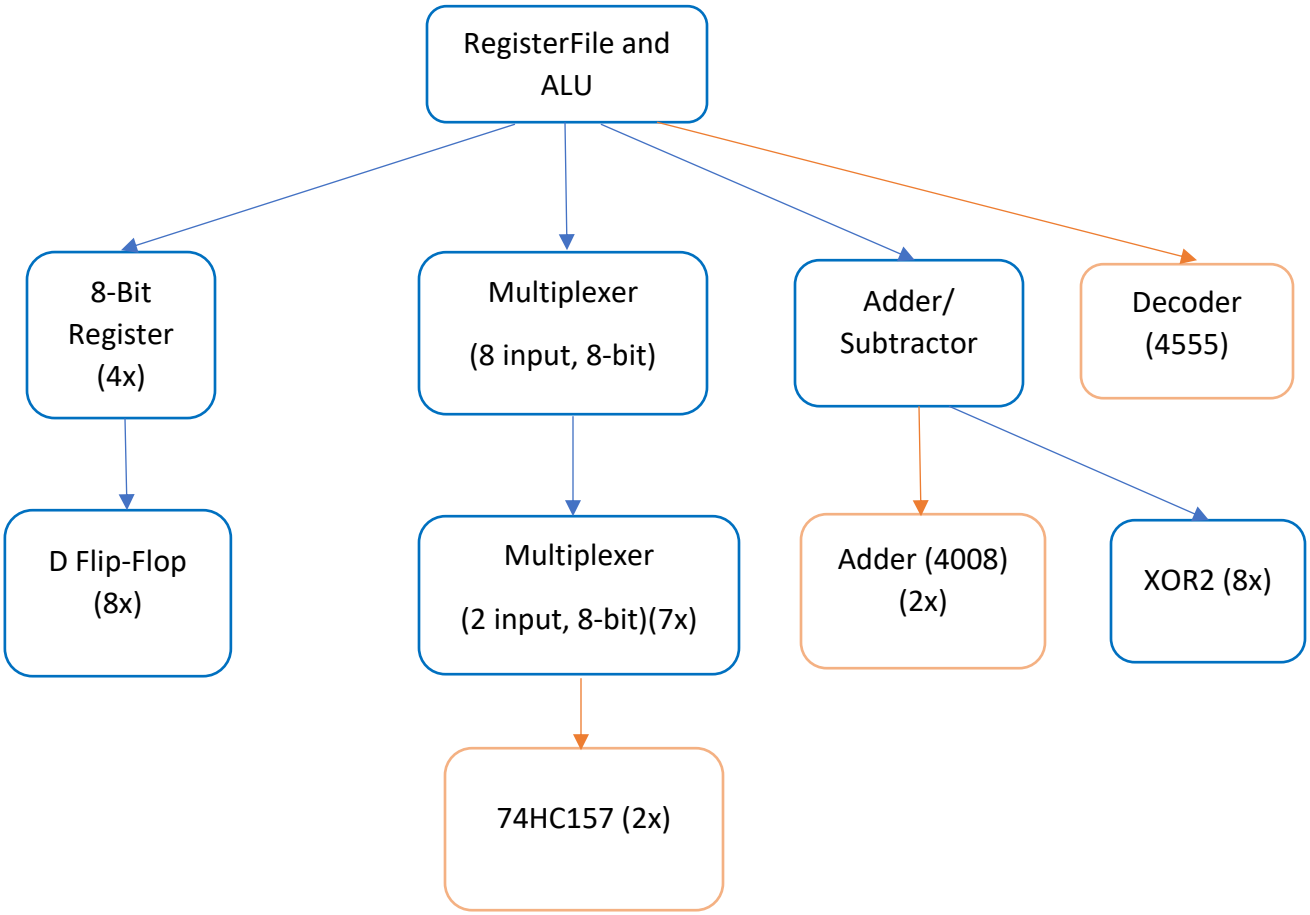
آزمایش پنجم: واحد محاسبه و بانک ثبات پردازنده

چکیده

در این آزمایش قصد داریم یک بانک ثبات چهار ثباته به همراه یک واحد محاسبات ساده که امکان جمع و تفریق دارد را پیاده‌کنیم. همچنین امکانات لازم جهت آدرس‌دهی ثبات‌ها و تعیین مسیر اطلاعات نیز در این آزمایش پیاده‌سازی شده است.

فهرست

سلسله‌مراتب طراحی صورت گرفته



همچنین برای دریافت ورودی‌ها از LOGICSTATE و برای مشاهده‌ی خروجی‌ها از LOGICPROBE استفاده شده است.

ورودی‌ها و خروجی‌های مدار

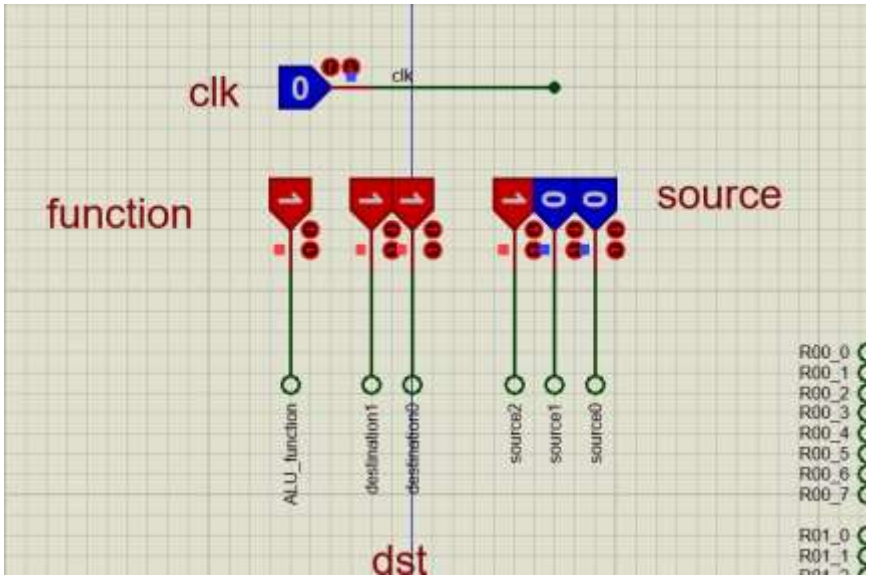
ورودی‌ها

clk: ورودی پالس ساعت مدار است که برای رجیسترها استفاده شده است.

function: برای انتخاب میان جمع یا تفریق است. اگر صفر باشد عملیات جمع است و در غیر این صورت تفریق.

dst (دو بیتی): نشان دهنده مقصد است یعنی ثباتی که حاصل وارد آن خواهد شد.

source (سه بیتی): نشان دهنده مبدا عملوند دوم عملیات است (عملوند اول همواره R0 است).



خروجی‌ها

در این مدار در واقع خروجی‌ای نداریم، ولی برای تحلیل صحت عملکرد مدار، محتوای تمامی ثبات‌ها به عنوان خروجی فرض شده‌اند و به آنها LOGICPROBE وصل شده تا بتوان مقدار آنها را بررسی کرد.

بنابراین خروجی مدار چهار عدد هشت بیتی R0,R1,R2,R3 است.

مقدمه طراحی صورت گرفته

نحوه طراحی مدار دقیقا براساس شماتیک گزارش آزمایش است. مدار کلی از ۴ بخش تشکیل شده است.

۱ -ثبات ها

۲-دیکودر

۳-مولتی پلکسر

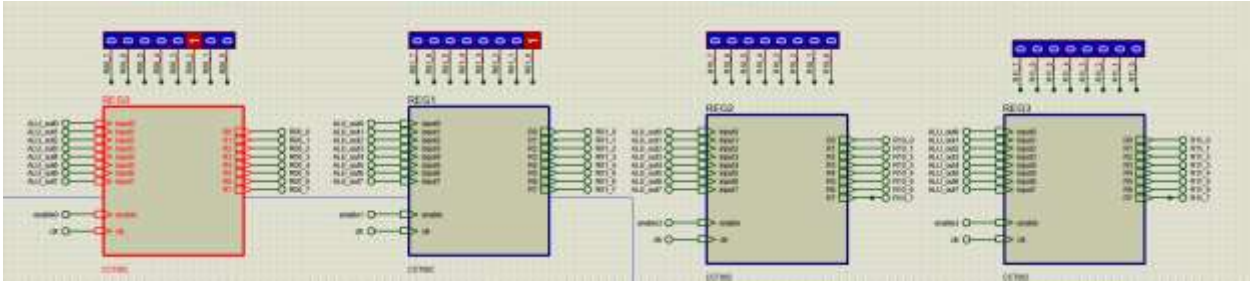
۴-جمع کننده

در ۴ بخش زیر به تفضیل درباره نحوه پیاده سازی هر کدام از این بخش ها می پردازیم.

دقت داریم که در شماتیک قرار داده شده در دستورکار آزمایش، خروجی دیکودر انتخاب‌گر destination متصل به کلاک ثبات‌هاست. برای پرهیز از مشکلات ناشی از gated clock و ایجاد شدن حالاتی که هازارد رخ می‌دهد (مانند ایجاد هازارد داده‌ای در زمان‌هایی که مقدار دیکودر تغییر کند یا ...) و برای پرهیز از آسنکرون شدن این قسمت از مدار، در طراحی انجام شده، ثبات‌هایی با پایه‌ی enable طراحی شده‌اند و یک ورودی کلاک برای

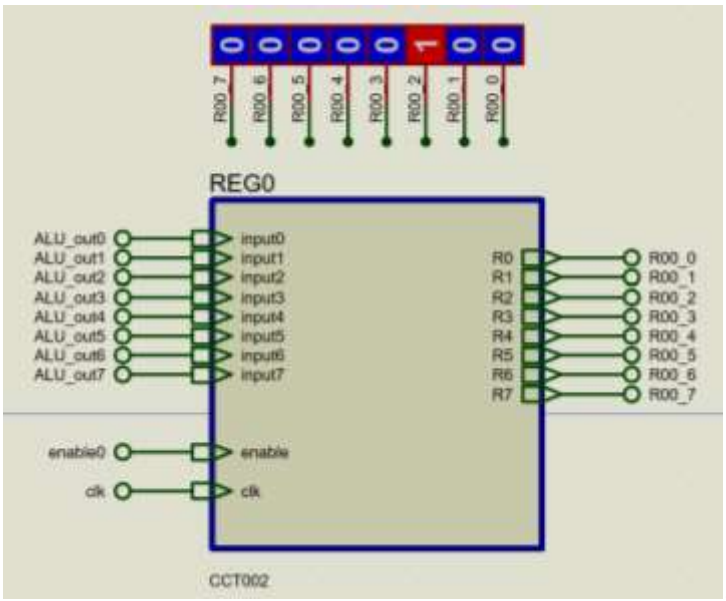
مدار در نظر گرفته شده است، تا طراحی صورت گرفته مشابه بانک ثبات‌های پردازنده‌هایی که در درس معماری کامپیوتر با آنها آشنا شده‌ایم باشد.

بخش ثبات ها



نیاز به یک ثبات ۸ بیتی با سیگنال enable داریم. بدلیل اینکه از شلوغی مدار خودداری کنیم ثبات ۸ بیتی را به صورت یک ماژول طراحی می کنیم.

در تصویر زیر نمای کلی این ماژول را می بینید.

