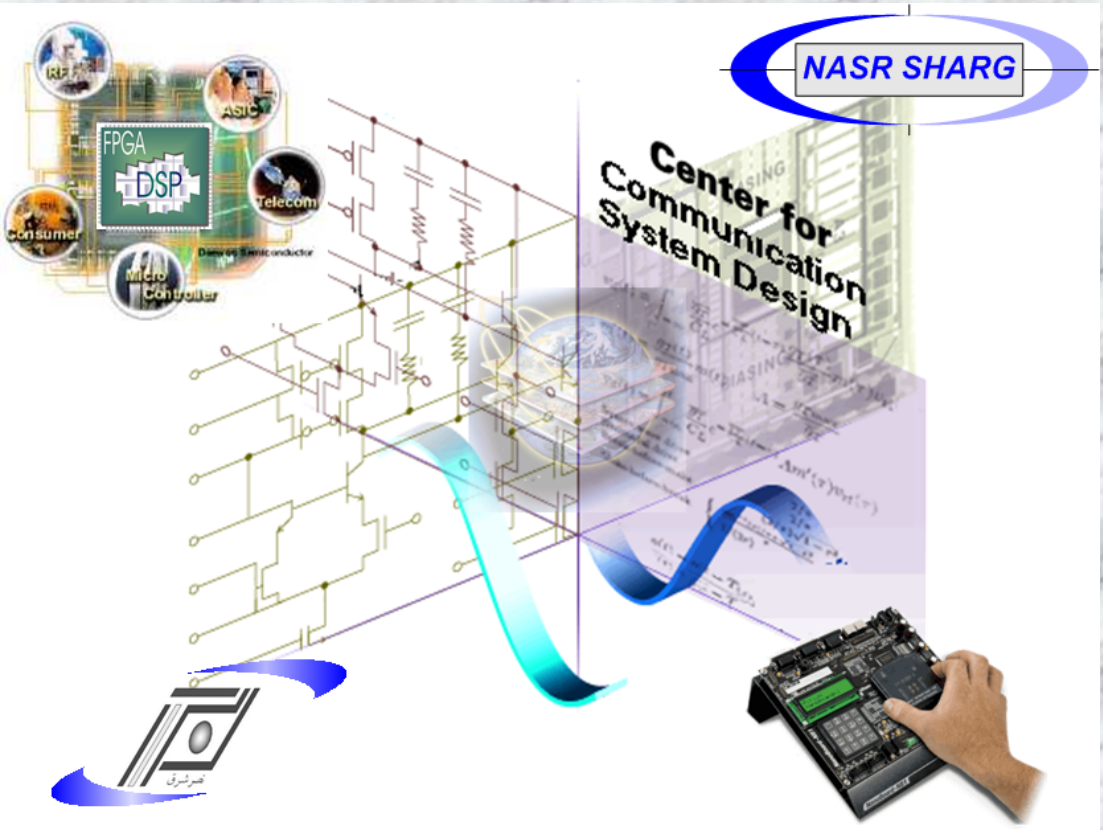
به نام خدا

دستور کار آزمایشگاه معماری کامپیوتر



شرکت نوآوران صنعت رایانه شرق

فهرست مطالب

پیشگفتار

تراشه های قابل برنامه ریزی با توجه به کارآیی و توانایی بالا در سیستم‌های صنعتی و تحقیقاتی، باعث تحول فوق العاده ای در صنعت میکرو الکترونیک شده اند. اما به دلیل فقدان آموزش و ابزار مورد نیاز در دانشگاههای کشورمان، همواره شکاف عمیقی میان محیط دانشگاهی و محیط صنعتی ملموس و محسوس بوده است. در این راستا شرکت نوآوران صنعت رایانه‌ای شرق با مطالعات فراوان بر روی تراشه های قابل برنامه ریزی FPGA ، محصول شرکت ALTERA را به عنوان مهمترین بستر آموزشی برای دانشگاههای کشور عزیزمان در نظر گرفته است.

آزمایشگاه معماری کامپیوتر با هدف ایجاد زیرساخت لازم جهت اتوماسیون طراحی های الکترونیکی، آسان نمودن طراحی های پیشرفته دیجیتال و ساخت نمونه های آزمایشگاهی ارائه شده است. در این میان طراح نیز باید نسبت به مفاهیم پایه آگاهی و دانش کامل داشته باشد. به همین منظور در این آزمایشگاه سعی شده است تا این مفاهیم با دیدی آموزشی به بهترین نحو ممکن انتقال داده شود. این مفاهیم عبارتند از :

- ✓ آشنایی با ساختار تراشه های قابل برنامه ریزی FPGA
- ✓ روند طراحی با یک تراشه قابل برنامه ریزی FPGA
- ✓ آشنایی و برنامه نویسی با زبان توصیف سخت افزار Verilog یا VHDL
- ✓ آشنایی و طراحی با نرم افزار Quartus II شرکت ALTERA
- ✓ طراحی، پیاده سازی و تست انواع مدارهای دیجیتال بر روی FPGA
- ✓ طراحی و پیاده سازی مدار ارتباط با نمایشگر های مختلف از جمله مجموعه LED ها ، نمایشگر های هفت قسمتی، نمایشگرهای LCD
- ✓ طراحی و پیاده سازی انواع پروتکل های سری و ارتباط سری با کامپیوتر از طریق FPGA
- ✓ طراحی و پیاده سازی مدار ارتباط با حافظه SRAM از طریق FPGA
- ✓ طراحی و پیاده سازی مدار ارتباط با ADC و DAC ها از طریق FPGA

بخش اول
آشنایی با نرم افزار آزمایشگاه

آشنایی با نرم افزار شرکت ALTERA

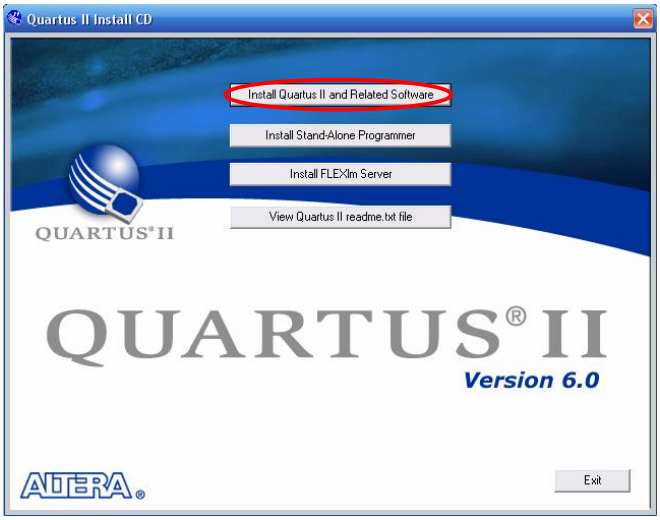
با آشنائی مختصری که در زمینه FPGA حاصل شد ، این جلسه با نرم افزار شرکت ALTERA آشنا می شوید. شرکت ALTERA دارای دو نرم افزار معروف است :

1. MAX + plus II
2. Quartus

در این آزمایشگاه دانشجویان می‌بایست نحوه کار با نرم‌افزار Quartus را فراگیرند.

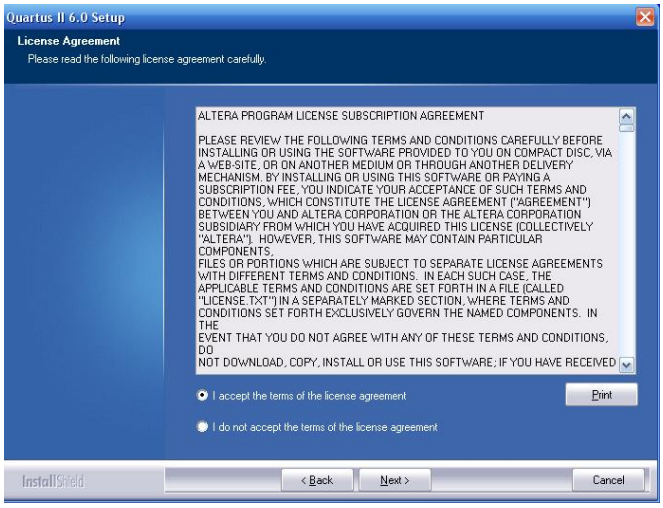
روش نصب نرم افزار QUARTUS

برای نصب نرم افزار Quartus ابتدا CD1 را در درایو قرار داده و از محتویات CD بر روی گزینه install دوبار کلیک می کنیم. سپس از پنجره باز شده گزینه Install Quartus II and Related Software را انتخاب می کنیم (شکل 1-2-1).



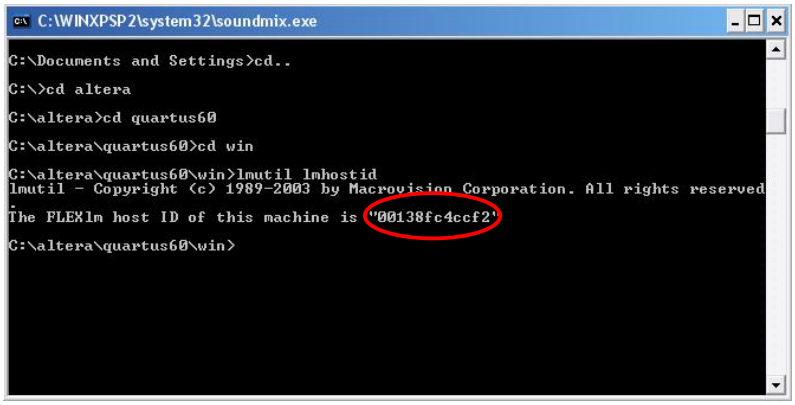
شکل 1-2-1

در دو پنجره بعدی گزینه Next را انتخاب می کنیم. در شکل زیر گزینه I accept the terme of the license agreement را انتخاب و سپس بر روی گزینه Next کلیک می کنیم.



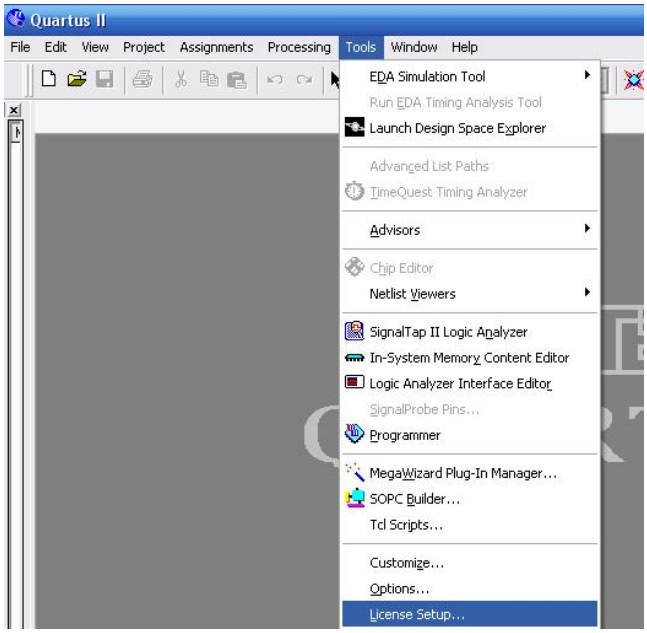
شکل 2-2-1

در پنجره های بعدی نام کاربر ، نامی که می خواهیم نرم افزار با آن نصب شود و مسیر نصب آن را مشخص می کنیم. پس از پایان نصب برنامه یک command window باز می کنیم و عبارت lmutil lmhostid را تایپ می کنیم و کلید enter را می زنیم.



شکل 3-2-1

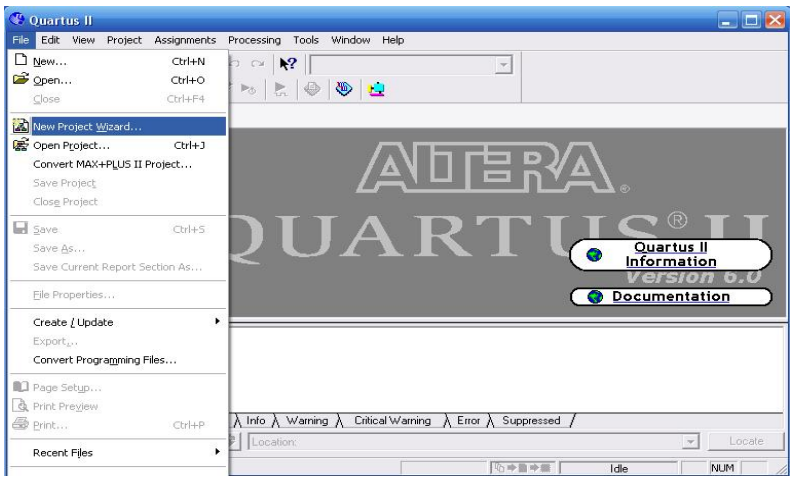
با این کار یک عدد چند رقمی داده می شود. آن عدد را کپی کرده و فایل license.dat را باز می کنیم و به جای همه عبارت های CHANGEM در متن، عدد داده شده را جایگزین می کنیم. سپس فایل های sys_cpt.dll و license.dat را در مسیر نصب نرم افزار در win folder کپی می کنیم. همچنین هنگامی که برای اولین بار برنامه Quartus را اجرا کردیم از مسیر Tools / License Set up محل فایل License را مشخص می کنیم.



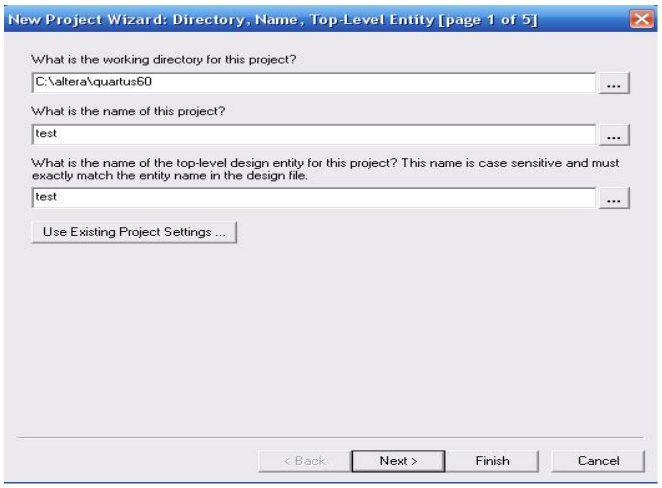
شکل 4-2-1

روش کار با نرم افزار Quartus

برای شروع یک طراحی دیجیتال می‌بایست محیط نرم‌افزار مورد نظر را باز کرد . بدین منظور نرم افزار Quartus را اجرا نمائید. در کلیه نرم افزارهای جدید برای شروع یک طراحی باید پروژه‌ای را باز کرد. در این نرم افزار نیز با استفاده از منوی File و انتخاب گزینه New Project Wizard پنجره ای برای ایجاد پروژه جدید باز می شود (شکل 6-2-1) . در پنجره شکل 6-2-1 نام پروژه و مسیر ذخیره شدن آن مشخص می شود.

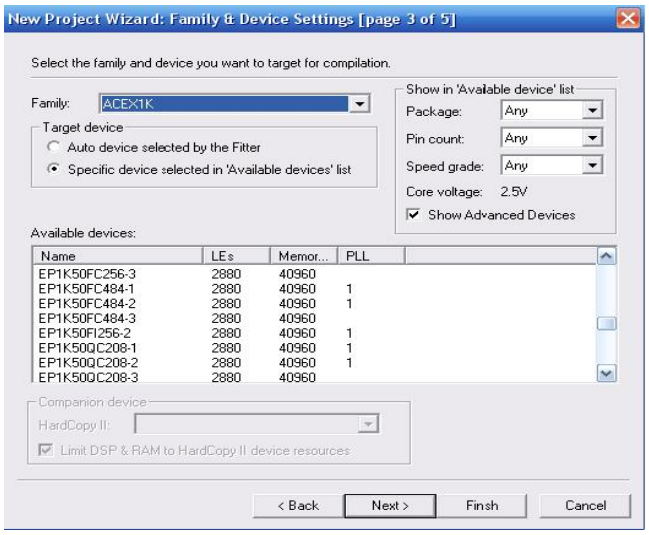


شکل 5-2-1



شکل 6-2-1

در پنجره شکل 7-2-1 می بایست خانواده CPLD یا FPGA و شماره آن مشخص شود .



شکل 7-2-1

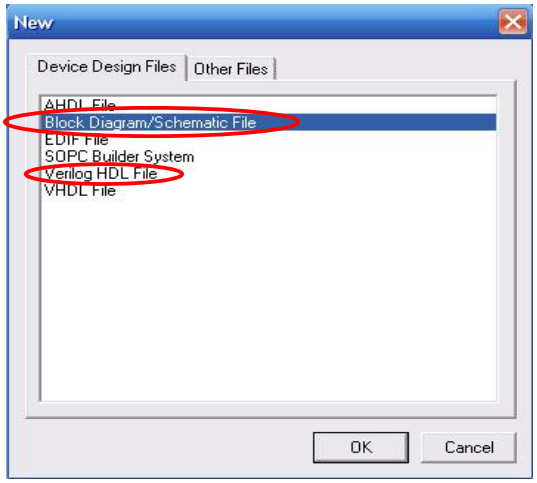
اکنون پروژه ای برای یک طراحی جدید در نظر گرفته شده است. برای شروع باید فایل طراحی در نرم افزار وارد گردد. دو نوع فایل طراحی وجود دارد:

۱- گرافیکی (GRAPHIC)

2- متنی (TEXT)

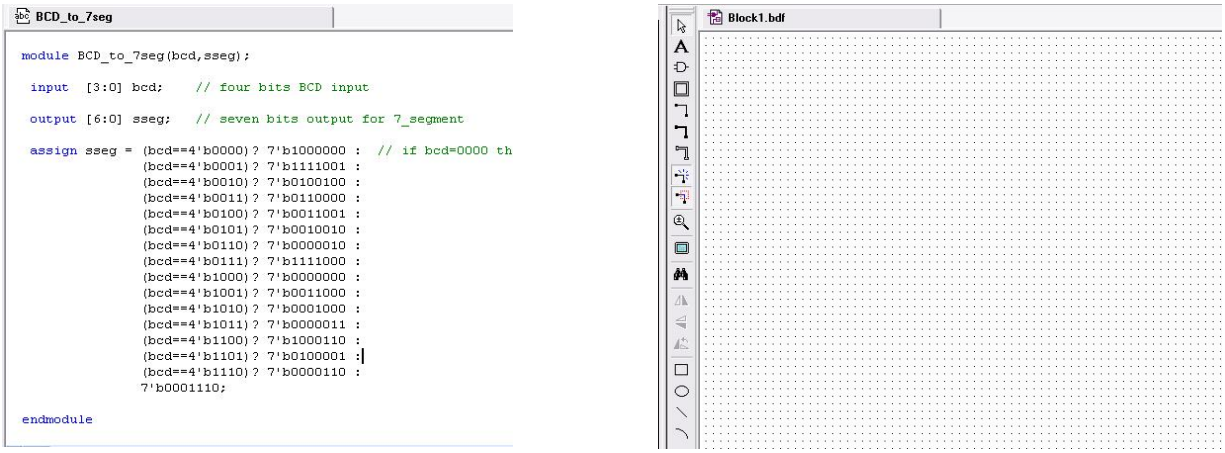
در فایلهای متنی برای طراحی از زبانهای HDL نظیر Verilog و VHDL استفاده می‌گردد. پسوند فایلهای متنی با زبانهای مذکور به ترتیبv.* وvhd.* است. شرکت ALTERA زبانی مخصوص به خود تحت عنوان AHDL نیز دارد که پسوند فایلهای آنahd.* می

باشد. برای شروع طراحی از منوی File گزینه New را انتخاب کنید . پنجره شکل 8-2-1 باز می شود . در این پنجره برای طراحی گرافیکی باید گزینهBlock Diagram/Schematic File و برای طراحی با کد Verilog گزینه Verilog HDL File انتخاب گردد.



