شبیه سازی قفل صندوق با پردازنده 8086

دانشجویان:عرفان زین‌الدینی– امیرحسین سلاجقه-
علیرضا عبداللهی‌نژاد

استاد: دکتر مهدیه قزوینی

درس: ریزپردازنده و زبان اسمبلی.

بهار ۱۴۰۳

فهرست

1. مقدمه

مقدمه

یکی از ساده‌ترین و پرکاربردترین سیستم‌های امنیتی که در زندگی روزمره برای محافظت از اشیاء با ارزش استفاده می‌کنیم، قفل صندوق است. قفل صندوق دیجیتال، یک سیستم الکترونیکی است که قابلیت قفل کردن و باز کردن صندوق با استفاده از یک کد عددی را دارد. برای این منظور، در این پروژه، یک سیستم قفل صندوق دیجیتال با استفاده از زبان اسمبلی برای پردازنده 8086 پیاده‌سازی شده است. این سیستم شبیه‌سازی شده قابلیت استفاده از یک کد اصلی (MASTERUNLOCK) و یک پین کد قابل تغییر (PINLOCK)را دارا می‌باشد.

زبان اسمبلی 8086، یکی از زبان‌های برنامه‌نویسی سطح پایین است که امکان کنترل مستقیم سخت‌افزار را فراهم می‌کند. با استفاده از این زبان، می‌توان به راحتی با پورت‌های ورودی/خروجی ارتباط برقرار کرد و عملیات سطح پایین را انجام داد.

شبیه‌سازی قفل صندوق، علاوه بر ظاهر ساده آن، شامل پیاده‌سازی‌های متعددی برای خواندن ورودی از صفحه کلید، پردازش کدها، کنترل وضعیت قفل، و نمایش وضعیت روی یک نمایشگر 7-سگمنتی است. برای طراحی این سیستم، به درک عمیقی از نحوه کار با پورت‌هایI/O ، مدیریت حافظه، و پردازش داده در سطح بیت نیاز داریم.

در ادامه این داکیومنت، ابتدا به معرفی و تشریح مختصر قطعات مدار سپس، شیوه طراحی و پیاده‌سازی سیستم را بررسی می‌کنیم. در نهایت، نحوه استفاده از سیستم و مثال‌هایی از عملکرد آن را ارائه خواهیم داد.

2. معرفی ابزارها و قطعات

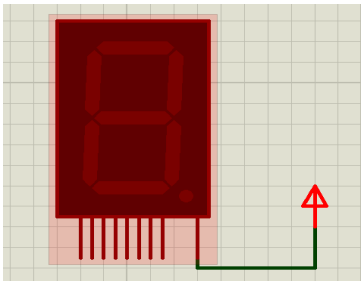
2.1. نمایشگر وضعیت[1]

یکی از کاربردی‌ترین اجزا در یک سیستم قفل صندوق دیجیتال، نمایشگر وضعیت می‌باشد. از این نمایشگر برای نشان دادن وضعیت قفل (باز یا بسته بودن) و همچنین نمایش برخی اطلاعات دیگر مانند تعداد تلاش‌های ناموفق استفاده می‌شود. یکی از رایج‌ترین نمایشگرهای مورد استفاده در چنین پروژه‌هایی، نمایشگر 7-سگمنتی است .

نمایشگر 7-سگمنتی از هفت قطعه LED تشکیل شده که می‌تواند اعداد 0 تا 9 و برخی حروف الفبا را نمایش دهد. در پروژه قفل صندوق ما، از این نمایشگر برای نمایش حرف "L" (Locked) در حالت قفل و "U" (Unlocked) در حالت باز استفاده شده است.

این نمایشگر دارای 7 پایه اصلی (یکی برای هر سگمنت) و یک پایه مشترک است. برای کنترل این نمایشگر، معمولاً از یک درایور استفاده می‌شود که می‌تواند از طریق پورت‌های I/O کنترل شود. در کد اسمبلی ما، مقادیر 0C7H برای نمایش "L" و 0C1H برای نمایش "U" استفاده شده است.

استفاده از نمایشگر 7-سگمنتی در این پروژه، علاوه بر ساده بودن، مزیت مصرف انرژی کم و قابلیت دید در شرایط نوری مختلف را دارد. همچنین، این نوع نمایشگر برای نمایش اطلاعات ساده مانند وضعیت قفل، بسیار مناسب و کارآمد است.