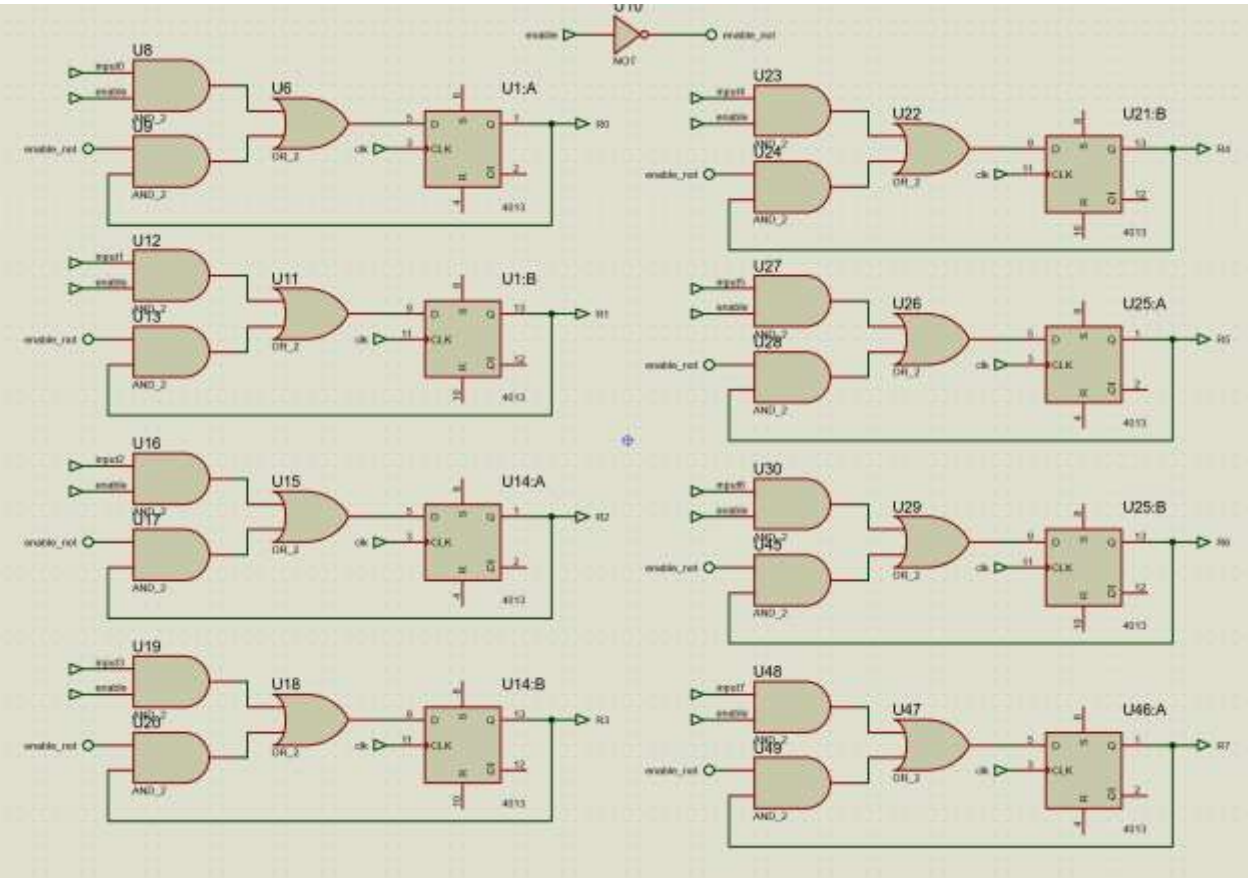


.

خروجی جمع کننده به ورودی هر کدام از چهار ثبات وصل است دقت کنید چونکه ثبات ها سیگنال enable دارند با کنترل این سیگنال در بخش دیکودر می توان تعیین کرد که خروجی در کدام ثبات نوشته شود.

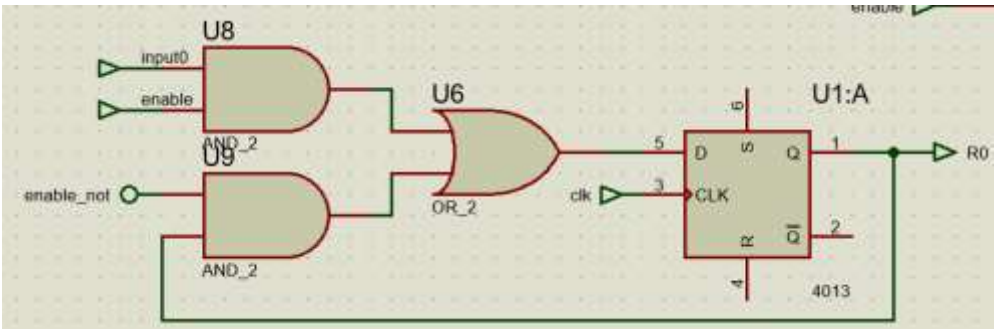
برای طراحی خود ماژول ثبات از تراشه ۴۰۱۳ استفاده شده است که یک D flipflop ساده است. چون ثبات هایمان ۸ بیتی است به ۸ تا از این تراشه ها نیاز داریم. این نوع تراشه سیگنال enable ندارد به همین دلیل باید قبل از خروجی آن یک مدار ترکیبی برای تولید سیگنال enable بگذاریم. این مدار اینکونه عمل می کند که اگر enable فعال باشد ورودی فعلی را در ورودی فلیپ فلاپ می گذارد و اگر فعال نباشد خروجی فلیپ فلاپ را در ورودی آن می گذارد.

در تصویر زیر نمای کلی داخل ماژول ثبات را می بینید.



در تصویر زیر نمای یک بیت از ماژول ثبات را می بینید. مدار پشت فلیپ فلاپ همان مدار توصیف شده برای تولید

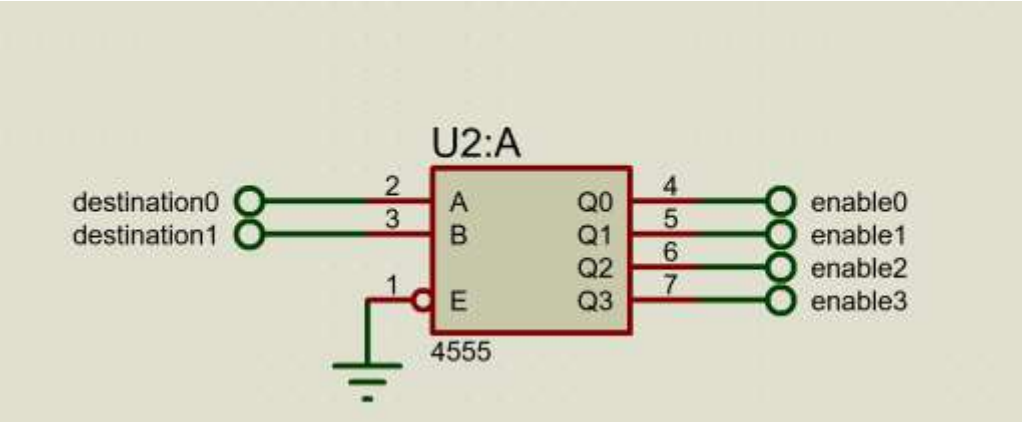
فلیپ‌فلاپ با سیگنال enable است.



بخش دیکودر

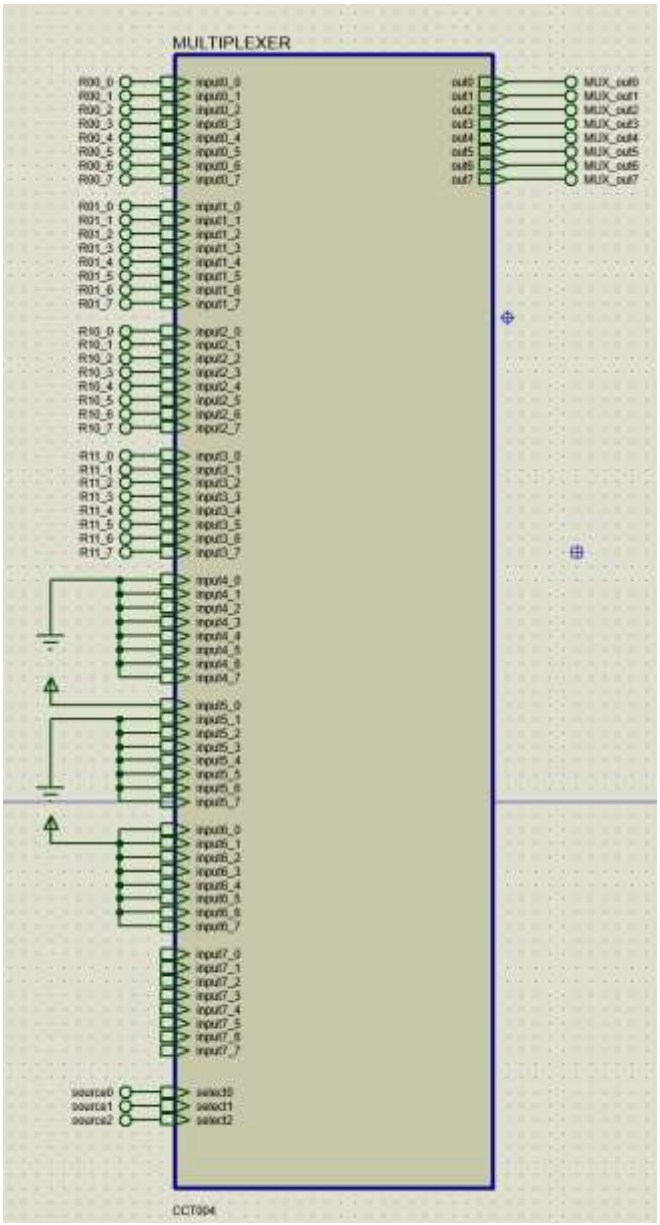
برای اینکه ثبات مقصد را انتخاب کنیم از یک دیکودر ۲ به ۴ استفاده می کنیم در حقیقت این دیکودر سیگنال enableثبات هایمان را برایمان درست می کند. ورودی های این دیکودر همان سیگنال ۲ بیتی destination است.

برای تحقق مدار دیکودر از تراشه ۴۵۵۵ استفاده شده است. در تصویر زیر نمای دیکودر را می بینید. همانطور که در قسمت قبل دیدید خروجی های این دیکودر به enable ثبات ها وصل شده بود.

