به نام خدا

درس : آزمایشگاه معماری کامپیوتر

استاد : دکتر سربازی

آزمایش دوم

اعضای گروه :

سید آرین علوی رضوی راوری ۴۰۰۱۰۹۷۹۲

سیده فاطمه موسوی ۴۰۰۱۰۵۲۵۲

محمدعرفان سلیما ۴۰۰۱۰۵۰۱۴

آرش ضیایی رازبان ۴۰۰۱۰۵۱۰۹

تیر ۱۴۰۲

پروتئوس آزمایش :

هدف آزمایش :

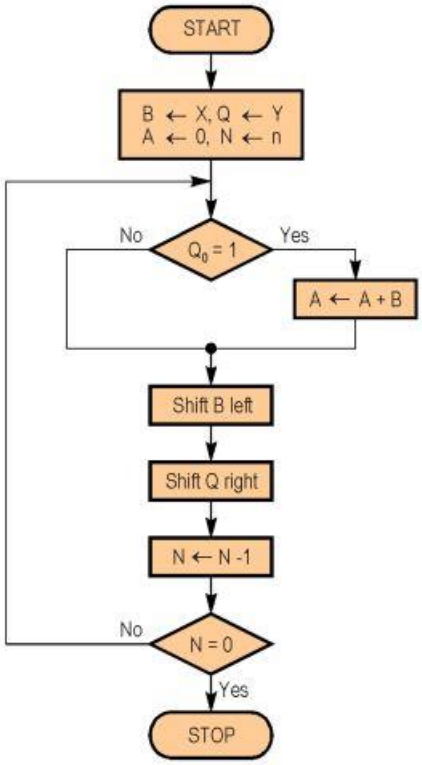
در این آزمایش می خواهیم با استفاده از نرم افزار پروتیوس یک ضرب کننده ی چهار بیتی طراحی کنیم. به این صورت که این ضرب کننده، دو عدد ۴بیتی را به عنوان ورودی می گیرد و سپس حاصل ضرب(۸ بیت) آن دو را به عنوان خروجی نمایش می دهد.

شرح آزمایش :

در این آزمایش برای ضرب دو عدد ۴ بیتی از الگوریتم **Add&Shift** استفاده می کنیم. در هر مرحله از این الگوریتم داریم:

کم ارزش ترین بیت مضروب فیه (در اینجا **B**) را در مضروب ضرب می کنیم. حاصل ضرب را با حاصل فعلی جمع می کنیم. سپس، عدد **B** را یک بیت به راست شیفت می دهیم. همچنین، حاصل کنونی را به سمت چپ شیفت می دهیم و بار دیگر این علمیات را تکرار می کنیم. این مراحل تا زمانی که تمام بیت های عدد **B** را در مضروب ضرب نکرده ایم، ادامه می دهیم.

با توجه به اینکه حاصل ضرب دو عدد چهار بیتی، عددی ۸ بیتی است؛ بنابراین، برای مضروب از رجیستر ۸ بیتی استفاده می کنیم(لازم به ذکر است که در اولین مرحله از الگوریتم در رجیستر ۸ بیتی عدد **A** را به عنوان مضروب اولیه لود می کنیم).



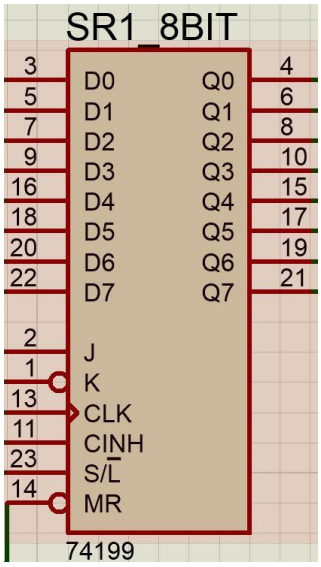
Shift&Add Flowchart

طراحی و پیاده سازی :

۱. بررسی طراحی:

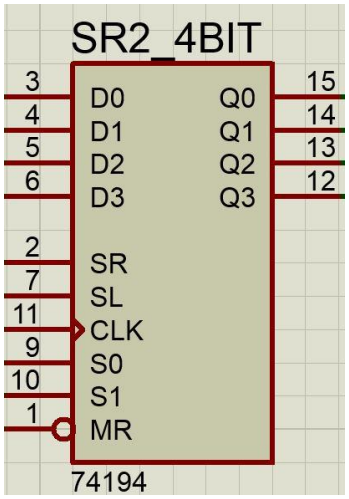
حال به بررسی قطعات متناظر برای طراحی و پیاده سازی الگوریتم مورد نظر می پردازیم:

۱. 8-bit Shift Register(74199)



از این قطعه برای شیفت دادن مضروب استفاده می کنیم. با توجه به اینکه حاصل نهایی ۸ بیت است، از نوع ۸ بیتی این رجیستر استفاده می کنیم. برای Load - Shift باید مقادیر J-K- CINH صفر باشند. برای تعیین عملیات میان Load و Shift از سیگنال S/L استفاده می کنیم. همچنین، با صفر شدن سیگنال MR خروجی این شیفت رجیستر صفر خواهد بود.

