فصل اول

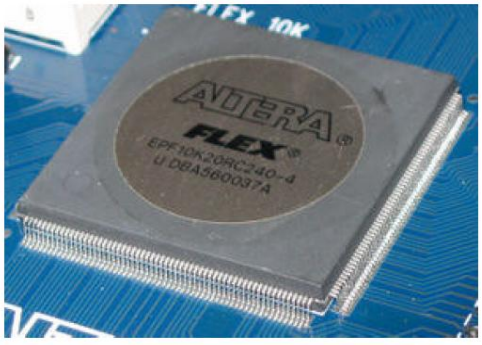
عنوان پروژه و تحقیق

طراحی و ساخت برد آموزشی Xilinx FPGA با زبان برنامه نویسی VHDL

سالها پیش که طراحی دیجیتال پابه عرصه وجود نهاد و ICهای استانداردی چون گیتها، فلیپ فلاپها، لچها شمارنده ها و...ساخته شدند به تدریج پردازنده هایی با قدرت محدود که اولین کامپیوتر های شخصی بر اساس آنها طراحی شده بود دنیای دیجیتال را به وجود آوردند، تصور روزی که فاصله سخت افزار و نرم افزار به حد کنونی برسد به طوری که تمام مرزهای طراحی رادر نوردیده و سخت افزار به نرمی و انعطاف پذیری درآید بسیار دشوار بود.

اما بعدها با طراحی حافظه های قابل برنامه ریزی دوباره و فن آوری EPROMحافظه های پایای با قابلیت برنامه ریزی و پاک سازی و PALآرایه های منطقی قابل برنامه ریزی و سرانجام فن آوری آرایه های سوئیچ های فیوزهای قابل برنامه ریزی چندباره،انقلابی نوین رادر عرصه طراحی دیجیتال به وجود آورد که مفهوم طراحی دیجیتال را دچار تحولی عظیم در عرصه های دیدگاه معماری،حجم طراحی،سرعت ونوع نگرش به طراحی دیجیتال نموده است.

به طوری که امروزه FPGAها آرایه های گیتی قابل برنامه ریزی میدانییک بوم نقاشی سفیدرا در اختیار طراح قرار می دهند که به او اجازه می دهد تا طراحی دیجیتال خود را آنچنان که می خواهد وباهر حجم وپیچیدگی لازم ،طراحی وسپس به جای انتخاب ICهای استاندارد وجداازهم وکنارهم قرار دادن آنهادر روی یک مدار ووصل کردن آنها از طریق یک بورد مدار چاپی (PCB)،بااستفاده از یکی از زبانهای توصیف سخت افزاری نظیر VHDL،هریک از قطعات دیجیتالی مورد نیاز را نوشته وبا وصل کردن نرم افزاری آنها ،سرانجام فایل کامپایل شده نهایی را از طریق یک رابط سخت افزاری برروی یک بسته سخت افزاری خام باتعداد پایه های مورد نیاز برنامه ریزی کرده وازاین IC جدید "خودساخته"استفاده کند.

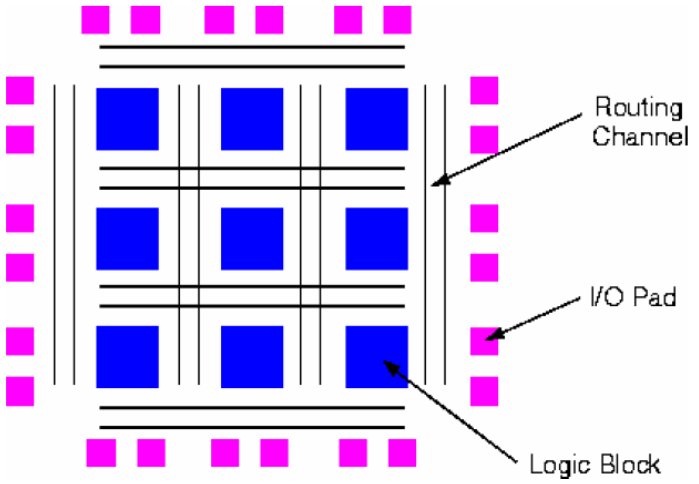


شکل شماره۱-۱ نمونه آی سی شرکت ALTRA

بررسی ساختار داخلی FPGA

FPGA را می توان به صورت جزیرهای مجزایی در نظر گرفت که توسط شاهراهایی به هم متصل می گردند. به عبارتی FPGA شامل یک سری بلوک منطقی و نیز سیم های بین انها می گردد.

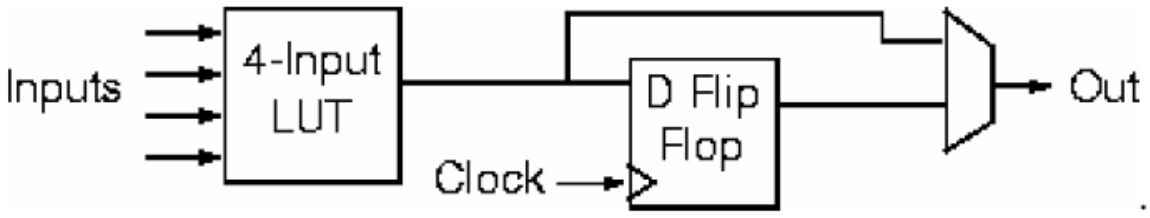
دو پد ورودی وخروجی نیز در انتهای هریک از ردیفها یاستوانها قرار داده شده است.خطوط اتصال دهنده بین بلوکها ازنظر تعداد واندازه یکسان می باشند که در شکل شماره ۲-۱ نیز نشان داده شده است.



شکل شماره ۲-۱ خطوط اتصال دهنده بین بلوکهای داخلی FPGA

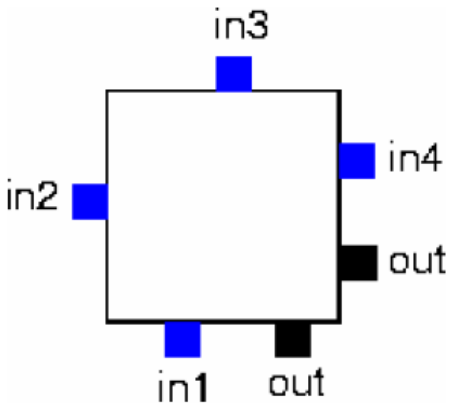
هرمدارمنطقی که روی FPGA بااستفاده از پروگرامر ریخته می شود طوری روی FPGA پیاده می شود که با استفاده از بلوکهای منطقی ومسیرها بین این بلوکها بتواند تابع داده شده را اجرا نماید.

هربلوک منطقی در FPGA دارای ۴ ورودی به جدول مراجعه ای (Look Up Table_LUT) و یک فلیپ فلاپ،می باشد. درشکل شماره ۳-۱ نشان داده شده است :



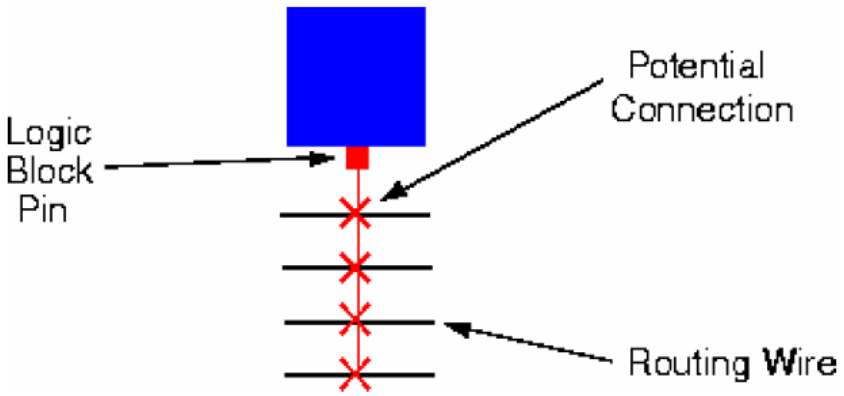
درشکل شماره ۳-۱ مدار داخلی در هر بلوک منطقی FPGA

تنها یک خروجی برای این بلوک وجود دارد که می تواند خروجی ثبت شده (Registerd)یا ثبت نشده (Un Registerd) قبلی را به خارج انتقال دهد ،ازآنجایی که پایه Clock به طور داخلی سیم کشی شده است. در نوع تجارتی می توانید این پایه را نادیده بگیرید ودر نهایت هر بلوک به شکل شماره ۴-۱ خواهد بود.



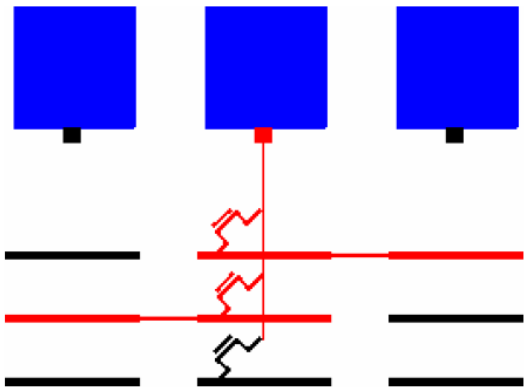
شکل شماره ۴-۱ شمای کلی هر بلوک داخلی FPGA

وضعیت هر پایه ورودی به گونه ای است که قابلیت اتصال به هریک از بخشهای کانال را داشته باشد شکل شماره ۵-۱ به طور کامل وضعیت آن را بیان می دارد.



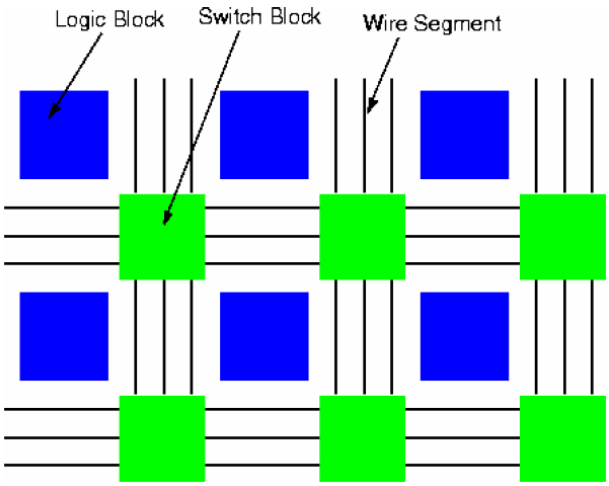
شکل شماره ۵-۱ وضعیت هر پایه ورودی FPGA

در شکل شماره ۶-۱ نیز نمایی از طریقه برقراری این اتصالات آورده شده است .این کار را می توان به راحتی با استفاده از سوییچ های MOSFET انجام داد .بافرمان به هر قسمت می توان آن مسیر را به پایه متصل نمود.



شکل شماره ۶-۱ برقراری اتصالات پایه ورودی FPGA

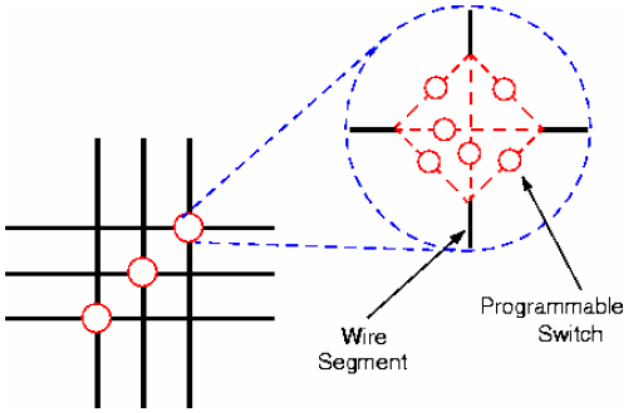
تعریف خطوط داخلی یابه عبارتی سیم کشی داخلی FPGAبه طور کامل از هم مجزا نیستند بلکه با استفاده از سویچهای این خطوط به هم راه می یابند .از این طریق می توان از تعداد خطوط کمتری برای اتصال میان بلوکها استفاده کرد .شکل شماره۷-۱ این مطلب را روشن تر بیان می دارد:



شکل شماره۷-۱

تعریف خطوط داخلی یابه عبارتی سیم کشی داخلی FPGA

سویچ هایی که برای استفاده در کانالها استفاده می شوند قابلیت برنامه ریزی داشته وباتوجه به برنامه وبلوک بندی آنها برنامه ریزی می شوند .اگر فرض کنیم که تعداد خطوط بین بلوکها ۳باشد می توان شکل این سوئیچها را به صورت شکل شماره ۸-۱ نشان داد:



شکل شماره ۸-۱ سوئیچ های قابل استفاده در کانالها

معمولا تکنولوپی ساخت چنین سوییچ هایی ازنوع Planerیا domain –based switch می باشد.

توجه داشته باشید در تغییر مسیر ها همانگونه که در شکل نیز نشان داده شده است سیم شماره ۱تنها می تواند به همان شماره از سیم(۱)در مسیر بعدی متصل گردد –از ۱به۲ در این سوییچ ها امکان پذیر نیست ومسیر اصلی همچنان حفظ می گردد.

چگونگی عملکرد و کاربرد FPGA

شاید تا به حال مدارهای منطقی را به وسیله گیتهای NOT،OR،AND ساخته اید.برای ساخت چنین مدارهایی ازقبیل شمارنده ها ،کنترل کننده ها ،و... .ابتدا باید تعریفی از مدار در دسترس باشد سپس با توجه به منطق اعداد دودویی یک جدول صحت برای مدار تشکیل می شود وحالتهای مختلف مورد بررسی قرار می گیرد سپس با توجه به جدول صحت مدار توسط گیتهای منطقی مانند NAND، NOT،OR،AND طراحی می شود پس از این مرحله نوبت به پیاده سازی مدار بر روی برد توسط آی سی های منطقی می رسد و همانطور که میدانید یکی از وقت گیر ترین وخسته کننده ترین مرحله ساخت یک مدار همین قسمت است. بعدازاین مرحله نوبت به تست مدار جهت اطلاع از درستی مراحل کارکرد مدار می رسد .اگر در یکی از مراحل قبل دچار اشتباه شده باشیم مطمئنا در مرحله تست مدار دچار اشتباه مشکل می شویم.در صورت اشتباه در مراحل قبل باید تمام مراحل را از آخر به اول یک به یک چک کنیم تا بتوانیم اشتباهات احتمالی موجود در نحوه بستن وسیم کشی مدار ،طراحی مدار رااز روی جدول صحت و درستی جدول صحت را برطرف کنیم . باتوجه به مطالب گفته شده حتما به این نکته اذعان خواهید داشت که بیشترین اشتباهات در مرحله سیم کشی وبستن مدار برروی برد پیش خواهد آمد.

ممکن است سیمی در جای اصلی وصل نشده باشد ویا ممکن است یک پایه به هیچ جا متصل نباشد ویا اشتباهات مشابه اینها ... از طرف دیگر می دانیم که هر چه مدار بزرگتر و پیچیده تر باشد اشتباهات بیشتر و عیب یابی مشکل تر خواهد بود .اینجاست که نقش آی سی های FPGA نمایان تر می شود .آی سی هایی که با داشتن انواع گیتهای مختلف درون خود بسیاری از مشکلات ناشی از عیب یابی مدارهای منطقی را برطرف کرده است.

قابلیت و توانایی های FPGA

اما آنچه که قابلیت و توانایی FPGAها رابالا برده است توانایی هایی است که پاره ای از آنها در زیر آمده است:

۱.امکان تعریف هریک از پایه های ICها به صورت ورودی یا خروجی یا هر دو.

۲.امکان تعریف وضعیت عملکرد هر پایه در هنگام استفاده یا عدم استفاده،به عنوان مثال عملکرد HIGH امپدانس (Z) در هنگام عدم استفاده ویا قرار گرفتن در یک وضعیت منطقی صفر یا یک در هنگام عدم استفاده.

۳.امکان تشخیص تغییرات سطوح یا لبه های پایین رونده یا بالا رونده منطقی اعمال شده به هر پایه.

۴.امکان برنامه ریزی چند باره از طریق پایه های برنامه ریزی JTAGیکی از استانداردهای برنامه ریزی (IEEEو تغییر معماری آن).

۵.امکان تغییر متناوب معماری داخلی با استفاده از سری های BOOTABLE که نقشه معماری آنها در یک حافظه خارجی نگهداری شده وبا تغییر آدرس برنامه ریزی می توان IC را با معماری جدید BOOTکرده و از آن استفاده کرد.

۶.امکان برنامه ریزی در مدار (ISP)که این قابلیت را به وجود می آورد تا بدون اعمال تغییرات سخت افزاری وتنها از طریق پورت برنامه ریزی JTAG ،معماری داخلی IC راتغییر داد.

۷.محدوده گستره ا ی از پایه های قابل استفاده در این ICها که از بسته های ۴۴پایه تا ۵۱۴ پایه و حتی بالاتر با حجم گیتی داخلی متفاوت که بسته به نیاز براساس میزان پیچیدگی داخلی و تعداد پایه های ICرا تغییر داد.

۸.ککهش حیرت انگیز حجم مدار و مجتمع سازی در ابعادی تنها به مساحت چند سانتی متر مربع.

۹.یکسان سازی عناصر طراحی واز میان بردن تمامی مشکلات ناشی از عدم تطابق استانداردهای مختلف.(LS,HC,s,AS,…)

۱۰.از میان بردن تمامی نویزهای ناشی از وجود قطعات مختلف ومجزا در مدار.

۱۱.ککهش چشمگیر توان مصرفی و اتلاف توان...

۱۲.افزایش سرعت پردازش وخطکهای انتشار به دلیل استفاده از فناوری پیشرفته ودستیابی به خطکهای انتشار تا ۴ns و فرکانس کلاک فرارتر از ۱۷۸مگاهرتز .

۱۳.کار بادو سطح ولتاژ۵v و۳.۳v جهت استفاده از آنها در دستگاههای قابل حمل مانند گوشی های موبایل.

۱۴.ضریب ایمنی صد در صد به دلیل عدم امکان دستیابی یه محتوای داخلی و عدم توان توصیف محتوای داخلی به دلیل انجام ساده سازی و فشرده سازی بسیار پیچیده.و بسیاری از قابلیتهای حیرت انگیز دیگر که امکان انجام یک طراحی مجتمع،کم حجم،بهینه و سریع را فراهم می آورد.