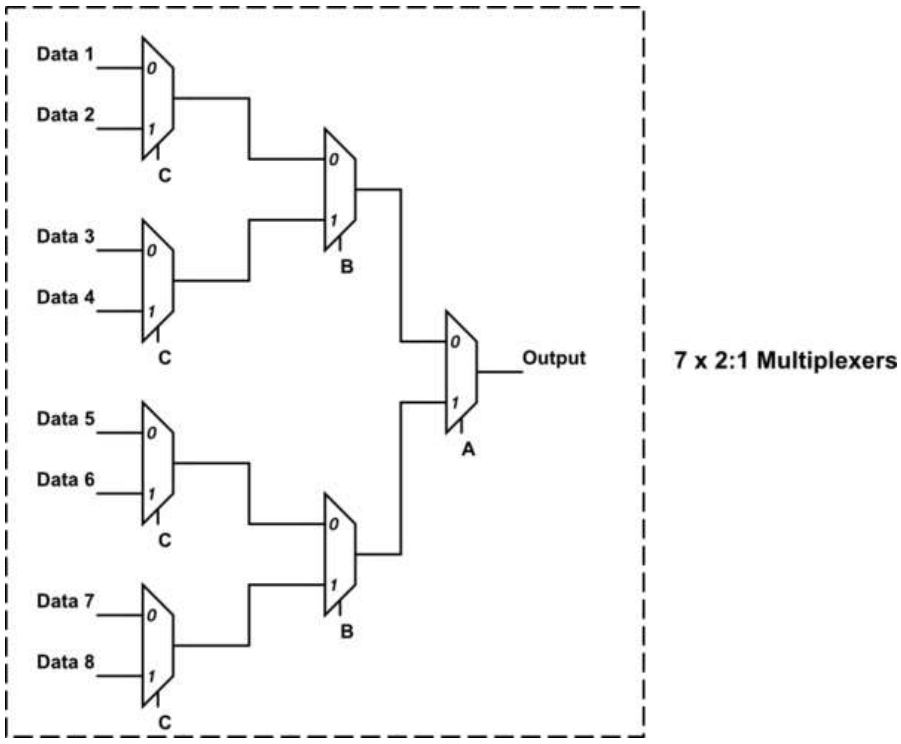


بخش مولتی‌پلکسر

برای تولید عملوند دوم جمع‌کننده/تفریق‌کننده نیاز به یک مالتی پلکسر ۸ به ۱ داریم. که ورودی و خروجی های آن ۸ بیتی هستند. برای این که مدار اصلی شلوغ نشود این مالتی پلکسر به صورت ماژول طراحی شده است. در تصویر زیر نمای کلی مولتی پلکسر را می بینید.

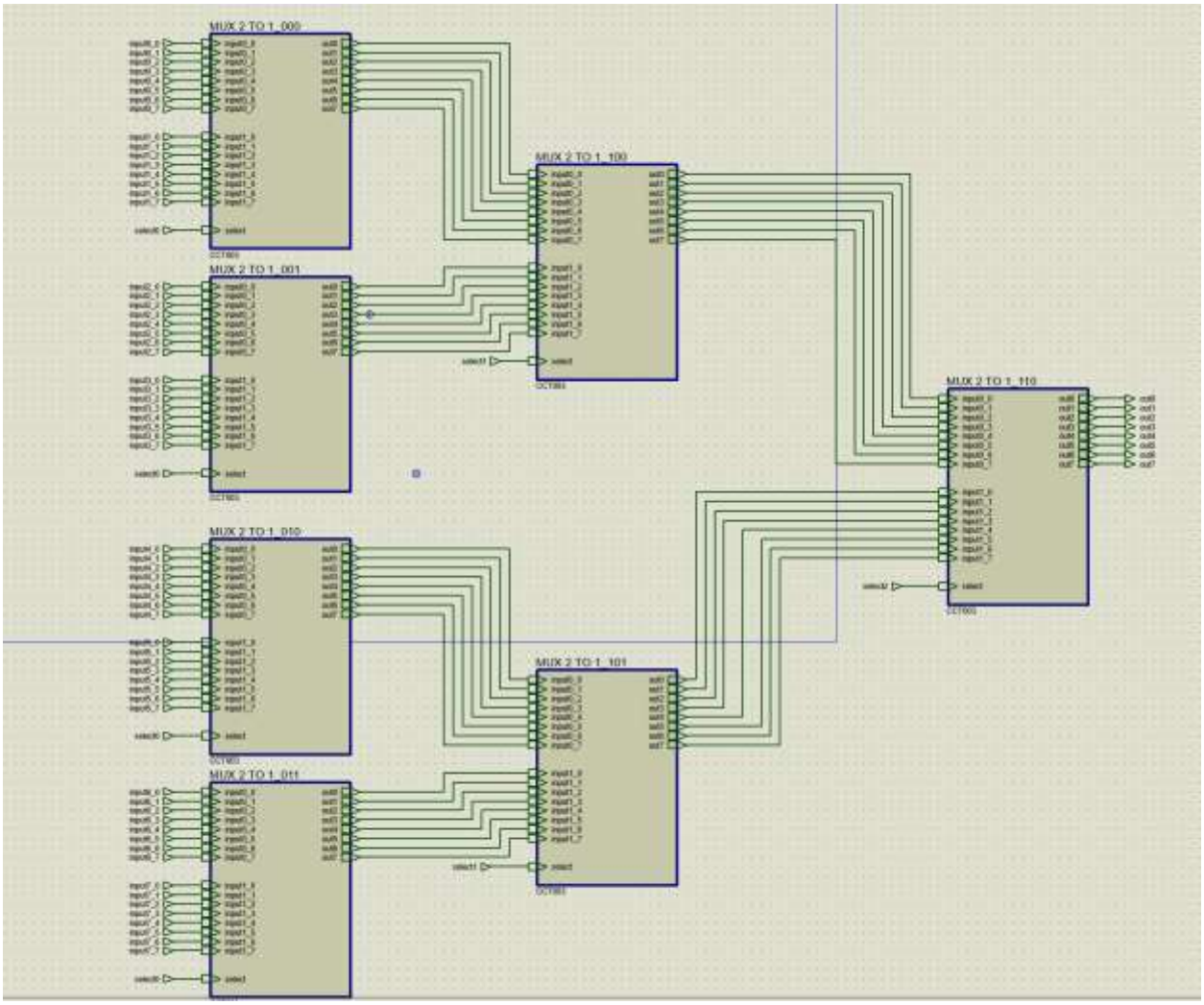


برای طراحی خود ماژول از ۷ مولتی پلکسر های ۲ به ۱ که خودشان یک ماژول هستند استفاده شده است.
شماتیک کلی این طراحی را در تصویر زیر می بینید.

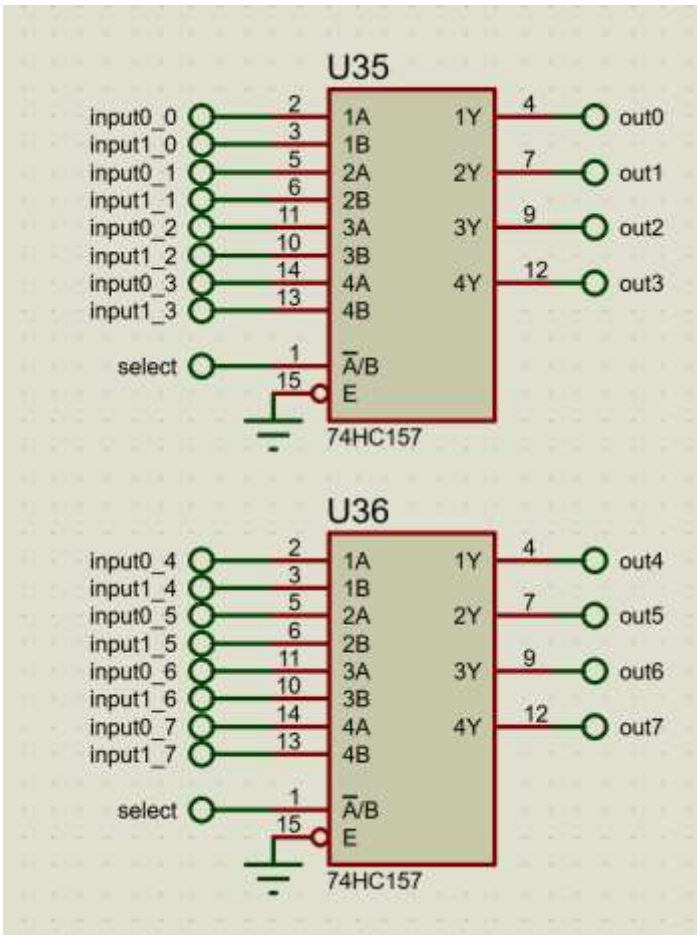


بنابراین برای پیاده‌سازی مالتی‌پلکسر با ۸ ورودی، همانطور که در دیاگرام سلسله‌مراتب ذکر شده است، ابتدا باید مالتی‌پلکسر دو ورودی تحقق پیدا کند. دقت داریم که در اینجا هر ورودی ما در واقع هشت بیت داده است.

در تصویری که در ادامه آمده است، نمای کلی این طراحی را می بینید:



حال صرفا کافیست مولتی پلکسر های ۲ به ۱ هشت بیتی را طراحی کنیم. برای طراحی این ماژول از ۲ تراشه 74HC157 استفاده می کنیم که هر کدام یک مولتی پلکسر ۲ به ۱ چهار بیتی هستند. در تصویر زیر نمای کلی این بخش را می بینید.



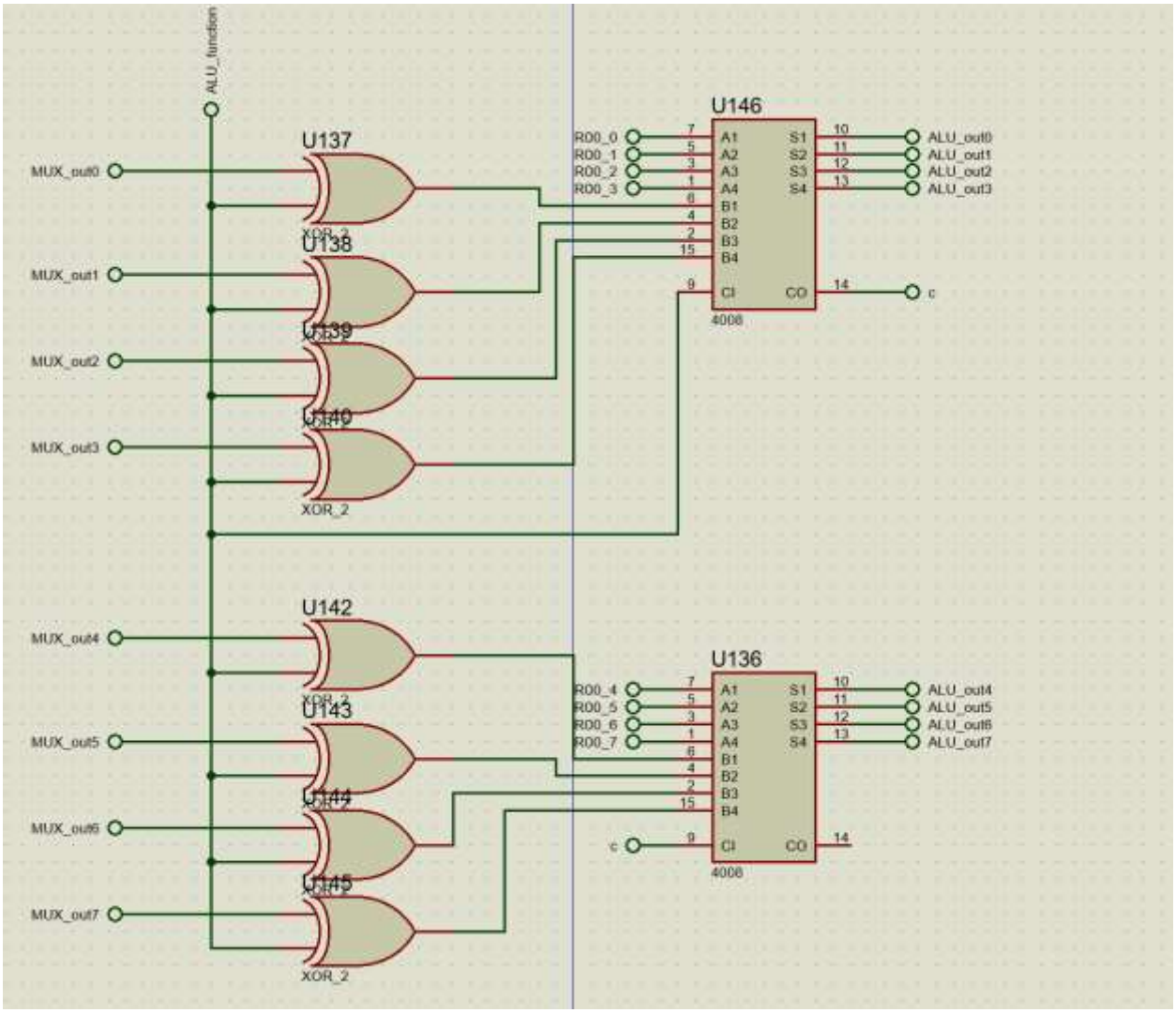
حال که مولتی پلکسر خود را طراحی کرده ایم کافیست ورودی های آن را همانند شماتیک گزارش کار قرار بدهیم.

