دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

دستورکار آزمایشگاه معماری کامپیوتر

نسخه ۱.۲

دکتر علی بهلولی

بهار ۱۳۹۳

تاریخچه بازبینی ها

شماره نسخه	تاریخ	شرح تغییرات
۱.۰	۱۳۹۲/۰۶/۳۰	نسخه اولیه
۱.۱	۱۳۹۲/۱۰/۳۰	اصلاحات نگارشی
۱.۲	۱۳۹۳/۰۳/۳۰	انجام تغییرات جزئی و جابجایی مطالب در آزمایش پنجم تا آزمایش دهم

پیشگفتار

هدف از آزمایشگاه معماری، آشنایی عملی دانشجویان با طرز کار یک کامپیوتر است. در سال‌های گذشته برای پیاده‌سازی یک کامپیوتر در آزمایشگاه، از تراشه‌هایی با بسته‌بندی دیپ و از برد بورد استفاده می‌شد. با توجه به منسوخ شدن تراشه‌های دیپ و دشواری‌های بستن یک کامپیوتر با تمام اجزاء آن روی بردبورد، تصمیم گرفته شد دستور کار این آزمایشگاه در دو قسمت سازماندهی شود.

در قسمت اول، اصول طراحی بوردهای مدارچاپی (به عنوان جایگزینی برای برد بورد) پرداخته می‌شود. این قسمت شامل پنج آزمایش می‌باشد. در طی این پنج آزمایش، دانشجو به صورت کامل از الف تا ی طراحی و ساخت بورد مدارچاپی را آموزش می‌بیند

در قسمت دوم آزمایشگاه، کامپیوتر پایه که در فصل پنجم کتاب موریس مانو ارائه شده است با استفاده از زبان‌های توصیف سخت افزار طراحی و روی بورد FPGA آزمایش می‌شود. آزمایش‌های این قسمت از دستورکار به نحوی سازماندهی شده‌اند که دانشجو بتواند بعد از انجام پنج آزمایش، کامپیوتر را به صورت کامل روی بورد FPGA به صورت عملی پیاده‌سازی کند و اجرای دستورات را مشاهده کند. در انتهای آزمایشگاه، دانشجو قادر خواهد بود یک بورد FPGA طراحی و از آن برای پیاده‌سازی هر نوع پردازنده‌ای استفاده کند.

برای هر یک از آزمایشها مطالبی که اطلاع قبلی از آنها برای انجام آن آزمایش ضروری بوده است، تحت عنوان "پیش آگاهی" تدوین و قبل از دستور کار آن آزمایش ارائه شده است. هر آزمایش شامل گامهایی است که با نظر مربی درس آزمایشگاه می تواند به تناسب کم یا زیاد شده و یا تغییر داده شود.

مقررات این آزمایشگاه نیز مطابق مقررات عمومی آزمایشگاههای دانشگاه است. جلسات آزمایشگاه بلافاصله بعد از حذف و اضافه شروع شده و به صورت هفتگی تا شروع امتحانات پایان ترم ادامه می یابد. دانشجویی که سه جلسه غیبت داشته باشد، مردود است.

روند کلی آزمایشگاه به این صورت است که هر یک از دانشجویان باید پیش‌آگاهی آزمایش هفتهٔ بعد را از قبل مطالعه کرده، در ابتدای جلسه، در یک امتحان کوچک ۱۵-۱۰ دقیقه‌ای در مورد آن پیش‌آگاهی تسلط خود را نشان دهد (این کار به خاطر هدایت دانشجویان به مطالعهٔ قبلی مطالب و تقلیل احتمال تقلب در پیش گزارشها در نظر گرفته شده است). پس از امتحان کوچک، دانشجویان هر آزمایش (که علی القاعده ۲ یا ۳ نفر هستند)، گامهای مشخص شده در دستور کار را دنبال کرده، نتایج را در کاربرگهای آن آزمایش (که توسط مربی در محل آزمایشگاه در اختیار دانشجو گذاشته می شود) ثبت می‌کنند. به این ترتیب کاربرگها به صورت یک سری برای هر گروه کامل می‌گردد.

در ارزیابی نهایی هر دانشجو، علاوه بر نمرهٔ امتحانات کوچک هر جلسه و نمرهٔ گزارش جلسات (که برای تمام اعضای هر گروه یکسان است)، نظر مربی در مورد آن دانشجو و نتیجهٔ امتحان پایان ترم دانشجو نیز دخالت دارد.

در انتها از تمام همکاران و دانشجویان درخواست می‌کنم با طرح پیشنهادات مشخص و مدون خود برای رفع معایب، ارتقای کیفی و پویایی به عنوان یک ویژگی ضروریِ آزمایشگاه معماری کامپیوتر، ادامه دهندهٔ راهی باشند که اولین گام آن برداشته شده است. لازم می‌دانم که از سرپرست محترم دانشکده مهندسی کامپیوتر جناب آقای دکتر شهرام اعتمادی و همچنین کارشناس محترم آزمایشگاه جناب آقای مهندس محمد علی آزادی که در تجهیز نمودن و به روز رسانی این آزمایشگاه از هیچگونه همکاری دریغ نفرمودند کمال تشکر را داشته باشم.

دکتر علی بهلولی

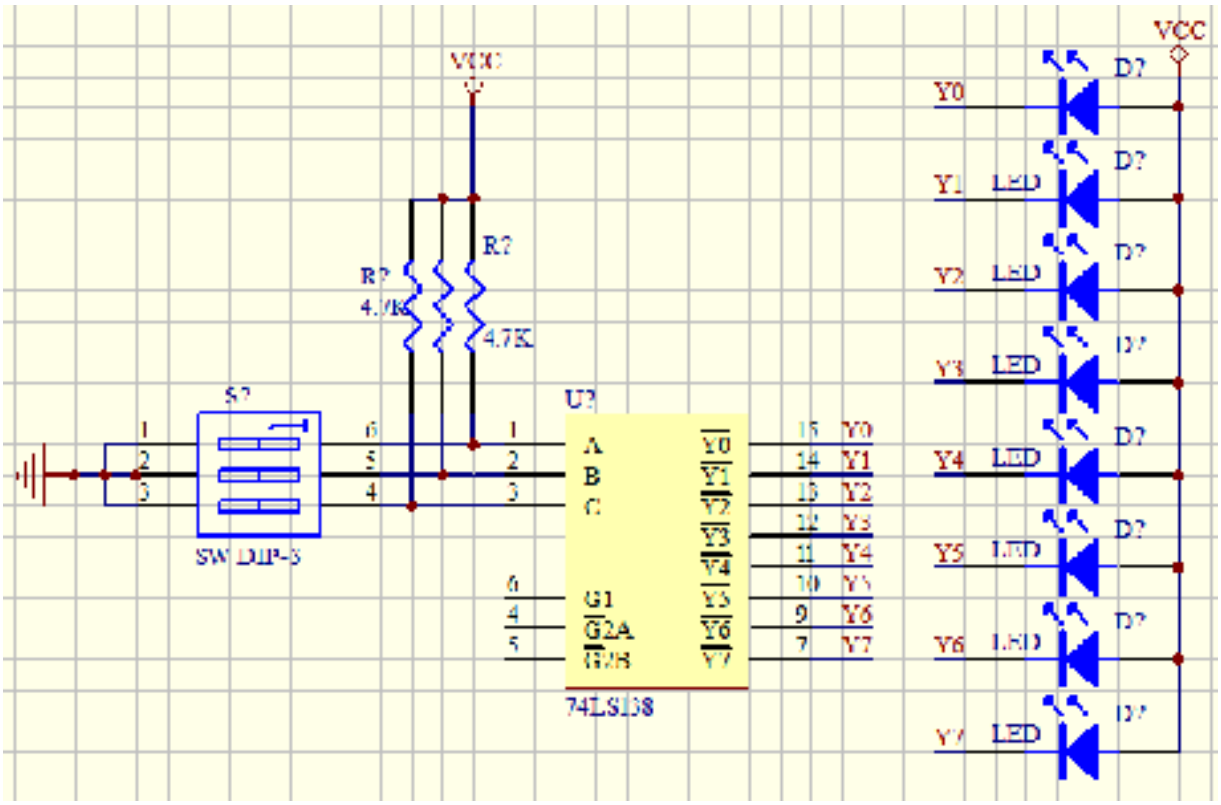
پاییز ۱۳۹۲

فهرست مطالب

۱ آزمایش اول: رسم شماتیک در نرم افزار پروتل

۱-۱پیش آگاهی

نرم افزار پروتل، یک نرم افزار تخصصی برای طراحی بورد مدار چاپی می‌باشد. برای تهیه بورد مدار چاپی یک مدار، باید در گام اول شماتیک مدار را رسم کرد. هدف این آزمایش رسم شماتیک یک مدار برای تست یک دیکدر می‌باشد. برای این منظور از دیکدر 74138 و هشت LED استفاده شده است در شکل ۱-۱، شماتیک مدار مورد نظر نمایش داده شده است.



شکل ۱-۱: شماتیک یک مدار برای تست دیکدر 74138

۱-۲پیش گزارش

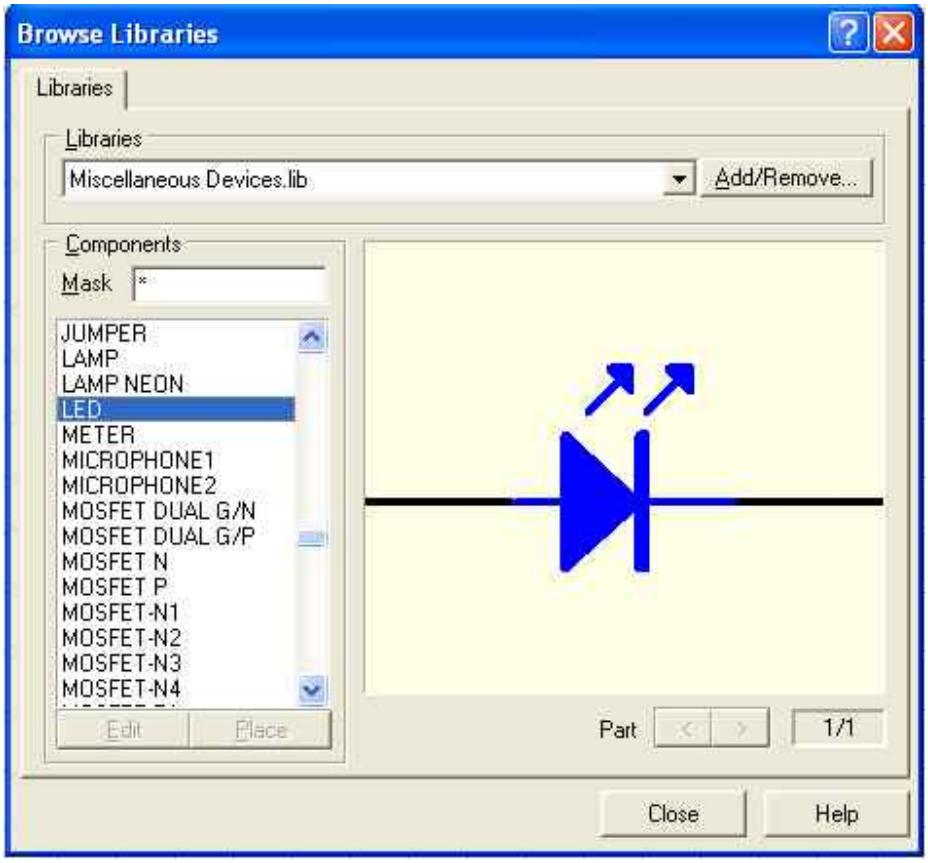
نرم افزار پروتل را طبق توضیحات داده شده در کلاس نصب کنید سپس با استفاده از آن، شماتیک نمایش داده شده در شکل ۱-۱، را رسم کنید و فایل آن را برای تکمیل کردن آن همراه خود بیاورید.

راهنمایی:

الف) برای یافتن تراشه 74138، از منوی Tools گزینه Find Components را انتخاب کنید و اسم تراشه را جستجو کنید(در جستجو می توان از علامت * نیز استفاده کرد، مثلا 138*74)

ب) شماتیک LED و سوئیچ در کتابخانه Miscellaneous Devices موجود است. برای یافتن آنها از منوی Place گزینه Part را انتخاب کنید و مطابق شکل ۱-۲، این دو قطعه را بیابید و در صفحه شماتیک قرار دهید.

ج) برای اینکه یک قطعه را در صفحه شماتیک دوارن دهید باید روی آن کلیک کنید و همزمان که کلید موس فشرده است، کلید Space را فشار دهید (با هربار فشردن کلید Space، قطعه به اندازه ۹۰ درجه دوران می یابد)



۲ آزمایش دوم: کتابخانه شماتیک در پروتل و ایجاد یک قطعه جدید

۲-۱پیش آگاهی

در نرم افزار پروتل امکان ساخت کتابخانه برای شماتیک وجود دارد. در کتابخانه می‌توان قطعات استفاده شده در شماتیک را ویرایش کرد و همچنین قطعات جدیدی را تعریف کرد. برای ایجاد کتابخانه شماتیک باید از منوی Design گزینه Make Project Library را انتخاب کنید. با انجام این کار یک فایل با پسوند lib به فولدر Document اضافه می‌شود. برای دیدن کتابخانه و قطعات آن باید در پنجره سمت چپ برگه Browse SchLib را انتخاب کنید.(در شکل ۲-۱، این برگه با حرف A علامت زده شده است) در ادامه به توضیح قسمتهای مختلف این پنجره پرداخته می‌شود.

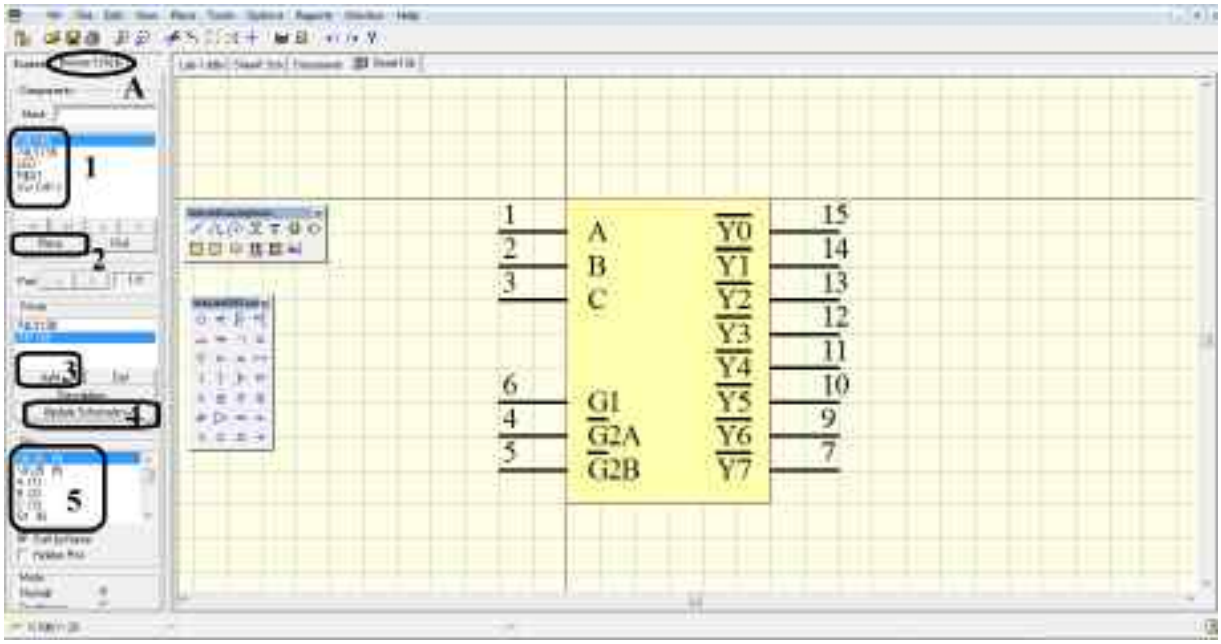
۱: این قسمت لیست تمام قطعاتی که در شماتیک شما مورد استفاده قرار گرفته و هم اکنون در کتابخانه تولید شده موجود است را نمایش می‌دهد

۲: دکمه Place، برای قرار دادن قطعه فعال فعلی (قطعه‌ای که در قسمت ۱، انتخاب شده است) در صفحه شماتیک می‌باشد

۳: برای تولید یک قطعه جدید و اضافه کردن آن به کتابخانه از دکمه add استفاده می‌شود. (برای تولید قطعه جدید بهتر است به جای استفاده از این گزینه، از منوی Tools گزینه New Component انتخاب شود)

۴: اگر قطعه‌ای را ویرایش کنید، برای اعمال تغییرات جدید در فایل شماتیک از این دکمه استفاده می‌شود

