

به نام خدا



واحد محاسبات و منطق

آزمایشگاه مدار منطقی

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نویسنده:

رادین چراغی

شماره دانشجویی:

۴۰۱۱۰۵۸۱۵

تاریخ ارائه تکلیف:

۱۴۰۲/۰۵/۳۱

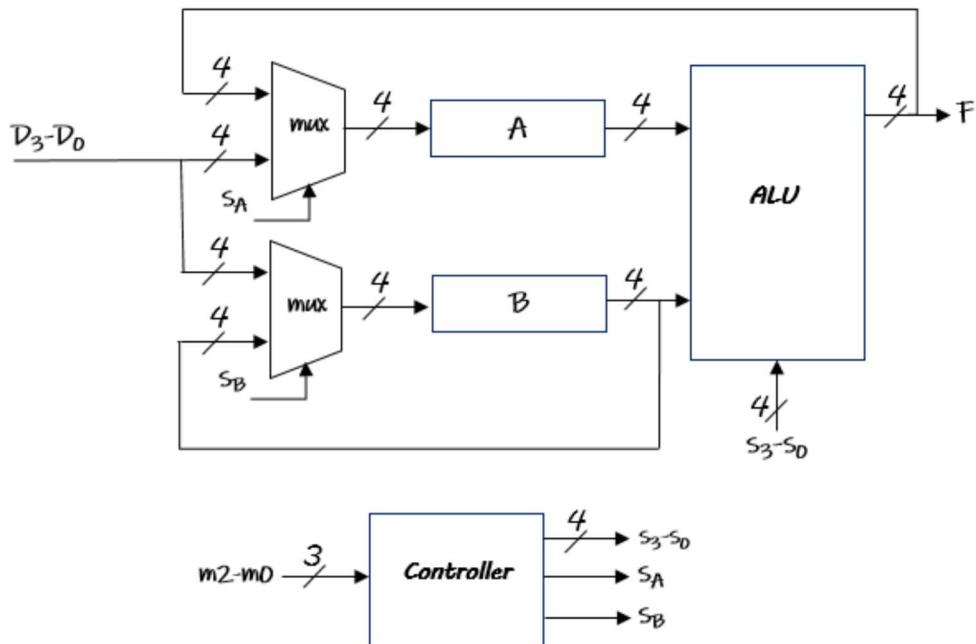
فهرست

۲	فهرست
۳	مقدمه
۳	آشنایی با تراشه ۷۴۱۸۱
۱۱	ساخت مدار داخلی ALU
۲۲	نتیجه گیری

هدف از انجام این آزمایش آشنایی با واحد محاسبات و منطق (ALU) است. در بخش اول این آزمایش مداری طراحی می‌کنیم که دارای دو رجیستر A, B و یک ایسی ALU است و با توجه به ورودی‌های داده شده به آن باید اعمالی را روی خروجی‌های رجیسترها انجام دهد. در بخش دوم از ما خواسته شده تا یک ALU مطابق آنچه در دستور کار آمده است طراحی کنیم.

آشنایی با تراشه ۷۴۱۸۱

ابتدا به شرح آزمایش می‌پردازیم. همانطور که در مقدمه گفته شد مدار این آزمایش دارای دو رجیستر A و B، دو مالتی‌پلکسر، یک ایسی ALU (۷۴۱۸۱) و یک کنترل کننده است. در قسمت کنترل کننده با توجه به ورودی‌ها و جدول آورده شده در دستور کار که در تصویر صفحه بعد قابل مشاهده است، عملیات مورد نظر بر روی اعداد رجیسترها مشخص می‌شود. این مدار دارای ۷ ورودی و دو کلید است که به آن‌ها می‌پردازیم. ورودی‌های M0-M2 مربوط به قسمت کنترل کننده هستند که عملیات مورد نظر با توجه به جدول زیر بر اساس آن‌ها مشخص می‌شود. چهار ورودی D0-D3، ورودی‌های مربوط به بارگذاری موازی است و برای زمانی است که می‌خواهیم در ثبات‌ها اعدادی را بارگذاری کنیم. همچنین دو push button برای clock و ریست کردن محتویات رجیسترها داریم. این مدار قادر سیگنال خروجی است و عملکرد آن از روی محتویات رجیسترها مشخص می‌شود. نمایی از مدار کلی در تصویر زیر قابل مشاهده است.