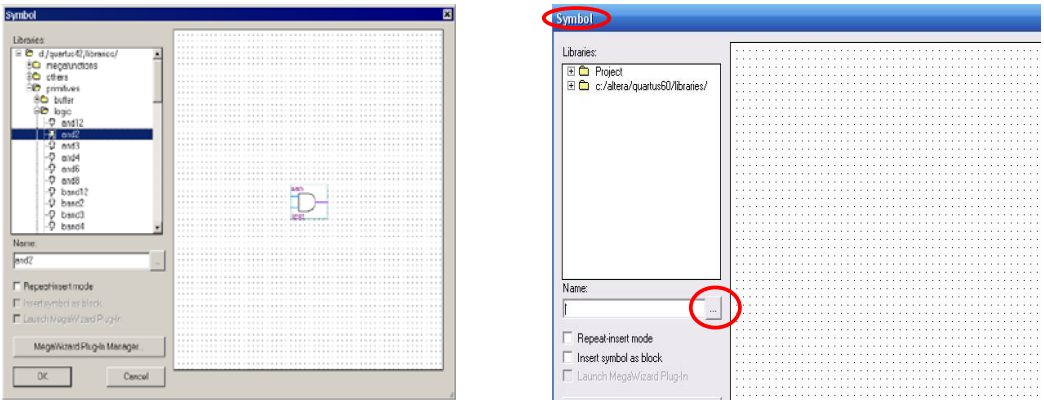
شکل 8-2-1

با انتخاب گزینه Block Diagram/Schematic File پنجره طراحی گرافیکی و با انتخاب گزینه Verilog HDL File پنجره طراحی با کد Verilog (شکل 9-2-1) باز می شود. از گزینه File >Save As برای ذخیره فایل و تعیین نوع فایل (*.bdf برای طراحی گرافیکی و *.v برای طراحی با کد Verilog) استفاده کنید. (Add File to Current Project را نیز کلیک نمائید.)



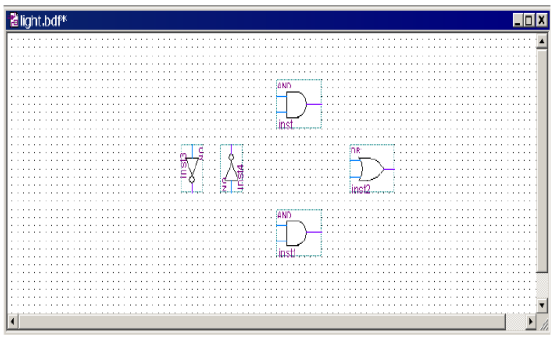
شکل 9-2-1

در طراحی گرافیکی برای استفاده از کتابخانه و اضافه کردن المانهای منطقی،با دو بار کلیک بر روی فضای خالی صفحه ، پنجره ای با نام Symbol باز می شود که قطعات مورد نظر را می توان از مسیر مشخص شده آورد. همچنین بوسیله آیکون  ،می توان المان مورد نظر را انتخاب نمود .



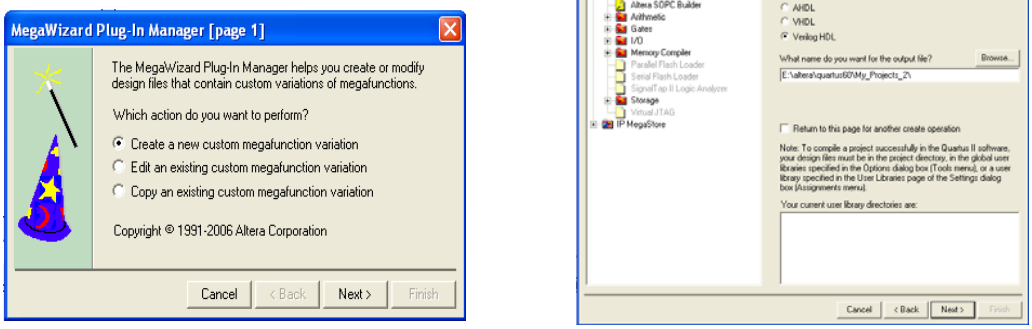
شکل 1-2-10

برای کپی یک المان که در محیط گرافیکی قرار دارد، روی آن راست-کلیک نموده و تا نقطه دیگری روی صفحه drag نمائید. با استفاده از آیکون  می‌توانید یک المان را 90 درجه بچرخانید. دیگر المانها را نیز با همین روش روی صفحه گرافیکی قرار دهید.



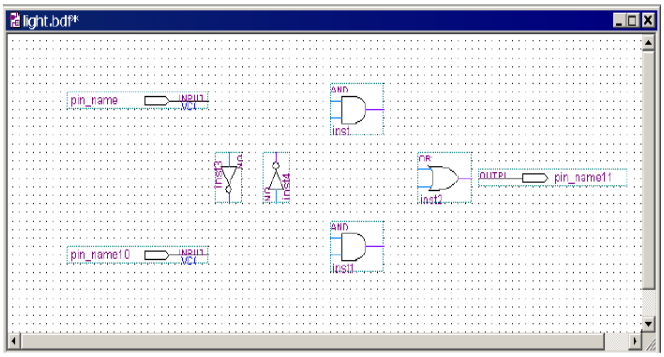
شکل 1-2-11

برای اضافه کردن بلوک‌های اساسی مانند ALU، شیفت‌دهنده، مالتی‌پلکسر، حافظه و ... می‌توان از از منوی Tools گزینه MegaWizard Plugin Manager استفاده نمود.



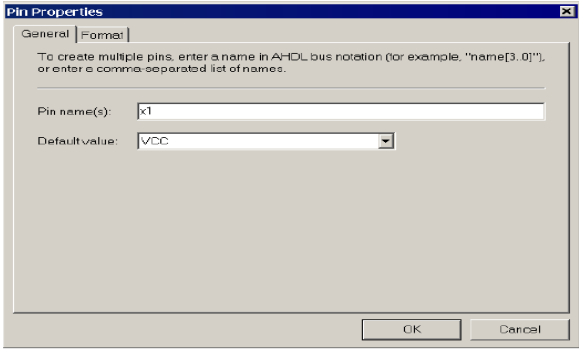
شکل 1-2-12

با استفاده از Wizard می‌توان بلوکهای محاسباتی مانند جمع‌کننده، ضرب‌کننده و ... یا حافظه‌ها و دیگر بلوک های مورد نیاز را که به صورت پارامتری موجود می‌باشد، تنظیم و طراحی نمود و آن را به طرح شماتیک خود اضافه کرد.برای اضافه کردن ورودی/خروجی از کتابخانه Primitive گزینه Pin را انتخاب کنید.



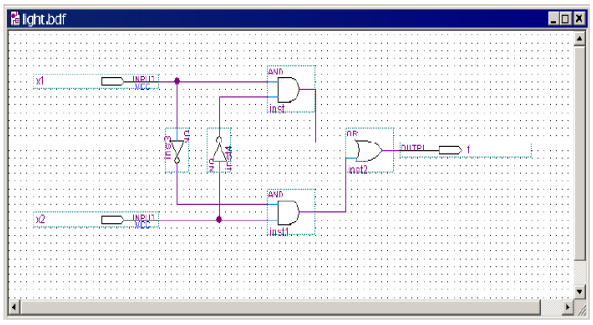
شکل 1-2-13

برای نامگذاری پایه‌های ورودی و خروجی، روی کلمه Pin _Name کلیک کنید. تا پنجره Pin Properties باز شود.



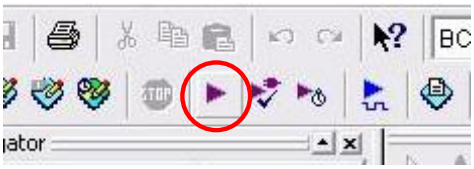
شکل 1-2-14

برای سیم بندی المانهای موجود در محیط گرافیکی روی آیکون  کلیک کرده تا ابزار Orthogonal Node فعال شود. روی نقطه شروع کلیک کرده و با drag کردن تا نقطه انتهایی، سیم بندی را انجام دهید.



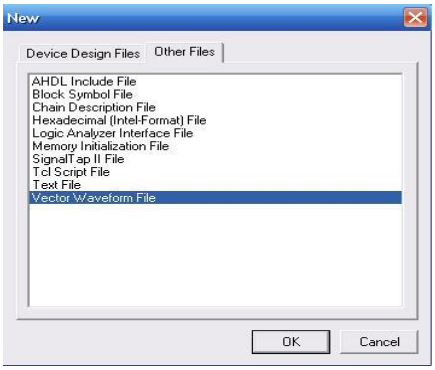
شکل 1-2-15

پس از طراحی کامل مدار،نوبت به Compile آن می رسد . اگر کلیه مراحل Compile مدار بدون هیچ خطایی به پایان برسد آنگاه مدار از لحاظ منطقی بی نقص می‌باشد. برای Compile کردن برنامه گزینه مشخص شده در شکل 1-2-16 انتخاب می شود .



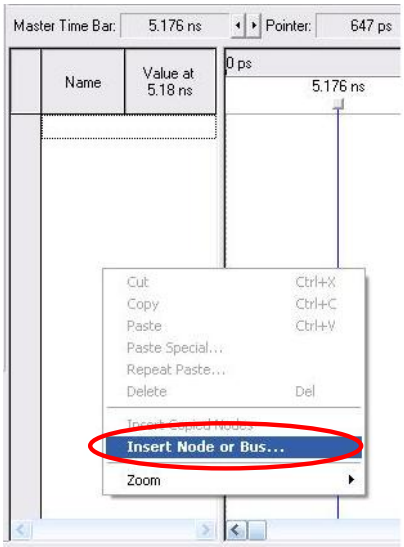
شکل 1-2-16

یکی از امکانات رایج نرم افزارها تحلیل نرم افزاری می باشد. تحلیل نرم افزاری کمک بسیار زیادی به تعیین میزان صحت عملکرد مدار قبل از پیاده سازی آن می‌نماید. برای تحلیل نرم‌افزاری باید فایل vwf. * (Vector Waveform File) از طریق گزینه New در منوی File باز گردد. این کار در یک پروژه بعد از Compile صحیح صورت می گیرد.

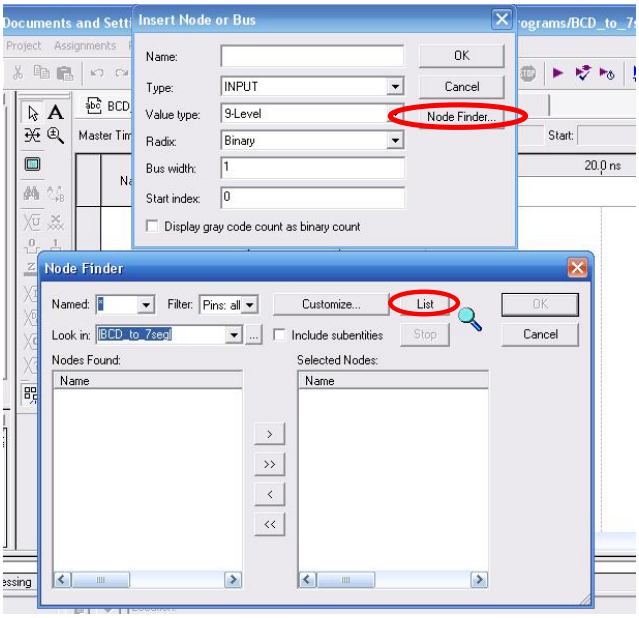


شکل 1-2-17

شکل 1-2-18 پنجره مربوط به تحلیل نرم افزاری را نمایش می‌دهد. ابتدا این صفحه باید با نام پروژه اصلی و با پسوند vwf.* ذخیره شود . سپس با کلیک راست در صفحه و انتخاب گزینه Insert Node or Bus پنجره شکل 1-2-19 باز می شود.

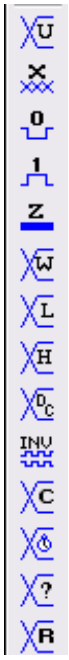


شکل 1-2-18



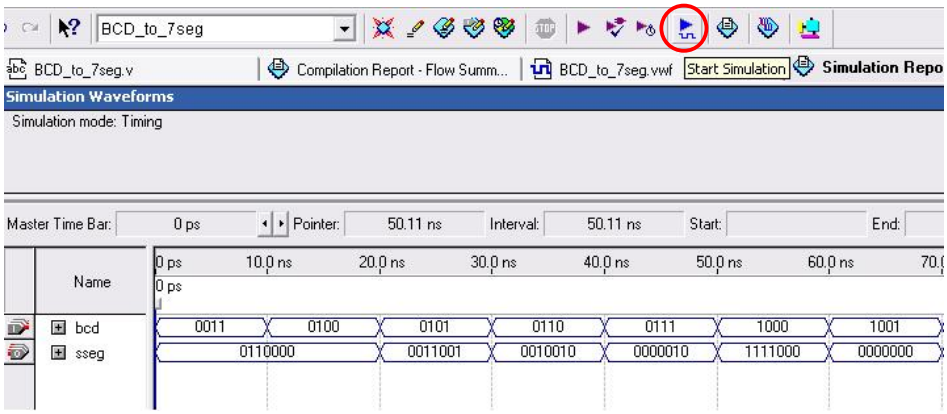
شکل 19-2-1

در این پنجره با کلیک بر روی گزینه Node Finder پنجره مذکور باز میشود. در پنجره Node Finder با کلیک بـر روی گزینـه List ، لیست ورودی ها و خروجی های مدار مربوطه ارائه می شود. برای مقدار دادن به ورودی ها با قرار دادن اشارهگر Mouse بـر روی هر ورودی و کلیک کردن بر روی آن ، کلیدهای کنار صفحه فعال خواهد شد. این کلید ها عبارتند از :



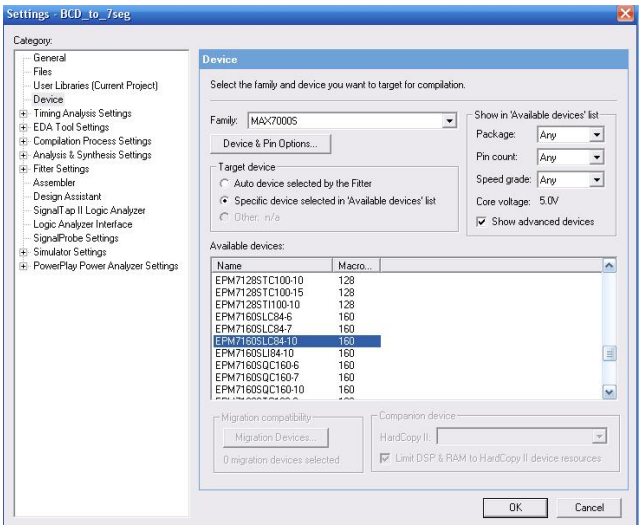
- برای عدم اختصاص مقداری در ورودی
- برای انتقال X روی ورودی
- برای انتقال صفر روی ورودی
- برای انتقال یک روی ورودی
- برای انتقال Z روی ورودی
- برای ایجاد پالسهای پریودیک
- برای ارسال مقادیر به صورت پریودیک بر روی BUS
- برای تخصیص خاص به یک قسمت از BUS
- برای انتقال مقادیر بی اهمیت به ورودی
- برای معکوس نمودن مقدار ورودی
- برای ایجاد پالسهای شمارشی با مقدار ابتدا و انتها و افزایش مشخص در هر پالس
- برای ایجاد سیگنال پریودیک clock
- برای انتقال یک مقدار مشخص به ورودی
- برای انتقال مقادیر رندوم به ورودی

لازم به ذکر است که در مدارهای منطقی علاوه بر منطق صفر و یک ، دو منطق دیگر نیز وجود دارد. منطق نا مشخص با (X) و منطق امپدانس بالا با (Z) نشان داده می‌شود. بعد از اختصاص مقادیر به ورودی ها برای مشاهده نتایج تحلیل نرم افـزاری ، گزینـه نشان داده شده در شکل زیر انتخاب می شود .



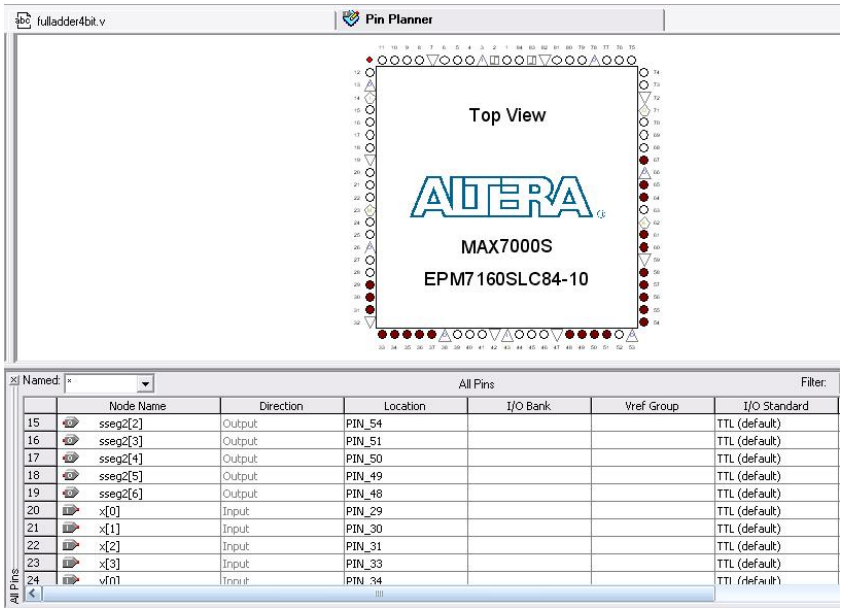
شکل 20-2-1

بعد از اتمام طراحی مدار و تحلیل نرم افزاری آن مدار آماده برنامه ریزی بر روی IC است . ابتدا باید قطعه‌ای که برای طراحـی در نظر گرفته شده (یا به عبارتی همان قطعه‌ای که بر روی برد آزمایشگاه قرار دارد) به عنوان قطعه CPLD یـا FPGA مـورد نظـر، مشخص شود. این کار با استفاده از کلید Device در منـوی Assign در بـالای صـفحه صـورت خواهـد گرفـت (شـکل 21-2-1). همچنانکه در فصل اول ذکر شد ALTERA قطعات مختلف و متنوعی دارد. برای انتخاب نوع قطعه باید شماره آن را به طور دقیـق مشخص نمود.



شکل 21-2-1

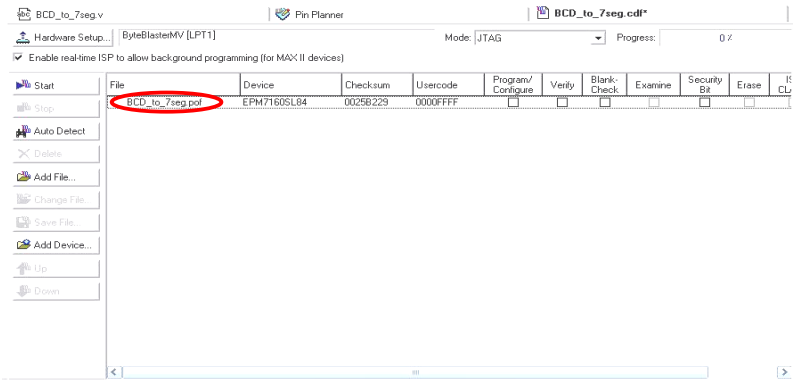
با انتخاب نوع قطعه دوباره پروژه می بایست Compile شود. این بار ارزیابی مدار بنا به قطعه مورد نظر انجام خواهد گرفت. اکنون باید ورودی‌ها و خروجی‌های مدار را به پایه‌ها نسبت داده شود . پنجره شکل شماره 22-2-1 که از مسیر Assignment /Pins باز شده است به طور واضح شکل IC، ورودی و خروجی ها و نوع قطعه را مشخص می‌نماید. با توجه به شماتیک برد آزمایشگاه هریک از پایه‌های ورودی و خروجی قطعه به یکی از قطعات موجود بر روی برد نظیر LED ها، سوئیچ ، IC مولد پالس ساعت و غیره متصل می‌باشد. بنابراین با داشتن شماتیک برد می‌توان پایه‌های قطعه را به ورودی و خروجی‌های مدار نسبت داد . همان طور که در شکل نیز مشاهده می شود در قسمت Node name ورودی و خروجی‌هایی که می‌بایست به پایه‌های قطعه نسبت داده شوند، مشخص می شوند .



