بسمه تعالی

آزمایشگاه معماری کامپیوتر

[طراحی، پیاده‌سازی و برنامه نویسی یک پردازنده چهار بیتی]

دانشکده فنی مهندسی دانشگاه زنجان

گروه مهندسی کامپیوتر

تالیف: رضا امیدی

بهار ۱۳۹۴

تاریخچه سند

نسخه	تاریخ	توضیحات
اولیه	اردیبهشت ماه ۱۳۹۴	--

مقدمه

در اینآزمایشگاه جهت آشنایی با معماری داخلی کامپیوتر (ریزپردازنده) پیاده‌سازی یکپردازنده چهار بیتی مد نظر قرار گرفته است. در این طرح سعی شده است حداقل سخت افزار لازم جهت آشنایی با عملکرد اجزای مختلف یکریزپردازنده شامل واحد محاسبه و منطق، شیف رجیسترها، عملگرهای منطقی، حافظه های داخلی، باس مشترک و غیره وجود داشته باشد. به طوری که در طرح ارائه شده امکان پیاده سازی انواع دستورالعمل وجود داشته باشد و از طرفی از لحاظ سیم بندی دارای حجم متناسب باشد.

به طور کلی اهدافی که از این آزمایشگاه به دنبال آن هستیم عبارتند از:

- آشنایی با افزاره‌های برنامه پذیر میدانی (FPGA[1]) در راستای پیاده‌سازی سخت‌افراز ریزپردازنده
- کار با بورد آزمایشگاهی Altera به صورت خاص (کار با برنامه Quartus-II، شبیه‌سازی و برنامه‌ریزی)
- آشنایی با نحوه پیاده‌سازی یک ریزپردازنده با ساختار مطلوب
- آشنایی عملی با مفاهیم اولیه در معماری کامپیوتر از جمله:
    - تحقق واحد محاسبه و منطق در یک ریزپردازنده
    - تحقق واحد کنترل به روش میکروپراگرام
    - مفهوم باس مشترک به طور خاص باس داده
    - حافظه برنامه‌ریزی جهت ذخیره برنامه کاربر
    - طراحی ریزدستورهای یک ریز پردازنده

[1] Field-programmable gate array

لیست مباحث

توصیه ها و تذکرات کلی

توصیه‌ها:

- دانشجویان عزیز در حفظ و نگهداری تجهیزات آزمایشگاهی نهایت اهتمام و همکاری را داشته باشند.
- قبل از شروع آزمایش از سالم بودن تجهیزات اطمینان حاصل کنید.
- پیش از اعمال ولتاژ به تراشه یا بورد آزمایشگاهی از مقدار ولتاژ مجاز برای آن (غالبا ۹ ولت یا استفاده از آداپتور) اطمینان حاصل کنید.
- صرفا از پروگرامر خاص FPGA(ByteBlaster) استفاده کنید.

تذکرات:

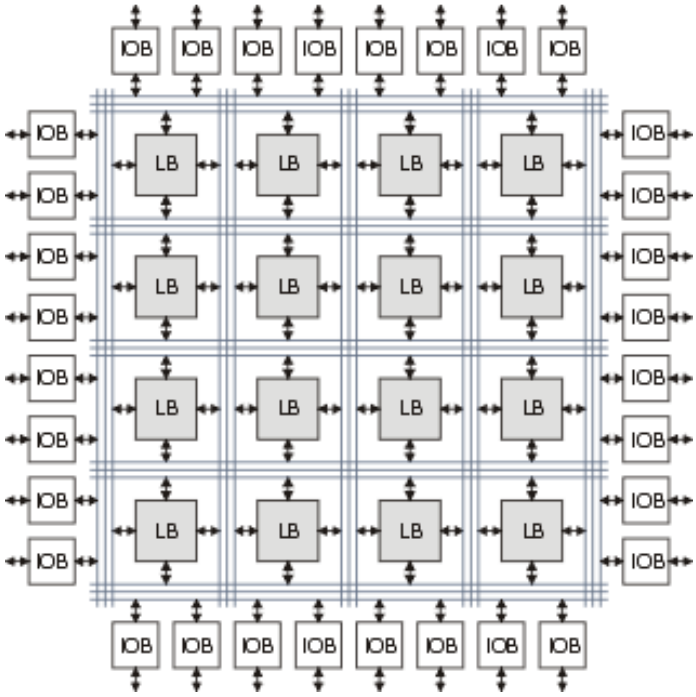
- حضور به موقع در آزمایشگاه الزامی است.
- رعایت نظم و مرتب کردن میز آزمایشگاه پس از اتمام آزمایش الزامی است.
- ارائه پیش گزارش قبل از انجام آزمایش الزامی است.
- **تهیه گزارش کار** حداکثر تا دو هفته پس از انجام آزمایش الزامی است.هرگروه دو یا سه نفره باید گزارشی شامل موارد خواسته شده در پیش گزارش و نتایج حاصل ازفعالیت‌هایآزمایشگاهی را به همراه شکل و نکات تجربی که هنگام کار با آن برخورد می‌کند و به تناسب هر آزمایشمطالب تکمیلی دیگر را حداکثر دو هفته پس از انجام آزمایش تحویل دهد.

جلسه اول: آشنایی با تجهیزات آزمایشگاه معماری کامپیوتر

بورد آزمایشگاهی موجود در آزمایشگاه معماری کامپیوتر در بردارنده افزاره برنامه‌پذیر میدانی Altera می‌باشد. این شرکت یکی از تولید کنندگان عمده افزاره‌های برنامه‌پذیر میدانی در دنیا می‌باشد. مشابه تمام شرکت‌های تولید کننده FPGA، این شرکت نیز برای برنامه نویسی و برنامه‌ریزی افزاره‌های خود نرم‌افزار خاصی به بازار ارائه داده است. نسخه قدیمی این نرم‌افزار Max+Plus می‌باشد که امروزه با نسخه به روزتر آن یعنی Quartus II جایگزین شده است. در این جلسه قصد داریم به معرفی محیط این نرم‌افزار بپردازیم.

سخت‌افزار آزمایشگاه

تا پیش از ساخت تراشه های قابل برنامه‌ریزی، پیاده‌سازی مدارهای منطقی بسیار سخت، وقت‌گیر، جاگیر وپرهزینه بود. ولی با ساخت تراشه‌های قابل برنامه‌ریزی، این مشکل‌ها تا حد زیادی حل شدند. افزاره‌های برنامه‌پذیریک تراشه قابل برنامه‌ریزی است که هر مدار منطقی‌ای را می‌توان با آن پیاده‌سازی نمود.بلوک دیاگرام یکFPGAبه طور ساده در شکل زیر نشان داده شده است. در این شکل IOBها معرف بلوک‌های ورودی خروجی است که در نهایت با دنیای بیرون تراشه ارتباط برقرار می‌کند. همچنین LBها بلوک‌های منطقی است که لاجیک مد نظر طراح، پس از سنتز در آنها نگاشت می-شود. حجم این بلوک‌های منطقی و بلوک‌های ورودی خروجی قابلیت‌های افزاره را نشان می‌دهد. این حجم به صورت معادل گیت مطرح می گردد، امروزه تراشه‌هایی از محدوده کیلو گیت تا میلیون گیت در بازار قابل دسترسی می‌باشد. شرکت‌های مختلفی در زمینه طراحی این تراشه‌ها فعالیت می‌کنند که مشهورترین آنها Xilinx، Actel و Altera می-باشد.



شکل ۱: ساختار داخلی FPGA

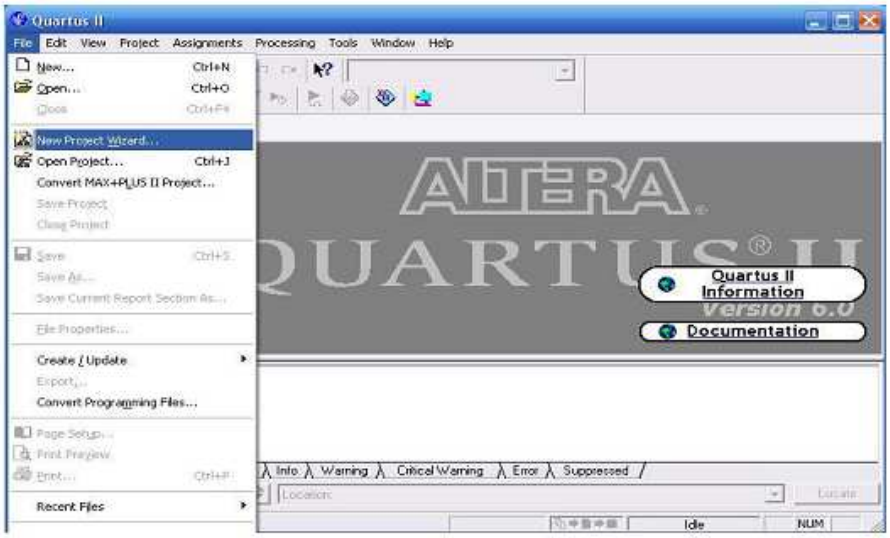
جدول ۱: مشخصات سخت‌افزار آزمایشگاه

پارامتر	
شرکت تولید کننده	Altera
بسته‌بندی	
تعداد ورودی خروجی	
گیت معادل	
نرم‌افزار	Quartus II

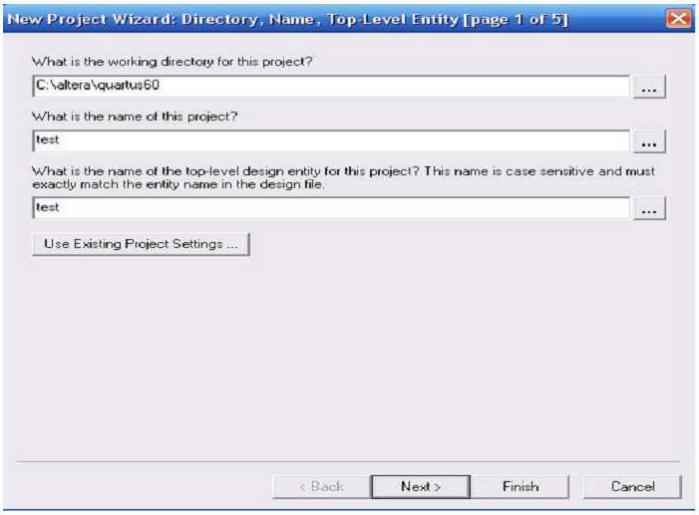
تحقیق کنید(اختیاری): ساختار داخلی FPGAهای متداول به چه صورت می‌باشد؟ پیاده‌سازی مدارهای منطقی در این ساختار به چه صورت انجام می‌شود؟

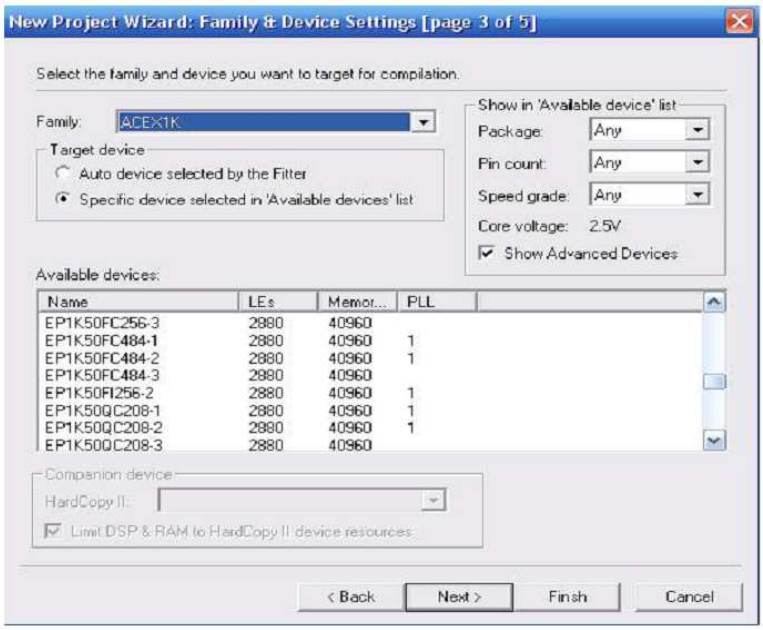
ایجاد پروژه جدید در محیط Quartus II:

برای شروع یک طراحی دیجیتال می‌بایست محیط نرم‌افزار مورد نظر را باز کرد. برای این منظور نرم افزار Quartusرا اجرا نمائید. در کلیه نرم افزارهای جدید برای شروع یک طراحی باید پروژهای را باز کرد. در این نرم افـزار نیـز بـا اسـتفاده از منوی Fileو انتخاب گزینه Wizard Project Newپنجره ای برای ایجاد پروژه جدید باز می شود.



