطی با هم ندارند و اندازه مقاومت الکتریکی بین هر دو سیم پیچ جفت شده، دو برابر مقاومت هر سیم پیچ نسبت به سر وسط است.



زاویه پله موتور

زاویه پله یکی از مشخصه‌های مهم موتورهای پله‌ای است که نشان می‌دهد موتور به ازای هر پله چند درجه می‌چرخد. زاویه پله در موتورهای مختلف متفاوت می‌باشد.

تعداد پله‌های یک دور کامل در هر موتور پله‌ای از رابطه زیر محاسبه می‌شود:

زاویه پله / ۳۶۰ = تعداد پله در دور

راه اندازی موتور پله‌ای

چرخش در موتورهای پله‌ای براساس جذب قطب غیرهمنام استوار است. درهر مرحله با دادن پالس الکتریکی به یکی از سیم پیچ‌های استاتور، شفت موتور به اندازه یک پله می‌چرخد تا قطب‌های غیر همنام روتور و استاتور دریک راستا قرار گیرند. بنابراین برای چرخش شفت موتور باید به صورت متوالی در هر مرحله به سیم پیچ‌های مناسب پالس الکتریکی اعمال کرد.

راه اندازی موتورهای پله‌ای به دو روش صورت می‌گیرد:

- ✓ تحریک پله کامل
- ✓ تحریک نیم پله

تحریک پله کامل

تحریک پله کامل به دو روش صورت می‌گیرد:

روش اول: تحریک یک سیم پیچ در هر مرحله

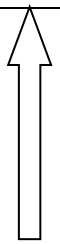
در این روش در هر مرحله و به ترتیب پالسی را برای یکی از سیم پیچ‌های D,C,B,A می‌فرستیم، تکرار این روند به صورت متوالی باعث چرخش مداوم محور موتور (روتور) خواهد شد. درصورت اجرای فرامین جدول زیر از بالا به پایین، چرخش روتور در جهت عقربه‌های ساعت و در صورت اجرای فرامین جدول از پایین به بالا، چرخش روتور در جهت خلاف عقربه‌های ساعت خواهد بود.

کد هگز	سیم پیچ D	سیم پیچ C	سیم پیچ B	سیم پیچ A	شماره پله
۰۸	۰	۰	۰	۱	۱
۰۴	۰	۰	۱	۰	۲
۰۲	۰	۱	۰	۰	۳
۰۱	۱	۰	۰	۰	۴

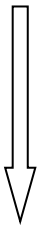
روش دوم: تحریک دو سیم پیچ در هرمرحله

در این روش در هر مرحله و به طور همزمان دو سیم پیچ تحریک می‌شوند. زاویه طی شده در این روش با روش قبل یکسان بوده و تنها تفاوت در موقعیت توقف روتور خواهد بود. دراین حالت روتور بین دو قطب تحریک شده استاتور توقف می‌کند.

فرامین لازم جهت این روش راه اندازی در جدول زیر مشاهده می‌شود.



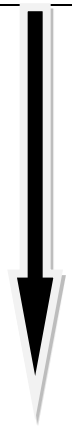
شماره پله	سیم پیچ A	سیم پیچ B	سیم پیچ C	سیم پیچ D
۱	۱	۰	۰	۱
۲	۱	۱	۰	۰
۳	۰	۱	۱	۰
۴	۰	۰	۱	۱



تحریک نیم پله

برای دست یابی به پله‌های ریزتر ودرنتیجه داشتن دقت بیشتر می‌توان با جریان دهی مناسب به موتور، آن را در زوایای کوچک‌تری نسبت به پله عادی خود موتور، بچرخانیم. بنابراین با ترکیب دوروش قبلی می‌توانیم موتور را در حالت نیم پله تحریک کنیم. به عنوان مثال موتوری که دارای زاویه پله 2 می‌باشد در تحریک پله کامل با طی 180 پله یک دور کامل می‌زند. اما با تحریک نیم پله با طی 360 پله یک دور کامل خواهد زد. فرامین لازم برای تحریک نیم پله در جدول زیر آمده است.

شماره پله	سیم پیچ A	سیم پیچ B	سیم پیچ C	سیم پیچ D
۱	1	0	0	0
۲	1	1	0	0
۳	0	1	0	0
۴	0	1	1	0
۵	۰	۰	۱	۰
۶	۰	۰	۱	۱
۷	۰	۰	۰	۱
۸	۱	۰	۰	۱



نکته: موتورهای پله ای وسایل پرمصرفی (پرتوان) هستند ومیکروکنترلرها مستقیما توان لازم جهت راه اندازی آن‌ها را ندارند. بنابراین باید با استفاده از بافرهای جریان توان لازم جهت راه اندازی موتورپله ای را ایجاد کنیم که بافر جریان می‌تواند ترانزیستورهای دارلینگتون TIP 122 یا ICهای راه اندازی مانند ULN2803 و L298 باشد.

راه اندازی موتور پله ای با L298

IC، L298 در دو مدل تولید می‌شود که در اینجا از مدل Moltiwatt استفاده می‌شود. این قطعه توانایی جریان دهی تا ۴ آمپر و ولتاژ ۴۶ ولت را دارد. تنها عیب این قطعه نبود دیودهای هرزگرد داخل آن است که باید از بیرون به IC متصل شود.