شکل 2.1 : تصویری از 7-segment در پرتئوس

زمانی که پایه RS این پردازنده در سطح پایین(0 منطقی) قرار گیرد, پایه های داده دستور را به صفحه نمایش منطقل می کنند. برای مثال برای تعیین مکان نمایش گر, نوشتن در خط اول یا دوم باید به کمک دستورات برنامه نویسی مناسب داده مناسب را روی بیت های داده این تراشه قرار دهیم. زمانی که در سطح بالا(1 منطقی) قرار گیرد, مقادیر به عنوان داده در نظر گرفته می شود. پایه RW زمانی که در

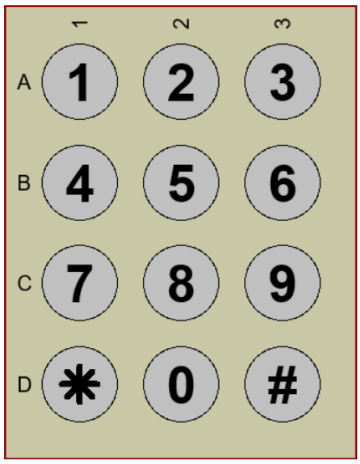
[1] 7-segment

سطح پایین قرار گیرد, داده ها از کنترل کننده به صفحه نمایش منتقل می شوند اما اگر در سطح بالا باشد, داده ها از صفحه نمایش به کنترل کننده منتقل می شوند.

2.2. صفحه کلید[2]

صفحه کلید, مهمترین جز برای ارتباط کاربر با قفل صندوق می باشد. از طریق صفحه کلید اعداد به دستگاه اعلام می شوند و همچنین عملیات مورد نظر انتخاب می شود. برای این منظور از یک صفحه کلید ماتریسی 4x3 استفاده می کنیم. این صفحه کلید شامل ارقام 0 تا 9 و کاراکترهای "#" و "*" میباشد

در نرم افزار پرتئوس[3], در دسته ابزار های صفحه کلید, صفحه کلیدی تحت عنوان keypad-phone وجود دارد. این ابزار 12 کلید اشاره شده را دارا است. همچنین این ابزار دارای 7 پایه ی خروجی می باشد. پایه های 1 و 2 و 3 که در بالا آن قرار دارند و همچنین پایه های A و B و C و D که در کنار آن قرار دارند, برای تعیین سطر و ستون کلید فشار داده شده به کار می روند. برای سادگی تعیین کلید فشار داده شده, تراشه خاصی برای تعیین کلید به کار می رود. در ادامه به معرفی این تراشه می پردازیم.



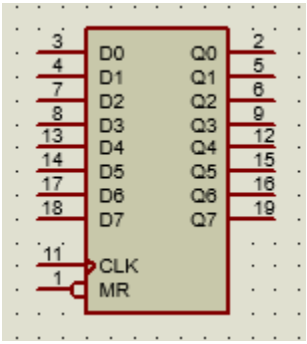
شکل 2.2 : تصویری از keypad-phone در پرتئوس

2.3. فلیپ فلاپ[4]

فلیپ فلاپ, همانن لچ ها یک نوع حافظه موقت هستند. فلیپ فلاپ ها انواع گوناگونی دارند که از دی فلیپ فلاپ(D-Flip Flop) استفاده می کنیم. این قطعه هنگامی که کلاک بخورد, مقدار ورودی اش در خروجی قرار می گیرد تا زمانی که دوباره کلاک بخورد. تراشه 74273 نمونه ای از فلیپ فلاپ 8 ورودی

[2] Keypad
[3] proteus
[4] Flip Flop

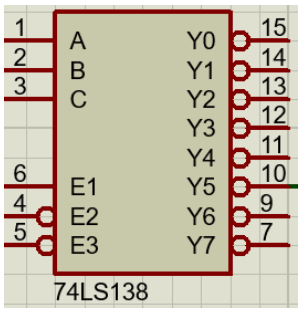
می باشد. این تراشه سیگنالی تحت عنوان MR را دارا می باشد. این سیگنال تعیین می کند که خروجی روشن یا خاموش باشد.



شکل 2.3 : شماتیک تراشه 74273 یا همان دی فلیپ فلاپ در پرتئوس

2.4.تراشه 74LS138

تراشه 74LS138 نوعی مدار مجتمع است که به عنوان دیکدر[5] 3 به 8 شناخته میشود به صورتی که سه ورودی باینری[6] گرفته و یکی از 8 خروجی را بر اساس سه ورودی فعال میکند.