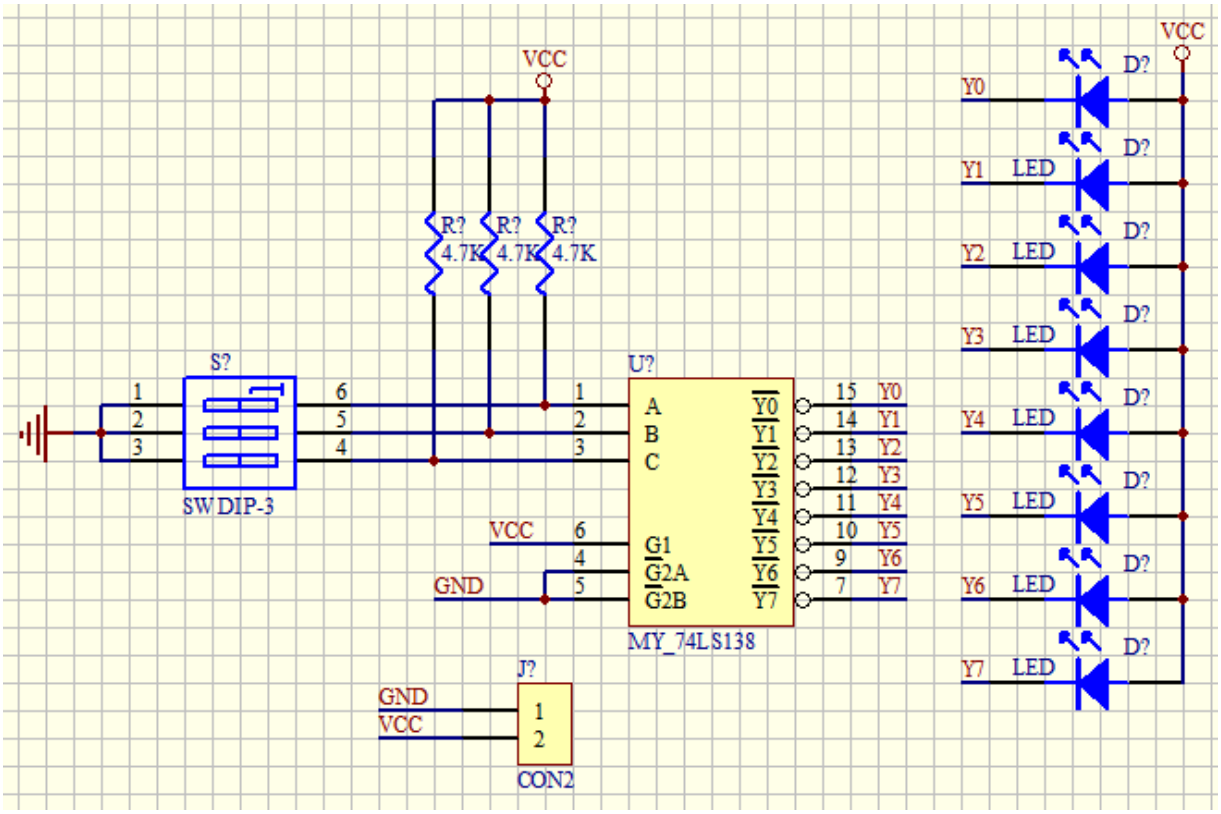
۵: لیست پایه‌های قطعه انتخاب شده را نشان می‌دهد. از طریق این لیست می‌توانید با پایه‌های مخفی نیز دسترسی داشته باشید



شکل ۲-۱:قسمتهای مختلف کتابخانه شماتیک در نرم افزار پروتل

۲-۲پیش گزارش

۱- مدار شماتیک نمایش داده شده در شکل ۲-۲ را رسم کنید. به تفاوتهای این مدار با مدار آزمایش ۱ دقت کنید(وصل شدن پایه‌های فعال ساز دیکدر، دایره‌های موجود در خروجی دیکدر، کانکتور تغذیه، مقدار مقاومتهای PullUp، تغییر Designatorهای قطعات)



شکل ۲-۲: شماتیک یک مدار برای تست دیکدر 74138

۲- با استفاده از قابلیت ایجاد قطعه جدید در نرم افزار پروتل، شماتیک قطعه نمایش داده شده در شکل ۲-۳ را رسم کنید.

۳- در شکل ۲-۴، یک تراشه نمایش داده شده است که دارای چند قطعه داخلی است. شماتیک این تراشه را به صورت یک قطعه با ۴ عدد Part طراحی کنید.

۴- با استفاده از پروتل به سوالات زیر در مورد تراشه با شماره 4012 پاسخ دهید:

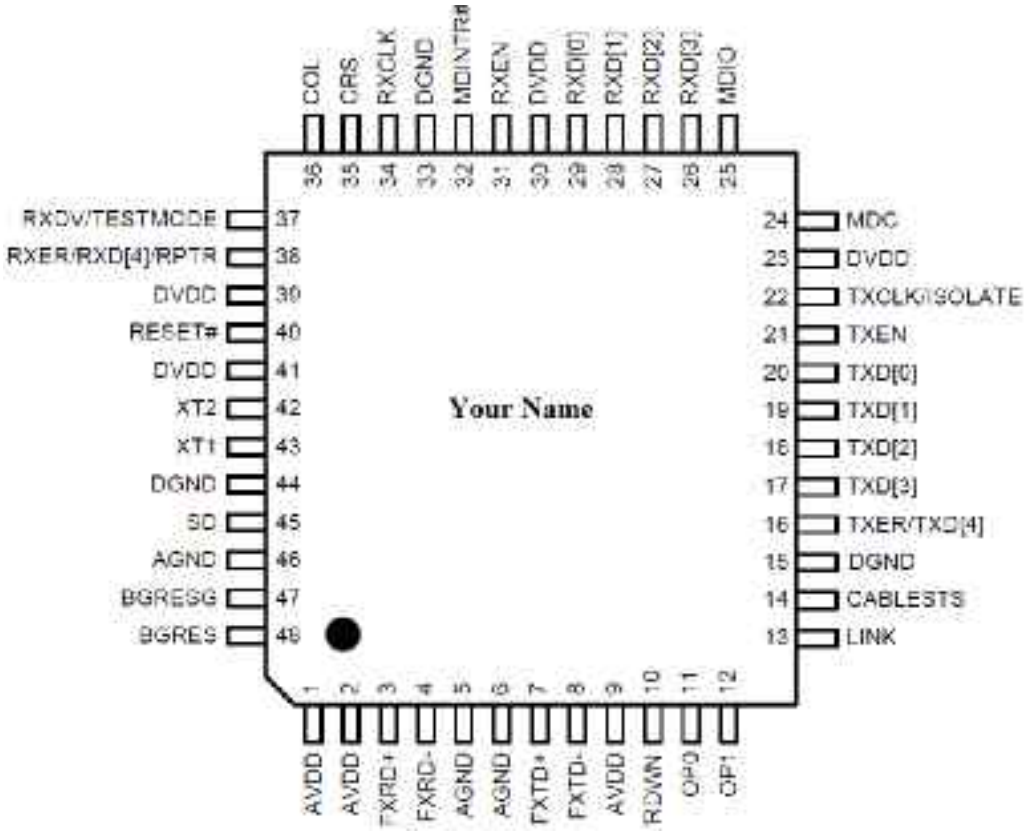
الف) عملکرد این تراشه چیست؟

ب) این تراشه چند پایه دارد؟

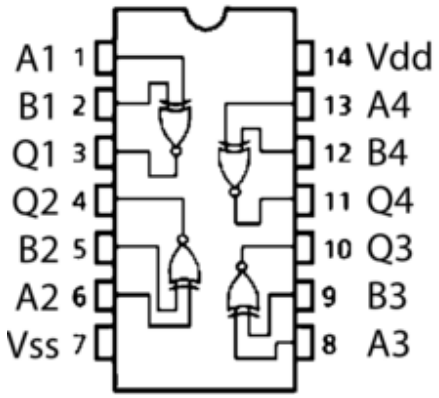
ج) شماتیک این تراشه حاوی چند Part است؟

د) شماره پایه تغذیه این تراشه چند است؟

ه) نام پایه‌های تغذیه این تراشه چیست؟



شکل ۲-۳: یک تراشه ۴۸ پایه



شکل ۲-۴: تراشه‌ای با چهار گیت xnor

۳ آزمایش سوم: آماده‌سازی شماتیک برای انتقال به PCB

۳-۱پیش آگاهی

در جلسات قبل نحوه رسم شماتیک و نحوه تعریف قطعه جدید و کار با کتابخانه شماتیک انجام شد. هدف از این آزمایش، نحوه آماده‌سازی شماتیک برای انتقال آن به مرحله رسم PCB است.

بعد از اینکه شماتیک مدار تکمیل گردید مراحل زیر باید انجام گردد:

۱- Annotate کردن شماتیک

۲- تعیین footprint برای تمام قطعات بکار رفته در شماتیک

۳- چک کردن مدار از لحاظ الکتریکی (ERC)

۴- انتخاب گزینه ... Update PCB از منوی Design

در ادامه به شرح موارد فوق پرداخته می‌شود.

۳-۱-۱ مفهوم واژه Annotate

هر قطعه‌ای که در صفحه شماتیک قرار داده می‌شود دارای یک فیلد به نام Designator است این فیلد معمولا شامل یک حرف و یک علامت سوال است (مثلا U? یا D? یا J? و ..) حرف اول نشان دهنده نوع قطعه است. در جدول ۳-۱ برخی از Designatorهای معرف آورده شده است.

جدول ۳-۱: برخی از Designatorهای معروف

Designator.	نوع قطعه
U?	تراشه (Unit)
R?	مقاومت
C?	خازن
L?	سلف
D?	دیود
J?	جامپر
S?	سوئیچ

با انتخاب گزینه ... Annotate از منوی Tools، نرم افزار پروتل به صورت اتوماتیک علامتهای " ?" را به شماره تبدیل می‌کند. به عنوان مثال اگر در شماتیک از ۱۰۰ خازن استفاده شده باشد. Designator این خازنها به ترتیب C1 تا C100 مقداردهی خواهند شد.(توجه داشته باشید که وجود Designator تکراری در شماتیک مجاز نیست)

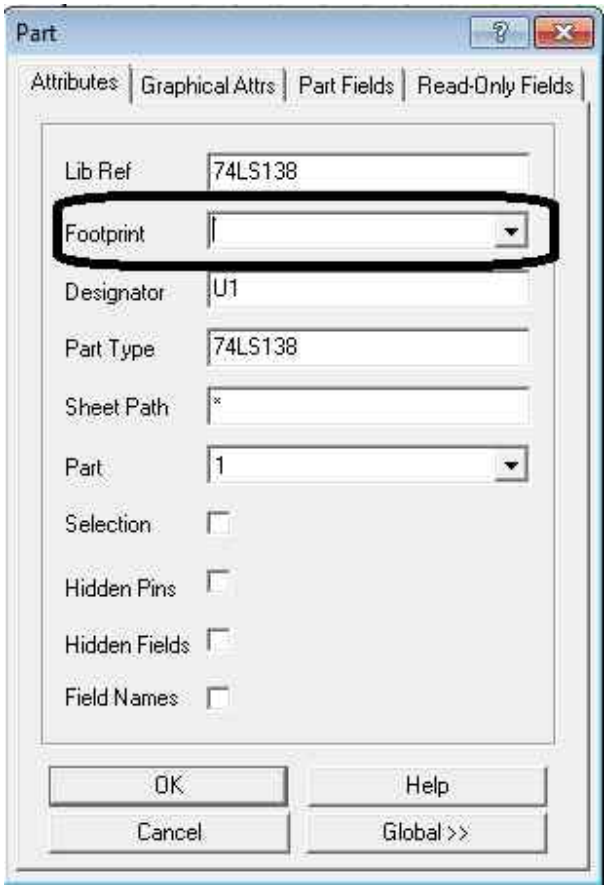
۳-۱-۲ تعیین FootPrint قطعات

شماتیک قطعات حاوی اطلاعاتی شامل تعداد پایه‌های ورودی/خروجی، شماره پایه و نام پایه می‌باشد و نشان دهنده شکل واقعی قطعه و نوع پکیج آن نمی‌باشد. این در حالی است که هنگام رسم PCB، شکل فیزیکی قطعه مهم است و اینکه فاصله پایه‌ها، ابعاد قطعه چقدر است. بنابراین مکانیزمی باید وجود داشته باشد که نشان دهد هر قطعه ای که در شماتیک قرار دارد شکل فیزیکی آن به چه صورتی است. پارامتر Footprint هر قطعه این مساله را حل می‌کند و مشخص می‌کند که شکل فیزیکی قطعه نهایتا روی برد به چه صورتی خواهد بود. برخی از footprintهای معروف در جدول ۳-۲ نمایش داده شده است.

جدول ۳-۲: برخی از footprintهای معروف

نام footprint	توضیح
DIP14	تراشه با پایه‌های سوزنی دوردیفه (تعداد کل پایه‌ها ۱۴ بوده است) تعداد پایه‌ها از ۴ تا ۶۴ می‌تواند باشد
SIP10	قطعه با پایه‌های سوزنی تک ردیفه(تعداد پایه‌ها از ۲ تا ۲۰ می‌تواند باشد)
AXIAL0.4	مقاومت که فاصله بین دو پایه آن 0.4 inch است
RB.2/.4	خازن الکترولیتی با شعاع 0.4 و فاصله پایه 0.2 اینچ
RAD0.4	خازن عدسی که فاصله بین دو پایه آن 0.4inch است
DIODE0.4	دیودی که فاصله بین دو پایه آن 0.4inch است

برای مشاهده و تعیین Footprint یک قطعه، می‌بایست روی قطعه دابل کلیک کنید. در شکل ۳-۱، نتیجه دابل کلیک کردن و محل فیلد footprint نمایش داده شده است.



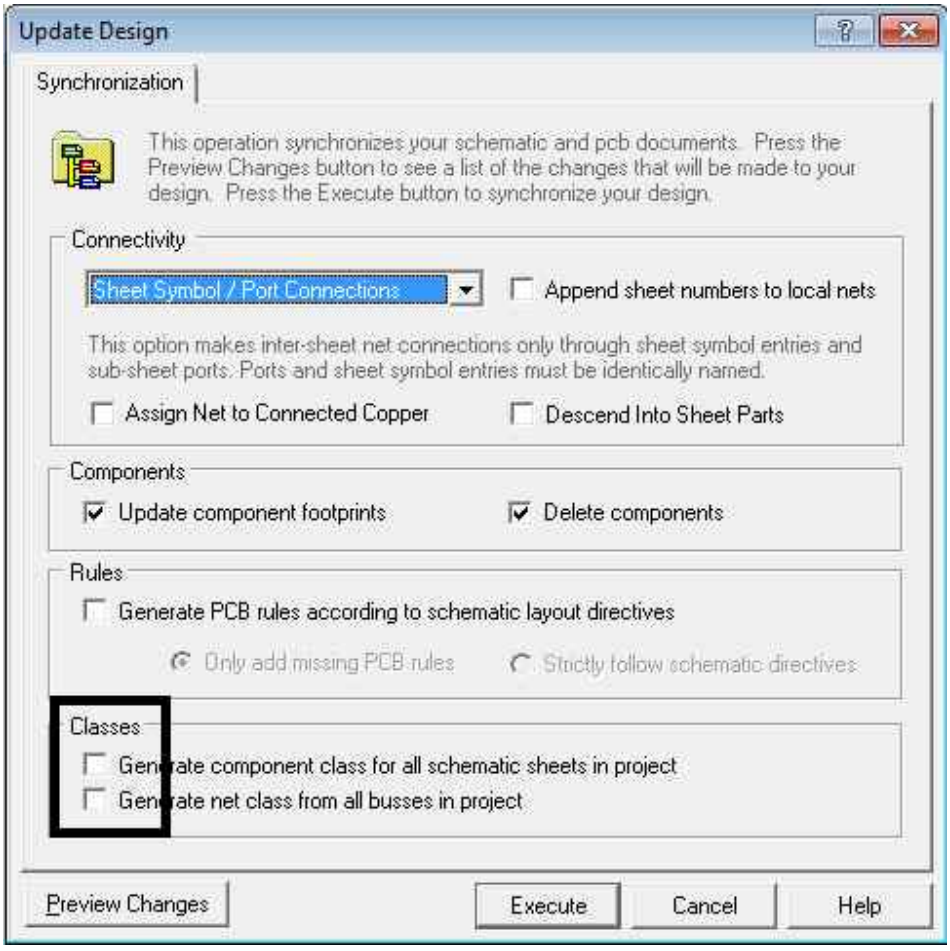
شکل ۳-۱:نتیجه دابل کلیک کردن روی هر قطعه و مکان فیلد footprint

۳-۱-۳چک کردن مدار از لحاظ الکتریکی (ERC)

بعد از انجام مراحل فوق، باید گزینه ERC… را از منوی Tools انتخاب کنید و در پنجره ظاهر شده پارامترهایی را که قصد چک کردن آن را دارید را تیک بزنید و نهایتا دکمه Ok را بزنید. نتیجه اجرای ERC به صورت یک فایل متنی نمایش داده می‌شود و خطاها لیست می‌شوند. محل خطاها نیز در شماتیک علامتگذاری می‌شوند.

