به نام خدا

(آموزش AVR جلسه سوم)

(Bascom)

مقدمه:

در جلسه قبل کد نویسی در محیط Bascom را آغاز کرده و دو پروژه ساده (چشمکزن و کلید) را طراحی و پیاده سازی کردیم. در این جلسه قصد داریم با ورودی/ خروجی های پیشرفته تر مثل کی پد و LCD کاراکتری آشنا شویم. اصولاً یک سیستم کامپیوتری کوچک که به آن کنسول نیز گفته می شود در ساده ترین حالت از یک ورودی (مثل کی پد) و یک خروجی (مثل LCD) تشکیل شده است؛ همانند یک ماشین حساب جیبی کوچک. با این تفاوت که میکروکنترلر برنامه پذیر بوده و قابلیت تبدیل به هر دستگاهی مثل ماشین حساب، قفل رمز دیجیتالی، کنترل از راه دور و ... را دارا می باشد.

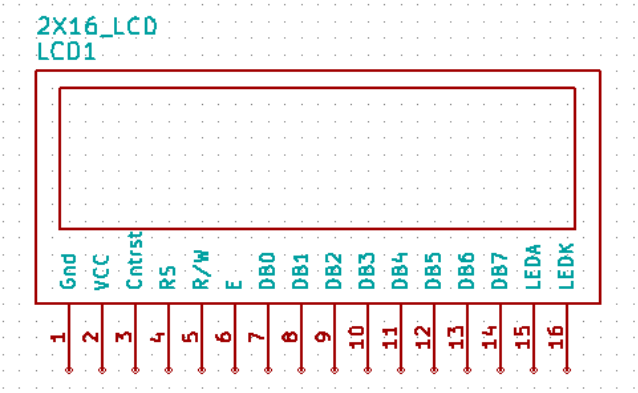
LCD

ساده ترین ابزار نمایش اطلاعات خروجی، LDC ها و Seven Segment ها می باشند. عموماً از LCD ها به دلیل قابلیت نمایش تمام حروف و اعداد انگلیسی و سادگی ارتباط دهی بیشتر از نمایشگر های هفت قسمتی استفاده می شود. در محیط بسکام توابع زیادی برای کار با LCD در نظر گرفته شده است ولی متاسفانه کد زیادی تولید کرده و به سرعت حافظه میکرو را پر می کند، از طرفی بسکام یک نرم افزار Third party بوده و نسخه Demoی آن اجازه کامپایل بیش از 4K کد را نمی دهد و نیز با نسخه دموی آن ساخت دستگاه های تجاری ممنوع است. متاسفانه امروزه در ایران افراد و سایت های بیشماری اقدام به توضیح غیر قانونی نسخه فول این نرم افزار و نرم افزار های مشابه می کنند که باعث شرمساری است.

LCD های کاراکتری انواع مختلفی دارند ولی ساده ترین و پر کاربرد ترین آنها دو سطر و ۱۶ ستون دارد که در شکل زیر تصویر یکی از آنها را مشاهده می فرمایید:

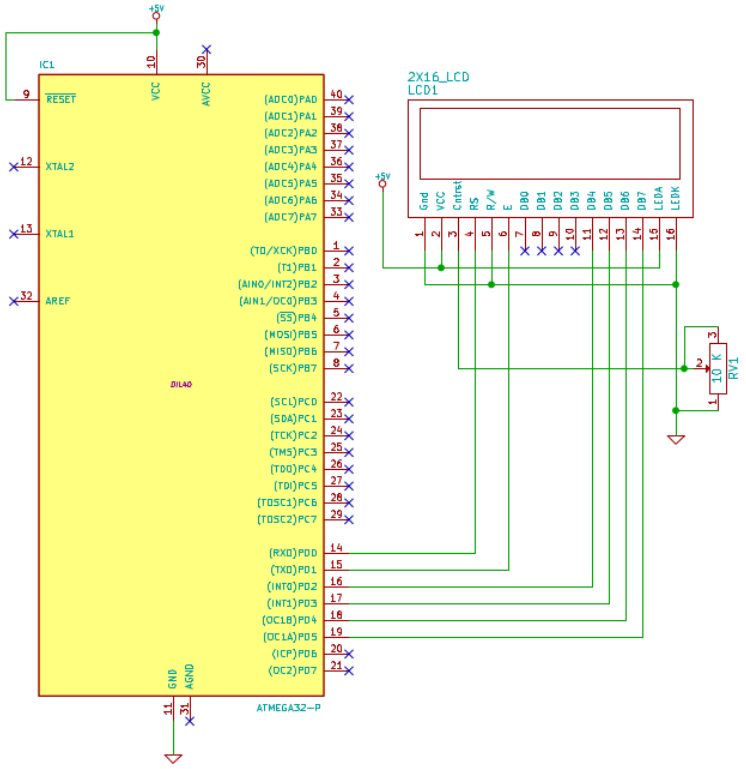
این نمایشگر های کریستال مایع ۱۶ سیم ارتباطی دارند که سیم های ۱ و ۱ به ترتیب تغذیه – و + و سیم شماره ۳ کنترل کنتراست نمایشگر و دو سیم ۱۵ و ۱۶ نیز تغذیه Back Light یا چراغ پشت نمایشگر می باشند. سیم های ۴ تا ۱۴ دیتا بوده و بایستی به میکرو متصل شوند. البته با تدابیری که در بسکام صورت گرفته شده از میان این ۱۱ سیم دیتا تعداد ۶ سیم استفاده می شود و بقیه بلا استفاده اند.

در شکل زیر پایه های این نوع LCD در محیط Kicad مشاهده می شود. KiCad یک نرم افزار رایگان برای ترسیم شماتیک مدارات الکترونیکی می باشد که بزودی آموزش آن تهیه خواهد شد.

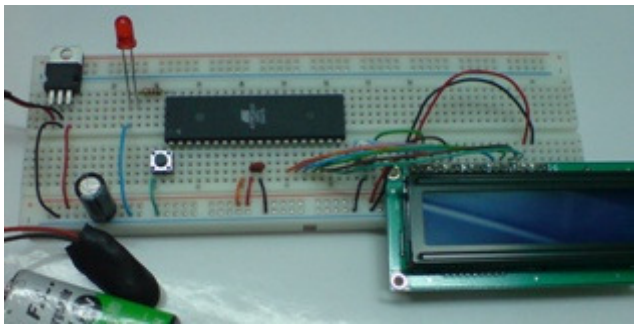


در شکل بالا پایه های ۱۵ و ۱۶ مربوط به بک لایت نمایش داده نشده اند. توجه داشته باشید که برای روشن شدن چراغ پشت صفحه پایه ۱۶ را به زمین و پایه ۱۵ را به ۵+ ولت متصل کنید. برای روشن شدن ماژول اصلی LCD نیز پایه ۱ را به زمین و پایه ۲ را به ۵+ ولت متصل کنید. پایه شماره ۳ نیز مربوط به درخشندگی کاراکترها بوده و عموماً توسط یک پتانسیومتر 10K به زمین وصل می شود. پایه شماره 5 به نام RW در صورتی که به زمین وصل شود ماژول LCD در حالت نوشتن قرار می گیرد و در حالتی که به VCC (یا همان ۵+) متصل گردد LCD در حالت خواندن قرار می گیرد که با توجه به عدم استفاده حالت دوم این پایه همیشه به زمین وصل می شود تا LCD همیشه در حالت نوشتن باشد. در بسکام پایه های ۴ و ۶ و ۱۱ تا ۱۴ که ۶ پایه می شوند به میکرو متصل شده و به دلیل اینکه بسکام یک زبان سطح بالا می باشد و کاربر را از جزئیات سخت افزار دور می کند عملکرد این پایه ها را توضیح نمی دهیم و آن را به سری مقالات آموزش Assembly اختصاص می دهیم.