	Node Name	Direction	Location	I/O Bank	Vref Group	I/O Standard
15	sseg2[2]	Output	PIN_54			TTL (default)
16	sseg2[3]	Output	PIN_51			TTL (default)
17	sseg2[4]	Output	PIN_50			TTL (default)
18	sseg2[5]	Output	PIN_49			TTL (default)
19	sseg2[6]	Output	PIN_48			TTL (default)
20	x[0]	Input	PIN_29			TTL (default)
21	x[1]	Input	PIN_30			TTL (default)
22	x[2]	Input	PIN_31			TTL (default)
23	x[3]	Input	PIN_33			TTL (default)
24	y[0]	Input	PIN_34			TTL (default)

شکل 22-2-1

برای نسبت دادن ورودی و خروجی‌های مدار به پایه‌های قطعه روی هر پایه دوبار کلیک کرده و ورودی یا خروجی مورد نظر را انتخاب می شود. بعد از مشخص نمودن همه ورودی‌ها و خروجی ها دوباره مدار Compile می گردد. برای برنامه ریزی مدار بر روی IC نیاز به یک سخت افزار برنامه‌ریزی تحت عنوان JTAG PROGRAMMER می باشد . این سخت افزار با استفاده از پورت موازی (LPT1) به برد مورد نظر متصل می‌شود. انجام عمل برنامه‌ریزی با استفاده از گزینه Programer در منوی Tools صورت خواهد گرفت. شکل شماره 23-2-1 پنجره برنامه ریزی را نمایش می‌دهد. با باز شدن پنجره برنامه ریزی فایل pof.* نشان داده شده ، مربوط به پروژه مورد نظر می‌باشد.


BCD_to_7seg.v
Pin Planner
BCD_to_7seg.cdf*
Hardware Setup...
ByteBlasterMV [LPT1]
Mode: JTAG
Progress:
Enable real-time ISP to allow background programming (for MAX II devices)
Start
Stop
Auto Detect
Delete
Add File...
Change File...
Save File...
Add Device...
Up
Down
File
Device
Checksum
Usercode
Program/ Configure
Verify
Blank-Check
Examine
Security Bit
Erase
BCD_to_7seg.pof
EPM7160SL84
0025B229
0000FFFF

شکل 23-2-1

بخش دوم
دوره مدار منطقی

آزمایش اول

راه‌اندازی و کار با نمایشگر هفت قسمتی

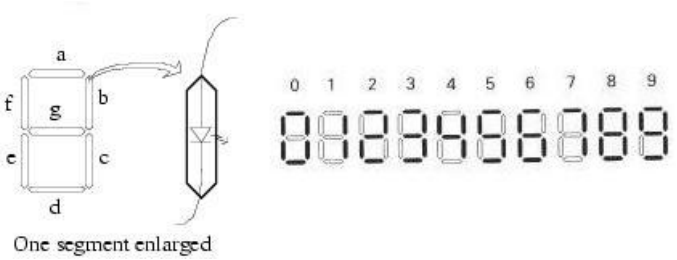
هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با نرم‌افزار Quartus
- ✓ کسب مهارت در انجام مراحل طراحی، تحلیل، تست و پیاده‌سازی

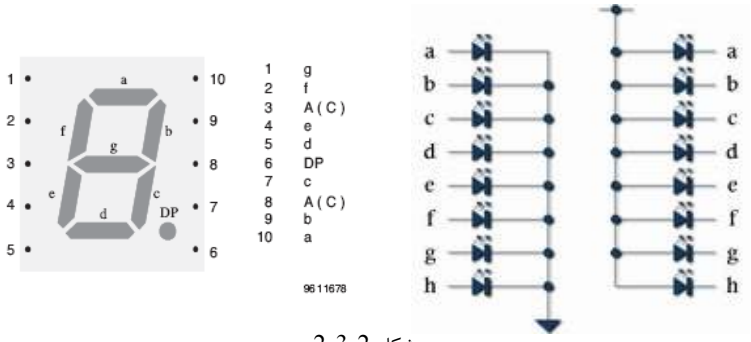
تئوری آزمایش

نمایشگر هفت قسمتی یکی از قطعات بسیار مهم در آزمایشگاه است که مشاهده نتایج آزمایش‌های صورت گرفته را ممکن می‌سازد. نمایشگر هفت قسمتی در واقع وسیله‌ای است که از هفت قطعه فتوالکتریک مستقل، مانند LED یا کریستال مایع، تشکیل شده است. قطعات مزبور به فرمی که در شکل زیر نمایش داده شده است در کنار هم قرار گرفته و امکان نمایش اعداد را مهیا می‌نمایند. همان طور که در شکل نیز مشاهده می‌نمایید با فعال نمودن گروه خاصی از این LED ها اعداد مورد نظر نمایش داده می‌شوند.



Binary State	INPUTS RBI	A_3	A_2	A_1	A_0	OUTPUTS P_1	P_2	P_3	P_4	P_5	P_6	P_7	RBO	Display
0	0	0	0	0	0	0	0	0	0	0	0	0	0	Blank
0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
1	x	0	0	0	1	1	0	0	1	1	1	1	1	1
2	x	0	0	1	0	0	0	1	0	0	1	0	1	2
3	x	0	0	1	1	0	0	0	0	1	1	0	1	3
4	x	0	1	0	0	1	0	0	1	1	0	0	1	4
5	x	0	1	0	1	0	1	0	0	1	0	0	1	5
6	x	0	1	1	0	0	1	0	0	0	0	0	1	6
7	x	0	1	1	1	0	0	0	1	1	1	1	1	7
8	x	1	0	0	0	0	0	0	0	0	0	0	1	8
9	x	1	0	0	1	0	0	0	1	1	0	0	1	9
10	x	1	0	1	0	0	0	0	1	0	0	0	1	A
11	x	1	0	1	1	1	1	0	0	0	0	0	1	b
12	x	1	1	0	0	0	1	1	0	0	0	1	1	C
13	x	1	1	0	1	1	0	0	0	0	1	0	1	d
14	x	1	1	1	0	0	1	1	0	0	0	0	1	E
15	x	1	1	1	1	0	1	1	1	0	0	0	1	F

شکل 1-3-2



شکل 2-3-2

به طور کلی نمایشگرهای هفت قسمتی به دو نوع کاتد مشترک و آند مشترک تقسیم می‌شوند. در نمایشگر هفت قسمتی کاتد مشترک، کاتد تمامی LED ها به هم متصل شده و به منظور روشن نمودن LED ها مورد نظر می‌بایست ولتاژ آند آنها را نسبت به کاتد مشترک افزایش داد. در نمایشگر هفت قسمتی آند مشترک، آند تمامی LED ها به هم متصل شده و به منظور روشن نمودن LEDها مورد نظر می‌بایست ولتاژ کاتد آنها را نسبت به آند مشترک کاهش داد.

پس از آشنایی با ساختار نمایشگر هفت قسمتی و نحوه عملکرد آن به چگونگی راه‌اندازی و کار با آن می‌پردازیم. به منظور استفاده از نمایشگر هفت قسمتی می‌بایست اعداد BCD را به کدهایی تبدیل کرد که LED های متناسب با عدد مورد نظر را روشن نمایند. با استفاده از جدولی که در شکل 1-3-2 آورده شده است بسادگی می‌توان مدار ترکیبی راه‌اندازی نمایشگر هفت قسمتی را طراحی نمود .

تکالیف پیش از آزمایش

پیش از ورود به آزمایشگاه و آغاز آزمایش مطالب این بخش را مطالعه، موارد خواسته شده را انجام و به سؤالات مطرح شده پاسخ دهید و نتایج بدست آمده را در گزارش خود ثبت نمایید.

1- با استفاده از گیت های NOT ، OR و AND ، مدار مبدل BCD به نمایشگر هفت قسمتی را که تنها قادر به نمایش اعداد یک رقمی باشد طراحی نمایید. سپس توسط نرم افزار Quartus برنامه آن را نوشته و خروجی را پس از شبیه سازی با تئوری مقایسه کنید. آیا جواب بدست آمده درست است ؟

به منظور نمایش اعداد چند رقمی، که به عنوان مثال علاوه بر رقم یکان دارای رقم دهگان نیز می‌باشند چه راه حل و مداری را پیشنهاد می‌نمایید. مدار مورد نظر را با استفاده از گیت‌های منطقی طراحی کرده و در انتها کلیه موارد فوق را برای این مدار نیز اجرا کنید.

تکالیف داخل آزمایشگاه

1- برنامه مدار مبدل BCD تک رقمی به نمایشگر هفت قسمتی را توسط نرم افزار مورد نظر اجرا و مقدمات پیاده سازی برروی برد را مهیا نمایید.

2- مراحل پیاده سازی طرح مورد نظر برروی برد را اجرا و نتایج حاصل را برروی نمایشگر هفت قسمتی مشاهده کنید. بدین منظور با استفاده از DIP SWITCH هایی که روی برد قرار دارد عدد BCD مورد نظر را وارد و عدد دسیمال متناظر با آن را بر روی نمایشگر هفت قسمتی مشاهده نمایید.

3- نتایج حاصل از اجرای برنامه برروی برد را با آنچه قبلاً بدست آورده اید، مقایسه نمایید. مدار مبدل اعداد دو رقمی بر روی نمایشگر هفت قسمتی را برروی FPGA یا CPLD پیاده سازی و نتایج حاصل از اجرای برنامه برروی برد را مشاهده و با آنچه قبلاً بدست آورده اید، مقایسه کنید.

نکته به منظور دستیابی به پایه‌های نمایشگر هفت قسمتی استفاده شده بر روی برد و کسب اطلاعات لازم در مورد ترتیب پایه‌های آن به پیوست مراجعه شود.

آزمایش دوم

جمع کننده ها

هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

- ✓ طراحی و شبیه سازی نیم- جمع کننده، تمام جمع‌کننده و جمع‌کننده 4- بیتی و مقایسه آن با تئوری
- ✓ پیاده‌سازی مدارهای طراحی شده برروی FPGA
- ✓ تست مدار پیاده‌سازی شده و بررسی خروجی‌های بدست آمده

تئوری آزمایش

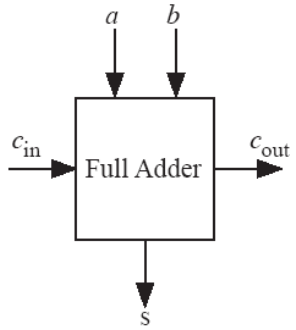
همان طور که از اهداف آزمایش نیز مشخص است، هدف نهایی در این آزمایش طراحی و پیاده سازی جمع کننده 4- بیتی و 16- بیتی می باشد. اما از آنجایی که ساده ترین و پایه ای ترین مدار در طبقه‌بندی مدارهای جمع کننده، مدار نیم- جمع کننده[1] است، لذا در ابتدا به طراحی مدار نیم-جمع‌کننده می‌پردازیم. پس از شبیه‌سازی و پیاده‌سازی این مدار و بررسی خروجی آن به طراحی مدار تمام- جمع‌کننده خواهیم پرداخت.

در بخش بعدی با بهره‌گیری از مدار تمام- جمع‌کننده طراحی شده (به عنوان یک بلوک با ورودی‌های و خروجی‌های مشخص) به طراحی مدار جمع‌کننده 4- بیتی خواهیم پرداخت. لازم به ذکر است که در این بخش نیز با توجه به اینکه خروجی نهایی مدار بر روی نمایشگر هفت قسمتی قابل مشاهده می‌باشد، لذا بکارگیری و استفاده از دیکد کننده دودویی به هفت قسمتی که در آزمایش اول طراحی نمودید را نیز تجربه خواهید نمود.

پیش از انجام آزمایش مختصری به تئوری مربوط به مدارهای جمع‌کننده می‌پردازیم. همان طور که می‌دانید مدار نیم- جمع‌کننده مداری است که دو عدد تک بیتی را با هم جمع و حاصل را به همراه بیت نقلی تولید می‌نماید. مدار تمام-جمع کننده[2] تک بیتی نیز مداری است که دو عدد تک بیتی را با بیت نقلی ورودی جمع و مجدداً حاصل را به همراه بیت نقلی تولید می‌نماید (شکل 1-4-2).

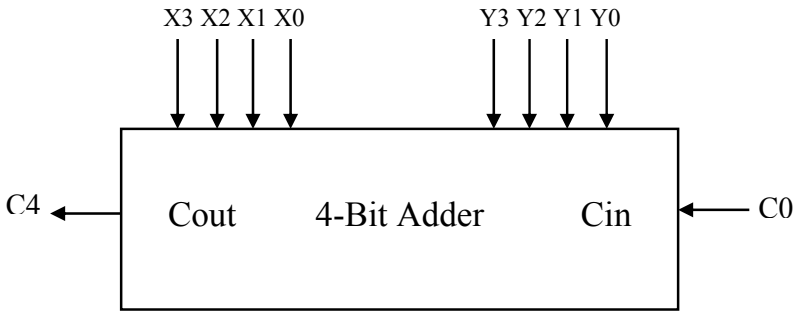
---

[1] Half-Adder
[2] Full-Adder



شکل 1-4-2 تمام- جمع کننده تک بیتی

تمام-جمع کننده را می‌توان به عنوان یک بلوک پایه در طراحی جمع کننده‌های n-بیتی مورد استفاده قرار داد. به عنوان مثال در طراحی جمع کننده 4-بیتی از 4 تمام-جمع کننده استفاده می‌شود. در این جمع کننده دو عدد 4-بیتی با احتساب بیت نقلی ورودی با هم جمع و حاصل که یک عدد 4-بیتی و بیت نقلی خروجی است، نتیجه می‌شود.



شکل 2-4-2 جمع کننده 4- بیتی

تکالیف پیش از آزمایش

پیش از ورود به آزمایشگاه و شروع آزمایش، مطالب این بخش را مطالعه نموده، موارد خواسته شده را انجام داده و بـه سـؤالات مطرح شده پاسخ دهید و نتایج بدست آمده را در گزارش خود ثبت نمایید.

1- نیم - جمع کننده

1-1) با استفاده از گیت های NOT، OR و AND مدار یک نیم جمـع کننـده را طراحـی نمـوده ، سـپس کـد Verilog آن را نوشته و توسط نرم افزار Quartus آن را اجرا نمایید. همچنین خروجی آن را پس از شبیه سازی با تئوری مقایسه نمایید. آیا جواب بدست آمده درست است ؟

2- تمام - جمع کننده

2-1) مدار یک تمام- جمع کننده را طراحی نموده و توسط نرم افزار Quartus برنامه آن را نوشته و خروجی حاصل از شبیه سازی را با تئوری مقایسه نمایید، آیا جواب بدست آمده درست است؟

3- جمع کننده 4 - بیتی

3-1) با استفاده از مدار تمام-جمع کننده (به عنوان یک بلوک طراحی) یک جمع‌کننده 4- بیتی طراحی نمایید. ورودی و خروجی های S_0 تا S_3، رقم نقلی خروجی و ارتباط داخلی مابین تمام جمع‌کننده‌ها را مشخص نمایید.

تکالیف داخل آزمایشگاه

1) نیم جمع کننده

1-1) برنامه نیم- جمع‌کننده را توسط نرم افزار Quartus اجرا و مقدمات پیاده‌سازی برروی برد را مهیا نمایید.

1-2) مراحل پیاده‌سازی طرح مورد نظر بر روی FPGA را اجرا نمایید. بدین منظور با استفاده از DIP SWITCH هایی که در روی برد قرار دارد دو بیت مورد نظر را وارد نمایید و خروجی را بر روی LED ها مشاهده نمایید.

1-3) نتایج حاصل از اجرای برنامه برروی برد را با آنچه قبلاً بدست آورده‌اید، مقایسه نمایید.

2) تمام جمع کننده

2-1) برنامه تمام– جمع‌کننده را توسط نرم افزار Quartus اجرا و مقدمات پیاده‌سازی برروی برد را مهیا نمایید.

2-2) مراحل پیاده‌سازی طرح مورد نظر بر روی FPGA را اجرا نمایید. بدین منظور با استفاده از DIP SWITCH هایی که در روی برد قرار دارد دو بیت مورد نظر و رقم نقلی ورودی را وارد نمایید و خروجی را بر روی LED ها مشاهده نمایید.

2-3) نتایج حاصل از اجرای برنامه برروی برد را با آنچه قبلاً بدست آورده اید، مقایسه نمایید.

3) جمع کننده 4 بیتی

3-1) برنامه جمع کننده 4-بیتی را توسط نرم افزار Quartus اجرا و مقدمات پیاده سازی برروی برد را مهیا نمایید.

3-2) مراحل پیاده‌سازی طرح مورد نظر بر روی FPGA را اجرا نمایید. بدین منظور با استفاده از DIP SWITCH هایی که روی برد قرار دارند دو عدد BCD مورد نظر و رقم نقلی ورودی را وارد نمایید و خروجی را روی نمایشگرهای هفت قسمتی مشاهده نمایید.

به منظور نمایش خروجی جمع کننده 4-بیتی برروی نمایشگر هفت قسمتی، از دیکد کننـده‌ای کـه در آزمـایش اول طراحـی نمودید استفاده نمایید.

3-3) نتایج حاصل از اجرای برنامه برروی برد را با آنچه قبلاً بدست آورده اید، مقایسه نمایید.

پروژه پیشنهادی

با استفاده از برنامه جمع‌کننده 4- بیتی یک جمع‌کننده 16- بیتی طراحی نمایید. در نهایت برنامه را بر روی برد پیاده سـازی نمایید و نتایج حاصل را برروی نمایشگر هفت قسمتی مشاهده کنید.

آزمایش سوم

شمارنده ها

هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

- ✓ آشنایی با مفاهیم طراحی مدارهای ترتیبی
- ✓ طراحی و پیاده سازی شمارنده‌ها

تئوری آزمایش

شمارنده ها ، در طراحی سیستم های دیجیتال به طور عام و بویژه در پردازشگرها بکار می روند. این مدارها برای تولید مقادیر زمانی به منظور تنظیم و کنترل عملیات در یک دستگاه دیجیتال مفید هستند. یک شمارنده ضرورتاً یک ثبات است که به ازای پالسهای ورودی به آن، دنباله‌ای از حالتهای از پیش تعیین شده را ایجاد می‌کند. شمارنده‌ها را می توان به دو دسته سنکرون و آسنکرون (موج گونه) طبقه بندی نمود.

اما نکته بسیار مهم در مدارهای ترتیبی، نحوه طراحی آنها است. در طراحی مدارهای ترتیبی برخلاف مدار ترکیبی که با ترسیم یک جدول درستی طراحی مدار کاملاً مشخص می‌شود، به جدول و دیاگرام حالت نیز نیاز است. طراحی مدار ترتیبی شامل تعیین فلیپ فلاپ و سپس یافتن ساختار مدار ترکیبی است که همراه با فلیپ فلاپ ها نیازهای تعیین شده را برآورده می سازد. روش طراحی مطرح شده شامل مراحل زیر است:

1- توصیف عملکرد مدار توسط دیاگرام حالت یا دیاگرام زمانی.