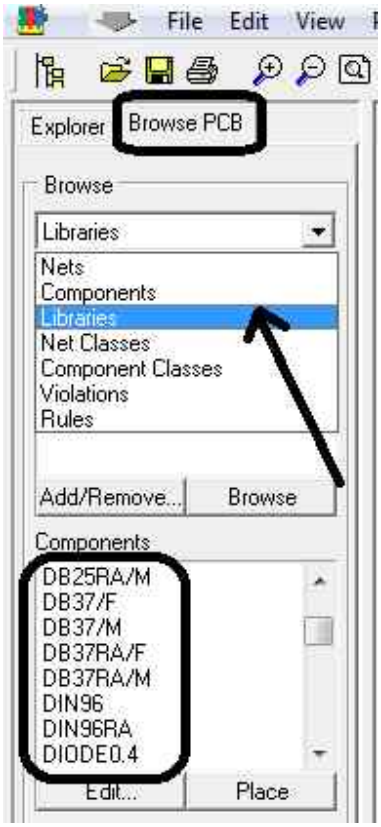
۳-۱-۴ انتخاب گزینه Update PCB از منوی Design

هرگاه مراحل سه گانه فوق انجام شد و هیچ خطایی در شماتیک موجود نبود با انتخاب گزینه … Update PCB از منوی Design می‌توانید وارد محیط PCB شوید در شکل ۳-۲، نتیجه اجرای این گزینه نمایش داده شده است. (اگر برای اولین بار باشد که این گزینه را انتخاب می‌کنید، نرم افزار پروتل به صورت اتوماتیک برای شما یک فایل با پسوند pcb تولید می‌کند. در دفعات بعدی فقط عمل Update شدن PCB انجام خواهد شد)



شکل ۳-۲: نتیجه انتخاب گزینه ... Update PCB از منوی Design

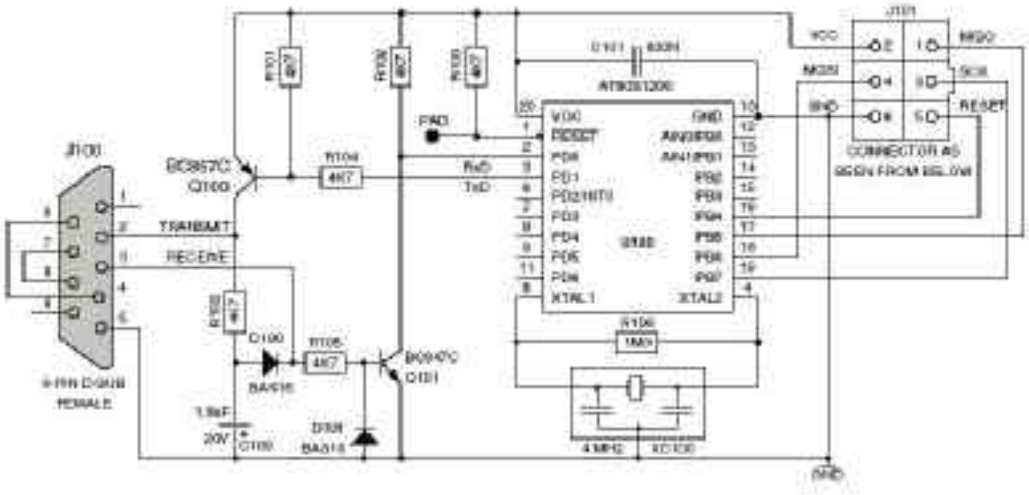
در پنجره ظاهر شده تیکهای مربوط به Class را بردارید تا از شلوغ شدن PCB جلوگیری شود. با کلیک کردن دکمه Preview Changes می‌توانید لیست تغییرات را مشاهده کنید. همچنین اگر اشکالی در footprintها وجود داشته باشد در این قسمت قابل مشاهده خواهد بود(توجه داشته باشید که تمام خطاهای نمایش داده شده باید رفع گردد در غیر اینصورت قطعه‌های خطا دار در PCB ظاهر نخواهند شد یا اینکه برخی از اتصالات قطعات نمایش داده نمی‌شود که هر دو مورد منجر به ایجاد خطا در بورد نهایی خواهد شد). با کلیک روی دکمه Execute، انتقال به محیط PCB انجام خواهد شد و صفحه PCB نمایش داده خواهد شد. در صفحه باز شده که دارای پس زمینه سیاه رنگ است قطعات شماتیک بر اساس اتصالات تعیین شده و footprintهای تعیین شده ظاهر شده اند. در قسمت پایین این پنجره امکان انتخاب لایه‌های مختلف PCB وجود دارد. در پنجره عمودی سمت چپ (قسمتی که لیست فایلها و فولدرها را نمایش میدهد) اگر برگه Browse PCB را انتخاب کنید (شکل ۳-۳) می‌توانید کتابخانه‌های footprint و لیست آنها را مشاهده کنید و در صورت نیاز footprintهای شماتیک خود را تغییر دهید.



شکل ۳-۳: نحوه مشاهده footprintها و کتابخانه آن (گزینه Browse PCB فقط زمانی وجود دارد که در قسمت PCBباشید و در شماتیک این گزینه وجود ندارد)

۳-۲پیش گزارش

۱- برای مدار نمایش داده شده در شکل ۳-۴، مراحل لازم برای انتقال به PCB را اجرا کنید. (۴ مرحله ذکر شده در بخش ۳-۱)



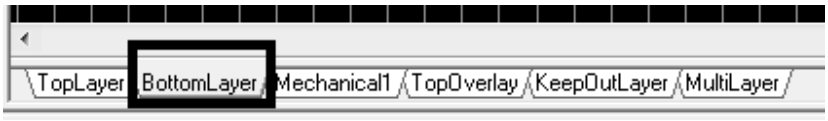
شکل ۳-۴: شماتیک یک مدار برای پراگرام کردن تراشه های شرکت Atmel

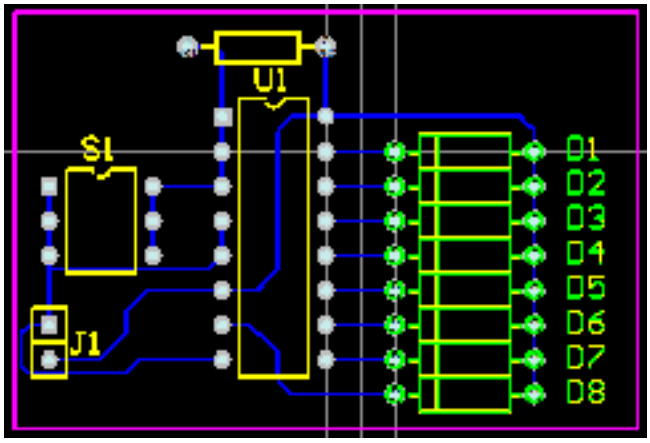
۲- با استفاده از کتابخانه footprint در قسمت PCB، footprintهای مناسب برای قطعات زیر را بیابید.

۳-۳ دستور کار (این قسمت در آزمایشگاه انجام خواهد شد)

۱- بعد از انتقال مدار به PCB، قطعات را بچینید و تمام تراکهای (Track) مورد نیاز را رسم کنید (برد شما تک لایه و فقط از لایه زیر استفاده کنید).

راهنمایی: ابتدا در قسمت پایینی صفحه، برگه BottomLayer را انتخاب کنید. سپس با زدن کلیدP و سپس کلید T به مود رسم تراک وارد شوید و شروع به رسم تراکها در لایه زیر کنید . رنگ پیش فرض برای لایه زیر در نرم افزار پروتل، آبی است





شکل ۳-۵: نمونه یک مدار PCB، رسم شده

۲- کادر صورتی رنگ دور مدار را در لایه KeepOutLayer رسم کنید. تحقیق کنید که این لایه چه کاربردی دارد؟

۳- با استفاده از قابلیت انجام تغییرات به صورت Global در نرم افزار پروتل کارهای زیر را انجام دهید

الف) ضخامت همه تراکها را به 30 mil تغییر دهید (نتیجه را در یک فایل PCB مجزا ذخیره کنید . برای این منظور از منوی File گزینه ...New را انتخاب و از لیست نمایش داده شده گزینه PCB Document را انتخاب کنید سپس با استفاده از قابلیت سلکت و کپی و پیست مدار PCB قبلی را به داکیومنت جدید منتقل کنید)

ب) همه تراکها را از لایه زیر به لایه رو، بیاورید (نتیجه را در یک فایل PCB مجزا ذخیره کنید)

۴- قسمتهایی که به رنگ زرد هستند در چه لایه ای می‌باشند؟

۴ آزمایش چهارم: آشنایی با PCB و مفاهیم آن

۴-۱پیش آگاهی

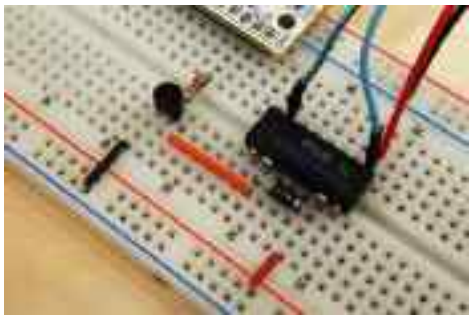
در جلسه قبل نحوه انتقال شماتیک به PCB و رسم PCB انجام شد. در این جلسه نکاتی در مورد طراحی PCB ارائه خواهد شد سپس نحوه ایجاد کتابخانه PCB و طراحی footprint شرح داده خواهد شد.

۴-۱-۱ واحدهای اندازه گیری فاصله در PCB

رعایت صحیح فاصله‌ها در PCB، از اهمیت ویژه ای برخوردار است و ممکن است عدم دقت در این رابطه، منجر به سخت شدن عملیات مونتاژ قطعات و بعضی اوقات چاره ای جز طراحی مجدد PCB و ساخت بورد جدید نیست. مواردی که هنگام طراحی PCB باید به فاصله‌های آنها دقت کرد عبارتند از: ابعاد فیزیکی قطعات (فضایی که روی بورد اشغال می‌کنند)، فاصله بین پایه‌های قطعات، قطر سوراخ پایه‌ها روی بورد(برای تعیین نوع مته ای که برای سوراخکاری استفاده می‌شود)، ابعاد کل بورد، ضخامت تراک‌ها، فاصله بین تراکها(Clearance) و فاصله بین قطعات مختلف روی بوردبرخی از این فاصله‌ها را باید هنگام طراحی FootPrint قطعه‌ها لحاظ کرد (ابعاد فیزیکی قطعه، فاصله بین پایه‌ها، قطر سوراخ پایه‌ها) و موارد دیگر باید هنگام رسم PCB در نظر گرفته شود.

واحدهای اندازه گیری فاصله در پروتل عبارتند از mil و mm. یک میل برابر با ۱ هزارم اینچ می‌باشد. (این واحد به این خاطر استفاده می‌شود که اکثر سازندگان قطعات در ساخت قطعه از واحد اینچ استفاده می‌کنند) مثلا فاصله بین دو پایه متوالی در پکیج دیپ برابر با ۰.۱ اینچ می‌باشد. در مورد مته‌ها، عموما از واحد میلی متر استفاده می‌شود. بنابراین برای تعیین قطر سوراخهای بورد بهتر است از واحد mm استفاده شود. (هر میلی متر به صورت تقریبی برابر با 39.37mil است)

نکته کاربردی: فاصله بین دو سوراخ متوالی روی برد بوردها[1] دقیقا برابر با 100mil می‌باشد. این نکته می‌تواند به عنوان معیار مناسبی برای اندازه گیری فاصله بدون داشتن خط کش یا کولیس باشد. به عنوان مثال با توجه به شکل ۴-۱، می‌توان متوجه شد که فاصله پایه‌های تراشه نصب شده روی برد بورد برابر با 100mil و طول تراشه برابر با 800mil و عرض تراشه برابر با 300mil است. یا در footprint مربوط به مقاومت نصب شده روی برد بورد، فاصله بین پایه‌های مقاومت برابر با 300mil باید قرار داده شود و برای دیود 400mil است.



شکل ۴-۱: استفاده از برد بورد برای اندازه گیری فاصله قسمتهای مختلف یک تراشه

رعایت فاصله‌ها در رسم PCB

[1] BreadBoard

همانطور که گفته شد، ضخامت تراکهای رسم شده در PCB و همچنین حداقل فاصله بین دو تراک موازی از پارامترهای مهم در رسم PCB است و اندازه آنها معمولا بر حسب mil سنجیده می‌شود. دو عامل در تعیین این دو فاصله دخالت دارند که عبارتند از: محدودیتهای تکنولوژی ساخت بورد مدار چاپی و محدودیتهای که به نوع و ماهیت سیگنال عبوری از این تراک بستگی دارد

محدودیتهای ناشی از تکنولوژی ساخت بورد

این محدودیت به دقت دستگاه‌هایی بستگی دارد که بورد مدار چاپی را تولید می‌کنند. معمولا این دستگاه‌ها در حداقل ضخامت تراکها و حداقل فاصله بین دو تراک موازی محدودیت دارند و اگر این حداقل‌ها رعایت نشود منجر به قطع شدن تراکها یا به هم چسبیدن تراکها خواهد شد. در حال حاضر اکثر این دستگاه‌ها قادرند بوردهایی با تراکهایی با ضخامت 10mil و فاصله 10mil را تولید کنند. بهتر است قبل از شروع کشیدن PCB، با شرکتی که قرار است بورد را بسازد مشورتی انجام شود و از محدودیتهای دستگاه‌های آنها اطلاع حاصل شود.

محدودیتهای ناشی از ماهیت سیگنال عبوری از تراک

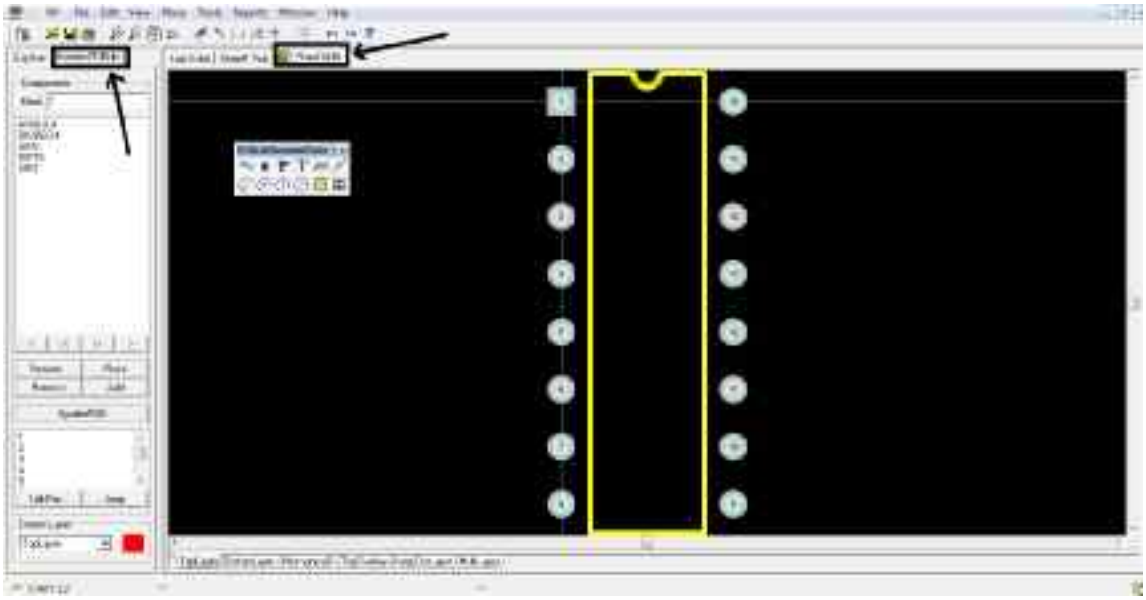
برای سیگنالهای تغذیه باید تراک با ضخامت زیاد رسم کرد (مثلا 30mil یا بیشتر). سیگنالهایی که فرکانس تغییرات آنها زیاد است (مثلا بیشتر از ۵۰۰ کیلو هرتز) اگر نزدیک همدیگر رسم شوند روی همدیگر اثر همشنوایی[2] خواهند داشت و باعث ایجاد خطا می‌شود برای کاهش اثر همشنوایی بهتر است فاصله بین تراکها بیشتر از 10mil باشد.

۴-۱-۲ ایجاد Footprint جدید

برای ایجاد Footprint جدید باید کتابخانه PCB برای پروژه خود ایجاد کنید. برای ایجاد کتابخانه PCB باید از منوی Design، گزینه Make Library را انتخاب کنید (توجه داشته باشید که باید در محیط رسم PCB باشید). در شکل ۴-۲، نحوه باز کردن فایل library و همچنین نحوه دیدن قطعات علامت زده شده است.

برای تعریف Footprintجدید باید از منوی Tools، گزینه New Component را انتخاب کنید (توجه باید داشته باشید که باید فایل Library باز باشد و در محیط کتابخانه باشید تا گزینه New Component در منوی Tools وجود داشته باشد). بعد از انتخاب گزینه فوق، ویزاردی باز می‌شود و در ابتدا باید انتخاب کنید که Footprintی را که قصد ایجاد آن را دارید شبیه کدامیک از footprintهای استاندارد است. بعد از انتخاب یکی از footprintهای استاندارد، در ادامه مشخصات قطعه و اندازه قسمتهای مختلف آن را می‌توانید روی شکل تغییر دهید و نهایتا footprint مورد نظر خود را ایجاد کنید. در آخرین مرحله، نام footprint را تایپ کنید و کلید finish را بزنید. برای تهیه footprint هر قطعه باید اطلاعاتی در مورد ابعاد قطعه، تعداد پایه‌ها، فاصله بین پایه‌ها، ضخامت هر پایه، طول هر پایه و ... داشته باشید. این اطلاعات را می‌توان با اندازه گیری قطعه واقعی بدست آورد یا اینکه از دیتا شیت قطعه استخراج کرد. معمولا در صفحات انتهایی دیتاشیت هر قطعه، ابعاد فیزیکی قطعه بر حسب اینچ و میلیمتر ارائه می‌شود.