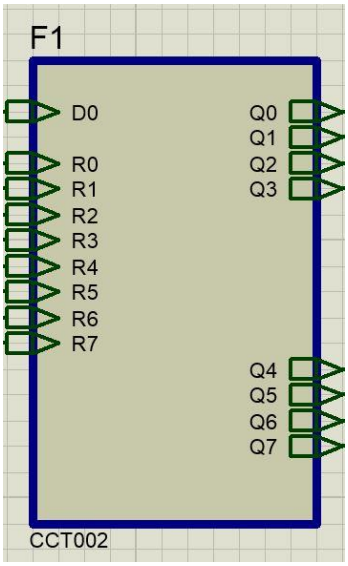
۲. 4-bit Shift Register(74194)

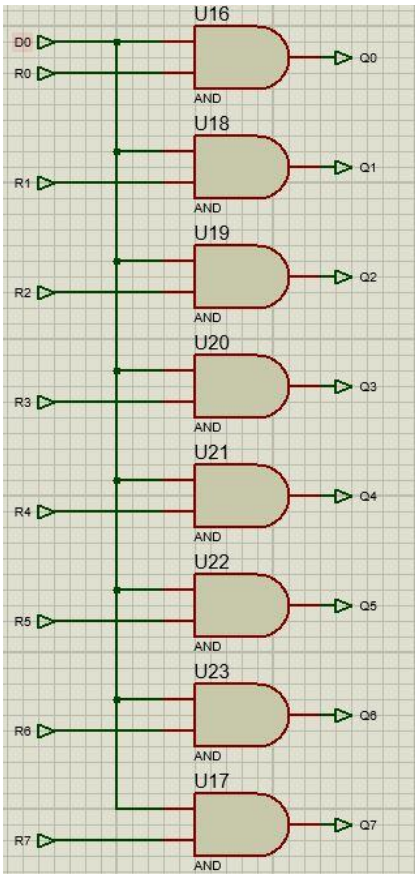


از این قطعه برای شیفت دادن مضروب فیه(B) استفاده می کنیم. با توجه به اینکه عدد B چهار بیت است، از نوع چهار بیتی این رجیستر استفاده می کنیم. برای عملیات Load , Shift مقدار S1 باید یک باشد. برای تعیین شیفت راست یا چپ در این شیفت رجیستر از سیگنال های SL, SR استفاده می کنیم.

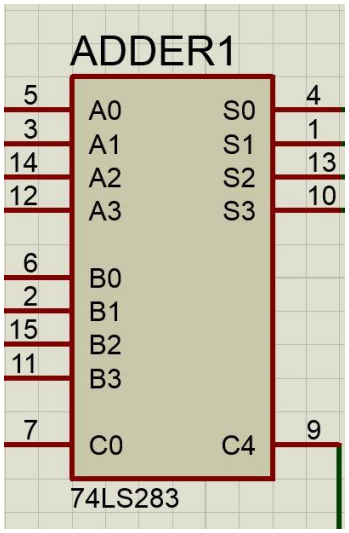
۳. Custom And(F1)

برای پیاده سازی ضرب کم ارزش ترین بیت مضروب فیه در مضروب، یک ماژول به نام **F1** طراحی کرده ایم. در این ماژول از هشت گیت **And** استفاده شده است. در این ماژول، کم ارزش ترین بیت مضروب فیه را با تمام بیت های مضروب **And** می کنیم. خروجی این ماژول نیز شامل هشت بیت است که حاصل ضرب کم ارزش ترین بیت مضروب فیه را در مضروب نمایش می دهد.