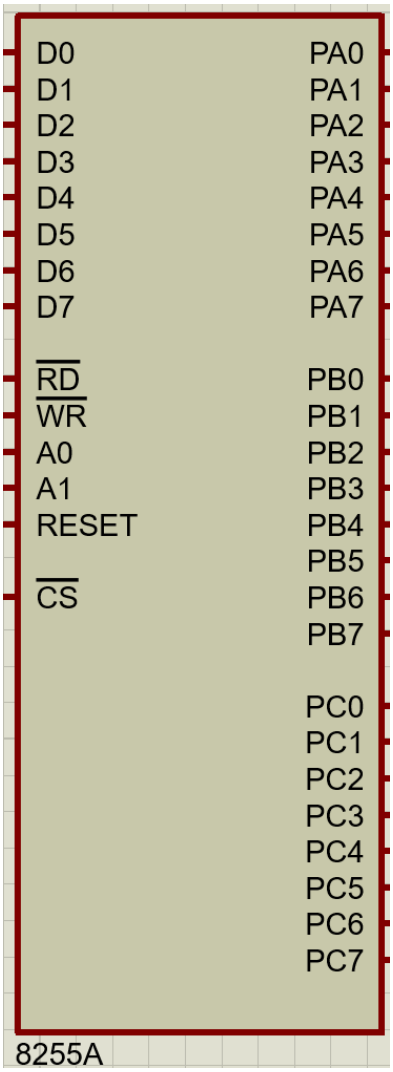
شکل 2.4: شماتیک تراشه 74LS138 در پرتئوس

2.5.تراشه 8255

رابط قابل برنامه ریزی 8255 یک تراشه بسیار متداول است که امروزه کاربردهای زیادی یافته است . این تراشه دارای 24 پایه برای I/O است که در گروههای 12 پایه ای قابل برنامه ریزی هستند . هر گروهی می تواند در به حالت مجزا کار کند I/O : ساده ، IOاستروب شده و I/O دو طرفه ، 8255 قادر است هر وسیله I/O موازی سازگار با TTL را به آسانی به ریز پردازنده ارتباط دهد. برای ایجاد برخی از سیگنال های کنترلی دستگاه ها, قرار دادن داده ها در خروجی از این تراشه قابل برنامه ریزی استفاده می کنیم. این تراشه 3 گروه 8 بیتی پورت انتقال داده دارد که در یک لحظه تنها یکی از آن ها می تواند فعال باشد. همچنین این تراشه دارای 8 بیت برای دریافت یا قرار دادن داده روی باس دارد. به کمک دوپایه ی A1,A0 می توان انتخاب کرد که کدام پورت باید انتخاب شود. همچنین حالت قابل انتخاب دیگر حالت کنترل تراشه است. در این حالت داده های ورودی به ریجستر کنترلی داخل تراشه

[5] decoder
[6] binary

منتقل می شوند و تنظیمات کنترلی تراشه بر آن اساس انجام می شود. دو سیگنال RD,WR که هر دو در سطح پایین فعال می شوند, برای تعیین نوع عملیات(نوشتن یا خواندن) به کار می روند. دو پایه ی دیگر این تراشه, پایه های انتخاب تراشه(CS) و پایه راه اندازی مجدد(RESET) می باشد.



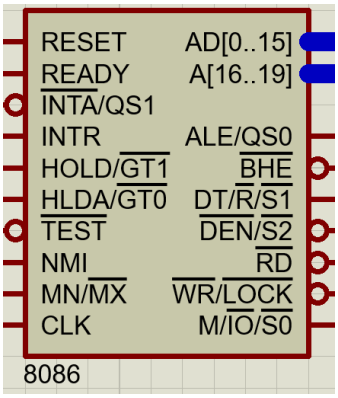
شکل 2.8 : تراشه 8255 در پرتئوس

2.6. میکروپردازنده 8086[7]

در هر پروژه ای, بخشی از پروژه وظیفه کنترل و مدیریت سایر اجزا را به عهده دارد. این کار توسط پردازنده و در پروژه های کوچک تر توسط میکروپردازنده انجام می شود. میکروپردازنده 8086, یک میکروپردازنده 40 پایه با دیتاباس 16 بیتی و آدرس باس 20 بیتی است. از آن رو که پایه های این پردازنده در زمان های متفاوت عملکرد متفاوتی دارند, باید در هنگام اتصال دستگاه ها به باس ها, از بافر برای

[7] Microprocessor

قرار دادن داده و از لچ برای نگهداری داده و جلوگیری از هدر رفتن آن استفاده کنیم. برای تعیین این که چه عملی قرار هست انجام شود یا چه نوع داده یا سیگنالی روی پایه ها قرار گرفته اند, از 16 پایه این پردازنده برای سیگنال های کنترلی استفاده شده است.



شکل 2.9 : تصویری از میکروپردازنده 8086 در پرتئوس

3. روش کار و پیاده سازی

در این فصل به کمک سخت افزار و تراشه های معرفی شده و همچنین گیت های منطقی, چگونگی طراحی مدارهای مورد نیاز را شرح می دهیم. در ابتدا در یک تقسیم بندی کلی, می توانیم سخت افزار پروژه مان را به 5 بخش تقسیم کنیم:

1. میکروپردازنده و ایجاد باس مشترک آدرس و داده
2. جداسازی آدرس و داده
3. ایجاد سیگنال های فعال سازی و رمزگشایی آدرس پورت ها
4. صفحه کلید و طریقه استفاده
5. نمایشگر صحفه و طریقه استفاده

در ادامه به تشریح هر یک از این بخش ها و شیوه عملکرد و ساخت آن هارا شرح می دهیم.