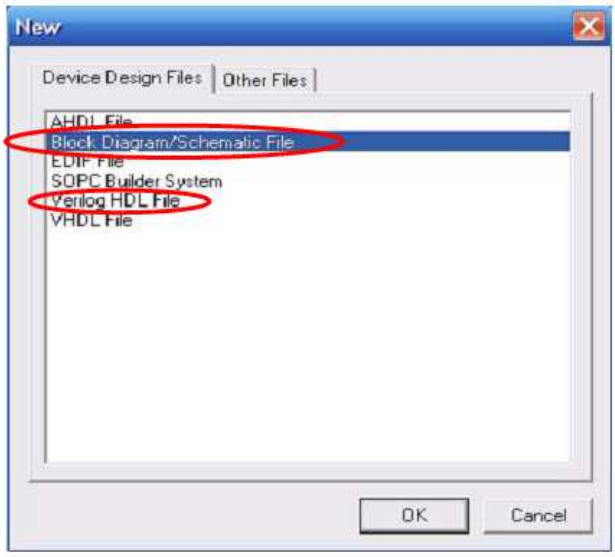
ودر پنجره شکل نام پروژه و مسیر ذخیره شدن آن مشخص می شود.در پنجره شکل زیر می بایسـت خـانواده CPLDیـا FPGA و شماره آن مشخص شود.



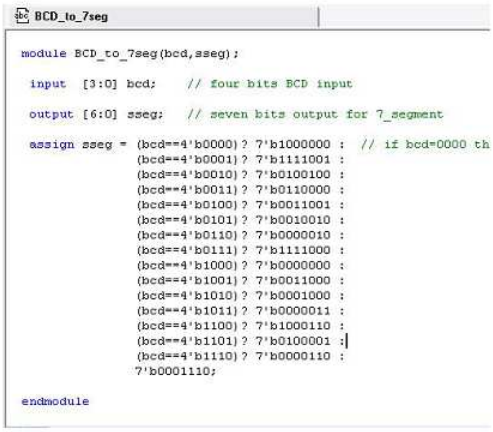


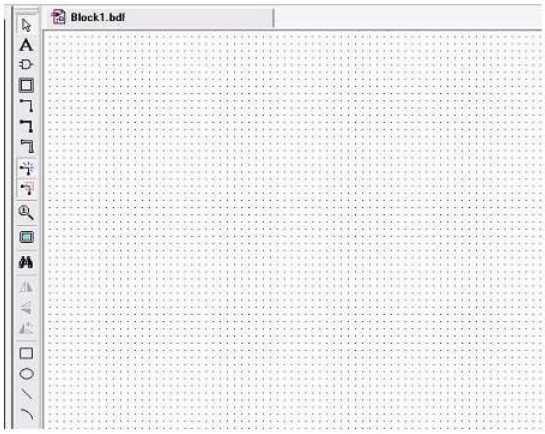
اکنون پروژه ای برای یک طراحی جدید در نظر گرفته شده است. برای شروع باید فایل طراحی در نرم افزار وارد گردد. دو نوع فایلطراحی وجود دارد: فایل شماتیک یا گرافیکی و فایل متنی (VHDL، Verilog).

در فایل‌های متنی برای طراحی از زبان‌های HDLنظیر Verilogو VHDLاستفاده می‌گردد. پسوند فایل‌هـای متنـی بـا زبان‌های مذکور به ترتیب *.vو *.vhdاست. شرکت ALTERAزبانی مخصوص به خود تحت عنوان AHDLنیز دارد کـه پسوند فایلهای آن *.ahdمی‌باشد. برای شروع طراحی از منوی Fileگزینه Newرا انتخاب کنید . پنجـره شـکل زیـر بـاز می‌شود. در این پنجره برای طراحی گرافیکی باید گزینه File Schematic/Diagram Blockو برای طراحی با کد Verilog گزینه File HDL Verilogانتخاب گردد.

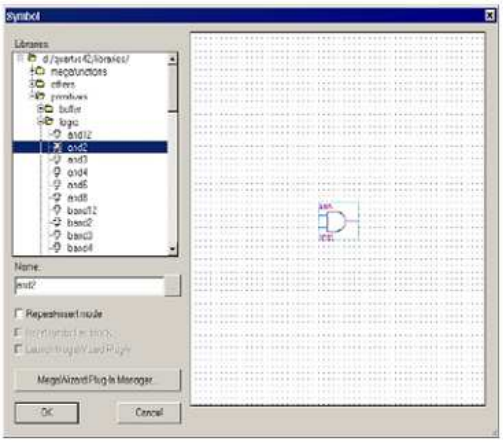


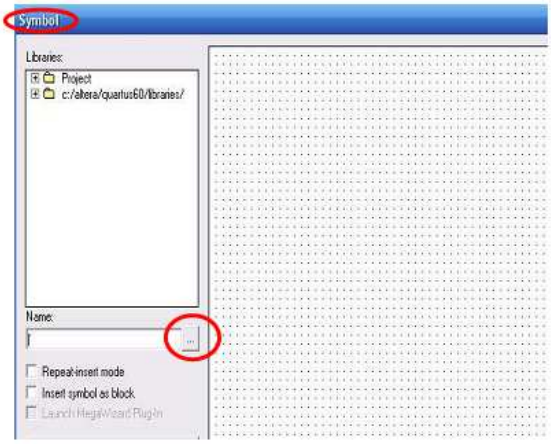
با انتخاب گزینه File Schematic/Diagram Blockپنجره طراحی گرافیکی و با انتخاب گزینه Verilog HDL Fileپنجره طراحی با کد Verilogباز می‌شود. از گزینه File > Save Asبرای ذخیره فایل و تعیین نوع فایل(*.bdfبرای طراحی گرافیکی و *.vبرای طراحی با کدVerilog)استفاده کنید. (Add File to Current Projectرا نیز کلیک نمائید).



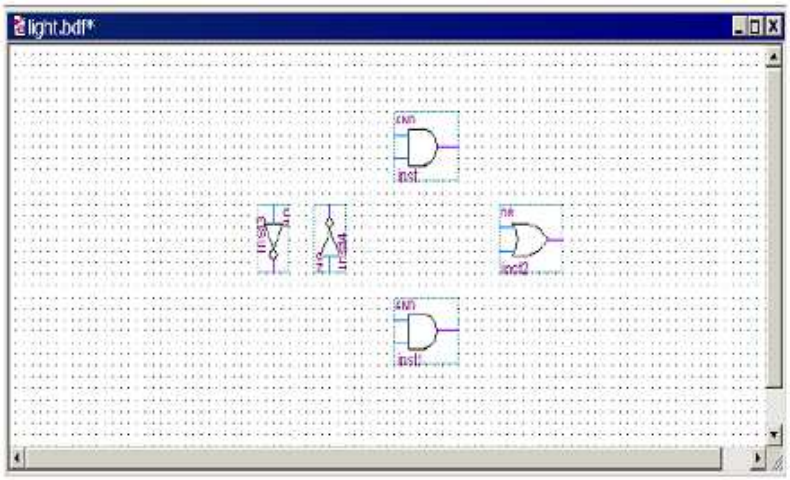


در طراحی گرافیکی برای استفاده از کتابخانه و اضافه کردن المان‌های منطقی،با دو بار کلیک بر روی فضای خالی صفحه، پنجره ای با نام Symbolباز می شود که قطعات مورد نظر را می توان از مسیر مشخص شده آورد. همچنین بوسیله آیکون  ،می‌توان المان مورد نظر را انتخاب نمود.

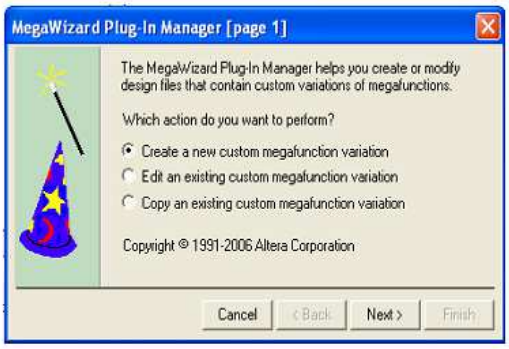




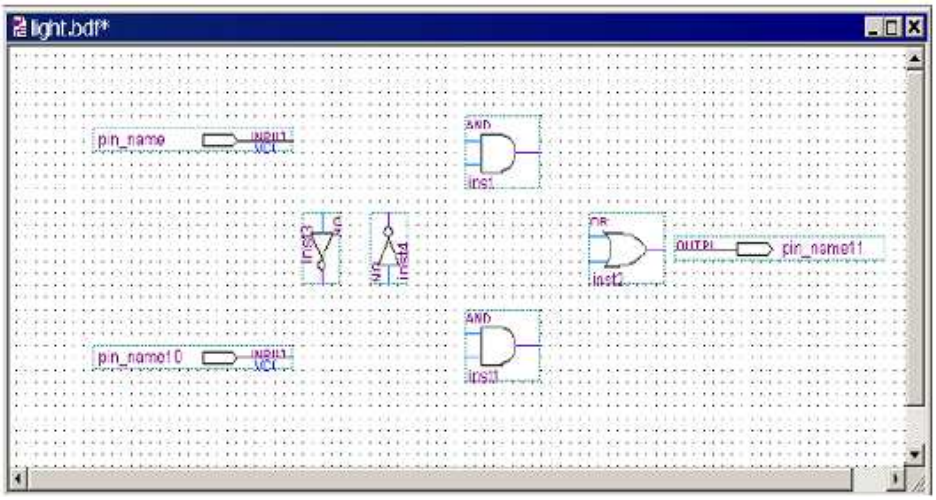
برای کپی یک المان که در محیط گرافیکی قرار دارد، روی آن راست-کلیک نموده و تا نقطه دیگری روی صفحه drag نمائید. با استفاده از آیکون  می‌توانید یک المان را ۹۰ درجه بچرخانید. دیگر المان‌ها را نیز با همین روش روی صفحه گرافیکی قرار دهید.



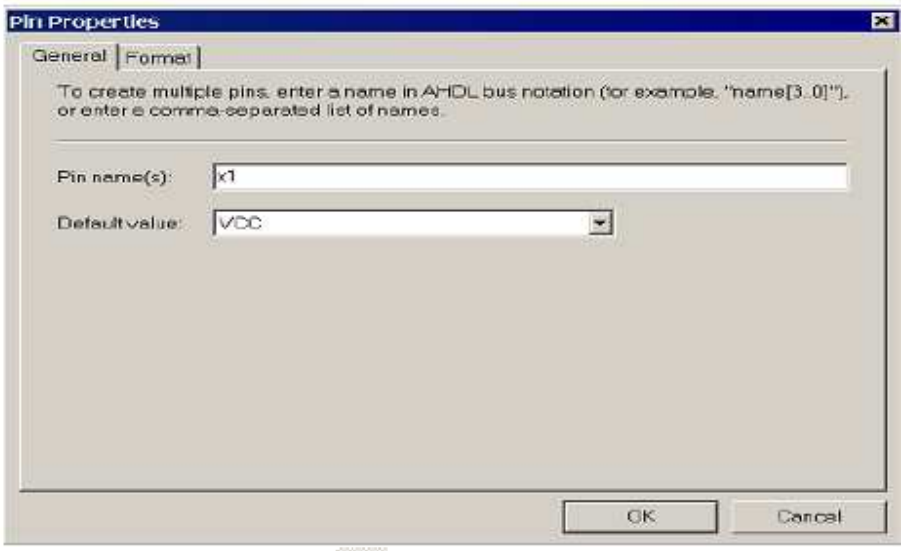
برای اضافه کردن بلوک‌های اساسی مانند ALU، شیفت‌دهنده، مالتی‌پلکسر، حافظه و ... می‌توان از منوی Tools گزینهMega Wizard Plug-In Managerاستفاده نمود.



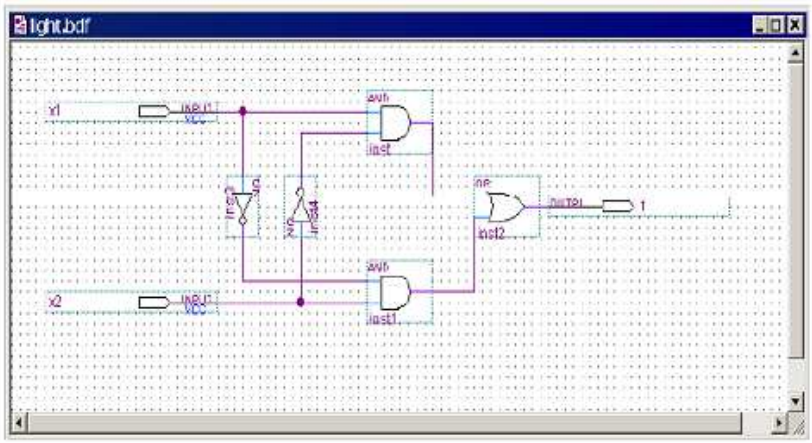
با استفاده از Wizardمی‌توان بلوک‌های محاسباتی مانند جمع‌کننده، ضرب‌کننده و ... یا حافظه‌ها و دیگر بلوک های مورد نیاز را که به صورت پارامتری موجود می‌باشد، تنظیم و طراحی نمود و آن را به طرح شماتیک خود اضافه کرد.برای اضافه کردن ورودی/خروجی از کتابخانه Primitiveگزینه Pinرا انتخاب کنید.



برای نامگذاری پایه‌های ورودی و خروجی، روی کلمه Pin Nameکلیک کنید. تا پنجره Pin Propertiesباز شود.