کارخانه های تولید کننده FPGA

گرچه شرکتهای بسیاری بسته های FPGAرا تولید می کنند اما از میان آنها در شرکت ALTERAو Xilinx از جمله عمده ترین تولید کنندگان این محصول هستند که از این میان شرکت Xilinx نوع دیگری از این بسته ها رابانام CPLD را تولید می کند که به صورت Bootable عمل می کنند.بدین معنی که داده های برنامه ریزی معماری داخلی خود را از یک حافظه ی خارجی خوانده وخود را پیکربندی کرده وسپس آماده کار می شوند.تمامی این محصولات با توجه به تعداد پایه ها وحجم پیچیدگی قابل برنامه ریزی در آنها براساس تعداد گیتهای داخلی در بازار موجود و قابل دسترس هستند .از جمله سری های پرقدرت وپر حجم آنها سری flex از محصولات شرکتALTERA که در نمونه ای از آن می توان یک CPU مدل ۴۸۶ را جای داد.

زبانهای برنامه نویسی FPGA

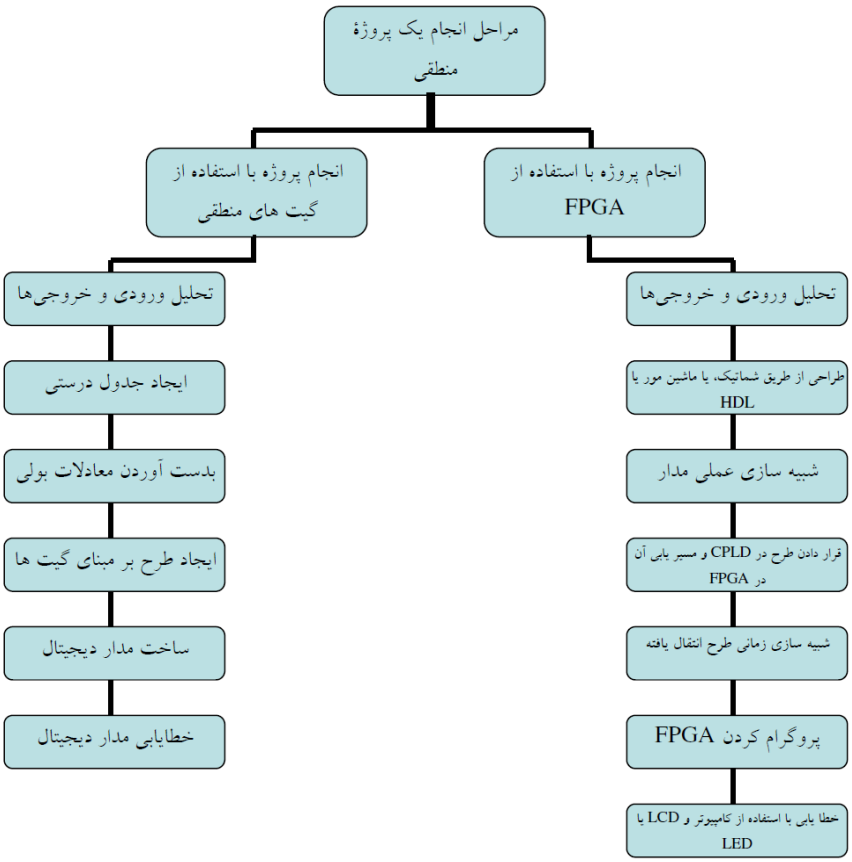
از جمله زبانهای متداول برنامه نویسی سخت افزار VHDL،VERILOG،AHDL،ABEL هستند که هریک با استفاده از syntaxخاص خود برای توصیف سخت افزار مورد استفاده قرار می گیرند که با استفاده از هریک از آنها می توان هر طراحی دیجیتالی را به زبان آنها نوشته وتحلیل وسپس استفاده کرد.

نرم افزارهای مربوط FPGA

معمولا هر یک از کارخانه های تولید این آی سی ،نرم افزاری را جهت استفاده از تولیدات معرفی می کند. معروفترین و پرکاربردترین این نرم افزارها مربوط به شرکت Xilinx می باشد.که کلیه مراحل اعم از شبیه سازی پردازش و پروگرام کردن درآن قابل اجرا می باشد.

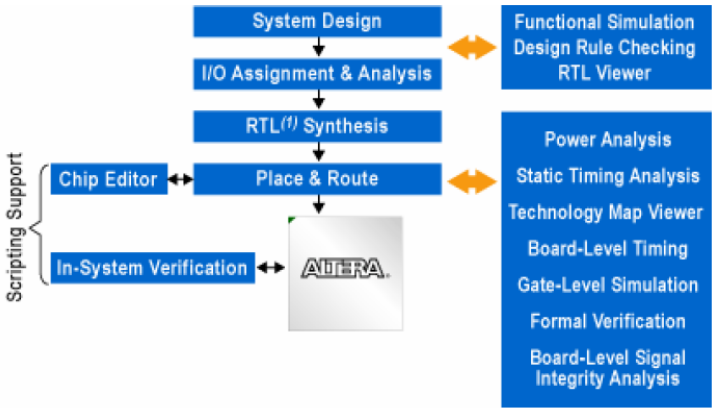
از نرم افزار دیگر می توان Protel DXP اشاره کرد که همه قابلیت های فوق را دارد علاوه برآن می تواند نقشه PCB آن را نیز ارائه دهد.

در فلوچارت شکل شماره ۹-۱ مقایسه بین روشهای انجام پروژه با استفاده از گیتهای منطقی ویا FPGA با هم مقایسه شده است.



شکل شماره ۹-۱ فلوچارت مقایسه بین روشهای انجام پروژه در FPGA

شرکت ALTRA نیز مراحل مختلف مربوط به استفاده از آی سی تولید خود را به صورت شکل شماره ۱۰-۱ بیان داشته است.



شکل شماره ۱۰-۱ مراحل مختلف مربوط به استفاده از آی سی ALTRA

فصل دوم
تشریح نقشه فنی پروژه و سفت افزارطراحی شده

آی سی XC9572

آی سی XC9572 در سیستم های برنامه نویسی پیشرفته عملکرد بسیار بالایی داردو قابلیت یکپارچه سازی سیستم های منطقی را دارا می باشد.

برای هر طراحی می توان شرایط عملیاتی خاص را با معدله زیر برای آی سی XC9572 محاسبه نمود:

$$I_{CC}\ (mA) = MC_{HP}\ (1.7) + MC_{LP}\ (0.9) + MC\ (0.006\ mA/MHz)\ f$$

Where:

MC_{HP} = Macrocells in high-performance mode

MC_{LP} = Macrocells in low-power mode

MC = Total number of macrocells used

f = Clock frequency (MHz)

از ویژگی های این آی سی می توان به موارد زیر اشاره نمود:

- ✓ تاخیر منطقی پین به پین در تمام پین های ۷.۵ نانو ثانیه
- ✓ ۱۶۰۰ گیت قابل استفاده
- ✓ منبع تغذیه ۵ ولتی و نگه داری ۱۰۰۰۰ برنامه و پاک شدن بصورت چرخه ای
- ✓ افزایش پین قفل معماری
- ✓ انعطاف پذیر بلوک تابع ۳۶V18

  کنترل سرعت بر روی خروجی ها

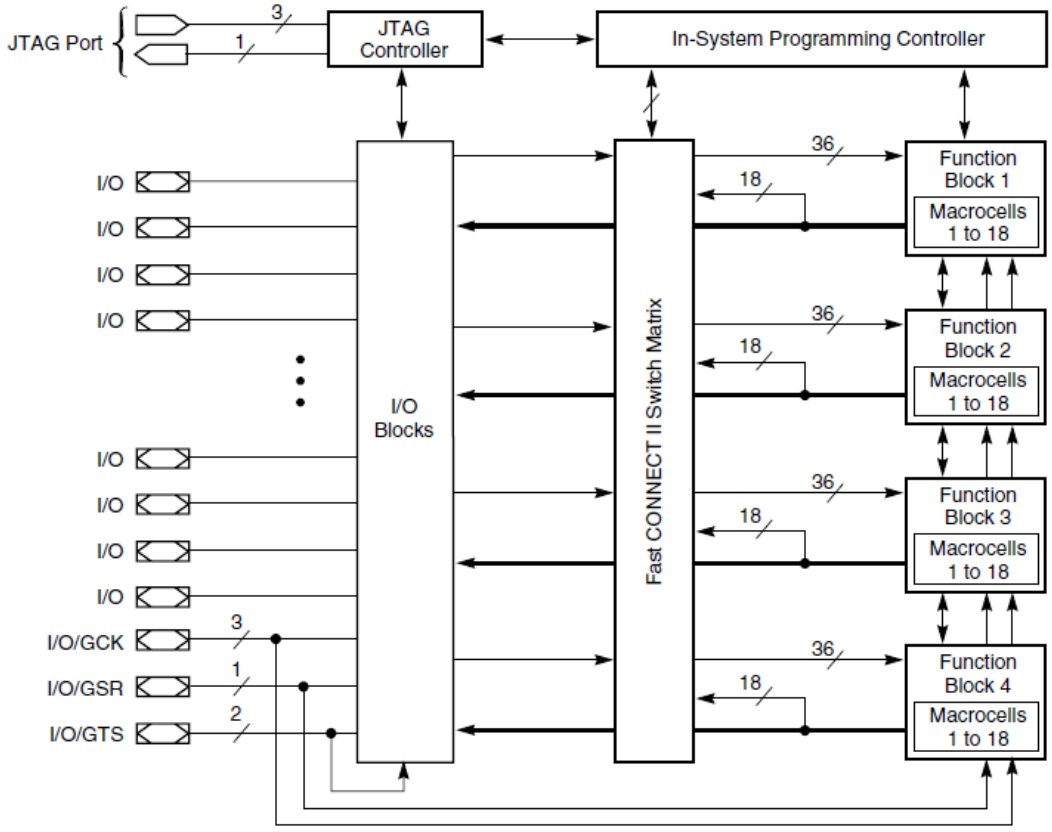
  برنامه ریزی کاربری

  حفاظت توسعه یافته امنیتی برای طراحی

  خروجی ۲۴ میلی آمپر خ
- ✓ قابلیت I / O ۵V یا ۳.۳V

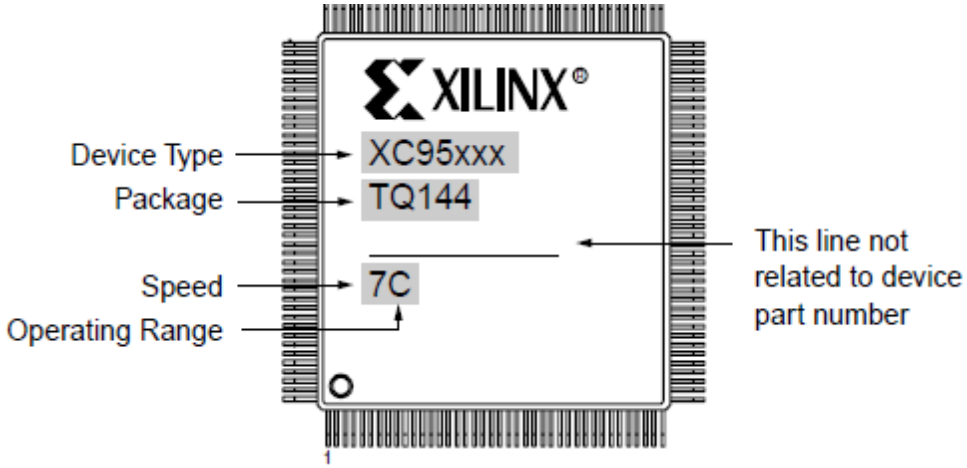
  فن آوری پیشرفته CMOS
- ✓ پشتیبانی از برنامه نویسی موازی
- ✓ و....

در شکل شماره ۱-۲ معماری داخلی آی سی XC9572 نشان داده شده است.



شکل شماره ۱-۲ معماری داخلی آی سی XC9572

در شکل شماره ۲-۲ اطلاعات قرار داده شده بر روی بدنه آی سی XC9572 نشان داده شده است.



شکل شماره ۲-۲ اطلاعات قرار داده شده بر روی بدنه آی سی XC9572

LCD کاراکتری ۱۶*۲

LCD ها ابزاری برای نمایش اطلاعات هستند که از کریستال مایع ساخته می شوند.کریستالهای مایع موادی هستند که ظاهر مایع دارند، اما مولکول های آنها آرایش خاصی نسبت به یکدیگر دارند، درست مانند جامدات، به همین دلیل کریستال مایع خصوصیاتی شبیه به مایع و جامد دارد.یک LCD دارای ۱۶ پایه بوده که ۸ خط آن مربوط به فرستادن یاخواندن داده ها یا دستورالعمل ها می باشد.

LCD ها در مقابل تغییرات دما عکس العمل نشان داده و به عنوان دماسنج طبی استفاده میشوند.انواع مختلفی از مواد شناخته شده اند که در دمای معمولی چنین خصوصیاتی دارند. اما دسته ای از آنها هستند که به جریان الکتریسیته هم حساس هستند و مولکولهای آن متناسب با جریان برق ورودی می چرخند و تغییر زاویه می دهند.

وقتی نور از درون یک کریستال مایع این چنین عبور کند، پلاریزاسیون یا قطبش آن هم جهت با مولکولهای کریستال می شود.از همین خاصیت برای LCD ها استفاده می شود.با این توضیح که چون کریستال های مایع شفاف وهادی الکتریسیته هستند، به راحتی می توان آنها را در جریان الکتریسیته قرار داد و نور را از آن عبور داد.

برای این کار به جز کریستال مایع به ۲ تکه از این شیشه پلاروید یا قطبشگر هم نیاز است.اگر دو تکه از این شیشه ها را روی هم قرار دهید نور به راحتی از آن عبور می کند.اما وقتی یکی از آنها را ۹۰ درجه نسبت به دیگری بچرخانید، دیگر نوری عبور نمی کند.این اتفاق به این دلیل روی می دهد که هر شیشه نور را فقط در جهت محور خود عبور می دهد.اگر دو شیشه هم محور باشند نور به راحتی عبور می کند اما اگر محورها با هم زاویه ۹۰ درجه داشته باشند نور عبور نخواهد کرد.

برای ساخت LCD دو شیشه پلاروید را با ۹۰ درجه اختلاف نسبت به یکدیگر قرار میدهند و یک کریستال مایع بین آنها قرار می دهند.وقتی کریستال به جریان برق وصل نباشد، نور از قطبشگر اول می گذرد و وارد کریستال مایع می شود جهتش ۹۰ درجه تغییرکرده و به همین دلیل از قطبشگر دوم هم عبور کرده و به چشم می رسد.اما وقتی که جریان به کریستال وصل باشد،نور دیگر چرخشی نخواهد داشت و نمی تواند از کریستال دوم عبور کند.

همچنین LCD ها ابزاری برای نمایش اطلاعاتی هستند که شامل حروف و اعداد و همچنین برخی کاراکترهای گرافیکی میشود. با بکار گیری LCD اطلاعات را بصورت زیبا وکاملتر میتوان نمایش داد. چیزی که از آن بعنوان LCD یاد می شود در واقع یک صفحه نمایشگر LCD مانند صفحه ماشین حساب است که همراه با آی سی کنترلر و مدارهای جانبی اش و عموما با لامپ پشت صفحه در یک بسته پیش ساخته عرضه می شود.

همانطور که گفته شد LCD دارای یک کنترلر است که با فرستادن اطلاعات به آن این اطلاعات رادر صفحه ای که عموما به چند سطر و ستون تقسیم شده نمایش می دهد.

LCD ها از طریق مقدار اطلاعاتی که میتوانند در صفحه نمایش بدهند انتخاب و خریداری می شوند.انواع معمول آن عبارتند از ۱۶ ، ۲۰ ، ۳۲ و ۴۰ کاراکتردر هر خط در ۱ یا ۲ یا ۴ سطر. مثلا ۲ در ۱۶ یعنی صفحه دارای دو خط و هر خط ۱۶کاراکتر است.همچنین LCD موردنظر میتواند همراه بالامپ پشت صفحه (light Back) یا بدون آن انتخاب شود.LCD ها کاراکتر ها را درماتریس های ۷x۵ pixel نمایش می دهند.

در کل دو نوع LCD وجود دارد . یکی از آنها را LCD کارکتری گویند که فقط قابلیت نمایش حروف و اعداد و کارکترهایی همچون ! و غیره را دارد و نوع دیگر LCD گرافیکی است که قابلیتهای LCD گرافیکی بعلاوه ی نمایش تصویر در آن جمع شده اند.

در تصویر زیر یک نمونه۱۶*۲ مشاهده می شود:

منظور از ۲ در ۱۶ بودن یک LCD این است.که LCD دارای ۲ ردیف است که هر ردیف آن دارای ۱۶ ستون است.

در شکل شماره ۳-۲ نمای واقعی یک LCD نشان داده شده است.

شماره ۳-۲

نمای واقعی یک LCD

بررسی پایه های LCD کاراکتری ۱۶*۲

همانطور که گفته شده LCD ۱۶*۲ یک LCD متنی است و دارای دو سطراست که هرسطر دارای ۱۶ مکان برای نمایش کاراکتر می باشد.

شماره پایه	سمبول	توضیحات
۱ ۲ ۳	VSS VDD V0	زمین منبع تغذیه ولتاژ5+ ولت منبع تغذیه ولتاژ کنترل کنتراست
۴	RS	اگر RS=0 باشد ثبات دستور انتخاب می شود و اگر RS=1 باشد ثبات داده انتخاب می شود.
۵	R/W	R/W=0 برای نوشتن در LCD R/W=1 برای خواندن از LCD
۶	E	فعال ساز
۷-۱۴	D0 – D7	بیت های ۰ تا ۷ دیتا
۱۵ ۱۶	- -	آنود لامپ LED پشت LCD کاتود لامپ LED پشت LCD

جدول شماره ۱-۲ بررسی پایه های LCD کاراکتری ۱۶*۲

ولتاژهای VSS و VDD تغذیه ی LCD را فراهم می کنند. ولتاژ VO ولتاژ کنتراست است که تنظیم میزان روشنایی کاراکترها را روی LCD به کمک ولتاژهای VSS و VDD و یک مقاومت متغیر 10K انجام می دهد.