بخش جمع کننده/تقریق کننده

برای این بخش از ۲ عدد از تراشه ۴۰۰۸ استفاده می کنیم این تراشه یک جمع کننده ی ۴ بیتی است. برای درست کردن یک جمع کننده ی ۸ بیتی کافیست دو عدد از این تراشه را در کنار هم بگذاریم و خروجی cout اولی را به cin دومی متصل کنیم.

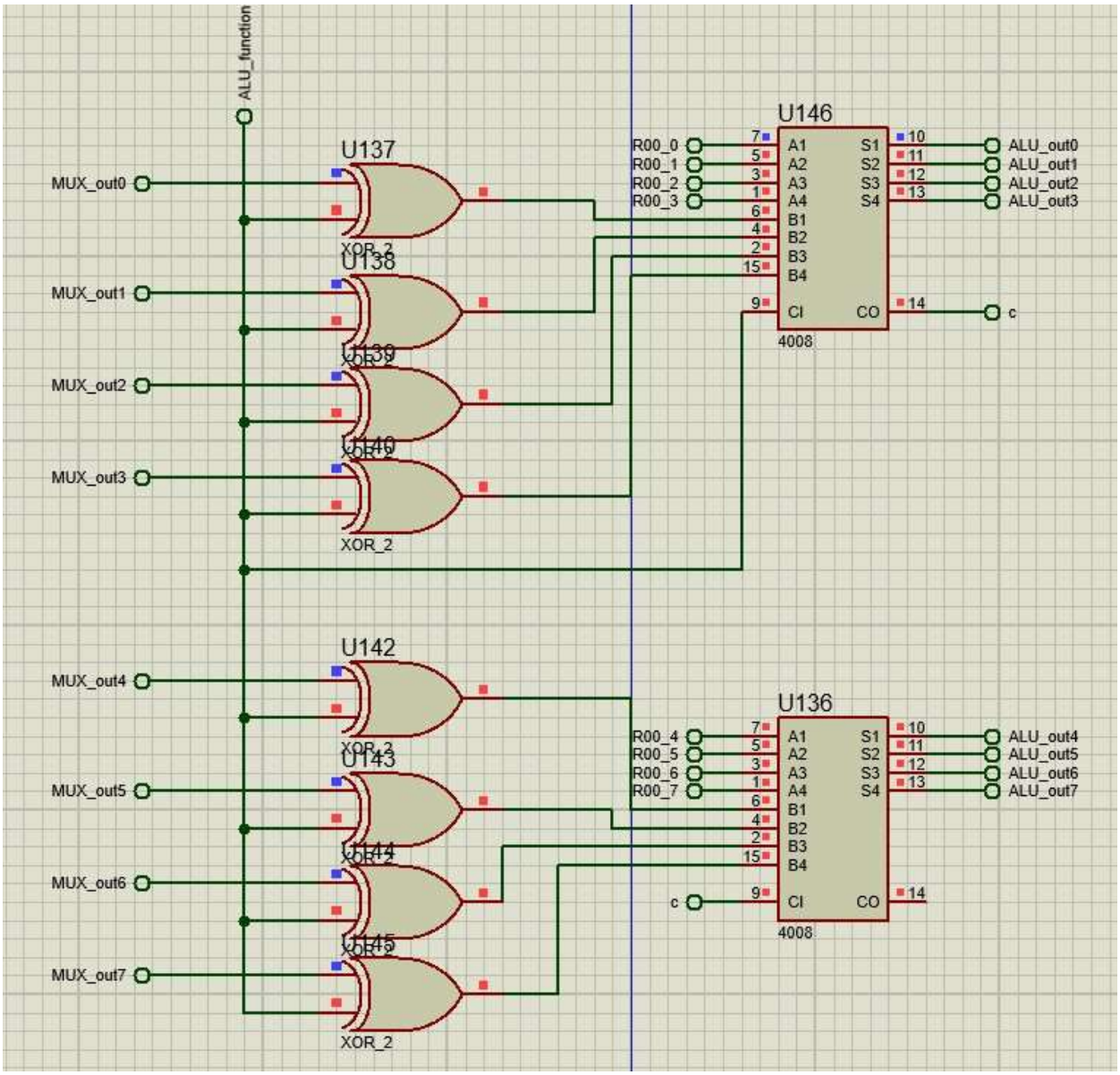
برای این که این قسمت بتواند تفریق هم انجام بدهد کافیست در صورتی که ALU_function فعال باشد نقیض عملوند دوم را به جمع کننده بدهیم و همچنین عدد حاصل را با ۱ جمع کنیم که این کار را با فعال کردن cin جمع کننده اول انجام می دهیم (دلیل این موضوع این است که اعداد منفی را بصورت مکمل دو نشان می دهیم).

در تصویر زیر نمای کلی این بخش را می بینید.