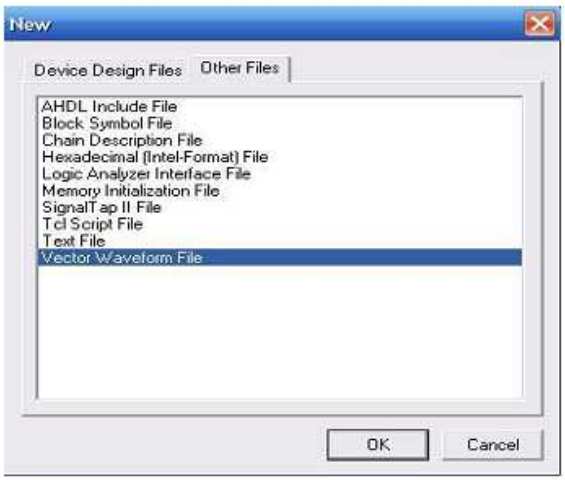
برای سیم‌بندی المان‌های موجود در محیط گرافیکی روی آیکون  کلیک کرده تا ابزار Node Orthogonalفعال شود. روی نقطه شروع کلیک کرده و با Dragکردن تا نقطه انتهایی، سیم‌بندی را انجام دهید.



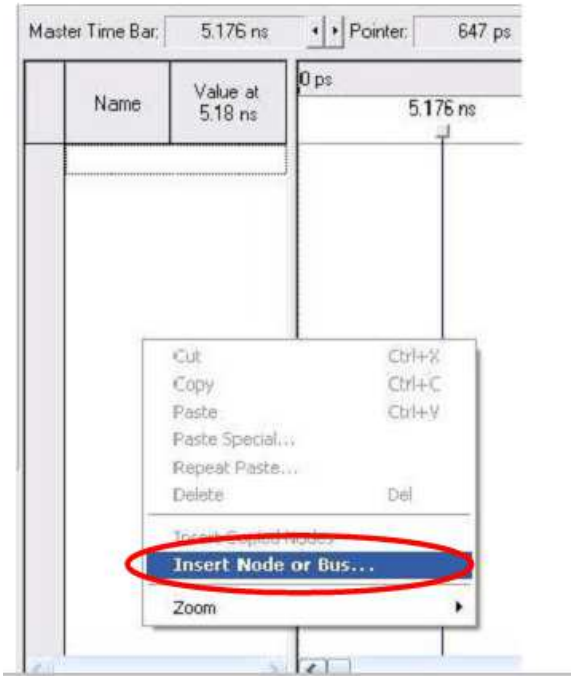
پس از طراحی کامل مدار،نوبت به Compileآن می رسد. اگر کلیه مراحل Compileمدار بدون هیچ خطایی به پایان برسد آنگاه مدار از لحاظ منطقی بی نقص می‌باشد. برای Compileکردن برنامه گزینه مشخص شده در شکل زیر انتخاب می‌شود:

شبیه سازی مدار طراحی شده:

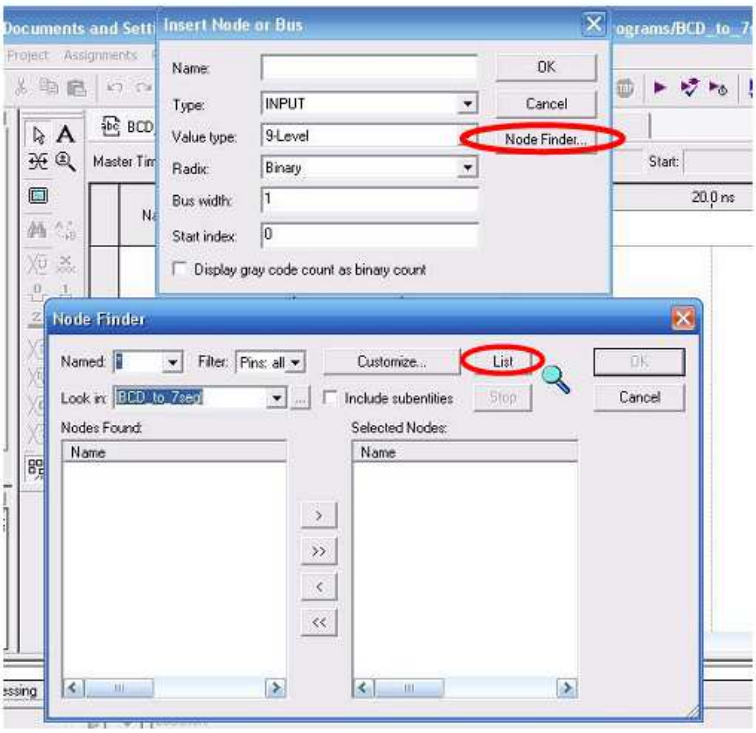
یکی از امکانات رایج نرم افزارها تحلیل نرم افزاری می باشد. تحلیل نرم افزاری کمک بسیار زیادی به تعیین میزان صحت عملکرد مدار قبل از پیاده سازی آن می‌نماید. برای تحلیل نرم‌افزاری باید فایل (VectorWaveformFile) * .vwfاز طریق گزینه Newدر منوی Fileباز گردد. این کار در یک پروژه بعد از Compileصحیح صورت می‌گیرد.



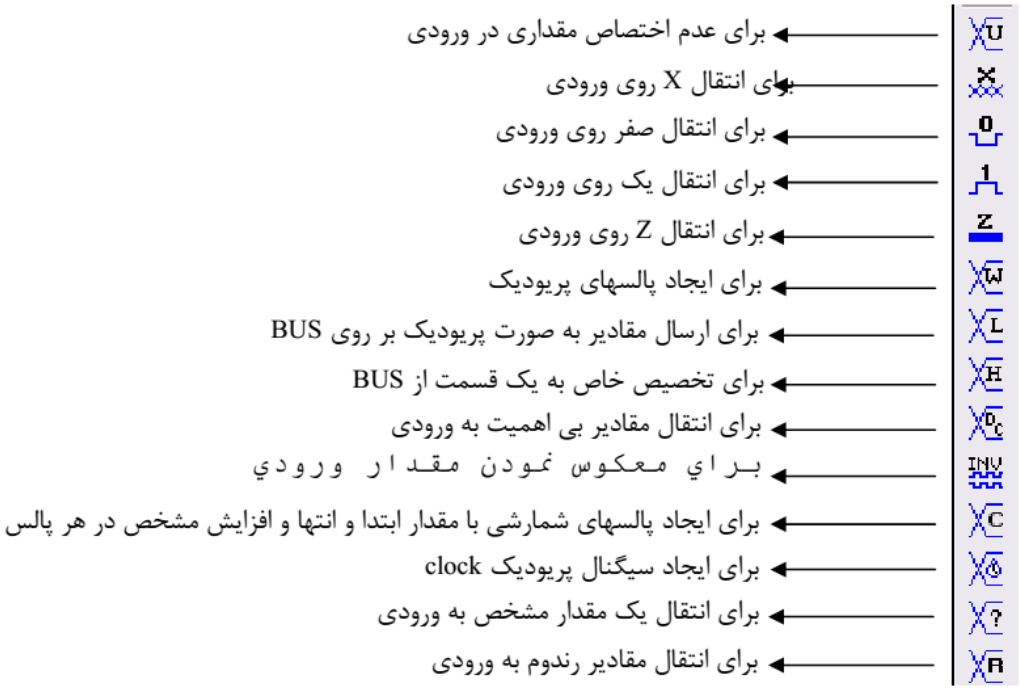
شکل زیر پنجره مربوط به تحلیل نرم افزاری را نمایش می‌دهد. ابتدا این صفحه باید با نام پروژه اصلی و با پسوند *.vwfذخیره شود.



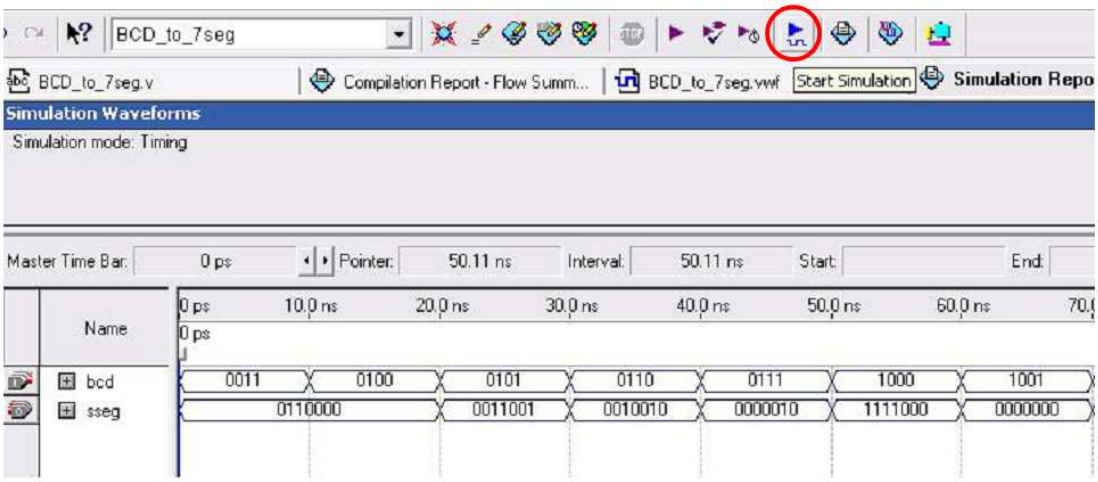
سپس با کلیک راست در صفحه و انتخاب گزینه Bus or Node Insertپنجره شکل زیر باز می شود:



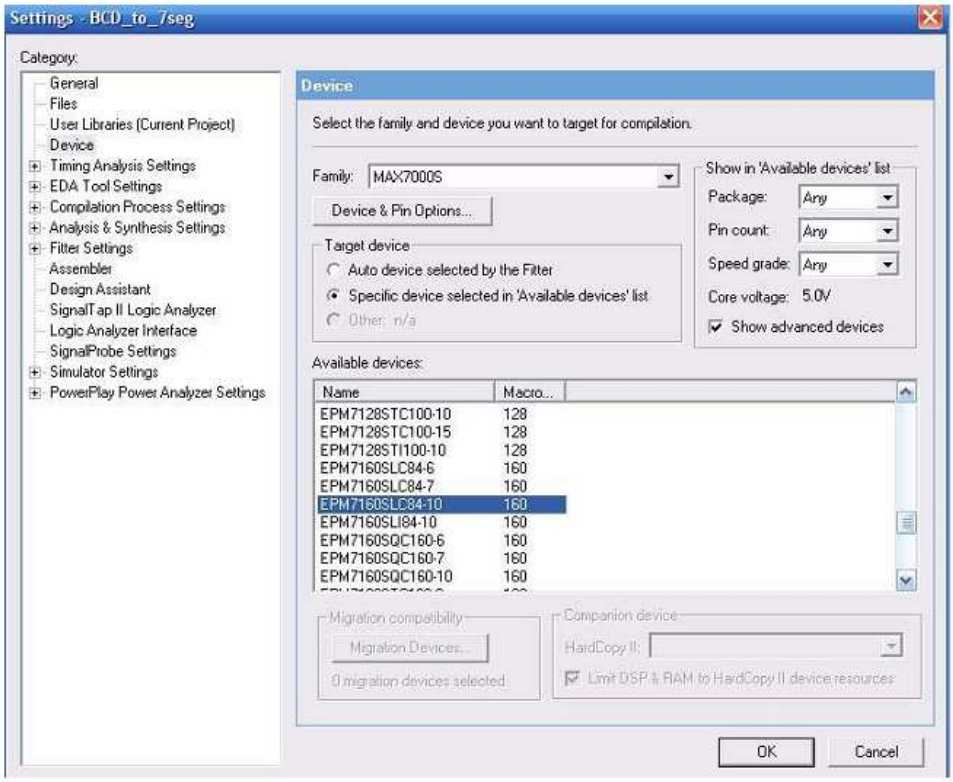
در این پنجره با کلیک بر روی گزینه Node Finderپنجره مذکور باز می‌شود. در پنجره Node Finderبا کلیک بر روی گزینه List، لیست ورودی ها و خروجی های مدار مربوطه ارائه می شود. برای مقدار دادن به ورودی ها با قرار دادن اشاره‌گر Mouseبر روی هر ورودی و کلیک کردن بر روی آن، کلیدهای کنار صفحه فعال خواهد شد. این کلید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



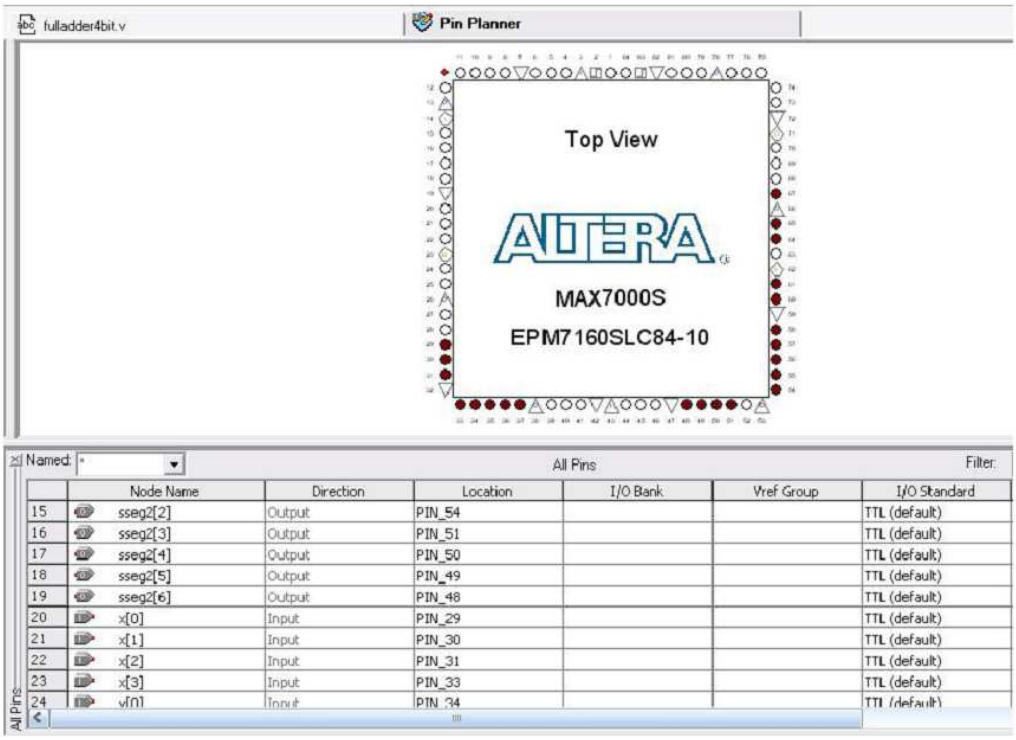
لازم به ذکر است که در مدارهای منطقی علاوه بر منطق صفر و یک، دو منطق دیگر نیز وجود دارد. منطق نا مشخص با (X)و منطق امپدانس بالا با (Z) نشان داده می‌شود. بعد از اختصاص مقادیر به ورودی ها برای مشاهده نتایج تحلیل نـرم افـزاری، گزینـه نشان داده شده در شکل زیر انتخاب می‌شود.