در داخل LCD دو ثبات وجود دارد که توسط پایه RS انتخاب می شود.اگر RS=0 باشد ثبات دستور IR انتخاب می شود تا اطلاعات ورودی به عنوان فرمان مشخص شوند.LCD این اطلاعات را دریافت می کند و فرمان تعریف شده را اجرا می کند .

لیستی از این دستورات در جدول صفحه بعد موجود است:

در صورتیکه RS = 1 باشد ثبات داده DR انتخاب می شودتا کاربر بتواند اطلاعاتی را روی LCD بنویسد یا بخواند.

جدول دستورات به شرح جدول شماره ۲-۲ می باشد:

عملکرد فرمان	کد هگزادسیمال فرمان
صفحه نمایش پاک می شود	
مکان نما به محل اولیه بر می گردد	۱
مکان نما پس از نوشتن هر حرف یا عدد به چپ شیفت پیدا می کند	۲
مکان نما پس از نوشتن هر حرف یا عدد به راست شیفت پیدا می کند	۴
کاراکترها به راست شیفت پیدا می کنند	۶
کاراکترها به چپ شیفت پیدا می کنند	۵
کاراکترها و مکان نما خاموش می شوند	۷
کاراکترها خاموش و مکان نمای زیر خط ثابت روشن می شود	۸
کاراکترها روشن و مکان نما خاموش می شود	0A 0C
مکان نمای چشمک زن فعال می شود	0D
مکان نما به چپ شیفت پیدا می کند	۱۰
مکان نما به راست شیفت پیدا می کند	۱۴
کل به چپ شیفت پیدا می کند	۱۸
کل به راست شیفت پیدا می کند	1C
آدرس اولین کاراکتر سطر اول	۸۰ C0
آدرس اولین کاراکتر سطر دوم	۳۸
LCD به صورت دو سطری می شود	

جدول شماره ۲-۲

جدول دستورات

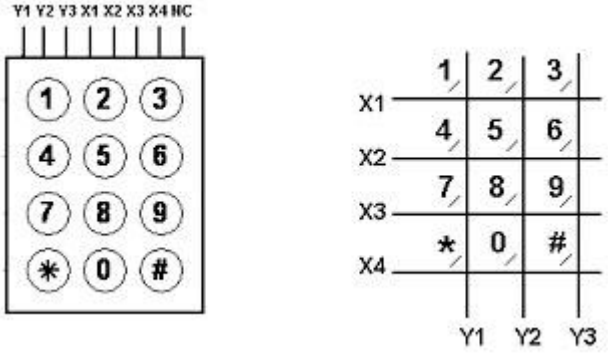
پایه پنجم پایه خواندن یا نوشتن است.برای نوشتن روی LCD , باید R/W=0 باشد و برای خواندن اطلاعات از LCD باید , R/W = 1 باشد.

پایه ۶ پایه فعال کردن (E) است.اگر در پایه E پالسی از ۱ به ۰ قرار داده شود, در این صورت اطلاعاتی که در پایه های ۷ تا ۱۴ قرار دارند در ثبات های LCD ذخیره می شوند.به عبارت دیگر در لبه منفی پالس ورودی به پایه E اطلاعات به LCD منتقل می شود.

پایه های ۷ تا ۱۴ , ۸ بیت اطلاعات ارسالی به LCD و یا دریافتی از آن می باشند. کد باینری دستورات و کد اسکی کاراکترها روی این پایه ها قرار می گیرند. پایه های ۱۵ و ۱۶ برای لامپ پشت LCD می باشند.

صفحه کلید ماتریسی

ساختمان داخلی این نوع صفحه کلیدها به صورتی که در شکل شماره ۴-۲ نشان داده شده است، می‌باشد. این شکل، یک صفحه کلید ۱۲ کلیدی را نشان می‌دهد. نمونه آن را در شماره گیرهای تلفن هم دیده‌اید. تنوع صفحه کلیدهای ماتریسی زیاد، ولی اصول کار اکثر آن‌ها یکسان است. در بعضی صفحه کلیدهای ماتریسی، ممکن است به جای X ها، به حرف R و به جای Y ها، به حرف C بر بخورید.



شکل شماره ۴-۲ ساختمان داخلی صفحه کلیدهای ماتریسی

وقتی هیچ کلیدی فشار داده نشده باشد در خطوط ۱Y، ۲Y، ۳Y و ۴Y مقدار '۱' منطقی می‌بینیم چون ۱Y تا ۴Y با مقاومت Pull-up به Vcc متصل شده‌اند و اگر تمام سطرها (۱X تا ۳X) را به GND وصل کنیم با فشرده شدن یک کلید در یک ستون، خط Y متناظر آن، '۰' منطقی می‌شود.

اصول کار این صفحه کلیدها به این صورت است که هر بار یکی از سطرهای آن را '۰' و بقیه را '۱' می‌کنیم. برای نمونه در ابتدای کار به ۱X، '۰' منطقی و به ۲X، ۳X و ۴X، '۱' منطقی می‌دهیم.

حال اگر کلیدی در سطر 1X (یکی از کلیدهای ۰، ۱، ۲ و ۳) فشار داده شده باشد، ستون متناظر آن '۰' خواهد شد و دیگر ستون‌ها '۱' می‌ماند.

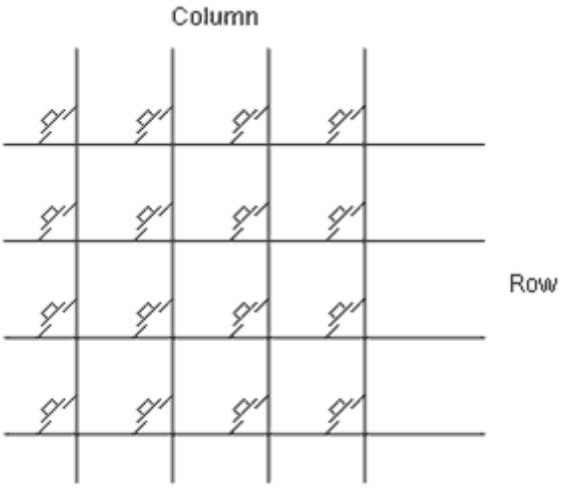
پس کافی است پس از '۰' کردن سطر 1X، ستون‌های 1Y تا 4Y را خوانده هرکدام '۰' شد، کلید متناظر آن را به عنوان کلید فشرده شده در نظر بگیریم.

اگر هر چهار ستون '۱' ماندند، در این سطر کلیدی فشرده نشده و سراغ سطر بعدی می‌رویم و همین کار را برای آن سطر انجام می‌دهیم. منتها برای این سطر، ستون‌های 1Y، 2Y، 3Y و 4Y متناظر کلیدهای ۴، ۵، ۶ و ۷ است.

این کار را برای تمام سطرها انجام می‌دهیم و دو باره به سطر اول بر می‌گردیم و این کار را به طور متناوب تکرار می‌کنیم.

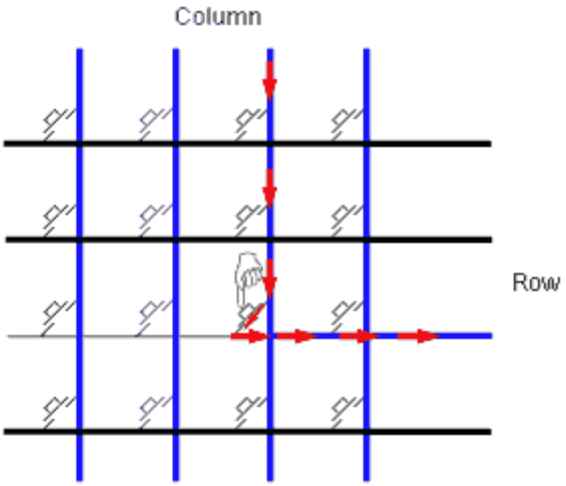
در ضمن اگر تشخیص داده شد که ستونی '۰' است باید یک مدت زمانی صبر کرد تا این که پارازیت‌های احتمالی به عنوان فشار کلید تعبیر نشوند. اگر پس از این مدت زمانی (چیزی حدود ۲۵۰$^{\mu s}$) باز هم ستون مربوط '۰'بود، می‌فهمیم که این سیگنال پارازیت نیست.

شکل شماره ۵-۲ یک صفحه کلید ماتریسی ۴*۴ را نشان می دهد .



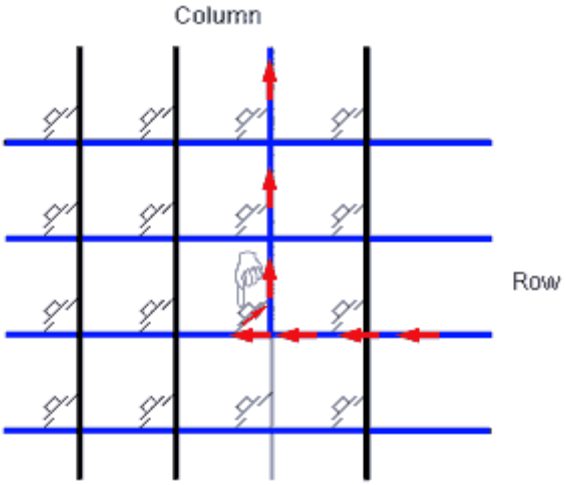
شکل شماره ۵-۲ یک صفحه کلید ماتریسی ۴*۴

در این روش ابتدا پینهای پورتی که به سطرها متصل است را ورودی و پینهای پورتی که به ستون ها متصل است را خروجی صفر قرار می دهیم . حال اگر کلیدی زده شود یک سطر و یک ستون به هم می چسبند در نتیجه آن سطر نیز توسط ستون کلید زده شده صفر خواهد شد . اکنون برای پیدا کردن سطر صفر شده کافی است پین های متصل به سطر را خوانده و شماره سطری که صفر شده است را بدست آوریم (توسط عمل چرخش) .



شکل شماره ۶-۲ یک صفحه کلید ماتریسی ۴*۴

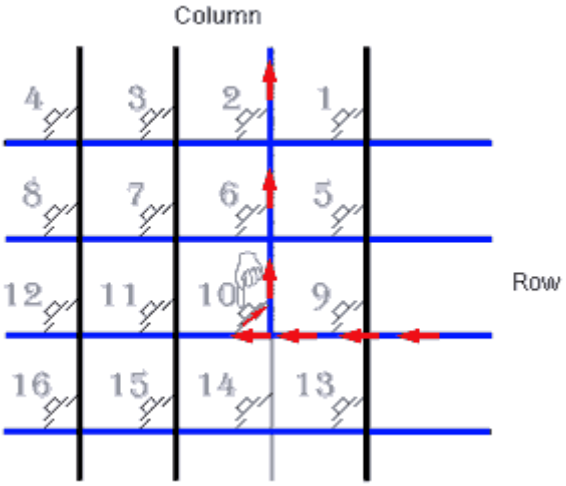
سپس جای پینهای ورودی و خروجی را عوض می کنیم یعنی پینهای پورتی که به سطرها متصل است خروجی صفر وپینهای پورتی که به ستون ها متصل است ورودی قرار می دهیم . همان طور که مشخص است ستون کلیدی فشرده شده صفر خواهد شد و با خواندن پینهای متصل به ستون ها و پیدا کردن پین صفر شده شماره ستون را نیز بدست می آورد .



شکل شماره ۷-۲ یک صفحه کلید ماتریسی ۴*۴

سپس با توجه به رابطه زیر شماره کلید زده شده را مشخص می کنیم . اعمالی که گفته شد بسیار سریع انجام می گیرد و درصورتی که اسکن کلیدها درون یک حلقه استفاده شود و کلیدی زده شود آی سی تصور می کند که شما چندین بار این کلید را فشرده داده اید برای رفع این مشکل یک تاخیر ایجاد کنیم و بعد از آن سطر را دوباره ورودی و ستون را خروجی می کنیم تا بتوانیم بار دیگر عمل فشردن کلید را تشخیص دهیم . فلوچارت زیر این اسکن کلید را نشان می دهد.

شماره کلید = (شماره سطر بدست آمده-۱) * کل ستون ها + شماره ستون بدست آمده

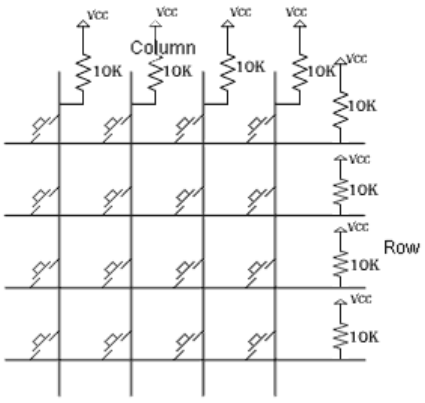


شکل شماره ۸-۲ یک صفحه کلید ماتریسی ۴*۴

مثلاً در شکل شکل شماره ۷-۲

(3-1)*4+2=10

دقت شود که در این مسئله فرض شده که پورت ها وقتی به عنوان ورودی انتخاب می شوند از مقاوت بالاکش داخلی پورت استفاده می کند در غیر اینصورت باید مقاومتهایی با ظرفیت K۱۰ در خطوط سطر و ستون اضافه شوند. که بهتر است از مقاومت آرایه ای استفاده کنید.



شکل شماره ۹-۲

یک صفحه کلید ماتریسی ۴*۴

سون سگمنت (seven segment)

برای نشان دادن اعداد در ساعتهای دیجیتالی ،چراغ راهنما، ماشین حساب ، ترازوی دیجیتالی و کلا" نمایش مقادیر سیستم ها و دستگاه ها و ... از یک قطعه به نام seven segment یا هفت قسمتی استفاده می کنند. که اغلب به رنگ سبز و قرمز هستند. شکل شماره ۱۰-۲ استفاده از چندین سون سگمنت جهت نمایشگر یک سیستم را نشان می دهد:

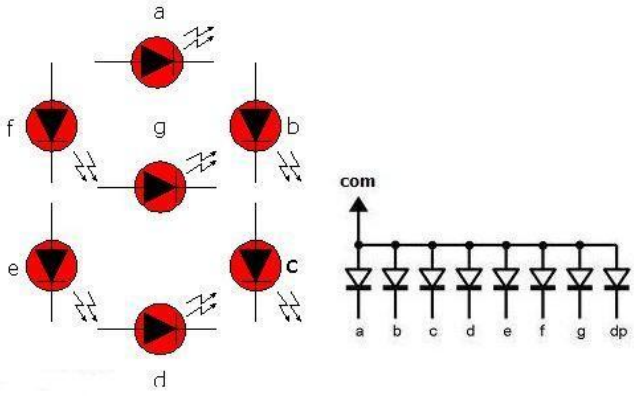
شکل شماره ۱۰-۲ استفاده از چندین سون سگمنت جهت نمایشگر

این قطعه در واقع هفت LED (دیود نورانی) می باشد که کنار هم بترتیب خاصی قرار گرفته اند و روشن یا خاموش بودن این LED ها اعداد را به ما نشان می دهد . البته امروزه با کاهش قیمت lcd ها و به علت مصرف پائین آن در دستگاه های مختلف از lcd بیشتر از گذشته استفاده میشود ولی همچنان سون سگمنت استفاده فراوانی در مدارت و دستگاه ها دارد.

سون سگمنت ها (seven segment) در دو نوع آند مشترک و کاند مشترک تولید می شوند که در عملکرد تفاوتی با هم ندارند و فقط جهت دیودهای نوری بکار رفته در آنها متفاوت می باشد و طراح مدار با توجه به نوع طرح و قطعاتی که استفاده کرده است می تواند از نوع آند مشترک یا کاند مشترک در مدار خود استفاده کند.

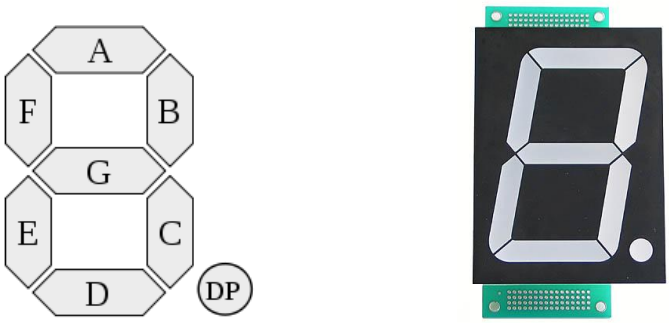
اگر هرکدام از این هفت قسمت را با حروف a b c d e f g در جهت عقربه های ساعت نام گذاری کنیم، آنگاه مثلا برای نمایش عدد "۱" کافیست که فقط قطعه های (سگمت های) b و c روشن بشوند.

شکل شماره ۱۱-۲ ساختار یک سون سگمنت آند مشترک را نشان می دهد:



شکل شماره ۱۱-۲
ساختار یک سون سگمنت آند مشترک

سگمنت های یک نمایشگر سون سگمنت مطابق شکل شماره ۱۲-۲ نام گذاری می شوند:



شکل شماره ۱۲-۲ سگمنت های یک نمایشگر سون سگمنت

آی سی بافر ULN2003

حتما تا به حال برایتان پیش امده که خواسته باشید بوسیله یک آی سی لامپ یا رله را روشن و خاموش کنید معمولا مستقیما نمیتوان آن را به مدار مربوطه وصل کرد و ممکن است در اثر کشیدن شدن جریان زیاد مدار آسیب ببیند. معمولا در این مواقع از یک ترانزیستور در حالت سوئیچ استفاده میکنند که به عنوان یک سوئیچ عمل میکند و خروجی مدار به بیس ترانزیستور وصل میشود و با تحریک بیس خروجی ترانزیستور بسته به نوع ترانزیستور خروجی تغییر حالت داده و جریان برقرار میشود که میتوان به کمک آن رله و یا مصرف کننده های دیگری را روشن و خاموش کرد و جریان مورد نیاز از ترانزیستور عبور می کند و تامین میشود و دیگر به مدار شما آسیبی نمیرسد .

اما اگر خواسته باشیم در مدار چندین خروجی داشته باشیم باید تعداد ترانزسیتورها را افزایش داد که باعث شلوغی و پیچیده تر شدن مدار میشود.

آی سی ULN2003 یک آی سی ۷ بیتی جریان بالا میباشد که برای آی سی های سری TTL و PMOS , NMOS , CMOS,هم سازگار شده ودر این پکیج از ۷ ترانزیستور جریان بالا استفاده شده است.