[2] CrossTalk

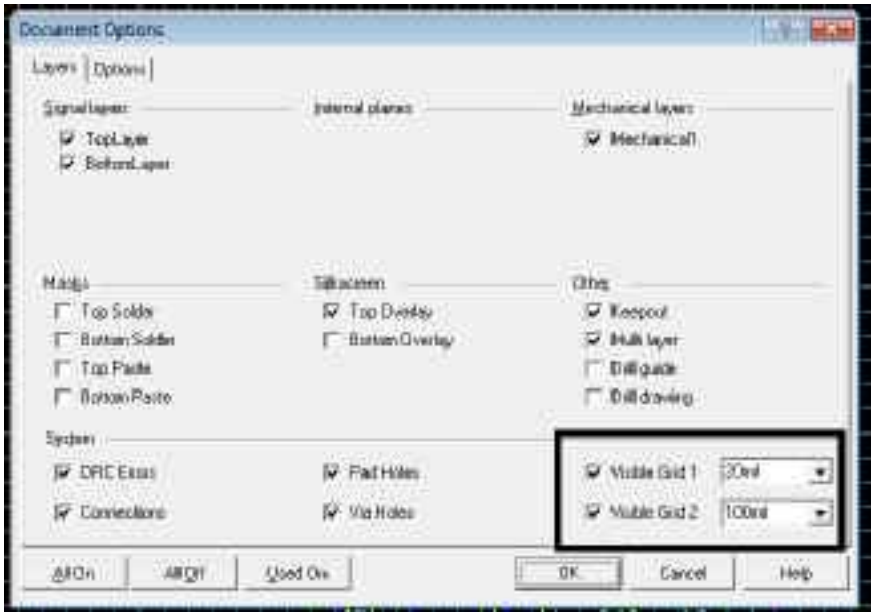


۴-۱-۳ مفهوم Grid

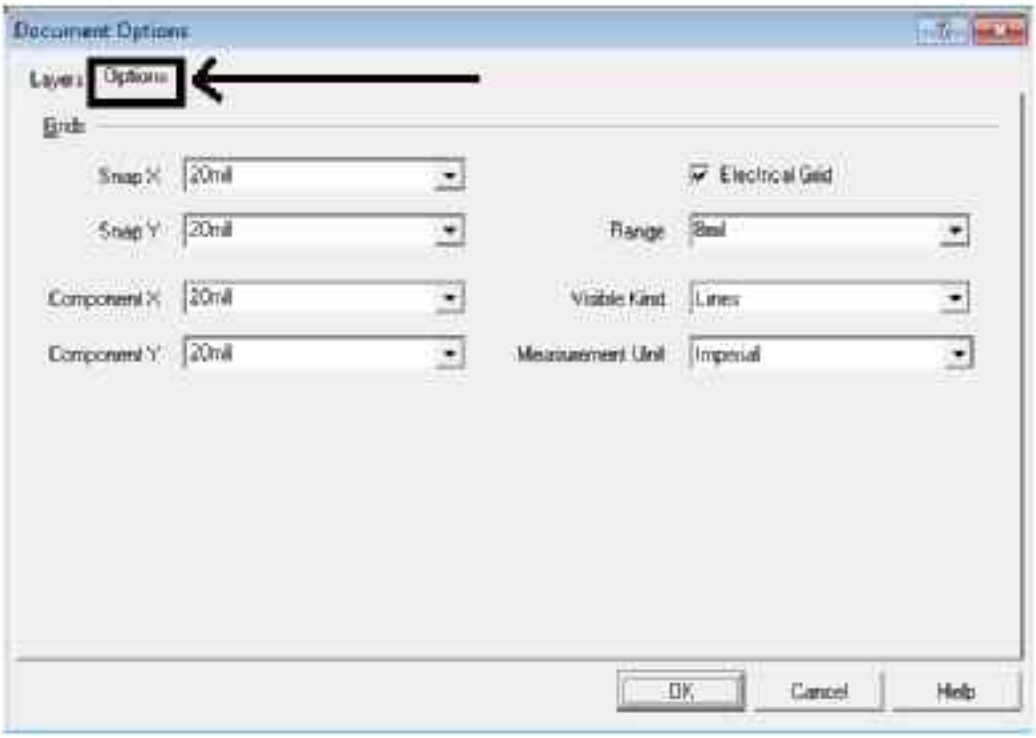
در نرم افزار پروتل به منظور کمک به طراح، صفحه رسم PCB به صورت شطرنجی می‌باشد. این خطوط شطرنجی به صورت دو سطحی است. طراح PCB با کمک گرید می‌تواند قطعات را با نظم مشخص روی صفحه بچیند و تراکها را روی خطوط گرید رسم کند تا شکل منظمی داشته باشند. برای تعیین اندازه Gridهای سطح ۱ و سطح ۲ باید گزینه ...Options در منوی Design انتخاب شود. در شکل ۴-۳، محل تعیین این دو پارامتر، مشخص شده است. همانطور که در این شکل ملاحظه می‌شود برای این دو سطح مقادیر 20mil و 100mil انتخاب شده است. وقتی که در صفحه رسم PCB با استفاده از دکمه‌های PgUp و PgDn روی مدار Zoom In یا Zoom Out می‌کنید، به صورت اتوماتیک نرم افزار پروتل، گرید مناسب را برای شما نمایش می‌دهد.

۴-۱-۴ مفهوم Snap

زمانی که در قسمت PCB از نرم افزار پروتل، قطعه ای را جابجا می‌کنید حرکت قطعه در جهت افقی و عمودی به صورت پله پله انجام خواهد شد و قطعه‌ها را نمیتوان به اندازه خیلی کم جابجا کرد. این خاصیت باعث می‌شود که نظم دادن به قطعه‌ها به راحتی امکان پذیر شود. مثلا برای اینکه پایه‌های دو قطعه دقیقا روبروی هم قرار گیرد نیازی به دقت زیاد کاربر ندارد و چون حرکتها به صورت پله ای است به راحتی می‌توان قطعات را روبروی هم دیگر و روی گریدها قرار داد. این مساله حرکت کردن قطعه‌ها به صورت پله ای، در مرحله رسم تراک نیز مشاهده می‌شود. در واقع هنگام رسم تراک، نرم افزار پروتل اجازه نمیدهد هر نوع زاویه ای به تراک بدهید و از هر مکانی که خواستید عبور دهید. برای تنظیم فاصله پله‌ها باید از منوی Design، گزینه ...Options را انتخاب کنید و در پنجره ظاهر شده، برگه Option را انتخاب کنید. در شکل ۴-۴، پنجره مربوط به تنظیم Snap نمایش داده شده است. همانطور که ملاحظه می‌شود مقدار Snap هم برای Component و هم برای تراکها قابل انتخاب است.



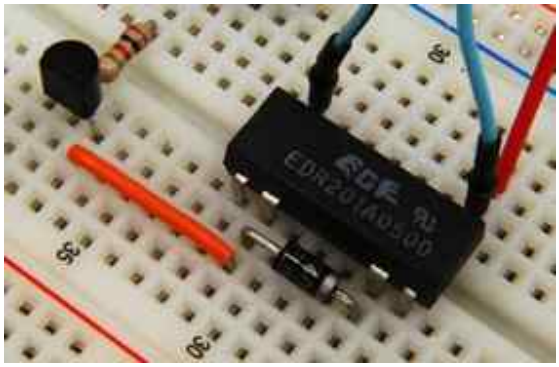
شکل ۴-۳: نحوه تعیین اندازه گریدهای سطح ۱ و سطح ۲ (با انتخاب گزینه ...Options از منوی Design، پنجره فوق ظاهر می‌شود)



شکل ۴-۴: نحوه تعیین Snap (با انتخاب گزینه ...Options از منوی Design، و انتخاب برگه Option پنجره فوق ظاهر می‌شود)

۴-۲پیش گزارش

۱- Footprintقطعات نشان داده شده روی شکل زیر را رسم کنید.



شکل ۴-۵: اندازه گیری فاصله ها با استفاده از بردبورد

۲- فایل lab4.ddb را باز کنید. این فایل حاوی شماتیک یک مدار می‌باشد. PCB این مدار را رسم کنید. در رسم PCB نکات زیر را رعایت کنید: (برای Placement قطعات از شکل ۴-۶ کمک بگیرید)

الف) ابعاد بورد ۱۲ سانتی متر در ۶ سانتیمتر باشد

ب)ضخامت تراکهای تغذیه را 30mil و ضخامت بقیه تراکها را 10mil در نظر بگیرید

ج) فقط از لایه زیر استفاده کنید

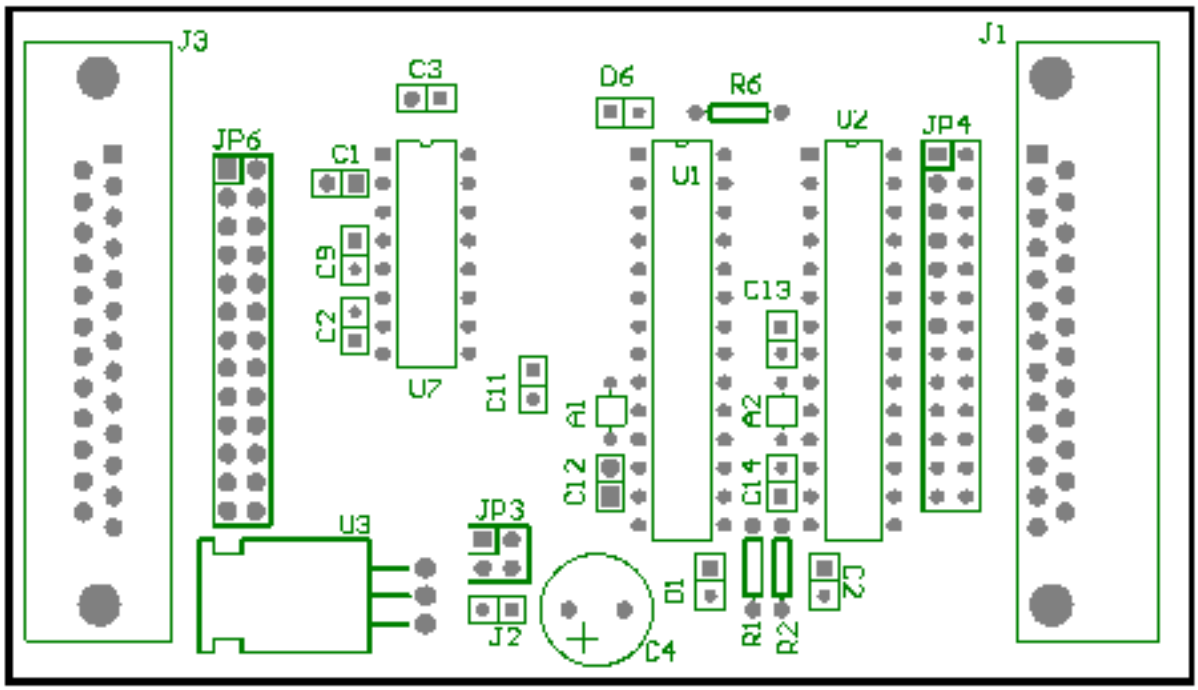
۳- اگر فاصلهٔ بین پایه‌های یک آی‌سی DIP، 100mil و قطر Pad آن 50mil باشد و بخواهیم دو تراک با پهنای 10mil از وسط پایه‌های آن رد کنیم به موارد زیر پاسخ دهید.

الف) مناسبترین اندازه برای Grid و Snap چند است؟ چرا؟

ب) ماکزیمم clearance باید چند باشد تا بتوان تراک را از بین پایه‌های آن رد کرد؟

کارهای انجام شده را در قالب یک فایل با پسوند ddb ایمیل کنید. موضوع ایمیل را Az-memari-H4 قرار دهید.

جواب سوال ۳ را روی کاغذ بنویسید و در آزمایشگاه تحویل دهید



شکل ۴-۶: نمونه پیشنهادی برای چینش قطعات

۵ آزمایش پنجم: تکمیل کردن PCB و آماده کردن آن برای ساخت

۵-۱پیش آگاهی

در جلسات قبل نحوه رسم PCB توضیح داده شد. در این جلسه قابلیتهای نرم افزار پروتل برای رسم PCB توضیح داده می‌شود.

۵-۱-۱مفهوم via

به سوراخهایی که روی بورد ایجاد می‌شود تا تراکها را از یک لایه به یک لایه دیگر منتقل کند via گفته می‌شود. در واقع با استفاده از via می‌توان لایه‌های مسی در لایه‌های مختلف را به یکدیگر متصل کرد. اگر در حین رسم تراک خواستید ادامه آن را در لایه دیگری رسم کنید باید از حالت کشید تراک خارج شوید سپس با زدن دکمه‌های P و V (Place Via) به حالت اضافه کردن via وارد شوید، با کلیک کردن موس روی تراک مورد نظرتان یک Via اضافه کنید، به صورت اتوماتیک نام via با نام تراک شما یکسان می‌شود. اگر نام تراک و نام via یکسان نباشد آنگاه امکان اتصال آنها به یکدیگر وجود ندارد

۵-۱-۲مفهوم بورد متالیزه

اگر داخل سوراخهای بورد (سوراخهای مربوط به پایه قطعات و سوراخهای مربوط به viaها) یک لایه فلزی گذاشته شود تا اتصال فیزیکی بین لایه‌ها برقرار شود آنگاه به بورد بورد متالیزه گفته می‌شود. توجه داشته باشید که این اصطلاح مربوط به نرم افزار پروتل نیست (در نرم افزار پروتل فرض بر این است که این لایه فلزی وجود دارد) این اصطلاح توسط شرکتهایی که عمل ساخت بورد PCB را انجام می‌دهد به کار می‌رود. بوردهای غیر متالیزه ارزان قیمت تر هستند و قرار دادن یک سیم مسی در سوراخها و لحیم کردن آن باید توسط شخص مونتاژ کار انجام شود.

۵-۱-۳رسم تراکها به صورت خودکار

برای رسم تراکها به صورت خودکار باید ruleهای مورد نیاز را تعریف کنید سپس با انتخاب گزینه ...All در منوی AutoRoute وارد پنجره Auto Route Setup شوید. این پنجره در شکل ۵-۱ نمایش داده شده است. استفاده از این گزینه فقط برای ساخت بوردهای نمونه، برای صرفه جویی در وقت استفاده میشود و توصیه می شود برای رسم PCB از این گزینه استفاده نشود.

شکل ۵-۱: نحوه انجام رسم تراکها به صورت خودکار

۵-۱-۴ اضافه کردن لایه

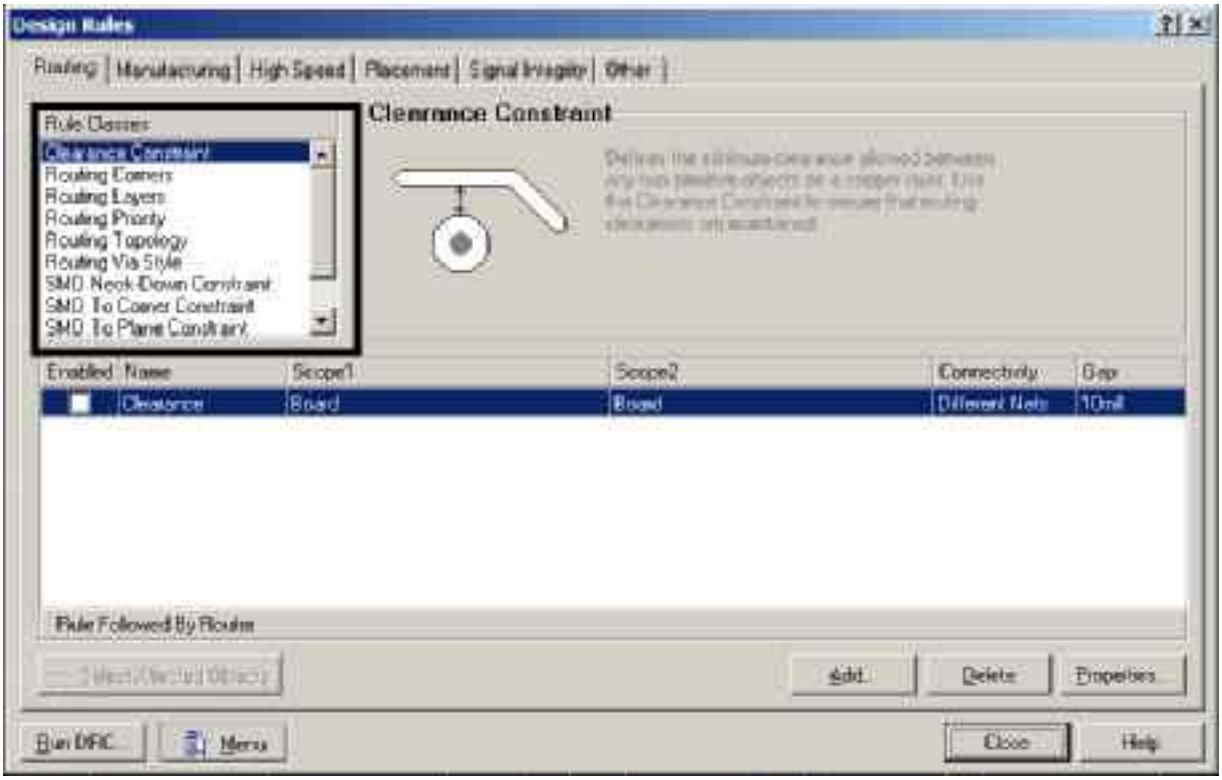
به صورت پیش فرض، تعداد لایه ها در نرم افزار پروتل دو لایه می باشد، لایه زیر و لایه رو. در صورتیکه نیاز به افزایش تعداد لایه ها باشد(معمولا برای بوردهایی با فرکانس کاری بالاتر از ۱۰مگاهرتز) با انتخاب گزینه ... Layer Stack Manager از منوی Design میتوان تعداد لایه ها و ترتیب آنها را مدیریت کرد.

۵-۱-۵ تعریفRules در پروتل

برای اینکه رسم PCB ساده تر انجام شود و از اشتباهات احتمالی جلوگیری شود می توان قوانینی را تعریف کرد تا هنگام رسم PCB، خود نرم افزار قسمتی از کارها را به صورت خودکار برای شما انجام دهد. مثلا در رسم PCB، تراکهای مربوط به تغذیه بورد باید ضخیمتر از تراکهای بقیه سیگنالها باشند. شما می توانید با ایجاد یک قانون مشخص کنید که اندازه تراک استفاده شده برای VCC و GND دقیقا 50 mil باشد. در صورتی که این قانون تعریف شده باشد، به محض اینکه با موس روی یک Pad متصل به GND کلیک کنید تا تراک را رسم کنید، خود به خود اندازه تراک 50mil می شود.