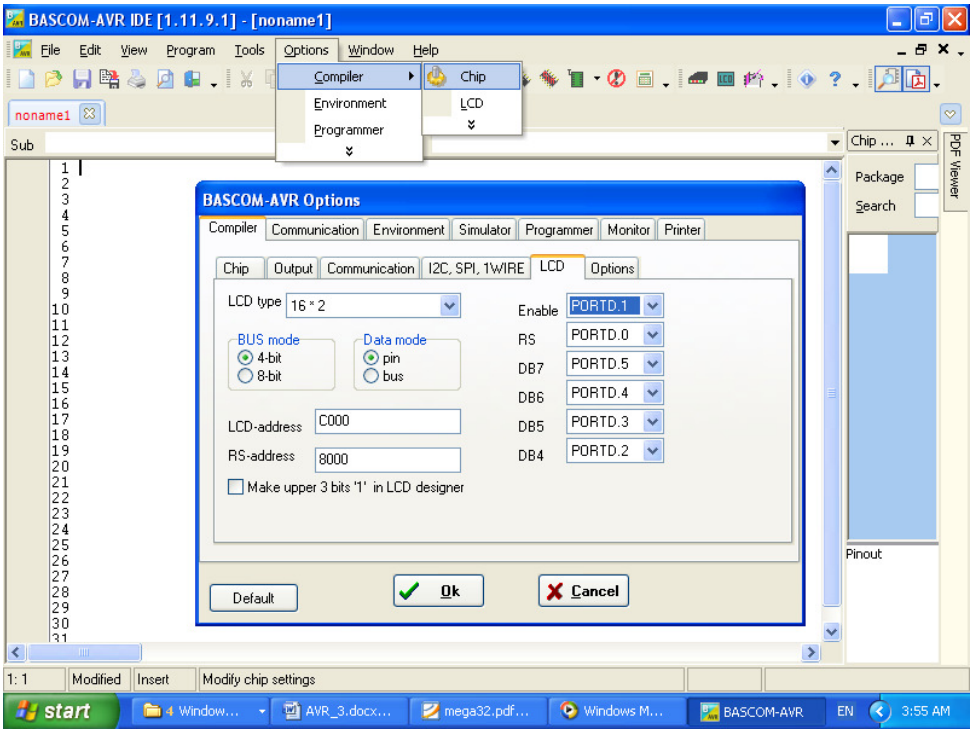
در بسکام پایه های ۴ و ۶ و ۱۱ تا ۱۴ را می توان به هر پورتی از میکرو متصل کرد. ما در اینجا پایه های LCD را به پورت D میکرو متصل می کنیم.



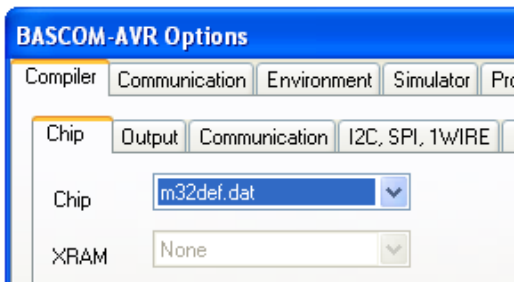
در شکل زیر طریقه لحیم پین هدر به LDC و چیدمان قطعات پروژه ارتباط با LCD بر روی Bread Board مشاهده می شود:

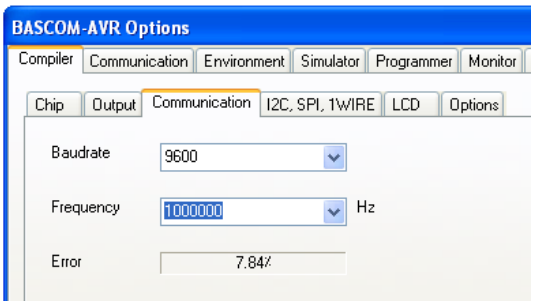


پس از بستن فیزیکی مدار نوبت به پیاده سازی برنامه آن در محیط بسکام می رسد. امروز با قسمتی از بسکام آشنا می شویم که توسط آن می توان برخی از تنظیمات را به صورت گرافیکی انجام داد و از نوشتن کد های اضافی در برنامه جلوگیری به عمل آورد. برای این منظور ابتدا File و سپس New را بزنید تا یک فایل جدید باز شود. سپس از منوی Options روی گزینه Compiler و سپس روی Chip یا LCD کلیک کنید؛ پنجره ای مطابق شکل زیر باز می شود. در این پنجره در سربرگ LCD تنظیمات ذیل را اعمال کنید.

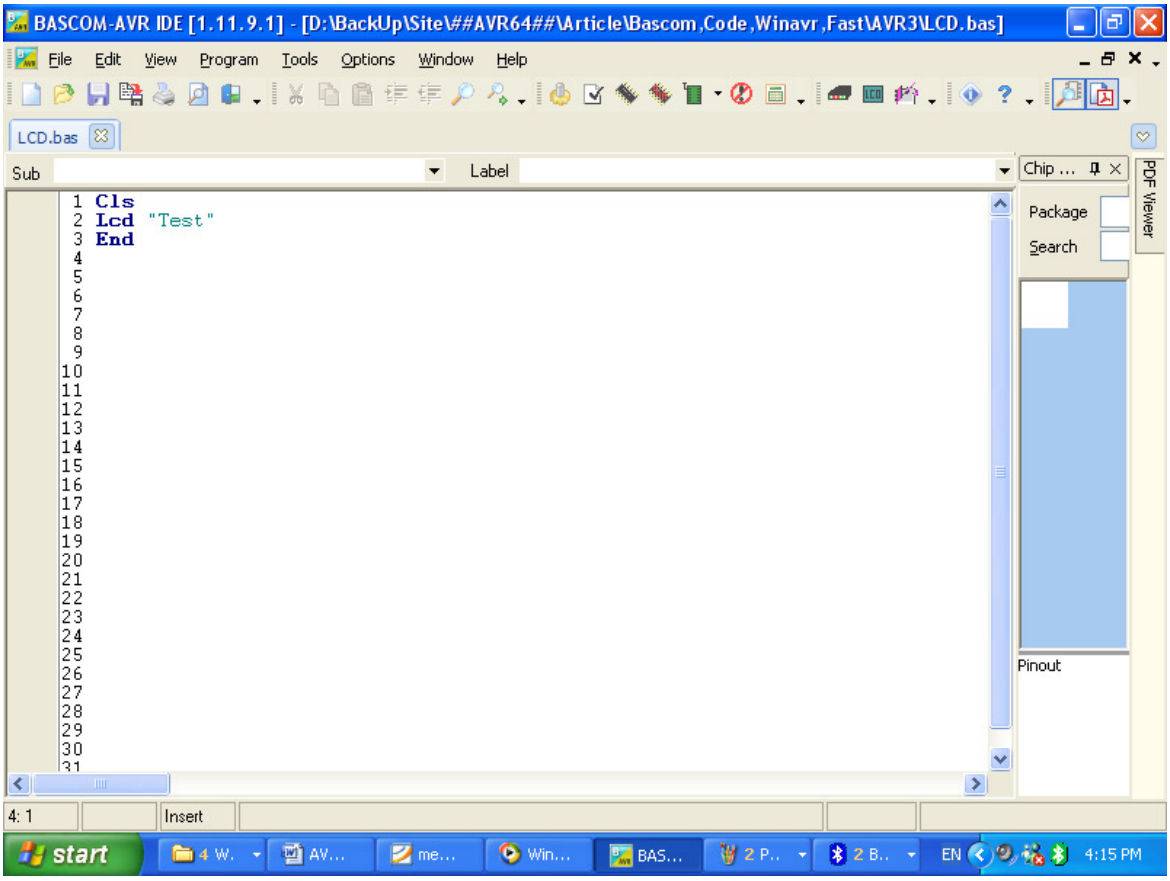


سپس وارد سربرگ Communication شده و فرکانس 1000000 را انتخاب نموده و در سربرگ Chip نیز m32def.dat را انتخاب نمایید. با انجام این دو تنظیم اخیر دیگر نیازی به تایپ دستورات $regfile="m32def.dat" و Crystal=1000000 در ابتدای برنامه نیست.