3.1. مدار های میکروپردازنده و ایجاد باس مشترک آدرس و داده

میکروپردازنده 8086 را از قسمت ابزار ها انتخاب و آن را به پروژه اضافه میکنیم. در قسمت تنظیمات تراشه فرکانس داخلی را انتخاب کرده و مقدار آن را روی 1500KHz قرار میدهیم. با این کار فرکانس کاری پردازنده را روی این مقدار تنظیم کرده ایم. در این فرکانس پردازنده بهترین عملکرد را برای این پروژه خواهد داشت. در گام بعدی باید برخی پایه ها را مقدار دهی کنیم. پایه ی راه اندازی مجدد[8]

[8] RESET

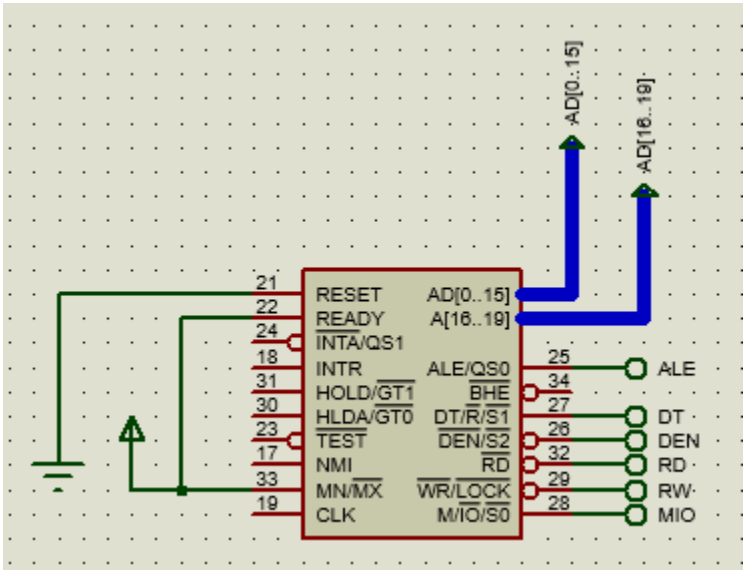
را به زمین اتصال میدهیم. همان طور که می دانید, در طول اجرا پروژه قفل صندوق حساب نیازی به استفاده از این پایه و راه اندازی مجدد پردازنده نداریم, پس مقدار این پایه را با اتصال به زمین برابر 0 منطقی قرار می دهیم تا از راه اندازی مجدد پردازنده جلوگیری کنیم.

در گام بعدی باید تعیین کنید که پردازنده در حالت بیشینه یا کمینه فعالیت می کند. برای این منظور از آن رو که پردازنده در حالت کمینه و به صورت مستقل مورد استفاده قرار می گیرد, پایه ی 33 پردازنده (پایه ی mn/mx) را به منبع تغذیه اتصال دهیم. با این کار 1 منطقی را روی این پایه قرار گرفته و سیگنال حالت کمینه فعال می شود.

در مرحله بعد پایه ی آمادگی[9] پردازنده را مقدار دهی میکنیم. از آن رو که از حافظه یا سخت افزار خاصی که تاخیری برای قرار دادن داده روی باس ها داشته باشد استفاده نمی شود, این پایه را نیز به 1 منطقی یا همان منبع تغذیه اتصال میدهیم.

برای سادگی در دسترسی به پایه های داده و آدرس, آن ها را در یک باس قرار دهید و نامی متناسب برای آن انتخاب کنید. در اینجا برای پایه های مشترک بین داده و آدرس, باس AD[0..15] و پایه های مشترک بین آدرس و وضعیت باس A[16..19] در نظر گرفته شده است.

از مقدار پایه ALE در مدار جداسازی آدرس, از پایه های RD و WR برای ایجاد سیگنال فعال سازی تراشه 8255 استفاده می کنیم. همچنین پایه های M/IO و DT و DEN نیز برای تولید برخی سیگنال ها به کار می روند. برای سادگی و جلوگیری از پیچیده شدن اتصالات, آن هارا نشانه گذاری کنید و روی آن اتصال, برچسب مناسبی قرار دهید.