نحوه کار بدین صورت میباشد که خروجی های مدارتان را به ورودی این آی سی وصل می کنید و با تحریک شدن ورودی خروجی ها تغییر حالت داده و مدار شما روشن میشود.

البته ترانزیستور های این آی سی به صورت open collector میباشد و اگر به شکل هایی که در ادامه آورده شده است خوب دقت کنید یک دیود هم به تمام خروجی ها وصل شده و از مدار خارج شده است که باید به تغذیه مثبت مدار وصل شود .

یکی دیگر از کاربرد های آن بدین صورت میباشد که در صورتی که این پایه زمین شود تمام خروجی ها صفر میشود. البته باید توجه داشت این آی سی که در آن به صورت بافر میباشد و به صورت معکوس عمل میکند.

از دیگر مزایای این آی سی روبرو هم بودن ورودی ها و خروجی ها هست به طوری که مثلا اگر پایه ۱ که ورودی هست خروجی آن پایه شماره ۱۸ میباشد که روبرو آن در طرف دیگر قرار دارد.

لازم به ذکر است که آیسی ULN2003 یک بافر NOT است که با داشتن دیودهای محافظ مدار شما را از جریان برگشتی از موتور و رله محافظت می کند.

در درون آی سی ULN2003تعداد 7 ترانزیستور دارلینگتون NPN قرار دارد که در ورودی بیس هر کدام هم مقاومتی معادل ۲.۷ کیلواهم قرار داده شده که باعث میشود با اعمال یک ولتاژ ۵ولتی هر ترانزیستور را روشن نمود.

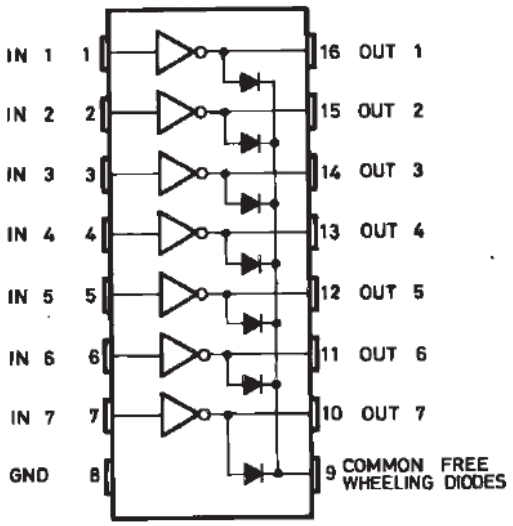
پایه های ۱ تا ۷ ورودی ها و پایه های ۱۶ تا ۱۰ هم خروجی های کلکتور هر ترانزیستور است. تمامی امیترهای ترانزیستورها هم به یکدیگر وصل شده و به پایه ۹ داده شده است به عنوان پایه زمین استفاده میشود.

استفاده از این آی سی هم مقرون به صرفه بوده و هم از تعداد قطعات روی برد کم کرده و در نتیجه باعث ساده تر شدن برد مدار چاپی میشود. جریان خروجی آن در حدود ۵۰۰ میلی آمپر است.

این آیسی بیشتر برای درایو کردن موتور پله ای و همچنین جهت درایو کردن خروجی‌های میکروکنترولر در مدارات مورد استفاده قرار می گیرد.

نکته عملی که در مورد این آی سی میتوان به ان اشاره کرد این است که خروجی سطح صفر آن کاملا زمین نیست و در حالت خروجی صفر، چیزی حدود ۶. تا ۷. ولت روی پایه های خروجی وجود دارد واین موضوع به دلیل دیود داخلی روی پایه ها است.

شماره پایه ها و نمای داخلی آی سیULN2003 به صورت شکل شماره ۱۳-۲ می باشد:



شکل شماره ۱۳-۲

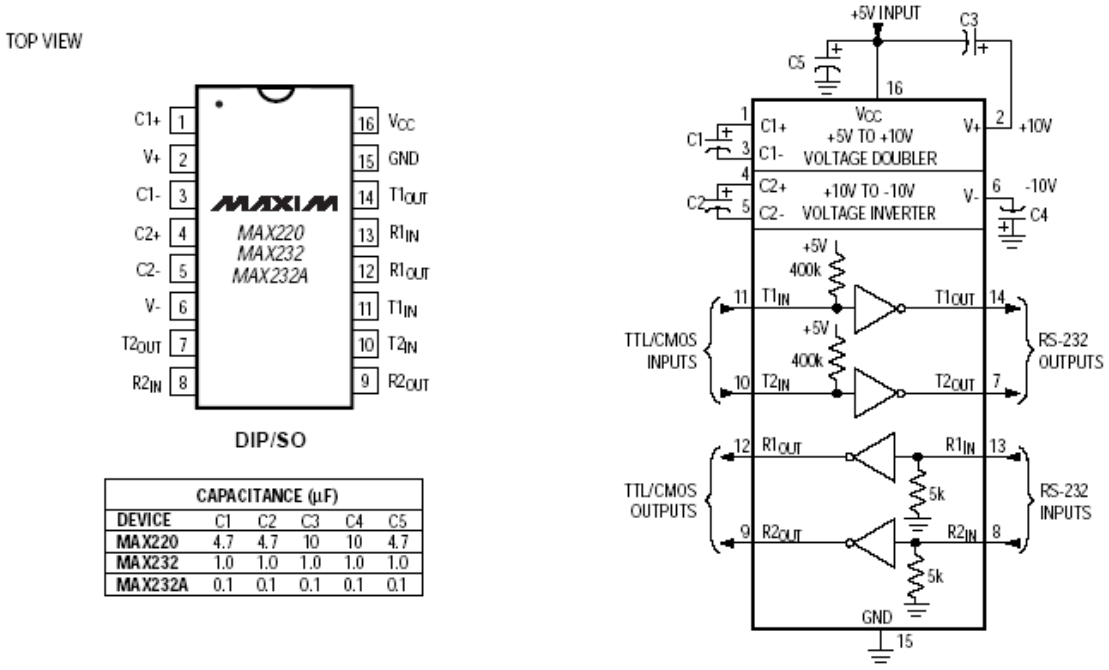
شماره پایه ها و نمای داخلی آی سیULN2003

پورت سریال و MAX232

ارسال داده های سریال دارای استاندارد RS232 می باشند و طبق این استاندارد ۰ منطقی دارای سطح ولتاژی بین (-۵تا-۱۵) و ۱ منطقی درای سطح ولتاژ (۵تا۱۵) می باشد در AVR-micro برای ارسال اطلاعات از سطح ولتاژ ۵ v (TTL-level) استفاده می شود، بنابراین سیگنال نیاز به تبدیل شدن دارد . این کار را می توان توسط MAX232 انجام داد که فقط نیاز به یک منبع ۵ ولت برای تبدیل سیگنال از (TTL-level) به RS232 و به عکس دارد و انتقال اطلاعات بین دووسیله RS232 می تواند با بیشترین فاصله (۱۵ meter) انجام گردد .

MAX232 IC یک مبدل TTL به RS232 است که سطوح ولتاژ TTL را به سطوح ولتاژ قابل قبول در استاندارد RS232 تبدیل می کند. این IC بر روی برد TB51 نصب شده و خروجی بر روی برد برای اتصال کابل سریال از طریق DB-9 در دسترس می باشد. پین شماره ۲ آنRxD و پین شماره ۳ آن TxD از میکروکنترلر است. پین شماره ۵ زمین است.

بر فرض مثال ما ۸ بیت اطلاعات را به دست آورده ایم و می خواهیم آن را برای کامپیوتر ارسال کنیم برای همین ار رابط سریال میکرو کنترلر استفاده می کنیم اما ولتاژ پورت سریال میکرو منطقی می باشد یعنی ۰ و ۵ ولی کامپیوتر با استاندارد RS232 یعنی ۱۰ و -۱۰ ولت کار می کند بنابر این از یک مبدل ولتاژ منطقی به RS232 استفاده می کنیم که همان آی سی Max232 می باشد که نقشه پایه های آن به صورت شکل شماره ۱۴-۲ است .



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

شکل شماره ۱۴-۲ نقشه پایه های Max232

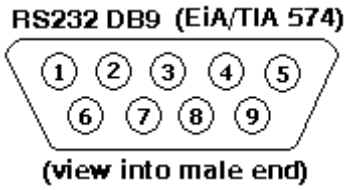
این آی سی به ۵ خازن برای ذخیره و رها سازی و رها سازی ولتاژ های ۱۰ و -۱۰ ولت نیاز دارد که مطابق شکل بالا وصل می شود. این آی سی امکان دارای ۲ ورودی و ۲ خروجی است که در اکثر اوقات فقط از یک ورودی و یک خروجی آن استفاده می شود .

برای وصل کردن پورت سریال به مدار از ۳ سیم ارسال می شود :

۱- گراند ۲ - RXT (دریافت) ۳- TXT (ارسال)

دقت کنید که سیم RXT از مدار به TXT کامپیوتر وصل شود و TXT از مدار به RXT کامپیوتر وصل شود.گراند پین شماره ۵ ، RXT پین شماره ۲ و TXT پین شماره ۳ می باشد در پورت سریال رایانه. گراند پین شماره ۵ ، RXT پین شماره ۳ و TXT پین شماره ۲ می باشد .

نمای پورت سریال به صورت شکل شماره ۱۵-۲ است .

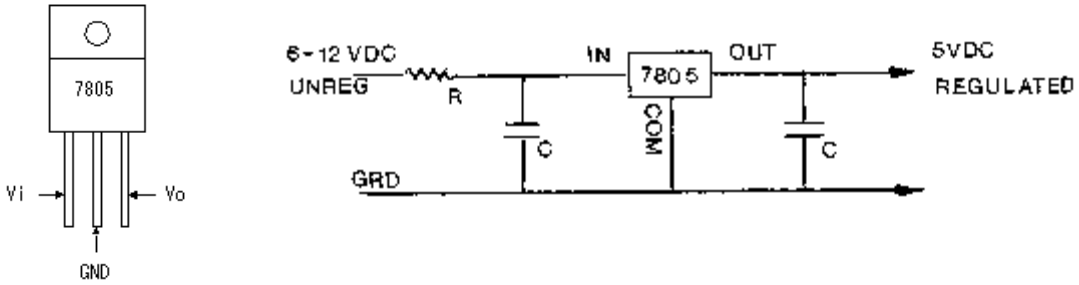


شکل شماره ۱۵-۲

نمای پورت سریال

رگولاتور ۷۸۰۵

این آی سی ها جهت تثبیت ولتاژ به ترتیب به میزان ۵ ولت جهت مصارف قطعاتی که این حد از ولتاژ برای آنها تعریف شده مورد استفاده قرار می گیرد .

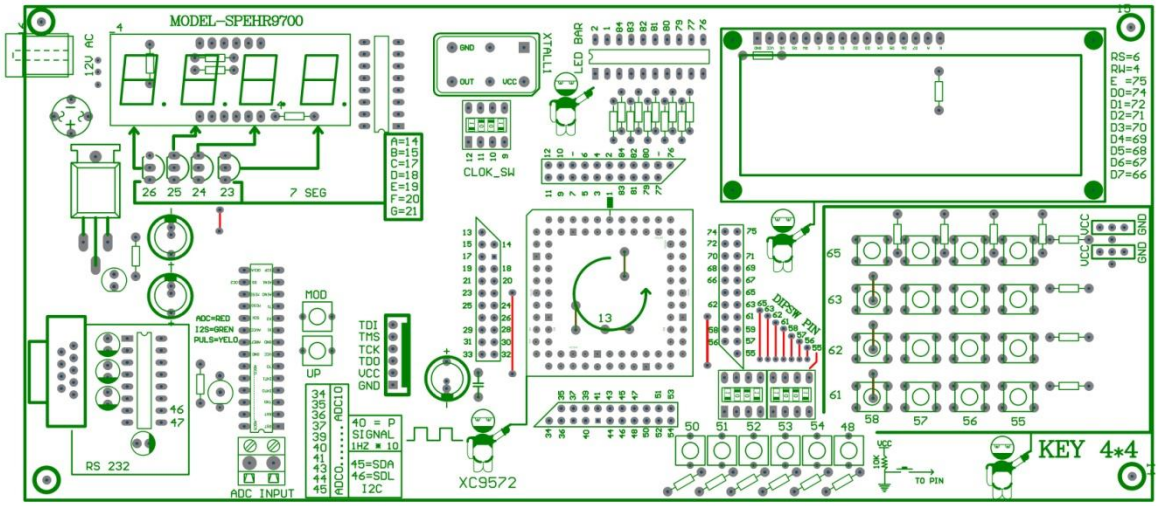


شکل شماره ۱۶- ۲ نقشه مدار داخلی آی سی ۷۸۰۵

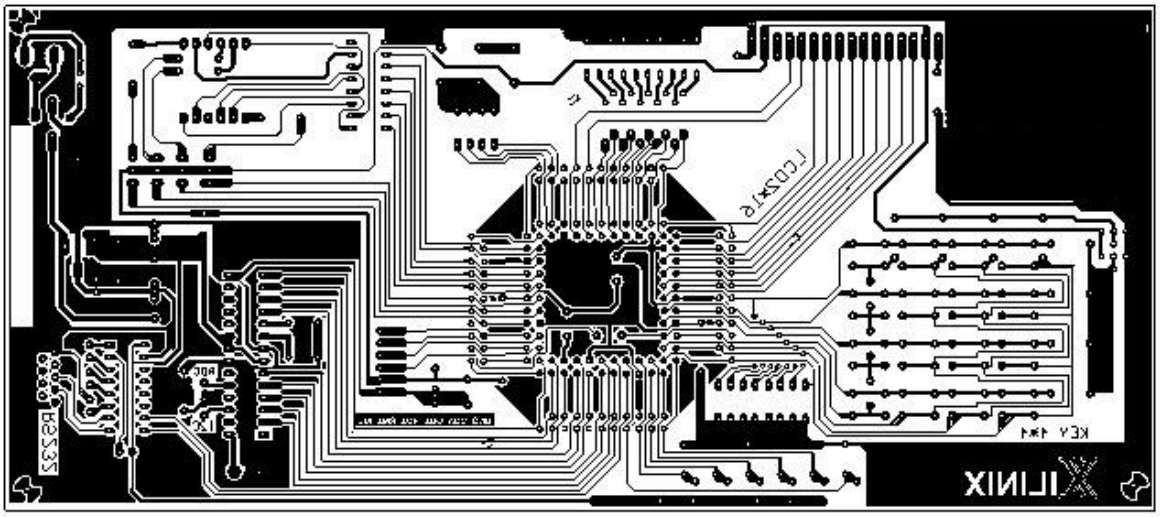
تغذیه ۵ ولت که نیاز به آمپر بالایی ندارد و برای ایجاد آن از رگولاتورهای معمولی استفاده نموده ایم و از آن جهت تغذیه و راه اندازی پردازنده استفاده نموده ایم.

تغذیه استفاده شده در پروژه باید به طور کامل صاف و رگوله باشد و هیچ گونه نویز و مزاحمتی برای عملکرد دستگاه ایجاد ننماید، به همین دلیل از خازن های صافی جهت از بین بردن هر گونه نویز در سیستم استفاده شده است.

نقشه جانمایی قطعات بر روی بورد



نقشه PCB مدار



فصل سوم

تشریح نرم افزار و برنامه های پروژه

درباره ی VHDL

با ورود FPGAها به بازار ادوات نیمه‌هادی، استفاده از زبانهای توصیف سخت‌افزار به منظور برنامه‌ریزی این ادوات بیش از قبل مورد توجه قرار گرفت. روشهای مختلفی برای پیاده‌سازی طرح مورد توصیف توسط تراشه‌های FPGA در دسترس می‌باشد که در این میان طراحی شماتیک، زبانهای توصیف سخت افزار و هسته های اینترنتی از معمولترین شیوه‌ها می‌باشند. زبانهای توصیف سخت‌افزار به دلیل انعطاف‌پذیری و قابلیتهای گسترده، بیشتر مورد استفاده واقع می‌شوند و در این میان دو زبان VHDL و Verilog بیشتر مورد توجه می‌باشند. زبان VHDL نخستین بار توسط وزارت دفاع امریکا به منظور طراحی و توصیف مدارت مجتمع سرعت بالا طراحی شد و مورد استفاده قرار گرفت. سپس در سال ۱۹۸۷ توسط انجمن IEEE در قالب استاندارد IEEE۱۰۷۶-۱۹۸۷ در اختیار عموم قرار گرفت.

به طور کلی می‌توان مزایای زیر را در استفاده از زبان VHDL عنوان نمود :

- ✓ با توجه به انکه VHDL یک زبان استاندارد می‌باشد، کد نوشته شده توسط آن را می‌توان به روی سنتزکننده‌ها و تراشه‌های تولیدکنندگان مختلف پیاده‌سازی نمود و نیازی به تغییر کد وجود ندارد .
- ✓ شبیه‌سازها و کامپایلرهای این زبان در دسترس و ارزان قیمت می‌باشند .
- ✓ با استفاده از این زبان می‌توان سیستمها را به صورت ساختاری یا رفتاری مدلسازی نمود، توصیف رفتاری نشان‌دهنده نحوه عملکرد سیستم و چگونگی تولید خروجیها بر اساس سیگنال‌های ورودی می‌باشد.
- ✓ با استفاده از این توصیف می‌توان عملکرد کلی سیستم را بیان کرد و از درگیر شدن با جزئیات بلوکهای سازنده سیستم که در طرحهای بزرگ به پیچیدگی توصیف سیستم منجر می‌شود اجتناب نمود. در مقابل مدل ساختاری نشان‌دهنده نحوه ارتباط بلوکهای سازنده سیستم است و بیانگر جزئیات بیشتری از ساختار سخت افزار می‌باشد . به این ترتیب با استفاده از این زبان امکان توصیف سخت افزار از سطح گیت تا سیستم فراهم می‌شود .
- ✓ با استفاده از توصیف ساختاری می‌توان سیستم های پیچیده را توسط ارتباط بین بلوکهای سازنده آنها مدلسازی نمود به این ترتیب پیاده‌سازی این سیستمها توسط زبان VHDL، ساده‌تر از زبان‌های برنامه‌نویسی نرم‌افزاری از قبیل C می‌باشد.

