

به نام خدا



آزمایشگاه معماری

آزمایش ششم: واحد محاسبه با امکان انتخاب ثبات مبدأ و مقصد

اعضای گروه:

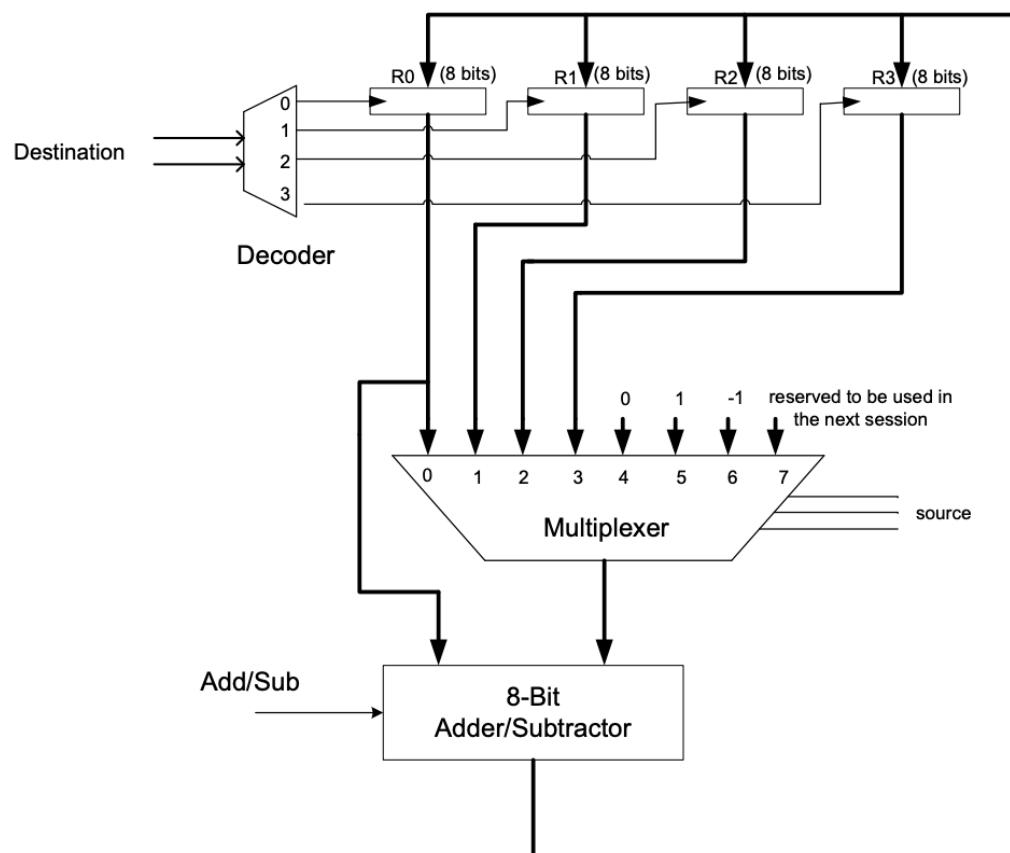
امیراردلان دهقانپور 401105901

رادین شاهدایی 401106096

باربد شهرآبادی 401106125

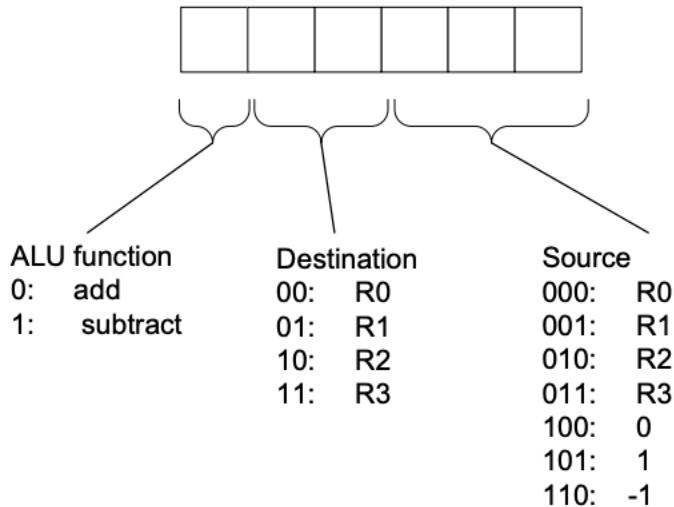
مقدمه:

در این آزمایش می خواهیم واحد محاسبات و مجموعه ثبات های عمومی یک کامپیوتر ساده را طراحی کنیم. در این معماری امکان انجام جمع و تفریق با انتخاب ثبات های مبدا و ثبات نگهدارنده فراهم می شود. چهار ثبات عمومی R0, R1, R2, R3 هشت بیتی هستند و یکی از عملوند های ALU به صورت ثابت شود. چهار ثبات های R0, R1, R2, R3 را مقدار ثابت ۰ و ۱ دارند. حاصلی که از ALU بدست می آید به یکی از ثبات های R0 تا R3 منتقل می شود. شماتیک مداری که باید آن را پیاده سازی کنیم به صورت زیر می باشد:



شکل ۱

و همچنین قالب شش بیتی که برای فرمان های این معماری وجود دارند به صورت زیر می باشد:

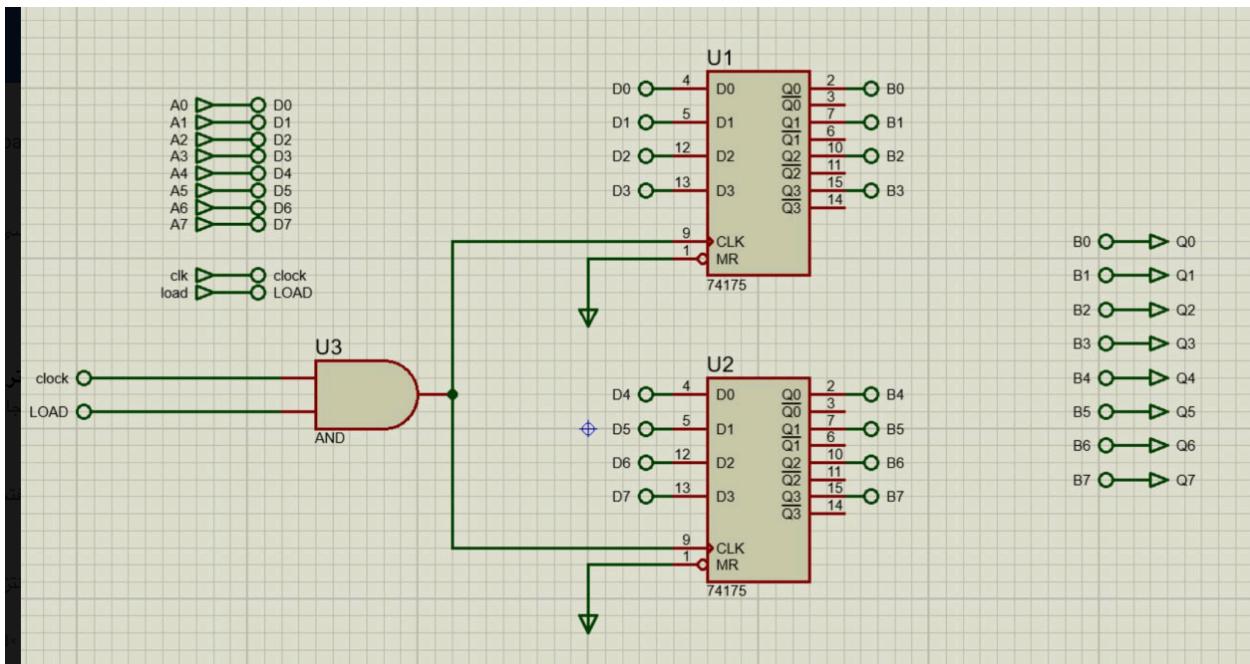


شکل ۲

همانطور که در شکل نیز مشخص است قالب دستورات به صورت شش بیتی هستند که تک بیت اول آن را بخش ALU function می دهد که نشان دهنده آن است که می خواهیم عمل جمع را انجام بدھیم و یا تفریق، سپس در دو بیت بعدی مشخص می کنیم که مقصد ما کدامیک از این ۴ ثبات خواهد بود و سپس در سه بیت آخر مشخص می کنیم که دومین سورسی که در ALU از آن استفاده می کنیم چه باشد که می تواند یکی از ثبات ها باشد و یا عدد ۰ و ۱ یا -۱ باشد، پس می توانیم به این صورت قالبی که برای دستورات داریم را ایجاد کنیم، حال به سراغ پیاده سازی این معماری در پروتیوس می رویم:

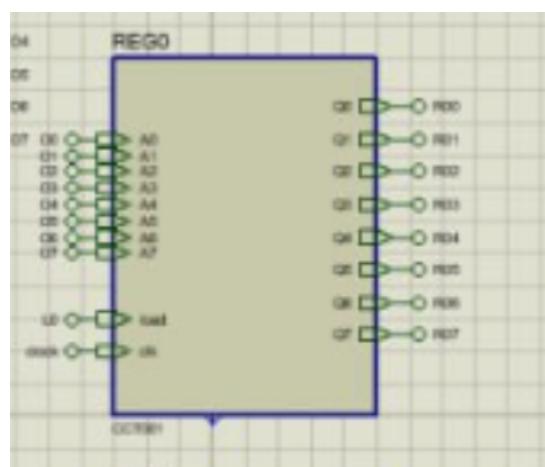
پیاده سازی:

در این بخش در ابتدا رجیسترها را پیاده سازی می کنیم برای این کار یک ماژول رجیستر در پروتیوس طراحی می کنیم که به صورتی می باشد که با سیگنال کلک می توان محتوا را در آن منتقل کرد و به نوعی یک فلیپ فلاپ ۸ بیتی می باشد، این ماژول رجیستر به صورت زیر پیاده سازی شده است:



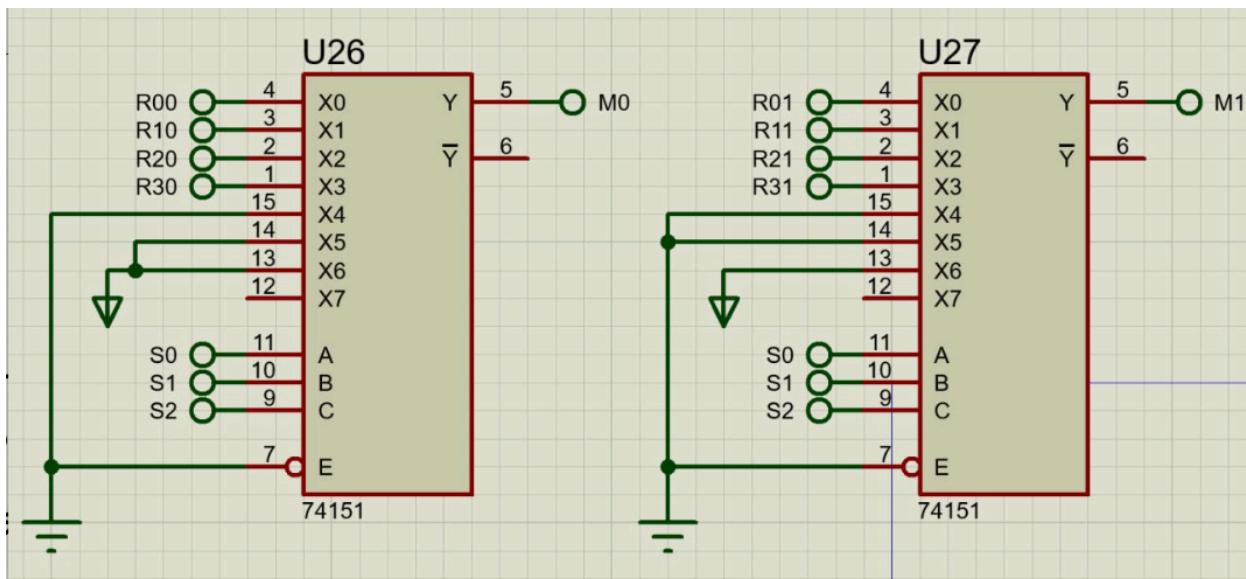
شکل ۳

همانطور که مشخص است دارای ورودی های A0 تا A7 می باشد که آنها به دو فلیپ فلاب ۴ بیتی که وجود دارند متصل شده اند و با استفاده از کلاک و هنگامی که سیگنال load برابر با یک باشد آنها را به عنوان خروجی منتقل می کند. پس این بخش که شامل پیاده سازی خود رجیستر است به صورت زیر پیاده سازی شده است و اگر بخواهیم نگاه کلی تری به آن داشته باشیم به صورت زیر است:



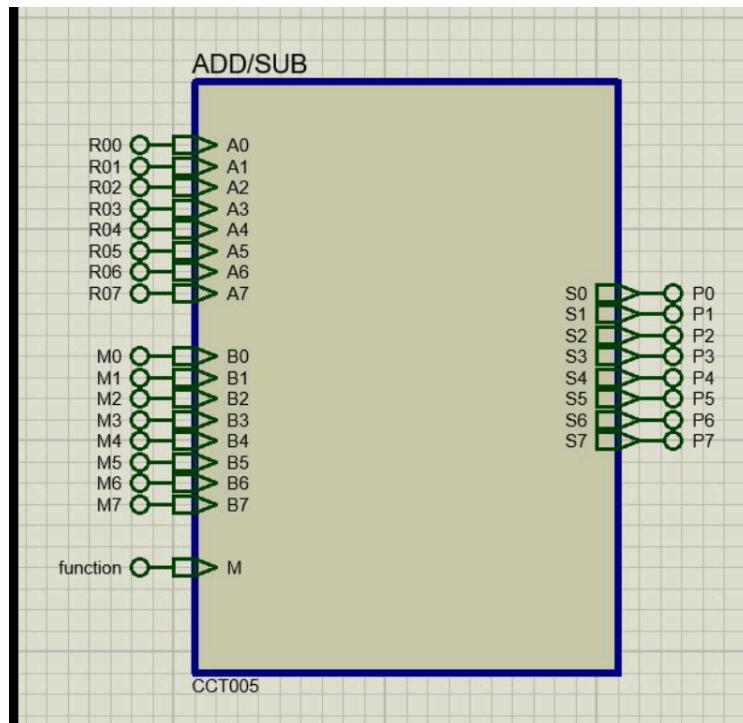
شکل ۴

که در بالا ثبات R0 نشان داده است، حال بخش دیگری که آن را پیاده سازی کرده ایم انتخاب بین اینکه source دوم در ALU از چه بخشی وارد شود است که برای این کار نیاز به این داریم که از ۸ مالتی پلکسر استفاده کنیم که هر کدام برای یک بیت است و هر یک از این ۸ مالتی پلکسر، یک مالتی پلکسر ۸ ۱ هستند که بنابراین نیاز به ۳ بیت select دارند که این ۳ بیت همان S0, S1, S2 هستند که به عنوان ۳ بیت در قالب دستور وارد می شوند، این مالتی پلکسراها طبق چیزی که صورت سوال خواسته به صورت زیر هستند:



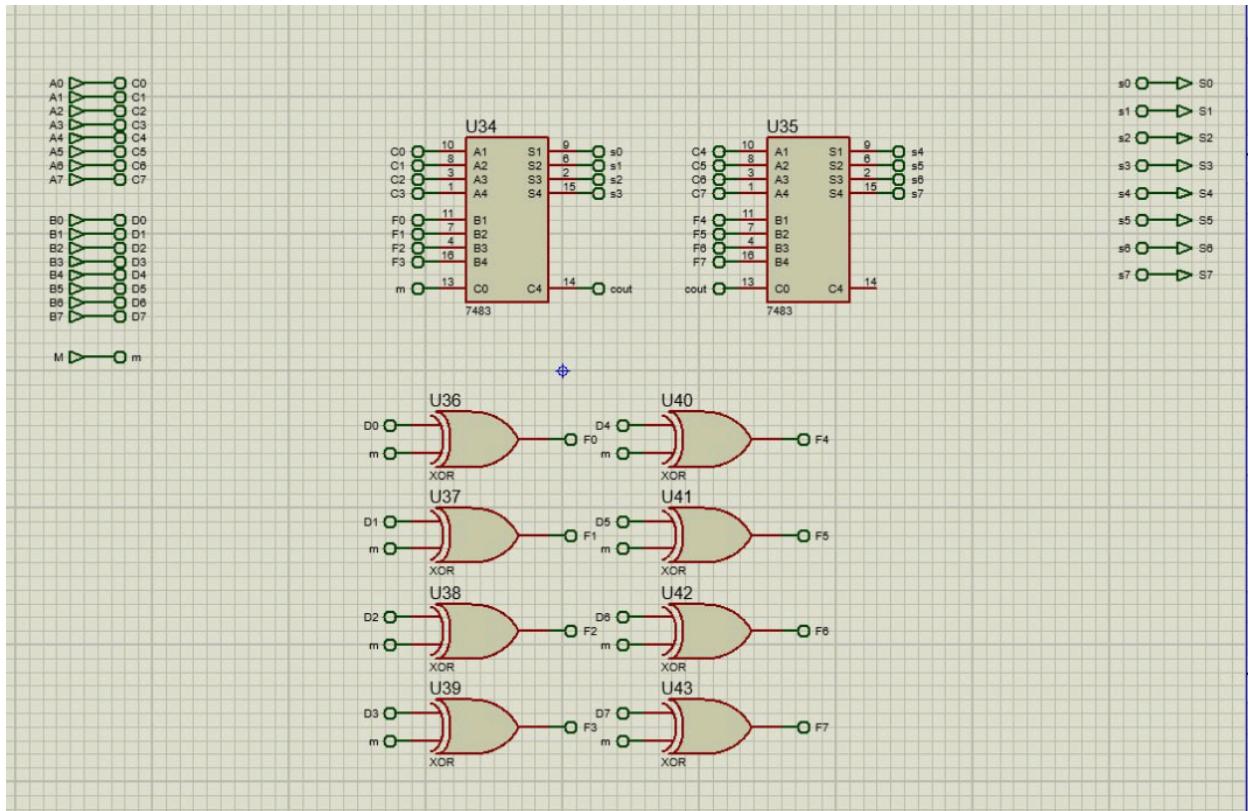
شکل ۰

همانطور که می بینیم با استفاده از این ۸ بیت انتخاب می کنیم که از ثبات های R0 تا R3 استفاده کنیم و یا اینکه عدد ۰ و یا ۱ و یا -۱ را با آن جمع کنیم، برای جمع کردن صفر که همه بیت ها را به زمین وصل می کنیم برای ۱- همه بیت هارا یک می گذاریم که نشان دهنده ۱- است و برای عدد ۱ نیز تنها بیت اول آن را ۱ می گذاریم و بقیه صفر هستند، این ۸ بیت که M0 تا M7 هستند به عنوان ورودی وارد ALU می شوند که این مأذول به صورت شکل زیر است:



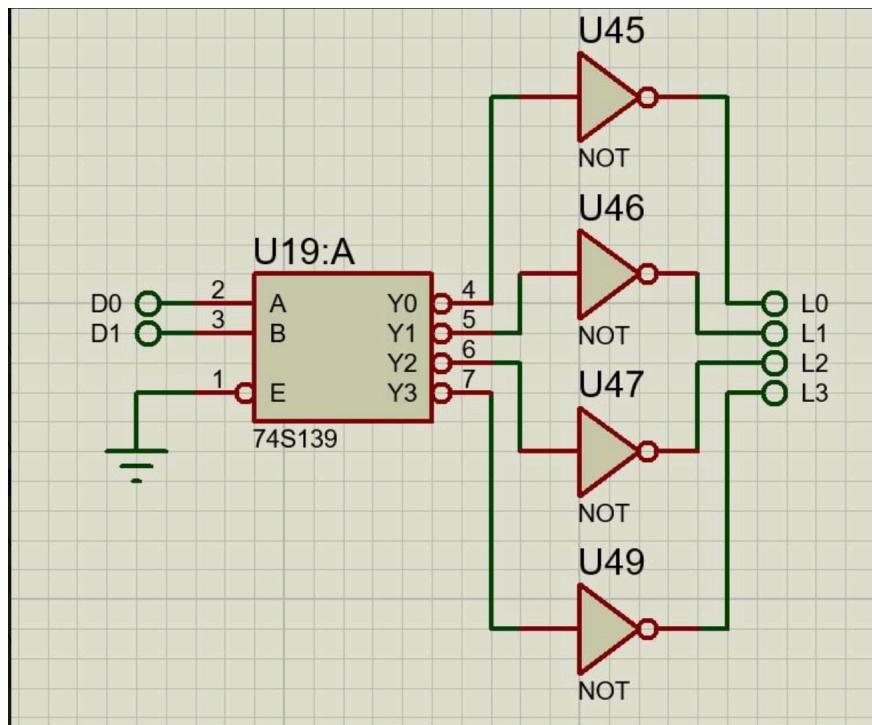
شکل ۶

همانطور که نشان داده شده است همواره به صورت ثابت یکی از عملوندهای این ALU ثبات صفر و بخش دیگری که دارد M0 تا M7 است که از مالتی پلکسرهای بالا می آیند. خروجی های این واحد نیز P0 تا P7 هستند. داخل این ALU نیز به این صورت می باشد که در آن بیت m به این خاطر وجود دارد که اگر بخواهیم جمع کنیم تغییری ایجاد نکند و جمع را انجام دهیم در غیر این صورت مکمل ۲ عدد دوم برای تفیریک در آن وارد می شود و بیت carry در جمع کننده ها نیز یک می باشند، دو full adder ؟ بیتی که وجود دارند نیز کار جمع کردن و یا تفیریک را برای ما انجام می دهند، تصویری از شمای داخلی این بخش از مدار نیز در صفحه بعد آمده است:

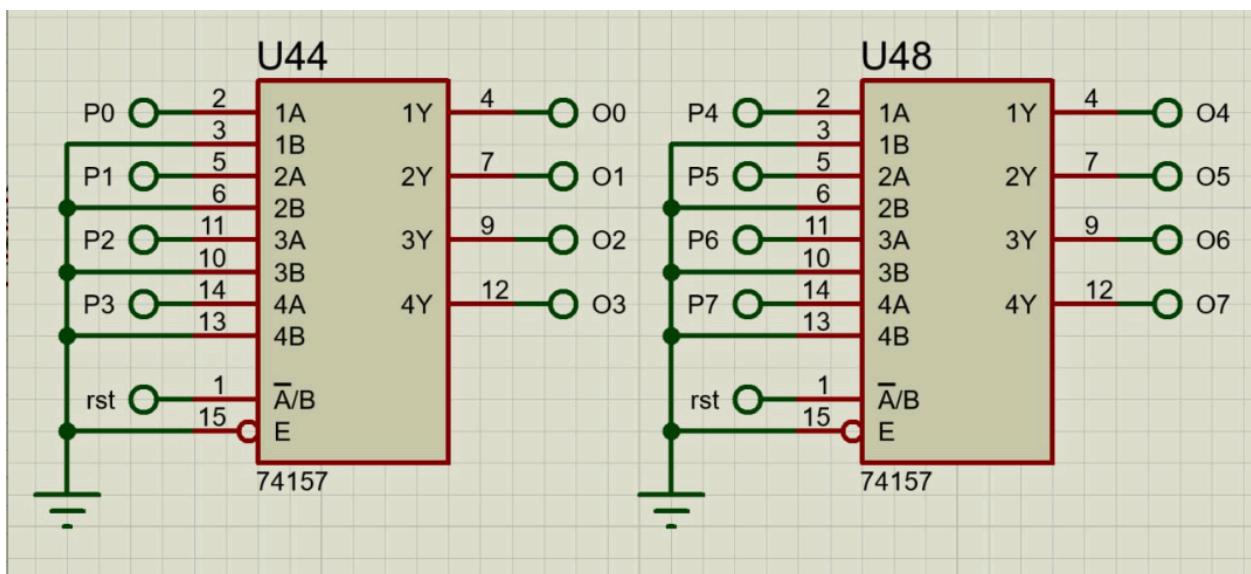


شکل ۷

بخش دیگری که در قالب دستورات وجود دارند و باید آن را پیاده سازی کنیم بخش انتخاب مقصد برای ریختن حاصل در آن است که برای این کار از یک decoder استفاده می کنیم که ۲ ورودی و ۴ بیت برای خروجی دارد که به علت active low not بودن آنها را کرده ایم و سپس هر یک از این بیت ها را تحت عنوان load به رجیسترهای که داریم وصل می کنیم تا بتوانیم با استفاده از آن خروجی واحد محاسبات که تولید شده است را در یکی از رجیسترها بریزیم و همچنین بخش دیگر مدار که به پیاده سازی کرده ایم یک مالتی پلکسر است که با توجه به اینکه سیگنال reset صفر است و یا یک خروجی آن که داخل آن رجیستری که می خواهیم می شود را مشخص می کند، تصویرهایی از پیاده سازی این دو بخش نیز در صفحه بعد آمده اند.