نکته: دیودهای هرزگرد به صورت پک پل دیودی، بر روی برد موجود است و نیازی به اتصال آن‌ها در خارج از برد نیست.

نحوه اتصال پایه های L 298:

وظیفه پایه	نام پایه	Powerso20	Multiwatt 15
- بسته به نوع میکروکنترلر ولتاژی بین ۱.۵ تا ۵۰ ولت به این پایه اعمال می‌شود. - ولتاژ این پایه ۱ ولت از تغذیه موتور باید بیشتر باشد. - برای کاهش نویز بین این پایه و زمین یک خازن 100nf قرار می‌گیرد.	تغذیه موتور	۶	۴
- این پایه تغذیه خود ای سی بوده و در حدود ۷ ولت می‌باشد. - برای کاهش نویز بین این پایه و زمین یک خازن 100nf قرار می‌گیرد.	تغذیه ای سی	۱۲	۹
	زمین ای سی	۱،۱۰،۱۱،۲۰	۸
- با دادن ولتاژی بین ۲.۳ تا ۷ ولت به هر کدام از این پایه‌ها می‌توان موتورهای A و B را فعال کرد. - با زمین کردن این پایه‌ها می‌توان	فعال ساز خروجی‌های A و B	۸، ۱۴	۶ ، ۱۱

موتورهای A و B را غیر فعال کرد.			
- این دو پایه برای کنترل موتور A از میکروکنترلر گرفته می‌شوند.	ورودی ۱ و ۲	۹ ، ۷	۷ ، ۵
- این دو پایه برای راه اندازی موتور A ، مستقیما به و پایه موتور وصل می‌شوند.	خروجی ۱ و ۲	۵ ، ۴	۳ ، ۲
- این دو پایه برای کنترل موتور B از میکروکنترلر گرفته می‌شوند.	ورودی ۳ و ۴	۱۵ ، ۱۳	۱۲ ، ۱۰
- این دو پایه برای راه اندازی موتور B ، مستقیما به و پایه موتور وصل می‌شوند.	خروجی ۳ و ۴	۱۷ ، ۱۶	۱۴ ، ۱۳
- از این دو پایه به عنوان سنسور جریان استفاده می‌شود. - توسط یک پتانسیومتر به زمین وصل می شوندکه می‌تواند جریان بار را کنترل کند. - در صورتی که کنترل جریان مهم نباشد توسط سیم به زمین وصل می‌شود.	حس کننده A و B	۱۹ ، ۲	۱۵ ، ۱
	بدون اتصال	۱۸ ، ۳	

مدارات عملی :

۱- برنامه‌ای بنویسید که توسط آن یک موتور پله‌ای را به روش پله کامل راه اندازی کرد؟

```
$regfile = "m32def.dat"
$crystal = ۸000000

'----------------------
Config Portb = Output
Config Portd = Output
'----------------------
Do
  Portb = &B10000000
  Waitms 500
  Portb = &B01000000
```


فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

```
  Waitms 500
  Portb = &B00100000
  Waitms 500
  Portb = &B00010000
  Waitms 500
Loop
End
```

۲- برنامه‌ای بنویسید که توسط آن بتوان یک موتور پله‌ای را به روش نیم پله راه اندازی کرد؟

```
$regfile = "m32def.dat"
$crystal = ۸000000
'-----------------------
Config Portb = Output
Config Portd = Output
'-----------------------
Do
  Portb = &B00010000
  Waitms 250
  Portb = &B00110000
  Waitms 250
  Portb = &B00100000
  Waitms 250
  Portb = &B01100000
  Waitms 250
  Portb = &B01000000
  Waitms 250
  Portb = &B11000000
  Waitms 250
  Portb = &B10000000
  Waitms 250
  Portb = &B10010000
  Waitms 250
Loop
End
```

نکته مهم: زمان تاخیری که بین پالس‌ها است، نباید خیلی کم باشد چون در غیر این صورت موتور نمی‌تواند پالس‌ها را دنبال کند و فقط در جای خود می‌لرزد.

ارتباط سریال SPI

ارتباط سریال SPI یک پروتکل ارتباطی سنکرون با سرعت بالا است که می‌تواند برای ارتباط میکروکنترولرهای AVR با یکدیگر و یا ارتباط میکروکنترولر AVR با وسیله های دیگری که قابلیت این نوع ارتباط را دارا هستند، به کار برده شود. رجیسترهای مربوط به این نوع ارتباط در تمام AVRها یکسان است.

خصوصیات ارتباط سریال SPI

- ارسال و دریافت داده همزمان
- استفاده از چهار سیم برای انتقال اطلاعات
- بیت‌های قابل برنامه ریزی برای تنظیم سرعت انتقال دیتا
- دارای پرچم وقفه اتمام ارسال
- ارتباط به صورت‌های MASTER / SLAVE
- بیدار شدن از حالت بیکاری (IDLE)