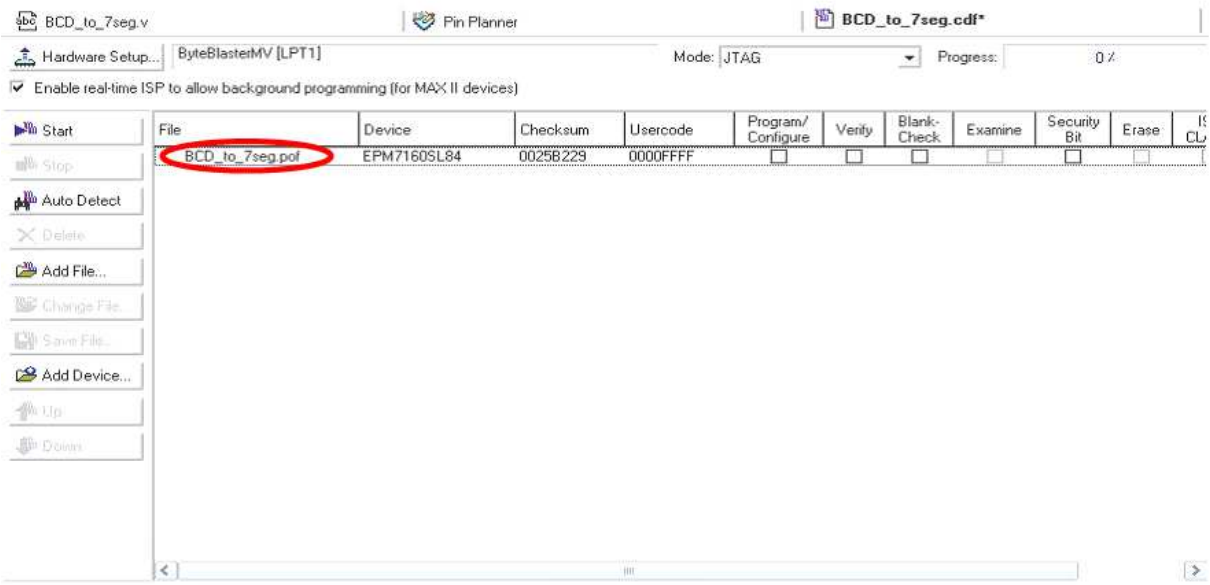
بعد از اتمام طراحی مدار و تحلیل نرم افزاری آن مدار آماده برنامه‌ریزی بر روی تراشه است. ابتدا باید قطعه‌ای که برای طراحی در نظر گرفته شده (یا به عبارتی همان قطعه‌ای که بر روی برد آزمایشگاه قرار دارد) به عنوان قطعه CPLDیا FPGAمورد نظر، مشخص شود. این کار با استفاده از کلید Deviceدر منوی Assignدر بالای صفحه صورت خواهد گرفت. همچنانکه قبلا نیز ذکر شد ALTERAقطعات مختلف و متنوعی دارد. برای انتخاب نوع قطعه باید شماره آن را به طور دقیق مشخص نمود.



با انتخاب نوع قطعه دوباره پروژه می بایست Compileشود. 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 زیر که از مسیر /AssignmentPinsباز شده است به طور واضح شکل تراشه، 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 LEDها، سوئیچ ، ICمولد 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 Name Nodeورودی و خروجی‌هایی که می‌بایست به پایه‌های قطعه نسبت داده شوند، مشخص می‌شوند.



برای نسبت دادن ورودی و خروجی‌های مدار به پایه‌های قطعه روی هر پایه دوبار کلیک کرده و ورودی یا خروجی مـورد نظـر را انتخاب می‌شود. بعد از مشخص نمودن همه ورودی‌ها و خروجی ها دوباره مدار Compileمی‌گردد. برای برنامـه-ریزی مدار بر روی تراشه نیاز به یک سخت افزار برنامـه‌ریـزی تحـت عنـوان PROGRAMMER JTAGمـی‌باشـد . ایـن سـخت افـزار بـا اسـتفاده از پـورت موازی (LPT1)به برد مورد نظر متصل می‌شود. انجام عمل برنامه‌ریزی با اسـتفاده از گزینـهProgrammerدر منـوی Toolsصـورت خواهد گرفت. شکل زیر پنجره برنامه ریزی را نمایش می‌دهد. با باز شـدن پنجره برنامه ریـزی فایـل *.pofنشـان داده شده، مربوط به پروژه مد نظر است.


BCD_to_7seg.v
Pin Planner
BCD_to_7seg.cdf*
Hardware Setup...
ByteBlasterMV [LPT1]
Mode: JTAG
Progress:
0 %
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

شروع به کار

با استفاده از یک سری پین ورودی/خروجی همچنین دوجهته (Bidir) و اتصال برخی از آنها بـه Vcc و Gnd و همچنین Dip-Switch ها و نصب خروجی به LED ها منطق روشن خاموش شدن آنهـا را تعیـین کنیـد. عـدد دلخواه را با سوئیچ ها روی LED ها نمایش دهید.

طراحی و پیاده‌سازی

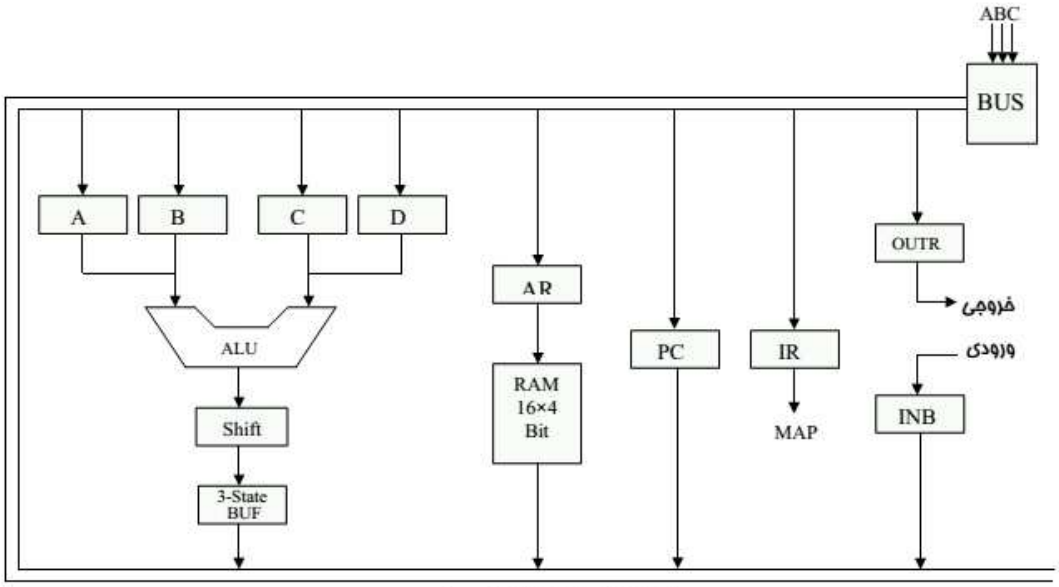
جمع کننده چهار بیتی یکی از مدارهایی منطقی پرکاربرد می‌باشد که بخشی از کارایی آن مشابه واحد محاسبه و منطق است. در این بخش قصد داریم یک جمع کننده چهار بیتی با استفاده از لاجیـک‌هـای پایـه (AND, OR, XORو....) و همچنین یک تراشه ALU(74181) را در مد جمع تنظـیم کنـیم. ایـن دو مـدار را در افـزاره برنامه‌پذیر طراحی، شبیه‌سازی و پیاده‌سازی کنیم.

- مدار جمع‌کننده ۴ بیتی با لاجیک‌های پایه را طراحی کنید.
- عملکرد تراشه 74181 را با توجه به برگه داده آن تعیین کنید.
- نتایج شبیه‌سازی و پیاده‌سازی هر دو روش را ارائه دهید.

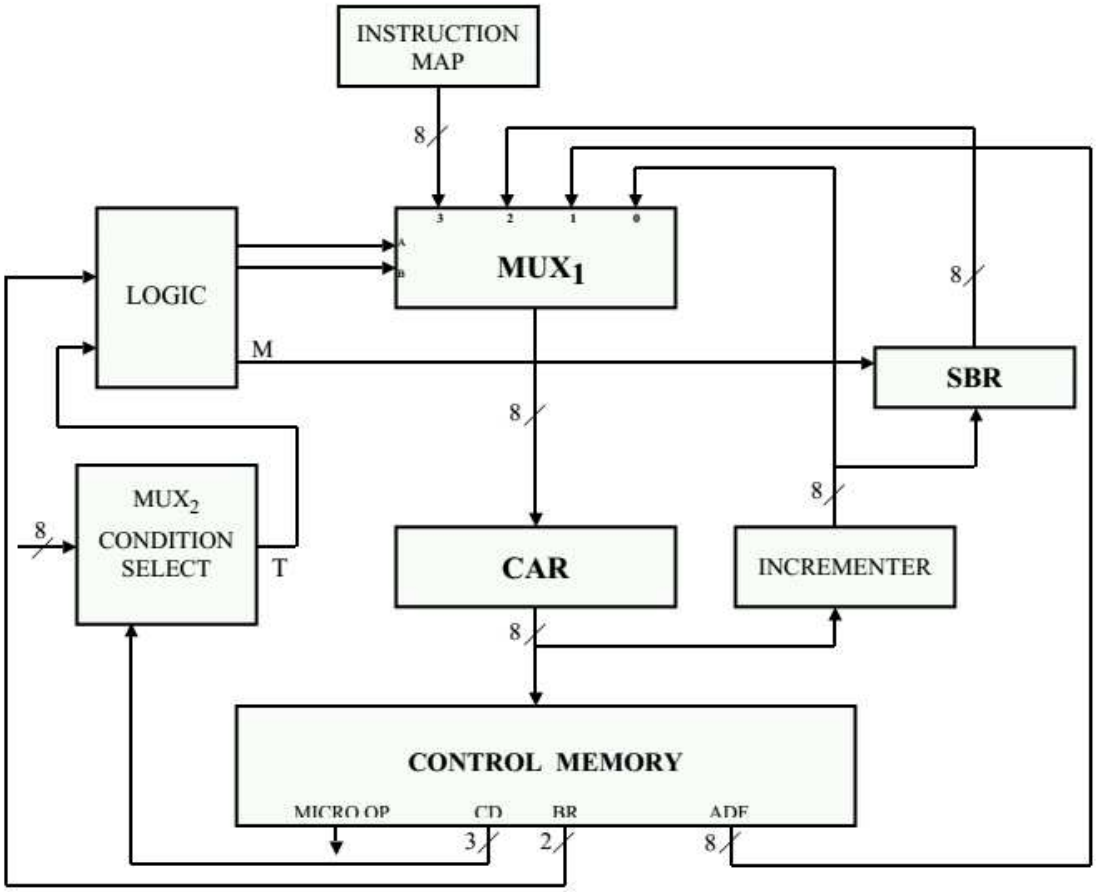
تحقیق کنید: ساختار جمع‌کننده بر اساس لاجیک‌های پایه (طراحی شما) با ساختار داخلـی تراشـه آمـاده جمـع-کننده (74181) چه تفاوتی دارد؟