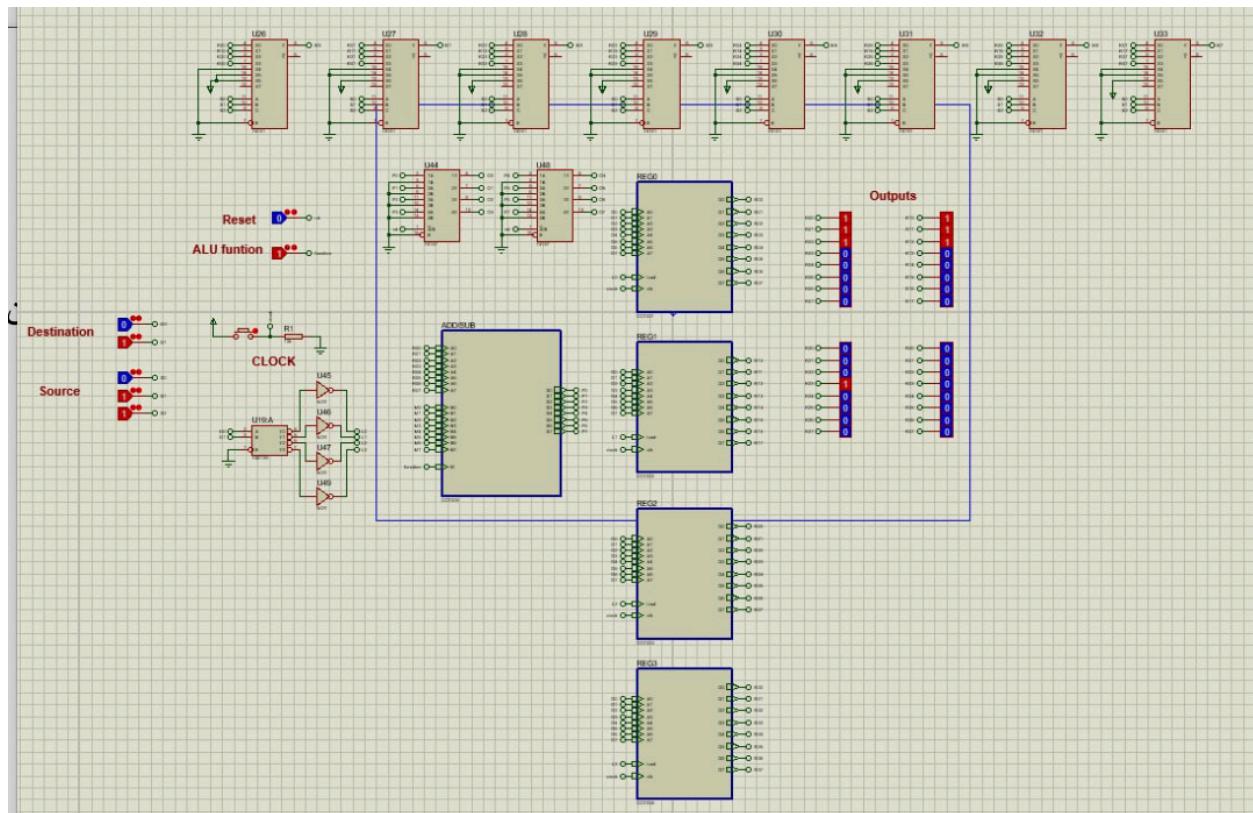


شکل ۸



شکل ۹

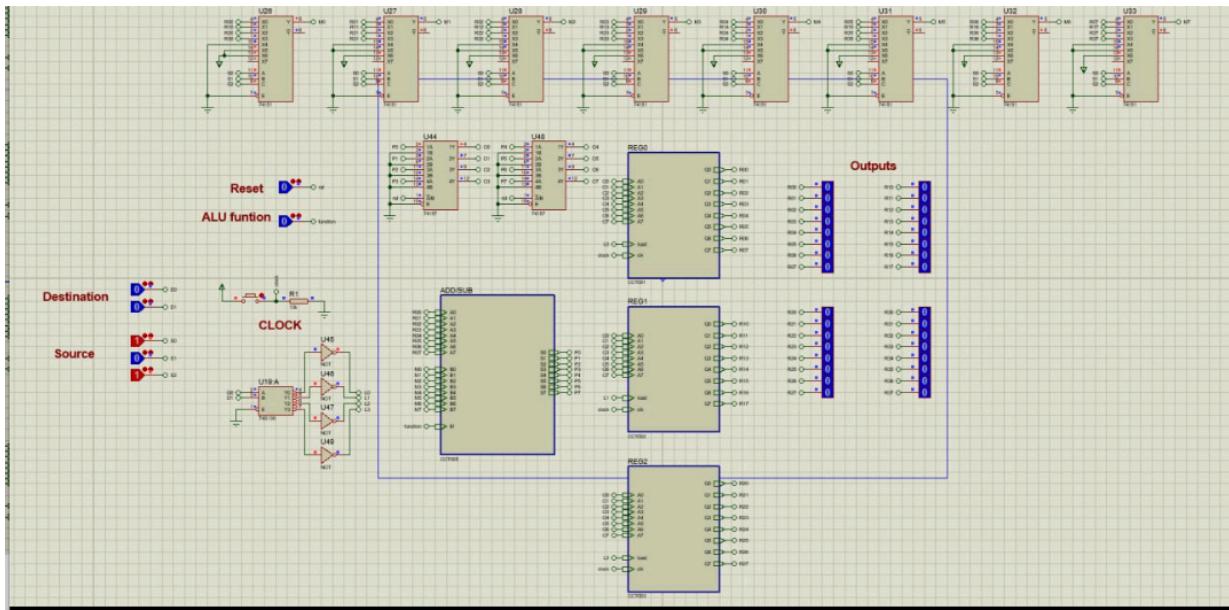
این بخش ها بخش هایی بودند که برای پیاده سازی این معماری به آن ها نیاز داشتیم و آنها را شرح داده ایم
شکل زیر یک نمای کلی از مدار می باشد:



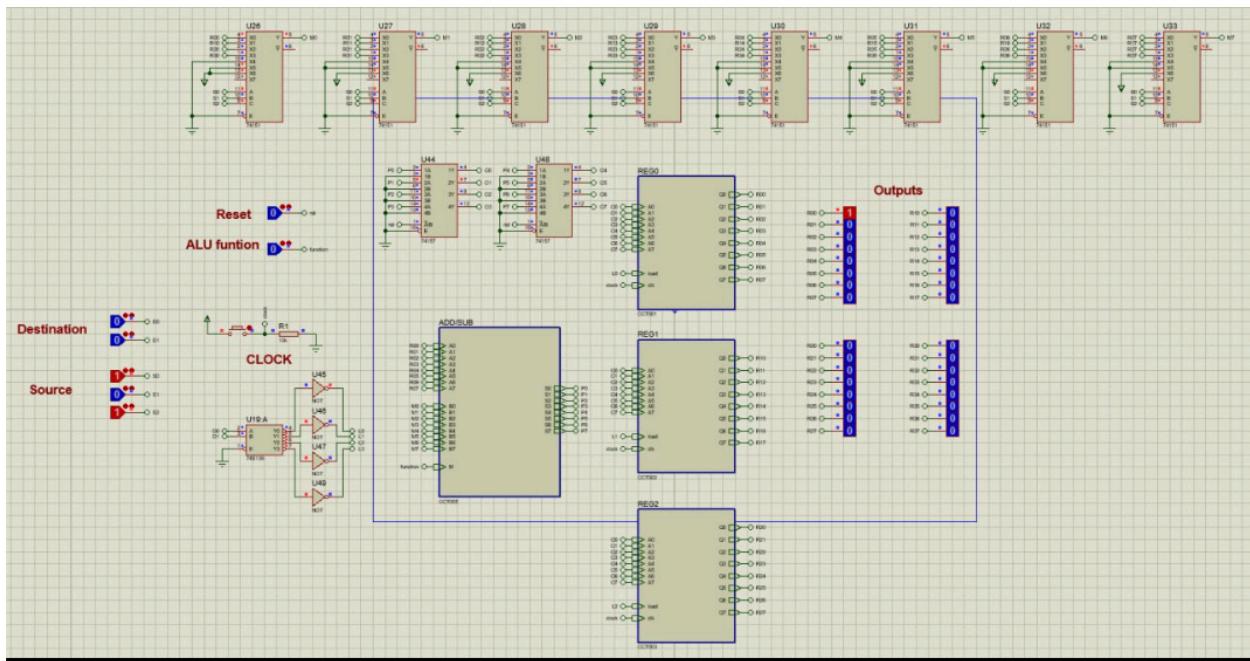
شکل ۱۰

تست مدار:

- در این بخش به سراغ تست کردن حالت هایی از مدار می رویم تا عملکرد مدار را بررسی کنیم:
- در این تست ثبات R0 که در ابتدا در آن صفر است را با یک جمع می کنیم و بعد از کلاک زدن عدد یک در آن ریخته می شود که عکس های آن در صفحه بعد پیوست شده اند:

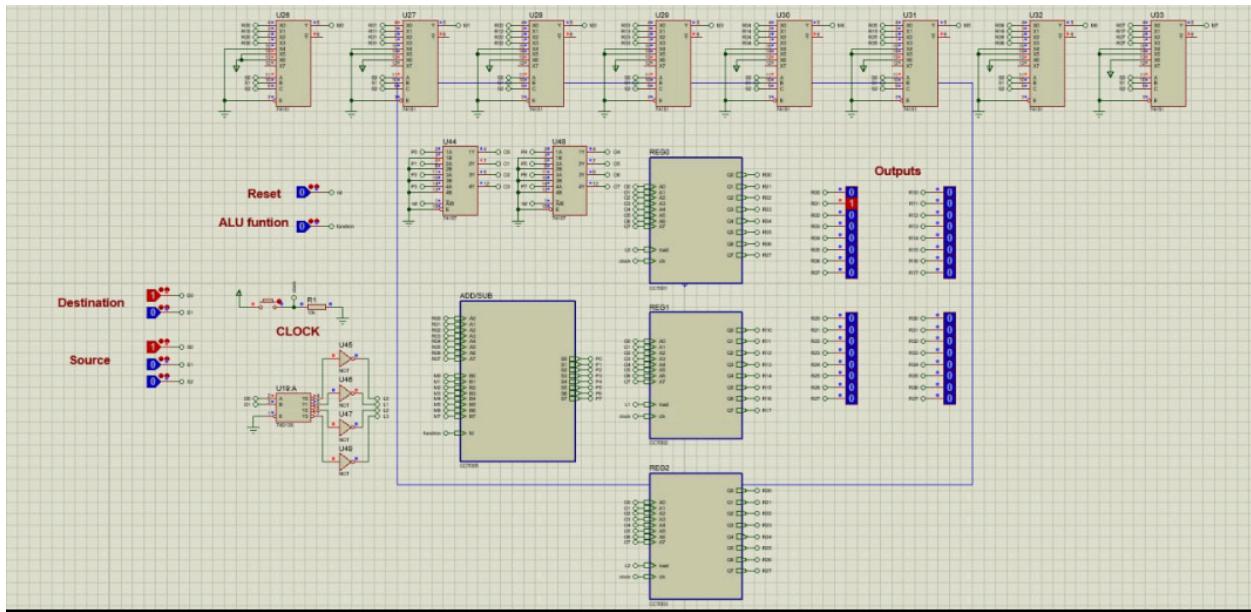


شکل ۱۱

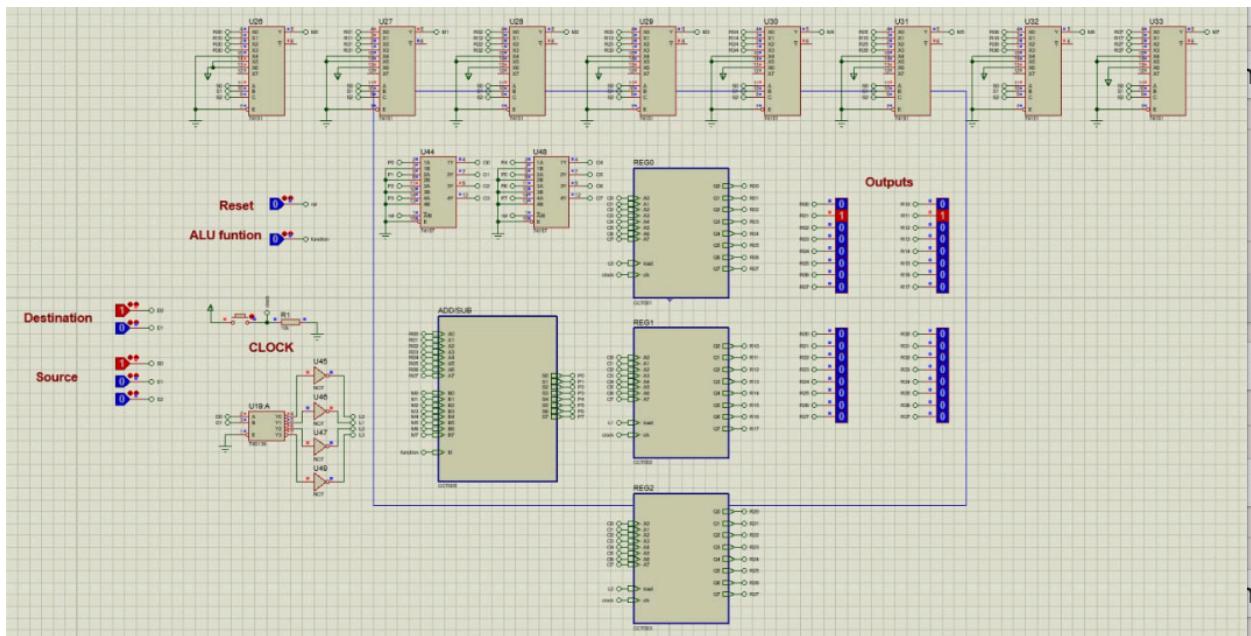


شکل ۱۲

- اضافه کردن R0 و R1 و ریختن حاصل در R1



شکل ۱۳



شکل ۱۴

همانطور که در ابتدای آزمایش نیز شرح دادیم هدف از این آزمایش ساخت یک واحد محاسبه با امکان انتخاب مبدا و مقصد بود که این کار را انجام دادیم.