<i>M2</i>	<i>M1</i>	<i>M0</i>	<i>Operation</i>
0	0	0	$A \leftarrow D_3 - D_0$
0	0	1	$B \leftarrow D_3 - D_0$
0	1	0	$A \leftarrow A$
0	1	1	$A \leftarrow B$
1	0	0	clear (A)
1	0	1	$A \leftarrow \text{not}(A)$
1	1	0	$A \leftarrow \text{and}(A, B)$
1	1	1	$A \leftarrow \text{add}(A, B)$

حال به بررسی قطعات استفاده شده می‌پردازیم. در این مدار دو رجیستر 4 بیتی A, B را داریم که ورودی‌های آن‌ها به خروجی‌های یک مالتی‌پلکسر و خروجی‌های رجیسترها به ایسی ALU متصل است. یکی دیگر از قطعات مدار، ایسی ۷۴۱۸۱ یا همان ALU است. این قطعه یک مدار ترکیبی دیجیتال است که عملیات حساب و منطق را روی اعداد دودویی صحیح انجام می‌دهد. ابتدا به ورودی‌های آن می‌پردازیم. ورودی‌های A_0-A_3 و B_0-B_3 مربوط به دو عدد 4 بیتی‌ای است که روی آن‌ها عملیات انجام می‌شود. ورودی‌های CN و M نوع عملیات ایسی را مشخص می‌کنند که در جدول زیر مشاهده می‌شود. در صورتی که M یک باشد مدار در حالت عملیات منطقی و اگر M صفر بوده و CN یک باشد مدار در حالت عملیات حسابی قرار دارد. همانطور که در جدول زیر مشاهده می‌شود ورودی‌های $S0-S3$ یک عملیات خاص را مشخص می‌کنند.

MODE SELECT — FUNCTION TABLE

MODE SELECT INPUTS				ACTIVE HIGH INPUTS & OUTPUTS	
S_3	S_2	S_1	S_0	Logic ($M = H$)	Arithmetic** ($M = L$) ($C_n = H$)
L	L	L	L	\bar{A}	A
L	L	L	H	$\overline{A+B}$	$A+B$
L	L	H	L	\bar{AB}	$A+\bar{B}$
L	L	H	H	Logical 0	minus 1
L	H	L	L	\overline{AB}	A plus $A\bar{B}$
L	H	L	H	\bar{B}	$(A+B)$ plus $A\bar{B}$
L	H	H	L	$A \bullet B$	A minus B minus 1
L	H	H	H	\bar{AB}	AB minus 1
H	L	L	L	$\bar{A}+B$	A plus AB
H	L	L	H	$\overline{A \bullet B}$	A plus B
H	L	H	L	B	$(A+\bar{B})$ plus AB
H	L	H	H	AB	AB minus 1
H	H	L	L	Logical 1	A plus A^*
H	H	H	H	$A+B$	$(A+B)$ plus A
H	H	H	L	$A+B$	$(A+\bar{B})$ plus A
H	H	H	H	A	A minus 1