طرح کلی مدارپردازنده چهار بیتی

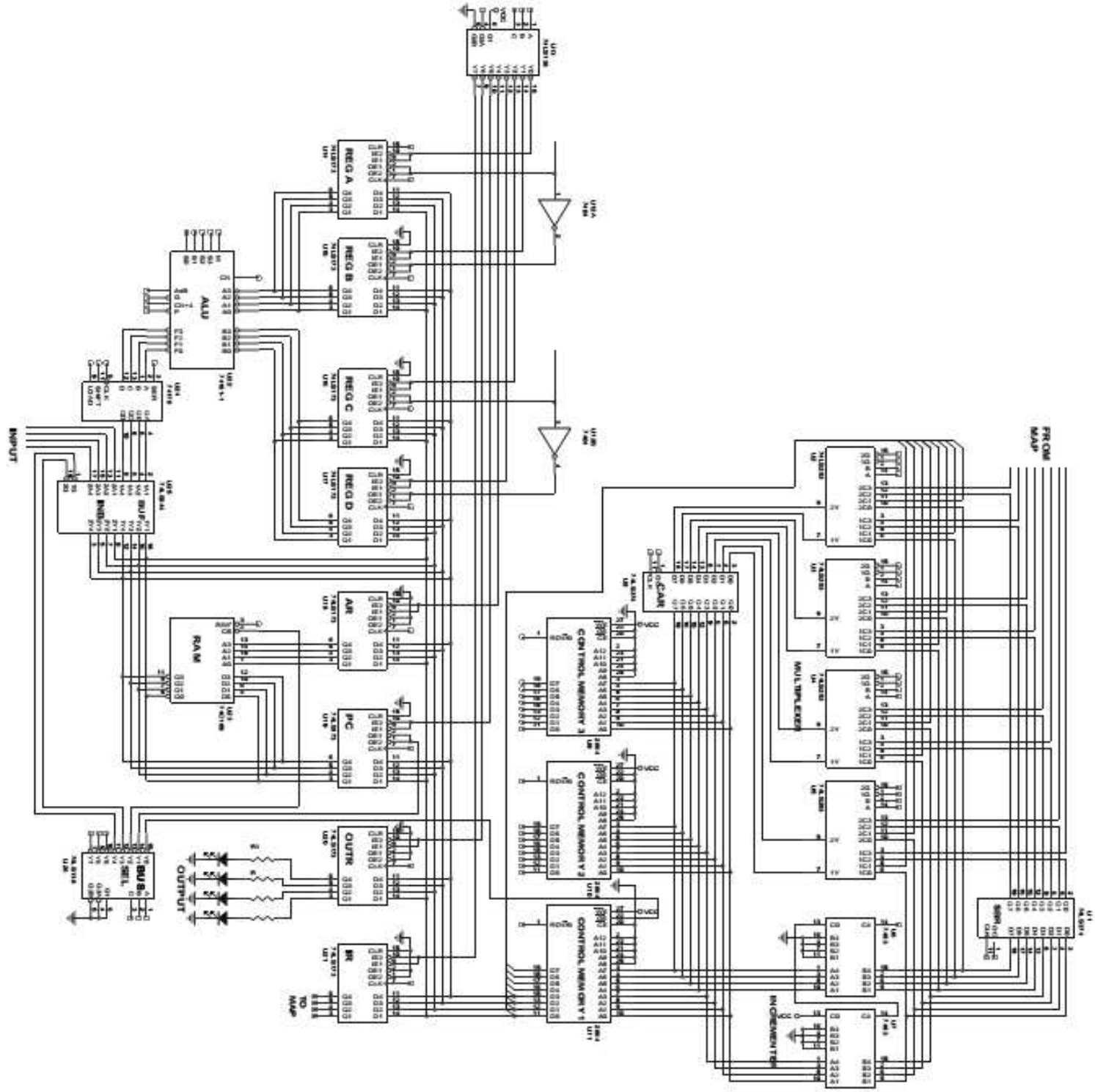
هدف ساخت یک CPU ی چهار بیتی با سازمان شکل زیر می‌باشد. این سازمان شامل چهار ثبات عمومیچهاربیتی A ، B ، C ، D سه ثبات خاص AR ، PC ، IR یک پورت خروجی OUTP و یـک پورتـورودی INP و یـک ALU و SHIFTER و RAMچهار بیتی می‌باشد. مشخصات هریک از این قطعات رادر جدول زیر ملاحظه فرمایید.واحد کنترل این CPU بروش میکروپراگرام طراحی شده و از ساختار کلی شکل (۲) پیروی می‌کند. شکل(۳) نقشه شماتیک کامل مدار را نشـان مـی-دهد که هم شامل واحد کنترل و هم سازما CPU است. البته اتصالاتسیگنال‌هـای کنتـرل و مـدارات مربـوط بـه انشـعاب شرطی و بیت‌های وضعیت بعهده دانشجویان گذاشته که براساس دستورکار تکمیل نمایند.



شکل (۱) سازمان CPU ی ۴ بیتی



شکل(۲) واحد کنترل بروش میکروپروگرام



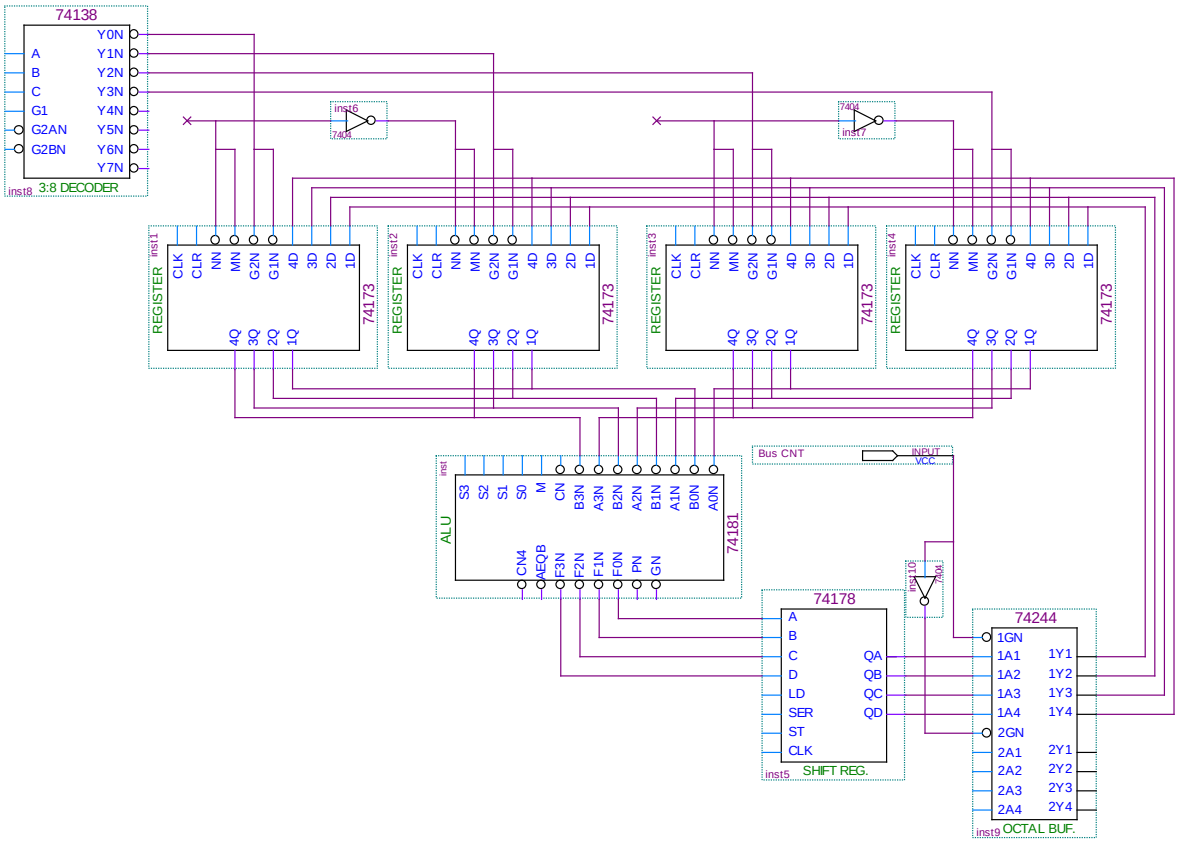
شکل(۳) نقشه شماتیک کل مدار

آزمایش شماره یک: واحد محاسبه و منطق

اهداف آموزشی:

- یادگیری روند بارگذاری داده در رجیستر ها از داخلی پردازنده
- یادگیری روند بارگذاری رجیسترهای مختلف برای عملیات ALU
- یادگیری اجرای ریزعمل‌های منطقی و محاسباتی در ریزپردازنده
- یادگیری ریزعمل‌های جابه‌جایی(شیفت) در ریزپردازنده

هدف ما در این آزمایش، پیاده سازی بخش محاسبه و منطق CPU می‌باشد، که شامل چهار ثبات عمومی چهاربیتی A ، B، C ، D و ALU، به همراه شیفت‌رجیستر و Bus Data است (مطابق شکل آزمایش یک). در این آزمایش دانشجویان با نحوه انتخاب یک رجیستر برای بارگذاری در ALU آشنا می‌شوند. همچنین با تنظیم پایه های انتخاب مد کاری واحد محاسبه و منطق (M, S0, S1, ….) عملیات مد نظر روی ورودی‌ها انجام شده و حاصل پس از رجیستر شدن یا عملیات شیفت به واسطه تراشه ۷۴۱۷۸ با یک بافر سه حالته (۷۴۲۴۴) به باس مشترک داده متصل می‌گردد. مقدار باس داده قابل بارگذاری در هر یک از رجیستر ها می‌باشد. ورودی رجستر مد نظر با استفاده از دیکدر ۷۴۱۳۸ فعال می‌شود تا داده قرار گرفته روی باس در رجیستر مد نظر بارگذاری گردد.



شکل مدار آزمایش شماره یک

فعالیت‌های قبل از آزمایش

۱- هر یک از تراشه های بکار رفته در مدار آزمایش یک را شناسایی و با مراجعه بهDataSheetعملکرد کلی هر یک از تراشه‌ها و ورودی‌ها و خروجی‌ها را بررسی نمایید. به عنوان نمونه ورودی های تراشه ۷۴۱۳۸ (G1, G2A, G2B) چه مقادیری باید باشند تا این دیکدر درست عمل کند؟ یا برای تراشه ALU مقادیر M, S0, S1, ... به چه سطوح منطقی باید وصل شوند تا تراشه عملیات جمع ریاضی انجام دهد؟

۲- مدار آزمایش یک را بررسی و تحلیل نمایید. به سوالات زیر پاسخ دهید.

- کلیه زوج ثبات هایی که می‌توانند در یک عملیات ALU شرکت کنند را نام ببرید؛ برای این منظور سطح منطقی – صفر یا یک- گیت های ۷۴۰۴ چه مقادیری باید باشد؟ **به عنوان نمونه، صفر بودن ورودی گیت ۷۴۰۴ سمت چپ باعث انتخاب رجیستر A خواهد شد.**
- انجام یک عمل مشخص(مثلا جمع) روی محتوای دو ثبات مورد نظر را مرحله به مرحله مشخص نمایید.

- استفاده از بافر 74244 در خروجی شیفت رجیستر 74178 چه لزومی دارد؟
- چگونه می‌توان حاصل عملیات ALU و شیفت را در یک ثبات دلخواهبارگذارینمود؟ ترتیب فعال کردنسیگنال‌ها را تعیین کنید؟

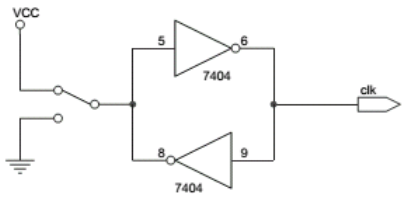
۳- مدار آزمایش یک را در محیط شماتیک Quartus رسم کنید و با کامپال کردن آن از درستی آن اطمینان حاصل پیدا کنید. تذکر مهم: اگر پایه خروجی تراشه ها را فعلا استفاده نکرده اید، برای ممانعت از خطا در کامپایل آنها را به صورت Unused (راست کلیک روی تراشه، Properties، تب Ports، انتخاب پورت مورد نظر، Unused کردن) تبدیل کنید.

۴- فایل شبیه‌سازی مناسبی برای هر یک از موارد بند ۲ در محیط Quartus تدارک دیده و موارد ذکر شده را با شبیه-سازی اعتبارسنجی کنید.

۵- شرح مختصری از موارد فوق را بعنوان پیش گزارش آماده نموده و در ابتدای جلسه آزمایشگاه تحویل دهید (نتایج شبیه‌ساز نیز ارائه گردد).

انجام آزمایش

تکه مدار زیر را جهت وارد نمودنکلاک پالسبصورت دستی به شماتیک قبلی خود اضافه کنید:



مدار تکمیل شده آزمایش یک را که قبلا در محیط شماتیک Quartusرسم و شبیه‌سازی کرده‌اید را با اختصاص پایه در FPGA پروگرام کنید.

مدار تکمیل شده چیست؟در مدار بخش ALUیک سری از پایه های ورودی و خروجی به جایی متصل نشده‌اند، این پایه ها باید توسط دانشجویان طراحی و تکمیل شوند. پایه هایی که در طول آزمایش مقدار منطقی ثابت دارند، به Gnd یا Vcc وصل می‌شوند. پایه هایی که دانشجویان نیاز دارند طی تست آنها را تغییر دهند به صورت پورت ورودی (input) به DipSwitch ها متصل گردد. همچنین برای اطمینان از روند انجام عملیات مقدار باس داده با چهار LED نمایش داده شود.

جهت اطمینان از درستی مدار:

۱) با استفاده از سیگنال‌های کنترلی که به سوئیچ‌ها اختصاص داده اید، مقادیر متفاوتی در ثبات‌های A، B ، Cو Dبارگذاری کنید.در هر مرحله از بارگذاری با نگاه به وضعیت LEDاز روند بارگذاری اطمینان حاصل کنید.