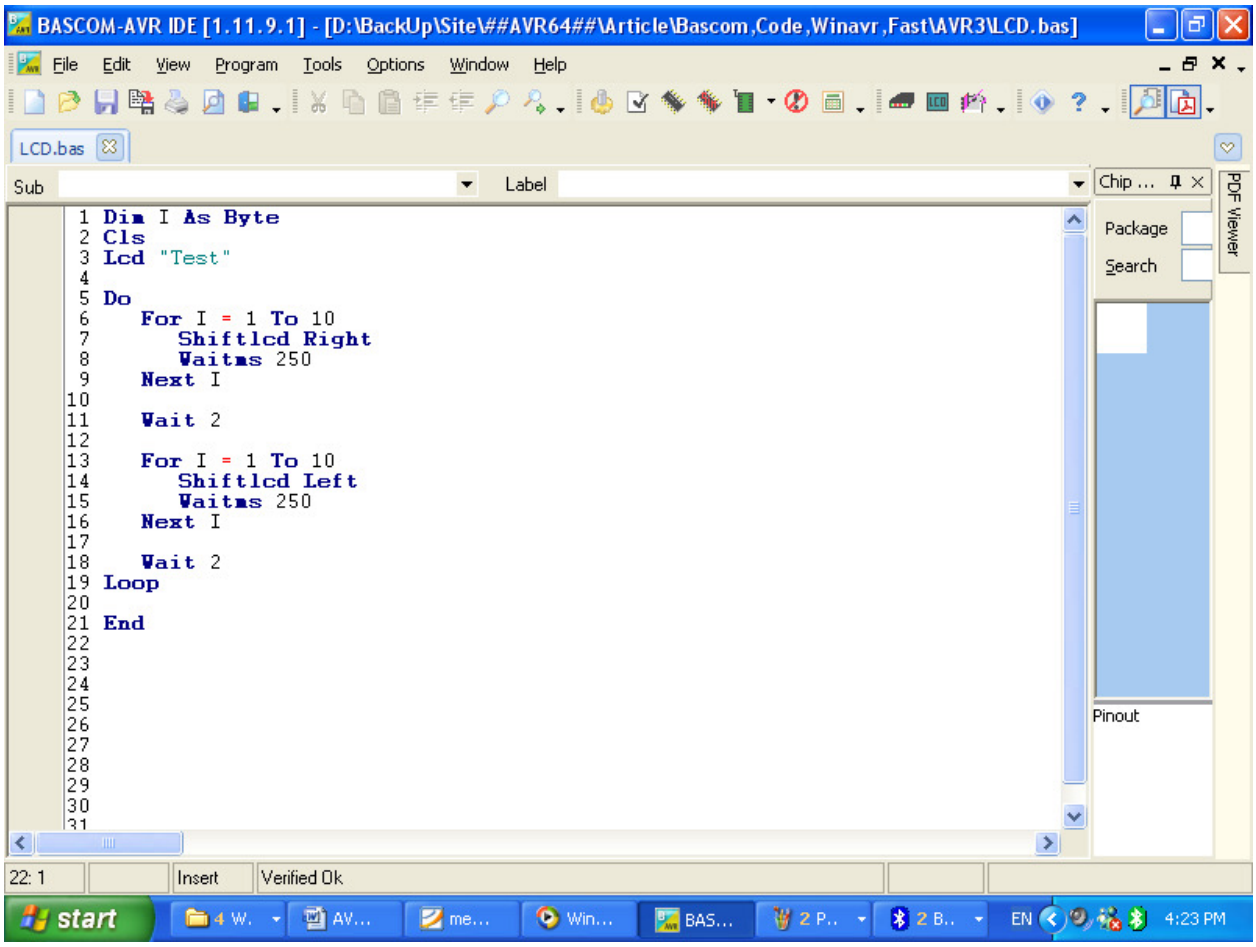
پنجره Options را OK کرده و به صفحه کد نویسی بسکام باز میگردیم. مطابق شکل زیر در خط اول دستور Cls به معنای پاک کردن صفحه نمایش و در خط بعدی دستور Lcd و در جلوی آن در داخل " رشته مورد نظر را وارد می کنیم. با زدن کلید F7 برنامه کامپایل شده و با زدن کلید F4 محیط پروگرمر باز می شود که می توانید برنامه را داخل آی سی آپلود نمایید.



پس از قرار دادن آی سی روی تخته آزمایش و اتصال تغذیه باید خروجی زیر را داشته باشید، در صورتی که خط بالای نمایشگر به صورت تعداد زیادی مستطیل روشن شده بود و یا هیچ چیزی نمایش داده نمی شد پتانیسومتر را کمی بچرخانید.

در محیط بسکام دستورات زیادی برای کار با LCD در نظر گرفته شده اند. یکی از این دستورات Cursor off می باشد که مکان نمای LCD را که عموماً به صورت _ است خاموش می کند. دستور بعدی `cursor on` است که مکان نما را روشن می کند. دستورات cursor blink و cursor noblink نیز سبب چشمکزدن و عدم چشمکزدن مکان نما می شوند.

یکی از دستورات ترکیبی که برای افکت های متن در Bascom تدارک دیده شده است جفت دستور shiftlcd right و shiftlcd left می باشند که باعث شیفت متن به چپ و راست شده و حالتی مانند تابلو روان را در نمایشکر پدیدی می آورند. البته هر کدام از این دستورات با هر بار اجرا متن موجود در LCD را فقط یک کاراکتر جابجا می کنند و برای ایجاد حالت رونده باید در یک حلقه محدود و با تاخیر مشخصی به کار روند. در مثال بعدی متن Test را به سمت راست نمایشگر و به اندازه ۱۰ کاراکتر شیفت می دهیم و پس از ۲ ثانیه دوباره به جای اول باز میگردانیم.



در برنامه بالا یک متغیر از نوع بایت تعریف کرده ایم که برای شمارنده به کار می رود. در این برنامه For اول مجموعه دستورات Shiftlcd Right و Waitms 250 را اجرا کرده و با این کار با تاخیر ۲۵۰ میلی ثانیه ای محتوای نمایشگر را ۱۰ کاراکتر به سمت راست شیفت می دهد. سپس دستور Wait 2 به مدت ۲ ثانیه تاخیر ایجاد کرده و مجدداً دستور For بعدی عبارات داخل LCD را ۱۰ کاراکتر به چپ شیفت می

دهد و ۲ ثانیه صبر می کند. نظر به اینکه کل این مجموعه دستورات داخل Do..Loop محصور شده اند این ماجرا تا ابد تکرار می شود.

در برنامه بالا اولین باری بود که متغیر تعریف کردیم. متغیرها بخشی از فضای RAM میکرو را رزرو کرده و برای موارد خاصی که نیاز به یک حافظه کوتاه مدت داریم استفاده می شوند. در تعریف متغیرها باید نکات زیادی را مد نظر داشته باشید. مورد اول دامنه تغییرات یک متغیر است. در مثال بالا با توجه به اینکه از ۱ تا ۱۰ شمارش میکنیم نوع Byte پاسخ گوی نیاز ما می باشد. یک متغیر Byte می تواند اعداد طبیعی از ۰ تا 255 را در خود جای دهد. نکته دوم فضای محدود RAM میکروها می باشد. (توجه داشته باشید که میکروها دو حافظه RAM و ROM دارند که عموماً مقدار ROM بیشتر از RAM می باشد. کدی را که مینویسید در فضای ROM توسط پروگرمر ذخیره می کنید و این فضا در پروژه های معمولی فقط خواندنی میباشد و با شروع کار میکرو خط به خط از آدرس اول تا جایی که برنامه داشته باشد توسط CPU داخل میکرو خوانده و اجرا می شود. این فضا در میکروی Mega32 به میزان ۳۲ کیلوبایت است. ROM که غالباً آن را Flash نیز می نامند را می توانید همانند هارد کامپیوتر فرض کنید که برنامه ها از آن اجرا می شوند. RAM نیز دقیقاً عملکردی شبیه RAM کامپیوتر دارد که برنامه ها موقع اجرا روی آن بار میشوند. البته این مورد اخیر در رابطه با میکروها کاملاً صادق نمیباشد و برنامه به طور مستقیم از روی همان ROM اجرا می شود، ولی بخش هایی از برنامه که نیاز به ذخیره موقت اطلاعات دارند از فضای قابل خواندن و نوشتن RAM بهره می گیرند. مثلاً در مثال بالا متغیر i که مقدار شمارنده را در حافظه خود نگه می دارد یک بایت از فضای رم را به خود اختصاص می دهد و با دستور Dim این فضا برای i رزرو می شود. فضای رم در Mega32 برابر با ۲ کیلوبایت است و باید در تعریف متغیر ها دقت کنید تا این فضا سر ریز نکند.

فضای دیگری در میکروها وجود دارد که از RAM کمتر بوده ولی اطلاعات آن با قطع برق نابود نمیشود که به آن EEPROM می گویند.

ROM > برنامه میکرو – با قطع برق از بین نمی رود - سرعت دسترسی بالا – خواندنی نوشتی – محدودیت 10000 بار نوشتن.