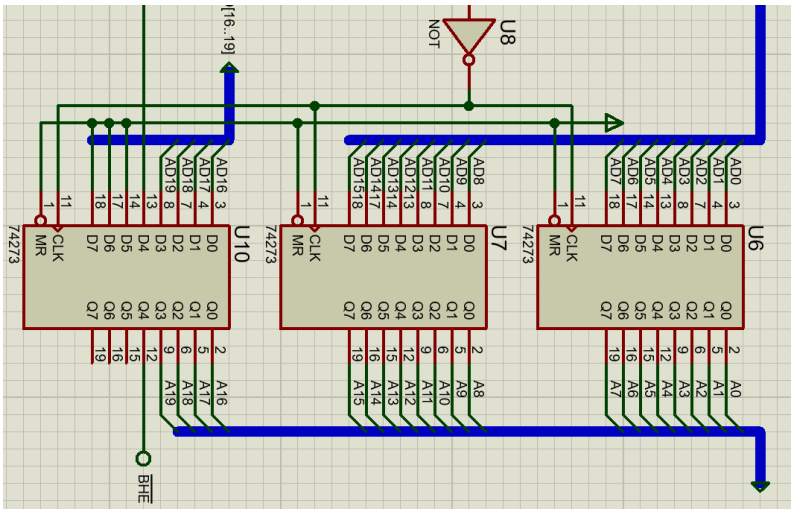


شکل 3.1 : نمایی از پردازنده 8086 به همراه اتصالات مورد نیاز

[9] READY

3.2.جداسازی آدرس و داده

همان طور که اشاره کردیم, پایه های داده و آدرس در پردازنده 8086 مشترک هستند. برای جدا سازی آدرس ها از داده, از تراشه[10] 74273 استفاده می کنیم. در پردازنده 8086 , 20 خط آدرس وجود دارد. 16 خط کم ارزش این آدرس ها مشترک با داده می باشند و در یک باس, و 4 بیت پر ارزش آن مشترک با وضعیت و در یک باس دیگر قرار دارند. تراشه های موجود, 8 بیتی می باشند. برای جداسازی کامل 20 خط آدرس از سه تراشه 8 بیتی استفاده کنید. ورودی های تراشه اول را به باس مشترک بین داده و آدرس, یعنی باس AD[0..15] متصل میکنیم. پس از اتصال ورودی ها, روی هر ورودی برچسب مناسبی قرار میدهیم. این برچسب نشان می دهد که این سیم به کدام بیت از باس متصل شده است. 4 بیت کم ارزش از تراشه سوم را به باس مشترک بین وضعیت و آدرس یا همان باس A[16..19] اتصال میدهیم و برچسب مناسب را روی هر اتصال قرار میدهیم. سیگنال کنترلی فعال ساز[11] ها باید مشترک باشد. این سیگنال باید زمانی فعال شود که پردازنده قصد قرار دادن آدرس را روی باس دارد. خود پردازنده پایه و سیگنالی تحت عنوان ALE دارد. این پایه را که از پیش برچسب گذاری کرده بودیم, اکنون با استفاده از یک عملگر NOT به طور مشترک به هر سه تراشه اتصال می دهیم. سیگنال کنترلی دیگر تراشه ها که برای روشن یا خاموش بودن تراشه ها به کار می رود,سیگنال MR است. این سیگنال در سطح بالا فعال می شود. از آن رو که همیشه نیاز داریم که لچ های ما روشن باشند, آن را به یک منطقی یا منبع ولتاژ اتصال میدهیم. اکنون نوبت به اتصال خروجی تراشه ها می رسد. برای سادگی در دسترسی و جلوگیری از شلوغی, خطوط باس برای خروجی طراحی میکنیم. این باس مقدار آدرس را روی خود دارد, پس برچسب مناسب با آن A[0..19] است. خروجی سه تراشه را به این باس اتصال می دهیم و روی هر اتصال برچسب مناسب با آن اتصال قرار می دهیم. اکنون یک آدرس داریم که هرجا نیازمند استفاده و رمزگشایی آدرس باشد, می توان از آن استفاده کرد.



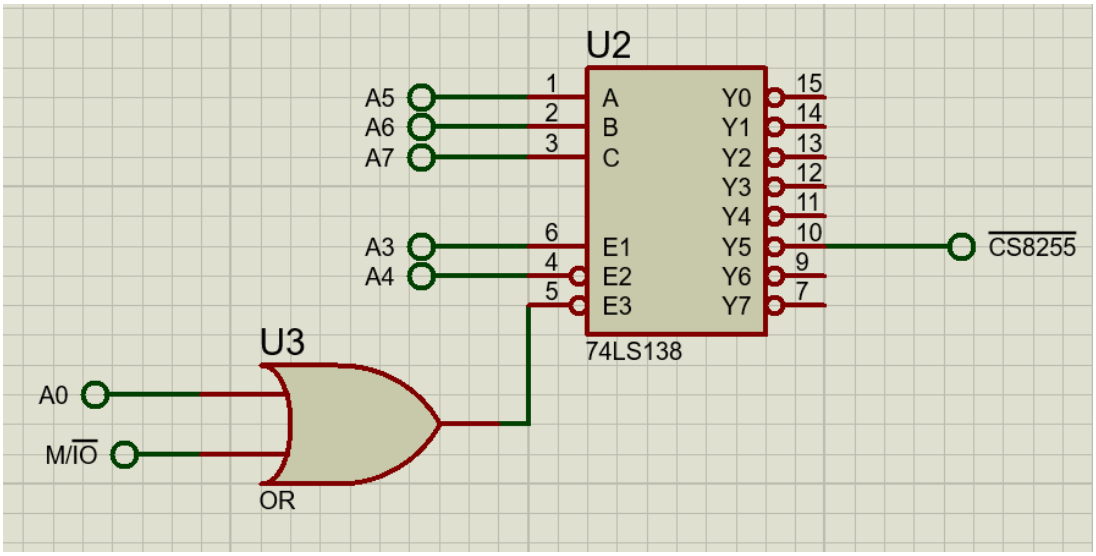
شکل 3.2 : شیوه اتصال تراشه های 74273 به باس ها و جداسازی آدرس