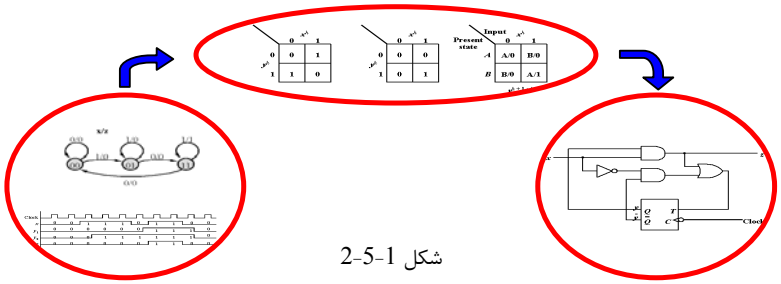
2- تهیه جدول حالات با توجه به اطلاعات مفروض.

3- ساده سازی جدول حالات، در صورت امکان.

4- تعیین تعداد و نوع فلیپ فلاپ‌های مورد نیاز و اختصاص یک سمبل به هر کدام.

5- به دست آوردن توابع خروجی مدار و توابع ورودی فلیپ فلاپ ها، با بکارگیری روش نقشه یا هر روش ساده سازی دیگر.

6- ترسیم دیاگرام منطقی مدار.



شکل 1-5-2

تکالیف پیش از آزمایش

پیش از ورود به آزمایشگاه و شروع آزمایش مطالب این بخش را مطالعه، موارد خواسته شده را انجام و به سؤالات مطرح شده پاسخ دهید و نتایج بدست آمده را در گزارش خود ثبت نمایید.

1) طراحی شمارنده 4 - بیتی سنکرون (3- افزا)

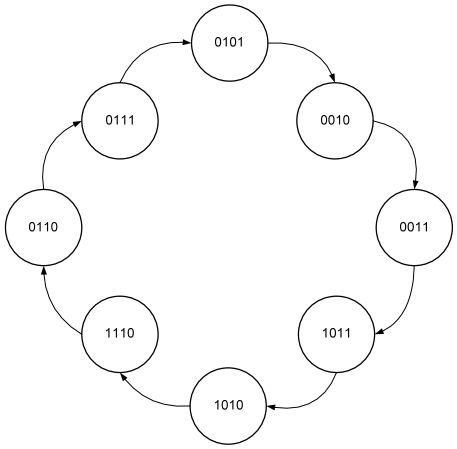
1-1) با استفاده از فلیپ فلاپ های نوع JK شمارنده 4 - بیتی سنکرونی طراحی نمایید که دارای یک ورودی X باشد. این ورودی جهت شمارش رو به بالا (X=1) و رو به پایین (X=0) را تعیین می‌نماید. شمارنده مورد نظر 3- افزا است . بدین معنی که باید اعداد 0 تا 15 را به طوری که اختلاف دو عدد متوالی همواره 3 باشد، بشمارد.

1-2) برنامه شمارنده مورد نظر را نوشته و تست نمایید.

1-3) برنامه را به منظور نمایش خروجی برروی نمایشگر هفت قسمتی تکمیل نمایید.

2) طراحی شمارنده اختصاصی (شبیه کد گری)

2-1) مدار شمارنده‌ای را طراحی نمایید که دیاگرام حالت آن به صورت زیر باشد. در طراحی خود از فلیپ فلاپ های نوع D استفاده کنید. مراحل طراحی خود را در گزارش ثبت نمایید.



شکل 2-5-2

2-2) برنامه مدار طراحی شده را نوشته و خروجی مدار را توسط شبیه سازی مشاهده و تست نمایید.

2-3) برنامه را به منظور نمایش خروجی برروی نمایشگر هفت قسمتی تکمیل نمایید.

راهنمایی: مدار مورد نظر یک شمارنده دو دویی می‌باشد که تنها اعداد 5 , 7 , 6 , 14 , 10 , 11 , 3 , 2 را می شمارد . این مدار نوعی کد کننده گری[3] است.

تکالیف داخل آزمایشگاه

نکات : دانشجویان عزیز برای مشاهده عملکرد مدارهای طراحی شده باید مقدمات زیر را فراهم نمایند:

1) برای قابل رویت شدن نتایج باید پریود پالس ساعت را در حدود 500 ms قرار داد. بدین منظور می‌توانید از مدارات مقسم فرکانس و پایه پنج F0o*=8KHz U0 استفاده نمایید.

2) برروی برد چهار LED و 4 عدد نمایشگر هفت قسمتی وجود دارند. از این LED ها و نمایشگرها به منظور نمایش خروجی شمارنده ها استفاده نمایید.

3) از سوئیچ‌های موجود بر روی برد به منظور فرمان دادن یا بار نمودن شمارنده‌ها استفاده نمایید.

1- شمارنده 3- افزا طراحی شده را برروی برد پیاده سازی نموده و با استفاده از سوئیچ های موجود برروی برد نحوه شمارش (بالا شمار - پایین شمار) را مشخص و نتایج حاصل را برروی نمایشگرهای هفت قسمتی مشاهده نمایید.

2- شمارنده اختصاصی کد گری طراحی شده را برروی برد پیاده سازی نموده و نتایج حاصل را برروی نمایشگرهای هفت قسمتی مشاهده نمایید.

[3] Gray Coder

آزمایش چهارم

دیاگرام حالت

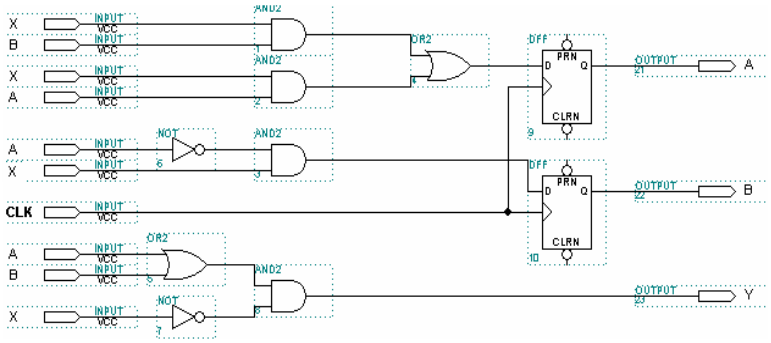
هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ کسب مهارت بیشتر در زمینه طراحی مدارات ترتیبی

تئوری آزمایش

ترتیب زمانی ورودی و خروجی ها و حالات فلیپ فلاپ ها را می‌توان در جدول حالات برشمرد. مدار شکل 1-6-2 که در کتاب مانو آمده است دارای جدول حالتی به صورت جدول 1-6-2 می‌باشد.

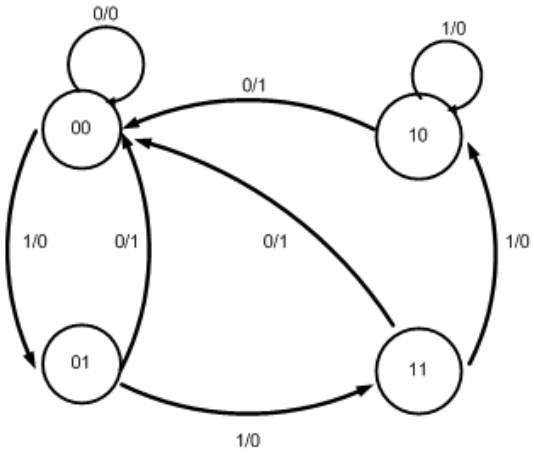


شکل 1-6-2

حالت فعلی	ورودی	حالت بعدی	خروجی
A B	X	A B	Y
0 0	0	0 0	0
0 0	1	0 1	0
0 1	0	0 0	1
0 1	1	1 1	0
1 0	0	0 0	1
1 0	1	1 0	0
1 1	0	0 0	1
1 1	1	1 0	0

جدول 1-6-2 جدول حالات مدار شکل 1-6-2

شکل شماره 1-6-2 در واقع با استفاده از جدول 1-6-2 طراحی شده است. اطلاعات موجود در یک جدول حالات را می‌توان به صورت گرافیکی تحت عنوان دیاگرام حالات نمایش داد. شکل 2-6-2 دیاگرام حالات جدول 1-6-2 می‌باشد.



شکل 2-6-2 دیاگرام حالات جدول 1-6-2

با توجه به مقدماتی که بیان شد و مطالبی که از آزمایش شمارنده نیز بیاد دارید، دریافتیم که به آسانی می‌توان با داشتن جدول حالت یا دیاگرام حالت، مدار ترتیبی مورد نظر را طراحی نمود. طراحی مدارات ترتیبی دارای مراحل مشخصی می‌باشد، اما در طراحی مدارات ترتیبی که دارای نظم و روال منظمی نبوده و از مجموعه حالات منظمی پیروی نمی‌نمایند، علاوه بر نکات ذکر شده در آزمایش شمارنده، بیان چند نکته دیگر نیز مهم به نظر می‌رسد. در ادامه علاوه بر مرور نکات ذکر شده قبلی، به نکات ضروری جدید را نیز بیان خواهیم نمود:

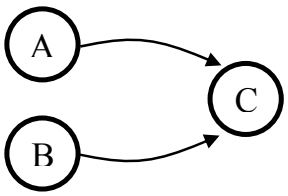
1- بیان منطقی مسئله با توجه به بیان لفظی آن.

2- رسم دیاگرام حالات ، با توجه به صورت مسئله .

3- دستیابی به جدول حالات سیستم با استفاده از دیاگرام حالات .

4- ساده سازی جدول حالات در صورت امکان .

* حالت معادل حالتی است که از آن دو حالت سیستم به حالت مشخصی رود که خروجی مشخصی دارد.



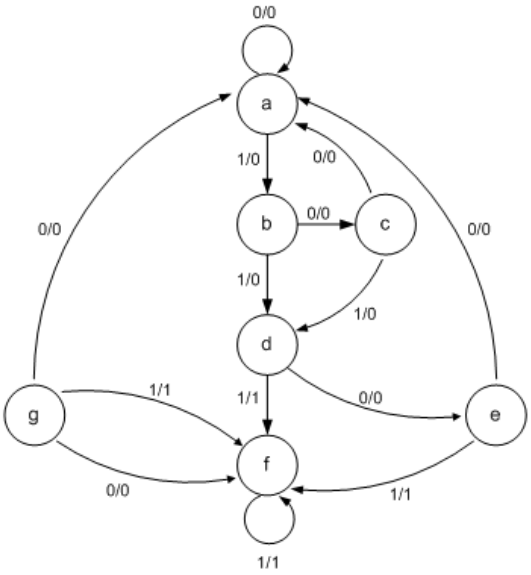
5- مشخص شدن تعداد فلیپ فلاپ ها، با توجه به حالت های باقیمانده.

6- تخصیص حالات.

7- بدست آوردن توابع ورودی فلیپ فلاپ ها به کمک جدول حالات نهائی.

8- ساده کردن توابع حاصل و رسم مدار .

نکته مهم در طراحی مدارات ترتیبی (بغیر از مدارات شمارنده) ساده سازی جدول حالات است. اکنون مدار مربوط به دیاگرام حالاتی که در شکل 3-6-2 آمده است را با بکارگیری مطالب فوق طراحی می‌نمائیم.



شکل 3-6-2 دیاگرام حالات

	حالت بعدی		خروجی	
	X=0	X=1	X=0	X=1
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*g*	*f*	0	1
g	*a*	*f*	0	1

جدول 2-6-2 جدول حالات شکل 9-2

با توجه به جدول 2-6-2 حالت‌های g و e دارای خروجی‌ها و حالت‌های یکسان هستند، بنابراین g حذف شده و بجای g، e قرار می‌دهیم.

	X=0	X=1	X=0	X=1
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*e*	*f*	0	1

جدول 2-6-3 حدول حالات با حذف حالت g

در جدول 2-6-4 نیز d و f معادل هستند. با جایگزین کردن d به جای f , یک حالت دیگر نیز از سیستم کم خواهد شد.

	X=0	X=1	X=0	X=1
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*d*	0	1
e	*a*	*d*	0	1

جدول 2-6-4 جدول حالات با حذف f

مرحله بعدی تخصیص حالات می باشد. این کار با استفاده از سه فلیپ فلاپ C , B , A صورت خواهد گرفت.

	A	B	C
a	0	0	0
b	0	0	1
c	0	1	0
d	0	1	1
e	1	0	0

جدول 5-6-2 تخصیص حالات

اکنون جدول حالات فلیپ فلاپ ها را با استفاده از جدول 5-6-2 طراحی نمائید.

تکالیف پیش از آزمایش

1- آزمایش اول

1-1- برنامه مدار شکل 1-6-2 را در نرم افزار Quartus پیاده‌سازی نمائید.

1-2- ورودی را به یک سوئیچ، خروجی مدار را به یک LED تک رنگ و حالات مدار را به دو LED تک رنگ متصل نمائید.

2- آزمایش دوم

2-1- جدول حالات فیلپ فلاپها را با استفاده از جدول 5-6-2 ترسیم نمائید.

2-2- با فلیپ فلاپ JK مدار ترتیبی جدول 5-6-2 را رسم نمائید.

بخش سوم
ثبات ها

از مهم‌ترین بخش‌های تشکیل دهنده سیستم‌های دیجیتال ثبات ها می‌باشند. در این سیستم‌ها بسیاری از اعمال داخلی توسط ثبات ها یا داده‌هایی که در آنها ذخیره می‌شوند انجام می‌گیرند.

هدف اصلی این بخش طراحی مهم‌ترین المان یک کامپیوتر پایه است. ثبات ها به صورت‌های مختلفی در داخل سیستم‌های دیجیتال یا کامپیوتر مورد استفاده قرار می‌گیرند. کاربردهای مختلفی در دروس مدار منطقی و معماری کامپیوتر برای شما ارائه گردیده است که در این آزمایش‌ها مشاهده می‌کنید. این بخش از سه آزمایش تشکیل شده است.

در آزمایش اول شما با طراحی یک ثبات که به صورت سری و موازی قابل خواندن و نوشتن می‌باشد آشنا می‌گردید. همچنین نحوه طراحی و پیاده‌سازی ثبات های انتقالی[4] چرخشی و ریاضی به شما آموزش داده خواهدشد.

در آزمایش دوم ثبات هایی که دارای قابلیت‌های مختلفی نظیر متمم یک، متمم دو افزایشی یک و کاهش یک است را طراحی می‌نمایید.

آزمایش سوم تکمیل کننده دو آزمایش قبل است. در این آزمایش شما باید یک ثبات که دارای همه قابلیت‌هایی که در آزمایش‌های اول و دوم انجام داده بودید به صورت مجتمع طراحی نمایید.

[4] Shift Register

آزمایش پنجم

ثبات 1

هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

- ✓ طراحی و پیاده سازی یک ثبات انتقالی
- ✓ طراحی و پیاده سازی یک ثبات 16 بیتی که نحوه ورودی و خروجی آن به شکل زیر باشد :
  - قابلیت ثبت موازی و سری و خواندن موازی و سری[5]
  - ثبات انتقالی چرخشی به چپ و راست
  - ثبات انتقالی ریاضی به چپ و راست

تئوری آزمایش

ثبات ها یکی از قسمتهای مهم ساختار یک سیستم پردازشگر دیجیتال می‌باشند. ثبات ها اغلب برای ذخیره موقت اطلاعات باینری استفاده می شوند و شامل مجموعه ای از فلیپ فلاپ ها هستند. یک ثبات n بیتی شامل n فلیپ فلاپ است. ثباتها را می توان از لحاظ ورودی و خروجی به چهار دسته تقسیم نمود. این تقسیم بندی عبارتست از :

- ✓ ورودی موازی - خروجی موازی (PIPO)
- ✓ ورودی سری - خروجی موازی (SIPO)
- ✓ ورودی موازی - خروجی سری (PISO)
- ✓ ورودی سری - خروجی سری (SISO)

یک ثبات انتقالی از مجموعه ای از فلیپ فلاپ ها تشکیل شده است که به صورت زنجیره‌ای به هم متصل هستند و ورودی هر فلیپ فلاپ به خروجی فلیپ فلاپ قبلی متصل است. از ثبات‌های بسیار مهم می توان به ثبات انتقالی اشاره نمود. ثبات انتقال بنا به نوع سیگنال کنترلی آن ممکن است اطلاعات را به سمت راست یا به سمت چپ انتقال دهد. از دیدگاه زمان بندی ثبات ها را می توان به دو دسته سنکرون و آسنکرون تقسیم بندی نمود. ثبات سنکرون ثباتی است که دارای سیگنال تحریک یکسان برای هر فلیپ فلاپ

---

5 SISO, SIPO, PIPO PISO

می باشد. در ثبات آسنکرون این عمل وجود ندارد. در کامپیوتر پایه بنا به نوع کاربردی که درنظر گرفته شده است دو نوع شیفت ریاضی و چرخشی اضافه گردیده است. بنابراین در کنار هر ثبات انتقالی یک فیلیپ فلاپ قرار خواهدگرفت که به وسیله آن ثبات انتقالی چرخشی و ریاضی با ثبات انتقالی مرسوم تکمیل می‌گردد.

تکالیف پیش از آزمایش

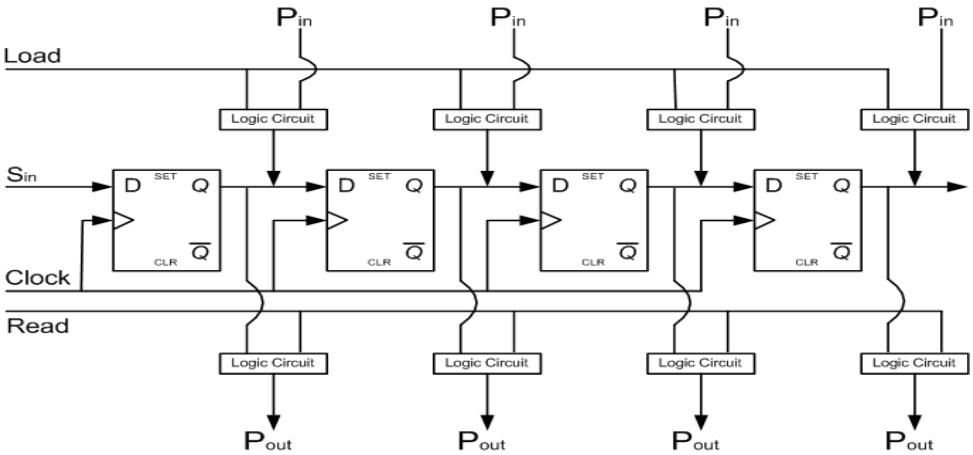
1) در این آزمایش ابتدا یک ثبات انتقالی طراحی نمایید که دارای یک سیگنال کنترلی است. این سیگنال کنترلی مشخص کننده حالت انتقال به چپ یا به راست می‌باشد. لازم به ذکر است که در کلیه طراحی‌ها از فلیپ فلاپ D استفاده نمایید. برنامه ثبات مورد نظر را نوشته و خروجی آن را تحلیل نمایید.

2) ثبات انتقالی که به صورت موازی و سری ثبت و به صورت موازی و سری خوانده میشود را طراحی نمایید.

3) اکنون که طراحی شماتیک کامل گردید کد Verilog ثبات انتقالی را که دارای قابلیت‌های ذکر شده در بند2 می‌باشد را تولید نمایید.

4) با دانستن نحوه طراحی یک ثبات انتقالی با DFF، اکنون مراحل 2 و 3 را برای ثبات انتقالی چرخشی و ریاضی تکمیل کنید . در این مرحله شما می توانید تمامی قابلیت‌های فوق را بر روی یک ثبات پیاده‌سازی نمایید.