- ✓ با بکارگیری کتابخانه‌ها و component ها در زبان VHDL، می‌توان از المانهای موجود نوشته شده در سایر طراحی های استفاده نمود . در واقع عملکرد انها شبیه DLL ها و توابع در زبانهای برنامه نویسی نرم افزاری می باشد .
- ✓ سرعت طراحی و پیاده سازی سیستمهای پیچیده توسط این زبان بسیار بیشتر از طراحی شماتیک است زیرا جزئیات چگونگی اتصال گیتها و بلوکها، توسط نرم‌افزار سنتز کننده تعیین می‌شود، به این ترتیب می‌توان سیستمهای پیچیده را در مدت زمان کوتاهی پیاده‌سازی کرده تغییرات و اصلاحات مورد نیاز را در برنامه اعمال نمود.
- ✓ استفاده از این زبان بستر مناسبی برای شبیه‌سازی سیستم مورد توصیف ایجاد می‌کند و پس از اطمینان از صحت عملکرد کد نوشته شده در محیط شبیه‌ساز، می‌توان توصیف سیستم را به روی تراشه مورد نظر پیاده‌سازی نمود.

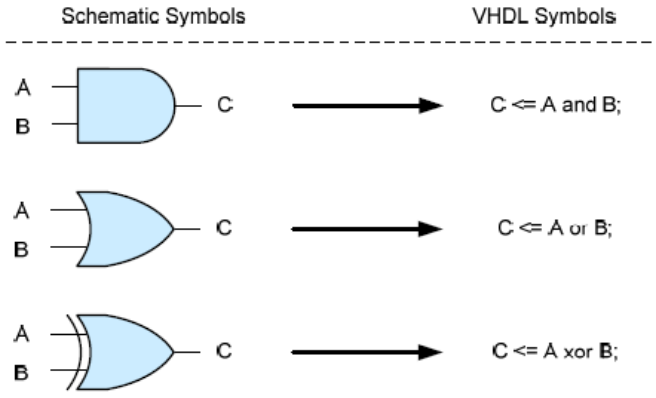
آشنایی با زبان های توصیف سخت افزار

با افزایش حجم سیستم های دیجیتال وپیچیدگی آنها ،استفاده از ابزارهای CAD اجتناب ناپذیر است. استفاده از کاغذ و قلم در طراحی دیگر چندان کاربرد ندارد .بیشتر افراد،طراحی به صورت شماتیکی را به سایر روشهای طراحی ترجیح می دهند.زیرا در شماتیک ارتباط بین بلوک های مختلف طراحی ،واضح تر می باشد .برای سالهای طولانی روش شماتیکی در طراحی ،بعنوان یک روش خوب مورد توجه بود.باوجود راحتی در استفاده از روش شماتیک ،استفاده از این روش دارای معایبی نیز می باشد.روش شماتیک تنها ارتباط بین بلوک ها ویاگیتها رانشان می دهدوویژگی های سیستم را به طور واضح مشخص نمی کند.همچنین باآمدن FPLDهایی که دارای چندین هزار(وحتی چندین میلیون)گیت منطقی هستند، بکاربردن واداره آنها ،درروش شماتیک بسیار مشکل است.

با بکاربردن زبان توصیف سخت افزاری (HDL)می توان بر این مشکلات فائق آمد.بعنوان مثال بیشترHDLها اجازه استفاده از ماشین با حالت محدود را برای مدارهای ترتیبی وجدول درستی برای مدارهای ترکیبی را می دهند.

زبانهای HDL،اولین بار در FPLDها بکاربرده شد.امروزه HDLهای مختلفی وجود دارد که از جمله مشهورترین آنها VHDL،VerilogوAbelمی باشد .HDLجهت توصیف سخت افزار با اهداف شبیه سازی ،تست،طراحی و یا مستند سازی بکار می روند.این زبانها یک نمایش سلسله مراتبی جهت نمایش جزییات سیم بندی ویاتابعی سیستم دیجیتال فراهم می سازند.بعضی از HDLها دارای یک مجموع های ازنمادها هستند که جایگزین نمودارهای موجود در شماتیک می باشند.به عنوان مثال ،شکل زیر نمونه ای از معادل نمادهای شماتیکی در VHDLرا نشان می دهد .طراحی با استفاده از HDLها می تواندبه صورت سلسله مراتبی باشد به گونه ای که هر بلوک ازچندین بلوک تشکیل شده باشد.

نمونه ای از معادل نمادهای شماتیکی در VHDL بصورت شکل شماره ۱-۳ نشان داده شده است.



شکل شماره ۳-۱

نمونه ای از معادل نمادهای شماتیکی در VHDL

نرم افزارهای موجود برایHDLها حاوی شبیه سازها وابزارهای سنتز سخت افزاری است.شبیه سازها بااستفاده از خود HDLویابه صورت شماتیکی برای شبیه سازی واعتبار سنجی طراحی انجام شده بکار میروند. وابزارهای سنتز نیز طراحی انجام شده با استفاده ازHDLرااز سطوح مختلف طراحی به سطح گیت منطقی تبدیل می کنند.

تاریخچه VHDL

در تابستان سال ۱۹۸۱وزارت دفاع ایالات متحده،کارگروهی ترتیب داد.هدف ازآن،مطالعه روشهای مختلف توصیف سخت افزار ،نیاز به یک زبان استاندارد وویژگیهای چنین زبانی بود تا بتوانند،ابزار استاندارد،برای طراحی ومستند سازی مدارهای مجتمع بسیار سریع (VHSIC) بوجود آورند.

این کار گروه وتمام دست آوردهای آن در آن سال،تحت انحصار ITARبود.درسال ۱۹۸۳ ،وزارت دفاع ایالات متحده یک زبان استاندارد توصیف سخت افزاری برای VHSIC(VHDL)رامطابق باآنچه که در سال ۱۹۸۱ بررسی شده بود،درخواست کرد.در تابستان ۱۹۸۳کاربر روی VHDLشروع شد.درآن زمان اگر چه VHDLدر انحصارITARنبود اما همچنان تحت انحصار ایالات متحده بود.

پس از شش ماه از شروع کار،VHDL 2.0منتشر شد.این نسخه دارای کمبودها ومشکلاتی بود ،پس از مدتی کوتاه ،نسخه اصلاح شده VHDL6.0منتشر شد ودرهمین زمان بود که توسعه ابزارهای مبتنی بر VHDLشروع شد.

درسال ۱۹۸۵،VHDLاز انحصار ITARدرآمدوبه IEEEواگذار شد تاتوسعه بیشتری یابد.براثر تلاشهایی که برای تعریف نسخه جدید VHDLدر سال۱۹۹۰انجام گرفت.درسال ۱۹۹۲یک نسخه جدید که بنام VHDL:93بود،کامل شدوبعنوان یک زبان استاندار برای طراحی،شبیه سازی ومستند سازی مدارهای دیجیتال در اختیار طراحان قرار گرفت .گسترش و توسعه این زبان همچنان ادامه دارد،به طوری که نسخه ای از ان بنام VHDL AMSبرای طراحی مدارهای آنالوگ بکار می رودکه هنوز تحت بررسی و مطالعه می باشد.

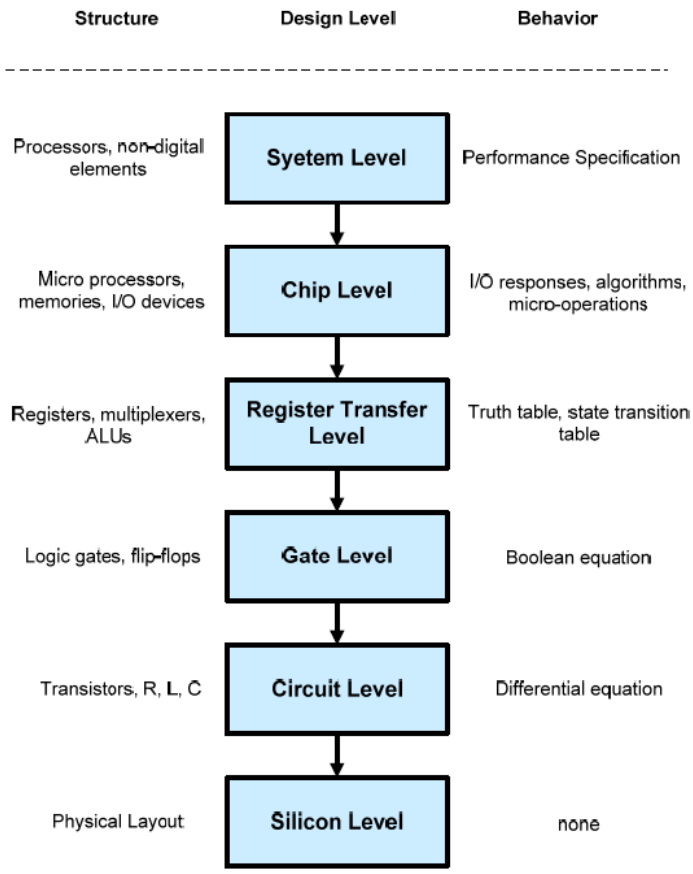
سطوح طراحی در VHDL

در طراحی یک سیستم دیجیتال سطوح مختلف طراحی مورد توجه است.بسته به این که طراحی در کدام یک از این سطوح انجام می شود نیاز به ابزار خود را دارد.این سطوح طراحی در صفحه ی بعد نشان داده شده است .هریک ازاین سطوح طراحی را می توان برحسب ساختار یا رفتار مورد تحلیل قرار دارد.نکنولوژی کنونی آنقدر پیچیده است که روشهای معمول طراحی ،کل سطوح طراحی را پوشش نمی دهند.باگسترش ابزارهای خودکار طراحی در سالهای اخیر می توان به راحتی از بالاترین سطح (سیستم)تا پایین ترین سطح(layout)پیش رفت.

طی کردن سطوح طراحی ،نیاز به سوییچ کردن از یک ابزار طراحی به ابزار دیگررا دارد. که این خود وابسته به این است که ابزارهای بکار رفته در مسیر طراحی سازگاری نیز بایکدیگر داشته باشند.

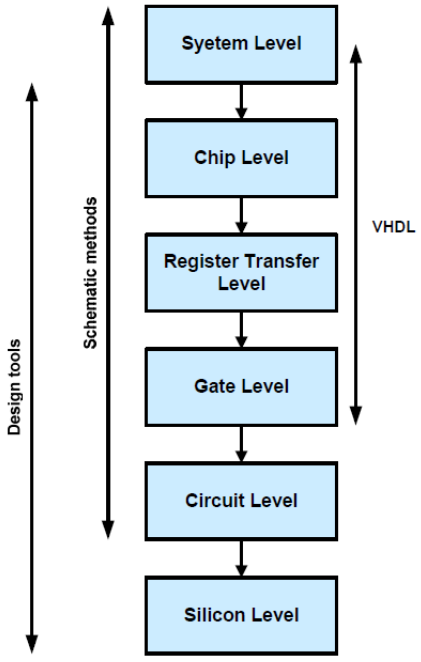
این مشکل را می توان تاحدودی توسط VHDLحل نمود.VHDLسطوح زیادی از طراحی را پوشش می دهد.اگرچه VHDLسطوح مدار و سیلیکون را پوشش نمی دهد اما وجود ابزارهای CADاین مشکل را حل نموده است.در VHDLیک سیستم می تواند هم به صورت ساختاری وهم به صورت رفتاری توصیف

شود.این قابلیت توانایی زیادی به طراح در هنگام پیاده سازی طرح خود می دهد. سطوح طراحی VHDL بصورت شکل شماره ۲-۳ نشان داده شده است.



شکل شماره ۲-۳
سطوح طراحی VHDL

جایگاه VHDL در سطوح طراحی بصورت شکل شماره ۳-۳ نشان داده شده است.



شکل شماره ۳-۳
جایگاه VHDL در سطوح طراحی

مقدمه ای بر VHDL

VHDL یک زبان توصیف سخت افزاری است که جهت طراحی یا شبیه سازی یک سیستم دیجیتال بکار می رود.پیچیدگی طرح مورد نظر می تواند ازچندین گیت تایک سیستم کامل دیجیتال باشد.VHDLیک زبان استاندارد صنعتی ،جهت توصیف سخت افزارازسطح انتزاع تاسطوح مختلف طراحی می باشد.

VHDLدارای قابلیتهایی است که با استفاده ازآنها می توان به توصیف رفتار (چه به صورت ترتیبی وچه به صورت همروند) ،یا ساختار هر سیستم دیجیتالی چه با زمان بندی یا بدون زمان بندی پرداخت.از انجایی که امروزه ،اززبانهای توصیف سخت افزاری بطور وسیعی استفاده می شود ،در نتیجه در این روش می تواند بعنوان یک روش مستند سازی مدارها توسط طراحان باشد تامدارها برای سایر طراحان قابل فهم باشد. برای CADهای سیستمهای دیجیتال ،VHDLبه عنوان یک طرح ورودی ،بسیار مناسب است این زبان از طراحی به صورت سلسله مراتبی ومتد بالا به پایین ومتد پایین به بالا به خوبی پشتیبانی می کند. طرحهای نوشته شده به زبان VHDLمی توانند توسط یک شبیه ساز VHDL،اعتبار سنجی شوند.

هنگامی که یک مدار دیجیتال باید طراحی شود ،در بالاترین سطح توسط یکentity ویک architecture توصیف می شود.همچنین برای بلوکهای متعلق به یک مدار نیز یک entity ویک architecture تعریف می شود.اعلان entity شامل لیستی از سیگنالهای واسط و نوع آنها است که می توانند به سایر ماژول ها یا دنیای خارج متصل شوند. در اعلان architecture،رفتار مدار و اجزاء متعلق به آن توصیف می شود.

رفتار می تواند به صورت ساختاری (به عنوان مثال اتصال چندین بلوک به هم)ویا به صورت مجموعه ای از عبارتهای ترتیبی و یا همروند توصیف شود.

فایل حاوی دستورالعمل ها ی VHDLشامل دو قسمت است:یکی Entity و دیگری Architecture.

مانند هرزبان برنامه نویسی ،VHDLنیز دارای تعدادی قوانین وخصوصیات است که بعضی از آنها در زیر آمده:

- ✓ تعدادی کلمات رزرو شده وجود دارد که دارای معنی مشخصی هستند ونمی توانند برای نام گذاری شناسه یا هر نامگذاری دیگری بکار روند
- ✓ در VHDL،عبارتهای طولانی می توانند در یک یاچند سطر ادامه پیداکنند.بنابراین سر سطر رفتن ویا جای خالی به هر تعداد مشکلی بوجود نمی آورد.
- ✓ خطوطی که با دوعلامت منها (--)شروع می شوند به عنوان توضیح بشمار می روندو توسط ابزار سنتز نادیده گرفته می شود.
- ✓ علامت نقطه ویرگول (؛) بعداز هر عبارت بعدی ضروری است.
- ✓ کلمه TIME یک type از پیش تعریف شده است .واحدهای زمانی که در VHDLوجود دارند، در جدول شماره۱-۳ آورده شده است.

معادل در VHDL	توضیح
fs	Femtosecond
ps	Picosecond
µs	Microsecond
ms	Millisecond
s	second
min	Minute
hr	hour

جدول شماره۱-۳

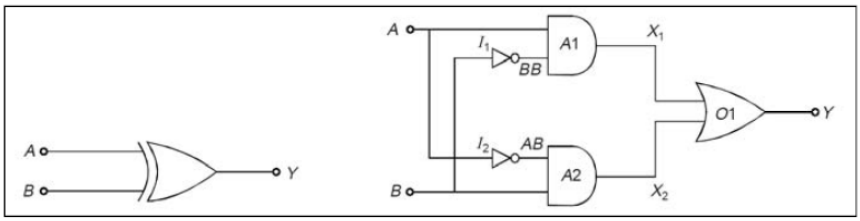
جدول واحدهای زمانی موجود در VHDL

- ✓ یک شناسه در VHDLمی تواند از یک یا چندین حرف ،عدد ویا کاراکتر (_) تشکیل شود.اولین کاراکتر باید حرف باشد.
- ✓ VHDLبه بزرگ و کوچک بودن حروف حساس نیست.
- ✓ عملکردهای دودویی مانندAND،OR،NOT،NAND،NOR،XOR،وxnor به طور پیش فرض در VHDLوجود دارد.
- ✓ در VHDL کتابخانه استانداردی وجود دارد که توسط IEEEاستاندار شده است که گیتهای منطقی استاندارد وبعضی توابع خاص را در اختیار طراح می گزارند.

اعلان Entity

عبارت entity در VHDL،به هر بلوک دیجیتال که دارای یک منطقی است که می تواند با دنیای خارج خود از طریق سیگنال های واسطی ،ارتباط برقرار کند،الحاق می شود.در واقع آن یک تجرید سخت افزاری از یک سیستم دیجیتال واقعی است.

یک entity ممکن است خود از چندین entity در سطوح پایین تر تشکیل شده باشد .به عنوان مثال یک گیت منطقی یای انحصاری از entityهای AND،OR، وnot مطابق شکل شماره ۴-۳ تشکیل شده است.



شکل شماره ۴-۳ گیت منطقی یای انحصاری از entityهای AND،OR، وnot

در VHDL تمامی طرح ها از entity استفاده می کنند .یک entity می تواند به عنوان یک بلوک در entity های دیگر مورد استفاده قرار گیرد ویا خود می تواند از چندین entity تشکیل شده باشد.

همان طور که بیان شد ،VHDL از روش طراحی پایین به بالا و روش بالا به پایین پشتیبانی می کند.این روشها به عنوان طراحی سلسله مراتبی به حساب می آیند که برای بررسی مدارهای پیچیده بسیار مناسب است.

عبارت ENTITY یک کلمه رزرو شده در VHDL است که به شروع یک entity اشاره دارد. هر اعلان Entity ، سیگنالهای یکتایی را برای ارتباط با دنیای خارج تعریف می کند.این اعلان،تعداد گذرگاهها جهت گذرگاهها ونوع آنهارا مشخص می کند.بعضی اطلاعات مانند زمانبندی می تواند در این اعلان قرار گیرد.