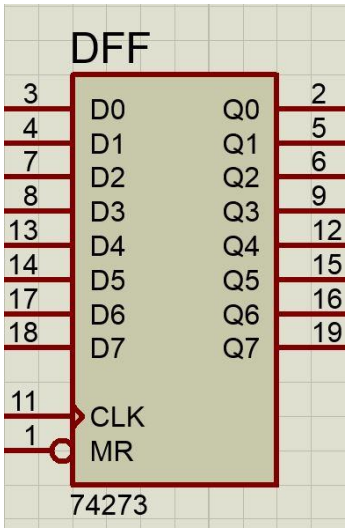


۴. 4-bit Adder(74LS283)



در هر مرحله از الگوریتم، باید حاصل ضرب کم ارزش ترین بیت شیفت داده شده ی عدد B در مضروب را با حاصل فعلی جمع کنیم. به این ترتیب به جمع کننده نیاز داریم. با توجه به اینکه حاصل نهایی، ۸ بیت است؛ در نتیجه، از دو جمع کننده ی چهار بیتی استفاده می کنیم.

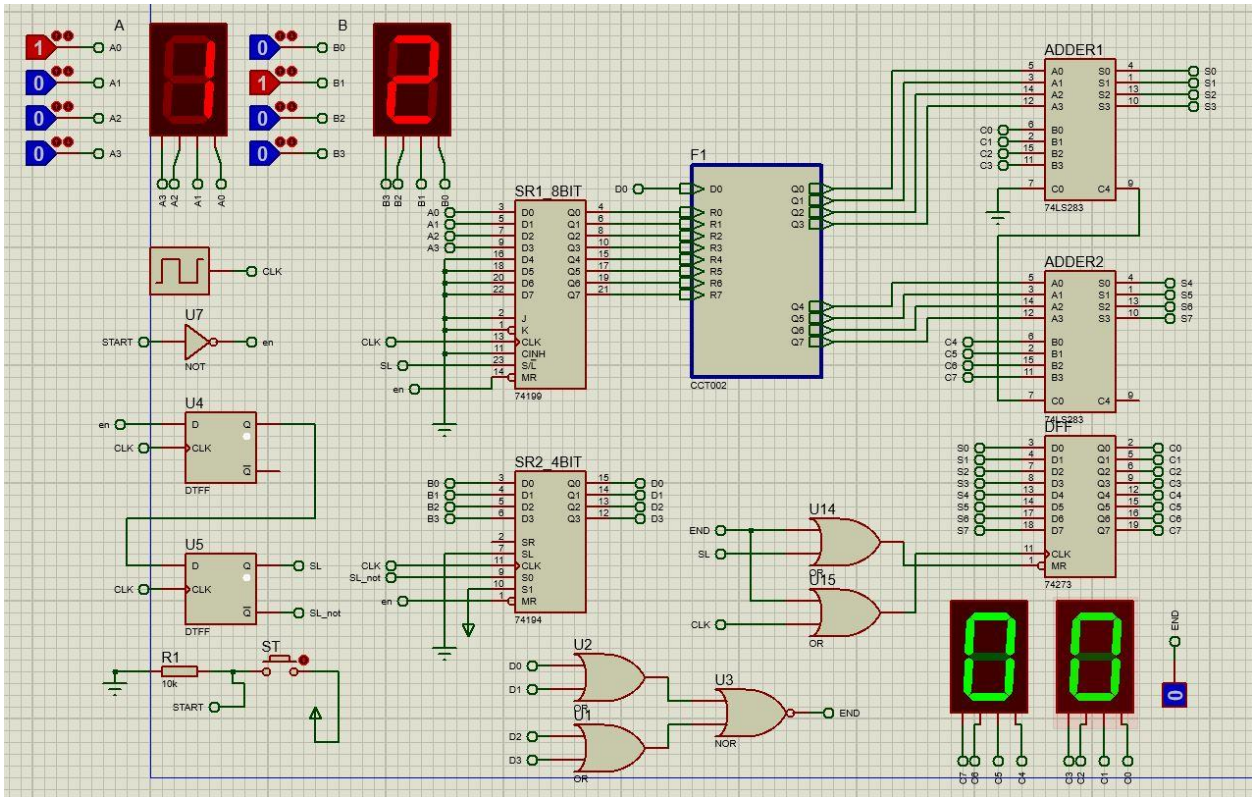
۵. 8-bit DFF(74273)



از D-FlipFlop برای ذخیره کردن حاصل قبلی مدار استفاده می کنیم. چرا که بایدحاصل کنونی ضرب را با حاصل قبلی جمع کنیم. ورودی های DFF را از خروجی Adder ها می گیریم.

۲. نحوه ی اتصال:

نحوه ی اتصالات و شکل کلی مدار را می توانید در تصویر زیر مشاهده کنید:



نکاتی در خصوص پیاده سازی:

۱. همانطور که در شکل بالا مشاهده می کنید، ورودی **start** را با استفاده از یک **button** در نظر گرفتیم که با فشردن آن، این سیگنال یک می شود.