راهنمایی : هر یک از این ثبات ها نیازمند سیگنالهای کنترل نظر CLK , Read و Load و ... می‌باشند که بنا به نوع تبادل باید در نظر گرفته شوند. شکل زیر به طور خلاصه راهنمائی های لازم را برای شما خواهد داشت. لازم به ذکر است که این بلوک دیاگرام بسیار کلی است و کلیه حالات فوق را در بر دارد. بنابراین در هرکدام از مدارات فوق بخشی از این شکل بکار می‌رود.



شکل 1-7-3 بلوک دیاگرام کلی مداراها

تکالیف داخل آزمایشگاه

نکات : دانشجویان عزیز برای مشاهده عملکرد مدارهای طراحی شده باید مقدمات زیر را فراهم نمایند.

- ✓ برای قابل رویت شدن باید پریود Clock را در حدود 500ms قرار داد.
- ✓ برروی برد، چهار LED یک رنگ وجود دارد از این LED ها به عنوان بیت‌های خروجی ثبات‌ها استفاده نمایید. از سوئیچ-های موجود برروی برد به عنوان سیگنالهای کنترلی و ورودی استفاده نمایید.

1) مدار ثبات انتقال PIPO، PISO، SIPO وSISO را برروی برد پیاده سازی و تست نمایید. نتایج بدست آمده را در گزارش خود ثبت نمایید.
2) مدار ثبات انتقال چرخشی به چپ و راست را برروی برد پیاده سازی و تست نمایید. نتایج بدست آمده را در گزارش خود ثبت نمایید.
3) مدار ثبات انتقال ریاضی به چپ و راست را برروی برد پیاده سازی و تست نمایید. نتایج بدست آمده را در گزارش خود ثبت نمایید.
4) مدار ثبات انتقالی که همه قابلیتهای فوق را داراست برروی برد پیاده سازی و تست نمایید. نتایج بدست آمده را در گزارش خود ثبت نمایید.

آزمایش ششم

ثبات 2

هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

طراحی و پیاده سازی یک ثبات 16 بیتی که به شکل زیر باشد :

- ✓ با قابلیت متمم یک
- ✓ با قابلیت متمم دو
- ✓ با قابلیت افزایشی یک
- ✓ با قابلیت کاهشی یک

تئوری آزمایش

در این آزمایش شما با کاربردهای دیگری از ثبات‌ها آشنا خواهید شد. این کاربردها، که در بالا به آنها اشاره شده است، در قسمت‌های مختلف یک سیستم دیجیتال مورد استفاده قرار می‌گیرند. به‌طور مثال در سیستم کامپیوتر پایه برای انجام عملیات افزایش یک یا کاهش یک به راحتی در یک سیکل با سیگنال‌های INC یا DEC انجام خواهند شد. طراحی چنین ثباتی که دارای قابلیت‌های فوق است، در سیکل‌های اجرائی بسیاری از دستورالعمل‌ها صرفه‌جویی خواهد نمود.

شکل 1-8-3

تکالیف پیش از آزمایش

1) در این آزمایش ابتدا یک ثبات با قابلیت افزایشی یک با استفاده از فلیپ فلاپ D طراحی نمایید. سپس کد Verilog مرتبط با این مدار را نوشته و تست کنید.

2) مرحله 1 را به‌طور کامل برای سه کاربرد دیگر ، طراحی و تست می‌نمایید.

یک ثبات طراحی و تست نمایید که دارای هر چهار قابلیت فوق یعنی متمم یک، متمم 2 ، افزایشی یک و کاهشی یک باشد. سپس کد Verilog مرتبط با این مدار را نوشته و تست نمایید.

تکالیف داخل آزمایشگاه

نکات : دانشجویان عزیز برای مشاهده عملکرد مدارهای طراحی شده باید مقدمات زیر را فراهم نمایند.

- ✓ برای قابل رویت شدن باید پریود Clock را در حدود 500ms قرار داد.
- ✓ برروی برد، چهار LED یک رنگ وجود دارد از این LED ها به عنوان بیت‌های خروجی ثبات‌ها استفاده نمایید.
- ✓ از سوئیچ‌های موجود برروی برد به عنوان سیگنالهای کنترلی و ورودی استفاده نمایید.

برنامه ثبات با قابلیت های فوق را برروی برد پیاده سازی و تست نمایید. نتایج بدست آمده را در گزارش خود ثبت نمایید.

آزمایش هفتم

ثبات 3

هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

طراحی و پیاده سازی یک ثبات 4 بیتی که به شکل زیر باشد :

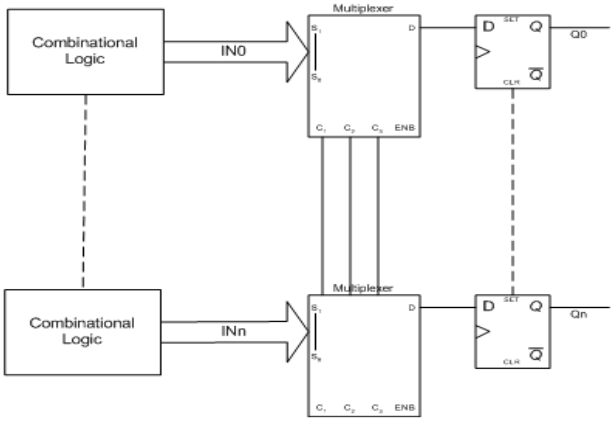
- ✓ SIPO, SISO, PIPO, PISO
- ✓ Circulate Right shift, Circulate Left shift
- ✓ Arithmatic Right shift, Arithmatic Left shift
- ✓ One’s complement, Two’s complement
- ✓ INC, DEC
- ✓ Clear

تئوری آزمایش

این آزمایش تکمیل کننده دو آزمایش قبل است. شما در آزمایش‌های گذشته با طراحی و پیاده‌سازی هر یک از عملگرهای فوق به صورت مجزا آشنا شدید. در این آزمایش شما با دانش قبلی که در زمینه طراحی تک تک آنها به دست آورده‌اید. اکنون به راحتی می‌توانید یک ثبات با قابلیت‌های فوق طراحی نمایید. همچنانکه در آزمایش قبل ذکر شد، داشتن چنین ثباتی باعث کاهش سیکل های اجرائی یک دستورالعمل در کامپیوتر می‌گردد. به عبارت دیگر ریز دستورالعمل‌های مورد نیاز برای یک دستورالعمل کاهش پیدا می‌کند.

تکالیف پیش از آزمایش

در این آزمایش با استفاده از Multiplexer ورودی های فلیپ فلاپ D مشخص می‌گردد. شکل 1-9-3 بلوک دیاگرام کلی مدارات مورد نظر را نمایش می دهد. مدار مورد نظر را ترسیم ، برنامه آن را نوشته و خروجی را تحلیل نمایید.



شکل1-9-3 بلوک دیاگرام کلی مدارات مورد نظر

تکالیف داخل آزمایشگاه

نکات : دانشجویان عزیز برای مشاهده عملکرد مدارهای طراحی شده باید مقدمات زیر را فراهم نمایند.

- ✓ برای قابل رویت شدن باید پریود Clock را در حدود 500ms قرار داد.
- ✓ برروی برد، چهار LED یک رنگ وجود دارد از این LED ها به عنوان بیت‌های خروجی ثبات‌ها استفاده کرده و ثبات‌ها را 4 بیتی طراحی کنید. از سوئیچ‌های موجود برروی برد به عنوان سیگنالهای کنترلی Multiplexer استفاده نمایید.

برنامه ثبات با قابلیت های فوق را برروی برد پیاده سازی و تست نموده و نتایج بدست آمده را در گزارش خود ثبت کنید .

بخش چهارم
کامپیوتر مبنا

آزمایش هشتم

واحد محاسبه و منطق[۶]

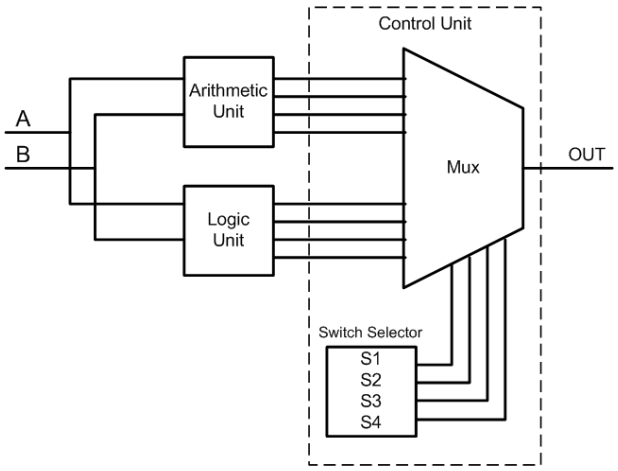
هدف

در این آزمایش اهداف زیر دنبال می‌شوند :

- ✓ آشنایی با ALU و ساختار آن
- ✓ طراحی و پیاده سازی مدارهای محاسباتی ، منطقی و ثباتی یک CPU

تئوری آزمایش

بخش محاسبات و منطق یکی از مهمترین قسمت های یک CPU می‌باشد. به طور کلی ALU از سه قسمت محاسبات، منطق و کنترل تشکیل شده است. هر بخش با توجه به وظایفی که دارد یک سری عملکردهایی را انجام می‌دهد. قسمت محاسباتی اعمالی نظیر جمع، تفریق، ضرب و تقسیم را بر عهده دارد. در قسمت منطقی عملیاتی نظیر AND, OR, XOR, NOT صورت می‌پذیرد و در نهایت قسمت کنترل نیز وظیفه تعیین واحد عملیاتی و عملیات مورد نظر را بعهده دارد. در این آزمایش هدف پیاده سازی یک ALU بسیار ساده می‌باشد. شکل 1-10-4 بلوک دیاگرام سیستم مورد نظر را نمایش می‌دهد.



شکل 1-10-4 بلوک دیاگرام ALU

[6] ALU

S1 S0 cin

جدول زیر کدهای عملیاتی را که توسط بخش محاسبات و منطق می‌شود، نمایش می‌دهد. دانشجویان با استفاده از سوئیچ‌های موجود برد، عملکرد مورد نظر را مشخص و داده ورودی را به ALU اعمال می‌نمایند. لازم به ذکر است که تمام عملیات های مورد نظر حداکثر 4 می‌باشند.

Add	0	0	0	
Add with carry	0	0	1	
Sub with borrow	0	1	0	
Sub	0	1	1	
A	1	0	0	
INC	1	0	1	
DEC	1	1	0	
A	1	1	1	

پیش از آغاز آزمایش لازم است تا کمی بیشتر در مورد بلوک‌های سیستم به بحث بپردازیم. در بخش محاسباتی دانشجویان عزیز می‌توانند از بلوک جمع کننده‌ای که در آزمایش2 طراحی شد به عنوان جمع 4 - بیتی استفاده نمایند. با استفاده از همان مدار نیز می‌توانند افزایش یک واحد را نیز پیاده سازی نمایند. اما مدار تفریق کننده و کاهش به اندازه یک واحد را که در آزمایش‌های گذشته به آنها نپرداخته‌اند می‌بایست از ابتدا طراحی نمایند. البته طراحی تفریق کننده 4 - بیتی نیز بر پایه تمام جمع کننده امکان پذیر است و از اینرو با جمع کننده 4 - بیتی اختلاف بسیار کوچکی دارد.

به منظور سادگی سیستم، بخش کنترلی تنها به عنوان یک مالتی‌پلکسر عمل می‌نماید. ورودی این بلوک حاصل کلیه اعمال مورد نظر (محاسباتی: جمع، تفریق، ... و منطقی: AND، OR، ...) است و با توجه به سیگنال کنترلی، ورودی مورد نظر به خروجی منتقل می‌شود.

	S1	S0
AND	0	0
OR	0	1
XOR	1	0
NOT	1	1

جدول 1-10-4

تکالیف پیش از آزمایش

تا کنون کلیه مدارهای مورد نیاز در این بخش در آزمایش‌های قبل پیاده سازی شده‌اند، بجز تفریق کننده 4 - بیتی که می‌بایست برنامه آن را نوشته و پیاده سازی نمایید.

تکالیف داخل آزمایشگاه

نکات : دانشجویان عزیز برای مشاهده عملکرد مدارهای طراحی شده باید مقدمات زیر را فراهم نمایند.

- ✓ برروی برد چهار عدد LED و نمایشگر هفت قسمتی وجود دارد. از این LED ها و نمایشگرها بـه منظـور نمـایش خروجی ALU استفاده نمایید.
- ✓ از سوئیچ‌های موجود بر روی برد به عنوان ورودی‌ها، سیگنال‌های کنترلی و رقم نقلی ورودی استفاده نمایید.

1) برنامه مدار ترسیم شده را نوشته و برروی FPGA برنامه ریزی نمایید.

2) تست های لازم را انجام و نتایج را در گزارش خود ثبت نمایید.

آزمایش نهم
گذرگاه داده[۷]

هدف

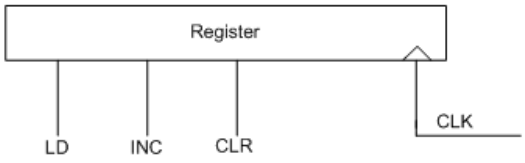
در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با BUS و ساختار آن
- ✓ طراحی و پیاده سازی مدارهای محاسباتی ، منطقی و ثباتی یک CPU

تئوری آزمایش

BUS یکی از واسطهای بسیار مهم در سیستم‌های دیجیتالی است. در این سیستم‌ها برای ارسال و دریافت داده از BUS به عنوان محیط انتقال استفاده می‌گردد. همانطور که در معماری کامپیوتر با آن آشنا شده‌اید به دو طریق BUS ها قابل پیاده‌سازی هستند. آن دو راه به کارگیری Multiplexer و استفاده از بافر tri-state می‌باشد. در کامپیوترها به علت دوطرفه بودن BUS ، امکان استفاده از Multiplexer برای یک خط که خواندن و نوشتن از روی آن صورت می‌گیرد امکان‌پذیر نمی‌باشد. بنابراین این عمل از طریق tri-state انجام خواهد شد.

در فصل‌های قبل با نحوه طراحی ثبات ها آشنا شدید. در سیستم کامپیوتر پایه، یک ثبات ممکن است دارای قابلیت‌های مختلفی باشد. اما در کامپیوتر پایه‌ای که برای این آزمایشگاه طراحی می شود یک ثبات باید قابلیت های Increment، LD ، Clear و Read را دارا باشد (شکل 1-11-4) .



شکل 1-11-4 ثبات داخلی کامپیوتر پایه

لازم به ذکر است که ثباتهای مذکور ثبات های داخلی کامپیوتر و مورد استفاده در قسمت کنترلی و BUS خواهند بود. نکته بسیار مهم نحوه ارتباط این ثبات ها با BUS سیستم می باشد. همانطور که می‌دانید به علت دو طرفه بودن BUS داده، خواندن و نوشتن از یک مسیر صورت خواهد گرفت. بنابراین هر ثبات باید در زمان مورد نیاز BUS را در اختیار گیرد و سپس بعد از اتمام ارسال یا دریافت

[7] BUS

آن را رها نماید. با توجه به مطالب فوقT در قسمت ورودی هر ثبات با سیگنال کلاک و LD مشکل وارد شدن داده جدید به ثبات حل می‌شود.

اما در قسمت خروجی باید در مسیر هر بیت خروجی یک tri-state قرار گیرد که با سیگنال Read هر ثبات فعال شده و بعد از خواندن، خروجی ثبات به صورت امپدانس بالا درآید. با طراحی چنین ثباتی به راحتی می‌توان تعداد زیاد ثبات را که از یک BUS واحد استفاده می‌نمایند، به یکدیگر متصل نمود.

اکنون با داشتن یک ثبات که قابلیت‌های اساسی سیستم کامپیوتر پایه را دارا می‌باشد، به معرفی ثبات های این پردازشگر پایه می‌پردازیم.

ثبات های مهم این سیستم عبارتند از:

1) ثبات PC[8] : آدرس شروع برنامه و خط بعدی برنامه را دارا می‌باشد.
2) ثبات AR[9] : آدرس دستورالعمل یا داده را برای حافظه تأمین می‌نماید.
3) ثبات IR[10] : کد باینری دستورالعمل از طریق حافظه در این ثبات قرار می‌گیرد.
4) ثبات DR[11] : ثباتی که محتویات داده هر عملگر که براساس حافظه است در آن قرار می‌گیرد. به تعبیر دیگر، هر یک از اعمال ریاضی و منطقی که با دو داده انجام میشود یک طرف آن در DR خواهد بود.
5) AC : ثبات پردازنده که مهم‌ترین ثبات پردازشگر و تمام اعمال ریاضی، منطقی ثباتی، ورودی و خروجی با این ثبات انجام می‌شود.
6) TR[12] : ثبات موقتی که در حین انجام بعضی از دستورالعمل‌ها ، داده به طور موقت در آن ذخیره خواهدشد.
7) INPR[13] ثباتی که برای ورود داده به داخل کامپیوتر مورد استفاده قرار می گیرد.
8) OUTR : برای ارسال داده به خارج از کامپیوتر از این ثبات استفاده می‌شود.

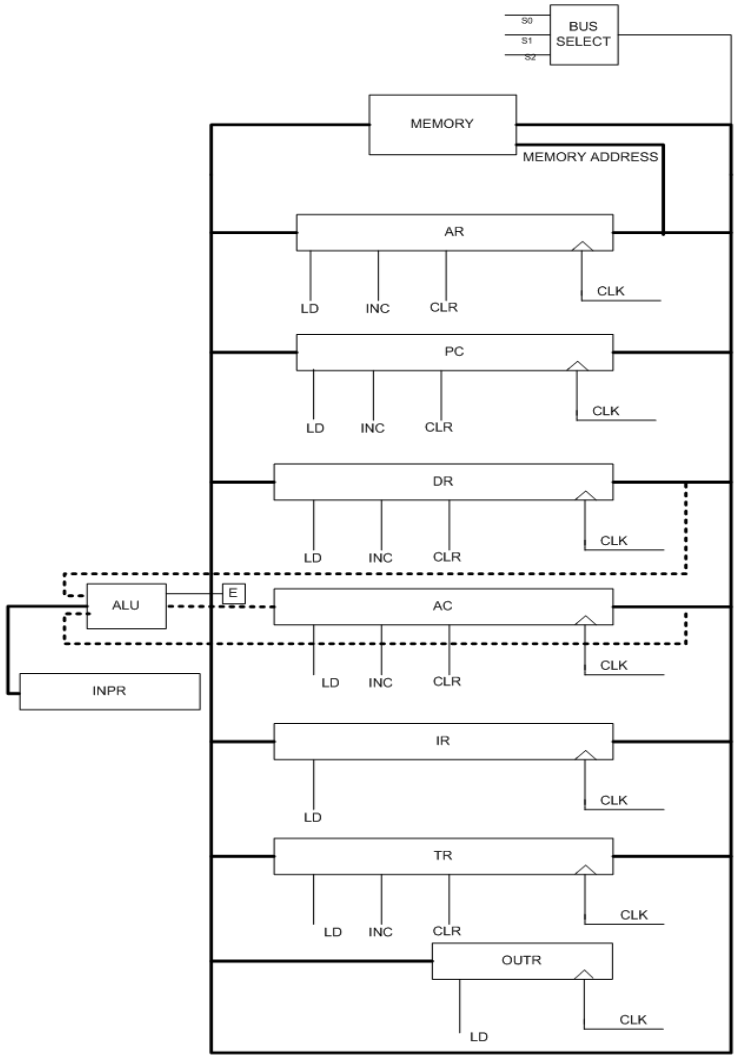
شکل2-11-4 نحوه ارتباطات بین این ثبات ها و سیگنال‌های کنترل مورد نیاز آنها را نمایش می‌دهد.