برای ایجاد قوانین، شما می توانید از منوی Design، گزینه Rules را انتخاب کنید. با انتخاب این گزینه، لیستی از قوانینی که قادر به تعریف آنها هستید نمایش داده می شود. در شکل ۵-۲ پنجره مربوط به مدیریت قوانین نمایش داده شده است. از لیست ظاهر شده برای قوانین موارد زیر خیلی مهم هستند و قطعا باید قوانینی برای آنها تعریف شود:

- Clearance Constraint
- Routing Layers
- Width Constraint



شکل ۵-۲: پنجره مربوط به تعریف و ویرایش قوانین رسم PCB

۵-۱-۶ قراردادن قطعه در لایه زیر

با دابل کلیک کردن روی یک قطعه و انتخاب گزینه Bottom Layer در قسمت Layer می توان قطعه را به پشت بورد منتقل کرد

۵-۱-۷ نوشتن متن در لایه زیر

معمولا برای راهنمایی استفاده کننده از بورد نوشته هایی روی سطح بورد قرار داده میشود(مثلا نام بورد، ورژن آن، محلهای اتصال تغذیه و ...) که این نوشته ها می تواند در لایه Top Overlay باشد یا به صورت مسی در لایه رو یا در لایه زیر نوشته شود. با انتخاب کلیدهای میانبر P و سپس S، میتوان یک رشته متنی را روی نوشت و با دابل کلیک روی آن، لایه مورد نظر را انتخاب کرد.

۵-۱-۸ چک کردن‌های نهایی بورد PCB

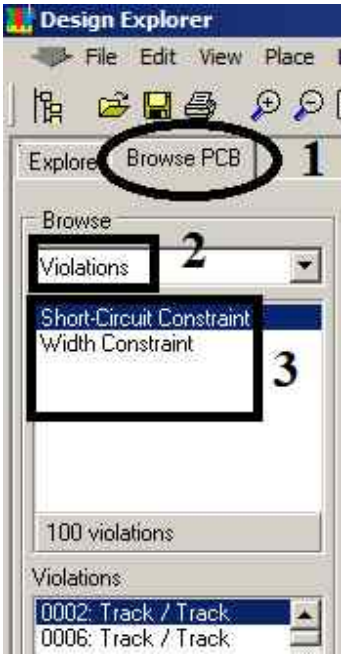
با توجه به اینکه رسم PCB یک فرایند فوق العاده حساس است و کوچکترین اشتباه منجر به عدم کارایی بورد می شود. برای اطمینان از صحت رسم PCB از موارد زیر می توان به عنوان چکهای نهایی بورد استفاده کرد:

الف) DRC

با انتخاب گزینه Design Rule Check... از منوی Tools پنجره مربوط به DRC باز میشود پس از انتخاب موارد مد نظر روی دکمه Run DRC کلیک کنید. کلیه خطاهایی که در طراحی PCB وجود دارد برای شما نمایش داده می شود.

ب)چک کردن Violations:

بعد از اینکه DRC انجام شد، در پنجره PCB، گزینه Browse PCB را انتخاب کنید(در شکل ۵-۳، این گزینه با شماره ۱ مشخص شده است). سپس در لیست نمایش داده شده گزینه Violations را انتخاب کنید(در شکل ۵-۳، این گزینه با شماره ۲ مشخص شده است). در قسمت سوم، لیستی از Ruleهایی که در رسم PCB رعایت نشده است را نشان می دهد و می توانید آنها را برطرف کنید. همانطور که در شکل ۵-۳، نمایش داده شده است، در این طرح دو قانون Short-Circuit و Width در طراحی PCB رعایت نشده است و طرح دارای ۱۰۰ خطا می باشد. این خطاها باید رفع گردند و DRC مجدد انجام شود و تا زمانی که تعداد Violationها به صفر نرسیده این پروسه باید تکرار شود.



شکل ۵-۳: نحوه دیدن Violationها

ج)چک کردن قطر سوراخ viaها و پایه‌های قطعات:

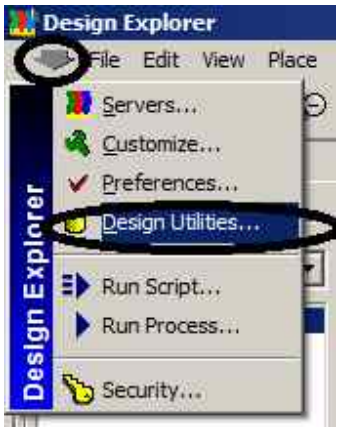
با توجه به اینکه ضخامت پایه های قطعاتی که روی بورد مونتاژ می شوند متفاوت است بنابراین باید قطر Padها و سوراخهای روی بورد را با توجه به مشخصات فیزیکی قطعات تنظیم کرد. برای تنظیم قطر padها باید روی آنها دابل کلیک کرد و با تنظیم مقدار مناسب در فیلد Hole Size این عملیات را انجام داد. با توجه به اینکه واحد اندازه گیری برای قطر مته های استفاده شده بر حسب میلیمتر است، بهتر است با زدن دکمه Q، واحد اندازه گیری را از mil به mm تغییر دهید آنگاه روی Padها دابل کلیک کنید و قطر Pad را مشخص کنید. عدد متداول برای قطر padهای مربوط به تراشه های دیپ، 0.8mm و برای کانکتورها 1mm است. با استفاده از قابلیت Global می توان تمامی پایه های مشابه را با هم انتخاب کرد و قطر Padآنها را تعیین کرد و نیازی به دابل کلیک روی تک تک پایه ها نیست.

د)کپی کردن PCB در یک طرح جدید

برای ساخت بورد شما باید فایل پروژه پروتل خود را در اختیار شرکتهای سازنده بورد مدار چاپی قرار دهید، برای اینکه طرح شما مورد سوء استفاده احتمالی قرار نگیرد بهتر است یک پروژه جدید در پروتل ایجاد کنید و یک صفحه PCB ایجاد کنید و با Select کردن مدار PCB خود و استفاده از کپی و Paste، مدار PCB خود را در این پروژه جدید قرار دهید. به این ترتیب در پروژه جدید فقط PCB وجود دارد و شماتیک مدار وجود ندارد.

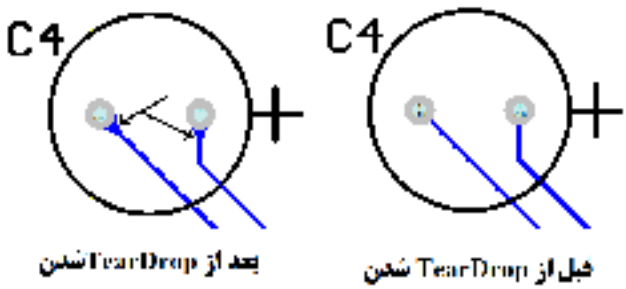
ه) استفاده از قابلیت Compact پروتل

در صورتی که حجم پروژه شما خیلی زیاد شده باشد با استفاده از قابلیت Compact می توانید آن را فشرده کنید. در شکل ۵-۴، نحوه باز کردن پنجره Compact نمایش داده شده است. توجه کنید که در این پنجره شما می توانید گزینه ای را فعال کنید که همیشه با بستن نرم افزار پروتل، پروژه شما Compact شود.



شکل ۵-۴: نحوه باز کردن پنجره Compact

برای محکم کردن اتصال تراکها به Padها می توانید این گزینه را از منوی Tools انتخاب کنید. در شکل ۵-۶، تاثیر استفاده از این گزینه روی محل اتصال تراکها با Padها نمایش داده شده است



شکل ۵-۵: تاثیر استفاده از گزینه TearDrop

ز) استفاده از PolyGon

در طراحی مدارهای فرکانس بالا، برای اینکه امپدانس تراک مربوط به GND کاهش یابد می توان تا حد امکان ضخامت تراکهای GND را افزایش داد. با استفاده از Polygon می توان به نرم افزار پروتل این فرمان را داد که کلیه فضای باقیمانده از بورد را با تراکهای پیوسته GND پر کند. برای اینکار با استفاده از کلیدهای میانبر P و سپس G، پنجره مربوط به polygon باز می شود و می توان سیگنال مورد نظر را انتخاب نمود و تنظیمات آن را انجام داد سپس با Ok کردن این پنجره و مشخص کردن ناحیه مورد نیاز برای رسم polygon، این عملیات را انجام داد.

لیستی از کلیدهای میانبر پر کاربرد در پروتل

شروع به رسم تراک	P+T
تغییر واحد اندازه گیری از mm به mil و برعکس	Q
اندازه گیری فاصله بین دو نقطه	R+M
حذف کردن قطعات، تراکها و ...	E+D
رسم Polygon	P+G
قرار دادن via	P+V

۵-۲پیش گزارش

۵- PCB مربوط به آزمایش شماره ۴ را با استفاده از Ruleهای زیر مجددا رسم کنید(از AutoRoute استفاده کنید) سپس تمام مراحل گفته شده در بخش ۵-۱-۸ را روی آن اجرا کنید.

قانون ۱: کلیه تراکهای VCC برابر 20mil باشد

قانون۲: کلیه تراکهای GNDبرابر با 25milباشد

قانون۳: تراکهای باقیمانده برابر با 8mil باشند

قانون۴: فقط از لایه زیر استفاه شود

قانون ۵: فاصله تراکها از همدیگر کمتر از 12milنباشد

قطر padهای مربوط به همه تراشه ها 0.7 میلیمتر و برای مقاومت و خازنها 0.8میلیمتر باشد

کارهای انجام شده را در قالب یک فایل با پسوند ddb ایمیل کنید. موضوع ایمیل را Az-memari-H5 قرار دهید

۶ آزمایش ششم: آشنایی با نرم افزارISE با مثال

۶-۱پیش آگاهی

در این جلسه نحوه شبیه سازی کدهای VHDL با استفاده از نرم افزار ISE توضیح داده می شود و نحوه پیاده سازی یک جمع کننده ۲ بیتی با استفاده از تمام جمع کننده ها شرح داده می شود و نهایتا جمع کننده تولید شده شبیه سازی می گردد.

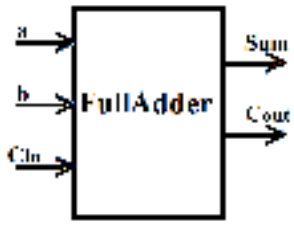
۶-۱-۱ ساختار یک برنامه *VHDL*

برای توصیف هر قطعه سخت افزاری باید یک فایل VHDL ایجاد کرد. این قطعه سخت افزاری می تواند یک گیت خیلی ساده باشد تا یک پردازنده. هر فایل VHDL از سه بخش تشکیل شده است: معرفی کتابخانه ها، تعریف اینترفیس قطعه با دنیای بیرون خود، توصیف عملکرد قطعه

برای معرفی کتابخانه از دستورات Library و use استفاده می شود. به عنوان نمونه، سه خط زیر در اغلب فایلهای VHDL آورده می شود.

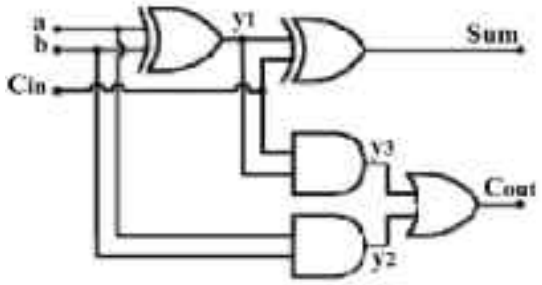
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_UNSIGNED.ALL;
```

برای تعریف اینترفیس قطعه با دنیای بیرون خود از کلمه کلیدی entity استفاده می شود. به عنوان مثال برای یک تمام جمع کننده می توان این اینترفیس را به صورت زیر تعریف کرد. (معمولا نامی که برای فایل انتخاب می شود با نامی که برای entity انتخاب می شود یکسان است)




```
entity FullAdder is
    Port (
       a    :in  std_logic;
       b    :in  std_logic;
       Cin  :in  std_logic;

       Sum  :out std_logic;
       Cout :out std_logic
                  );
end FullAdder;
```

برای بیان توصیف عملکرد قطعه، از کلمه architecture استفاده می شود. در تعریف یک architecture باید مشخص شود که این توصیف برای کدام entity نوشته می شود.



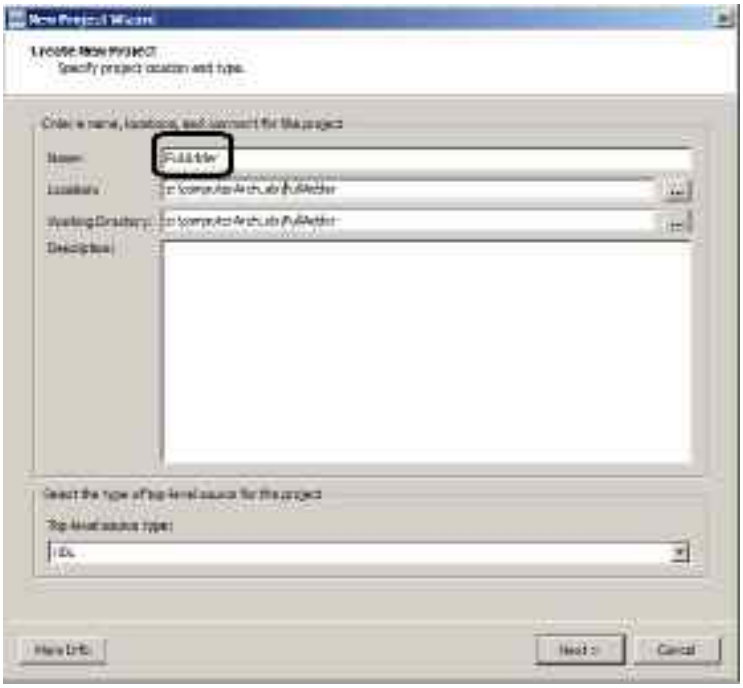
به عنوان مثال توصیف عملکرد داخلی یک تمام جمع کننده به صورت زیر خواهد بود:

```
architecture Behavioral of FullAdder is
 signal y1,y2,y3:std_logic:='0;'
begin
       y1 <= a xor b;
       Sum <= y1 xor Cin;