۲. با توجه به اینکه خروجی **End** زمانی یک می شود که عدد مضروب فیه صفر شود، پس آن را برابر با **Or** بیت های مضورب فیه قرار می دهیم.

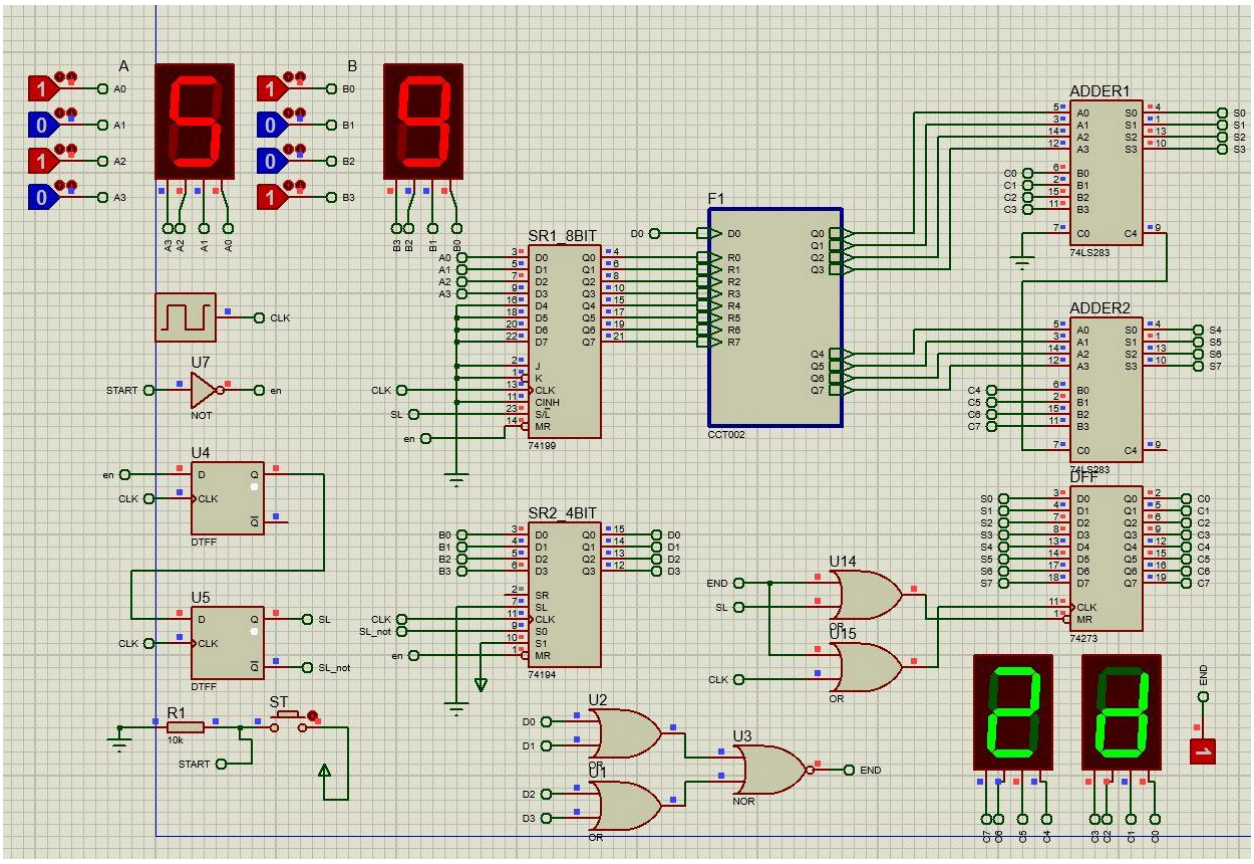
۳. با توجه به اینکه ابتدا باید اعداد مورد نظر در شیفت رجیستر ها **Load** شوند و سپس ضرب روی آن ها صورت گیرد، سیگنال **Start** را به ورودی یک **D-FF** می دهیم. خروجی آن را نیز به ورودی **D-FF** دیگر می دهیم، تا بعد از دو کلاک ضرب آغاز شود(در کلاک اول اعداد **Load** می شوند و در کلاک بعدی سیگنال **SL** مشخص می شود و ضرب صورت می گیرید).

۴. هنگامی سیگنال **End** یک می شود(یعنی ضرب کامل انجام شده است) باید **MR** مربوط به **8-bit DFF** را یک کنیم و کلاک آن نیز یک شود تا حاصل داخل آن باقی بماند(حالت **Hold**). به این منظور، کلاک این قطعه را برابر با **CLK or End** و **MR** را برابر با **SL or End** قرار می دهیم.