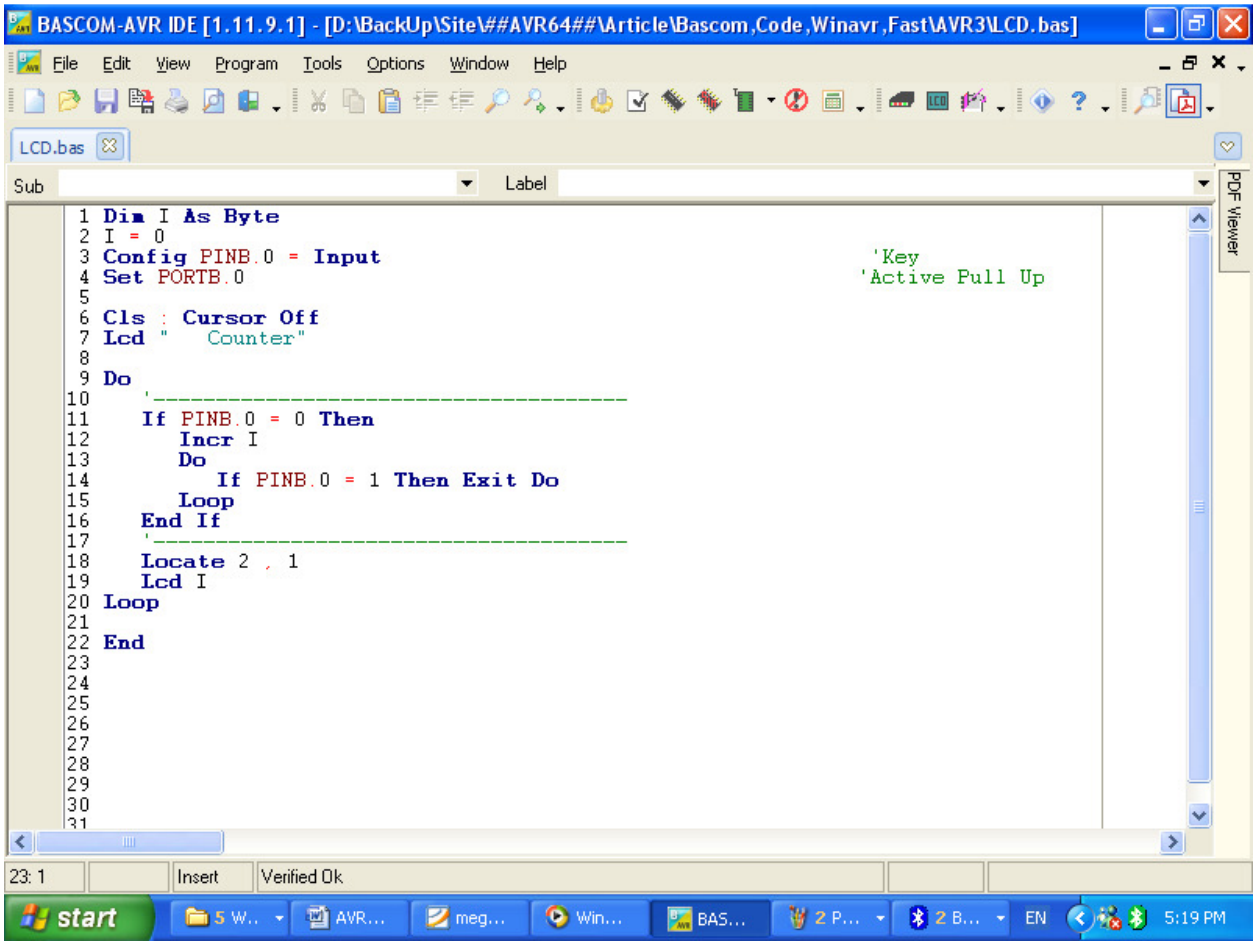
RAM > متغیرهای موقتی – با قطع برق پاک می شود - سرعت دسترسی بالا – خواندنی نوشتنی – بدون محدودیت خواندن/ نوشتن.

EEPROM > متغیرهای دائمی – با قطع برق از بین نمی رود - سرعت دسترسی پایین – خواندنی نوشتنی – محدودیت 100000 بار نوشتن و پاک کردن.

در مورد آدرس های حافظه و فضای Stack و ... مطالب زیادی وجود دارند که از بیان آنها در اینجا خودداری کرده و توضیحات بیشتر را به سری مقالات آموزش اسمبلی واگذار می نماییم. اصولاً در پروژه های ساده که با بسکام طراحی می شوند نیازی به دانستن سطح صفر سیستم نیست و میتوان با آموزش اندکی بیسیک پروژه مورد نظر را طراحی و بهره برداری نمود. البته هیچ تضمینی وجود ندارد که چنین پروژه ای در مکان های صنعتی و شرایط مختلف به طور دقیق کار کند؛ در واقع هنگ کردن میکرو در اثر تغییر ناخواسته محتوای Ram بر اثر نویز ناگزیر است، هر چند که تدابیری مثل Watchdog برای این منظور اندیشیده شده است که لازمه آن Reset خودکار میکرو و از بین رفتن کلیه مقادیر جاری است. در هر صورت هدف از آموزش بسکام فقط و فقط آشنایی با میکروها و قابلیت های آنها می باشد و استفاده از آن به هیچ عنوان در پروژه های حرفه ای توصیه نمی شود. همانطوریکه می دانید هر سال ورژن های جدیدتر این قبیل نرم افزارها به بازار عرضه می شود و در History آنها تمام ایرادها و Bug های نسخه های قبلی آورده شده و نیز یک دوجین دستور جدید به برنامه اضافه می شود! حال آنکه اسمبلی، زبان ماشین و استاندارد میکرو بوده با داشتن حدود ۱۳۱ دستور به راحتی قابل یادگیری است و با توجه به اینکه هر دستور مستقیماً یک عمل خاص را انجام می دهد هیچ باگی نداشته و بهتر است بدانید که اکثر برنامه های سیستمی و برنامه های راه انداز موبایل ها و کامپیوترهای جیبی و PC ها که نیاز به کد کم، دقت بالا، و سرعت اجرای فوق العاده دارند با Assembly نوشته می شوند و تمام کتابخانه های بسکام، کدویژن، Win AVR، E-LAB و... نیز با اسمبلی نوشته شده اند و در واقع اسمبلی زبان مادری میکرو بوده و تمام زبان ها در نهایت بایستی برای اجرا به به زبان تبدیل شوند. پس چه بهتر که از همان ابتدا تمام کد ها را با همین زبان رسمی میکرو بنویسیم و مطمئن باشیم که در صحبت کردن با میکرو دچار هیچ خطایی نشده ایم.

کاربرد عملی متغیرها در پروژه ها:

در این بخش کمی بیشتر با متغیرها آشنا شده و نحوه تعریف و نمایش محتوای آنها را روی LCD شرح می دهیم. در پروژه اول یک شمارنده طراحی می کنیم که با هر بار فشار دادن کلید یک واحد به شمارشگر اضافه کند.



در خط اول مثال بالا متغیر i را از نوع Byte تعریف کرده و در خط دوم محتوای اولیه آن را برابر با ۰ قرار داده ایم. خطوط ۳ و ۴ را از جلسه قبل به یاد دارید که مربوط به تعریف پین متصل شده به کلید برای ورودی و فعال سازی مقاومت پول آپ داخلی می باشند. در خط ۶ نمایشگر را پاک کرده و با قرار دادن : بلافاصله در جلوی آن دستور خاموش کردن مکان نما را صادر کرده ایم. توجه داشته باشید که در بیسیک استاندارد می توانیم دستورات را به جای اینکه در خطوط جداگانه بنویسیم در جلوی یکدیگر و در یک خط تایپ کنیم که برای جدا سازی دستورات از هم از : استفاده می شود. معمولاً برای خوانایی برنامه این کار را انجام

نمی دهد. ما در اینجا برای جلوگیری از زیاد شدن خطوط برنامه برای نمایش آنها در یک صفحه این کار را انجام داده ایم. (تعداد زیادی دستور را می توان با : در یک خط نوشت که بستگی به نوع کامپایلر دارد).

خط ۹ با Do آغاز شده و بیانگر ایجاد یک حلقه نامحدود Forever می باشد. خط ۱۰ زیاد مهم نیست و چون با ' شروع شده در کامپایل برنامه نادیده در نظر گرفته می شود و برای زیبایی کد آورده شده. خطوط ۱۱ تا ۱۶ بخش اصلی این برنامه هستند که در اینجا به توضح آن می پردازیم. خط ۱۱ وضعیت کلید را اسکن می کند؛ در صورتی که فشرده نشده باشد (۱ باشد) ادامه برنامه را از Endif مر بوط به خود که در خط ۱۶ قرار دارد پی گیری می کند. (تورفتگی کد ها به خوانایی برنامه کمک می کند و سبب می شود تا با تعیین عمق کد ابتدا و انتهای مربوط به هم دستورات دوبخشی را راحت تر پیدا کنیم؛ مثلاً در مثال بالا با توجه به اینکه دو دستور If نوشته شده اند با این روش انتهای دستور اول را خیلی ساده تر از حالتی پیدا می کنیم که تمام دستورات در یک سطح نوشته شده باشند.