[10] chip
[11] enable

3.3. ایجاد سیگنال های فعال سازی و رمزگشایی آدرس پورت های 8255

برای خواندن از یک دستگاه و یا نوشتن در یک دستگاه, نیازمند طراحی پورت متناسب با آن در گاه هستیم. هر پورت یک آدرس دارد. همچنین هر پورت در 8255 نیازمند یک سری سیگنال ها مانند RD RW , می باشد. در این تراشه میتوانیم خطوط RD و WR پردازنده 8086 را مستقیم به 8255 وصل کنیم.

به کمک باس آدرس و سیگنال های کنترلی که پردازنده ایجاد می کند می توانیم سیگنال فعال ساز یا CS ، ۸۲۵۵ را با استفاده از دیکودر 74LS138 ایجاد کنیم، این تراشه باید با آدرس 0x8000 فعال شود در نتیجه خطوط ادرس A5,A6,A7 را به ترتیب چپ به راست به ورودی ها دیکودر و خطوط آدرس A3,A4 را به Enable یک و دو وصل کرده و سیگنال های A0 و M/IO را با استفاده از گیت OR به E3 وصل میکنیم



شکل 3.3 : پیاده سازی سیگنال CS با استفاده از دیکودر 74LS138

از دیگر سیگنال های مورد استفاده, سیگنال های A0,A1 برای ورودی های مربوطه در 8255 می باشد. این ورودی ها نشان می دهند که کدام پورت از این تراشه باید انتخاب شود. جدول درستی این تراشه به صورت زیر می باشد:

A1	A0	Result
0	0	Select port A
0	1	Select port B
1	0	Select port C
0	1	Select control register

آدرس ProtA: 0A8H

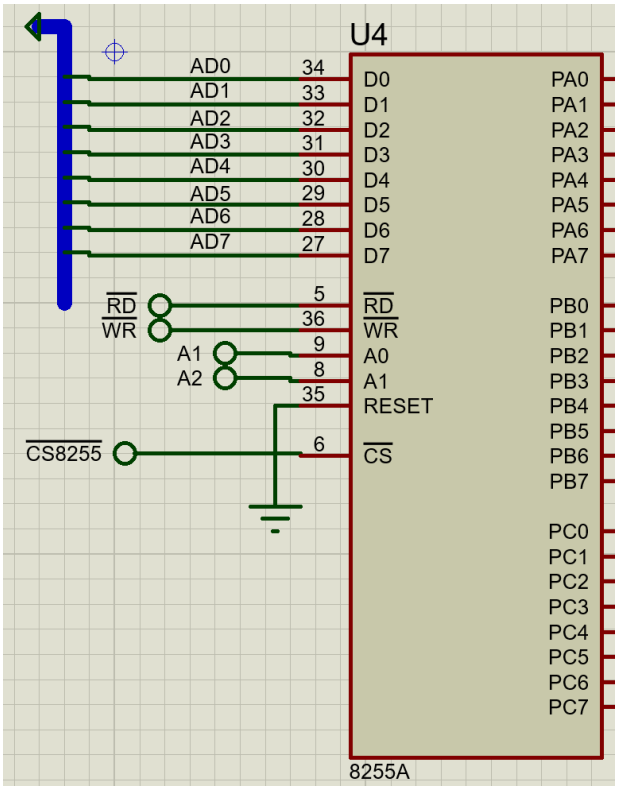
آدرس portB: 0AAH

آدرس PortC: 0ACH

جدول 3.1 : جدول درستی 8255

مقدار دو این دو سیگنال را به کمک خطوط آدرس A1,A2 ایجاد می کنیم.