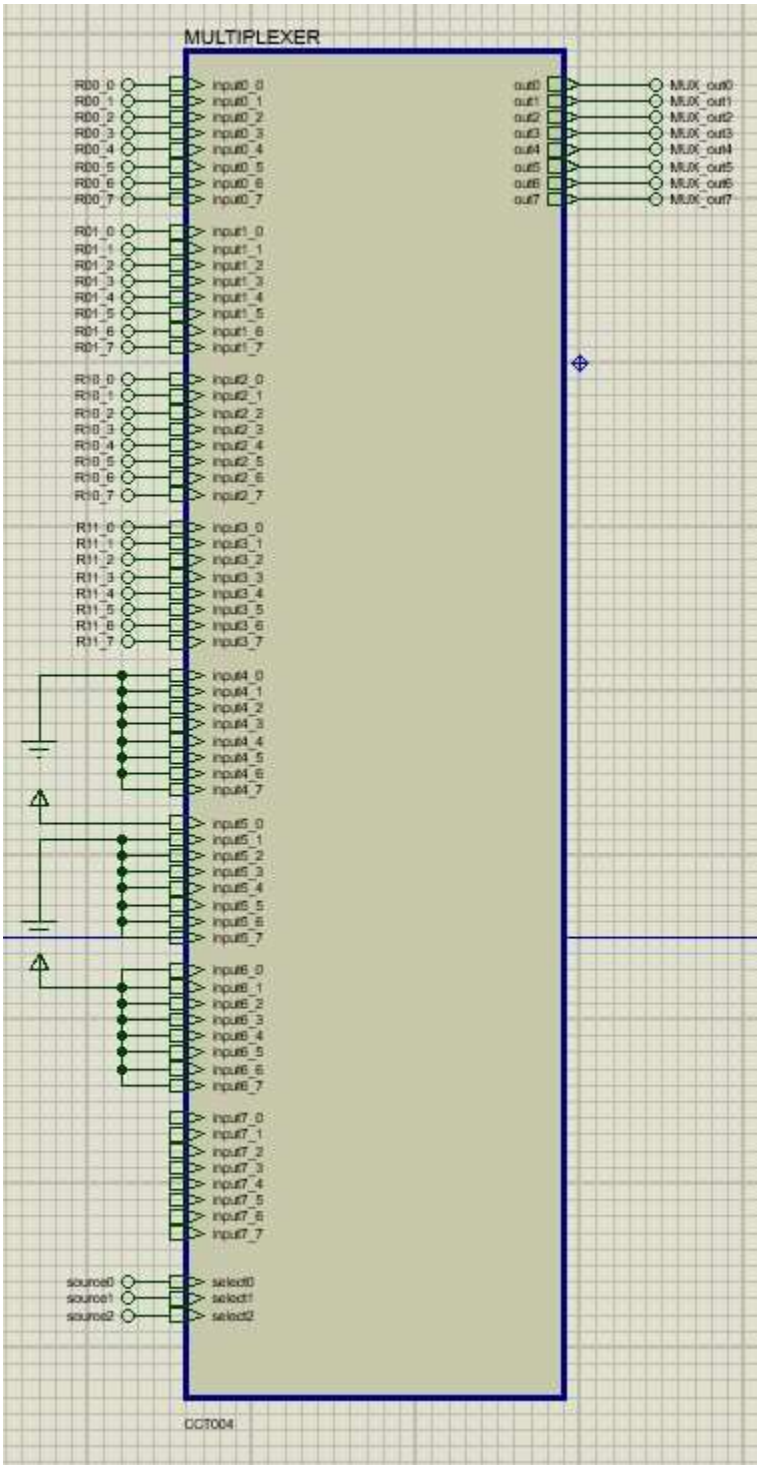


تصاویر کلی از مدار نهایی

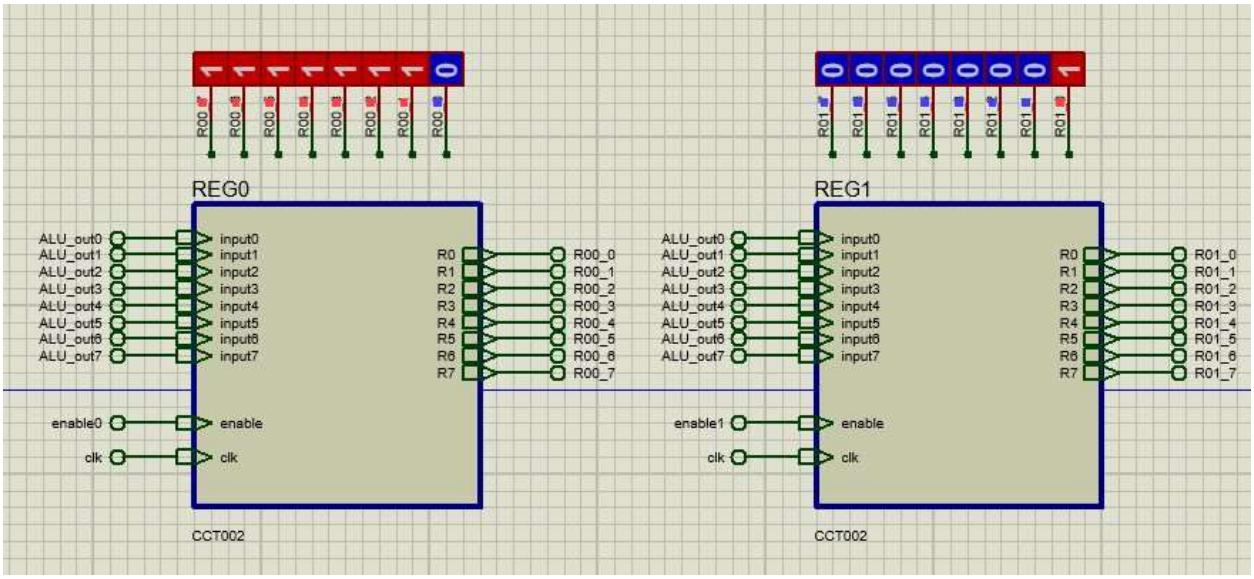
بخش محاسبات و منطق



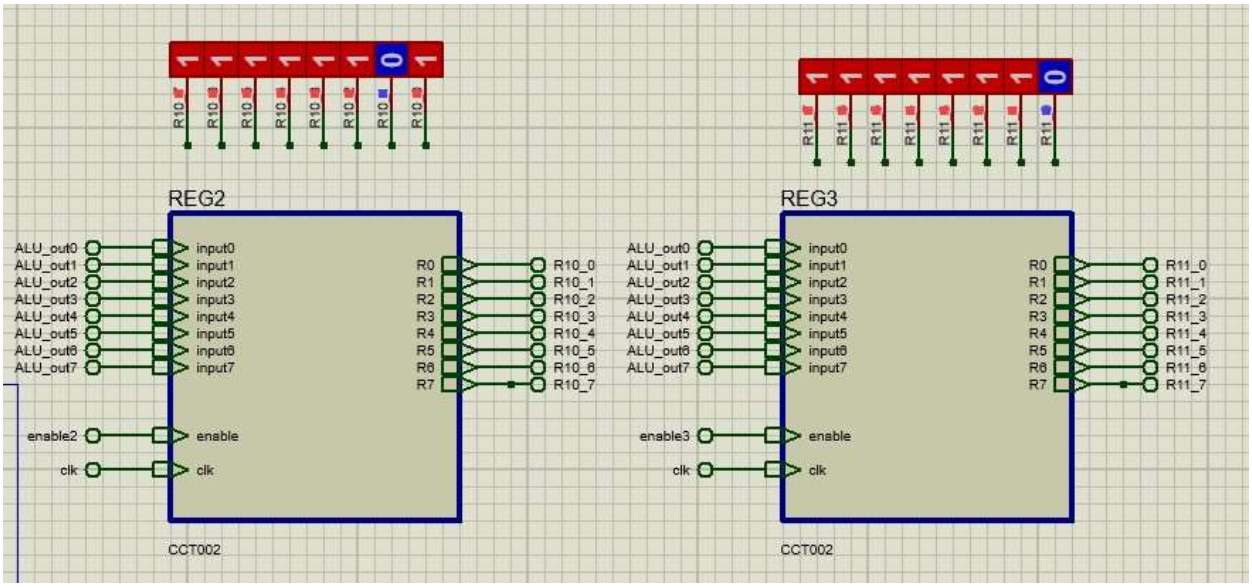
مالتی‌پلکسر



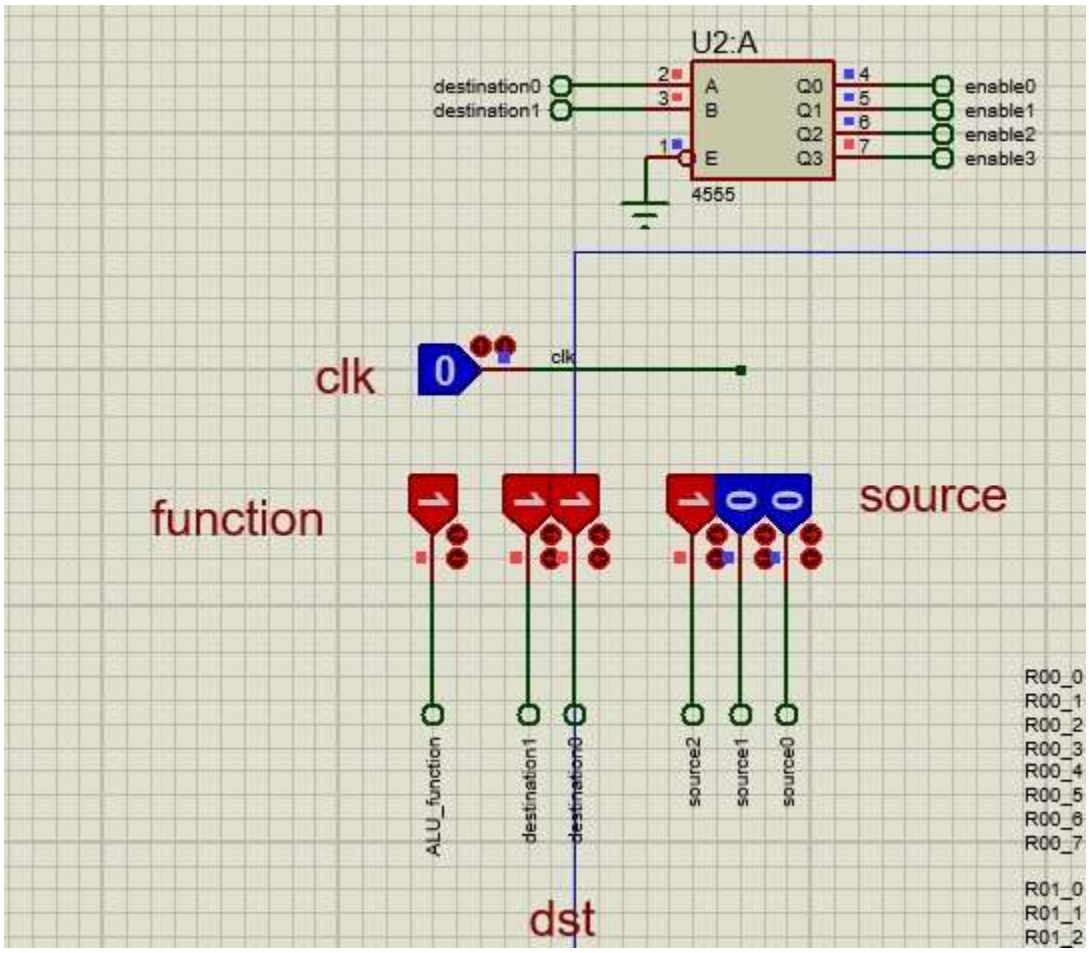
ثبات‌های R0 و R1



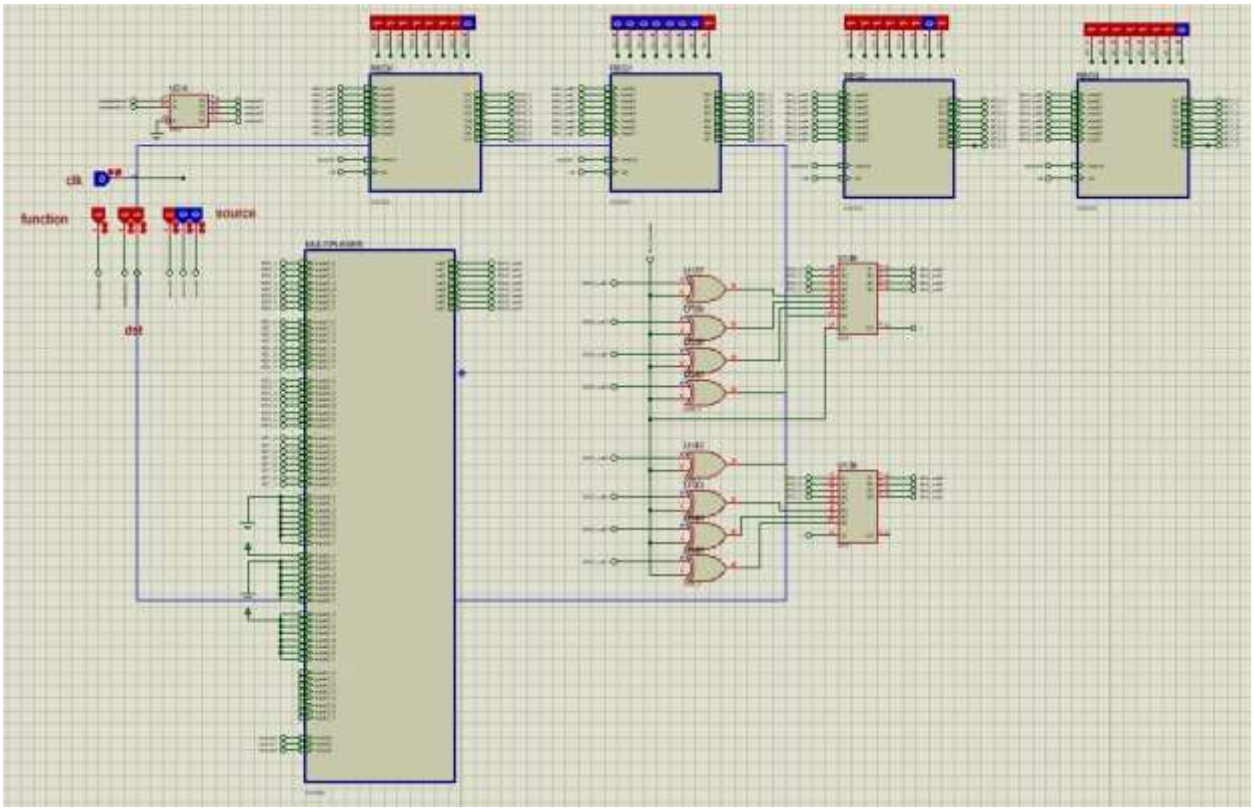
ثبات‌های R2 و R3



دیکودر و ورودی‌های مدار



شمای کلی مدار



حال در ادامه باید با دادن ورودی‌های نمونه از صحت عملکرد طراحی صورت گرفته مطمئن شویم.

آزمودن مدار با ورودی‌های نمونه

در این قسمت، جنبه‌های مختلف مدار را با دادن ورودی‌های نمونه و انجام عملیات‌های متفاوت، بررسی می‌کنیم و از صحت عملکرد مدار طراحی شده اطمینان حاصل می‌کنیم. در ادامه ۲۰ عملیات مختلف را در ۲۰ کلاک انجام خواهیم داد که در این عملیات‌ها تمامی مبداها و مقصدها و فانکشن‌های واحد محاسبه تست خواهند شد.

بارگذاری مقدار ۱ با عملیات جمع روی تمامی ثبات‌ها

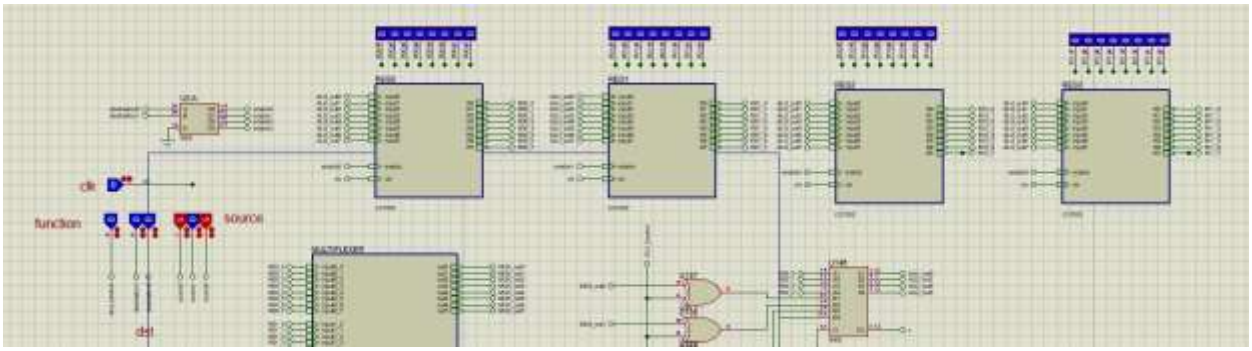
در ابتدا برای شروع محاسبات، با قابلیتی که برای بانک ثبات در نظر گرفته شده، مقادیر اولیه موجود را به عنوان یک عملوند استفاده می‌کنیم.