۵. برای نمایش اعداد نیز از **7-segment** استفاده کرده ایم. با توجه به اینکه خروجی حاصل ضرب ۸ بیت است، برای نمایش آن از دو **7-segment** استفاده کرده ایم، که آن را در مبنای ۱۶ نمایش می دهند.

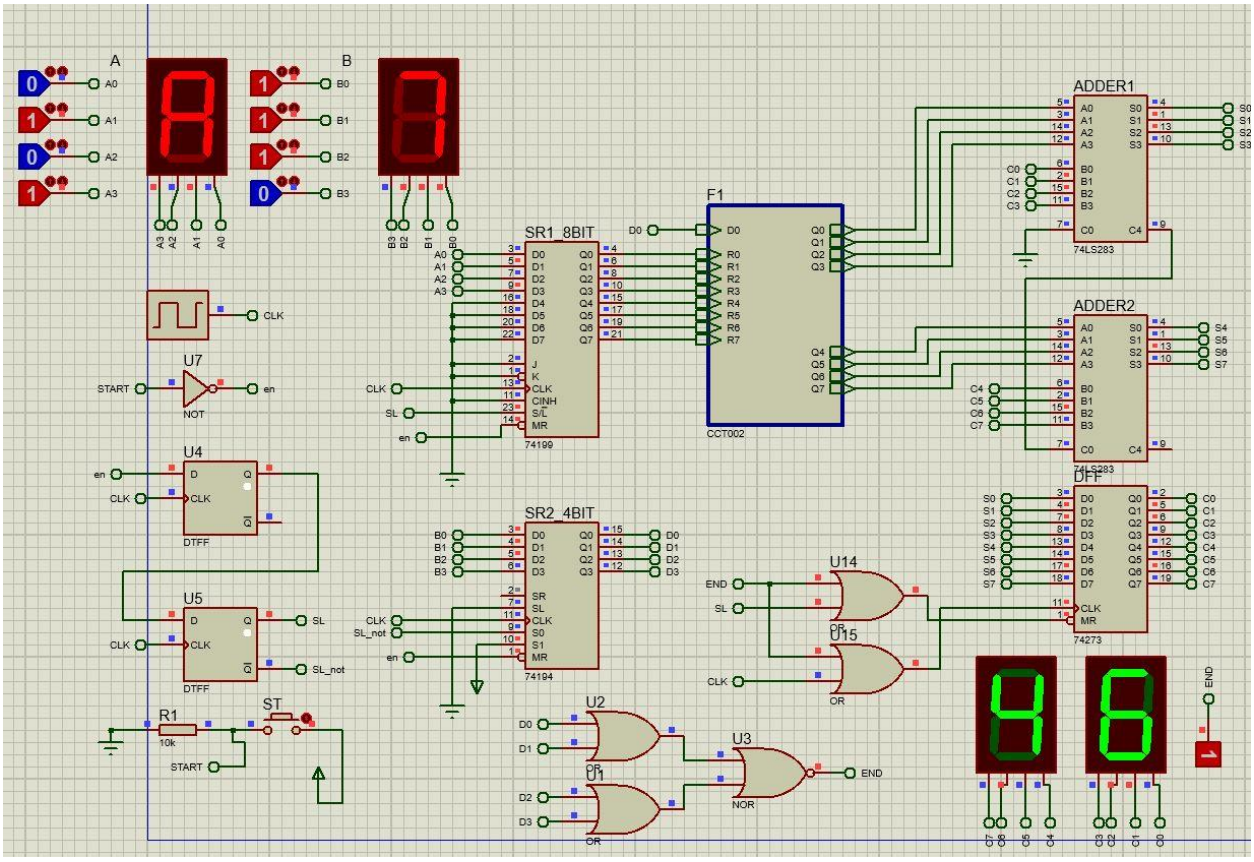
بررسی کارکرد مدار:

در این قسمت می خواهیم درستی مدار خود را آزمایش کنیم. به این منظور به ورودی ها اعداد زیر را می دهیم: **A = 0101 (5)** و **B = 1001 (9)**. همانطور که مشخص است باید چواب نهایی به صورت $(2D)_{16}$ = 45 باشد. **Start button** را به مدت دو کلاک نگه می داریم، تا اعداد مورد نظر لود شوند و ضرب شروع شود. نتایج را در شکل زیر مشاهده می کنید:



همانطور که مشاهده می شود، نتیجه ی مدار درست است.

تست دیگری را نیز در شکل زیر مشاهده می کنید:(A = 1010 (10) و B = 0111 (7))



همانطور که مشاهده می شود، جواب نهایی برابر با $(46)_{16}$ شده است که برابر است با 70.