شرح عملکرد ارتباط سریال SPI

در ارتباط سریال SPI از چهار پایه SCK، MISO، MOSI، SS استفاده می‌شود. پایه SCK در مد Master به عنوان خروجی کلاک و در مد Slave به عنوان ورودی کلاک مورد استفاده قرار می‌گیرد. با نوشتن رجیستر داده SPI در Master، پردازنده شروع به تولید کلاک SPI کرده و داده‌ها از پایه MOSI خارج شده و به پایه MOSI در Slave وارد می‌شوند. بعد از انتقال کامل داده توسط Master، کلاک SPI قطع شده و پرچم وقفه پایان ارسال داده (SPIF) یک می‌شود و برنامه وقفه اجرا می‌شود. دو شیفت رجیستر ۸ بیتی در Masterو Slave را می‌توان به عنوان یک شیفت رجیستر ۱۶ بیتی در نظر گرفت. به عبارت دیگر زمانی که داده‌ای از Master به Slave ارسال می‌شود، می‌توان در همان حال در جهت مخالف، داده‌ای از Slave به Master ارسال کرد. بنابراین در طول ۸ کلاک SPI، داده‌های Master و Slave با هم عوض می‌شوند.

جهت پایه SS:

جهت پایه‌ی SS (خروجی یا ورودی بودن) درمد Master توسط کاربر تعیین می‌شود:

- اگر پایه SS به صورت ورودی به خروجی تعیین شود از آن به عنوان خروجی عادی استفاده می‌شود به این صورت که هیچ تاثیری در ارتباط SPI ندارد.

- اگر پایه SS به صورت ورودی تعیین شود بایستی High یا ۱ باشد تا عملیات Master با اطمینان انجام شود.

جهت پایه SS (خروجی یا ورودی بودن) درمد Slave توسط کاربر تعیین نمی‌شود:

در این حالت پایه SSهمیشه به عنوان ورودی می‌باشد:

- زمانیکه Lowباشد SPIفعال می‌شود (در این حالت پایه MISO خروجی و بقیه پایه‌ها ورودی هستند)
- زمانیکه High باشد SPI بیکار بوده و تمام پایه‌ها به صورت ورودی می‌باشند.

پیکربندی SPI

پیکربندی SPI به دو صورت سخت افزاری ونرم افزاری امکان پذیر است.

پیکره بندی سخت افزاری

دراین نوع پیکره بندی، پایه های پیش فرض یعنی پایه های SS , SCK , MISO , MOSI بکاربرده می‌شوند وقابل تغییر نیستند.

پیکربندی SPI ..... CONFIG SPI

شکل کلی دستور:

CONFIG SPI = HARD , INTERRUPT = ON|OFF , DATA ORDER = LSB|MSB , MASTER = YES|NO ,_
POLARITY = HIGH|LOW , PHASE = 0|1 , CLOCKRATE = 4|16|64|128 , NOSS = 1|0

Hard: تعیین نوع پیکره بندی سخت افزاری.

INTERRUPT = ON|OFF: استفاده یا عدم استفاده از وقفه در SPI. (ON: استفاده از وقفه که دراین حالت میکروکنترلر در زمان انتقال داده کار می‌کند)

DATA ORDER = LSB|MSB: درصورت انتخاب LSB، ابتدا LSB داده و سپس MSB آن ارسال می‌شود و بالعکس.

MASTER = YES|NO: تعیین میکروکنترلر جاری به عنوان Master یا Slave.

POLARITY = HIGH|LOW: Low یا High کردن پایه کلاک (SCK) در حالت بیکاری میکروکنترلر.

PHASE = 0|1: صفر در نظر گرفته می‌شود.

CLOCKRATE = 4|16|64|128: مشخص کننده فرکانس کلاک ارتباط SPI است.

NOSS = 1| 0: زمانی که در حالت Master نخواهیم سیگنال SS ایجاد شود، این پارامتر را 1 می‌کنیم. در این حالت بایستی در برنامه پایه Slave مورد نظر را صفر کنیم.

پیکربندی سخت افزاری به شکل ساده تری نیز انجام می‌گیرد:

```
Config SPI = Hard, INTERRUPT = OFF , DATA ORDER = MSB , MASTER = YES , POLARITY = HIGH ,_ PHASE = 0 , CLOCKRATE = 4
```

پیکره بندی نرم افزاری

دراین نوع پیکره بندی می‌توان هرکدام از پایه‌های میکروکنترلر را به جای پایه‌های پیش فرض، بکار برد.

```
CONFIG SPI = SOFT , DIN = PIN , DOUT = PIN , SS = PIN|NONE , CLOCK = PIN
```

SOFT: تعیین نوع پیکره بندی نرم افزاری.

DIN: نشانگر پایه MISO بوده وPIN نام پایه دلخواهی از میکروکنترلر است.

DOUT: نشانگر پایه MOSI بوده و PIN نام پایه دلخواهی از میکروکنترلر است.

SS: نشانگر پایه SS بوده و PIN نام پایه دلخواهی از میکروکنترلر است. (درصورت تمایل به نداشتن سیگنال SS از حالت NONE استفاده می‌شود)

CLOCK: نشانگر پایه SCK بوده وPIN نام پایه دلخواهی از میکروکنترلر است.

شروع کار با درگاه SPI

مقدار دهی اولیه پایه های SPI ..... Spiinit

بعد از پیکره بندی ارتباط سریال، توسط این دستور پایه‌های ارتباط سریال برای ارتباط SPI فعال شده ودیگر نمی‌توان از آن‌ها به عنوان I/O استفاده کرد.