یک entity دارای یک اسم و اعلان مربوط به سیگنالهای ورودی و خروجی است.سیگنالهای ورودی خروجی بنام پورت شناخته می شوند.یک پورت توسط کلمه رزرو شده PORT مشخص می شود.هر سیگنال ورودی یا خروجی دارای خصوصیتی بنام mode است که مشخص می کند آن سیگنال ورودی (توسط کلمه رزرو شده IN)،خروجی (توسط کلمه رزرو شده OUT) ویا باس دوطرفه (توسط کلمه رزرو شده INOUT)است. هر یک از این سیگنالها دارای typeهستند.گرامر اعلان یک entity به صورت زیر است:

```
ENTITY entity-name IS
  PORT (لیست نام ورودی‌ها و خروجی‌ها به همراه نوع آن‌ها);
END entity-name;
```

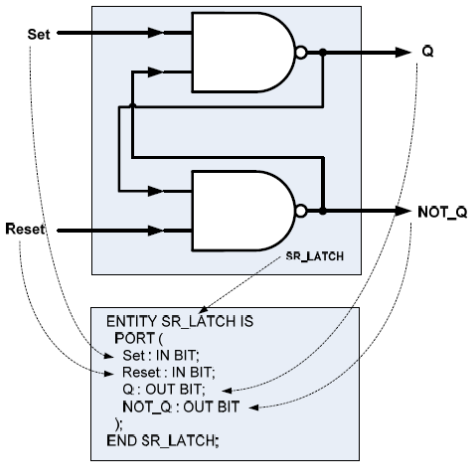
کلماتی که با حروف بزرگ نوشته شده اند جزءکلمات رزرو شده VHDL می باشند.منظور از Entity-name، نام entity است که معمولا این نام یک کلمه معنی دار و مربوط به عملکرد آن entity می باشد.داخل اعلان PORTورودی / خروجی همراه با نوع و modeآنها مشخص می شود. به عنوان مثال یک entity با دو ورودی AوBو یک خروجی Y که دارای نام my-xorمی باشد به صورت زیر نوشته می شود:

```
ENTITY my_xor IS
   PORT (
      A : IN BIT;
      B : IN BIT;
      Y : OUT BIT
   );
END my_xor;
```

Type ورودی های A و Bو همچنین type خروجی Yاز نوع BIT (۰یا ۱) می باشد. Mode این سیگنالها یا IN یا OUT مشخص شده است که نشان می دهد ،ورودی یا خروجی اند.کلمه رزرو شده END، پایان اعلان ENTITY را مشخص می کند.

مطابق آنچه که در مثال بالا آورده شده است،واضح است که اعلان entity تنها سیگنالهای ورودی و خروجی را مشخص می کند وهیچ جزییاتی در مورد رفتار entityیا اینکه بلوکها چگونه بهم متصل شده اند ،نمی دهد.در واقع اعلان entity نمی تواند رفتار یک بلوک را توصیف کند وتنها به عنوان یک نمای خارجی هسته می باشد.شکل زیر اعلان entityاز یک Set/Reset Latch را نشان می دهد.همانطور که مشاهده می شود،اعلان entity ای Set/Reset Latch مشخص نمی کندکه ساختار داخلی این latch به چه صورت می باشد.

شکل شماره ۵-۳ یک SR latch و معادل نمای بیرونی آن در VHDL را نشان می دهد.



شکل شماره ۵-۳ یک SR latch و معادل نمای بیرونی آن در VHDL

در زیر اعلان entity برای گیتهای منطقی AND،OR و NOT آمده است:

```
-- Entity declaration for AND GATE
ENTITY and_gate IS
  PORT ( A , B : IN BIT; Y : OUT BIT);
END and_gate;

-- Entity declaration for OR GATE
ENTITY or_gate IS
  PORT ( A , B : IN BIT; Y : OUT BIT);
END or_gate;

-- Entity declaration for NOT GATE
ENTITY not_gate IS
  PORT (A : IN BIT; Y OUT BIT);
END not_gate;
```

همانطور که در اعلان فوق مشاهده می شود ،می توان ورودی ها ویا خروجی ها را یکجا تعریف کرد .به عنوان مثال برای اعلان and – gate و یا or- gate ورودیهای Aو B یکجا تعریف شده اند.

قسمت Architecture

یک طرح که به زبان VHDLتوصیف می شود،از یک اعلان entity و حداقل یک بدنه architecture استفاده می کند. اعلان entity نمای بیرونی از یک بلوک را توصیف می کند؛در صورتی که بدنه architecture حاوی ساختار و یا توصیف رفتار آن ،می باشد .توصیف داخلی می تواند به یکی از راههای زیر مشخص شود:

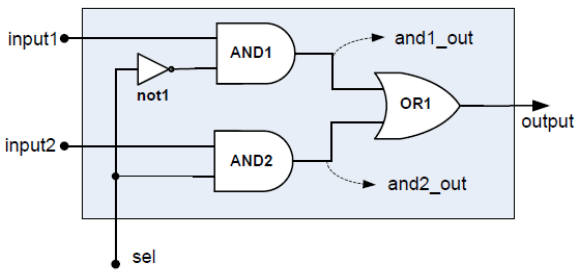
- ✓ مجموعه ای از بلوک ها که توسط یک سری سیگنال به یکدیگر متصل شده اند.(مدل ساختاری).
- ✓ مجموعه ای از عبارتهای همروند که رفتار entityرا مشخص می کند.(مدل جریان داده).
- ✓ مجموعه ای از عبارتهای ترتیبی که رفتار entityرا مشخص می کند(مدل رفتاری).
- ✓ ویا می تواتد بدنه architecture ،ترکیبی از سه روش قبلی باشد(مدل ترکیبی).

یک بدنه architecture از دو قسمت تشکیل شده است .قسمت اعلانی و قسمت عبارتها. گرامر بدنه architecture بصورت زیر است:

```
ARCHITECTURE architecture-name OF entity-name IS

[قسمت اعلانی]

BEGIN

[قسمت عبارت‌ها]
```

کلماتی که با حروف بزرگ نوشته شده است جزءکلمات رزرو شده VHDL می باشد بنابراین نمی توان از آنها به عنوان شناسه یا اسم یک entityاستفاده کرد .قسمت اعلانی که قبل از کلمه کلیدی BEGIN وجود دارد ، برای اعلان سیگنالها ،نوعهای تعریف شده توسط کاربر و اعلان بلوکها استفاده می شود.قسمت عبارتها بین دو کلمه کلیدی BEGIN و END قرار می گیرد.

شکل شماره۶-۳ ساختار یک مالتیپلکسر ۲ به ۱ را نشان می دهد.می توان این منطق ترکیبی را به یکی از چهار روش :مدل ساختاری،مدل جریان داده،مدل رفتاری و مدل ترکیبی توسط زبان VHDLتوصیف نمود.



شکل شماره۶-۳
ساختار یک مالتیپلکسر ۲ به ۱

قبل از هر چیز باید اعلان entity فوق نوشته شود که در زیر آمده است:

```
ENTITY mux2to1 IS
  PORT (
    input1 : IN BIT;
    output2 : IN  BIT;
    sel : IN BIT;
    output : OUT BIT
  );
END mux2to1;
```

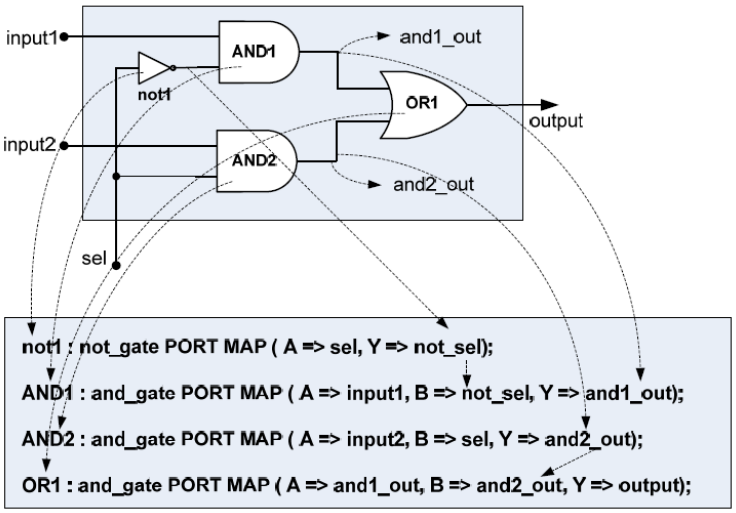
مدل ساختاری

در توصیف مدل ساختاری در زبان VHDL از اتصال بلوکهای سازنده مدار استفاده می شود.به عنوان مثال برای توصیف مالتیپلکسر صورت ساختاری باید از اتصال گیتهای منطقیAND،OR، و NOT استفاده کرد. در زیر این نوع توصیف برای مالتیپلکسر شکل فوق آمده است:

```
ARCHITECTURE structural OF mux2to1 IS

  COMPONENT and_gate
    PORT ( A, B : IN BIT; Y : OUT BIT);
  END COMPONENT;

  COMPONENT or_gate
    PORT ( A, B : IN BIT; Y : OUT BIT);

END COMPONENT;

COMPONENT not_gate
  PORT ( A : IN BIT; Y : OUT BIT);
END COMPONENT;

-- interconnection signal declaration
SIGNAL not_sel    : BIT;
SIGNAL and1_out : BIT;
SIGNAL and2_out : BIT;

BEGIN

  not1 : not_gate PORT MAP ( A => sel, Y => not_sel);

  AND1 : and_gate PORT MAP ( A => input1, B => not_sel, Y => and1_out);
```

مطابق آنچه که در صفحه قبل آمده است، بلوکهایی که در ساختار مالتیپلکسر وجود دارد ،معرفی شده است. این معرفی با استفاده از کلمه کلیدی COMPONENT در قسمت اعلانی بدنه architecture انجام میشود. سپس سیگنالهای که در اتصال داخلی بلوک های استفاده شده است یا استفاده از کلمه کلیدی SIGNAL به همراه نوع آنها تعریف می شوند.این سیگنالهای محدود به بدنه architecture هستند و از بیرون بدنه قابل دسترسی نمی باشند .

در قسمت اعلانها که بعد از کلمه کلیدی BEGIN در بدنه architecture قرار دارد ،به اتصال بلوکها مطابق شکل شماره ۷-۳ می پردازیم.



شکل شماره ۷-۳ اتصال بلوکها در VHDL

عمل عضو گرفتن بایک اسم برای هر بلوک شروع می شود که قبل از یک علامت دو نقطه (:) قرار می گیرد. سپس نام component ای که مورد نیاز است ،می آید. همانطور که در شکل فوق مشاهده می شود تنها بلوک های سازنده به یکدیگر متصل شده اند و رفتار انها در بدنه architecture نیامده است .سیگنالها به ورودی / خروجی بلوک ها توسط کلمات کلیدی PORT MAP نگاشت می شوند.

بنابراین در توصیف ساختاری کافیست که سیگنالهای داخلی تعریف شوند و سپس بلوک های داخلی به یکدیگر متصل شوند .این نوع توصیف در VHDL برای طراحی بصورت سلسله مراتبی بسیار مناسب است.

مدل جریان داده

در این توصیف جریان داده از میان entity با استفاده از عبارتهای همروند انتساب سیگنال بیان می شود.همانطور که از اسم این نوع توصیف مشخص است،عبارتهاب همروند مقادیری را به سیگنالها نسبت می دهند .در این نوع توصیف ،هنگام شبیه سازی ،تمامی عبارتها به صورت همروند اجرا میشوند.به عبارت دیگر بر خلاف زبانهای برنامه نویسی که به صورت سریال اجرا می شوند ،عبارتهای همروند به طور موازی اجرا می شوند .هر عبارت همروند توسط ابزار سنتز به گیتهای منطقی معادل تبدیل می شود.انتساب یک سیگنال در VHDL به صورت زیر است:

$$A <= B;$$

عبارت فوق بدین معناست که A مقدار کنونی B را می گیرد.به عبارت دیگر مانند آنست که دو سیم به هم متصل شوند و جهت جریان از سیم B به A باشد.جهت شبیه سازی ،تاخیرهای زمانی را نیز می توان در این عبارت های همروند اعمال کرد .مانند:

$$A <= B \text{ AFTER } 10 \text{ ns};$$

باید توجه داشت که عبارت فوق تنها برای شبیه سازی است و قابل سنتز به گیت منطقی نیست و ابزارهای سنتز نمی توانند ای عبارت را تبدیل به گیتهای منطقی کنند ،بنابراین در یک طراحی واقعی نباید این تاخیرها اعمال شود و تنها هنگام شبیه سازی باید اعمال شود.در زیر مدل جریان داده مالتی پلکسر آورده شده است:

```
ARCHITECTURE data_flow OF mux2to1 IS
  -- Signal declaration
  SIGNAL and1_out : BIT;
  SIGNAL and2_out : BIT;

BEGIN

  and1_out <= NOT ( sel ) AND input1;

  and2_out <= sel AND input2;

  output <= and1_out OR and2_out;
```

برای شبیه سازی می توان تاخیرها را نیز اعمال نمود .به عنوان مثال فرض کنیم قرار است که این مالتیپلکسر بر روی تکنولوژی ای پیاده سازی شود که هر گیت آن دارای تاخیر 5ns باشد.در این صورت معادل بدنه architecture فوق به صورت زیر می شود:

```
ARCHITECTURE data_flow_sim OF mux2to1 IS

  -- Signal declaration
  SIGNAL not_sel    : BIT;
  SIGNAL and1_out : BIT;
  SIGNAL and2_out : BIT;

BEGIN

  not_select <= NOT ( sel ) AFTER 5 ns;

  and1_out <= not_sel AND input1 AFTER 5 ns;

  and2_out <= sel AND input2 AFTER 5 ns;

  output <= and1_out OR and2_out AFTER 5 ns;

END data_flow_sum;
```

مدل رفتاری

در این نوع از توصیف ،رفتار entity مورد نظر ،توسط عبارتهایی بیان می شود که به صورت سریال اجرا میشوند. این نوع توصیف یا عبارت process صورت می گیرد.رفتار یک entity به صورت یک الگوریتم در عبارت process بیان می شود.

عبارت process یا کلمه کلیدی PROCESS شروع و با کلمات کلیدی END PROCESS خاتمه می یابد .تمام عباراتی که در بین کلمه کلیدی PROCESS و END PROCESS هستند جزء عبارت process به حساب می آیند . عبارت process از سه قسمت تشکیل شده است:لیست حساس ،قسمت اعلانی،و قسمت عبارات.

- ✓ لیست حساس: یک عبارت process همیشه فعال است ودر تمامی زمانها اجرا می شوند مگر آنکه متوقف شود.بعد از کلمه کلیدی PROCESS ،لیست(سیگنالهای) حساس در داخل پرانتز مشخص می شوند.هنگامی که روی یکی از سیگنالهای حساس اتفاقی روی دهد،process برانگیخته خواهد شد .هنگامی که اجرا به آخرین عبارت ترتیبی رسید ،process تا اتفاق دیگری بر روی سیگنالهای حساس،به حالت تعلیق در می آید.

- ✓ قسمت اعلانی: قسمت اعلانی جهت تعریف ثابتها یا متغیرهای محلی که تنها در داخل process قابل دسترسی هستند ،مورد استفاده قرار می گیرند .قسمت اعلانی process بین قسمت لیست حساس و کلمه رزرو شده BEGIN قرار دارد.

- ✓ قسمت عبارات:قسمت عبارات ،بین کلمات کلیدی BEGIN و END PROCESS قرار دارد .در این قسمت تمام عبارات به صورت ترتیبی در شبیه سازی اجرا می شوند.ویک عبارت تا زمانی که عبارت قبلی آن اجرا نشده است ،اجرا نمی شوند.

گرامر یک عبارت process به صورت زیر است:

```
PROCESS ( لیست حساس )

  [قسمت اعلانی]

BEGIN

  [ قسمت عبارات ]

END PROCESS;
```

برای مثال مالتی پلکسر ۲به۱،بدنه architecture برای توصیف رفتاری به صورت زیر است:

```
ARCHITECTURE behavioral OF mux2to1 IS
BEGIN

  PROCESS (input1, input2, sel)
  BEGIN

    IF (sel = 1) THEN
      output <= input2;
    ELSE
      output <= input1;
    END IF;

  END PROCESS;

END behavioral;
```

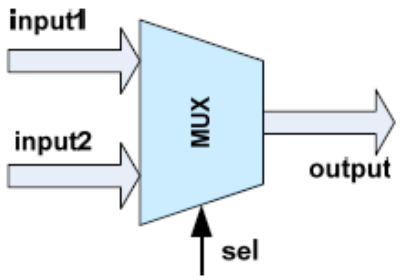
همانطور که از عبارات فوق مشخص است ،تنها مدل رفتاری مالتیپلکسر نوشته شده است.یعنی هر وقت sel=1 شد،خروجی مقدار ورودی دوم یعنی input2 و هنگامی که sel =0 باشد،خروجی مقدار ورودی اول یعنی input1 را داشته باشد.

این نوع توصیف یعنی توصبف رفتاری برای شبیه سازی و نوشتن test bench بسیار مناسب است .پیاده سازی یک طرح با استفاده از توصیف رفتاری نیاز به توجه وتجربه دارد.زیرا از آنجایی که این توصیف در سطح بالاتری از سطح گیت نوشته شده است .ابزار سنتز باید این توصیف را به سطح گیت ببرد.وچنانچه این توصیف به گونه ای نوشته شده باشد که قابل فهم برای ابزار سنتز نباشد ،نتیجه تبدیلی که ابزار سنتز به سطح گیت انجام می دهد ،کاملا متفاوت با آنچه که مورد نظر طراح می باشد ،خواهد بود! توصیف یک مدار با استفاده از مدل رفتاری نیاز به توجه و دقت زیاد دارد و دارای پیچیدگی خاص خود می باشد.زیرا چنانچه مدار کمی پیچیده باشد احتمال اشتباهاتی در سطح رفتار زیاد می شود واز آنجایی که این توصیف در سطح رفتاری است نه سطح گیت ،بررسی و پیدا کردن اشتباهات بسیار مشکل است .اما با این وجود ،این نوع مدل بسیار مناسب برای شبیه سازی و test bench است.

کلمه کلیدی GENERIS

زبان VHDL دارای قابلیتها و تواناییهای زیادی است که به طراح اجازه می دهد طراحی خود را بسیار انعطاف پذیر وبهینه طراحی نمایید.

