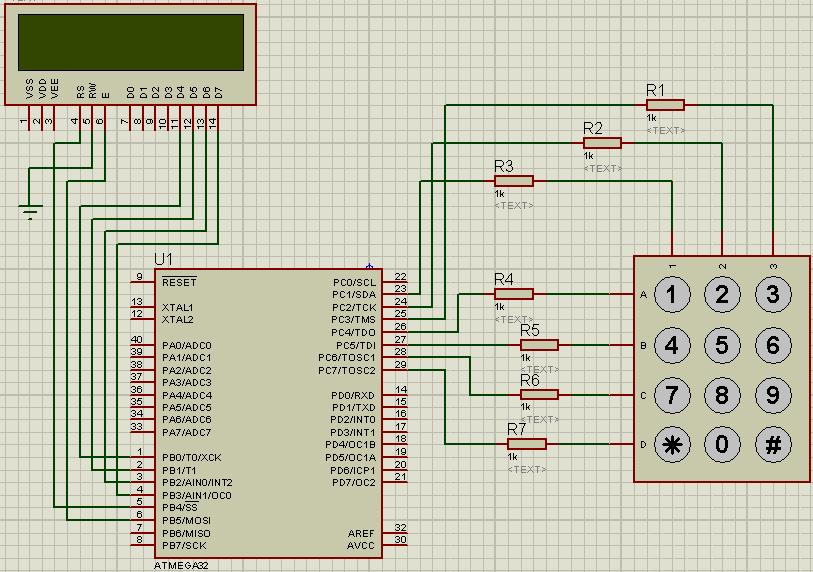
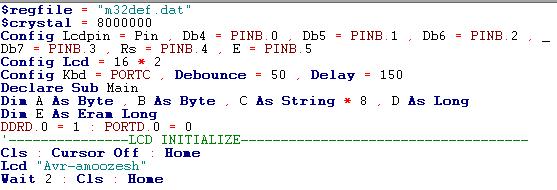
سلام به تمام دوستان گلم.http://blogfa.com/images/smileys/24.gif امیدوارم که همگی خوب باشید. فکر کنم این اولین پروژه عملی باشه که دارم رو وبلاگ میزارم. اما مطمئن باشید که آخریش نیست.http://blogfa.com/images/smileys/14.gif خوب عارضم به خدمت شما که توی این پروژه که اینجا براتون گذاشتم ما یه قفل رمز رو داریم که کد پیش فرض اون ۱۲۳۴۵۶ است و دفعه اول که مدار روشن میشه این کد رو از شما میخواد. و شما می تونید با وارد کردن رمز و زدن کلید # یه خروجی رو فعال کنید .اما شما میتونید بعد از این هر رمزی رو که دلتون خواست جایگزین اون کنید و این رمز میتونه هر عددی از یک رقم تا هشت رقم باشه، اما ترجیحاً به خاطر امنیت بالاتر سعی کنید که عددی رو که به عنوان رمز انتخاب میکنید بزرگتر از ۳ رقم باشه. حله؟ راستی اینم بگم چون مخاطب این مطلب افراد مبتدی فرض شده اند من راجع به همه چیز خیلی توضیح دادم اگر زیاد حوصله خوندن ندارید و مبتدی نیستید جملات داخل پرانتز رو نخونید.(با خودتون بگید، دلش خوش بوده یه چی گفته،بیخ

مداری رو که نقشه شماتیک اون رو در زیر مشاهده می کنید تشکیل شده از یک کیپد ۴\*۳ تلفنی یک Lcd ۲\*۱۶ و یک میکرو Atmega32 (البته چون من خونه فقط Mega32 داشتم اونو انتخاب کردم)

و برنامه اون هم که در زیر آورده شده به زبان بیسیک نوشته شده تا هم آموزشی برای دستورات بیسیک باشه و هم یه کار عملی کرده باشیم و لازم به ذکره که این پروژه رو من خودم روی برد برد بستم و هم توی Proteus شبیه سازی شده پس تو کارکرد اون شک و شبه ای نیست.