ارسال داده به درگاه SPI ..... SPIOUT

شکل کلی دستور: Spiout Var , Byte

توسط این دستور به تعداد Byte، داده Varبه درگاه SPIاطلاعات ارسال می‌شود.

تعداد Byte به نوع متغیر Var بستگی دارد. مثلا اگر Var از جنس Word باشد آنگاه Byte=2 می‌شود.

دریافت داده از درگاه ..... SPIIN SPI

شکل کلی دستور: Spiin Var , Byte

توسط این دستور به تعداد Byte، داده Varاز درگاه SPIاطلاعات دریافت می‌شود.

تعداد Byte به نوع متغیر Var بستگی دارد. مثلا اگر Var از جنس Word باشد، آنگاه Byte=2 می‌شود.

ارسال و دریافت همزمان داده ..... SPIMOVE

شکل کلی دستور: Var = Spimove (Byte)

توسط این دستور متغیر Byte به درگاه SPI ارسال شده و همزمان داده دریافت شده از درگاه SPI در متغیر Var قرار می‌گیرد.

نکته مهم: در صورتی که می خواهید دو میکروکنترلر را با استفاده از این پروتکل به هم متصل کنید دقت کنید که فرکانس هر دو میکرو باید عینا برابر باشد در غیر این صورت اطلاعات به درستی تبادل نخواهد شد.

راه اندازی حافظه های جانبی MMC

خیلی وقت ها پیش میاد که حافظه میکروکنترلر جهت ذخیره داده ها کم میاره. خب احتمالا همگی برای اولین گزینه بریم سراغ حافظه های eeprom خارجی اما متوجه میشیم که اونا هم انچنان چنگی به دل نمی زنن.

گذشته از این ها خیلی وقتا شاید بخوایم یه آلبوم دیجیتال یا یه waveplayer بسازیم، اون موقع اطلاعات رو کجا ذخیره کنیم؟

گزینه مناسب برمی گیرده به حافظه های MMC که توانایی حجم زیادی از داده ها رو در فضایی خیلی کم دارن.

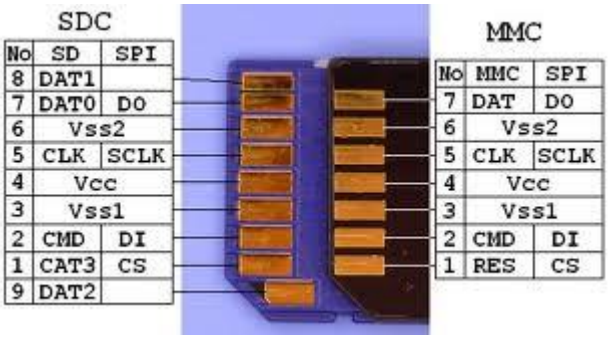
شکل ظاهری این حافظه های به صورت زیر می باشد:

ارتباط با این حافظه ها از ۲ طریق ممکن شده:

۱. پروتکل مخصوص حافظه یا mmc
۲. پروتکل spi

پروتکل انتخابی ما با توجه به سخت افزار موجود در میکروکنترلر spi بوده و می خواهیم با استفاده از این پروتکل با MMC ارتباط برقرار کرده و اطلاعاتی را از آن خوانده یا در آن بنویسیم.

پایه های حافظه MMC به صورت تصویر زیر می باشد:



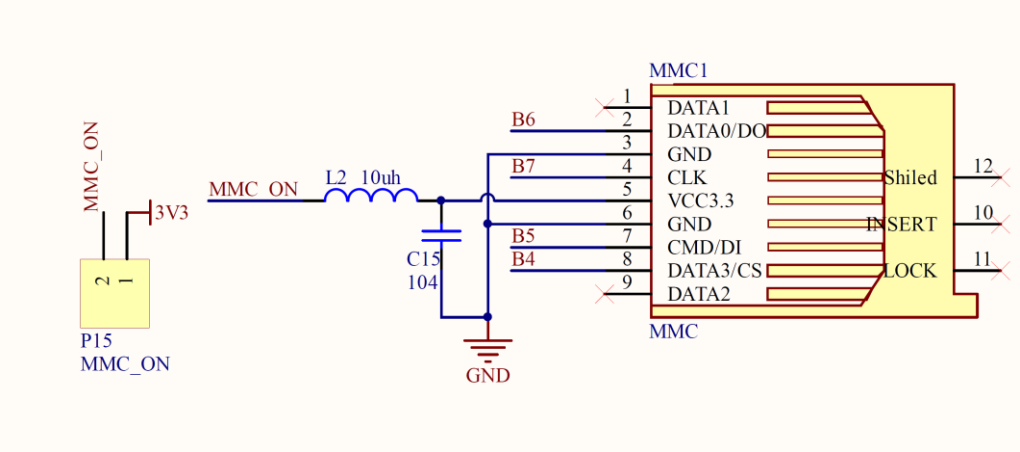
دو روش جهت ارتباط و خواندن و نوشتن اطلاعات در حافظه وجود دارد:

۱- خواندن/نوشتن به صورت سکتور سکتور

۲- خواندن/نوشتن با استفاده قوانین حاکم بر فرمت fat

در بسکام جهت راه اندازی این حافظه ها کتابخانه ای در نظر گرفته شده که در این قسمت می خواهیم با استفاده از کتابخانه موجود و توابع آن کار بر روی MMC SD را آغاز کنیم.