دربسیاری مواقع طراح نیاز دارد یک بلوکی که برای مدار خاصی طراحی کرده است را برای طراحیهای بعدی خود نیز از آن استفاده کند ،بدون آنکه نیاز باشد فایل VHDL آنرا دوباره بنویسد.به عنوان مثال طراح می خواهد مالتیپلکسر ۲به۱ را برای باس داده مطابق شکل شماره ۸-۳ پیاده سازی نماید.



شکل شماره ۸-۳ مالتیپلکسر ۲به۱ برای باس داده

فرض کنیم که باس داده دارای اندازه ۸ باشد.در این صورت پیاده سازی آن به صورت زیر خواهد بود:

```
ENTITY mux2to1_bus IS
  PORT (
    input1 : IN BIT_VECTOR ( 7 downto 0);
    input2 : IN BIT_VECTOR( 7 downto 0);
    sel : IN BIT;
    output : OUT BIT_VECTOR( 7 downto 0) );
END mux2to1_bus;

ARCHITECTURE data_flow OF mux2to1_bus IS
BEGIN

  output <= input1 WHEN ( sel = '0' ) ELSE input2;

END data_flow;
```

حال چنانچه طراح بخواهد این مالتیپلکسر را برای یک باس ۱۶تایی بکار برد باید تغییرات زیادی درآن دهد.در اینگونه موارد VHDL دارای قابلیتی است که به طراح اجازه می دهد طرح خود را تنها با تغییر یک متغییر ،بانیاز جدید خود سازگار کند.این قابلیت با کلمه کلیدی GENERIS امکان پذیر است.

مثلا طراح در همان ابتدا می تواند شکل فوق را به صورت زیر توصیف نماید:

```
ENTITY mux2to1_generic IS
  GENERIC ( size : INTEGER := 8);
  PORT (
    input1 : IN BIT_VECTOR ( size-1 downto 0);
    input2 : IN BIT_VECTOR( size-1 downto 0);
    sel : IN BIT;
    output : OUT BIT_VECTOR( size-1 downto 0) );
END mux2to1_generic;

ARCHITECTURE data_flow OF mux2to1_generic IS
BEGIN

  output <= input1 WHEN ( sel = '0' ) ELSE input2;

END data_flow;
```

در توصیف بالا چنانچه مقدار size از ۸به ۱۶ تغییر کند این مالتیپلکسر می تواند برای باس ۱۶تایی یکار رود. توصیف یک مدار به گونه ای که دارای چنین قابلیت انعطاف پذیری باشد نیاز به تجربه درVHDL و آگاهی از توانایی ها و محدودیتهای آن دارد.در غیر این صورت ،چه بسا ممکن است توصیف نهایی نه تنها چنین قابلیتی نداشته باشد بلکه نتیجه حاصل از سنتز کاملا متفاوت با آنچه که طراح نیاز دارد ،باشد. باید در این جا متذکر شدکه مدارهایی که به صورت GENERIS طراحی می شوند ممکن است از لحاظ سرعت ویا حجم بهینه نباشند.

نرم‌افزار طراحی و پیاده‌سازی سخت‌افزار ISE

این نرم‌افزار یکی از محصولات شرکت Xilinx است و به منظور اجرای پروژه‌های طراحی سخت‌افزاری مرتبط با تراشه‌های این شرکت مورد استفاده قرار می‌گیرد. از جمله FPGAهایی که این شرکت سازنده آن است می‌توان به تراشه‌های سری SPARTAN و Virtex اشاره نمود.

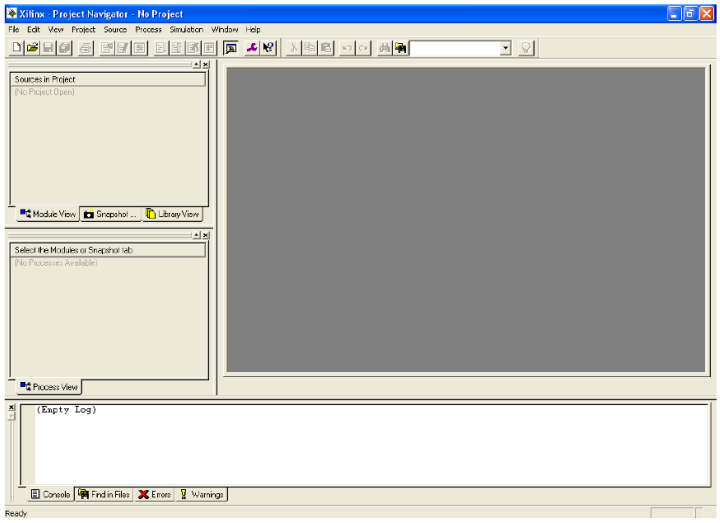
نرم افزار Xilinx ISE

آشنایی با نرم افزارISE

در این فصل چگونگی طراحی توسط نرم افزار Xilinx ISE را بطور دقیق ,با جزئیات کامل آورده شده است و همچنین چگونگی کار با انواع روشهای طراحی با تراشه های شرکت Xilinx بررسی شده است.

روشهای طراحی ذکر شده در اینجا شامل طراحی بوسیله نوشتن کد با زبانVerilog/VHDL ، طراحی شماتیکی با محیط ECS و طراحی بوسیله دیاگرام حالت با محیط StateCAD می باشد .

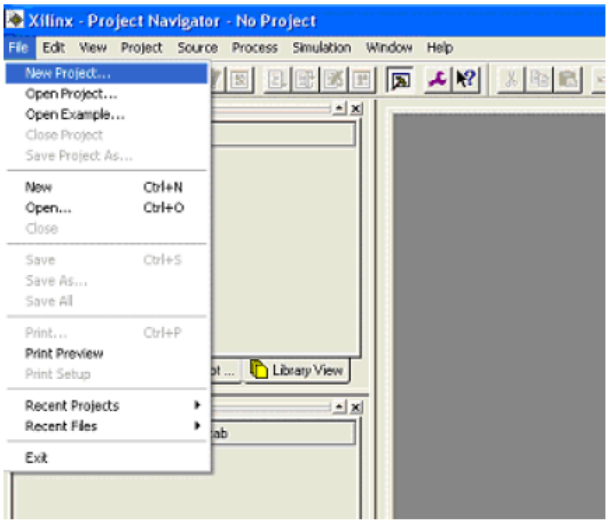
شکل شماره ۱-۳ محیط کلی این نرم افزار را نشان می دهد.



شکل شماره ۱-۳
محیط کلی نرم افزار ISE

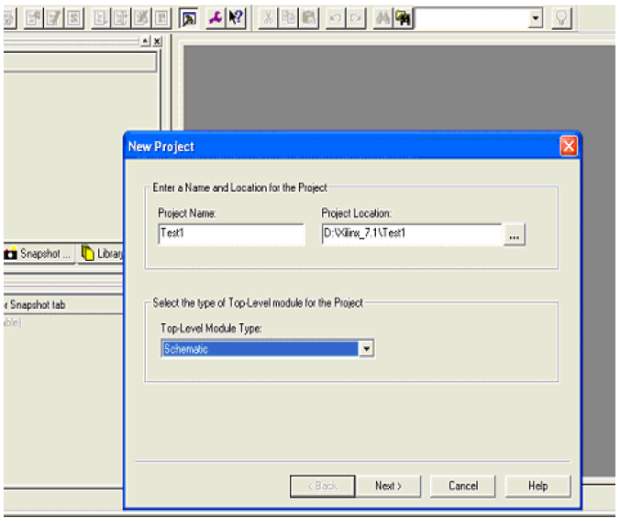
ایجاد یک پروژه جدید

گزینه File>New Project را انتخاب نموده و اسم پروژه خود را در جای مربوطه وارد کنید (شکل شماره ۲-۳).



شکل شماره ۲-۳ ایجاد پروژه ی جدید

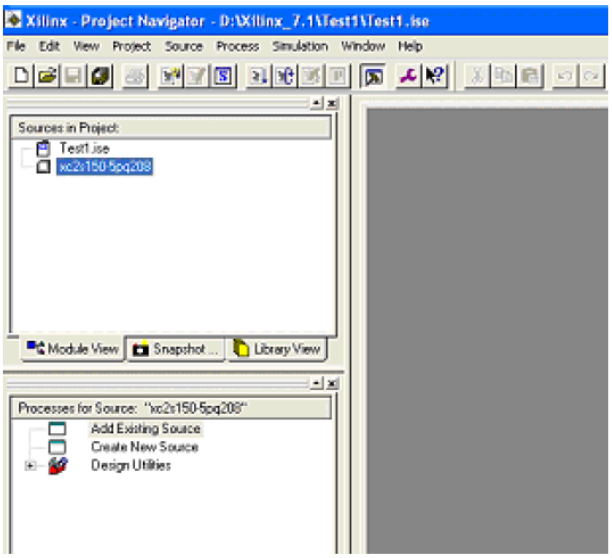
گزینه Next را کلیک کرده , نوع و شماره IC خود را به همراه دیگر مشخصات آن و نیز زبان برنامه نویسی را انتخاب نمائید .بقیه پنجره ها را بصورت پیش فرض بپذیرید و در انتهاFinish را کلیک نمائید (شکل شماره ۳-۳).



شکل شماره ۳-۳

ایجاد پروژه ی جدید

در نهایت پنجره نرم افزار شما به صورت شکل شماره۴-۳ در خواهد آمد.

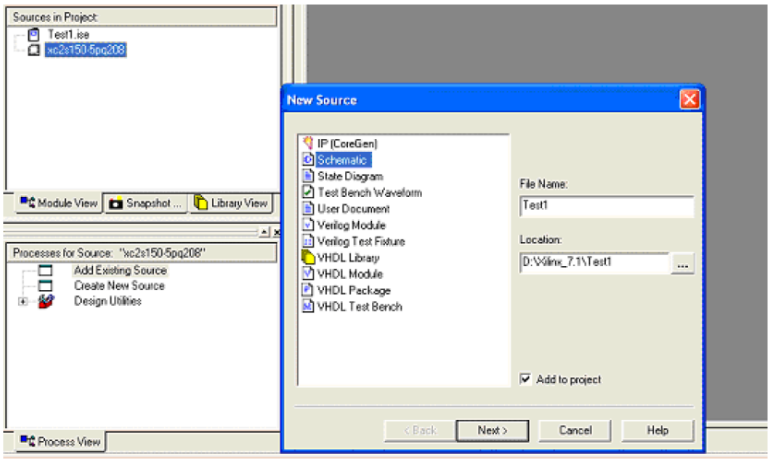


شکل شماره۴-۳

پنجره نرم افزار Xilinx ISE

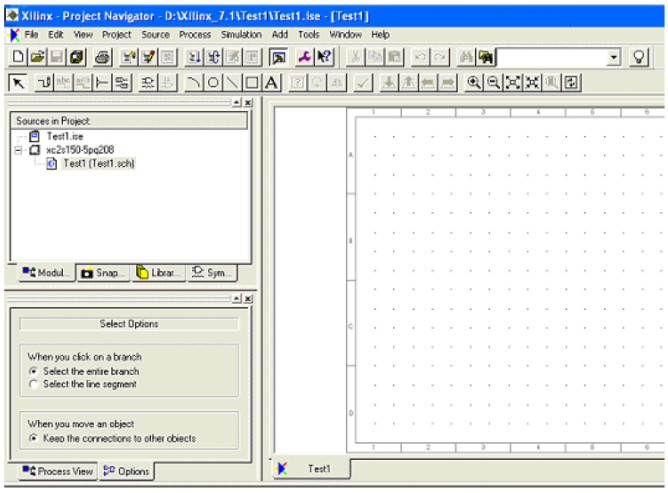
اضافه کردن فایل شماتیک جدید به پروژه

در Project Navigator گزینه Project>New Source را انتخاب کنید (شکل شماره ۵-۳).



شکل شماره ۵-۳ طریقه اضافه کردن فایل شماتیک جدید به پروژه

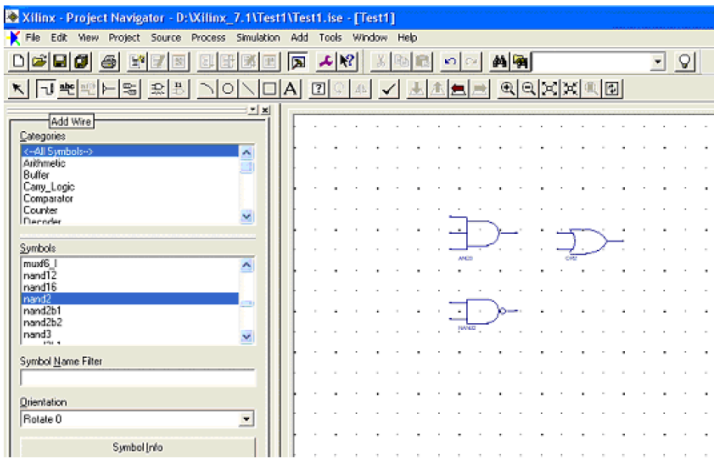
گزینه Next و سپس Finish را بزنید . محیط طراحی شماتیک با نام فایل خودتان ایجاد می گردد (شکل شماره ۶-۳) .



شکل شماره ۶-۳

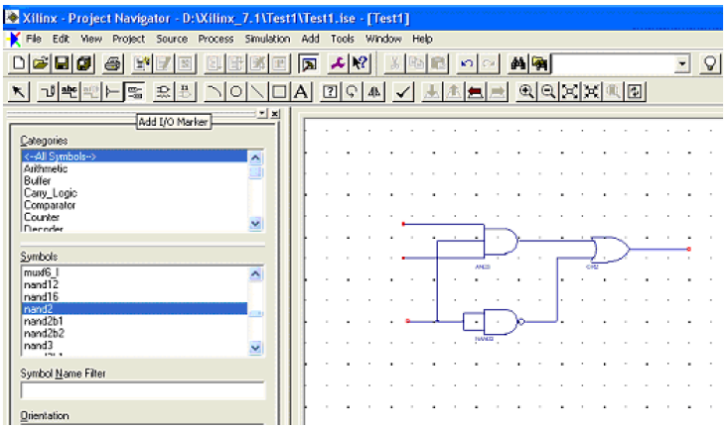
طریقه اضافه کردن فایل شماتیک جدید به پروژه

برای انتخاب المان از دکمه Add Symbol استفاده نمایید .اکنون پنجره شما به ترتیب زیر خواهد بود . سمبل and 3 که همان گیت and با سه ورودی است انتخاب نموده و سپس روی صفحه شماتیک کلیک کنید و شکل شماره۷-۳ را میسازیم.



شکل شماره۷-۳ طریقه برنامه نویسی گرافیکی

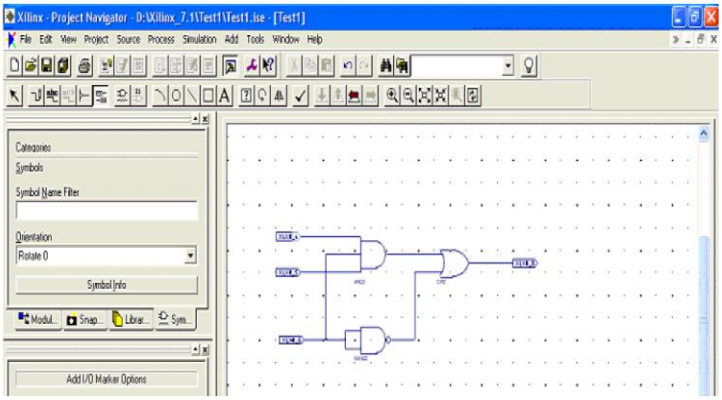
حال باید Wiring (سیم بندی)مدار را انجام دهید .پس دکمهAdd Wire را انتخاب و با کلیک کردن روی هر نقطه شماتیک مشغول Wiring شوید .حالا Wiring شماتیک به صورت شکل شماره۸-۳ در آمده است.



شکل شماره۸-۳

طریقه برنامه نویسی گرافیکی

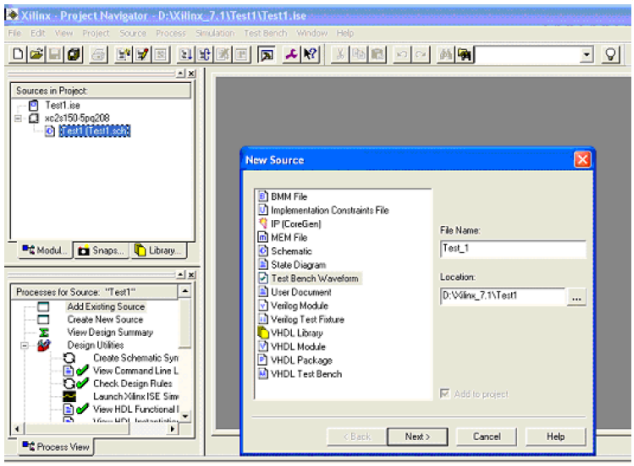
حالا باید سیگنالهای ورودی و خروجی را به صورت یک پورت تعریف نمائید . پس دکمه Add I/O Marker را انتخاب و در قسمتAdd I/O Marker Option گزینه Add an Input Marker را کلیک کنید .حالا روی سیگنالهای ورودی کلیک کنید . برای تعریف پورت خروجی نیز گزینه Add an Output Marker را کلیک کنید و روی سیگنال خروجی کلیک کنید . در نهایت پورتها به صورت شکل شماره ۹-۳ به شماتیک اضافه می شوند.



شکل شماره ۹-۳ طریقه برنامه نویسی گرافیکی

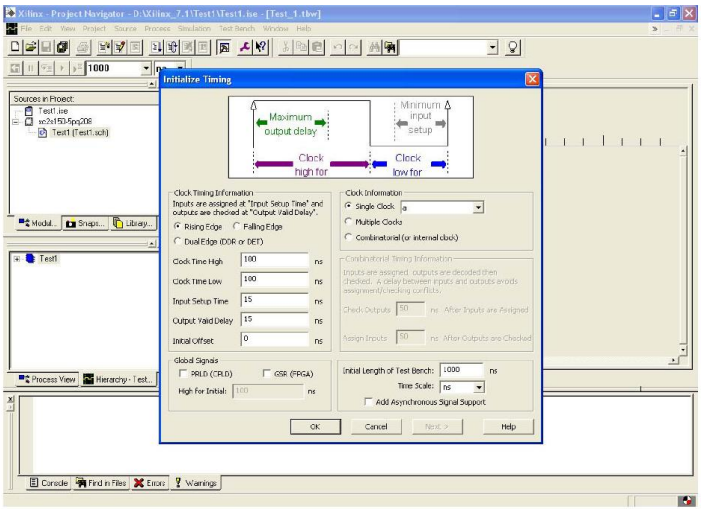
برای نامگذاری پورتهای ورودی و خروجی دکمه Add Net Name را انتخاب ، اسم پورتها را وارد و روی پورت مورد نظر کلیک کنید .حالا باید شماتیک خود را ذخیره کرده و سپس توسط دکمه چک نمائید تا دارای اشکال نباشد .پس از اینکه از شماتیک خود مطمئن شدید، از محیط شماتیک خارج شده و به محیط اصلی نرم افزار باز گردید.