۲) با انتخاب خروجی مناسب از صحت مقادیر موجود در ثبات‌ها مطمئن شوید.

۳) با دادن ورودی مناسب به ALU نتیجه عملیات جمع و تفریق و AND و... را مشاهده نمایید. راهنمایی مختصر برای اعمال ورودی: برای دادن مقدار به رجیستر ها چهار پین مسیر بلااستفاده تراشه ۷۴۲۴۴ را به DIP Switch ها وصل کنید. خروجی این بخش را به شکل مناسب به باس داده متصل کنید.

۴) یکی از مقادیر خروجی عملیاتALUرا در شیفت رجیسترLoadنموده و عملیات شیفت را روی آنآزمایش کنید.

۵) مقدار حاصل را در یکی از ثبات‌های A، B، C یاDقرار دهید.

تذکر: با توجه به اینکه در همه ریزپردازنده ها منحصرا یک خط کلاک و یک خط بازنشانی (ریست) وجود دارد، تمام خطوط کلاک به هم وصل شود و به عنوان یک پین ورودی (CLK) لحاظ گردد. در رابطه با خط بازنشانی (RST)نیز همین کار صورت پذیرد.

آزمایش ۲: پیاده‌سازی واحد کنترل به روش Microprogramming

اهداف آموزشی:

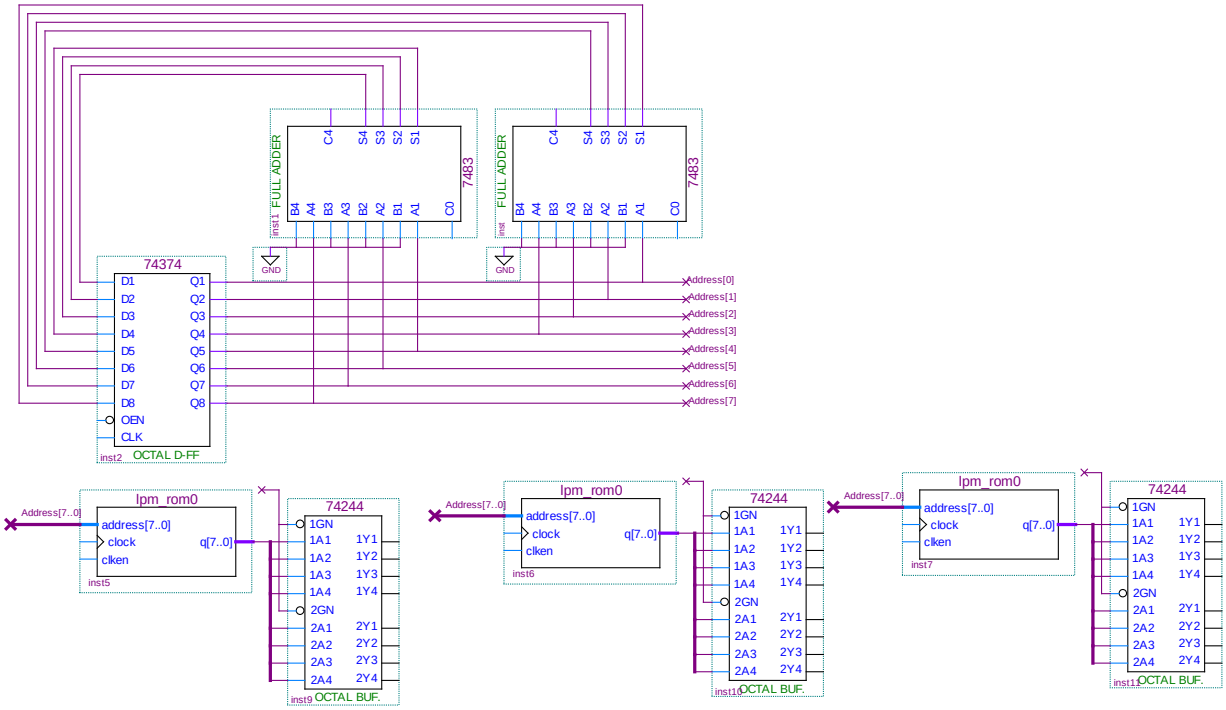
- آشنایی با مفهوم گذرگاه آدرس مشترک
- آشنایی با روند ایجاد کدهای دستورالعمل‌ها در یک پردازنده
- آشنایی با بارگذاری دستورات در حافظه E2PROM به صورت دستی
- آشنایی با روند اجرای برنامه از حافظه برنامه‌نویسی
- آشنایی با روند آدرس دهی مستقیم حافظه برنامه نویسی

واحد کنترل به دو صورت قابل تحقق می‌باشد: Hardwired control unit و Microprogramming. در روش اول با استفاده از گیت‌های منطقی پایه (AND, OR, XOR,....) مدار کنترلی طراحی می‌گردد. در روش دوم ریزدستورات در حافظه ذخیره شده و با حداقل سخت افزار به نقاط کنترلی اعمال می‌گردد. این روش نسبت به روش اول از نظر حافظه بهینه نمی‌باشد و حافظه بیشتری نیاز است. اما توسعه دستورات در آن آسان است و نیازی به طراحی مجدد واحد کنترل نیست.

هدف ما در این بخش پیادهسازی بخشی از واحد کنترل با رویکرد Microprogramming می‌باشد. این بخش دربردارنده مولد آدرس بعدی (NextAddressGenerator) وتولیدسیگنال‌هایکنترلیبعنوانبخشیازواحد کنترل ریزپردازنده است. که این بخش‌ها شامل:

- حافظه برنامه نویسی (Control Memory)ساخت حافظه با Mega-Wizard
- رجیستر کنترل آدرس (Control Address Register) تراشه ۷۴۳۷۴
- افزایش دهنده آدرس (Incremental) تراشه های ۷۴۸۳

مطابق مداربندی این قسمت،فقطامکاناجرایریزدستورالعمل‌های متوالی ایجادمی‌شود.در آزمایش‌هایبعدیبا افزودنمالتیپلکسرامکاناجرایریزدستورالعمل‌های پرش(JUMP)، فراخوانی (CALL) و بازگشت از توابعRETURN رابهواحدکنترلاضافهخواهیمنمود.



شکل مدار آزمایش شماره دو

فعالیتهایقبلازآزمایش

۱- تفاوت دو روش Hardwired control unit و Microprogramming در طراحی واحد کنترل را شرح دهید.

۲- هریکازتراشههایبکاررفتهدرمدارشکلبالاراشناسـاییوبامراجعهبه دیتـا شـیت عملکردکلـی تراشـه وورودیهاوخروجیهارابررسینمایید. به عنوان مثال برای اینکه ۷۳۳۷۴ برای اینکه عمل رجیستر کردن داده را بـه درسـتی انجام دهد، چه تنظیماتی روی پایه های (OC و CLR) نیاز می‌باشد.

۳- مدارشکلآزمایش دورابررسیوتحلیلنمایید. بهسوالاتزیرپاسخدهید.

- نحوه ساخت حافظه PROM با Mega-Wizard، بارگذاری و نحوه ساخت فایل مقداردهی اولیه بـا MIFGenerator را تعیین کنید.
- نحوهاتصال C0 و C4 درتراشههای7483رامشخصکنید.**راهنمایی: این بخش برای ایجاد آدرس بعدی قرار داده شده است. با دو جمع کننده چهار بیت یک جمع کننده ۸ بیتی نیاز داریم که در هر کلاک یک واحد افزایش پیدا کند.**
- چنانچـه مـی بینیـد، در هـر کـلاک آدرس بـا شـروع از صـفر، یـک واحـد افـزایش مـی‌یابـد. روشـیبرای بازنشـانی مدارجهتشروعبرنامهازآدرسصفرارائهدهیدونحوهاتصال CLK و OC در تراشههای۷۴۳۷۴رامشخصکنید.

- باتوجهبهشکل "نقشهشماتیککلمدار"کاربرداتصالنیبلپایین CONTROLMEMORY 1 به گذرگاه مشترک داده چیست؟

۴- ریز برنامهایبنویسیدکهعملیاتزیررانجامدهد.

- چهارمقدارمتفاوتراددرچهارثبات A تا D قراردهد.
- محتوای B وC را باهمجمعکردهوحاصلبهثباتA منتقلکند.

مواردخواستهشدهبندهایفوقرابعنوانپیشگزارشآمادهنمودهودرابتدایجلسهآزمایشگاهتحویل دهید.

انجامآزمایش

۱- مدارشکل آزمایشرا صرفا با یک PROM ببندید، و مقادیر قرار داده شده در حافظه PROM را که به هنگام ایجاد آن با Mega-Wizard و از طریق فایل mif. بررسی کنید. لازم به ذکر است که برای دیدن محتویات مد نظر خود، از ۸ عدد LED در خروجی PROM استفاده کنید.

۲- این مدار را به مدار آزمایش یک متصل کنید. به گونهی که خروجی هایE2PROM برای دادن داده به گذرگاه مشترک داده و همچنین کنترل سایر نقاط آزمایش یک استفاده شود. مدار را کامپال کنید تا خطایی وجود نداشته باشد. سعی کنید صرفا دو خط کلاک و بازنشانی به عنوان ورودی و ۴ خط گذرگاه داده به عنوان خروجی داشته باشید. بقیه قسمت هایی که نیاز به کنترل دارند، از طریق خروجی های E2PROM تعیین وضعیت شوند. به عنوان مثال اگر در آزمایش یک انتخاب رجیستر از طریق پایه های دیکدر را با استفاده از سوئیچ انجام میدادید، حال این کار را با خطوط خروجی E2PROM کنترل کنید.

۳- ریزبرنامه بارگذاری ثبات ها و جمع کردن محتویات را متناسب با سیمبندی خود از زبان اسمبلی به زبان ماشین تبدیل کرده و روند اجرای آن را در هر کلاک نشان دهید.

برنامه اسمبلی نمونه:

```
LD      A       3;
LD      C       7;
ADD     D ← A + C;
AND     A       D;
```

زبان ماشین:

Address(Hex)	E2PROM 2	E2PROM 1	E2PROM 0
00			
01			
02			

تذکر: با توجه به اینکه ممکن است به واسطه خطاهای دیبانسینگ کلیدها در عمل با مشکلاتی مواجه شوید پیشنهاد می شود از مدارهای تاخیر دهنده در مسیر کلاک استفاده کنید یا با تقسیم کلاک اسیلاتور روی بورد یک کلاک یک هرتز(یک ثانیه‌ای) برای اجرای دستورات ذخیره شده در E2PROM تولید کنید.

آزمایش ۳: اجرای دستورات و انتقال داده از حافظهRAM

اهداف آموزشی:

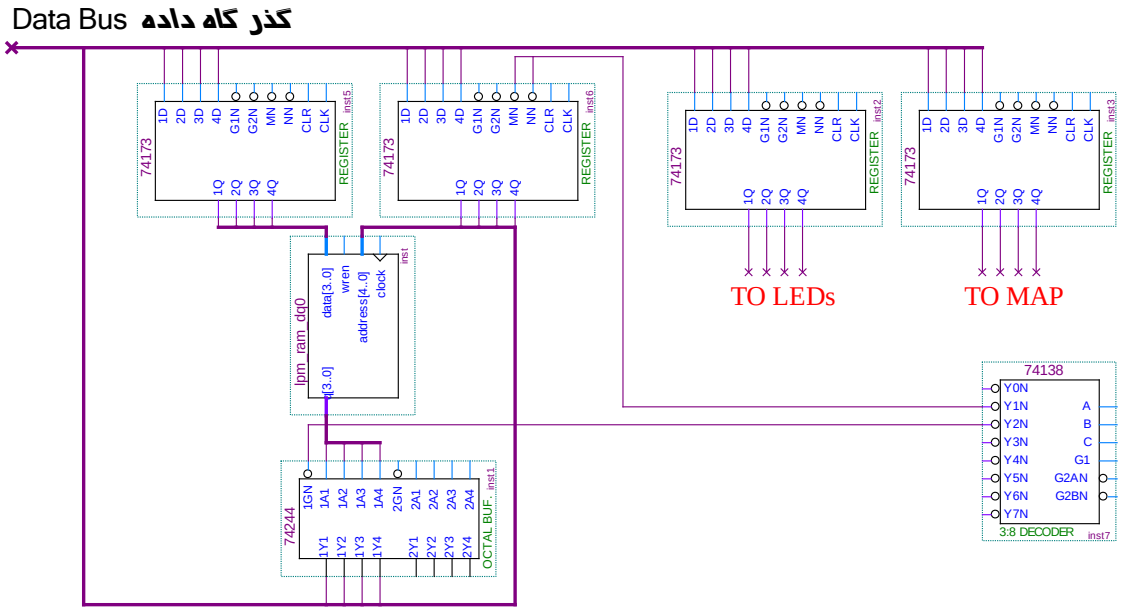
- آشنایی با روند انتقال دستورات از حافظه سرعت پایین (حافظه برنامه‌نویسی) به حافظه سرعت بالا (RAM)
- روند اجرای دستورات از حافظه RAM
- ذخیره موقت داده در حافظه RAM
- انتقال داده های حاصل از محاسبات از حافظه RAMبه حافظه دائمی

چنانچه می‌دانیم سرعت اجرای دستورات و به طور کلی تبادل اطلاعات در حافظه های RAM به مراتب بالاتر از حافظه‌های E2PROM می‌باشد. به همین دلیل پس از روشن شدن کامپیوترها بخش‌هایی از برنامه از حافظه E2PROM ها به حافظه هایی از جنس RAM منتقل می‌شوند (حافظه های Cash یا RAM)، و سپس اجرا می‌شوند. البته بحث حافظه نهان بحث مبسوطی است که خارج از حوزه بحث ماست. در این آزمایش صرفا یک آشنایی اولیه با این روند مد نظر می‌باشد. در این آزمایش قصد داریم به بررسی و پیاده سازی روال‌های انتقال و ذخیره داده از یک آدرس مشخص RAM به طور مستقیم و غیر مستقیم بپردازیم.