قبل از هر چیز بهتر است که شماتیک و نحوه اتصال کارت حافظه را به میکروکنترلر نمایش دهیم:



در بسکام کتابخانه avr dos این امکان را به کاربر می دهد تا با استفاده از دستورات سطح بالا با کارت حافظه ارتباط برقرار کند.

در این آموزش سعی می کنیم به بررسی یک سری از دستورات پایه بپردازیم.

قبل از چیز دو کتابخانه CONFIG_AVR-DOS.Bas و config_mmc.bas که اولی مربوط به توابع کار با mmc و دومی تنظیمات اولیه و یکسری تنظیمات ارتباطی و خطاها را شامل می شود.

این دو کتابخانه را از مسیر نصب بسکام پیدا کرده و در پوشه پروژه خود قرار دهید.

در کتابخانه config_mmc.bas می توانید به اختیار خود از spi سخت افزاری یا نرم افزاری و پین های مورد نظر خود استفاده کنید.

تشریح توابع پایه کار با MMC:

فرم دستور	کاربرد	برگشتی تابع(در صورت وجود)
Ver()	برگرداندن ورژن کتابخانه	---------
Drivecheck()	چک کردن وجود mmc	در صورت درستی صفر برگشت داده می شود.
Driveinit()	Init کردن اولیه کارت	در صورت درستی صفر برگشت داده می شود.
Initfilesystem(1)	چک کردن فرمت کارت حافظه	در صورت درستی صفر برگشت داده می شود.
Disksize()	نمایش ظرفیت mmc بر حسب بایت	----------
Diskfree()	نمایش فضای آزاد بر حسب بایت	----------
File data&time	تاریخ و زمان تشکیل فایل ها	----------
Filelen()	اندازه فایل بر حسب بایت	----------
dir	نمایش فایل های موجود در کارت حافظه	----------
Open filename	باز کردن فایل	در صورت درستی صفر برگشت داده می شود.
close	بستن فایل باز شده	----------
Line input	خواندن یک خط از اطلاعات فایل(محتویات فایل)	----------
Eof	مشخص کردن انتهای فایل	در صورت رسیدن به انتهای فایل صفر را برمی گرداند.

----------	حذف فایل	Kill filename

جهت آشنایی هر چه بیشتر به مثال زیر دقت کنید.

برنامه ای بنویسید که توسط آن حجم کارت حافظه و فضای آزاد آن نمایش داده شود سپس یک فایل را ایجاد کرده و اطلاعاتی را در آن بنویسد و بعد از بستن فایل مجددا آن را باز کرده و اطلاعات ان را بخواند؟

تنظیمات روی برد:

تغذیه را در حالت ۳.۳ ولت قرار داده و جامپرهای مربوط به نمایشگر کاراکتری را نیز قرار دهید. درنهایت جهت تغذیه و اتصال کارت حافظه جامپر MMC را وصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
$swstack = 32
$hwstack = 64
$framesize = 64
$include "CONFIG_AVR-DOS.Bas"
$include "config_mmc.bas"
'----------------------------
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
'----------------------------
Enable Interrupts
```

```
Dim S As String * 10
'----------------------------
Cls
Lcd "avr_dos Ver" ; Ver()
Wait 1
While Drivecheck() <> 0
Wend
Cls
Lcd "MMC Inserted"
Wait 1
While Driveinit() <> 0
Wend
Cls
Lcd "MMC Init"
Wait 1
While Initfilesystem(1) <> 0
Wend
Cls
Lcd "MMC Format OK!"
Wait 1
Cls
Lcd "space" ; Disksize() ; " b"
Locate 2 , 1
Lcd "free" ; Diskfree() ; " b"
```