در خط ۱۸ دستور Locate x,y موقعیت مکان نمای LCD را در سطر دوم و ستون اول قراز می دهد و دستور بعدی مقدار i را روی نمایشگر چاپ می کند. توجه داشته باشید که ما تصمیم داریم مقدار داخل i را روی نمایشگر چاپ کنیم، بنابراین از قرار دادن i در داخل کوتیشن " " خودداری میکنیم. چه در غیر این صورت کاراکتر i بر روی صفحه چاپ می شود نه محتوای آن! در نتیجه مقدار ۰ را روی خط پایین نمایشگر داریم و این عمل (اسکن کردن کلید و نمایش مقدار i) با رسیدن به LOOP به DO پرش میکند و تکرار می شود.

فرض کنیم که این بار کاربر کلید را فشار می دهد تا یک واحد به شمارشگر اضافه کند. در این حالت دستور IF در خط ۱۱، True شده و دستور زیرین را اجرا می کند. در خط ۱۲ دستور Incr I قرار داده شده است. این دستور مخفف Increase بوده و به هر بار اجرا یک واحد به متغیر i اضافه می کند. این دستور معادل دستور i=i+1 می باشد. سپس برنامه در خطوط ۱۳ تا ۱۵ در یک حلقه Do..Loop کوچک گرفتار می شود و شرط خارج شدن از این حلقه اینست که کاربر دست خود را از روی کلید برداشته و آنرا به حال ۱ برگرداند. در این صورت شرط داخل حلقه True شده و دستور Exit Do اجرا می شود و مقدار جدید i یعنی عدد ۱ بر روی نمایشگر مشاهده می گردد. توجه داشته باشید که با توجه به اینکه در جلوی If دوم فقط یک دستور Exit do قرار دارد دیگر نیازی به End If برای آن نمی باشد. دستور IF بیسیک مانندC در صورتی که بیش از یک زیر مجموعه داشته باشد نیاز به تعیین اسکوپ دارد. در C از { و }برای تعیین دامنه کاری و در

بیسک از End if برای این منظور استفاده می شود. همانند دستور For که در بیسیک با Next خاتمه می یابد در C و Pascal با { تمام می شود.

برنامه بالا را تایپ کرده و پس از کامپیال وارد میکرو نمایید. حال کلید را ۲۵۵ بار فشار دهید، مسلماً عدد ۲۵۵ را روی نمایشگر مشاهده می کنید. با فشار مجدد کلید ناگهان متوجه می شوید که به جای عدد ۲۵۶ عدد ۰ بر روی LCD ظاهر می شود! دلیل این امر آنست که ما متغیر i را از نوع Byte تعریف کرده ایم و دامنه این متغیر از ۰ تا ۲۵۵ می باشد و با افزایش از این مقدار سرریز کرده به عبارتی Overflow می شود. پدیده Overflow در تمام متغیر ها وجود دارد ولی وظیفه ما به عنوان برنامه نویس اینست که همیشه متغیری را انتخاب کنیم که سرریز آن از محدوده کاری ما خیلی بالاتر باشد. در جدول زیر انواع متغیرهای قابل تعریف در بسکام و محدوده سرریز آنها آورده شده است: (آپدیت: متغیر DWORD نیز اخیراً اضافه شده که یک متغیر ۴ بایتی از 0 تا 4294967295 می باشد.

Bit	0 and 1
Byte	0 to 255
Integer	-32768 to +32767
Word	0 to 65535
DWORD	0 to 4294967295
Long	-2147483648 to 2147483647
Single	1.5 x 10^-45 to 3.4 x 10^38
Double	5.0 x 10^-324 to 1.7 x 10^308
String	up to 254 bytes

متغیر نوع Bit ساده ترین نوع متغیر بوده و فقط می تواند مقادیر ۰ و ۱ را در خود نگه داری کند. متغیر String نیز برای نگهداری رشته ها در نظر گرفته شده است. نوع دیگری متغیر به نام آرایه نیز وجود دارد که در اصل یک نوع ساختمان داده می باشد و در مواردی از قبیل ساخت پروژه قفل رمز و دفتر تلفن کاربرد زیادی داشته و مانند یک Database کوچک عمل می کند. (در متغیر String از { } استفاده نکنید، کاراکترهای بالا مفهوم خاصی برای رشته داشته و عدد اسکی بین آنها به کاراکتر تبدیل شده و به رشته اضافه می شود، استفاده از این کاراکترها ممکن است باعث ایجاد خطا در محتویات رشته گردد).

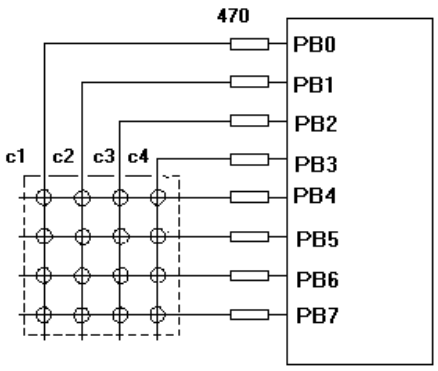
در مثال بالا مقدار i را از نوع Long تعریف کنید و شمارنده ای با قابلیت شمارش ۲ میلیارد رویداد بسازید!

تا اینجا کار با LCD و نحوه نمایش اطلاعات در خروجی را بـه خـوبی آمـوختیم. البتـه فانکشـن هـای زیادی برای کار با LCD وجود دارد ولی بهترین روش آموزش اینست که در ابتدا با شی جدیـد آشـنا شـویم و سایر امکانات اضافه را در طی پروژه های مختلف و هنگام نیاز به آنها فرا بگیریم تا در به خوبی در ذهنمان بـاقی بماند.

همانطوریکه در ابتدای این جلسه ذکر شد لزوم داشتن یک Console کامپیوتری کوچک وجود یـک ورودی/ خروجی استاندارد بر روی سیستم می باشد. ما در پروژه های قبـل از کلیـد فشـاری بـه عنـوان ورودی استفاده کردیم. البته می توان تعداد کلید های فشاری را زیادتر کرد، مثلاً ۱۶ کلید و هر کدام را به یک پـین از میکرو متصل نمود. ولی این کار باعث اشغال پایه های میکرو می شود. ما در سایر میکروهای کوچک پایـه هـای اضافی برای این کار نداریم و همیشه سعی میکنیم تا با تدابیری در صدد کاهش مصرف پایه ها وصرفه جویی در آنها برآییم. برای این منظور کی پد های 4x4 پا به عرصه وجود گذاشته اند. این کی پدها با داشتن ۱۶ کلید در طرح های مختلف و با قیمت های مناسب عرضه می شوند و با توجه به ماتریسی بودن سیم بندی داخلی فقط ۸ سیم ارتباطی دارند.

Keypad

در شکل زیر یک کی پد ۴ در ۴ ماتریسی، ساختمان داخلی و نحوه اتصال آن بـه یکـی از پـورت هـای میکرو (در بسکام) را مشاهده می فرمایید:

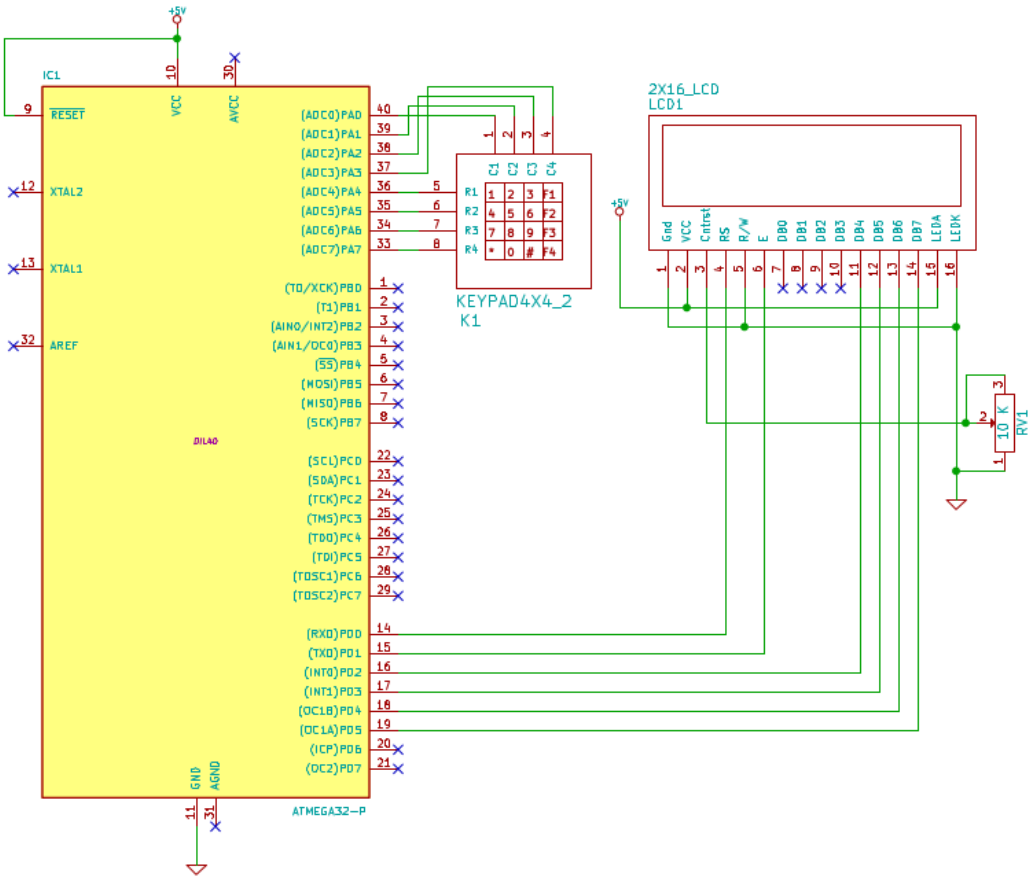


اکثر این کی پدها ساخت داخل بوده و از کیفیت خوبی برخوردارند. در داخل این کی پدها مدار خاصی وجود ندارد و در مدل ۴ در ۴ آن ۱۶ کلید فشاری کوچک قرار دارد که مطابق شکل سمت چپ به صورت ماتریسی به ۴ سیم سطر و ۴ سیم ستون متصل شده اند. برای خواندن کی پد در اسمبلی میتوانیم ۴ پین متصل شده به سطر ها را خروجی تعریف کرده و ۴ پین ستون را ورودی با پول آپ تعریف کنیم. سپس به طور مرتب سطرها را روشن کنیم و ستون ها را بخوانیم تا موقعیت کلید فشرده شده را بیابیم. در بسکام تمام این عملیات توسط دستور x=getkbd() انجام می شود و عدد مربوط به کلید فشرده شده (۰ تا ۱۵) در متغیر x برگردانده می شود. در صورتی که هیچ کلیدی فشرده نشده باشد عدد ۱۶ برگردانده می شود. این اعداد ایندکس هستند و به صورت زیر موقعیت کلیدها را بر میگردانند و هیچ ارتباطی با لیبل چسبانده شده روی کی پد ندارند.

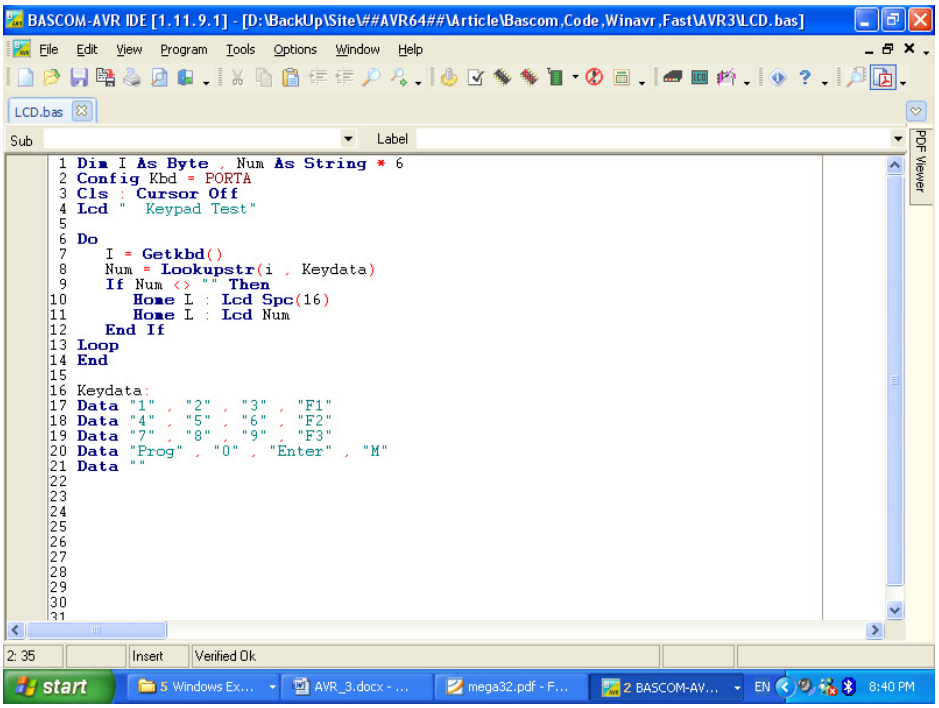
c1	c2	c3	c4	
0	1	2	3	R1
4	5	6	7	R2
8	9	10	11	R3
12	13	14	15	R4

کی پد را به هر پورتی از میکرو می توانید وصل کنید ولی در بسکام باید ترتیب آن به صورت شکل بالا باشد؛ یعنی ستون اول به پین ۰ و سطر آخر به پین ۷ پورت مورد نظر وصل شود.

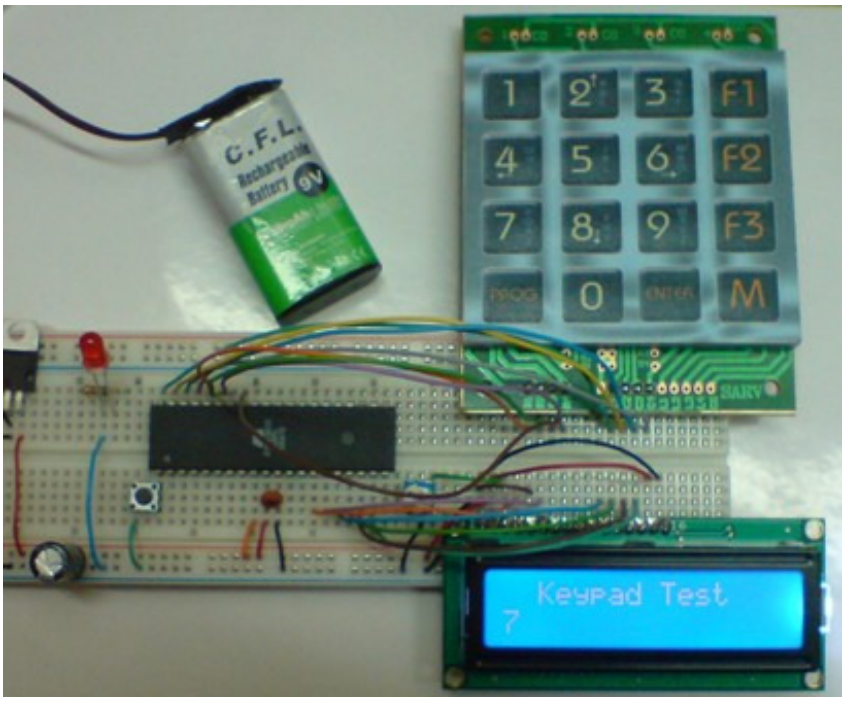
برای راه اندازی Keypad مداری مطابق شکل زیر ببندید. در شکل زیر عدد ۴ روی کی پد معرف ستون اول و کاراکتر D معرف سطر اول می باشد. در کی پد های استاندارد معمولاً ستون اول را با C1 و سطر اول را با R1 مشخص می کنند که باید به این نکته توجه داشته باشید. مداری مطابق زیر روی برد بورد ببندید تا ادامه آموزش را پیگیری کنیم.



کد های زیر را برای راه اندازی کی پد نوشته و وارد میکرو نمایید:



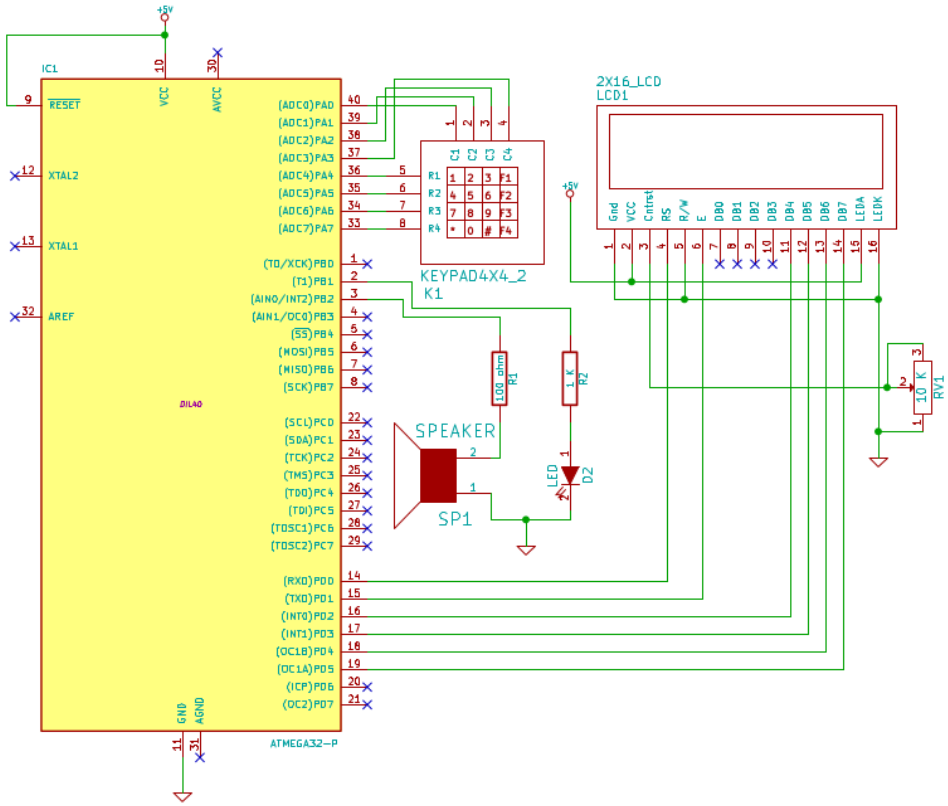
در خط اول دو متغیر یکی برای دریافت ایندکس به عنوان Byte و دیگری برای برگرداندن عبارت های چاپ شده روی کی پد به عنوان رشته ۶ تایی تعریف می کنیم. در خط دوم تعیین می کنیم که کی پد به کدام پورت وصل شده است. خطوط بعدی مانند پروژه قبل هستند. در حلقه محصور به Do..Loop در ابتدا کی پد را خوانده و ایندکس دریافتی را در i قرار می دهیم. دستور بعدی مقدار ایندکس i را در جدول keydata که پس از دستور End قرار گرفته بررسی میکند و کاراکتر متناظر با آن را در متغیر Num بر میگرداند. توجه کنید که آخرین ایندکس جدول Keydata یک رشته تهی است و شماره ایندکس آن هم برابر با ۱۶ می باشد (ایندکس ها از ۰ شروع می شوند)، سپس دستور بعدی محتوای Num را بررسی میکند؛ در صورتی که کلیدی فشرده نشده باشد مقدار ایندکس برابر با ۱۶ و مقدار Num یک رشته خالی خواهد بود که در نتیجه چیزی روی LCD مشاهده نمی شود. ولی در صورتی که عدد فشرده شده بین ۰ تا ۱۵ باشد کاراکتر مربوط به آن از جدول دیتا فراخوانی شده و مخالف با تهی خواهد بود و دستورات داخل بلوکIf اجرا میشود. در خط ۱۰ با دستور Home L مکان نما به خط دوم پرش میکند (این دستور معادل Locate 2,1 می باشد) سپس دستور جلوی آن ((LCD Spc(16) تعداد ۱۶ کاراکتر Space یا همان فضای خالی در سطر دوم چاپ می کند. این تنها روش برای پاک کردن محتویات سطر دوم بدون آسیب رساندن به محتویات سطر اول می باشد. دستورات خط بعدی نیز مقدار Num را در سطر دوم نمایشگر چاپ می کنند.



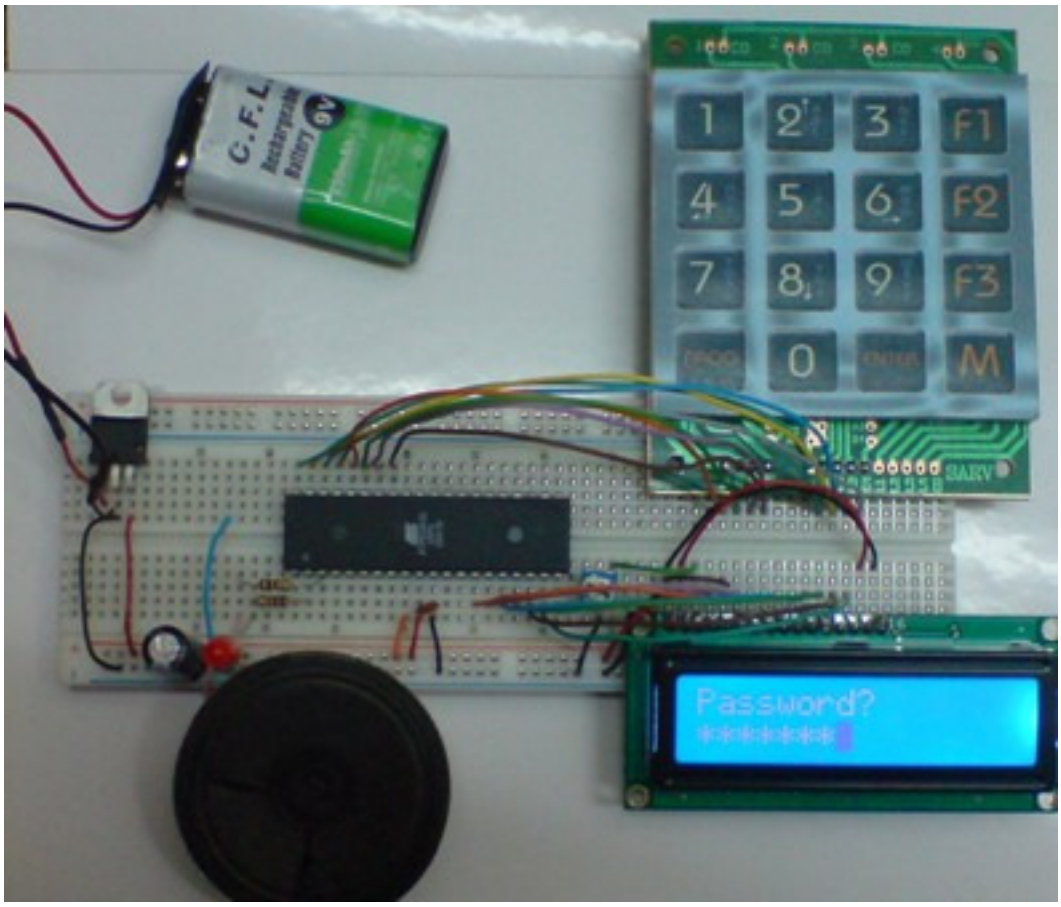
با استفاده از دستورات IF می توان کاراکتر های دریافتی را بررسی کرد و متناسب با آنها عبارات مختلفی را روی LCD نمایش داد و یا رله ای را برای کنترل بار روشن و خاموش کرد. بیسیک دستورات و توابع متنوعی از قبیل Sin() و... دارد که بهترین مرجع آن Help نرم افزار می باشد. همچنین با مشاهده مثال های گوناگون می توان ایده های مختلفی به دست آورد و در پروژه های بالا به کار برد. برای انتهای این مقاله تصمیم داریم یک پروژه قفل رمز دیجیتال ساده را طراحی و پیاده سازی نماییم که با دریافت رمز عبور 8 رقمی صحیح از کاربر یک دیود LED را روشن می کند که با تغییر اندکی می توان از آن برای تحریک یک رله و در نهایت باز کردن درب منزل استفاده کرد.

پروژه قفل امنیتی ۸ رقمی:

در شکل زیر نقشه قفل امنیتی دیده می شود و در این پروژه از یک LED به عنوان خروجی استفاده شده که می توان به جای آن یک رله قرار داد و در مقاله بعدی مدار لازم برای درایو رله شرح داده می شود. یک بلندگوی کوچک ۸ اهم نیز که با یک مقاومت ۱۰۰ اهمی سری شده است صداهای صفحه کلید و بوق خطا و موفقیت را به گوش کاربر می رساند.