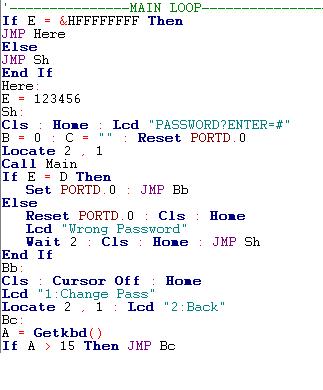
مشاهده فایل برنامه:<http://soslovek.persiangig.com/document/Code.txt>

که توی این برنامه ما ابتدا میکرو و فرکانس کاری اونو مشخص میکنیم و بعد  از اون LCD رو پیکر بندی می کنیم( یا به عبارتی به کامپایلر می فهمونیم که کدوم پایه های LCD رو به کدوم پایه ها از میکرو وصل کردیم تا وقتی که می خوایم رو LCD یه چیزی رو نمایش بدیم زحمت تولید پالس های کنترلی LCD به عهده کامپایلر باشه و به زبون ساده تر دیگه نیازی نیست که مثل یه زبان برنامه نویسی سطح پایین مثلاً اسمبلی تمام این پالسها رو تک تک برنامه نویسیشون رو خودمون انجام بدیم چون همونطوری که می دونید برای هر فرمانی که می خوایم به LCD بدیم باید کد اسکی اون دستور رو به LCD رسال کنیم و غیره و غیره و تمام این کارها رو کامپایلر برای ما، زمانی که می خواد برنامه رو کامپایل کنه انجام میده.)



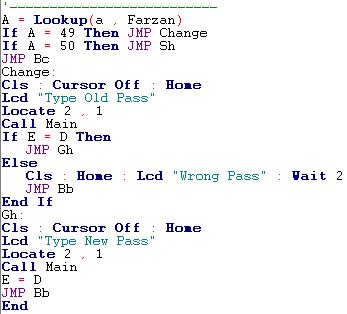
و بعد از اون کیپد متصل شده به میکرو رو پیکر بندی میکنیم( اینکه به کدوم پورت از میکرو وصل شده و اینکه و برای از بین بردن بانس(لرزش مکانیکی کلید ها، چون کلیدها قصعات مکانیکی هستند و موقعی که یه کلید رو فشار می دید چند میلی ثانیه ای کلید، بین صفر و یک لرزش میکنه) کلید ها چند میلی ثانیه می خواید تاخیر ایجاد کنید و دیگه اینکه چه میزان تاخیری رو می خواید بین فشرده شدن دو کلید ایجاد کنید یعنی اینکه وقتی کلیدی رو می خونید حداقل چه مدت بعدش فشرده شدن کلید بعدی رو آشکار کنه.) البته توضیحات مربوط به این دستورات رو تو پست های قبلی بهشون اشاره کردم.   
و بعد از اون یک سابروتین به نام Main تعریف کردیم که در ادامه بهش اشاره می کنم که توی این سابروتین چه اتفاقی می افته اما در این حد بگم که ما بعد از دستور END یعنی پایان برنامه، یک سری دستورات رو توی یک سابروتین که با Lable به نام Main شروع میشه قرار دادیم، چون که مجموعه دستوراتی که تو این سابروتین قرار داره، چند جای مختلف برنامه ما به این مجموعه دستورات نیاز داریم به جای اینکه هی این دستورات رو تو تنه ی اصلی برنامه تکرار کنیم، بعد از دستور END یک بار اون رو می نویسیم و هر جا تو تنه ی اصلی برنامه که به این دستورات نیاز داریم به این سابروتین پرش می کنیم. به همین سادگی این شد تعریف یک سابروتین. حله؟ که شکل این دستور به این صورته که شما قبل از اینکه این سابروتین رو در انتهای برنامه بنویسید و قبل از اینکه بخواید به اون سابروتین پرش کنید باید در ابتدای برنامه اون سابروتین رو توسط دستوری که مشاهده می کنید پیکر بندی کنید.

حالا متغیر های برنامه رو تعریف می کنیم که متغیر E همونطوری که میبینید توی حافظه EEPROM تعریف شده که در واقع ما رمز خودمون رو برای جلوگیری از پاک شدن در اثر قطع تغذیه مدار توی این حافظه میکرو می ریزیم که با قطع ولتاژ براش مشکلی پیش نیاد. چند تا دستورات بعدی هم که ساده اند دیگه اول LCD رو پاک می کنیم و اشاره گر LCD رو خاموش میکنیم و بعد جمله Avr-Amoozesh رو روی LCD به مدت 2 ثانیه نشون میدیم و بعد دوباره صفحه رو پاک می کنیم.