```
Wait 1
Cls
Lcd "create file"
Wait 1
Open "ECA.txt" For Output As #1
Print #1 , "www.ECA.ir"
Close #1
Cls
Lcd "file created & close"
Wait 1
Open "ECA.txt" For Input As #1
Do
Line Input #1 , S
Cls
Lcd S
Wait 1
Loop Until Eof(#1) <> 0
Do
Loop
End
```


فروشگاه تخصصی برق و الکترونیک www.eShop.ECA.ir ایمیل: eShop.ECA@Gmail.com
شماره تماس: ۰۴۱۱۵۵۳۳۷۳۲ فکس: ۰۴۱۱۵۵۳۹۷۶۹

ارتباط سریال I2C

پروتکل ارتباطی I2C پروتکل ساخته شده توسط شرکت فیلیپس می‌باشد که توسط آن و از طریق دو سیم می‌توان میان میکروکنترلر و هر وسیله‌ای که دارای چنین قابلیتی باشد ارتباط برقرار کرد.

ویژگی‌های پروتکل I2C

۱. در این ارتباط از دو سیم برای انتقال دیتا استفاده می‌شود.
۲. در این ارتباط می‌توان تعداد نامحدود وسیله جانبی با آدرس سخت افزاری متفاوت را به هم متصل کرد.
۳. بالاترین فرکانس کلاک سیستم ۴۰۰ کیلوهرتز است.
۴. کلاک ارتباط I2C به شدت به کلاک سیستم (فرکانس کریستال اصلی) وابسته است.
۵. حداکثر طول کابل ارتباطی با سیم شیلددار و تقویت کننده ترانزیستوری حداکثر ۸۰ سانتی متر است.

در این ارتباط از دو پایه SDA و SCl که یکی به عنوان خط دیتا و دیگری به عنوان کلاک مورد استفاده قرار می‌گیرد، استفاده می‌شود. همچنین در مسیر ارتباط باید توسط مقاومت دو خط را Pull Up کنید.

دستورات مربوط به پیکربندی و کار با ارتباط I2C

پیکربندی پایه دیتا ..... config SDA

شکل کلی دستور: Config SCL= PinX.Y

توسط این دستور پایه SDA میکروکنترلر (در Atmega 32 پین c.1) به عنوان پایه دیتا انتخاب می‌شود.

پیکربندی پایه کلاک ..... config SCL

شکل کلی دستور: Config SDA= PinX.Y

توسط این دستور پایه SCL میکروکنترلر (در Atmega 32 پین c.0) به عنوان پایه کلاک انتخاب می‌شود.

تعیین فرکانس کلاک I2cDelay

شکل کلی دستور : Config I2CDELAY = X

در این دستور مقدار X که می‌تواند بین ۱ تا ۴۰ باشد در 10K ضرب شده و به عنوان فرکانس کلاک ارتباط انتخاب می‌شود.

شروع به کار پروتکل ..... Start

توسط این دستور که شکل کلی آن به صورت زیر است پروتکل آماده دریافت و یا ارسال داده می‌شود.

I2cStart

همچنین توسط دستور زیر می‌توان ارتباط پروتکل را قطع کرد:

I2cStop

ارسال داده به پروتکل I2cSEND

شکل کلی دستور: I2CSEND Slave, var,byte

Slave: آدرس گیرنده اطلاعات است که می‌تواند یک عدد ثابت یا مقدار یک متغیر باشد.

VAR: عدد یا مقدار متغیری که به پروتکل ارسال می‌شود.

BYTE: تعداد بایت ارسالی مشخص می‌شود. (استفاده از این قسمت اختیاری است)

این دستور یک فرم خلاصه شده به شکل زیر را هم دارا می‌باشد:

I2CWBYE VAR

دریافت داده از پروتکل I2cReceive

شکل کلی دستور: I2CRECEIVE Slave, var,b2W,b2R

Slave: آدرس فرستنده اطلاعات است که می‌تواند یک عدد ثابت یا مقدار یک متغیر باشد.

VAR: عدد یا مقدار متغیری که از پروتکل دریافت می‌شود.

B2W: تعداد بایت نوشته شده.

B2R: تعداد بایت دریافتی.

این دستور یک فرم خلاصه شده به شکل زیر را هم دارا می‌باشد:

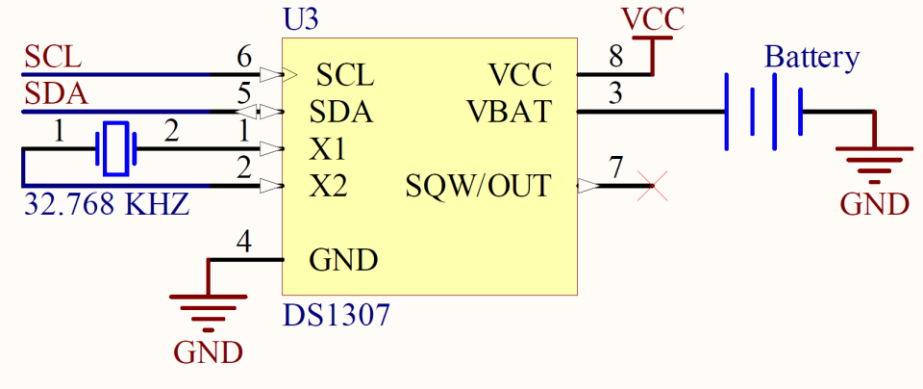
I2CRBYTE var, ack/nack

ACk و NACK: زمانی که بخواهیم بیشتر از یک بایت را از باس بخوانیم از ack و زمانی که بخواهیم آخرین بایت را از باس بخوانیم از nack استفاده می‌کنیم.

مدارات عملی :

۱- با استفاده از تراشه DS1307 موجود بر روی برد یک ساعت طراحی کنید؟

شماتیک مدار:



تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه را در حالت ۵ ولت قرار داده و جامپرهای SDA و SCL را وصل کنید. همچنین جهت نمایش اطلاعات جامپرهای مربوط به نمایشگر کاراکتری را متصل کنید.

```
$regfile = "m32def.dat"
$crystal = 8000000
$lib "ds1307clock.lib"
'-------------------------
Config Lcd = 16 * 2
```