[8] Program Counter
[9] Address Register
[10] Instruction Register
[11] Data Register
[12] Temporary Register
[13] Input Register

تکالیف پیش از آزمایش

1- ثباتی 16 بیتی مطابق با شکل 1-11-4 طراحی نمایید. در این ثبات از فلیپ فلاپ‌های D استفاده میشود و در قسمت خروجی باید در مسیر هر بیت خروجی یک tri-state قرار گیرد که با سیگنال Read فعال شده و بعد از خواندن، خروجی ثبات به صورت امپدانس بالا درآید.

2- شکل 2-11-4 را با استفاده از ثباتی که در بند اول طراحی نموده‌اید، پیاده‌سازی نمایید. در این قسمت به جای حافظه یک ثبات و به جای ALU یک بافر قرار دهید. سپس سپس کد Verilog مرتبط با این مدار را نوشته و تست نمایید.



شکل 2-11-4 نحوه ارتباطات بین ثباتها در BUS

تکالیف داخل آزمایشگاه

3- در این آزمایش شما باید در Simulator ، یک محیط تست برای برنامه BUS بند 2 ، نوشته و به طور کامل ارتباطات بین ثبات های مختلف را تست نمایید.

نکته: این BUS به عنوان یکی از المان‌های مهم کامپیوتر پایه می‌باشد. بنابراین باید هر گروه تا پایان ترم تمامی فایل‌های مربوط به آن را نگهداری نماید.

اکنون باید BUS و ثبات های آن را به ALU متصل نمایید و دوباره قسمت‌های ارتباطی را به همراه ALU تست نمایید.

آزمایش دهم

واحد کنترل[۱۴]

هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با طراحی واحد کنترل کننده و ساختار آن
- ✓ طراحی و پیاده سازی واحد کنترل کننده یک CPU

تئوری آزمایش

در آزمایش‌های قبل با بعضی از قسمت‌های مهم کامپیوتر پایه آشنا شدید. در این آزمایش، بعد از آشنایی مختصری که با سازمان کامپیوتر پیدا می‌کنید، به طراحی واحد کنترل کننده خواهیم پرداخت. یک سازمان کامپیوتر شامل 4 قسمت اصلی می‌باشد:

1) واحد محاسباتی منطقی
2) واحد کنترل کننده
3) واسط‌های ارتباطی
4) ثبات ها

3 قست مهم از کامپیوتر پایه در گذشته معرفی و پیاده‌سازی شده است. در این فصل به واحد کنترل کننده که شامل ساختار سیگنال‌های کنترلی و زمان‌بندی‌های موجود در سیستم است، پرداخته می‌شود.

به منظور طراحی قسمت کنترلی دانستن چند مطلب مورد نیاز است:

1) یکی از ثباتهای کارا در سازمان کامپیوتر، ثبات دستورالعمل است. همچنانکه از معماری کامپیوتر به خاطر دارید یک دستورالعمل در چهارمرحله متوالی انجام می‌شود. آن مراحل عبارتند از:

1- Fetch
2- Decode
3- Indirect
4- Execute

Control Unit [14]

به طور خلاصه، کد باینری هر دستورالعمل، در مرحله Fetch، از حافظه خوانده و وارد ثبات دستورالعمل می‌شود. در مرحله Decode ، این کد توسط واحد کنترل کننده ارزیابی می شود و سیگنال‌های کنترلی لازم برای انجام دستورالعمل ساخته خواهند شد. مرحله سوم در بعضی دستورالعمل‌های خاص که شامل آدرس دهی غیر مستقیم هستند، در یک سیکل زمانی اجرا می‌شود. درنهایت مرحله چهارم عمل منطقی یا ریاضی، ورودی و خروجی، انتقالی یا ثباتی مورد نظر دستورالعمل را انجام می‌دهد:

کدهای زیر به طور خلاصه دستورالعمل ADD در کامپیوتر پایه را نشان می‌دهند:

T_0 : $AR \leftarrow PC$ } Fetch

T_1 : $IR \leftarrow M[AR], PC \leftarrow PC+1$

T_2 : $D_0 \ldots D_7 \leftarrow Decode[IR(12\text{-}14)], AR \leftarrow IR(0\text{-}11)$ } Decode

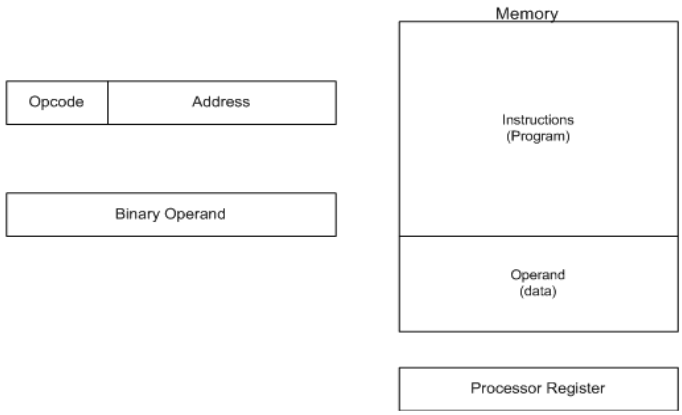
T_3 : $AR \leftarrow M[AR]$ Indirect

T_4 : $DR \leftarrow M[AR]$ } Execute

T_5 : $AC \leftarrow AC+DR, E \leftarrow Cout, SC \leftarrow 0$

2) کامپیوتر پایه‌ای که باید در این ترم طراحی شود، مطابق با کتاب معماری کامپیوتر مانو می‌باشد.

شکل زیر به طور خلاصه سازمان ذخیره برنامه در کامپیوتر را نمایش می‌دهد.



شکل 1-12-4 سازمان ذخیره برنامه در کامپیوتر

همچنانکه از معماری کامپیوتر به یاد دارید، ساختار دستورالعمل در یک ثبات، شامل Address و کد دستورالعمل می‌باشد. کد باینری دستورالعمل درفضائی از حافظه ذخیره شده است.

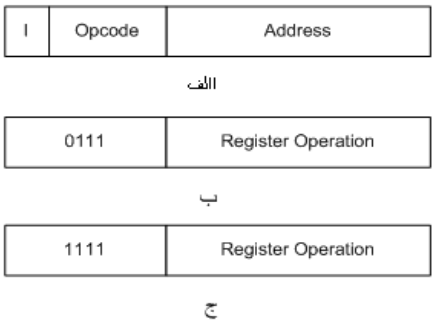
برای انجام هر دستورالعمل کد مربوطه از حافظه خوانده و به ثبات دستورالعمل وارد می‌شود. سیستم بعد از مشخص نمودن نوع Operation درصورت نیاز به عملوند[15] ، آن را از حافظه، بنا به آدرسی که در ساختار دستورالعمل موجود می‌باشد، می‌خواند و با ثباتهای پردازنده اجرا می‌نماید.

3) به طور کلی سه نوع فرمت برای کامپیوتر پایه در نظر گرفته‌ایم:

a) دستورالعمل‌هایی که براساس حافظه می‌باشند.

b) دستورالعمل‌هایی که براساس ثبات می‌باشند.

c) دستورالعمل‌هایی که براساس ورودی و خروجی می‌باشند.



شکل 2-12-4 الف) دستورالعمل‌های براساس حافظه ب) براساس ثبات ج) براساس ورودی و خروجی

اکنون ما با معرفی تعدادی دستورالعمل که در جدول 1-12-4 آمده است، به طراحی واحد کنترل می‌پردازیم:

Symbol	I=0	I=1	Description
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	ADD memory word to AC
LDA	2xxx	Axxx	LOAD memory word to AC
STA	3xxx	Bxxx	Store AC in memory
BUN	4xxx	Cxxx	Branch unconditionaly
BSA	5xxx	Dxxx	Branch and save return address

[15] Operand

CLA	7800	Clear AC
CLE	7400	Clear E bit
CMA	7200	Complement AC
CME	7100	Complement E bit
CIR	7080	Circulate right AC and E bit
CIL	7040	Circulate Left AC and E bit
INC	7020	Increment AC
SPA	7010	Skip next instruction if AC is positive
SNA	7008	Skip next instruction if AC is negative
SZA	7004	Skip next instruction if AC is zero
SZE	7002	Skip next instruction if E bit is zero
HLT	7001	Halt Computer
INP	F800	Input character to AC
OUT	F400	Output character from AC
SKI	F200	Skip on input flag
SKO	F100	Skip on output flag
ION	F080	Intrupt on
IOF	F040	Intrupt off

جدول 1-12-4 دستورالعمل‌های کامپیوتر پایه

با بیان مقدمه‌های فوق اکنون به معرفی ساختار کنترل کننده می‌پردازیم.

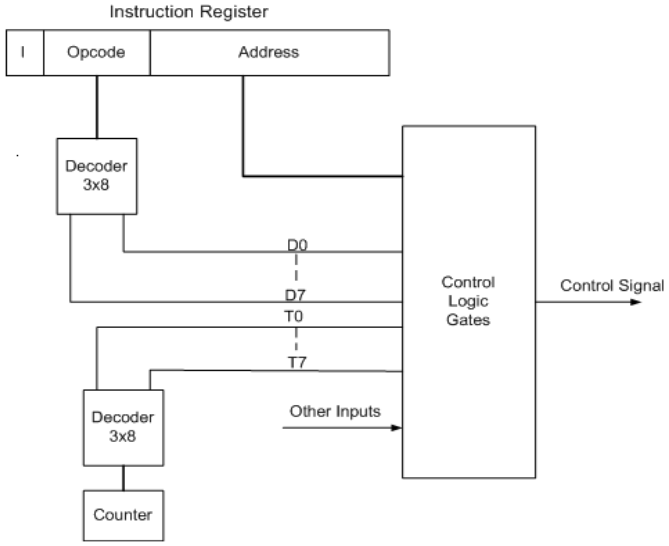
شکل 3-12-4 واحد کنترل کننده کامپیوتر پایه را نشان می‌دهد. این واحد از دو قسمت بسیار مهم تشکیل شده است:

1- واحد زمان‌بندی

2- مدارات ترکیبی

کامپیوتر پایه در هر سیکل زمانی تعدادی از ریزدستورالعمل‌ها را بنا به ساختار ALU، BUS و ثباتها انجام می‌دهد. در نتیجه باید یک واحد زمان‌بندی برای کنترل سیکل‌های اجرایی یک دستورالعمل از ابتدا تا انتها وجود داشته باشد. در کامپیوتری که باید در آزمایشگاه طراحی شود، ماکزیمم سیکل‌های هر دستورالعمل 8 پریود می‌باشد.

مدارات ترکیبی نیز با استفاده از بیت‌های ثبات دستورالعمل ساخته خواهند شد.

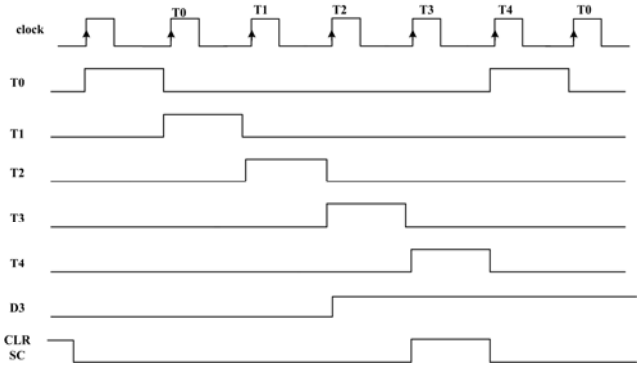


شکل 3-12-4 واحد کنترل کننده کامپیوتر پایه

تکالیف پیش از آزمایش

1) ابتدا کلیه ریز دستورالعمل‌های یک دستورالعمل، به طور کامل، بنا به سیکل‌های اجرایی آن باید نوشته شوند.

2) با استفاده از یک شمارنده و یک دیکدر باید سیگنال‌های T_0 ، T_1 و ... مطابق با شکل زیر ساخته شود:



شکل 4-12-4 سیگنالهای زمانبندی واحد کنترل

3) بعد از نوشتن تمامی ریز دستورالعمل‌ها، طراحی شمارنده T_0 تا T_7 و دیکدر مشخص کننده نوع عملگرها با استفاده از بیت های ثبات دستورالعمل ، اکنون باید کلیه سیگنال‌های کنترلی ثباتها به صورت مدارات ترکیبی پیاده‌سازی گردند.

4) با طراحی کلیه سیگنال‌های کنترل، اکنون باید واحد کنترل کننده را به صورت مجزا تست نمود. این کار با استفاده از ورودی دادن به ثبات دستورالعمل صورت می گیرد.

نکته مهم : واحد زمان‌بندی در ساختن بسیاری از سیگنال‌های کنترلی نقش بسیار مهمی را ایفا می‌نماید. به طور مثال در مرحله Fetch ، T_0 و T_1 از جمله سیگنال‌های کنترلی برای ساختن سیستم‌های مشخصه BUS می‌باشند.

5) مرحله آخر ترکیب نمودن BUS و ALU با واحد کنترلی می‌باشد. در این قسمت نیز با استفاده از تحلیل‌گر ALTERA یا XILINX به راحتی تست‌های مختلفی نظیر :

T_0: $AR \leftarrow PC$

را انجام خواهیم داد. هدف از این تست‌ها، آزمایش سیگنال‌های کنترل، انتقال داده از طریق BUS و انجام توابع ریاضی و منطقی بصورت غیرهمزمان می‌باشد. بنابراین شما به راحتی می‌توانند به جای حافظه فقط یک ثبات تعریف نمایید.

تکالیف داخل آزمایشگاه

در این آزمایش شما باید در Simulator، یک محیط تست برای برنامه نهایی حاصل از ترکیب BUS و ALU با واحد کنترلی ایجاد و آن را روی برد پیاده سازی نمایید. با استفاده از سوییچ های موجود بر روی برد و دستورالعمل‌های جدول 1-12-4 ورودی ثبات دستورالعمل را تعیین و نتایج حاصل را روی نمایشگرهای هفت قسمتی مشاهده نمایید.

آزمایش یازدهم

کامپیوتر مبنا

هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ طراحی و پیاده سازی کامپیوتر پایه
- ✓ تست کامپیوتر پایه با زبان ماشین

تئوری آزمایش

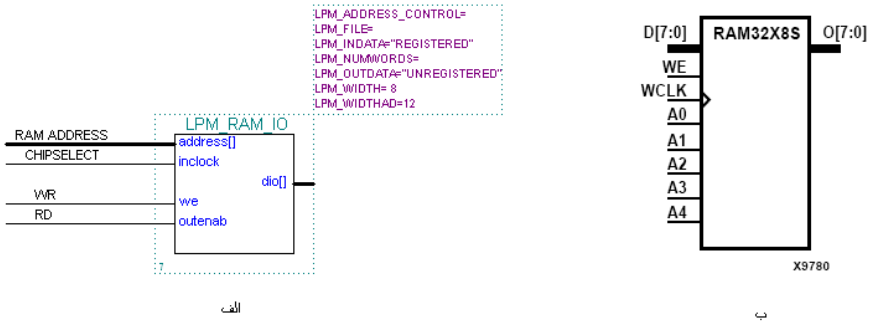
در آزمایش‌های گذشته، با طراحی کلیه بلوک‌های کامپیوتر پایه به طور مجزا آشنا شده‌اید. هدف کلی این آزمایشگاه و به طور مخصوص این آزمایش، طراحی یک CPU با امکانات بسیار اولیه و ساده می‌باشد. همچنان که در آزمایش‌های گذشته ذکر شد؛ یک سیستم کامپیوتر پایه از 4 قسمت Control unit , ALU, BUS و ثباتها تشکیل شده است. تمامی بلوک‌های مهم در آزمایش‌های قبل پیاده‌سازی گردیده‌اند. در این آزمایش باید همه بلوک‌ها و قسمت حافظه به صورت یک مدار مجتمع بر روی FPGA پیاده‌سازی گردند. برای انجام این عمل سه مقدمه مورد نیاز است:

1) طراحی حافظه در FPGA های XILINX یا ALTERA.
2) معرفی زبان ماشین برای سهولت در طراحی و پیاده‌سازی برنامه‌ها توسط یک کاربر.
3) نحوه ساختن زبان ماشین برای کدهای اسمبلی از طریق یک فایل exe.*

1- طراحی حافظه

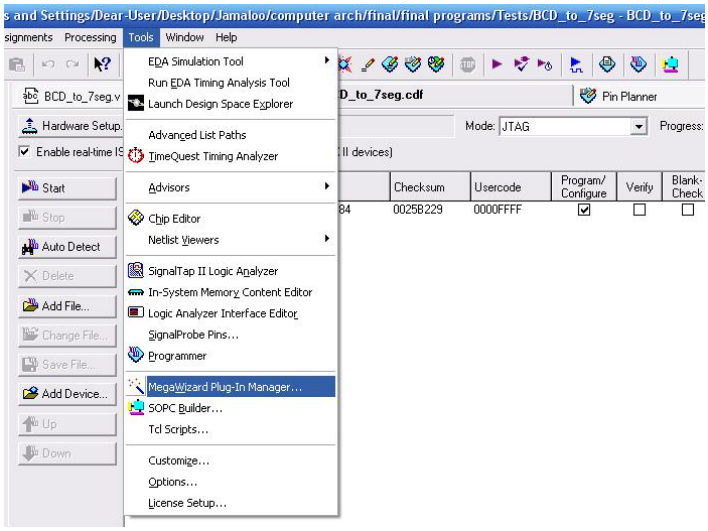
بعد از یکپارچه کردن قسمت‌های مختلف کامپیوتر پایه، اکنون باید حافظه را که متن کلیه برنامه‌ها و داده‌ها داخل آن قرار دارد به سیستم اضافه نمود. با استفاده از IPcore های موجود در نرم‌افزارهای ISE یا Quartus یا MAX این عمل انجام می‌شود. شرکت ALTERA با معرفی یک‌سری قطعات عمومی تحت عنوان LPM[16] امکان توسعه سخت‌افزار را تا حدود بسیار زیادی به کاربر می‌دهد. از جمله در زمینه تعریف حافظه ها LPM_RAM_IO است که به راحتی در داخل فایل گرافیکی قابل استفاده می‌باشد. در نرم‌افزار ISE شرکت XILINX نیز نظیر چنین قطعه‌ای به عنوان RAM32X8S در قسمت Memory Categuries موجود می‌باشد. شکل 1-4-13 نمونه‌هایی از بلوکهای حافظهٔ مربوط به هر دو شرکت را به نمایش گذاشته است.

[16] Library of Parameterized Modules



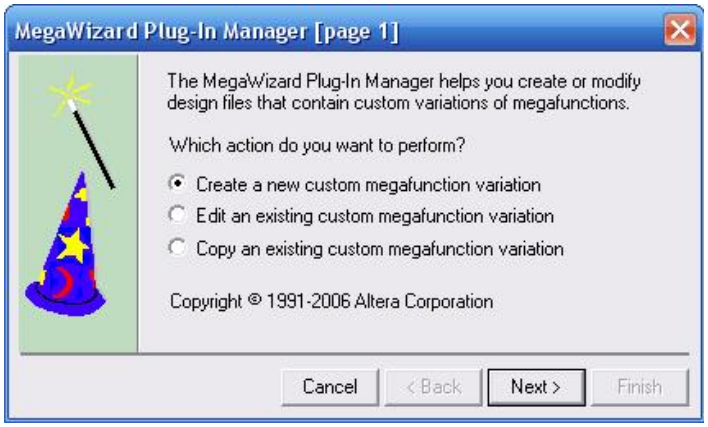
شکل 1-13-4 بلوک تعریف حافظه ها در نرم‌افزارهای الف) MUX ب) ISE

برای طراحی RAM از منوی Tools گزینه MegaWizard Plugin Manager را انتخاب می کنیم (شکل 2-13-4).