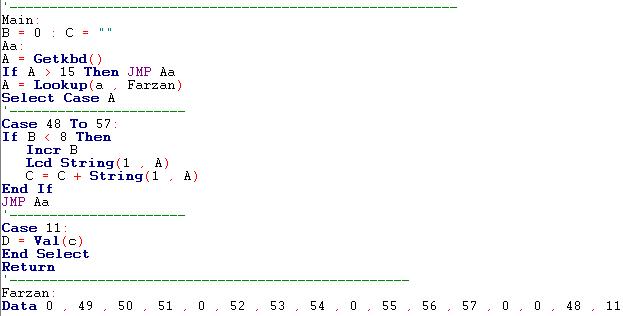
خوب حالا توی این قسمت اول ما محتوای متغیری که تو EEPROM تعریف کردیم رو با مقدار هگزادسیمال FFFFFFFF مقایسه می کنیم و اگر برابر بودند به HERE پرش می کند و در غیر اینصورت به SH پرش می کند. راستی اینو باید بگم که مقدار پیش فرض حافظه EEPROM صفر نیست یعنی اینکه تمام خانه های حافظه EEPROM در حالتی که هنوز مقدار دهی نشدند و عددی تو اون ها قرار نگرفته دارای مقدار یک منطقی هستند و متغیری هم که ما تو ابتدای برنامه تعریف کردیم از نوع LONG بوده که معادل چهار بایت از خانه های حافظه EEPROM است، که اگر تمام بیت های این چهار بایت برابر با یک منطقی باشند که در حالت پیش فرض نیز اینگونه است معادل با مقدار هگزا دسیمال FFFFFFFF خواهد بود و دلیل استفاده از این دستور شرطی اینه که بار اولی که میکرو روشن میشه، چون مقدار متغیر E که قبل از این هنوز مقدار دهی نشده  با FFFFFFFF برابر است، شرط حلقه برقرار میشه و دستور بعد از اون که عدد 123456تو متغیرمون میرزه اجرا میشه تا اینجا فهمیدید؟ اما وقتی که کاربر رمز رو عوض کنه و تغذیه مدار رو قطع کنه، دفعه بعد که میکرو روشن شد چون شرط حلقه برقرار نمیشه میپره به SH و دیگه کاری به عدد داخل E نداره. در واقع اگه ما این دستور شرطی رو به کار نبریم هر دفعه که رمز عوض شه و تغذیه قطع بشه و میکرو دوباره روشن شه، دوباره بدون هیچ سوال و جوابی همون رمز قبلی یعنی 123456 رو توی E میریزه. آخ دیگه از این بیشتر نمی تونم توضیح بدم امیدوارم که فهمیده باشید موضوع از چه قراره!

بعد از اون هم صفحه نمایش دوباره پاک میشه و عبارت Password?Enter=# نوشته میشه و بعد متغیر هارو با صفر مقدار دهی میکنه و در خط بعدی برنامه، مکان نمای LCD رو به سطر دوم میبره و بعد میپره  به سابروتین Main یا به عبارتی سابروتین رو  صدا میکنه و دستورات داخل این سابروتین اجرا میشند که کمی جلوتر بهش میرسیم فقط در همین حد بدونید که تو این سابروتین عدد رو از کیپد می خونه و با روشی که کمی جلوتر اون رو توضیح میدم، اعدادی که تک تک توسط کیپد آشکار میشند رو میگیره و همه رو توی یه متغیر به نام D میریزه و بعد برمیگرده به همون جایی از تنه اصلی برنامه که ازش پرش کرده و در یک دستور شرطی محتوای این متغیر، با متغیر E که تو  EEPROM قرار داره مقایسه میشه که در صورت تساوی PORTD.0 رو یک میکنه و به Bb میپره که تو این Lable شما امکان تغییر رمز و یا بازگشت به حالت درخواست رمز رو دارید اما اگر با متغیر E برابر نباشه LCD رو پاک میکنه و عبارت Wrong Pass رو نشون میده و دوباره به قسمت درخواست رمز یعنی Sh پرش میکنه.  
اما اگر عدد وارد شده با متغیر E برابر بود و به Bb پرش کرد، در این Lable ابتدا LCD پاک میشه و سپس دو عبارت Change Pass:1  و Back:2 رو LCD نشون داده میشند که به این معنیه که اگر عدد 2  رو وارد کنید برنامه به Sh پرش میکنه یا به عبارتی Back میشه به پرسش دوباره رمز و اگر 1 رو فشار بدید میپره به قسمتی که شما رمز رو میخواید تغییر بدید.