مدار قفل رمز را مطابق شکل زیر بر روی Bread Board ببندید:



برنامه زیر را در بسکام کپی کرده و کامپایل کنید تا به توضیح آن بپردازیم: (تمام برنامه های این مقاله با فرض اینکه تنظیمات Options را مطابق با مقادیر ذکر شده در بالا قرار داده باشید عمل می کنند. یکی از مشکلات کار با بخش Options اینست که برنامه را از حالت Plain Text خارج کرده و قابلیت حمل آن را کم می کند. در صورتی هم که از Options استفاده نکنید مجبور به نوشتن کد زیادی هستید)

```
Dim I As Byte , Num As String * 1
Dim Pass As String * 8
Config Kbd = Porta
Config Pinb.1 = Output                                  'LED (Relay)
Config Pinb.2 = Output                                  'Speaker
Cursor Off

Main:
Pass = ""
Cls : Cursor Blink
Lcd "Password?"
Sound Pinb.2 , 120 , 20
Home L

Do
```

```
   I = Getkbd()
   Num = Lookupstr(i , Keydata)
   If Num <> "" Then
      Lcd "*"
      Pass = Pass + Num
      Sound Pinb.2 , 120 , 20
      '----------- Check Pass ------------
      If Len(pass) = 8 Then                                    'وقتی 8 رقم وارد شد
         Cursor Noblink
         If Pass = "12345678" Then                             'رمز درست
            Cls : Lcd "Welcome!"
            Sound Pinb.2 , 120 , 100
            Sound Pinb.2 , 120 , 80
            Sound Pinb.2 , 120 , 60
            Set Portb.1 : Wait 1 : Reset Portb.1
            Goto Main
         Else                                                  'رمز غلط
            Cls : Lcd "Error!"
            Sound Pinb.2 , 120 , 400
            Waitms 500 : Goto Main
         End If
      End If
      '-----------------------------------
      Waitms 250
   End If
Loop
End

Keydata:
Data "1" , "2" , "3" , ""
Data "4" , "5" , "6" , ""
Data "7" , "8" , "9" , ""
Data "" , "0" , "" , ""
Data ""
```

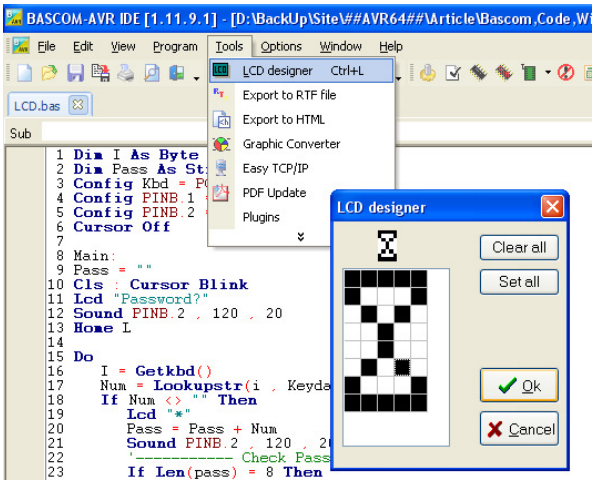
در خط اول مانند برنامه قبل دو متغیر یکی برای ذخیره ایندکس دریافتی از کی پد و دیگری برای دریافت رشته مربوط به کلید از جدول دیتا تعریف کرده ایم. در این برنامه متغیر دوم را یک رشته یک کاراکتری تعریف کرده و در جدول دیتا تمام کلیدها را با حداکثر یک کاراکتر مشخص کرده ایم و کلیدهایی که استفاده نمیشوند و همچنین ایندکس شانزدهم را با رشته تهی جایگزین کرده ایم. (رشته تهی با تایپ پشت سر هم دو کوتیشن "" بدون هیچ فضای خالی در بین آنها به وجود می آید). در خط دوم یک متغیر رشته ای با طول ۸ کاراکتر تعریف کرده ایم که رمز عبور دریافتی در آن قرار می گیرد.

در خطوط بعدی هم با دستورات Config دو پورت متصل شده به LED و بلندگو را به عنوان خروجی تعریف کرده ایم سپس مکان نما را خاموش کرده و برنامه اصلی را با برچسب Main آغاز کرده ایم. در این برچسب که پس از هر بار ورورد صحیح و یا غلط رمز به آن مراجعه می شود ابتدا محتویات دریافت شده از کاربر را برابر تهی قرار می دهیم (رشته حاوی رمز دریافتی صحیح یا غلط را خالی میکنیم) سپس صفحه را پاک کرده و مکان نما را به حالت چشمکزن درمی آوریم و در خط بالا عبارت Password? را می نویسیم و با دستور Sound یک بوق کوچک به بلندگو می فرستیم تا همانند کامپیوتر موفقیت آمیز بودن عمل Start-up را اعلام کنیم. سپس مکان نما را به خط دوم منتقل می کنیم.

به حلقه Do..Loop وارد شده و در آنها همانند برنامه قبل کاراکترها را از کی پد می گیریم ولی با این تفاوت که در صورت معتبر بودن کاراکتر (که در اینجا فقط اعداد ۰ تا ۹ معتبرند و بقیه با رشته تهی جایگزین شده و از شرط اعتبار عبور نمیکنند) به جای نمایش کاراکتر، علامت ستاره * را روی نمایشگر نشان می دهیم. (با هر بار اجرای دستور LCD مکان نما به طور خودکار یک ستون به سمت جلو حرکت می کند) سپس کاراکتر معتبر را با دستور + معمولی به رشته Pass اضافه می کنیم و با دستور Len طول رشته را می شماریم؛ در صورتی که ۸ کاراکتر وارد شد با دستور IF محتوای آن را چک می کنیم و در صورتی که مساوی با ۱۲۳۴۵۶۷۸ بود روی LCD عبارت Welcome را چاپ کرده و به مدت ۱ ثانیه LED را روشن می کنیم و ملودی کوتاهی می نوازیم و با دستور Goto به Main باز میگردیم. در صورت اشتباه بودن رمز نیز یک بوق خطا زده می شود و روی نمایشگر عبارت !Error نوشته می شود و به Main پرش میشود. دستور sound سه آرگومان می گیرد که اولی پورت متصل شده به بلندگو و دومی و سومی مربوط به فرکانس صدا می باشند. (در واقع دومی تعداد پالسها و سومی پریود هر پالس می باشد).

* * *

آخرین مطلب در مورد LCD حافظه موقتی ۸ کاراکتری آن است که برای طراحی کاراکترهای غیر استاندارد به کار می رود. به عکس ابتدای مقاله نگاه کنید؛ عبارت های فارسی نوشته شده بر روی نمایشگر و همچنین علامت سوال فارسی از جمله کاراکترهای طراحی شده هستند. برای طراحی یک کاراکتر خاص از منوی Tools روی LCD Designer کلیک کرده و کاراکتر خود را طراحی کنید و روی OK کلیک نمایید.




```
Deflcdchar ?,31,17,10,4,4,10,17,31
```

با این کار عبارتی مانند :

در جایی که مکان نما قرار دارد ظاهر می شود. بهتر است در خطوط ابتدای برنامه این کار را انجام دهید. سپس بایستی علامت سوال را با یک عدد بین ۰ تا ۷ جایگزین کنید تا چنین چیزی بدست آید:

```
Deflcdchar 0 , 31 , 17 , 10 , 4 , 4 , 10 , 17 , 31
```

برای نمایش کاراکتر مربوطه از دستور زبان زیر استفاده نمایید:

LCD Chr(0)

با این کار کاراکتر شماره ۰ بر روی نمایشگر مشاهده می شود. در صورتی که مـثلاً ۳ کـاراکتر مختلـف طراحـی کرده اید و تصمیم دارید که هر سه را پشت سر هم در یک جمله همراه با متغیر X حـاوی زمـان انتظـار نشـان دهید عبارات را با سمی کالن از هم جدا کنید (دستور LCD تعداد زیادی آرگومان می گیرد):

LCD “Wait,”;Chr(5);Chr(0);Chr(6);x;” Min”

در این سری مقالات فقط استارت اولیه کار زده شده و علاقه اولیه را در کاربر ایجاد می کنیم و تصمیم نداریم که کلیه دستورات را از همان ابتدا معرفی کنیم. نظر به اینکه در زبان های سـطح بـالا کـاربر از جزئیـات سخت افزار تقریباً دور نگه داشته می شود بهترین روش برنامه نویسی مطالعه کتابهای نوشته شده در زمینـه برنامـه نویسی کامپیوتر از قبیل C++ و VB می باشد؛ چرا که برنامه نویسی میکرو ها در این زبـان هـا فرقـی بـا کـامپیوتر نداشـته و بـا مطالعه کتابها و نمونه مثال های مختلف ایده های متفاوتی در ذهن کاربر ایجاد می شود. در نهایت مطالعه زبان هـایی مثـل ASP هم به دلیل شباهت بسیار زیاد ساختار VBScript به Bascom توصیه می شود. برای آشنایی با برنامه نویسی در حد مقدماتی مطالعه حداقل ۱۰ کتاب ۱۰۰۰ صفحه ای در زمینه اسمبلی، C، پاسکال و بیسیک توصیه می شود.

در جلسه بعد به معرفی دستگاه های ورودی خروجی پیشرفته تر از قبیل کی برد کامپیوتر و LCD گرافیکی می پـردازیم همچنین با سنسور دمای LM35 و نحوه راه اندازی بار 220v توسط رله و میکرو برای ساخت یک دماسـنج دیجیتـال هوشـمند آشنا می شویم.

ادامه دارد ...

مولف: بهنام زکی زاده
www.avr64.com
۲ فروردین ۱۳۸۹

آخرین ویرایش: ۱۳ مهر ۱۳۹۲ ✓