برای شبیه سازی مدار خود باید فایل دیگری با نام Test Bench Waveform را اضافه کنید .این فایل جدید برای شبیه سازی فایل اصلی استفاده می گردد (شکل شماره ۱۰-۳).



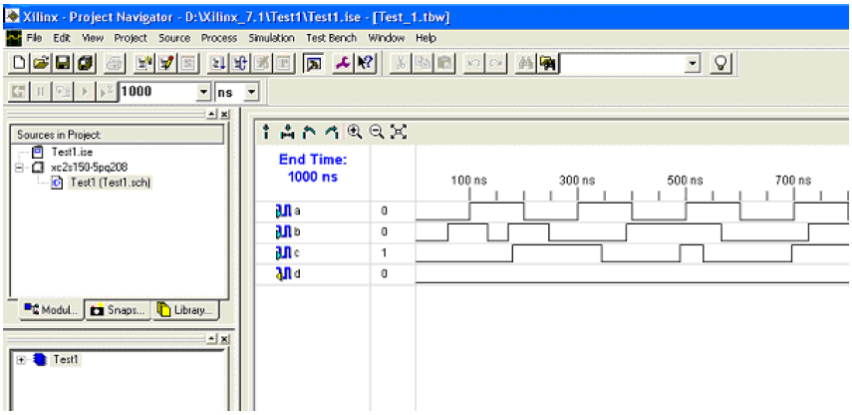
شکل شماره ۱۰-۳طریقه شبیه سازی مدار در ISE

سپس محیط تولید شکل موج برای پورتهای ورودی ظاهر می شود(شکل شماره ۱۱-۳).



شکل شماره ۱۱-۳ طریقه شبیه سازی مدار در ISE

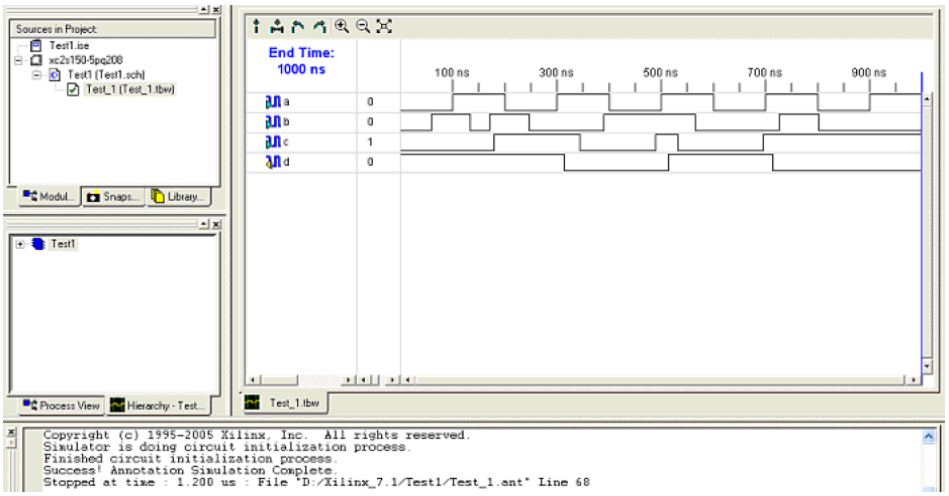
گزینه ADD Asynchronus Signal Support را انتخاب نموده و گزینه Next را کلیک نمائید .حالا یک سیگنال را به عنوان پالس ساعت تعریف کنید .گزینه Asynchronous Signal را انتخاب نموده تا بتوانید هر جایی از شکل موج را که می خواهید , تغییر دهید و به دنبال آن سیگنالهای باقیمانده را جهت مقداردهی Add کنید .پنجره زیر ظاهر می شود حالا می توانید با کلیک ماوس هر جایی از سیگنال را که بخواهید مقدار یک و یا صفر بدهید .مثلا می توانید شکل موج شماره ۱۲-۳ را در نظر بگیرید.



شکل شماره ۱۲-۳

طریقه شبیه سازی مدار در ISE

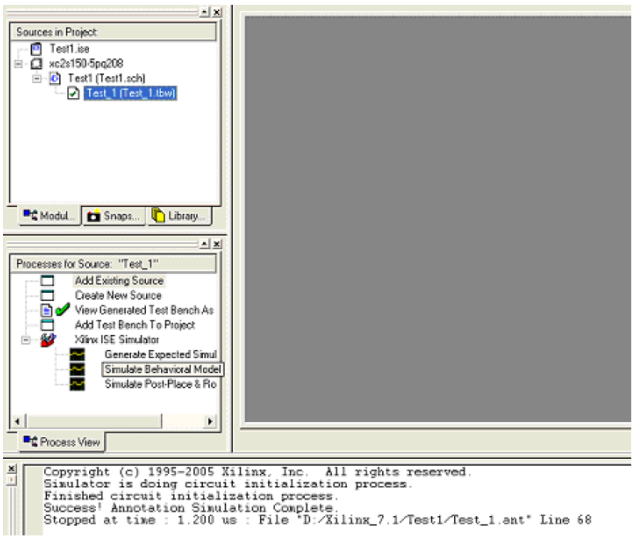
این فایل را ذخیره نموده و از این محیط خارج شده و به محیط اصلی باز گردید .در محیط اصلی روی فایلTest Bench Waveform کلیک کرده تا HighLight شود و سپس از پنجره Process view گزینه Generate Expected simulation result را دوبار کلیک کنید .پنجره محیط قبلی مجدد باز می شود با این تفاوت که سیگنال خروجی بر اساس شماتیک مقدار یافته است (شکل شماره۱۳-۳).



شکل شماره ۱۳-۳

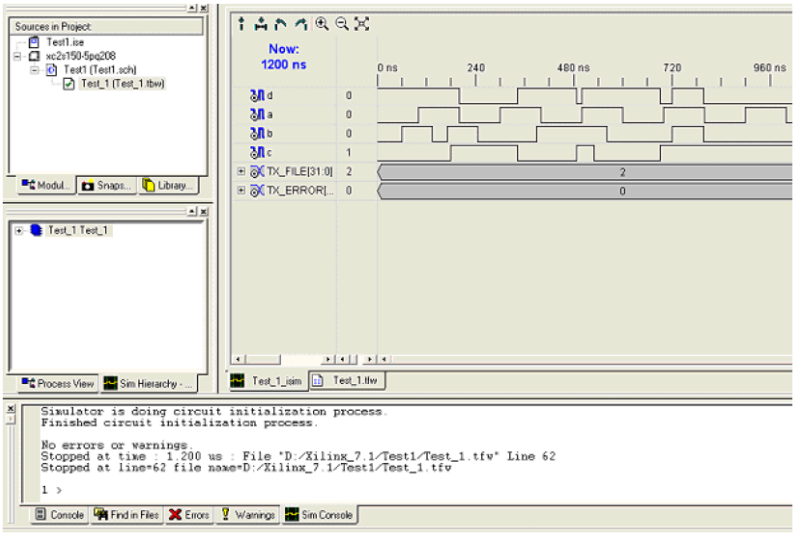
طریقه شبیه سازی مدار در ISE

فایل را ذخیره نموده و خارج شده و به محیط اصلی باز گردید . در این حالت دوباره فایل Bench Test Waveform را High Light نموده و از پنجره Process view گزینهBehavioral Model Simulate را دوبار کلیک کنید (شکل شماره ۱۴-۳).



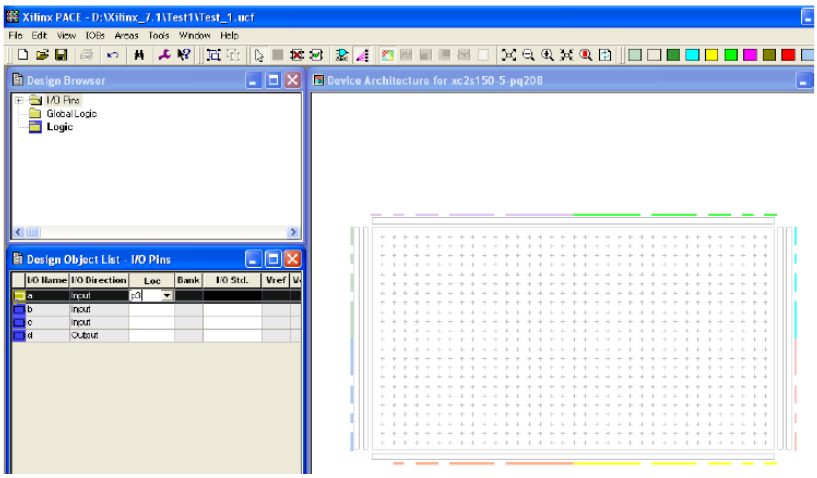
شکل شماره ۱۴-۳ کار با محیط Tese Waveform

پنجره شکل شماره ۱۵-۳ که نتایج شبیه سازی را نشان می دهد، ظاهر می شود.



شکل شماره ۱۵-۳ نتایج شبیه سازی

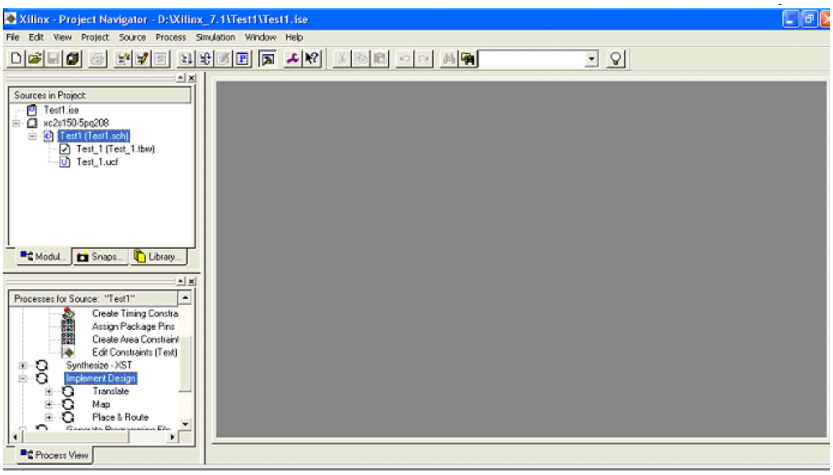
حالا می خواهیم مدار خود را برای پیاده سازی روی FPGA آماده کنیم , پس ابتدا باید برای پورتها روی FPGA پایه ای در نظر بگیریم . به این کار Pin Assignment می گویند . برای همین منظور فایل جدیدی را با نام Implementation Constraint باید به پروژه اضافه نمائیم .این فایل با پسوند *.ucf ساخته می شود .این فایل را High Light نموده و از پنجره Process view گزینه Assign Package Pins را انتخاب نمائید. پنجره شکل شماره ۱۶-۳ ظاهر می شود که در حقیقت FPGA را با پایه های آن نشان می دهد.



شکل شماره ۱۶-۳ محیط Assign Package

پورتهای مدار شما نیز در پنجره Design Object List – I/O Pin ظاهر شده اند. برای در نظر گرفتن یک پایه برای هر پورت ورودی یا خروجی در این پنجره و زیر ستون Loc عبارت P را به معنای پین و بعد عدد پایه را مشخص کنید . مثلا بنویسید: P3

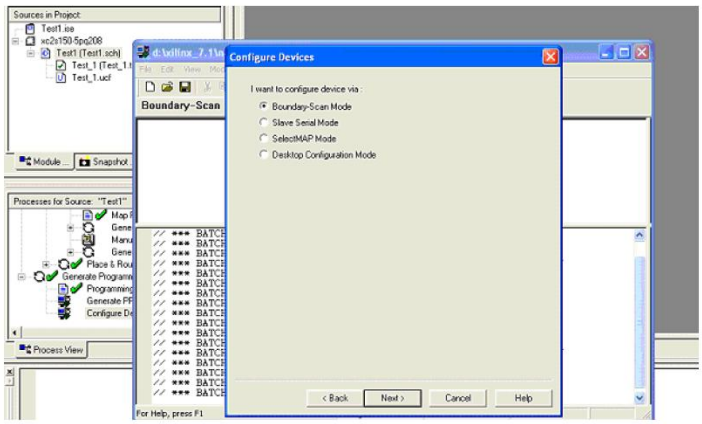
هنگامیکه برای تمامی پورتهای ورودی و خروجی پایهای در نظر گرفتید، این فایل را ذخیره نموده و از این محیط خارج شوید .در محیط اصلی نرم افزار فایل شماتیک اصلی خودتان راHigh Light نموده و از پنجرهProcess view گزینه Implement Design را دو بار کلیک نمائید (شکل شماره۱۷-۳).



شکل شماره۱۷-۳ محیط Implement Design

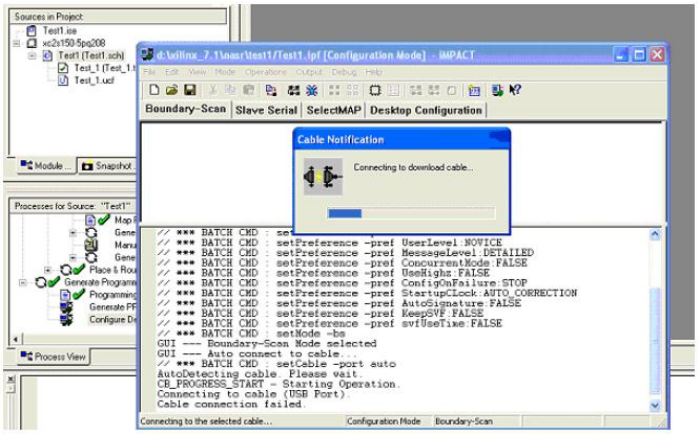
سپس نرم افزار مشغول سنتز و پیاده سازی فایل شماتیک شما رویFPGA خواهد شد .بعد از آن برای پروگرام نمودن FPGA روی برد , نرم افزار باید فایلی را با نام*.bit تولید نماید . این کار را با دو بار کلیک کردن روی گزینهGenerate Programming File در پنجرهProcess view انجام دهید

حالا نرم افزار باید از طریق پروگرامری که به پورت پرینتر کامپیوتر خود وصل کرده اید . FPGA روی برد را برنامه ریزی نماید . به همین منظور گزینهConfigure Device (iMPACT) را دوبارکلیک کنید . گزینه ها را به صورت پیش فرض پذیرفته و Next را کلیک نمائید(شکل شماره۱۸-۳).

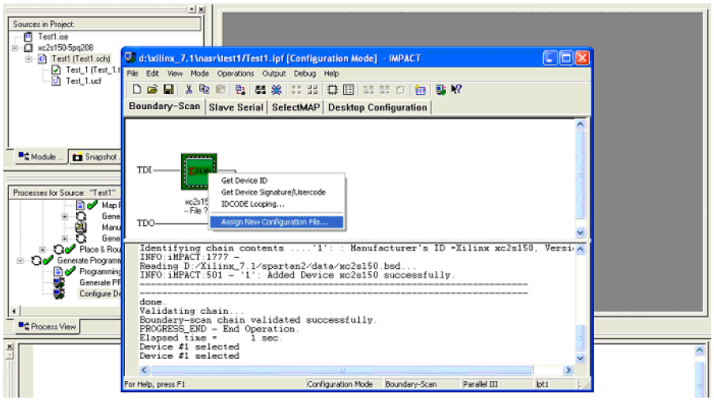


شکل شماره۱۸-۳
پروگرام نمودن FPGA در ISE

در نهایت نرم افزار به طور اتوماتیک کابل برنامه ریزی شما را تشخیص وFPGA روی برد را خواهد شناخت .سپس روی ،FPGA راست کلیک نمائید و فایلbit را که در مرحله قبل تولید شده را باز کنید (شکل شماره۱۹-۳ و ۲۰-۳).



شکل شماره۱۹-۳ شناساندن فایل برنامه به آی سی

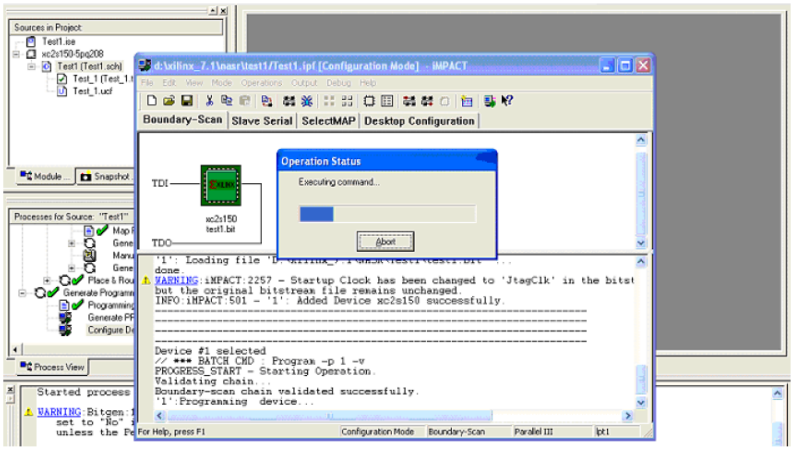


شکل شماره۲۰-۳

پروگرام نمودن FPGA در ISE

در این حالت فایل bit شما زیر FPGA شما در نظر گرفته می شود . روی FPGA راست کلیک نموده و گزینه Program را کلیک نمائید . نرم افزار شروع به برنامه ریزی FPGA از طریق کابل پروگرامر شما خواهد کرد (شکل شماره ۲۱-۳).

حالا شما طرح خود را با موفقیت روی FPGA پیاده سازی و برنامه ریزی نمودید.



شکل شماره ۲۱-۳

پیاده سازی و برنامه ریزی طرح بر روی FPGA