```
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
Enable Interrupts
Config Sda = Portc.1
Config Scl = Portc.0
'---------------------------------------
Const Ds1307w = &HD0
Const Ds1307r = &HD1
'---------------------------------------
Dim Second As Byte , Minute As Byte , Hour As Byte
'---------------------------------------
Cls
Main:
    Do
        Gosub Ds1307
        Cls
        Lcd Hour ; ":" ; Minute ; ":" ; Second ; "  "
        Waitms 500
    Loop
Ds1307:
      I2cstart
      I2cwbyte Ds1307w
      I2cwbyte 0
      I2cstart
      I2cwbyte Ds1307r
      I2crbyte Second , Ack
      I2crbyte Minute , Ack
      I2crbyte Hour , Nack
      I2cstop
      Second = Makedec(second) : Minute = Makedec(minute) : Hour = Makedec(hour)
      If Second > 59 Then Second = 0
      If Minute > 59 Then Minute = 0
      If Hour > 23 Then
            Hour = 0
            Gosub Seco
      End If
```

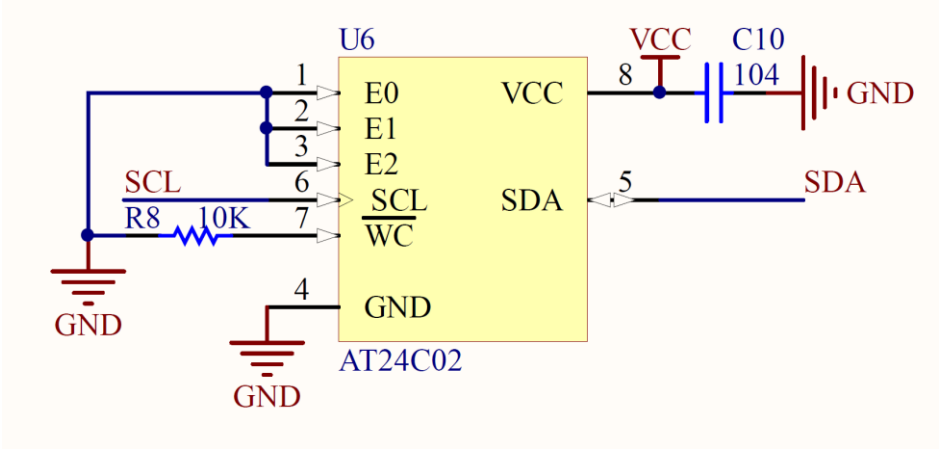
```
Return

Seco:
    Incr Second
    If Second > 59 Then Second = 0
    Second = Makebcd(second)
    I2cstart
    I2cwbyte Ds1307w
    I2cwbyte 0
    I2cwbyte Second
    I2cstop
Return
Mine:
    Incr Minute
    If Minute > 59 Then Minute = 0
    Minute = Makebcd(minute)
    I2cstart
    I2cwbyte Ds1307w
    I2cwbyte 1
    I2cwbyte Minute
    I2cstop
Return
Hours:
    Incr Hour
    If Hour > 23 Then Hour = 0
    Hour = Makebcd(hour)
    I2cstart
    I2cwbyte Ds1307w
    I2cwbyte 2
    I2cwbyte Hour
    I2cstop

Return
```

۲- برنامه ای بنویسید که توسط آن بتوان اطلاعاتی را بر روی حافظه EEprom خارجی موجود بر روی برد نوشت و یا خواند؟

شماتیک مدار :




```
$regfile = "m32def.dat"
$crystal = 8000000
'---------------------------
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
Enable Interrupts
Config Sda = Portc.1
Config Scl = Portc.0
Config I2cdelay = 1
'---------------------------
Dim Address As Word , A As Word
Dim A1 As Byte , A2 As Byte , Dat As Byte
'---------------------------
Address = &HA0
Cls
Lcd "write number 50"
wait 1
I2cstart
I2cwbyte Address
I2cwbyte &H10
I2cwbyte 50
I2cstop
Waitms 10
Cls
Lcd "Read number  "
wait 1
```