       y2<= a and b;
       y3<= y1 and Cin;
       Cout<= y2 or y3;
end Behavioral;
```

۶-۱-۲ آشنایی با نرم افزار *ISE*

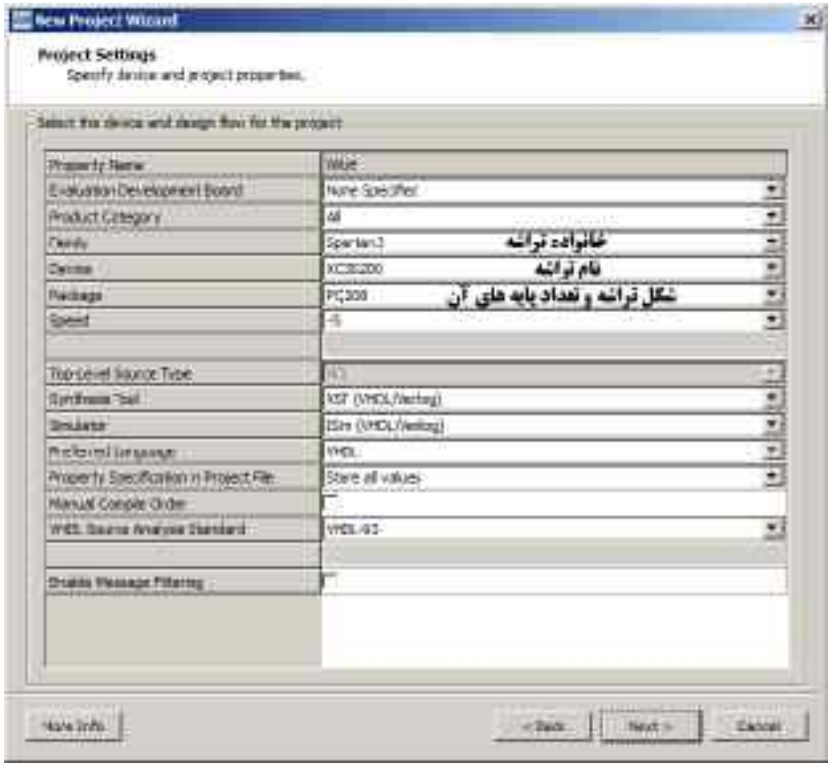
نرم افزار ISE از محصولات شرکت زایلینکس است که یک محیط مجتمع برای کار با FPGA می باشد. این نرم افزار قابلیت شبیه سازی، سنتز و ایمپلیمنت و پراگرام کردن تمام تراشه های این شرکت را دارد. در این آزمایش مراحل کامل نوشتن یک برنامه VHDL و شبه سازی آن برای یک تمام جمع کننده توسط این نرم افزار آموزش داده می شود.

از منوی File گزینه New Project را انتخاب کنید. پنجره شکل ۶-۱ نمایش داده می شود. در این پنجره در قسمت Name نام پروژه را وارد کنید (مثلا FullAdder)



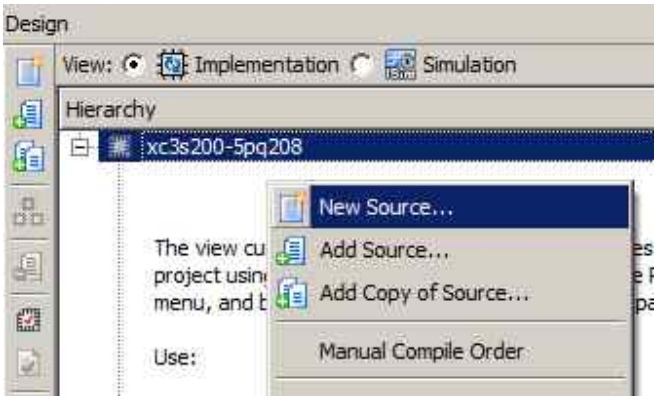
شکل ۶-۱: ایجاد یک پروژه جدید

در مرحله بعد باید نام و مشخصات FPGA را معین کرد. در شکل ۶-۲، مشخصات یکی از FPGAهایی که در آزمایشگاه استفاده می شود انتخاب شده است.



شکل ۶-۲: انتخاب FPGA

بعد از اتمام ایجاد پروژه، باید فایلهای VHDL خود را به پروژه اضافه کنید. برای ایجاد یک برنامه VHDL جدید، مطابق شکل ۶-۳ روی نام تراشه FPGA، کلیک راست نموده و گزینه New Source را انتخاب کنید.



شکل ۶-۳: ایجاد یک فایل جدید و اضافه کردن آن به پروژه

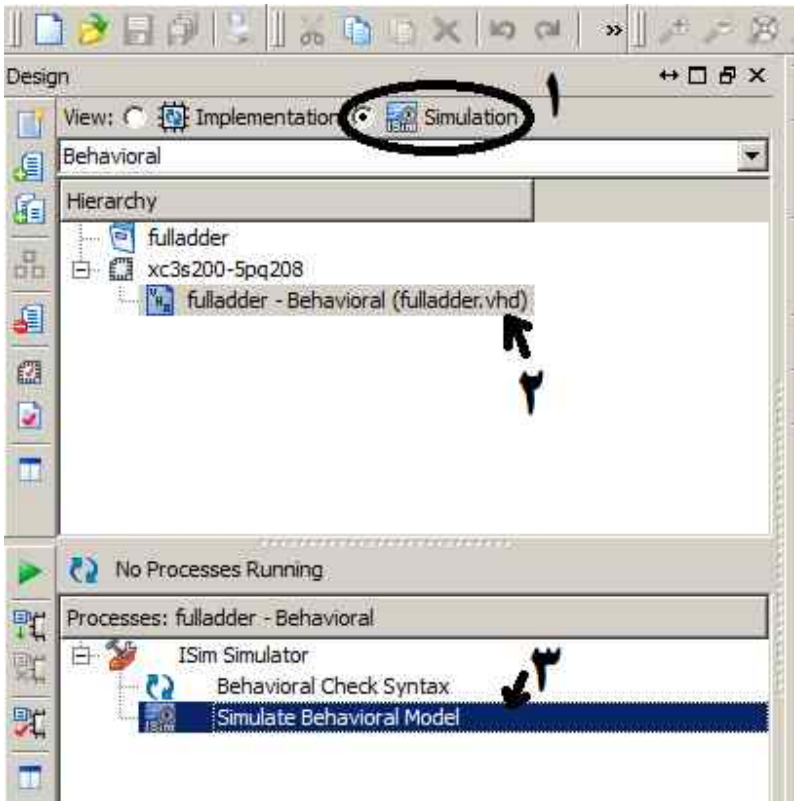
سپس در پنجره ظاهر شده، نوع فایل جدید را VHDL انتخاب کنید و نام فایل را وارد کنید. در مرحله بعد نام پورتها و نوع آن را مشخص کنید (با توجه به توضیحات داده شده در بخش ۶-۱-۱) تا نرم افزار ISE فایل VHDL را ایجاد کند.

شکل ۶-۴: انتخاب نوع فایل جدید و نامگذاری آن

بعد از اتمام مراحل فوق، فایل VHDL تولید شده نمایش داده می شود. در این فایل کتابخانه های مورد نیاز به فایل اضافه شده و همچنین Entity نیز به صورت اتوماتیک تولید شده است. تنها کاری که باید انجام شود این است که قسمت architecture نیز تکمیل گردد. طبق برنامه داده شده در بخش ۶-۱-۱، این قسمت را نیز تکمیل کنید.

۶-۱-۳ شبیه سازی مدار

برای وارد شدن به محیط شبیه سازی، باید سه گام انجام دهید. در شکل ۶-۵، این سه گام نمایش داده شده است. در گام اول گزینه Simulation را انتخاب کنید.



۶-۵: انتخاب گزینه Simulation

در گام دوم، روی نام فایل مورد نظر کلیک کنید (ممکن است در پروژه چندین فایل VHDL وجود داشته باشد که در این صورت با کلیک کردن روی نام فایل مورد نظر تعیین می کنید که قصد شبیه سازی کدام فایل را دارید)

در گام سوم، روی گزینه Simulate Behavioral Model، دابل کلیک کنید تا پنجره ISIM باز شود. در این گام به پنجره کنسول و پیغامهای نمایش داده شده نیز دقت کنید ممکن است در برنامه شما اشکالاتی وجود داشته باشد که پیغامهای آن در پنجره کنسول نمایش داده می شود.

در برنامه ISIM شما می توانید با کلیک راست روی نام سیگنالها و پورتها و انتخاب گزینه Force Constant... به سیگنالهای خود مقدار صفر یا یک را اختصاص دهید و شبیه سازی کنید.

۶-۱-۲ نوشتن برنامه *VHDL* برای قطعات پیچیده

توصیف سیستمهای سخت افزاری پیچیده توسط VHDL، عموما به صورت سلسله مراتبی انجام می شود. روش سلسله مراتبی به این صورت است که طرح بزرگ به طرحهای کوچکتر شکسته می شود و کد VHDL طرحهای کوچکتر نوشته می شود سپس با ترکیب آنها طرح بزرگتر ایجاد می گردد. به عنوان مثال برای برای پیاده سازی یک جمع کننده ۲ بیتی می توان ابتدا کد جمع کننده یک بیتی (تمام جمع کننده یا Full adder) را نوشت سپس با کنار هم قرار دادن ۲ تا از آنها می توان جمع کننده ۲بیتی را ایجاد کرد. در ادامه برنامه مورد نیاز برای ساخت یک جمع کننده ۲ بیتی با استفاده از ۲عدد FullAdder آورده شده است.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Adder2Bits is
    Port (
       a    :in std_logic_vector(1 downto 0);
       b    :in std_logic_vector(1 downto 0);
       Cin  :in std_logic;

       Sum  :out std_logic_vector(1 downto 0);
       Cout :out std_logic
                    );
end Adder2Bits;
architecture Behavioral of Adder2Bits is
component FullAdder is
    Port (
       a    :in std_logic;
       b    :in std_logic;
       Cin  :in std_logic;

       Sum  :out std_logic;
       Cout :out std_logic
                    );
end component;
signal c1:std_logic:='0';
begin
U0:  FullAdder
    Port map (
                         a    => a(0),
                         b    => b(0),
                         Cin  => Cin,

                         Sum  => Sum(0),
                         Cout => c1
                    );
U1:  FullAdder
    Port map (
                         a    => a(1),
                         b    => b(1),
                         Cin  => c1,

                         Sum  => Sum(1),
                         Cout => Cout
                    );

end Behavioral;
```

٦-٢پیش گزارش

۱- برنامه VHDL برای یک جمع کننده ۴ بیتی بنویسید به نحوی که از ۴ عدد fulladder استفاده شده باشد.

۲- برنامه VHDL برای یک جمع کننده ۸ بیتی بنویسید به نحوی که از ۴ عدد جمع کننده Adder2Bits استفاده شده باشد.

برای هر سوال کل پروژه مورد نیاز است(فقط فایل ارسال نکنید)

یک فولدر به نامYourName-H6 ایجاد کنید (به جای YourName ، نام خودتان را قرار دهید) و در داخل آن به ازای هر سوال یک زیرفولدر ایجاد کنید به نامهای Q1، Q2، Q3و.... و در داخل هر زیر فولدر کل پروژه "آی اس ای" مربوط به آن سوال را قرار دهید سپس آن را زیپ کنید و ارسال کنید.

نکته ۱: برای اینکه حجم فایلهای پروژه کاهش یابد، قبل از ارسال، گزینه CleanUp project files را از منوی project انتخاب کنید تا فایلهای موقتی را حذف کند

۷ آزمایش هفتم: پیاده‌سازی باس مشترک

۷-۱پیش آگاهی

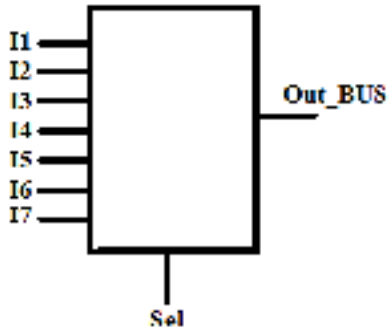
در این جلسه نحوه پیاده سازی مالتی پلکسر و استفاده از آن برای پیاده سازی باس مشترک و همچنین نحوه پیاده سازی مدارهای ترتیبی آشنا خواهید شد.

۷-۱-۱ روش پیاده سازی باس مشترک

در شکل ۷-۱، باس مشترک استفاده شده در کامپیوتر پایه نمایش داده شده است. یکی از روشهای ایجاد باس مشترک، استفاده از مالتی پلکسر می باشد. در زبان VHDL برای پیاده سازی مالتی پلکسر می توان از دستور انتساب شرطی استفاده کرد. در قطعه برنامه زیر نحوه پیاده سازی یک مالتی پلکسر 4x1 با زبان VHDL نمایش داده شده است:

```
Outp <= I0 when Sel="00"else
        I1 when Sel="01"else
        I2 when Sel="10"else
        I3 ;
```

ساختار باس مشترکی که در کامپیوتر پایه استفاده می شود در شکل ۷-۱ نمایش داده شده است. در آزمایشگاه ماجولی به نام Common_BUS ایجاد کنید و برنامه آن را بنویسید.



شکل ۷-۱: ورودی و خروجی های یک باس مشترک

۷-۱-۲ روش پیاده سازی مدارهای ترتیبی در *VHDL*

مدارهای ترتیبی در زبان VHDL را با مفهومی به نام process پیاده سازی می کنند. ساختار کلی Process به صورت زیر است:

```
Process(Clk)
Begin
    If(Clk='1' and Clk'event)then

    End if;
End process;
```

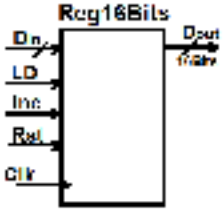
دستور if که در process فوق به کار برده شده است، بیانگر این است که دستورات داخل بدنه if زمانی اجرا شود که لبه بالارونده کلاک رخ دهد. اگر مدار به لبه پایین رونده کلاک حساس باشد آنگاه دستور Clk='1' باید به دستور Clk='0' تبدیل شود.

دستورات داخل Process به صورت ترتیبی اجرا می شوند. در ادامه نحوه پیاده سازی دو مدار ترتیبی که برای پیاده سازی کامپیوتر پایه مورد نیاز است آورده می شود

۷-۱-۳ پیاده سازی یک رجیستر ۱۶ بیتی

رجیسترهایی که در کامپیوتر پایه استفاده می شوند دارای مشخصات زیر هستند: حساس به لبه بالارونده، دارای قابلیت loadشدن، دارای قابلیت Increament، و دارای قابلیت Clear شدن.

Entity مربوط به رجیستر ۱۶ بیت را می توان به صورت شکل ۷-۲ در نظر گرفت:



شکل ۷-۲: ورودی و خروجی های یک رجیستر ۱۶ بیتی

کد مربوط به رجیستر ١٦ بیتی

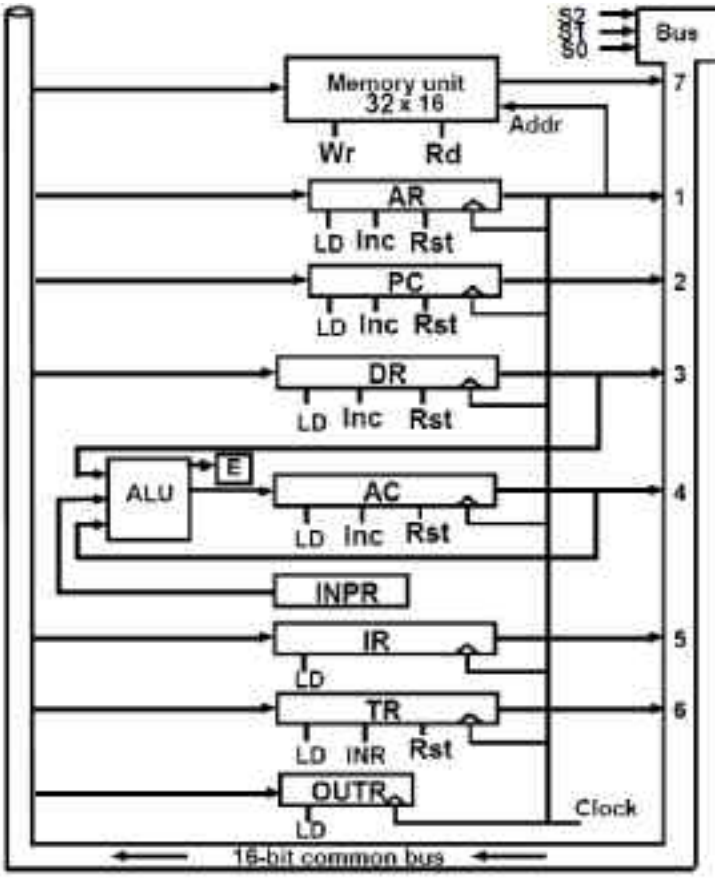
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_UNSIGNED.ALL;

entity Reg16Bits is
    Port (
            Clk   : in  STD_LOGIC;
            Rst   : in  STD_LOGIC;
            LD    : in  STD_LOGIC;
            Inc   : in  STD_LOGIC;

           Din   : in  STD_LOGIC_VECTOR(15 downto 0);
           Dout  : out STD_LOGIC_VECTOR(15 downto 0)
                  );
end Reg16Bits;

architecture Behavioral of Reg16Bits is

signal Dout_sig: STD_LOGIC_VECTOR(15 downto 0);
begin
       Dout <= Dout_sig;

       process(Clk)
       begin
              if(Clk='1' and Clk'event)then
                     if(Rst='1')then
                            Dout_sig<=(others=>'0');
                     elsif(LD='1')then
                            Dout_sig<= Din;
                     elsif(Inc='1')then
                            Dout_sig<= Dout_sig+1;
                     end if;
              end if;
       end process;
end Behavioral;
```

نکته ۱: در برنامه فوق، اولویت با Rst سپس با LD ونهایتا با Inc در نظر گرفته شده است. یعنی اگر همزمان این سه پایه با همدیگر فعال شدند اولویت به Rst داده می شود.

نکته ۲: با توجه به اینکه سیگنال Dout خروجی است نمیتوان برای افزایش آن از دستور Dout <= Dout +1; استفاده کرد، به همین دلیل یک سیگنال ۱۶ بیتی به نام Dout_sig تعریف شده است که عملیات روی آن انجام می شود و نهایتا با استفاده از دستور Dout <= Dout_sig مقدار آن به خروجی انتساب داده می شود.

۷-۲پیش گزارش

در این آزمایش قصد داریم با استفاده از برنامه های رجیستر و باس مشترک که در جلسه گذشته داشتیم، برنامه VHDL مربوط به شکل ۷-۳ را بنویسیم.



شکل ۷-۳: ساختار باس مشترک و DataPath در کامپیوتر پایه کتاب موریس مانو

۱- یک برنامه به نام Mano_DataPath بنویسید که دیاگرام شکل ۷-۳ را با استفاده از کامپوننتهای Common_Bus و Reg16 پیاده سازی کند(قسمتهای ALU و Memory و فلیپ فلاپ E در جلسه بعد توضیح داده خواهد شد).

راهنمایی: سیگنالهایی که در شکل ۷-۳ نمایش داده شده است به دو دسته تقسیم می شوند: دسته اول، سیگنالهایی هستند که به صورت داخلی می باشند و دسته دوم، سیگنالهایی هستند که باید به صورت پورتهای ورودی یا خروجی تعریف شوند.

پورتهای ورودی ماژول Mano_DataPath عبارتند از:

الف) تمام پایه های کنترلی رجیسترها (این پایه های کنترلی توسط واحد کنترل تولید خواهند شد و به ماژول Mano_DataPath خواهند آمد)

ب) خطوط انتخاب باس مشترک

ج) خطوط خواندن و نوشتن حافظه RAM

د) ۸ بیت ورودی برای به نام INPR

ه) هفت بیت به نام Control_Cmd که از واحد کنترل برای ALU می آید.

و)سیگنال کلاک و سیگنال Reset

ز) پایه های ریست ، مکمل ساز و لود برای فلیپ فلاپ e

پورتهای خروجی ماژول Mano_DataPath عبارتند از: ۸ بیت برای رجیستر OUTR، یک بیت برای E، ۱۶ بیت خروجی رجیستر IR (برای ارسال دستورالعمل به واحد کنترل)، ۱۶ بیت خروجی رجیستر AC، ۱۶ بیت خروجی رجیستر DR

پروژه ISE را که شامل برنامه های نوشته شده در جلسه قبل و فایل Mano_DataPath است را حداکثر تا ساعت ۱۲ شب قبل از آزمایشگاه با موضوع Az-Memari-H7 ارسال کنید.

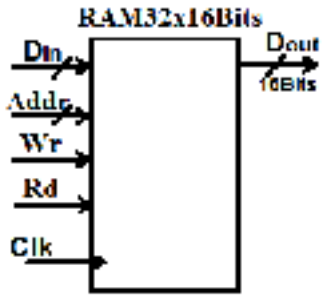
۸ آزمایش هشتم: تکمیل پیاده سازی بلوک DataPath و شبیه سازی آن

۸-۱پیش آگاهی

همانطور که در آزمایش ۷ توضیح داده شد، بلوک DataPath در کامپیوتر پایه دارای ۶ زیر بخش بود. در آزمایش قبل با زیر بخشهای Common_Bus، Reg16 آشنا شدید در این جلسه با نحوه پیاده سازی زیربخش Memory، فلیپ فلاپ E و ALU آشنا خواهید شد. همچنین با مفهوم TestBench و نحوه استفاده از آن برای شبیه سازی سخت افزار آشنا خواهید شد.

۸-۱-۱ پیاده سازی یک حافظه *RAM*

حافظه استفاده شده در کامپیوتر پایه به صورت ۴۰۹۶ کلمه ۱۶ بیتی است. برای پیاده سازی حافظه در زبان VHDL می توان از آرایه استفاده کرد. از آنجایی که آرایه بعد از پیاده سازی حجم وسیعی از منابع FPGA را اشغال می کند به جای پیاده سازی یک حافظه ۴۰۹۶کلمه ای یک حافظه ۳۲ کلمه ای پیاده سازی می کنیم. در شکل ۸-۱، ورودی ها و خروجی های حافظه RAM نمایش داده شده است.



شکل ۸-۱: ورودی و خروجی های حافظه RAM

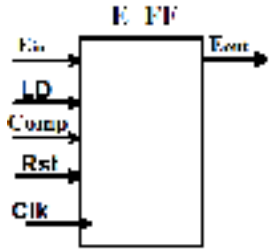
نکته پیاده سازی: در FPGAهای امروزی، بلوکهای مخصوص حافظه وجود دارد که به Block RAM معروف هستند و در صورت استفاده از آنها، منابع داخلی FPGAها هدر نمی رود. (استفاده از آرایه برای پیاده سازی حافظه باعث هدر رفتن منابع FPGAها و کند شدن فرایند سنتز و ایمپلیمنت خواهد شد)

برنامه VHDL برای توصیف یک حافظه ۳۲ کلمه ای که هر کلمه آن ۱۶ بیتی است

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.std_logic_unsigned.all;

entity RAM32x16Bits is
    Port (
           Addr     : in      STD_LOGIC_VECTOR(4 downto 0);
           Clk      : in      STD_LOGIC;
           Rd       : in      STD_LOGIC;
           Wr       : in      STD_LOGIC;
           Din      : in      STD_LOGIC_VECTOR (15 downto 0);
           DOut     : out     STD_LOGIC_VECTOR (15 downto 0)
  );
end RAM32x16Bits;

architecture Behavioral of RAM32x16Bits is
Subtype RAM_WORD is std_logic_vector(15 downto 0) ;
Type RAM_TABLE is array ( 0 to 31) of RAM_WORD;
signal RAM: RAM_TABLE :=
         (
             x"0000", x"0111", x"0222", x"0333",
             x"0444", x"0555", x"0666", x"0777",
             x"0888", x"0999", x"0aaa", x"0bbb",
             x"0ccc", x"0ddd", x"0eee", x"0fff",
             x"1000", x"1111", x"1222", x"1333",
             x"1444", x"1555", x"1666", x"1777",
             x"1888", x"1999", x"1aaa", x"1bbb",
             x"1ccc", x"1ddd", x"1eee", x"1fff"
             );
Begin

DOut <=  RAM(conv_integer ( Addr ));
process(Clk)
begin
       if(Clk='1' and Clk'event)then
              if(Wr='1')then
                     RAM(conv_integer ( Addr ))<=   Din;
              end if;
       end if;
end process;
end Behavioral;
```