در این مدار همانطور که گفته شد دو مالتیپلکسر داریم که هر کدام مربوط به یکی از رجیسترها است. ایسی این قطعه ۷۴۱۵۷ است که مجموعی از ۴ مالتیپلکسر ۲ به ۱ است که همگی آن‌ها یک خط آدرس مشترک دارند. این ایسی به طور کلی ۴ خروجی، ۴ ورودی دوتایی (هر جفت مربوط به یک بیت خروجی) و یک خط آدرس (برای مالتیپلکسر A، SA و برای B، SB) دارد که با توجه به خط آدرس یکی از هر جفت به بیت خروجی مربوط به آن متصل می‌شود. با توجه به جدول آورده شده در ابتدای صفحه قبل برای مالتیپلکسر مربوط به A ورودی‌های اول هر جفت را به ورودی‌های D0-D3 متصل می‌کنیم و ورودی‌های D0-D3 که F0-F3 است متصل می‌کنیم. برای مالتیپلکسر مربوط به B هم به همین صورت عمل می‌کنیم با این تفاوت که ورودی‌های ALU دوم هر جفت را به خروجی رجیستر B متصل می‌کنیم زیرا B فقط در یک سطر جدول ابتدای صفحه‌ی قبل تغییر می‌کند که آن هم زمانی است که در آن D0-D3 بارگذاری می‌شوند.

همانطور که گفتیم این مدار یک قسمت کنترل کننده دارد که با توجه به M0-M2، S0-S3 و SA و SB مشخص می‌شوند. با توجه به جداول صفحه قبل برای قسمت کنترل کننده به صورت زیر ساده سازی می‌کنیم.

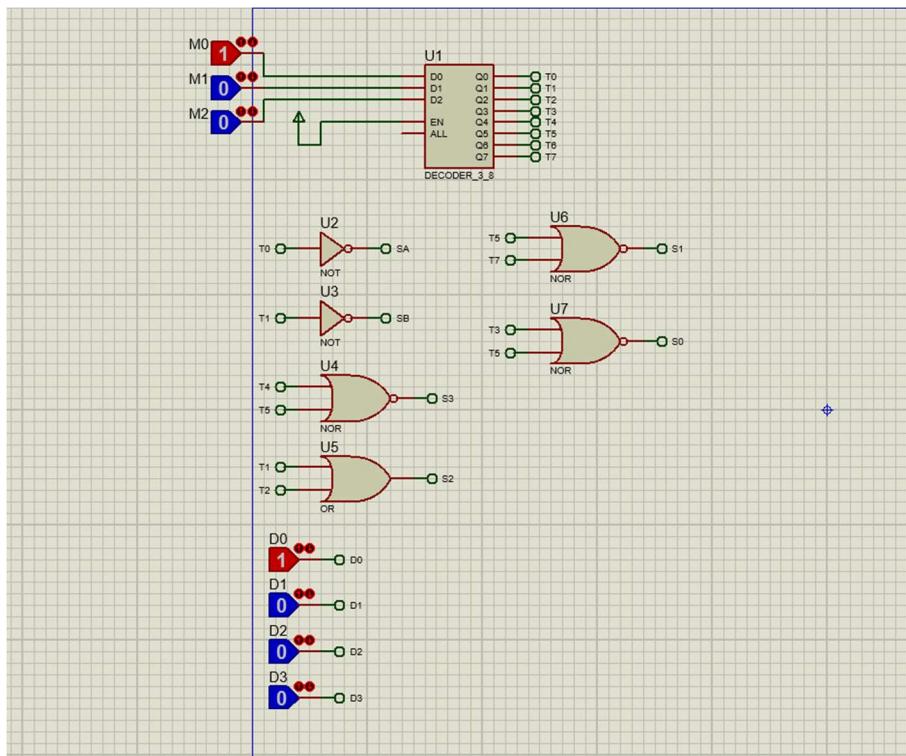
M ₂ M ₁ M ₀	SA	SB	S ₄	S ₂	S ₁ , S ₀	CN	M
T ₀ 0 0	0	1	X X	X X	X	X	1
T ₁ 0 0 1	1	0	1 1	1 1	1	X	1
T ₂ 0 1 0	1	1	1 1	1 1	1	X	1
T ₃ 0 1 1	1	1	1 0	1 0	1	X	1
T ₄ 1 0 0	1	1	0 0	0 1	1	X	1
T ₅ 1 0 1	1	1	0 0 0	0 1 1	1	X	1
T ₆ 1 1 0	1	1	1 0 1	1 0 1	1	X	1
T ₇ 1 1 1	1	1	1 0 0 1	1 0 0 1	1	1	0

$$SA = \overline{T_0} \quad SB = \overline{T_1} \quad S_4 = (\overline{T_4} + \overline{T_5}) \quad S_2 = T_1 + T_2$$