در این بخش مطابق شکل زیر یکسری رجیسترهای جدید به همراه RAM به معماری ریزپردازنده ۴ بیتی اضافه می‌کنیم. این رجیسترها عبارتند از:

- رجیستر آدرس (Address Register)
- شمارنده برنامه (Program Counter)
- رجیستر خروجی (Output Register)
- رجیستر دستور (Instruction Register)

چناچه در شکل می بینید، با توجه به اینکه تعداد المان‌های متصل به گذرگاه مشترک بیشتر شده است، برای افزودن امکان قطع و وصل شدن به باس، تراشه ۷۴۱۳۸ به طرح اضافه شده است. خروجی باس را نیز به واسطه یک رجیستری به LED ها وصل کرده‌ایم. پس نمایش خروجی از این به بعد نظام‌مندتر خواهد شد و نیازمند ریزعمل خاص خود می‌باشد که با کنترل دیکدر ۷۴۱۳۸ آزمایش یک قابل اجرا خواهد بود (به معماری کلی ریزپردازنده ۴ بیتی مراجعه کنید).



فعالیتهایقبلازآزمایش

۱- هریکازتراشههایبکاررفتهدرمدارشکلبالاراشناسـاییوبامراجعهبه دیتـا شـیت عملکردکلـی تراشـه وورودیهاوخروجیهارابررسینمایید.

۲- مدارشکلآزمایش دورابررسیوتحلیلنمایید. بهسوالاتزیرپاسخدهید.

- دلیل استفاده از تراشه‌های ۷۴۱۷۳ و ۷۴۱۳۸ چیست؟
- سرهای اتصال تمام تراشه ها را مشخص کرده و سیم‌بندی آنها را تکمیل کنید.
- روند اجری دستورات از RAM به چه صورت خواهد بود؟ این روند را با ریز دستورات مربوطه به طـور کامـل شرح دهید.

انجام آزمایش:

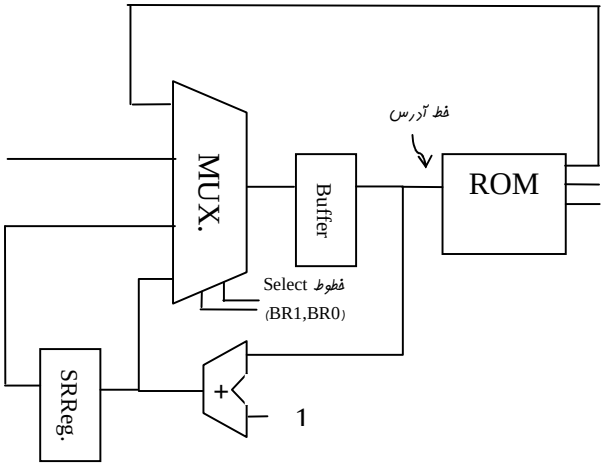
آزمـایش ۴: اجـرای دسـتورات پـرش (Jump)، فراخـوانی تـابع (Call)، بازگشـت از تـابع (Return)(فصل ۷ موریس مانو)

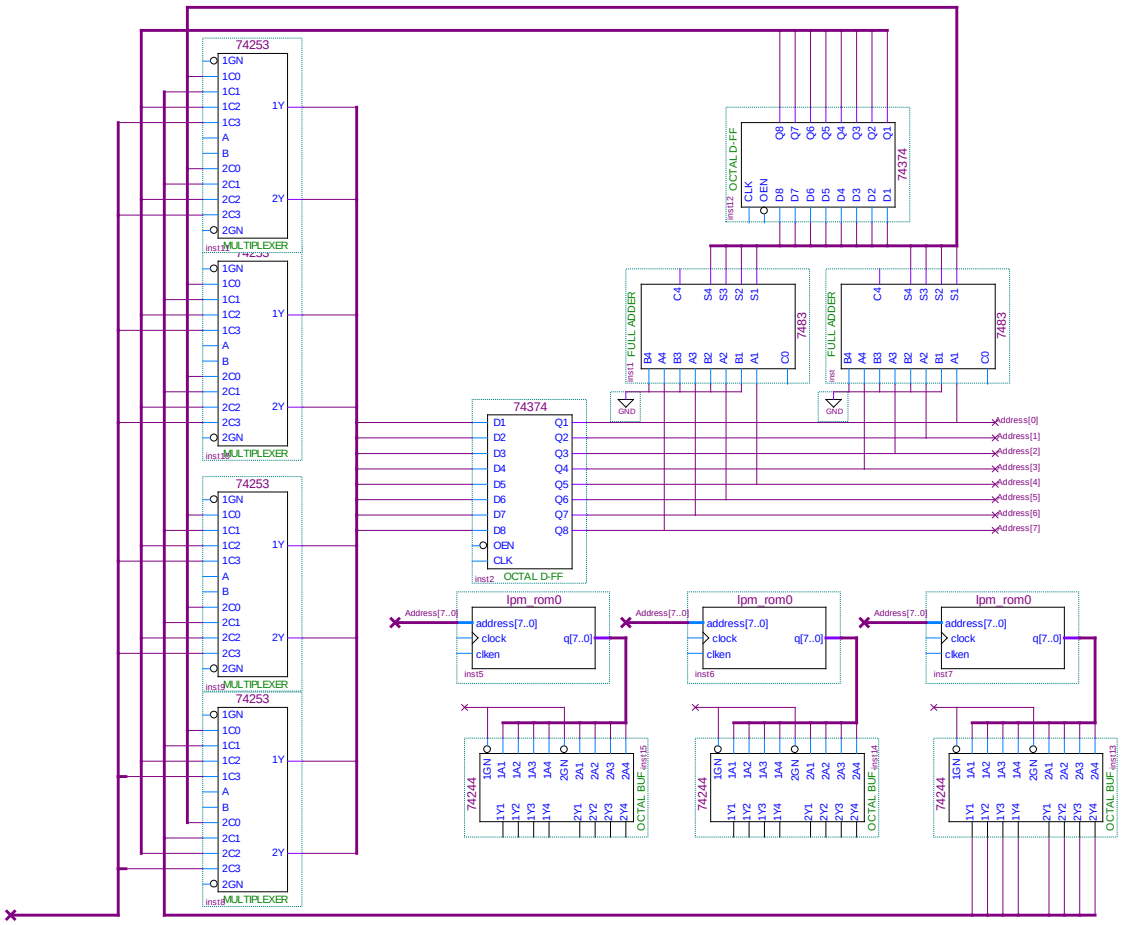
اهداف آموزشی:

- آشنایی بابخش کنترل آدرس در شرایط پرش و فراخوانی تابع
- اجرای دستورات پرش و فراخوانی تابع در روند اجرای برنامه
- مدیریت حافظه برنامه نویسی برای قرار دادن توابع مختلف

چنانچه می دانیم بخشی از دستورات در معماری کامپیوتر به دستورات پرش شرطی و غیر شرطی و همچنین فراخوانی و بازگشت از توابع اختصاص یافته است. در این بخش قصد داریم با توسعه بخش آدرس دهی حافظه برنامه‌نویسی، امکان اجرای دستورات پرش و فراخوانی توابع مختلف را فراهم سازیم. به این منظور بخش آدرس دهی را مطابق شکل زیر توسعه می‌دهیم. چنانچه در شکل مشاهده می‌شود آدرس دهی حافظه برنامه نویسی با استفاده از مالتی‌پلکسرهای ۷۴۲۵۳ از چهار مسیر مختلف تامین می‌گردد.

I. خروجی جمع‌کننده (افزایش عادی شمارنده آدرس یا Increment) که حالت عادی ادرس دهی بدون پرش می‌باشد.
II. خروجی رجیستر زیر روال (.Sub Routine Reg) که موظف به ذخیره آدرس محل پرش است.
III. خروجی حافظه برنامه نویسی که موظف به بارگذاری آدرس جدید برای پرش یا فراخوانی تابع است.
IV. ورودی از واحد MAPکه می‌توان از آن برای دسترسی مستقیم به حافظه استفاده کرد.





شکل مدار آزمایش ۴

فعالیت‌های پیش از آزمایش:

۱- با مطالعه برگه داده قطعات جدید اضافه شده، همچنین با توجه به روند مطلوب جهت اجرای دستورات پرش و فرخوانی، سیم‌بندی را تکمیل کنید.

۲- خطوط انتخاب آدرس حافظه برنامه‌نویسی (خطوط Select مالتی پلکسر) را به بخش از خروجی های حافظه E2PROM متصل کنید. تا بتوانید دستورات پرش را انجام دهید.

۳- روند بارگذاری آدرس از مسیرهای مختلف برای عملیات پرش، فراخوانی و بازگشت‌به/ از آدرس دلخواه را بنویسید. توجه داشته باشید دو بیت اختصاصی به انتخاب‌گر مالتی پلکسر به این صورت باشد که:

مفهوم	خطوط Select(BR1,BR0)
JMP	00
CALL	01
RET.	10

انجام آزمایش:

۱- پرش بدون شرط:برای برنامه زیر زبان ماشین متناسب با سیم‌بندی معماری کامپیوتر خود تهیه کنید (Acc همان رجیستر قرار داده شده در خروجی ALU می‌باشد.):

```
        LD      A       2;
        LD      C       3;
L1:     ADD     Acc← A + C;
        OUT     outr    Acc;
        SHL     A       1;
        JMP     L1;
```

۲- فراخوانی و بازگشت از تابع:می خواهیم یک برنامه بخش از برنامه را در آدرسی از حافظه بنویسیم به طوری که پس از روشن شدن پردازنده و اجرای چند آدرس اول برنامه به آن آدرس منتقل و پس از اجرای محتویات آن بخش برگردد. حالت ساده شده آن به صورت زیر در نظر بگیرید (L1 محل فراخوانی زیرروال را برابر 20Hex در نظر بگیرید):

```
00      LD      A       2;
01      LD      C       3;
02      GOTO    L1;
03      ADD     Acc←A + D;
04      OUT     outr    Acc;
.
.
L1:
20      ADD     Acc← A + C;
21      OUT     outr    Acc;
22      LD      D       Acc;
23      RETURN;
.
.
```

تذکر: در فایل mif. خود مقادیر آدرس‌های بلا استفاده صفر باشد. صرفا با شروع از آدرس صفر و آدرس 20Hex و به تعداد لازمحافظه مقداردهی گردد.

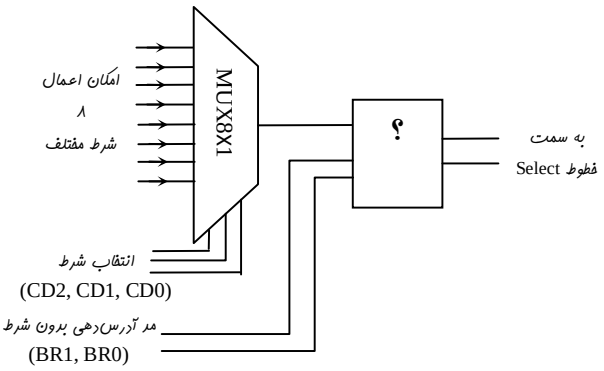
آزمایش ۵: تکمیل دستورات پرش(Jump)، فراخوانی تابع (Call) با افزودن بخش شرطی

(فصل ۷ موریس مانو)

اهداف آموزشی:

- آشنایی با بخش نحوه اجرای دستورات شرطی
- آشنایی با نحوه استفاده از بیت‌های شرطی در برنامه‌نویسی و فراخوانی و بازگشت از زیرروال
- آشنایی با نحوه ساخت و طراحی سیگنال‌های شرطی در معماری کامپیوتر

چنانچه می‌دانیم بخشی از دستورات و فراخوانی توابع به واسطه دستورات شرطی تکمیل می‌گردد. در این آزمایش قصد داریم به نحوه ایجاد و اعمال سیگنال‌های شرطی در معماری پردازنده ۴ بیتی بپردازیم.برای این منظور تکه مدار زیر را به بخش **خطوط Select در آزمایش چهار**اضافه می‌کنیم:



فعالیت‌های پیش از آزمایش:

۱- اگر از تراشه ۷۴۱۵۱ به عنوان مالتی‌پلکسر ۸ به یک در بخش اعمال شرط استفاده شود، با مطالعه برگه داده این قطعه سیم‌بندی آن را در معماری پردازنده خود تکمیل کنید.

۲- در شکل فوق مدار منطقی ؟را مطابق منطق زیر و به ساده‌ترین صورت (با تشکیل جدول کارنو) طراحی کنید.