```
I2cstart
I2cwbyte Address
I2cwbyte &H10
I2cstart
A = Address Or 1
I2cwbyte A
I2crbyte Dat , Nack
I2cstop
Lcd Dat
Do
Loop
End
```

پیکربندی پروتکل ارتباطی UART

این پروتکل یک ارتباط سریال قابل برنامه ریزی، در دو حالت نرم افزاری و سخت افزاری می‌باشد که بیشتر برای ارتباط میکروکنترلرها با کامپیوتر طراحی شده است. نکته‌ای که حائز اهمیت است سطح ولتاژ در منطق TTL می‌باشد که بین ۰ تا ۵ ولت قرار دارد ولی در پروتکل RS-232 بین ۱۵- تا ۱۵+ قرار دارد که این تبدیل سطح ولتاژ توسط تراشه‌هایی مانند MAX232 و یا MAX235 انجام می‌گیرد.

پیکربندی UART سخت افزاری

در این حالت از دو پایه RXD و TXD استفاده می‌شود. قبل از پیکربندی این ارتباط باید سرعت و نرخ ارسال دیتا بین سیستم‌ها برابر باشد تا دیتاهای صحیح بین آن‌ها تبادل شود. این نرخ باود (BAUD) نام دارد که تعیین کننده سرعت بین دو سیستم است.

تعیین نرخ باود ..... $BAUD=X

این نرخ باید در ابتدای برنامه تعیین شود که X مقادیر استانداردی مانند ۱۱۵۲۰۰، ۵۷۶۰۰، ...، ۱۲۰۰ می‌باشد.

پیکربندی UART ..... Config UART

Config SERIALOUT=BUFFERED,SIZE= X

توسط این دستور پروتکل UART برای ارسال دیتا به صورت سخت افزاری پیکربندی شده است. در قسمت SIZE به میزان X بایت از حافظه SRAM اشغال می‌شود تا داده‌های ارسالی از طریق این بافر به خروجی ارسال شود.

ارسال دیتا به پورت..... Print

توسط این دستور می‌توان داده مورد نظر را اعم از رشته، عدد و یا هردو را به درگاه ارسال کرد.

Print Var || Print " …. " || Print " ….. " ; Var

ارسال دیتا به صورت باینری Printbin .....

توسط این دستور متغیر Var به باینری تبدیل شده، سپس به پورت سریال ارسال می‌شود.

Printbin Var

پیکربندی UART برای دریافت دیتا ..... Config UART

Config SERIALIN=BUFFERED,SIZE= X

توسط این دستور پروتکل UART برای دریافت دیتا به صورت سخت افزاری پیکربندی شده است. در قسمت SIZE به میزان X بایت از حافظه SRAM اشغال می‌شود تا داده‌های دریافتی از طریق این بافر به خروجی ارسال شود.

دریافت دیتا از پورت ..... Waitkey

توسط این دستور می‌توان داده رشته یا عددی را از پورت دریافت کرد. این دستور تا زمانی که کاراکتری را دریافت نکند، برنامه را در همان خط متوقف می‌کند. شکل کلی دستور به صورت زیر می‌باشد:

Var = Waitkey()

دریافت داده از پورت ..... Inkey

این دستور برخلاف دستور قبل برنامه را متوقف نمی‌کند و با دریافت اولین کاراکتر آن را به کد اسکی تبدیل کرده و در متغیر Var می‌ریزد. اگر داده‌ای برای دریافت وجود نداشته باشد مقدار Var برابر صفر است.

شکل کلی دستور: Var = INKEY()

برای اینکه دستورات ارسال شده به کامپیوتر را نمایش دهیم باید از Terminal Emulator نرم افزار بسکام استفاده کنیم که برای دریافت اطلاعات در آن دستورات خاصی به شرح زیر وجود دارد:

دریافت داده از ترمینال ..... Input

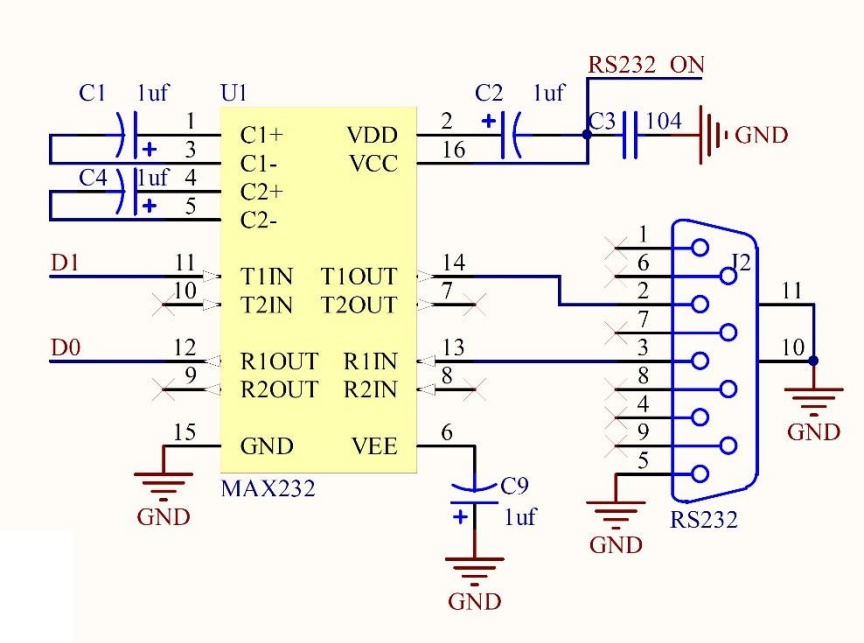
شکل کلی دستور : ,Var Input [" Data "]

توسط این دستور می‌توان در صفحه Terminal Emulator مقدار داده را دریافت کرده و در متغیر Var قرار داد.

Data: متنی که می‌تواند قبل از دریافت داده در محیط Terminal Emulator نمایش داده شود.

برنامه ای بنویسید که ابتدا میکروکنترلر پیغامی را چاپ کند سپس منتظر وارد کردن متن از طرف کاربر شود و متن وارد شده را بر روی نمایشگر نمایش دهد.

شماتیک مدار:



تنظیمات اعمال شده روی برد آموزشی:

جامپر تغذیه و RS232 را متصل کنید.همچنین جهت نمایش دیتا جامپرهای مربوط به نمایشگر کاراکتری را وصل کنید.

توجه: جهت رفع درصد خطا از کریستال خارجی ۱۱.۰۵۹۲ مگاهرتز استفاده شده است درنتیجه کریستال مربوطه را در جای مناسب قرار داده و فیوزبیت کلاک را بر روی کریستال خارجی قرار دهید.

```
$regfile = "m32def.dat"
$crystal = 11059200
$baud = 9600
'----------------------------
Config Lcd = 16 * 2
Config Lcdpin = Pin , Rs = Porta.0 , E = Porta.2 , Db4 = Porta.4 , Db5 = Porta.5 , Db6 = Porta.6 , Db7 = Porta.7
Cursor Off
'----------------------------
Dim S As String * 16
```

```
'-------------------------
Do
  Print "www.ECA.ir"
  Input S
  Cls
  Lcd S
Loop
End
```