برای خواندن و نوشتن از گذرگاه های AD[0-7] استفاده میکنم و به خطوط D[0-7] متصل میکنیم



شکل 3.4 : ورودی های تراشه ۸۲۵۵

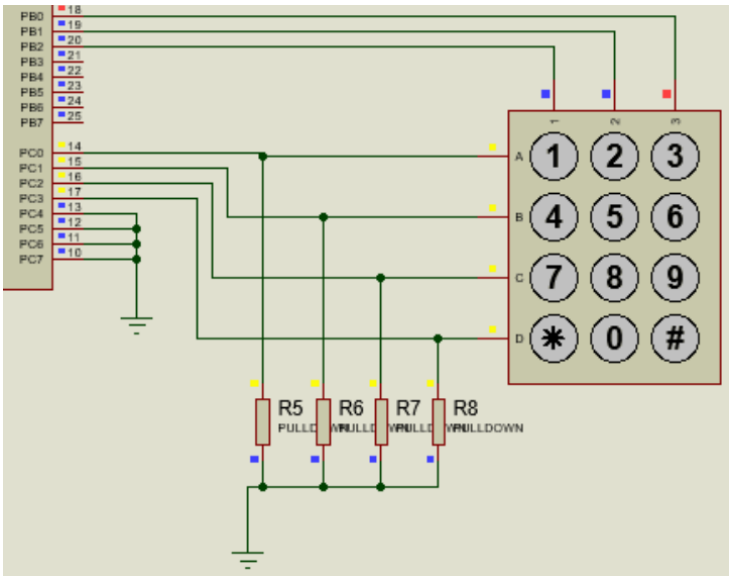
3.4.صفحه کلید

برای خواندن از صفحه کلید, از روش صحفه کلید ماتریکسی[12] استفاده میکنیم در این روش سطر ها را به صورت ورودی و ستون ها را خروجی در نظر میگیریم.

برای اتصال سطرها از خطوط ادرس Portc تراشه 8255 که آدرس آن 0ACH استفاده میکنیم برای نویز گیری از رجیستر polldown استفاده میکنیم و مقدار تاخیر 8255 را 20ns قرار میدهیم.

برای اتصال ستون ها از خطوط آدرس portB تراشه 8255 که آدرس آن 0AAH استفاده میکنیم.