$BR_1\ BR_0 = 00$ { C (شرط) = 1 → JMP ; C (شرط) = 0 → ادامه برنامه

$BR_1\ BR_0 = 01$ { C (شرط) = 1 → CALL ; C (شرط) = 0 → ادامه برنامه

$BR_1\ BR_0 = 10$ → RET.

$BR_1\ BR_0 = 11$ → MAP

BR1	BR0	C	S1	S0	مد
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

۳- اختصاص شرط های مختلف به ورودی مالتی‌پلکسر به صورت دلخواه می‌تواند انجام شود. ولی در این آزمایش مطـابق جدول زیر لحاظ کرده و تحلیل کنید:

شماره ورودی مالتی‌پلکسر	سیگنال متصل
0	ثابت GND
1	ثابت VCC
2	AEQB از واحد ALU
3	Q_0A خروجی اول رجیستر A
4	رزرو
5	رزرو
6	رزرو
7	رزرو

انجام آزمایش:

۱- **(برنامه ضرب):** در زیر بخشی از برنامه به زبان اسمبلی نوشته شده است که پس از بارگذاری مقادیر مطلـوب در دو رجیستر A و C آنها را در هم ضرب کرده و حاصل را روی خروجی نشان می‌دهد. این برنامه را بررسی و تکمیـل کـرده و متناسب با معماری کامپیوتر ۴ بیتی خود به زبان ماشین تبدیل، برنامه‌ریزی و تست کنید. (لازم به ذکر اسـت کـه تـابع ضرب در آدرس 20Hex قرار داده شود.OUT = 1 + INPUT ×4)**راهنمایی:** با توجه به اینکه بیت سـوم در عـدد چهـار(00000100) یک است، مطابق شرط چهارم جدول مالتی‌پلکسر از شیفت به چپ استفاده کرده‌ایم.

```
00      LD      A       4;
01      LD      C       INPUT;      -- بارگذاری از ورودی انجام می شود
02      GOTO    L1;
03      ADD     Acc←B + 1;
04      OUT     outr    Acc;
.
```

```
.
L1:
20      ADD     Acc← A + C;
21      SHL     A;
22      JPMC    Q0A  L2;
22      JPM     L1;
L2:
23      RETURN;
.
.
```

۲- (**برنامه مقایسه**): برنامه ای نمونه زیر که دو مقدار قرار داده شده از رجیستر های A و C را با هم مقایسه کرده و در صورت مساوی بودن مقدار FF_{Hex} و در غیر این صورت مقـدار 55_{Hex}را روی خروجـی نمـایش دهـد. برنامـه را بررسـی و تکمیل کرده و متناسب با معماری کامپیوتر ۴ بیتی خود به زبان ماشین تبدیل، برنامه‌ریزی و تست کنید.

```
00      LD      A       INPUT;      -- بارگذاری از ورودی انجام می شود
01      LD      C       INPUT;      -- بارگذاری از ورودی انجام می شود
02      CMP     A       C;
03      JUMC    L1;
04      OUT     outr    FF;
05      NOP;
L1:
06      OUT     outr    55;
07      NOP;
```

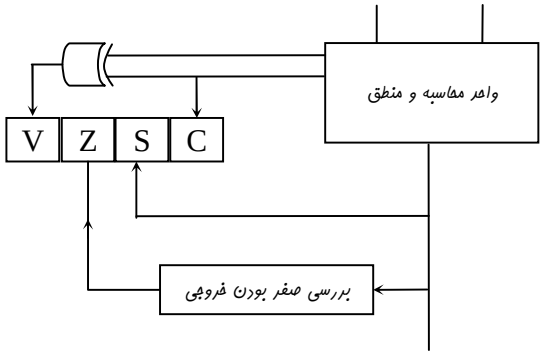
آزمایش ۶: افزودن بیت‌های پرچم

(فصل ۸ موریس مانو)

اهداف آموزشی:

- آشنایی بانحوه توسعه دستورات شرطی
- چک کردن شرط صفر شدن حاصل عملیات ALU
- چک کردن بیت نقلی به منظور نمایش محاسبات عددی بزرگ‌تر

بیت‌های پرچم نقش بسیار مهمی در معماری کامپیوتر دارند، از مهم‌ترین این بیت‌ها می‌توان به بیت صفر و بیت رقم نقلی اشاره کرد. بخش مهمی از دستورات پرش و حتی بازگشت از توابع بر مبنای صفر بودن خروجی واحد ALU(JZ, JNZ) استوار است. بر این اساس در این بخش قصد داریم امکان بررسی صفر بودن واحد محاسبه و منطق را فراهم آوریم. همچنین با توجه به اینکه ممکن است در برخی از دستورات محاسباتی ممکن است با سرریز شدن خروجی مواجه شویم، برای این موارد نیز دستورات چک کردن بیت نقلی اهمیت خاصی دارد و دستورات پرش خاص بیت نقلی در پردازنده ها لحاظ می‌گردد (JC, JNC).



فعالیت‌های پیش از آزمایش:

۱- مدار مناسبی برای بررسی صفر بودن خروجی ALU طراحی کنید. مدار طراحی شده را به بخش ALU معماری کامپیوتر ۴ بیتی خود اضافه نمایید.

۲- دو بیت پرچم Z و C را به شکل مناسبی به ورودی‌های رزرو آزمایش قبلی اضافه کنید. تا به عنوان چک بیت بتوانید استفاده کنید.

انجام آزمایش:

۱- (پیاده‌سازی LOOP):برنامه زیر برای نمایش معکوس از ۱۰ الی صفر لحاظ شده است. به گونه‌ای که با بارگذاری رجیستر A با شروع از ۱۰ در هر کلاک یک واحد این رجیستر را کاهش داده و روی خروجی منتقل می‌کند. در انتها با چک کردن صفر بودن خروجی ALU برنامه مجددا A را با ۱۰ بارگذاری کرده و مشابه قبل به روند کاهش و نمایش ادامه می‌دهد. این برنامه را بررسی و در صورت نیاز تکمیل کنید. با تبدیل آن به زبان ماشین روی کامپیوتر ۴ بیتی خود پیاده‌سازی کنید.

```
L1:
00      LD      A       10;
L2:
01      DEC     A;
02      OUT     outr    A;
03      JNZ     L2;
04      JMP     L1;
05      NOP;
```

۲- (ضرب اعداد بزرگ):چنانچه می دانیم حاصل ضرب دو عدد n بیتی باید در یک رجیستر 2n بیتی قرار داده شود تا مقدار آن بر اثر سرریز از دست نرود.در این بخش می‌خواهیم با استفاده از روش جمع‌های متوالی و با کمک پرچم صفر و پرچم بیت نقلی حاصل‌ضرب دو عدد ۴ بیتی دلخواه را محاسبه و در دو خانه از حافظه RAM ذخیره کنیم. سپس از حافظه RAM این دو کلمه ۴ بیتی را به صورت متوالی روی LEDهای خروجی نمایش دهیم.

آزمایش ۷: ارتباط با ورودی/ خروجی (IO)

(فصل ۱۱: سازمان ورودی خروجی کتاب معماری کامپیوتر موریس مانو)

اهداف آموزشی:

- آشنایی با نحوه تبادل داده با ورودی خروجی ها
- آشنایی با دست تکانی (Handshaking) در معماری کامپیوتر
- انتقال داده از ورودی به RAM و از RAM روی خروجی

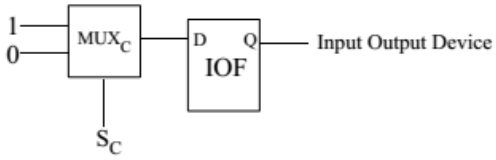
یکی از مهم‌ترین بخش‌های هر کامپیوتر ارتباط با ورودی خروجی‌ها می‌باشد. هدف از این ارتباط انتقال داده و دستورات می‌تواند باشد. به طور کلی دو روش انتقال هم‌زمان و انتقال غیر هم‌زمان در این حوزه مطرح است. در انتقال غیر همزمان با یکسری سیگنال‌ها دست‌دهی مانند Strobe، Acknowledge و غیره از انتقال داده اطمینان حاصل می‌گردد. در این آزمایش قصد داریم به معماری کامپیوتر خود این قابلیت را اضافه کنیم.

فعالیت‌های قبل از آزمایش:

۱- با افزودن یک‌سری تراشه ۷۴۱۷۳ باس داده را به ۴ LED دیگر متصل کنید. با این کار دو سری خروجی در معماری کامپیوتر ۴ بیتی خواهید داشت. نحوه کنترل این بخش اضافه شده را در طرح خود بررسی و تعیین نمایید.

۲- به یکی از پایه‌های مالتی‌پلکسر ۸ به ۱ که در بخش دستورات شرطی به صورت رزرو لحاظ شده بود را به ورودی از DIP Switch وصل کنید. تا به عنوان سیگنال دست‌دهی از آن استفاده نمایید. زمانی که این سیگنال فعال شود، به معنی این است که ورودی داده آماده دارد و می خواهد در RAM بارگذاری کند. این روند دست‌دهی را کامل کرده و تحلیل کنید.

۳- برای اجرای درخواست ارسال داده روی خروجی نیز می‌توانید مداری مشابه شکل زیر به معماری کامپیوتر ۴ بیتی خود اضافه کنید. نحوه عملکرداین مدار را بررسی کنید و سیگنال‌های کنترلی آن را در طرح خود تعیین کنید.



انجام آزمایش:

۱- برنامه‌ای بنویسید که با چک کردن سیگنال درخواست تبادل داده از ورودی (IN = 1) پنج داده مختلف را از ورودی دریافت و در حافظه RAM ذخیره کند. با پایان دریافت داده از ورودی آنها را یک در میان به خروجی‌های یک و دو بدهد.

پروژه ها:

۱- مجموعه دستوالعمل های امکان‌پذیر خود را برای سیستم طراحی شده ارائه داده و ریزبرنامه های لازم برای سیکل برداشت و سیکل اجرای هر دستور را به طور مرتب دسته‌بندی و هر یک را تست کنید.

- دستورات محاسباتی
- دستورات منطقی
- دستورات جابجایی داده ها در RAM
- دستورات I/Oو انتقال

۲- (**حافظه نهان**) یکی از روش‌های افزایش سرعت اجرای دستورات استفاده از حافظه نهان می‌باشد. که مطابق آن بخشی از دستورات از حافظه برنامه‌نویسی به حافظه نهان منتقل شده و سپس اجرا می‌گردد. راه‌حلی برای اضافه کردن این روند به معماری کامپیوتر ۴ بیتی ارائه کرده و تست کنید (سازمان حافظه فصل ۱۲ کتاب معماری کامپیوتر موریس مانو).

۳- (**واحد کنترلی Hardwire**) چنانچه اشاره شد روش کنترلی پیاده‌سازی شده در این آزمایشگاه روش میکروپرامینگ بوده است. یک مدار کنترلی به روش Hardwire طراحی و پیاده‌سازی کنید.

۴- (**کنترل کننده وقفه اولویت‌دار**) کنترل و مدیریت وقفه ها یکی دیگر از وظایف پردازنده‌ها می‌باشد که در شرایط حضور چند ورودی انجام می‌شود. بخش مدیریت و کنترل وقفه‌ها را به معماری کامپیوتر ۴ بیتی اضافه کرده و تست کنید.

۵- (**پردازنده ۸ بیتی ساده**) باس داده معماری کامپیوتر خود را به ۸ بیت ارتقا داده و دستورات پایه را برای آن تنظیم و تست کنید.

۶- (**پردازنده ۴ بیتی جدید**) در لینک (http://engold.ui.ac.ir/~nikmehr/Lab_Manual.pdf) طراحی یک پردازنده چهار بیتی طراحی و به طور کامل شبیه‌سازی و تست شده است. این پردازنده را متناسب با بوردهای آزمایشگاه بهینه سازی و پیاده‌سازی کنید.

مراجع

[۱] دکتر حسین صباغیان، "دستورکار آزمایشگاه معماری کامپیوتر"، گروه کامپیوتر، دانشکده مهندسی دانشگاه کاشان، زمستان ۱۳۸۲.

[۲] دکتر علی بهلولی، "دستورکار آزمایشگاه معماری کامپیوتر"،دانشکده مهندسی کامپیوتر، دانشگاه اصفهان، بهار ۱۳۹۳.

[۳] مرتضی نوریان، " آزمایشگاه معماری بورد آموزشی AMAZ-FPGA"، دانشکده مهندسی برق و کامپیوتر، دانشگاه صنعتی خواجه نصیر طوسی، پاییز ۱۳۹۰.

[۴] موریس مانو، "معماری کامپیوتر"، ترجمه دکتر قدرت الله سپیدنام، ۱۳۸۲.

[۵] دیوید پترسون و جان هنسی، "طراحی ، معماری و سازمان کامپیوتر"،مترجمان : دکتر احسان ملکیان-دکتر علی الذاکرالحسینی، ۱۳۹۳.

[۶] دکتر وحید رشتچی، "دستورکار آزمایشگاه مدارهای منطقی"،گروه برق، دانشکده فنی مهندسی دانشگاه زنجان، ۱۳۹۱.

[۷] حاتم عبدلی، "دستورکار آزمایشگاه مدارهای منطقی"،گروه کامپیوتر، دانشکده مهندسی دانشگاه بو علی سینا همدان.

[۸] Daniel J. Tylavsky, "Digital Design for the Laboratory", Center Point Publishing, 2009.