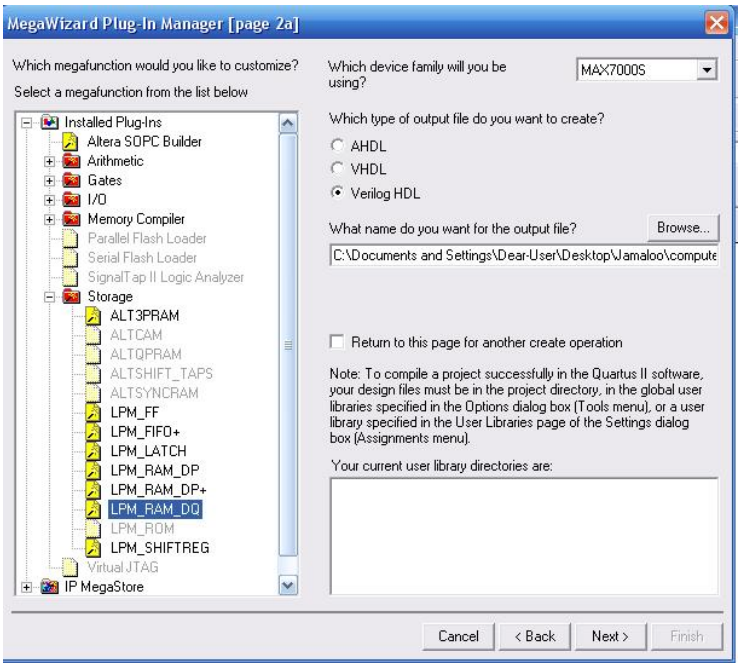
شکل 2-13-4

در پنجره شکل 3-13-4 گزینه اول را انتخاب می کنیم.



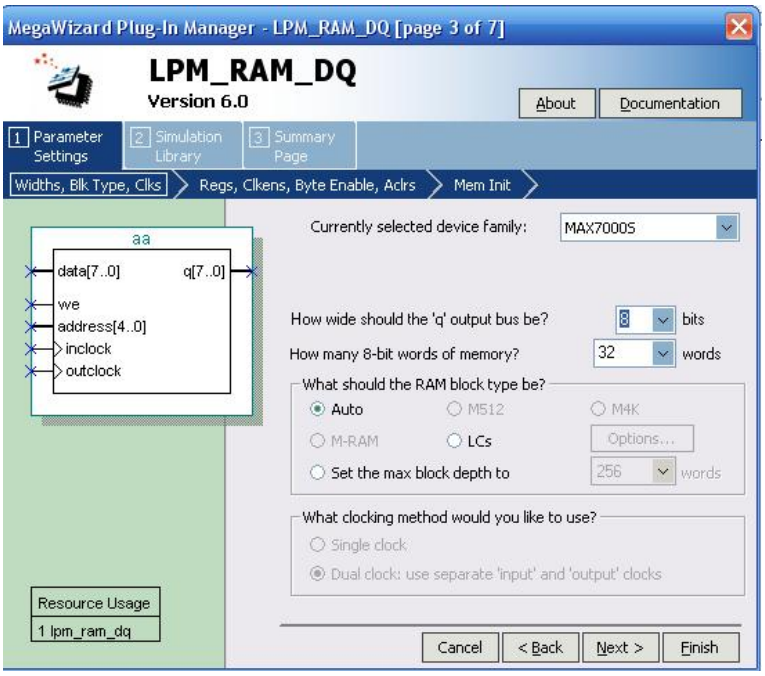
شکل 4-13-3

در پنجره شکل 4-13-4 خانواده CPLD یا FPGA را تعیین می کنیم . همچنین نامی را که می خواهیم برنامه با آن ذخیره شود و مسیر آن را تعیین می کنیم. از منوی Storage ، گزینه LPM_RAM_DQ را انتخاب می کنیم.



شکل 4-13-4

در مرحله بعد، تعداد بیت های آدرس و داده را تعیین و گزینه Finish را انتخاب می کنیم.



شکل 5-13-4

پس از طراحی ، RAM ابتدا باید به وسیله ارتباط سری در اختیار کاربر قرار گیرد تا برنامه مورد نظرش را روی آن برنامه ریزی نماید و سپس برای اجرای برنامه باید در اختیار CPU قرار گیرد. برای انجام این عمل، RAM نیاز به یک بیت selector دارد. عمل را بدین ترتیب که، به عنوان مثال ابتدا selector را صفر می کنیم تا از طریق ارتباط سری کاربر برنامه اش را برنامه ریزی نماید و سپس آن را یک می نماییم تا برای اجرای برنامه در اختیار CPU قرار گیرد.

2- معرفی زبان ماشین

همچنان که می‌دانید، زبان ماشین در حقیقت کدهای باینری هستند که به عنوان دستورالعمل مورد استفاده قرار می‌گیرند. کاربران برای سهولت در انجام برنامه‌نویسی نمادهایی را، به عنوان کدهای اسمبلی به کار می برند . این نمادها در جدول 1-12-4 آزمایش دوازدهم معرفی گردید. اکنون شما باید با استفاده از آن نمادها برنامه‌ای بنویسد که دو عدد را از حافظه می‌خواند و سپس آن دو را با هم جمع و در نقطه‌ای دیگر از حافظه ذخیره می‌کند.

بعد از نوشتن برنامه باید هر خط را به کد باینری، که همان زبان ماشین است، تبدیل نمایید. این فایل باینری را به کامپیوتر پایه اضافه نموده و FPGA را برنامه‌ریزی کنید.

بعد از راه‌اندازی FPGA دوباره حافظه SRAM را بخوانید و ببینید که آیا نتایج حاصله درست بوده است یا خیر؟

به همین صورت تست‌های دیگر را می‌توانید در روی سیستم کامپیوتر پایه انجام دهید.

3- نحوه ساختن زبان ماشین

برای سهولت در انجام برنامه‌ها ، شرکت نصرشرق فایلی را به عنوان مبدل اسمبلی به زبان ماشین طراحی نموده است. شما با استفاده از آن زبان اسمبلی برنامه خود را به کد باینری تبدیل می‌نمایید.

تکالیف داخل آزمایشگاه

1- بعد از یکپارچه کردن کامپیوتر پایه و انجام تست اولیه، اکنون با استفاده از ورودی و خروجی، برنامه‌ای بنویسید که از Switch select های خارج FPGA داده‌ای را بخواند و بعد از جمع کردن با نقطه 100 حافظه آن را روی نمایشگر هفت قسمتی نمایش دهد.

2- در این آزمایش Switch select به عنوان ورودی و نمایشگر هفت قسمتی به عنوان خروجی می‌باشد.

3- باید مبدل باینری به نمایشگر هفت قسمتی را به سیستم کامپیوتر پایه اضافه نمایید.

4- بدون استفده از IPcore های موجود در نرم‌افزار برنامه ای برای حافظه بنویسید، در این حالت دوباره برنامه را اجرا نمایید.

آزمایش دوازدهم

کنترل ریزبرنامه ریزی[17]

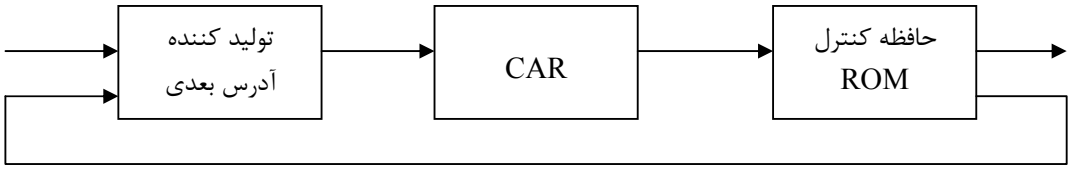
هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با طراحی واحد کنترل ریزبرنامه ریزی و ساختار آن
- ✓ طراحی و پیاده سازی میکروپروگرام

تئوری آزمایش

در آزمایشهای قبل با طراحی قسمت های مختلف کامپیوتر از جمله واحد کنترل آشنا شدید. واحد کنترل که در آزمایش های قبل طراحی نمودید ، کنترل سخت افزاری بود. راه دیگر برای طراحی واحد کنترل استفاده از کنترل ریزبرنامه ریزی می باشد. ریزبرنامه ریزی یک روش تقریباً نرم افزاری برای کنترل و اجرای ریز عملیات در کامپیوتر است. یکی از محاسن کنترل ریزبرنامه ریزی این است که اگر نیاز به تغییر ریز دستورات باشد، دیگر نیازی به تغییر سخت افزاری نیست ، بلکه کافیست دستورات جزئی در حافظه کنترل را تغییر دهیم. ساختار کلی یک واحد کنترل ریزبرنامه ریزی در شکل زیر نشان داده شده است.



شکل 1-14-4

تولید کننده آدرس بعدی همان طور که از نام آن مشخص است وظیفه تعیین آدرس بعدی را در کنترل ریزبرنامه ریزی بر عهده دارد. بطور خلاصه روشهای تولید آدرس در این بخش عبارتند از :

1- افزایش ثبات آدرس کنترل به میزان یک واحد.

2- اجرای انشعاب شرطی یا غیر شرطی.

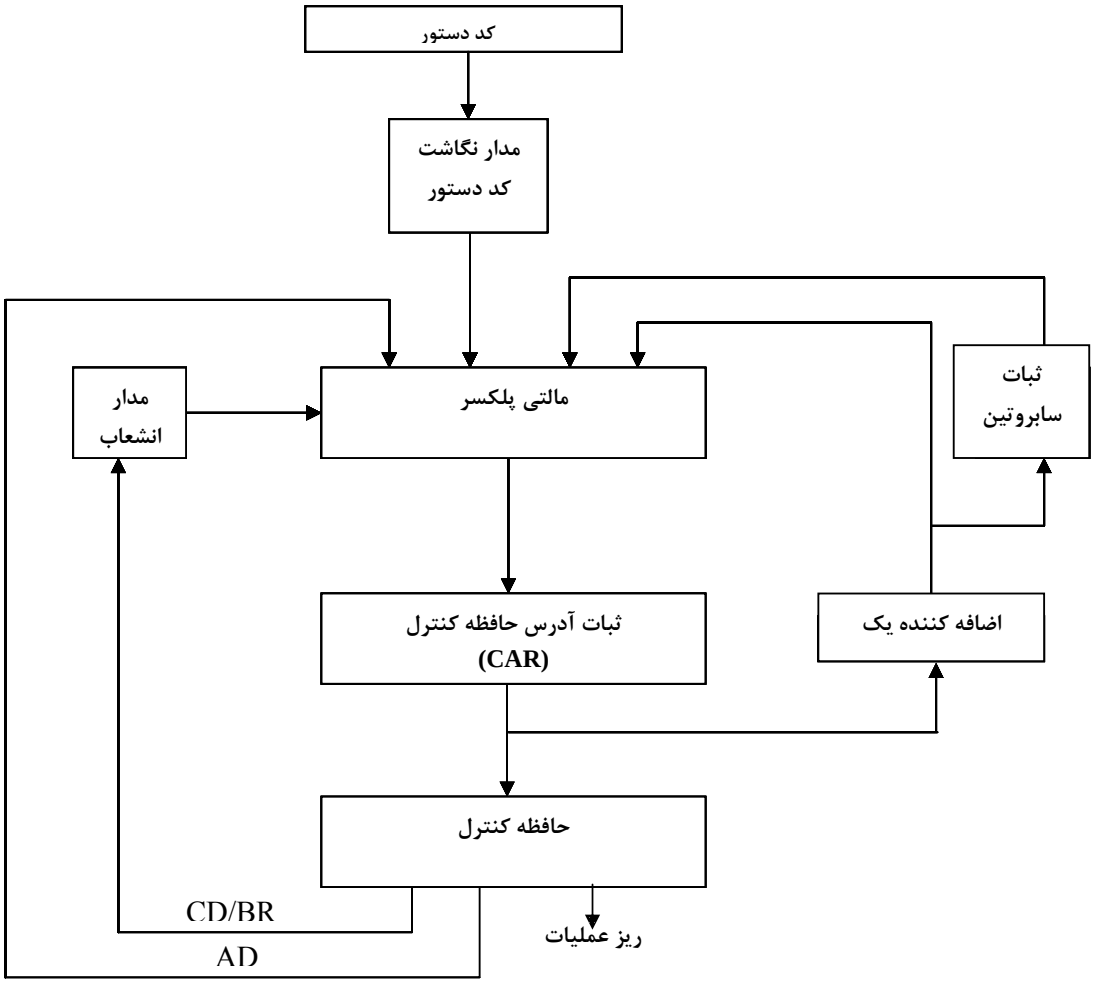
3- نگاشت بیت های دستور کامپیوتر به آدرس روتین نظیر در حافظه کنترل.

4- تسهیلاتی برای اجرای سابروتین و برگشت از سابروتین.

Micro Programming Control [17]

ثبات CAR[18] آدرسی را که از بخش تولید کننده آدرس بعدی می آید را به حافظه کنترل انتقال می دهد. میکروپروگرام از تعدادی ریز دستورات تشکیل می شود که در حافظه کنترل ذخیره می گردند. هر دستور کامپیوتر دارای روتین میکروپروگرام خود در حافظه کنترل می باشد ، که عملیات جزئی برای اجرای دستور مربوطه را تولید می کند. بنابراین برای اجرای هر دستور یا به عبارت دیگر برای هر کد دستور ، کامپیوتر می بایست روتین میکروپروگرام نظیر را در واحد کنترل اجرا نماید. تبدیل بیت های کد اجرای دستور کامپیوتر به آدرس شروع میکروپروگرام نظیر آن در حافظه کنترل، نگاشت کردن نام دارد.

در یکی دیگراز روشهای تولید آدرس می بایست امکان اجرای سابروتین و برگشت از آن فراهم باشد. سابروتین برنامه ای است که برای کار خاصی نوشته می شود و از هر کجای برنامه اصلی میکروپروگرام می تواند فراخوانی شود. بلوک دیاگرام انتخاب آدرس برای حافظه کنترل در شکل زیر نشان داده شده است.



شکل2-14-4 انتخاب آدرس برای حافظه کنترل

[18] Control Address Register

فرمت ریز دستورات حافظه کنترل به سه قسمت اساسی تقسیم می شود. قسمت ریز عملیات ، ریز عملیات کامپیوتر را مشخص می نماید. AD حافظه کنترل را آدرس دهی می کند و بالاخره CD/BR شرایط بیت های پرچم و نوع انشعاب را تعیین می کند.

برای طراحی واحد کنترل میکروپروگرام ابتدا باید یک ROM برای حافظه کنترل تعریف نمایید. بدین منظور از منوی Tools گزینه Megawizard Plugin Manager را انتخاب نمایید و همان طور که در بخش های قبل توضیح داده شد تنظیمات ROM را انجام دهید. تنها تفاوت موجود این است که در آزمایش قبل یک RAM طراحی کردید ، ولی این بار می بایست یک ROM تعریف نمایید. برای تعریف ROM می بایست محتویات آن را از قبل در یک فایل با پسوند mif. و یا hex. آماده نمایید تا در موقع تنظیمات ، آدرس آن فایل را به عنوان محتویات ROM بدهید.

تکالیف داخل آزمایشگاه

1) ابتدا کلیه ریز دستورالعمل‌های یک دستورالعمل به طور کامل بنا به سیکل‌های اجرایی آن باید نوشته شود.
2) بعد از نوشتن ریزدستورالعمل ها حافظه ROM با آنها باید پر شود.
3) سپس ریزعملیات خروجی از حافظه کنترل باید دیکد شود و گیت های منطقی برای ساختن سیگنالهای کنترلی طراحی شوند.

با توجه به مراحل گفته شده در بالا یک واحد کنترل ریز برنامه نویسی طراحی نمایید.

آزمایش سیزدهم

حافظه خارجی

هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با حافظه استاتیک
- ✓ بررسی نحوه ارتباط FPGA و حافظه خارجی

تئوری آزمایش

در برخی موارد، نیاز به استفاده از یک حافظه بزرگ می باشد، به طوریکه RAMهای داخلی FPGA نیز کافی نخواهند بود. استفاده از یک حافظه خارجی ، تنها راه حل ممکن می باشد. در RAM خارجی، مشابه انواع داخلی آن، ورودی های کنترلی write و enable ، هر دو عمل خواندن و نوشتن را انجام می دهند: وقتی we=0 باشد، مقدار دیتا باس در قسمت آدرس داده شده حافظه نوشته می شود. و هنگامی که we=1 و oe=0 باشد، محتویات آن قسمت حافظه، روی دیتا باس قرار خواهد گرفت. خط کنترلی Chip-Select نیز جهت فعال کردن خواندن از RAM یا نوشتن بر RAM به کار گرفته می شود. این باعث می شود که مقادیر در RAM بار شود و سپس FPGA را ، بدون نگرانی از invalid شدن RAM، برنامه‌ریزی کنید.

تکالیف داخل آزمایشگاه

برنامه کامپیوتر پایه ای را که در آزمایش 11 با استفاده از RAM داخلی نوشته اید، برای حافظه خارجی بازنویسی کنید و آن را تست نمایید.

جدول 1-15-4 اتصالات پین های FPGA و RAM استاتیک 128KB را نشان می دهد.

Pin Function	K6R1008V1C Pin
A0	24
A1	25
A2	26
A3	27
A4	38
A5	39
A6	40
A7	41
A8	63
A9	62
A10	61
A11	60
A12	58
A13	47
A14	46
A15	45
A16	44
I/O1	29
I/O2	30
I/O3	31
I/O4	36
I/O5	57
I/O6	56
I/O7	55
I/O8	54
/CS	28
/OE	53
/WE	37

جدول 1-15-4

آزمایش چهاردهم

پورت سریال

هدف

در این آزمایش اهداف زیر دنبال می شوند :

- ✓ آشنایی با پورت سریال
- ✓ بررسی نحوه ارتباط سریال FPGA از طریق پورت RS232 و تبادل اطلاعات بین این دو

تئوری آزمایش

ارسال و دریافت اطلاعات باینری به صورت بیت به بیت را انتقال سریال و پورت مورد استفاده برای این عمل را پورت سریال گویند. RS-232 یکی از استانداردهای پرکاربرد در کامپیوترهای شخصی و کاربردهای صنعتی است. این استاندارد هم ارتباط سریال سنکرون و هم آسنکرون را پشتیبانی کرده و به صورت Full Duplex عمل می نماید. کامپیوترهای شخصی تنها ارتباط آسنکرون را پشتیبانی می کنند و از طریق چیپUART موجود در برد اصلی، اطلاعات را از حالت موازی به سریال و یا بالعکس تبدیل کرده و با تنظیمات زمانی آن را از طریق پورت سریال ارسال یا دریافت می کند.

پورت سریال دارای یک کانکتور 9 پین می باشد و از آنجایی که این استاندارد در ابتدا برای طراحی با مودم طراحی شده بود، دارای پین های Handshaking و وضعیت می باشد. اما نوع خاصی از ارتباط با RS-232 به نام Null-Modem که تنها شامل پین-های ارسال و دریافت است، برای ارتباط با غیر از مودم استفاده می شود. بنابراین تنها دو پین Rx و Tx (و البته زمین) مورد نیاز است. در انتقال سریال قبل از ارسال هر کاراکتر یک بیت صفر (start bit) به معنی شروع و آمادگی و سپس 8 بیت اطلاعات و در آخر 1 یا 2 بیت (stop bit) به عنوان توقف یا پایان یک کاراکتر ارسال می شود.