۸-۱-۲ پیاده سازی فلیپ فلاپ *E*

بلوک دیاگرام فلیپ فلاپ E در شکل ۸-۲ نمایش داده شده است. پیاده سازی این فلیپ فلاپ مشابه برنامه نوشته شده در آزمایش ۷ برای پیاده سازی رجیستر می باشد.

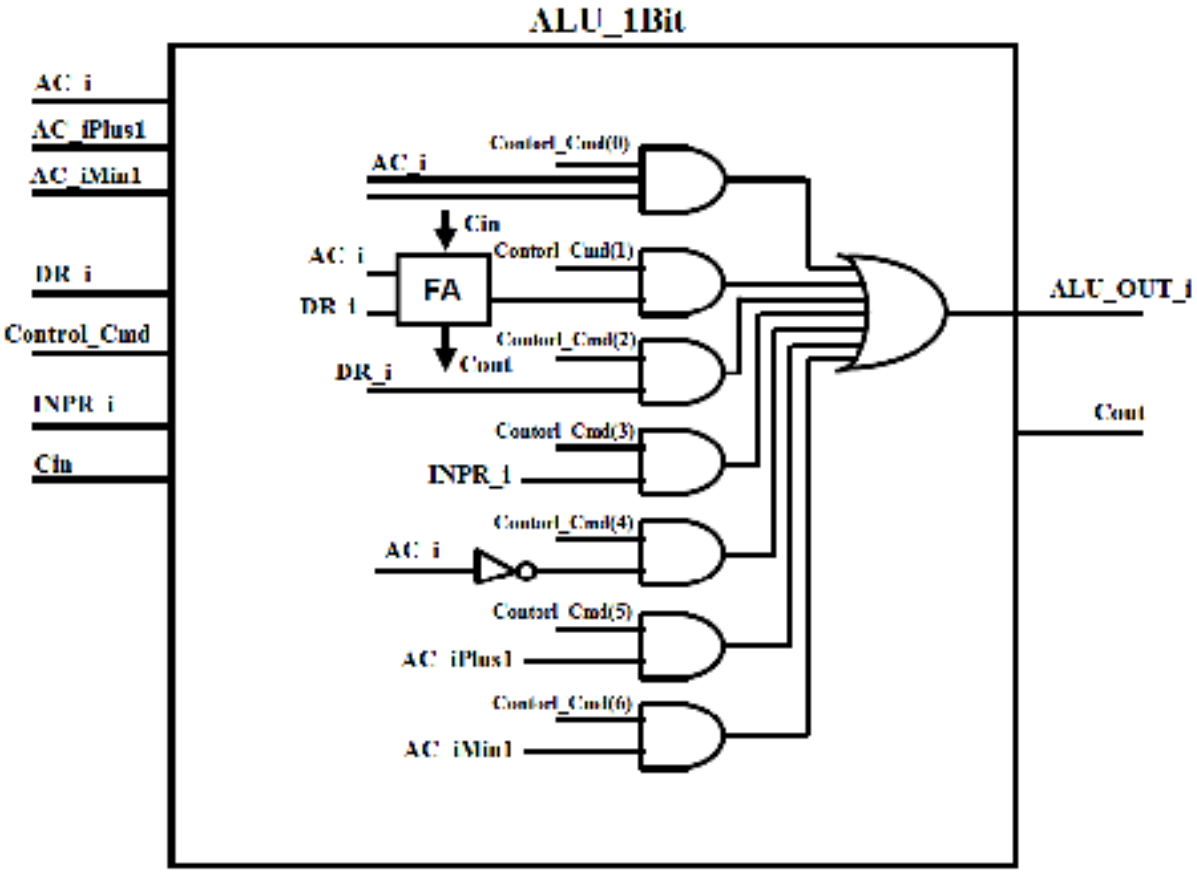


شکل ۸-۲: ورودی و خروجی های فلیپ فلاپ E

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity E_FF is
    Port (
            Clk   : in  STD_LOGIC;
            Rst   : in  STD_LOGIC;
            LD    : in  STD_LOGIC;
            Comp  : in  STD_LOGIC;
           Ein   : in  STD_LOGIC;
           Eout  : out STD_LOGIC
                    );
end E_FF;

architecture Behavioral of E_FF is

signal Eout_sig: STD_LOGIC;
begin
       Eout <= Eout_sig;
       process(Clk)
       begin
              if(Clk='1' and Clk'event)then
                     if(Rst='1')then
                            Eout_sig<= '0';
                     elsif(LD='1')then
                            Dout_sig<= Din;
                     elsif(Comp='1')then
                            Eout_sig<= not Eout_sig;
                     end if;
              end if;
       end process;
end Behavioral;
```

۸-۱-۳ پیاده سازی واحد *ALU*

برای پیاده سازی ALU بهتر است به روش ساختاری عمل کرد برای این منظور ابتدا یک ALU_1Bit طراحی می شود سپس با Port Map کردن ۱۶ عدد از آن، یک ALU شانزده بیتی طراحی می کنیم. در شکل ۸-۳، بلوک دیاگرام داخلی ALU یک بیتی نمایش داده شده است.



شکل ۸-۳: پورتها و بلوک دیاگرام داخلی واحد ALU

۸-۱-۴ مفهوم *TestBench*

برای شبیه سازی و تست یک برنامه VHDL، باید ورودیهای مورد نیاز برای آن را تولید کرد. برنامه ISIM، در محیط شبیه سازی خود یکسری سیگنالهای از قبل تعریف شده ای دارد که می توان از آنها برای شبیه سازی استفاده کرد ولی اگر بخواهیم شبیه سازی مدار را با انعطاف پذیری بیشتری انجام دهیم، باید یک برنامه VHDL نوشت که ورودیهای مورد نیاز برای تست قطعه مد نظرمان را تولید کند. به این برنامه VHDL که مخصوص شبیه سازی نوشته می شود، TestBench گفته می شود. TestBench هیچگونه پورت ورودی یا خروجی ندارد و در واقع Entity آن خالی است و در قسمت Architecture باید کامپوننت مربوط به برنامه ای که می خواهیم تست کنیم را معرفی کنیم. سپس سیگنالهای ورودی مورد نیاز را تعریف کنیم و بعد از کلمه کلیدی Begin، برنامه خود برای تولید سیگنالها را بنویسیم.

به عنوان مثال در ادامه یک برنامه testBench برای FullAdder آورده شده است.

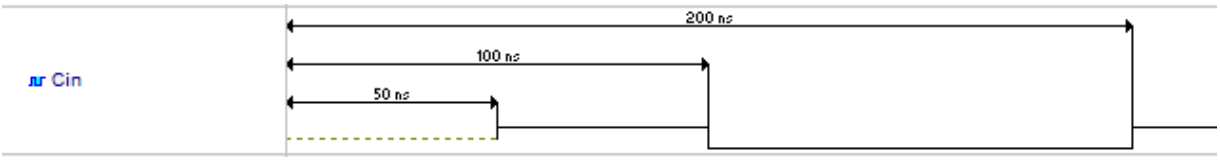
```
library ieee;
use ieee.STD_LOGIC_UNSIGNED.all;
use ieee.std_logic_1164.all;
entity fulladder_tb is
end fulladder_tb;
architecture TB_ARCHITECTURE of fulladder_tb is
component fulladder
	port(
		a : in STD_LOGIC;
		b : in STD_LOGIC;
		Cin : in STD_LOGIC;
		Sum : out STD_LOGIC;
		Cout : out STD_LOGIC );
end component;
	signal a : STD_LOGIC:='0';
	signal b : STD_LOGIC:='0';
	signal Cin : STD_LOGIC;
	signal Sum : STD_LOGIC;
	signal Cout : STD_LOGIC;

begin
	UUT : FullAdder
		port map (
			a    => a,
			b    => b,
			Cin  => Cin,
			Sum  => Sum,
			Cout => Cout
		);
	a   <= not a after 10ns;
	b   <= not b after 20ns;
	Cin <= '1' after 50 ns,     '0' after 100 ns, '1' after 200ns;
end TB_ARCHITECTURE;
```

توضیحات برنامه

سیگنالهای a و b در برنامه TestBench، مقدار دهی اولیه شده اند(در هنگام تعریف این دو سیگنال)

دستور a<= not a after 10ns; باعث می شود که سیگنال a هر به ۱۰ نانو ثانیه مکمل شود، بنابراین سیگنال a معادل یک کلاک ۵۰مگاهرتز خواهد بود (چرا؟!!!)

سیگنال Cin به این صورت مقداردهی شده است که بعد از ۵۰ نانو ثانیه(از لحظه شروع شبیه سازی)، مقدار آن برابر با ۱ شود بعد از ۱۰۰ نانوثانیه (از لحظه شروع شبیه سازی) مقدار آن صفر شود و نهایتا بعد از ۲۰۰نانوثانیه(از لحظه شروع شبیه سازی)، مقدار آن ۱ شود و تا بینهایت ۱ بماند. در شکل ۸-۴، شکل موج تولید شده برای سیگنال Cin نمایش داده شده است.

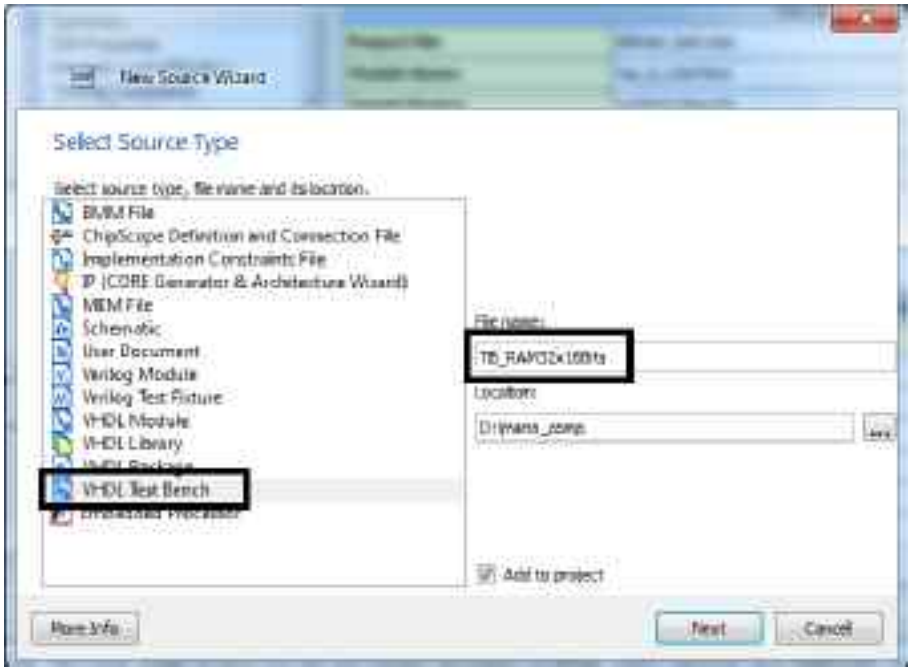


شکل ۸-۴: نتیجه دستور Cin <= '1' after 50 ns,'0' after 100 ns, '1' after 200ns

اگر وارد محیط شبیه سازی شوید و فایل testBench را برای شبیه سازی انتخاب کنید دیگر نیازی به استفاده از گزینه Simulators برای مقداردهی سیگنالها نیست و مستقیما می توانید دکمه شروع شبیه سازی را بزنید.

نکته پیاده سازی: دستور after قابل سنتز شدن روی FPGA نیست و فقط برای شبیه سازی استفاده می شوند. نرم افزارهای سنتز کننده از دستورهای ایجاد تاخیر صرف نظر می کنند. در صورت نیاز به تولید تاخیر در سخت افزار باید آن را با استفاده از شمارنده ایجاد کرد (در واقع تاخیر می تواند به صورت مضاربی از پریود کلاک باشد)

نرم افزار ISE این امکان را به کاربران می دهد تا چهارچوب برنامه TestBench را به صورت اتوماتیک تولید کند. برای ایجاد TestBench کافی است در پنجره پروژه، کلیک راست نموده و گزینه New Source... را انتخاب کنید سپس در پنجره ظاهر شده گزینه VHDL TestBench را انتخاب کنید و نامی برای این فایل انتخاب کنید(معمولا یک پسوند TB_ به ابتدا یا انتهای نام فایل اصلی اضافه می شود). در شکل ۸-۵، نحوه ایجاد یک TestBench نمایش داده شده است.



شکل ۸-۵: نحوه ایجاد یک TestBench

۸-۲پیش گزارش

۱- با توجه به شکل ۸-۳، برنامه ALU_1Bit را بنویسید سپس با استفاده از آن برنامه ALU_16Bits را بنویسید.

۲- برنامه Mano_DataPath را که در جلسه قبل نوشتید را کامل کنید و بلوکهای حافظه، ALU_16Bits و E_FF را به آن اضافه کنید.

پروژه ISE را که شامل برنامه های نوشته شده در جلسات قبل و برنامه تکمیل شده Mano_DataPath است را را حداکثر تا ساعت ۱۲ شب قبل از آزمایشگاه با موضوع Az-Memari-H8 ارسال کنید.

۹ آزمایش نهم: پیاده‌سازی واحد کنترل

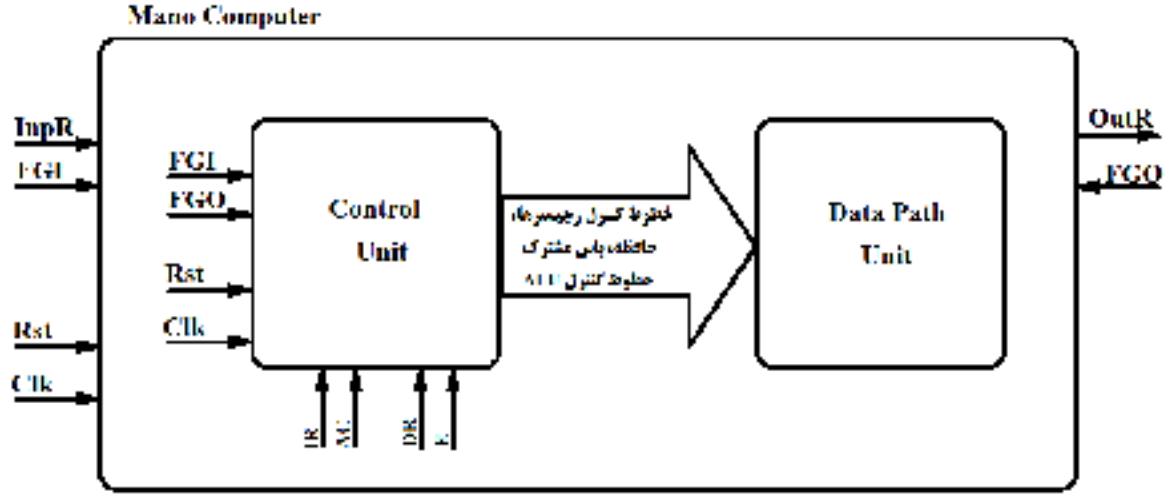
۹-۱پیش آگاهی

هدف از این آزمایش، شبیه سازی واحد کنترل و تکمیل کامل کامپیوتر پایه و شبیه سازی آن می باشد. در جلسات گذشته ، واحد Data_Path پیاده سازی و شبیه سازی شد در این جلسه واحد کنترل سخت افزاری کامپیوتر پایه شبیه سازی می شود و کامپیوتر پایه تکمیل و شبیه سازی خواهد شد.

۹-۱-۱ بلاک دیاگرام کامپیوتر پایه

در شکل ۹-۱، بلوک دیاگرام کلی کامپیوتر پایه نمایش داده شده است. ورودی ها و خروجی های این کامپیوتر پایه، سیگنالهای InpR و OutR و بیتهای فلگ آنها و سیگنالهای ریست و کلاک می باشد. سیگنال ورودی InpR توسط یک دیپ سوئیچ هشت کلیدی روی بورد مقدار دهی می شود و محتویات سیگنال OutR روی دو نمایشگر هشت قسمتی به صورت یک عدد مبنای شانزده نمایش داده می شود. برای مقدار دادن به پایه های FGI، FGO، Rst و Clk از چهار کلید فشاری روی بورد استفاده می شود.

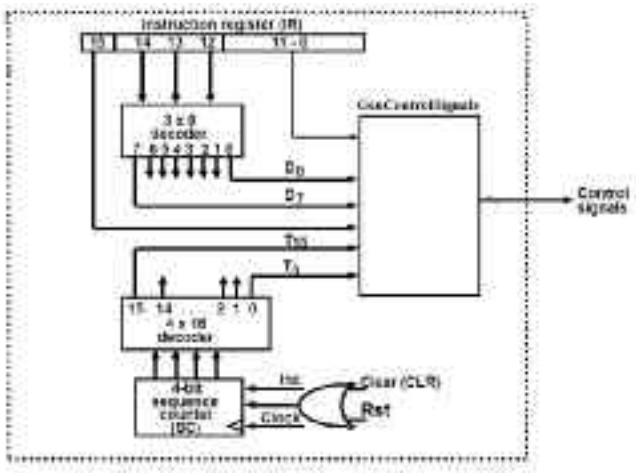
همانطور که در شکل مشاهده می شود کامپیوتر پایه از دو جزء اصلی به نامهای Data_path و Control_Unit تشکیل شده است. بلوک Data_path در جلسه هشتم تست و شبیه سازی گردید. در ادامه، توضیحاتی در مورد نحوه پیاده سازی بلوک Control_Unit ارائه خواهد شد.



شکل ۹-۱: بلوک دیاگرام کلی کامپیوتر پایه

۹-۱-۲ واحد کنترل

وظیفه واحد کنترل، دریافت سیگنالهای وضعیت از واحد Data_path و تولید سیگنالهای کنترلی برای آن واحد می باشد. سیگنالهایی که واحد کنترل دریافت می کند عبارتند از: بیت علامت AC، وضعیت صفربودن رجیستر AC، وضعیت صفر بودن رجیستر DR، رجیستر IR و مقدار فلیپ فلاپ E . بلاک دیاگرام داخلی واحد کنترل در شکل ۹-۲ نمایش داده شده است. اجزای تشکیل دهنده واحد کنترل عبارتند از: دو عدد دیکدر، یک شمارنده چهار بیتی و زیر بلوک GenControlSignals. در بخشهای بعدی روش پیاده سازی برخی از قطعات مورد نیاز برای طراحی واحد کنترل آورده می شود.



شکل ۹-۲: دیاگرام داخلی واحد کنترل

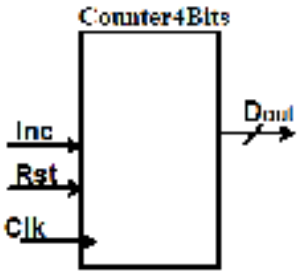
۹-۱-۳ روش پیاده سازی دیکدر

با توجه به اینکه دیکدر، جزء مدارهای ترکیبی می باشد بنابراین برای پیاده سازی آن می توان از دستور when استفاده کرد.. در ادامه کد یک دیکد 3x8 نمایش داده شده است. این دیکدر بدون پایه فعال ساز می باشد:

```
Outp<=  "00000001" when Inp="000" else
        "00000010" when Inp="001" else
        "00000100" when Inp="010" else
        "00001000" when Inp="011" else
        "00010000" when Inp="100" else
        "00100000" when Inp="101" else
        "01000000" when Inp="110" else
        "10000000" ;
```

۹-۱-۴ روش پیاده سازی شمارنده

با توجه به اینکه شمارنده یک مدار ترتیبی است بنابراین برای نوشتن کد آن، نیاز به یک Process می باشد. بلاک دیاگرام یک شمارنده ۴ بیتی در شکل ۹-۳ نمایش داده شده است.



شکل ۹-۳: ورودی و خروجی های یک شمارنده ۴ بیتی

کد مربوط به یک شمارنده ٤ بیتی

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_UNSIGNED.ALL;
entity Counter4Bits is
    Port (
            Clk   : in  STD_LOGIC;
            Rst   : in  STD_LOGIC;
            Inc   : in  STD_LOGIC;

           Dout  : out STD_LOGIC_VECTOR(3 downto 0)
                    );
end Counter4Bits;
architecture Behavioral of Counter4Bits is
signal Dout_sig: STD_LOGIC_VECTOR(3 downto 0);
begin
       Dout <= Dout_sig;
       process(Clk)
       begin
              if(Clk='1' and Clk'event)then
                     if(Rst='1')then
                            Dout_sig<=(others=>'0');
                     elsif(Inc='1')then
                            Dout_sig<= Dout_sig+1;
                     end if;
              end if;
       end process;
end Behavioral;
```

Table 2: Computer Micro-Operations and Controls for the Mano Machine.

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$,
		$AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	D_7IT_3:	$AR \leftarrow M[AR]$
Interrupt:		
$T_0'T_1'T_2'(IEN)(FGI + FGO)$:		$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$
Register-reference:		
		$D_7I'T_3 = r$ (common to all register-reference instructions)
		$IR(i) = B_i$ $(i = 0, 1, 2, \ldots, 11)$
	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow \overline{AC}$
CME	rB_8:	$E \leftarrow \overline{E}$
CIR	rB_7:	$AC \leftarrow$ shr AC, $AC(15) \leftarrow E$, $E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow$ shl AC, $AC(0) \leftarrow E$, $E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1)$
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$
Input-output:		
		$D_7IT_3 = p$ (common to all input-output instructions)
		$IR(i) = B_i$ $(i = 6, 7, 8, 9, 10, 11)$
	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR$, $FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7)$, $FGO \leftarrow 0$
SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$
SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

۹-۲پیش گزارش

۱- با استفاده از کدهای ارائه شده در بخش پیش گزارش، واحد کنترل کامپیوتر پایه را پیاده سازی کنید.

۲- یک Entity به نام mano_computer ایجاد کنید و با توجه به شکل ۹-۱، دو کامپوننت Data_path و Control_unit را در آن Port Map کنید.

۳- یک TestBench برای mano_computer ایجاد کنید و با مقدار دهی مناسب سیگنالهای Clk و Reset و صفر کردن فلگهای ورودی و خروجی، کامپیوتر را شبیه سازی کنید.

تمام عملیات فوق را در یک WorkSpace انجام دهید سپس آن را زیپ کنید و حداکثر تا ساعت ۱۲ شب قبل از آزمایشگاه با موضوع Az-Memari-H9 ارسال کنید.

۱۰ آزمایش دهم: پیاده‌سازی کامل کامپیوتر پایه و تست آن

۱۰-۱پیش آگاهی

در آزمایشهای قبلی برنامه VHDL مربوط به کامپیوتر پایه شبیه سازی شد و از لحاظ فانکشنال تست شد. هدف از این آزمایش، پیاده سازی کامپیوتر پایه روی یک بورد FPGA می باشد به نحوی که عملکرد کامپیوتر را به صورت واقعی مشاهده کنید. برای این منظور باید کد VHDL را سنتز و ایمپلیمنت کنید. یکی از نرم افزارهایی که برای این کار استفاده می شود، نرم افزار ISE از شرکت Xilinx است. اصولا هر شرکت تولید کننده FPGA باید نرم افزاری جهت پیاده سازی کدهای توصیف سخت افزار داشته باشد.

۱۰-۱-۱ مراحل پیاده سازی کد *VHDL* روی *FPGA*

برای پیاده سازی یک مدار روی FPGA مراحل زیر مورد نیاز است:

الف) طراحی مدار

ب) توصیف مدار با زبانهای توصیف سخت افزار

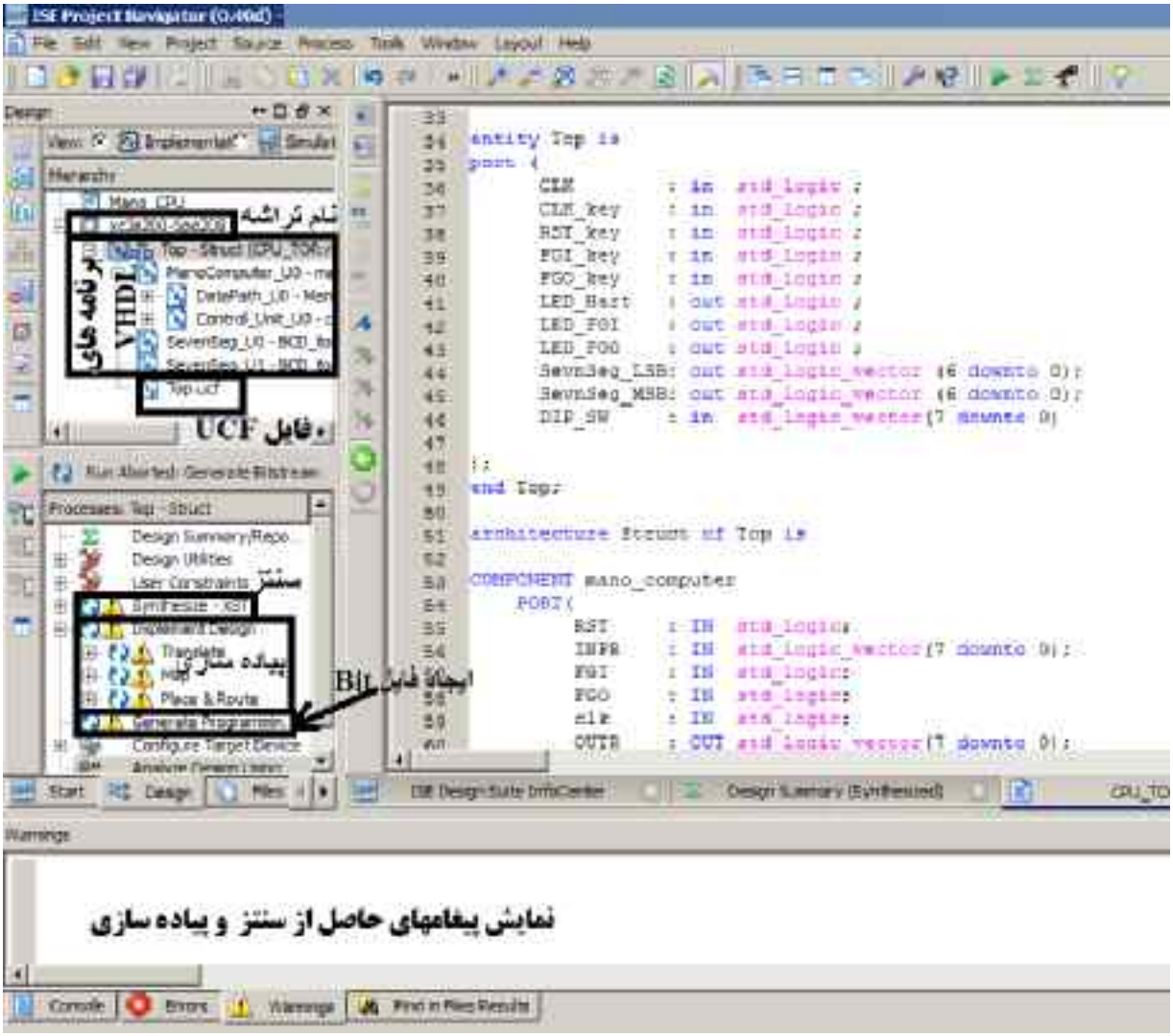
ج) شبیه سازی

د)سنتز

ه)پیاده سازی

اگر بخواهیم مراحل فوق را برای پیاده سازی کامپیوتر پایه روی FPGA، بررسی کنیم می توان گفت: طراحی مدار در کتاب معماری کامپیوتر آورده شده است، توصیف کامپیوتر پایه با زبان VHDL در جلسات قبلی آزمایشگاه انجام شد و با استفاده از نرم افزار ActiveHDL شبیه سازی گردید. در مرحله سنتز، برنامه نوشته شده به زبان VHDL به المانهای دیجیتال نظیر فلیپ فلاپ، حافظه، دیکدر، مالتی پلکسر و ... تبدیل می شود و نهایتا در مرحله پیاده سازی، کدهای سنتز شده با المانهای موجود در FPGA جایگزین و فایل نهایی برای پراگرام کردن FPGA آماده می شود.

برای سنتز و پیاده سازی برنامه VHDL، نرم افزارهای مختلفی وجود دارد. با توجه به اینکه در طراحی بوردهای آزمایشگاه، از FPGAهای شرکت Xilinx استفاده شده است بنابراین از نرم افزار تولید شده شرکت زایلینکس برای سنتز و پیاده سازی کدهای VHDL نوشته شده استفاده می کنیم. در شکل ۱۰-۱، شکل ظاهر این نرم افزار نمایش داده شده است. در این شکل، قسمتهای مختلف این نرم افزار مشخص شده اند. هنگام ایجاد یک پروژه جدید در این نرم افزار، باید نام خانواده FPGA، نام قطعه، نوع بسته بندی انتخاب گردد. مثلا FPGAهای استفاده شده در بوردهای آزمایشگاه از خانواده Spartan هستند و نام قطعه 3s400 یا 3s200 (نام قطعه را می توانید از روی تراشه بخوانید) نوع بسته بندی تراشه pq208 می باشد(تراشه از نوع چهارطرفه و دارای ۲۰۸ پایه). بعد از اینکه در این نرم افزار پروژه خود را ایجاد کردید نام تراشه به صورت اختصار در قسمت فوقانی پنجره می نویسد . تراشه ای که در شکل ۱۰-۱ استفاده شده است xc3s200-5pq208 می باشد که حاوی تمام اطلاعات تراشه مورد نظر می باشد.



شکل ۱۰-۱: قسمتهای مختلف نرم افزار ISE

بعد از ایجاد پروژه و اضافه کردن فایلها به آن، با دابل کلیک کردن روی گزینه Synthesize، عملیات سنتز شروع می گردد. گزارش مربوط به سنتز کردن را در پنجره console (قسمت پایینی پنجره) نمایش می دهد. در صورت موفقیت آمیز بودن عملیات سنتز، میتوان گام بعدی که پیاده سازی طرح باشد را شروع کرد. برای شروع عملیات پیاده سازی باید روی گزینه Implement Design دابل کلیک کنید. همانطور که در شکل ۱۰-۱ مشاهده می شود عملیات پیاده سازی از سه گام به نامهای Translate، Map و Place & Route تشکیل شده است. برای ایجاد فایل نهایی باید روی گزینه Generate Programming... دابل کلیک کنید.

برای پراگرام کردن FPGA می توان مستقیما روی گزینه Configure Target Device دابل کلیک کرد یا اینکه از نرم افزار Impact استفاده کرد. این نرم افزار جزء مجموعه ISE است و به صورت اتوماتیک هنگام نصب نرم افزار ISE، نصب می گردد.

۱۰-۲پیش گزارش

۱- برنامه ای بنویسید که عددی را از رجیتر INPR بخواند و با یک جمع بزند و آنرا در رجیستر OUTR بنویسد و این کار را در یک حلقه به صورت دائم تکرار کند.

۲- کد زبان ماشین برنامه فوق را استخراج کنید و فایل VHDL مربوط به حافظه کامپیوتر پایه را باز کنید و کدها را در آدرسهای صفر، یک، دو و سه حافظه وارد کنید

۳- کامپیوتر پایه را شبیه سازی کنید تا دستورات فوق را اجرا کند. از شبیه سازی اجرای هر دستور عکس بگیرید. (در صفحه wave Form، سیگنالهای کلاک، رجیستر IR، خروجی حافظه، سیگنالهای زمانبندی T0، T1 و... سیگنالهای کنترلی که برای اجرای دستورات فوق باید فعال شوند را اضافه کنید)

۴- نرم افزار ISE را روی لپ تاپهای خود نصب کنید تا در این جلسه عملیات سنتز و پیاده سازی کامپیوتر پایه را انجام دهیم.

تمام عملیات فوق را در یک WorkSpace انجام دهید سپس آن را زیپ کنید و حداکثر تا ساعت ۱۲ شب قبل از آزمایشگاه با موضوع Az-Memari-H10 ارسال کنید.