برای این منظور ابتدا source را برابر ۵ گذاشته و destination را از ۰ تا ۳ یکی یکی زیاد می‌کنیم.

از آنجایی که عملیات روی جمع (۰) است، در هر مرحله، مقدار R0 + 1 وارد مقصد می‌شود.

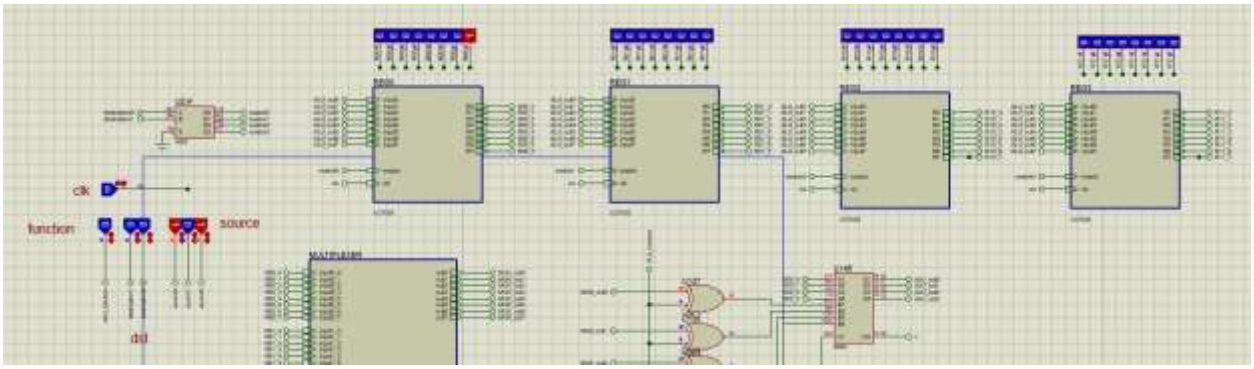
حالت اولیه:

در ابتدا مقدار تمامی ثبات‌ها توسط پروتئوس مقدار صفر گرفته است. بنابراین حاصل R0+1 برابر ۱ است.



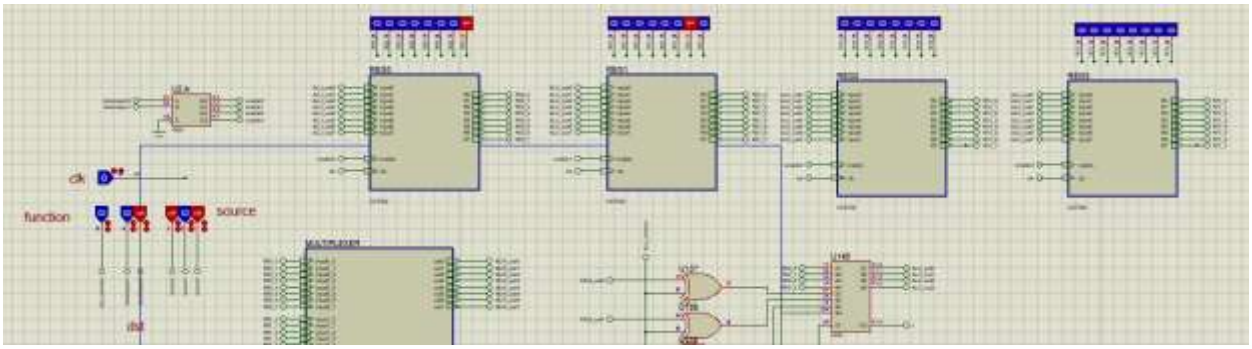
کلاک اول:

مقدار R0+1 که برابر با ۱+۰ است وارد مقصد یعنی R0 خواهد شد:



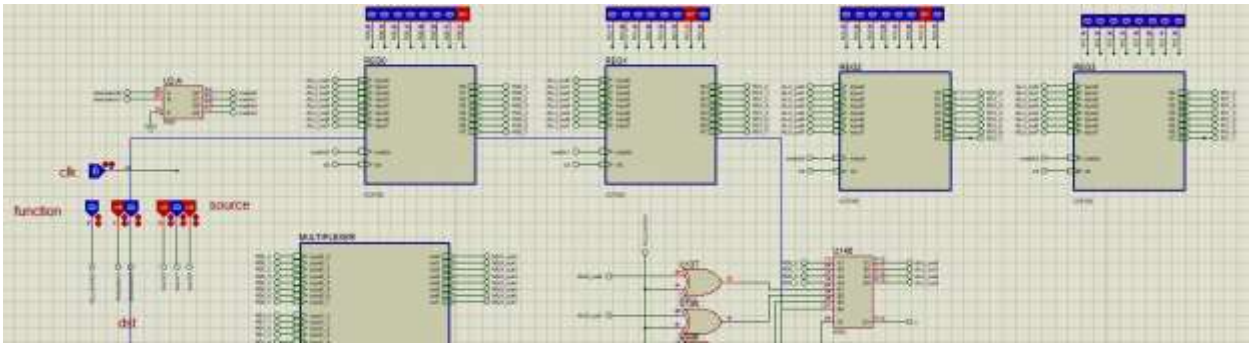
کلاک دوم:

مقصد را روی R1 تنظیم می‌کنیم. مقدار R0+1 که برابر با ۱+۱ است وارد مقصد یعنی R1 خواهد شد:



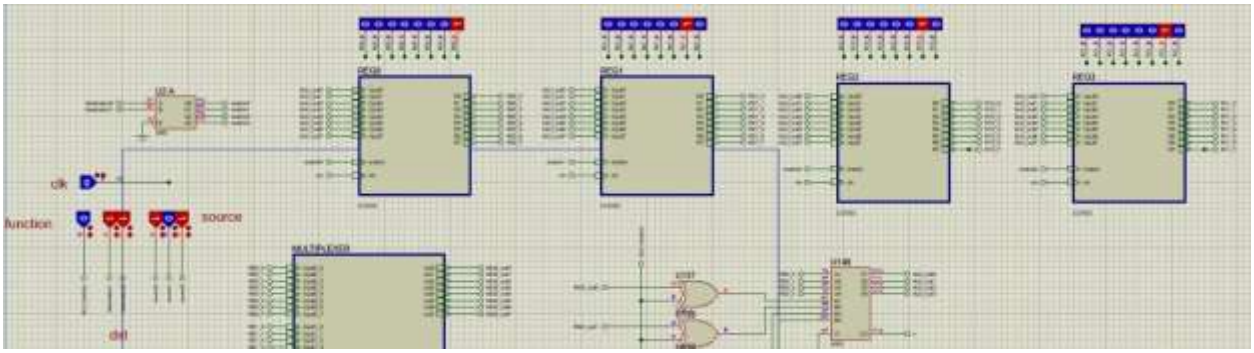
کلاک سوم:

مقصد را روی R2 تنظیم می‌کنیم. مقدار R0+1 که برابر با ۱+۱ است وارد مقصد یعنی R2 خواهد شد:



کلاک چهارم:

مقصد را روی R3 تنظیم می‌کنیم. مقدار R0+1 که برابر با ۱+۱ است وارد مقصد یعنی R3 خواهد شد:

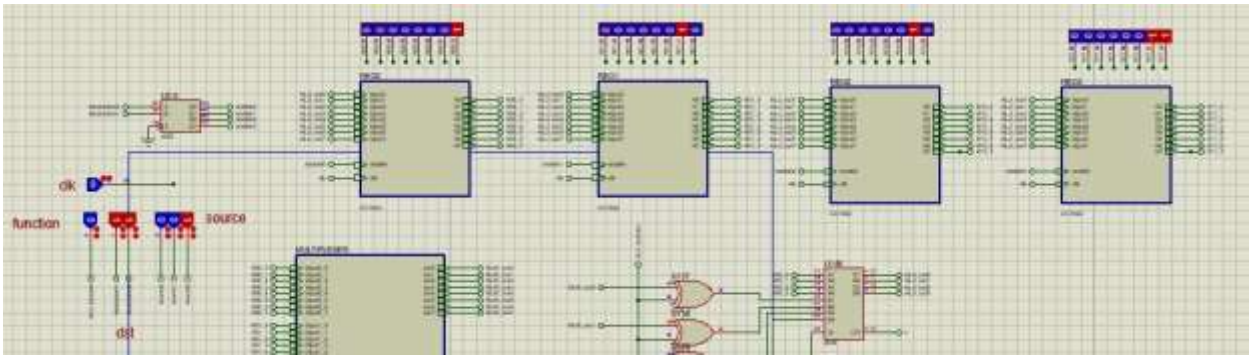


عملیات جمع میان ثبات‌های مختلف

در این قسمت، در ادامه‌ی قسمت‌های قبلی، مقدارهای ثبات‌ها را با هم جمع می‌زنیم تا sourceهای ۰ الی ۳ نیز بررسی شوند.

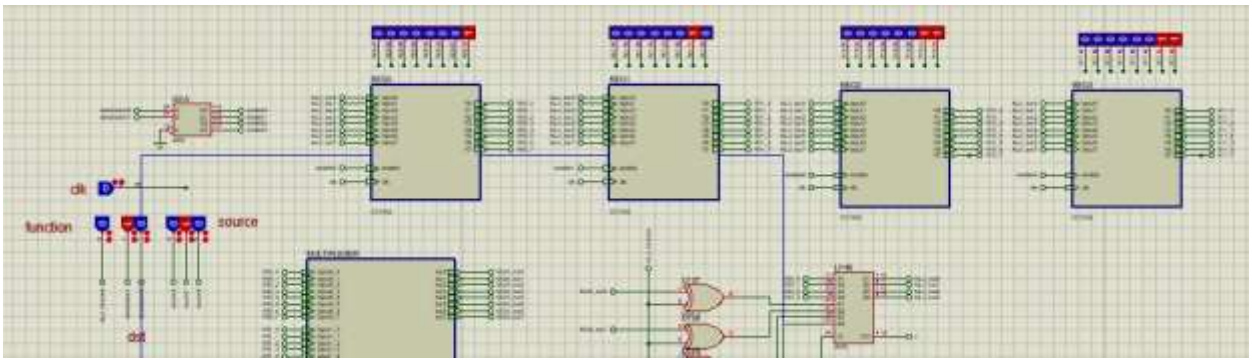
R0+R1 با مقصد R3

مقصد را R3 تنظیم کرده و source را روی ۱ می‌گذاریم. حال انتظار داریم ۱+۲ وارد R3 شود:



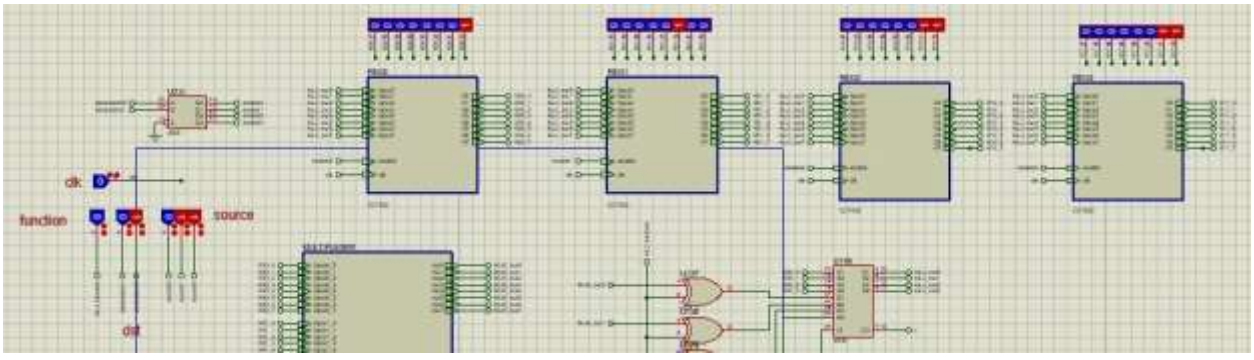
R0+R2 با مقصد R2

اینبار مقصد R2 است و source روی ۲ است. انتظار داریم ۲+۱ وارد R2 شود:



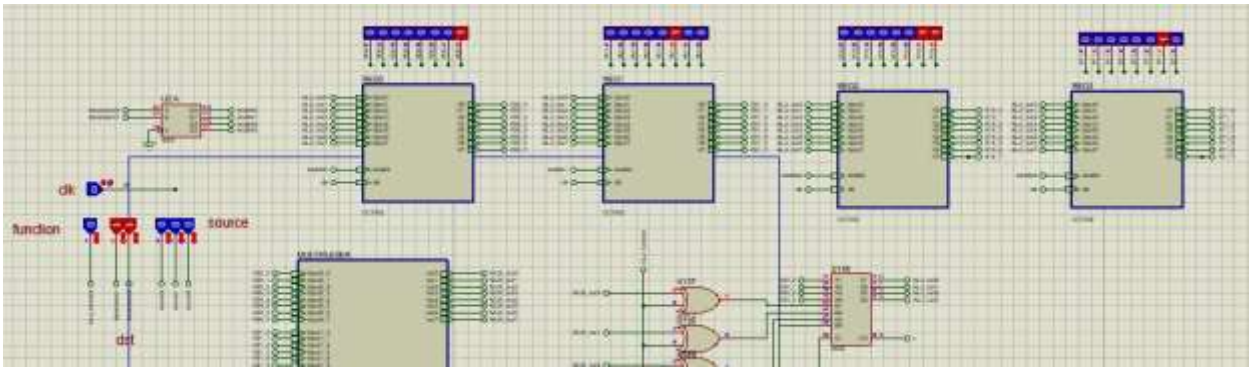
R0+R3 با مقصد R1

اینبار مقصد R1 است و source روی ۳ است. انتظار داریم ۳+۱ وارد R1 شود:



R0+R0 با مقصد R3

اینبار مقصد را مجددا R3 می‌گذاریم و source را روی ۰ قرار می‌دهیم. انتظار داریم ۱+۱ وارد R3 شود:

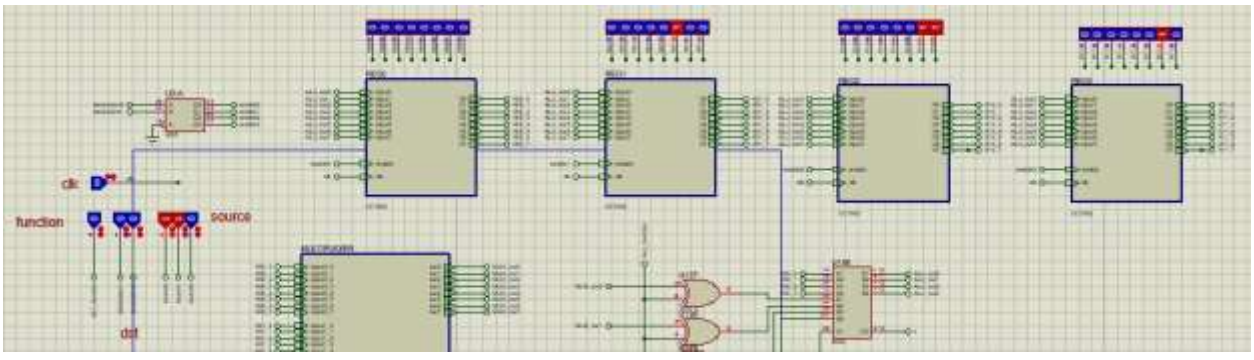


استفاده از مقدار اولیه ۱-

حال می‌توانیم از مقدار اولیه ۱- نیز استفاده کنیم تا صحت عملکرد این ورودی نیز بررسی شود. عملیات انجام شده کماکان جمع است.

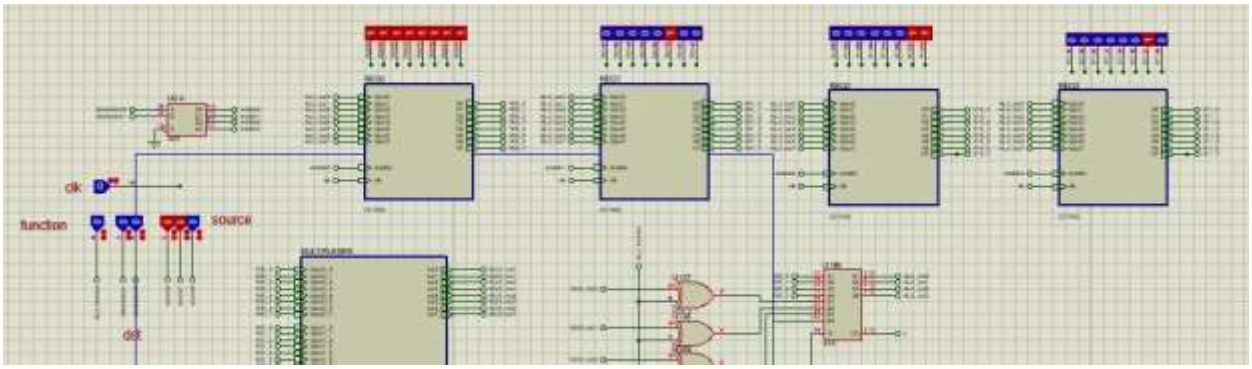
R0-1 با مقصد R0

برای این تست، source را روی ۶ گذاشته و مقصد را روی ۰ می‌گذاریم. انتظار داریم ۱-۱ وارد R0 شود:



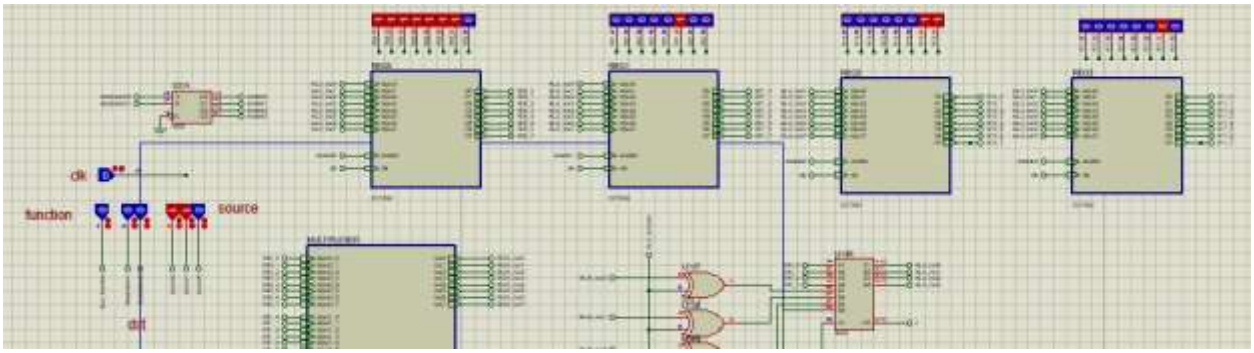
R0-1 با مقصد R0

صرفا یک بار دیگر کلاک را می‌زنیم. انتظار داریم مقدار R0 برابر ۱- شود یعنی 0xFF :



R0-1 با مقصد R0

مجددا یک بار دیگر کلاک می‌زنیم و انتظار داریم مقدار جدید R0 یکی کمتر شود یعنی 0xFE شود :



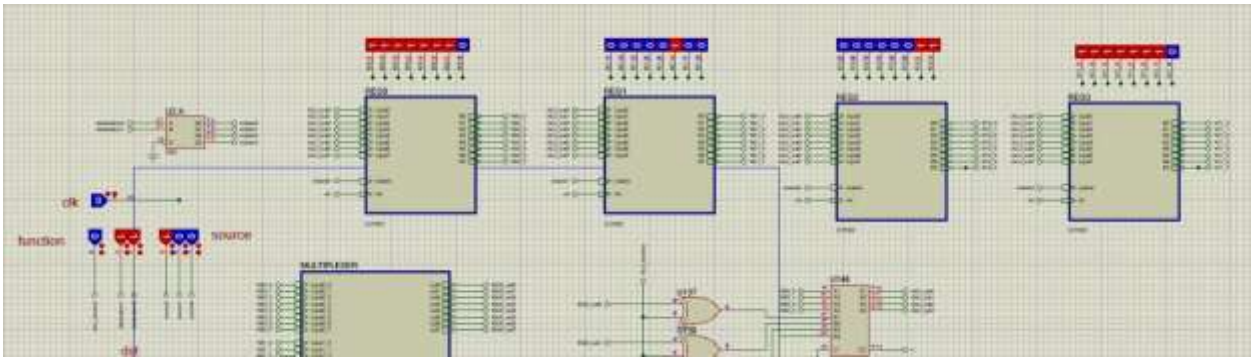
بنابراین مقداردهی ۱- نیز عملکرد درستی دارد. حال باید مقداردهی صفر یعنی سورس شماره ۴ را بررسی کنیم.

استفاده از مقدار اولیه ۰

در این حالت مقدار R0+0 وارد ثباتی به انتخاب ما می‌شود که درواقع برای انتقال مقدار R0 به سایر ثبات‌هاست.

R0+0 با مقصد R3

برای بررسی صحت عملکرد این قابلیت، مقدار R0 را به R3 انتقال می‌دهیم. source روی ۴ و destination روی ۳ قرار دارد:

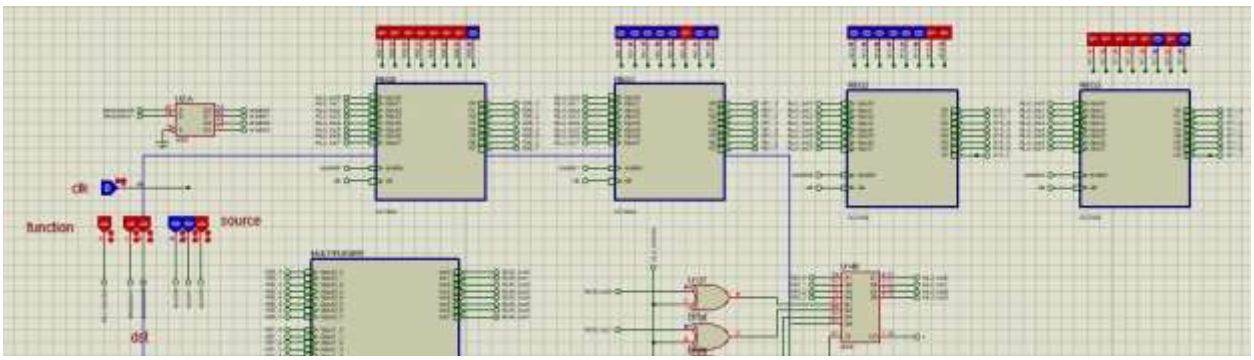


عملیات تفریق میان ثبات‌ها

تا اینجا کلیت عملکرد مدار بررسی شده است. تنها موردی که بررسی نشده است حالاتی است که عملیات ما بجای جمع، تفریق باشد. برای این منظور، عملیات مدار را به تفریق تغییر می‌دهیم و چند عملیات تفریق با مبدا و مقصدهای متفاوت را تحلیل می‌کنیم.