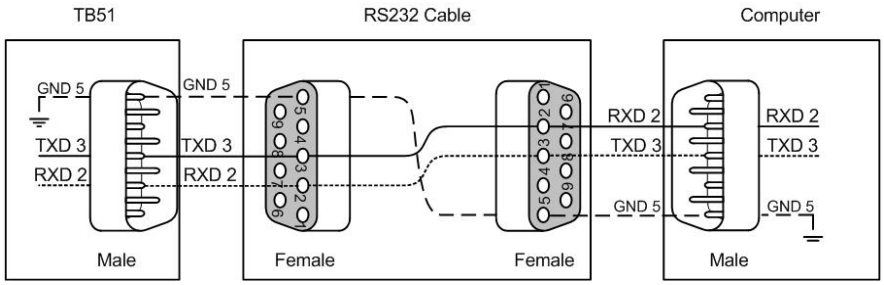
در استاندارد RS-232 سطح ولتاژ 3+ تا 12+ ولت نمایانگر وضعیت Space یا صفر منطقی و بازه 3- تا12- ولت نمایشگر وضعیت Mark یا یک منطقی می باشد. اگرچه تجهیزات استاندارد TTL با سطوح منطقی 0 و 5 ولت کا می کنند اما قالب اطلاعات ارسالی تفاوتی ندارد و با یک مدار تغییر سطح ولتاژ، PC می تواند با ادوات TTL ارتباط برقرار نماید. یکی از مبدل های سطح RS-232 به TTL مدار مجتمع MAX232 و یا HIN232 می باشد.

*** تذکر:**

شکل 1-16-4 کانکتور پورت سریال (**D9**)

PIN	Description
1	data carrier detect
*2	received data(RxD)
*3	transmitted data(TxD)
4	data terminal ready
*5	signal ground(GND)
6	data set ready
7	request to send
8	clear to send
9	ring indicator

جدول 1-16-4 عملکرد پین های D9



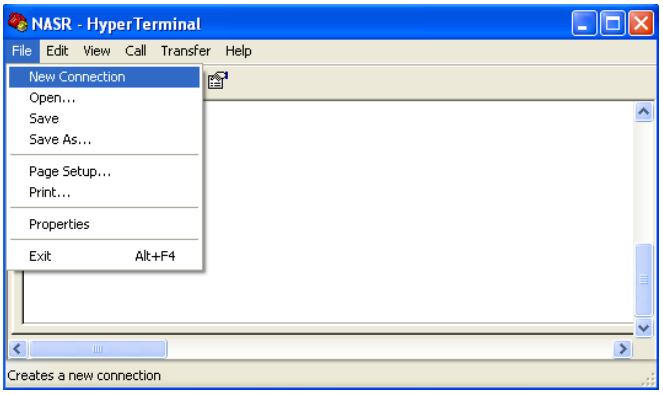
شکل 2-16-4 Null modem connection

تکالیف داخل آزمایشگاه

1) Hyperterminal **و ارتباط سریال از طریق** PC

Hyperterminal نرم افزاری همراه همه نسخه های سیستم عامل Windows است که می تواند به عنوان ترمینال ارتباط سریال استفاده شود. به این منظور ابتدا باید تنظیمات زیر انجام شوند:

ابتدا از پنجره File گزینه New Connection را انتخاب کنید.



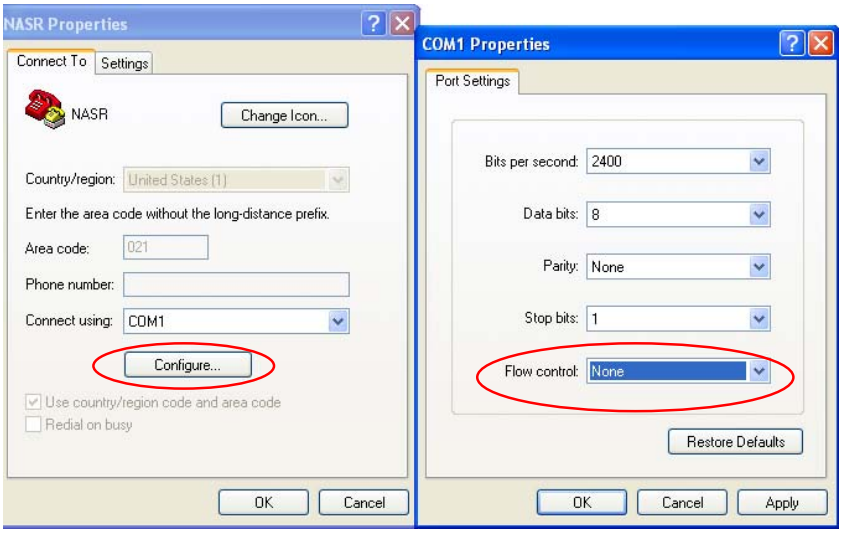
پس از انتخاب یک اسم، پنجره Connent to باز می شود.

برای تنظیمات New Connection ، وارد قسمتProperties \File شوید.

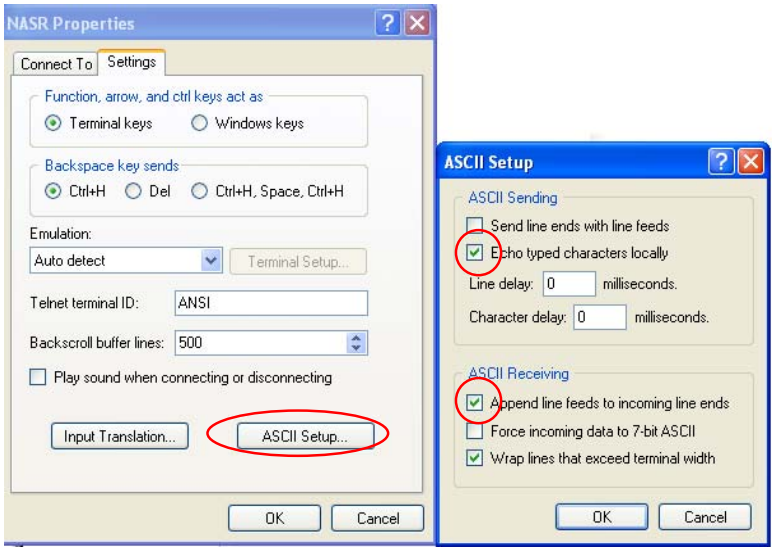
در قسمت Connect using گزینه Com 1 را انتخاب کنید. در پنجره COM Properties تنظیمات ارتباط سریال زیر را انجام دهید:

(ارتباط سریال با مشخصات 8 بیت بدون Parity Bit و یک Stop Bit و Buad Rate برابر 2400 bps)

Bits per sec = 2400, Data bits = 8, Parity = none, Stop bits = 1, Flow control = none



برای مشاهده کاراکترهایی که تایپ می کنید، ذیل پنجره File گزینه Properties و سپس Setting tab را انتخاب کنید. به قسمت ASCII Setup… بروید و گزینه Echo typed characters locally را فعال کنید.



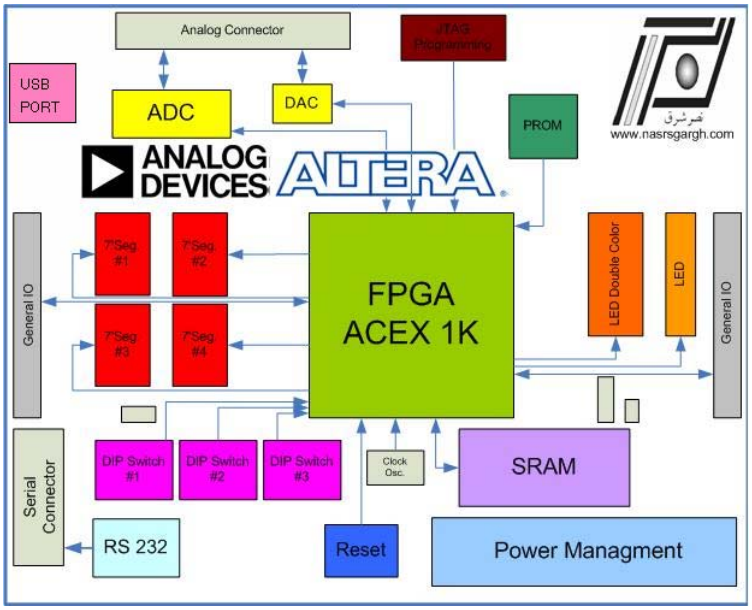
2) یک کاراکتر اسکی را به طور دائم از پورت سریالTxD ارسال کنید و آن را روی اسیلوسکوپ مشاهده کنید.BaudRate را 2400bps قرار دهید.

3) کد verilog مربوط به ارسال و دریافت دیتا را از طریق پورت سریال نوشته سپس از طریق آن یک کاراکتر اسکی را به طور مداوم به FPGA ارسال کنید. همچنین عملیات عکس آن یعنی ارسال از FPGA را نیز تست نمایید.

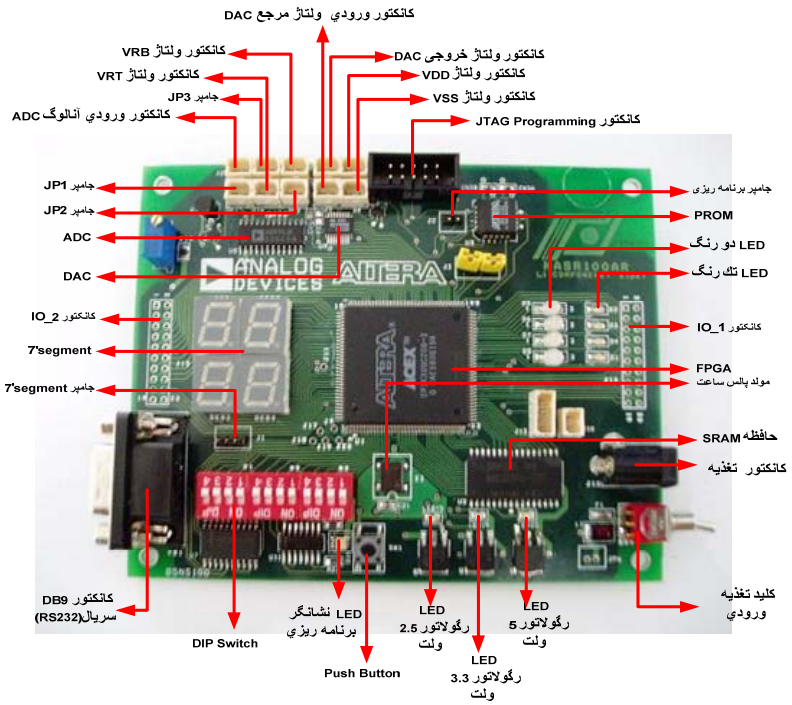
پیوست

آشنایی با برد آزمایشگاه FPGA

مجموعه آزمایشگاه FPGA شامل برد اصلی و پروگرامر شرکت ALTERA می باشد. شکل1- الف بلوک دیاگرام برد FPGA را نمایش می‌دهد. به طور کلی این برد شامل یک FPGA از شرکت ALTERA ، منابع تغذیه ، قسمت های مختلف آنالوگ و دیجیتال و کانکتورها می باشد . در این بخش به توضیح قسمت های مختلف خواهیم پرداخت.



شکل 1- الف- بلوک دیاگرام برد FPGA



شکل 1- ب - برد اصلی آزمایشگاه FPGA

تغذیه

تغذیه ورودی برد از طریق کانکتور J12 (کانکتور تغذیه) و کلید تغذیه ورودی تأمین می گردد . از طریق سه رگولاتور، ولتاژهای 5 ولت ، 3/3 ولت و 2/5 ولت ساخته می شود. LED های نشانگر ولتاژهای 5 ، 3/3 و 2/5 ولت در قسمت بالای رگولاتورها با شماره های LED18,19,20 برای نشان دادن وضعیت ولتاژهای سیستم به کار می روند . محدوده ولتاژ تغذیه ورودی از 7 تا 12 ولت بوده که بهتر است خروجی یک منبع تغذیه باشد.

تراشه FPGA

تراشه اصلی روی برد ACEX 1K است که یکی از FPGA های شرکت ALTERA می باشد. جدول 1 انواع تراشه های ACEX از نظر تعداد گیت ها ، بیت های حافظه و نوع بسته[۱۹] را نمایش می دهد . در روی این برد، FPGA با شکل PQFP است که دارای 208 پایه می باشد و بیشترین پایه ای که در اختیار کاربر قرار می دهد 171 عدد است . همچنانکه در جدول 1 مشاهده می نمایید ، بسته 208 پین دارای چگالی گیت های مختلفی می باشد . شرکت نصر شرق بنا به نوع سفارش و نوع کار آزمایشگاهی هر دانشگاه و

[19] package

درخواست آن ، از 30 k 50 k یا 100 k در نظر می گیرد . برای برنامه ریزی FPGA روی برد می توان از روش JTAG استفاده نمود. پروگرامر ALTERA از یک طرف به پورت موازی و از سمت دیگر به کانکتور J0 در روی برد متصل و با استفاده از نرم افزار Quartus II برنامه ریزی یا پیکربندی میگردد. اگر روی برد PROM (جهت ذخیره دائمی اطلاعات برنامه ریزی) وجود نداشت، باید jumper برنامه ریزی در حالت وصل قرار گیرد، در غیر این صورت این jamper در حالت قطع قرار می گیرد. EPC2 از PROM های شرکت ALTERA می باشد و برای برنامه ریزی این FPGA به کار می رود . این PROM دارای دو نوع بستهPLCC و DIP می باشد که در روی این برد از PLCC برای پیکربندی SRAM داخلی FPGA استفاده شده است .

مولد پالس ساعت

یک اسیلاتور با فرکانس 20MHz روی برد قرار گرفته که به یکی از پایه های FPGA متصل می باشد و می تواند به عنوان clock اصلی برد مورد استفاده قرار گیرد.

حافظه

یک حافظه 128 کیلو بیت شرکت SAMSUNG روی برد قرار دارد و باسهای آدرس، داده و سیگنالهای کنترلی آن به FPGA متصل شده است.

ارتباط سری RS-232

دو پایه از FPGA به عنوان خط ارسال و خط دریافت برای ارتباط با پورت COM کامپیوتر که از پروتکل RS-232 استفاده می کند در نظر گرفته شده است. این دو پایه از طریق تراشه MAX233 شرکت MAXIM به ولتاژهای RS-232 تبدیل شده و به کانکتور DB9 سری وصل می گردند.

سوئیچ های ورودی و RESET

یک Push Button برای سیگنالهای ورودی لحظه ای مانند Reset و . . . ، و سه عدد DIP Switch چهارتایی برای سیگنالهای ورودی setting و یا برای ارسال سه عدد BCD ، باینری و . . . روی برد در نظر گرفته شده که به یک سری از پایه های FPGA متصل هستند.

نمایشگر LED و هفت قسمتی

چهار LED تک رنگ و چهار LED دو رنگ برای نمایش سیگنالهای خروجی و نیز چهار عدد نمایشگر هفت قسمتی برای نمایش اعداد و . . . روی برد موجود است که به یک سری از پایه های FPGA متصل هستند. همچنین یک jamper برای نمایشگر های هفت قسمتی در نظر گرفته شده است تا در حالت آند-مشترک و یا کاتد مشترک (بسته به اینکه نمایشگرها چطور کار می کنند) قرار گیرد.

مبدل آنالوگ به دیجیتال

برای پردازش سیگنال آنالوگ توسط FPGA یک ADC شرکت ALTERA روی برد قرار گرفته است. این ADC هشت بیتی، با ولتاژ 5 ولت تغذیه شده و می تواند تا 20MSPS کارکند. باس داده و سیگنالهای کنترلی ADC به پایه های FPGA متصل شده است.

مبدل دیجیتال به آنالوگ

برای ارتباط با دنیای آنالوگ و ارائه نتیجه پردازش سیگنال آنالوگ به صورت خروجی، از یک DAC شرکت ALTERA استفاده شده است. این DAC هشت بیتی با تغذیه 2/5 تا 5 ولت کار می کند و باس داده و سیگنالهای کنترلی آن به پایه های FPGA متصل شده است.

کانکتورهای I/O (ورودی/خروجی)

برای ارتباط با خارج برد دو کانکتور ورودی-خروجی 20 پایه در نظر گرفته شده که به مابقی پایه های کاربری FPGA متصل شده است و می توان از آن برای ارتباط با اجزای جانبی مانند LCD و یا برد های دیگر مورد استفاده قرار داد. این دو کانکتور IO_1 و IO_2 می باشند .

کانکتور USB

برای ارتباط با پورت USB کامپیوتر کانکتور CN1 در نظر گرفته شده است که در قسمت سمت چپ برد واقع شده است.

FPGA Pin Number		FPGA Pin Number		FPGA Pin Number	
7	SW1-ain4	74	IO4	143	usbRD
8	SW1-ain3	75	IO12	144	usbD0
9	SW1-ain2	79	GCLK2-J10	147	usbD7
10	SW1-ain1	80	DED INPUT2	148	usbD3
11	SW2-ain8	81	GND-CKLK	149	7SEG2-DP
12	SW3-ain7	83	IO3	150	usbD4
13	SW2-ain6	85	IO11	157	7SEG1-g
14	SW2-ain5	86	IO2	158	7SEG1-a
15	SW3-ain12	87	IO10	159	7SEG2-g
16	SW3-ain11	88	usbWR	160	7SEG1-b
17	SW3-ain10	89	IO1	161	7SEG2-a
18	SW3-ain9	90	IO9	162	7SEG2-f
19	40MHZ	92	usbTXE	163	7SEG2-b
24	R1A0	93	D4-aled7	164	7SEG1-f
25	R1A1	94	D4-aled8	166	7SEG2-e
26	R1A2	95	usbRXF	167	7SEG2-c
27	R1A3	96	D3-alad5	168	7SEG2-d
28	R1CS0	97	D3-aled6	169	7SEG1-c
29	R1D0	99	usbPWREN	170	7SEG1-DP
30	R1D1	100	D2-aled3	172	7SEG3-a
31	R1D2	101	D3-aled4.	173	7SEG4-a
36	R1D3	102	D1-aled1	174	7SEG4-g
37	R1WE	103	D1-aled2	175	7SEG4-b
38	R1A4	104	DAC-D2	176	7SEG4-f
39	R1A5	107	Msel1	177	7SEG4-e
40	R1A6	108	Msel0	179	7SEG4-c
41	R1A7	111	DAC-D1	180	7SEG4-d
44	R1A16	112	DAC_D0	182	DED INPUT4
45	R1A15	113	DAC-CS	183	GCLK1-J11
46	R1A14	114	DAC-RW	184	DED INPUT3
47	R1A13	115	DAC-D3	186	7SEG1-e
52	nSTATUS	116	DAC-D4	187	7SEG1-d
53	R1OE	119	DAC-D5	189	7SEG3-c
54	R1D7	120	DAC-D6	190	7SEG3-b
55	R1D6	121	DAC-D7	191	7SEG3-g
56	R1D5	122	ADC-CLK	192	7SEG3-f
57	R1D4	125	ADC-D7	193	7SEG3-e
58	R1A12	126	ADC-D6	195	7SEG3-d
60	R1A11	127	ADC-D5	196	7SEG3-DP
61	R1A10	128	ADC-D4	197	usbD1
62	R1A9	131	ADC-D3	198	7SEG4-DP
63	R1A8	132	ADC-D2	199	usbD2
64	IO8	133	ADC-D1	200	IO29ii-j17
65	IO16	134	ADC-D0	202	TX0
67	IO7	135	ADC-OE	203	RX0
68	IO15	136	usbRSTn	204	RST*
69	IO6	139	usbD5	205	ain13-J18
70	IO14	140	usbSIWU	206	ain14-J19
71	IO5	141	usbD6	207	ain15-J20
73	IO13	142	usbXTIN	208	CLKIN-J9