  
اما حالا برنامه چطور این کار رو انجام میده؟ به این صورت که اول کیپد رو با دستور Getkbd میخونیم و نحوه کار این دستور به این صورته که اگه عددی فشار داده نشده باشد مقدار 16 رو برمیگدونه تو متغیر A و ما با یک دستور شرطی که در خط بعدی گذاشتیم خیلی راحت اگر کلیدی فشار داده نشده باشد به Bc پرش می کنیم و دوباره به دستور Getkbd میرسیم که این حلقه تا زمانی که عددی فشار داده نشود ادامه دارد.

اما اگر عددی فشار داده شود به خط بعدی برنامه میریم که تو اون میتونیم بفهمیم که عدد وارد شده چند بوده، این دستور یعنی (LOOKUP(A,Farzan از جدولی که بعد از لیبل Farzan قرار گرفته عدد A ام را به داخل متغیر A میریزد که در این جدول کلیه اعداد به دلیلی که جلوتر به اون اشاره میکنیم به معادل اسکی اون ها تبدیل میکنه و داخل متغیر A میریزه و بعد در خطوط بعدی برنامه چک میشه که کدوم دو عدد 1 یا 2 وارد شده اند که هرکدوم از شرط ها که برقرار بشه به لیبل مخصوص به خودشون پرش میکنن و اگر عددی غیر از 1 یا 2 وارد بشه هیچ کدوم از شرط ها برقرار نمیشه درنتیجه برنامه در خط بعدی دوباره به دستور Getkbd پرش میکنه و منتظر وارد شدن عددی میشه.  
چند خط بعدی برنامه هم که خیلی ساده است اگر عدد وارد شده 1 باشه یعنی بخوایم رمز رو تغییر بدیم ابتدا رمز قبلی از شما سوال میشه که اگر درست باشه به لیبل بعدی میره که اونجا رمز جدید رو از شما می خواد و به محض وارد کردن رمز جدید و زدن کلید # اون رو به جای مقدار قبلی متغیر E جایگزین میکنه و اگر رمز قبلی رو درست نزنید به شما میگه که رمز اشتباهه و دوباره به Bb پرش میکنه تا اینجا حله؟

خوب اما بریم سر وقت سابروتین اصلی برنامه که تقریباً اصل کار اونجا انجام میشه:



ببینید دوستان، ما برای اجرای یه پروژه قفل رمز که قراره دو تا عدد رو با هم مقایسه کنه دو تا راه بیشتر نداریم( چون اعداد که از کیپد وارد میشن به صورت منفصل میایند یعنی تک تک میان و ارزش مکانی ندارند و ما باید تو برناممون به اونا ارزش مکانی بدیم. میفهمید چی میگم؟ مثلاً تو عدد 3546 ارزش مکانی عدد 1000\*3 است نه صرفاً عدد3. اما وقتی که ما از کیپد عدد میگیریم این اعداد فقط وارد میشند بدون ارزش مکانی) یا اینکه باید بیایم هر عددی که از کیپد می خونیم رو توی یه متغیر جداگانه بریزیم و بعد رقم به رقم با رمز اصلی مقایسه کنیم که این روش خیلی غیر حرفه ایه. اما ترفند دیگه ای که ما اینجا زدیم اینه که اومدیم توی جدول LOOKUP تمام اعداد رو به فرم اسکی ( ASCII ) برگردوندیم و با دستور String (1 , A که کد اسکی A رو به رشته تبدیل میکنه اون رو به رشته تبدیل کردیم و با دستور  
 (C= C + String( 1 , A با اعداد قبلی همه رو ریختیم توی یه متغیر یعنی اول اعداد فشرده شده روی کیپد به کد اسکی تبدیل میشند بعد کد اسکی اون اعداد به رشته اون اعداد تبدیل میشه، یعنی مثلاً عدد 2 میشه "2" و با رشته اعداد بعدی جمع شده و توی یه متغیر رشته ای به نام C ریخته میشند و بعد زمانی که کلید # فشار داده میشه که به منزله کلید ENTER عمل میکنه، توسط دستور Val دوباره این رشته ی عددی تبدیل به یه متغیر عددی میشه یعنی، رشته "123456" تبدیل میشه به متغیر 123456 امیدوارم که درک کرده باشید که چه کردیم چون دیگه واقعاً زبونم از توضیح بیشتر قاصر است. یه سری توضیجات مربوط به جدول LOOPUP و سخت افزار مدار هم یکی دو روز دیگه مینویسم، مردونه الان دیگه خیلی خستم.