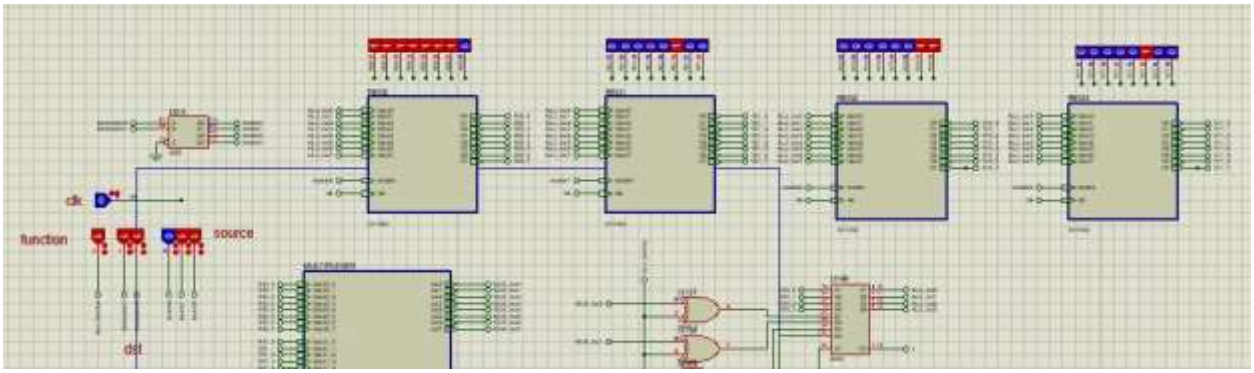
R0-R1 با مقصد R3

انتظار داریم 11111010 وارد R3 شود:



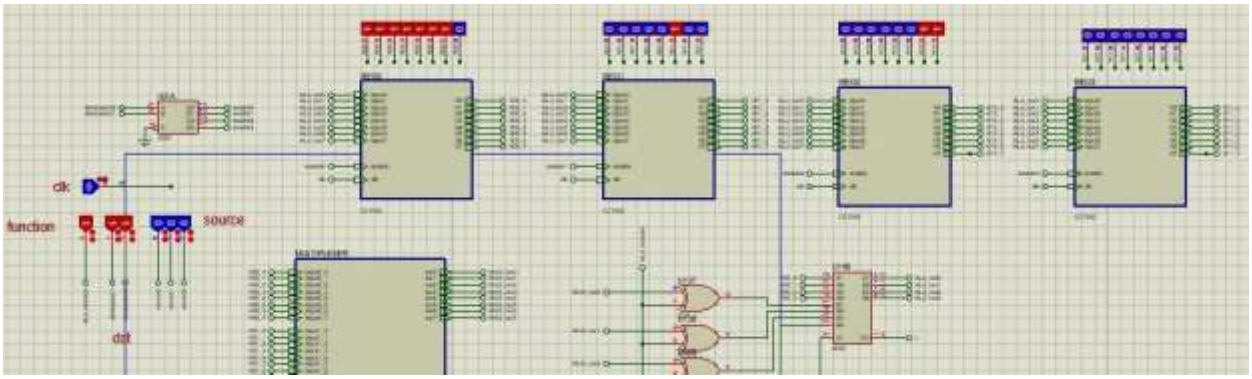
R0-R3 با مقصد R3

انتظار داریم 00000100 وارد R3 شود:



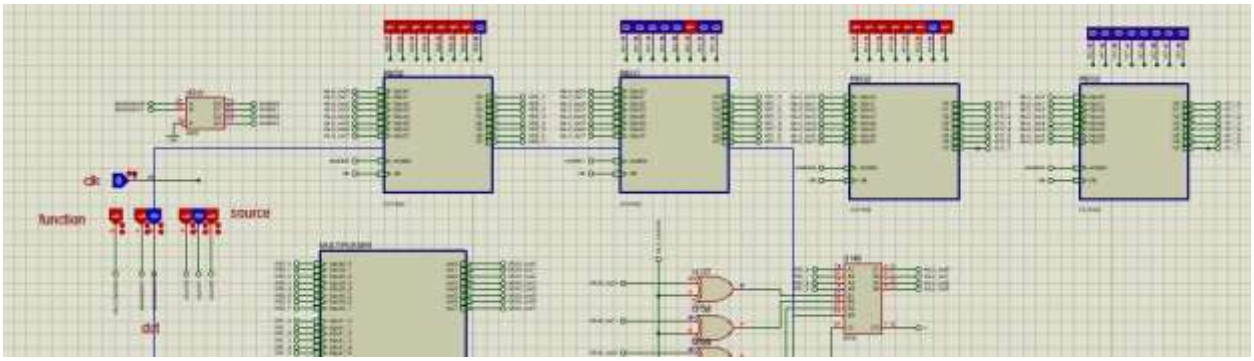
R0-R0 با مقصد R3

انتظار داریم مقدار صفر وارد R3 شود:



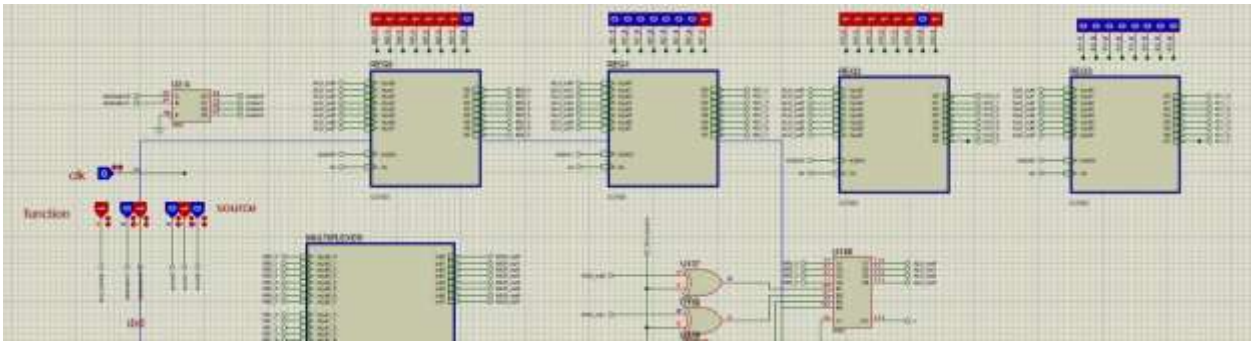
R0-1 با مقصد R2

انتظار داریم مقدار 11111101 وارد R2 شود. source روی ۵ قرار دارد.



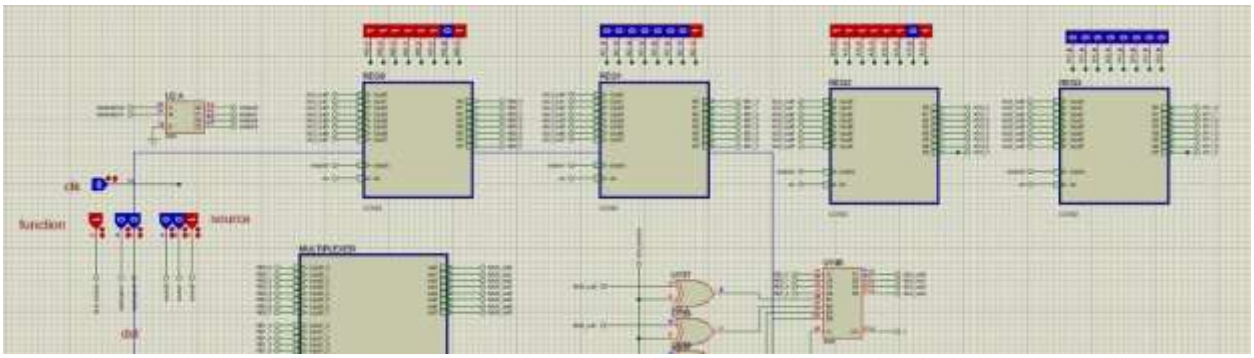
R0-R2 با مقصد R1

انتظار داریم مقدار 00000001 وارد R1 شود:



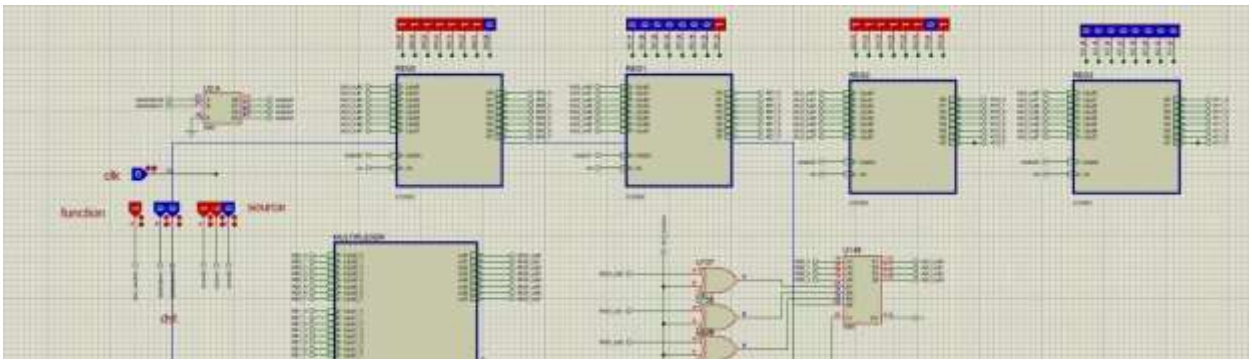
R0-R1 با مقصد R0

انتظار داریم مقدار 11111101 وارد R0 شود یعنی decrement شود:



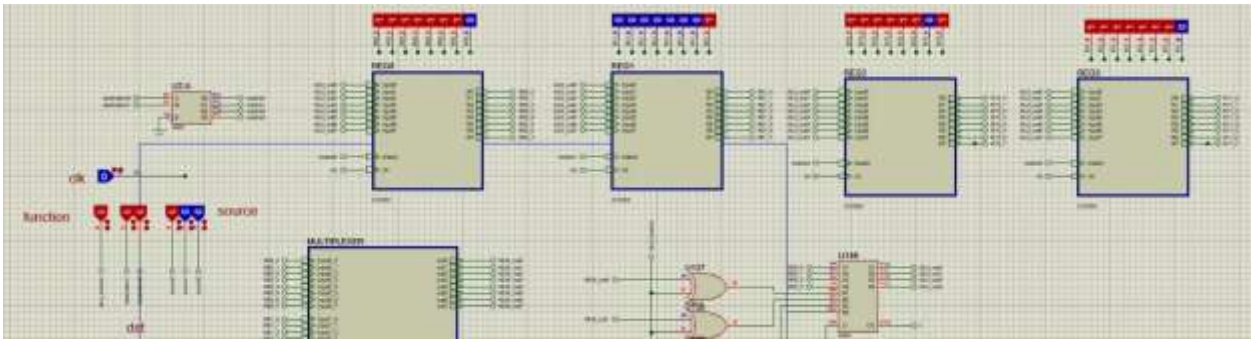
R0-(-1) با مقصد R0

انتظار داریم مقدار R0 یکی زیاد شود. source روی ۶ است (۱-) و مقصد R0 است:



R0-0 با مقصد R3

انتظار داریم مقدار R0 به R3 انتقال پیدا کند:



نتیجه‌گیری

بنابراین مشاهده می‌شود که طی ۲۰ تست انجام شده که تمامی مقصدها و مبداها هم برای جمع و هم برای تفریق تست شدند، همه قسمت‌های مختلف مدار به درستی کار می‌کند.