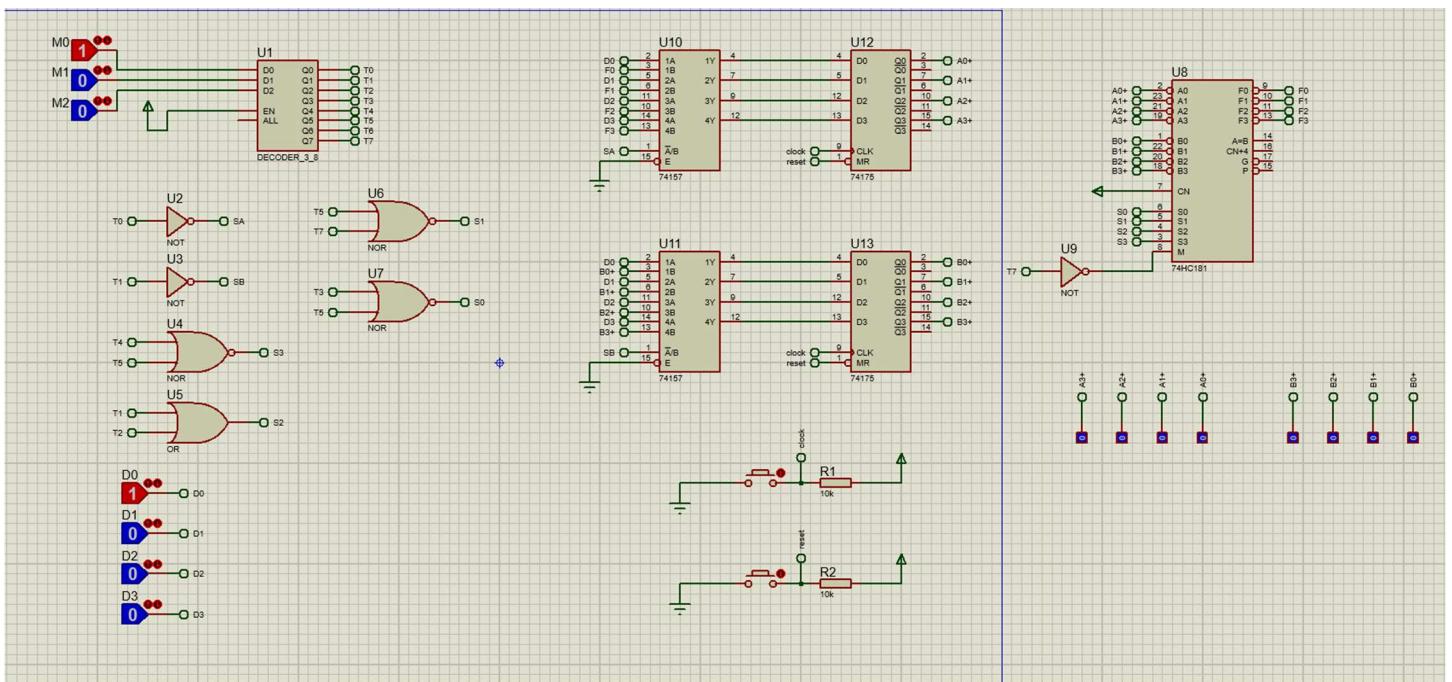
$$S_1 = (\overline{T_5} + \overline{T_7}) \quad S_0 = (\overline{T_4} + \overline{T_5})$$

$$CN = 1 \quad M = \overline{T_7}$$

حال با استفاده از روش دیکودر مدار را می‌سازیم به اینصورت که M0-M2 را به یک دیکودر ۳ به ۸ متصل می‌کنیم و هر کدام از خروجی‌های آن‌ها را به ترتیب به ترمینال‌های T0-T8 متصل می‌کنیم و با توجه به معادلات بالا SA, SB, S0-S3 را می‌سازیم که مدار مربوط به آن‌ها در تصویر صفحه بعد مشاهده می‌شود.