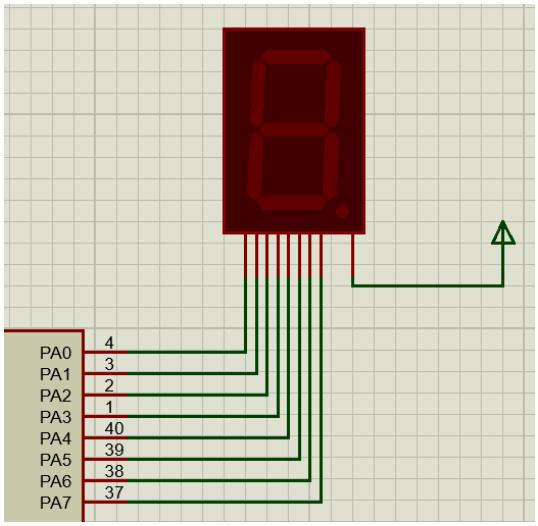
[12] matrix



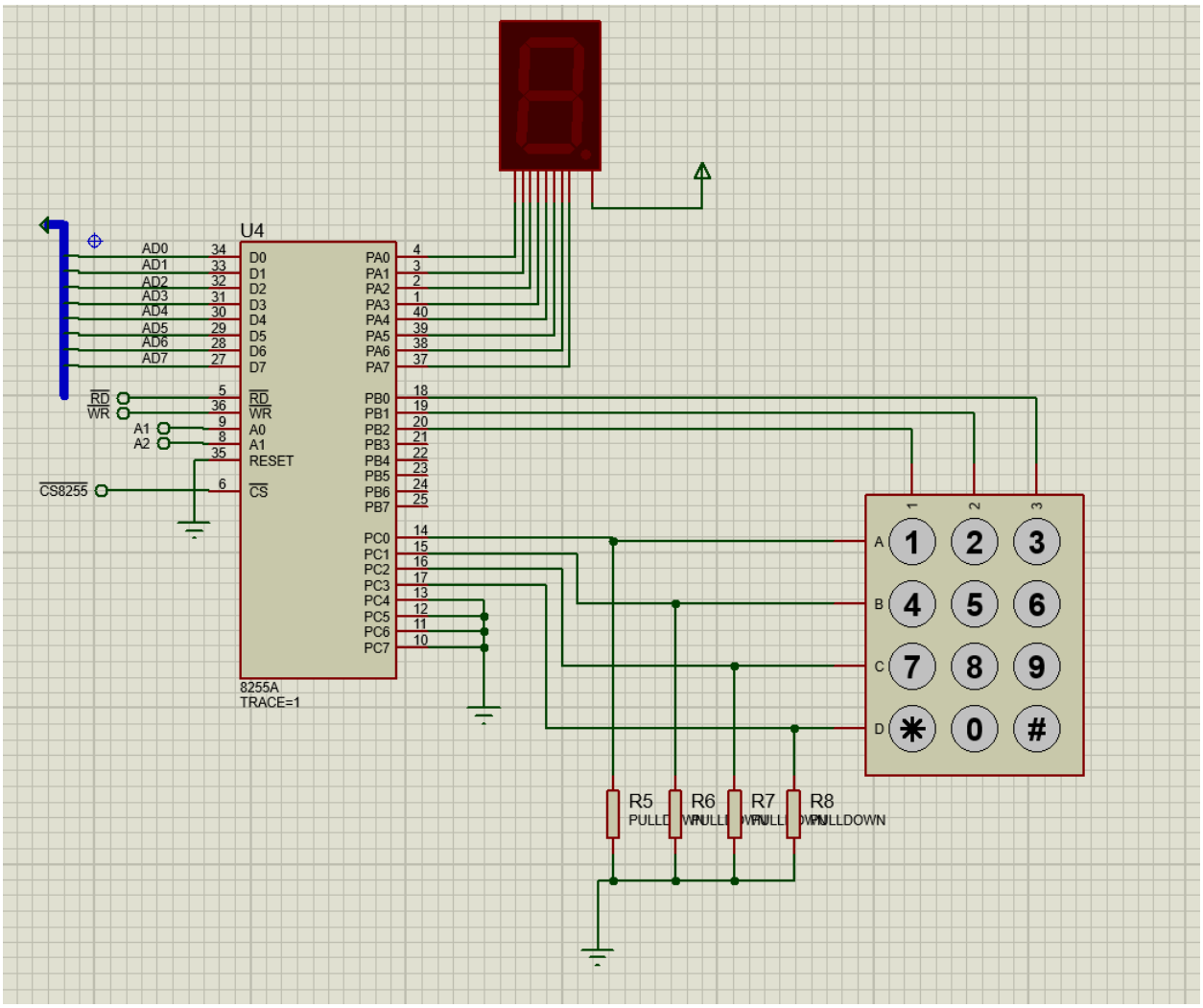
شکل 3.9 : اتصال صفحه کلید به 8255

3.5. نمایشگر صحفه

برای اتصال نمایشگر صحفه نیز مثل صحفه کلید عمل میکنیم، خطوط نمایشگر صحفه را به portA تراشه 8255 که آدرس آن 0A8H است وصل میکنیم و مقادیر 0C7H برای نمایش "L" و 0C1H برای نمایش "U" استفاده شده میکنیم.



شکل 3.11 : اتصالات بین صفحه نمایش و 8255

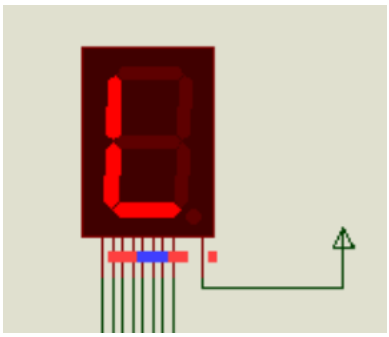


شکل 3.12 : اتصالات تراشه 8255

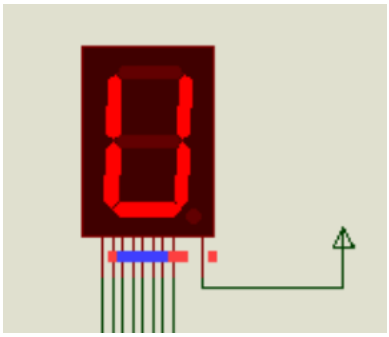
4. عملکرد و راهنمای کاربر

در این بخش عملکرد قفل صندوق ساخته شده را با برسی می کنیم. در ابتدا با اجرای پروژه ، به صورت پیش فرض قفل است و برای باز شدن کد دائمی[13] را وارد کرده و دکمه # را میزنیم .(به صورت پیش فرض ۱۲۳۴)

[13] masterkey



شکل 4.1 : صفحه قفل



شکل 4.2 : صفحه باز

بعد از باز شدن یک رمز چهار رقمی وارد میکنیم و صندوق خودکار قفل میشود.

پس از قفل شدن ، میتوان از رمز وارد شده یا رمز دائمی برای باز کردن قفل استفاده کرد.

در صورت اشتباه وارد کردن رمز می‌توان دکمه * را فشار داد تا صحفه کلید reset شود.

مشکلات مواجه شده و روش حل آن

کامپایل نشدن کد و ارور Not match: code ends :

در این مورد میتوان از کامپایلر برنامه پروتئوس استفاده کرد یا جای سگمنت CODE را تغییر داد

فعال نشد خط CS8255 :

استفاده از گیت Or اشتباه در نتیجه خروجی نامناسب

تغییر نکردن masterkey حتی با تغییر آن در کد پروژه :

در این مورد بعد از تغییر masterkey باید کد پروژه روا از اول کامپایل کنید

دسترسی به پروژه

فایل های پروژه شامل, فایل پرتئوس, کدهای اسمبلی, مستندات فایل و سایر پیوست ها در گیت هاب erph82 قرار دارد.

برای دسترسی به آن ها می توانید روی این لینک در گیت هاب به آدرس https://github.com/erph82/vault-8086 وارد شوید