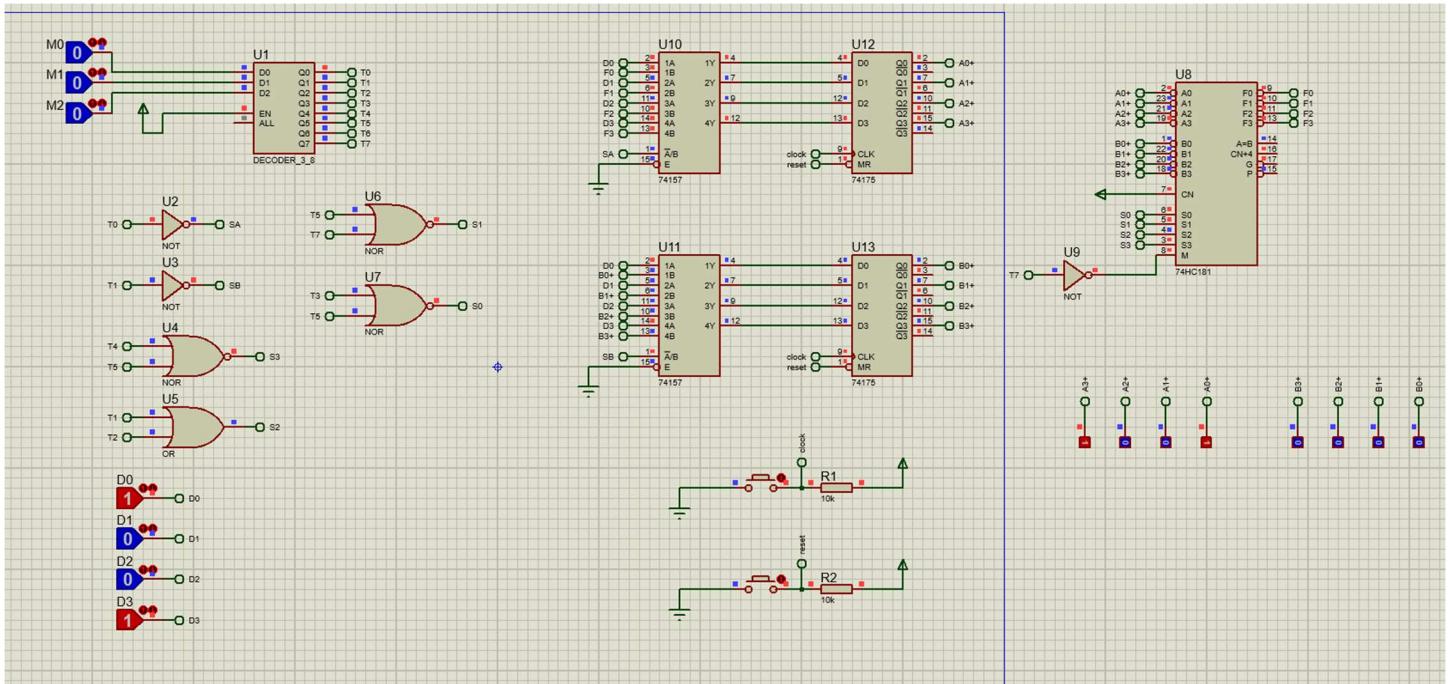
در نهایت تصویر کلی مدار در Proteus به صورت زیر می‌شود:



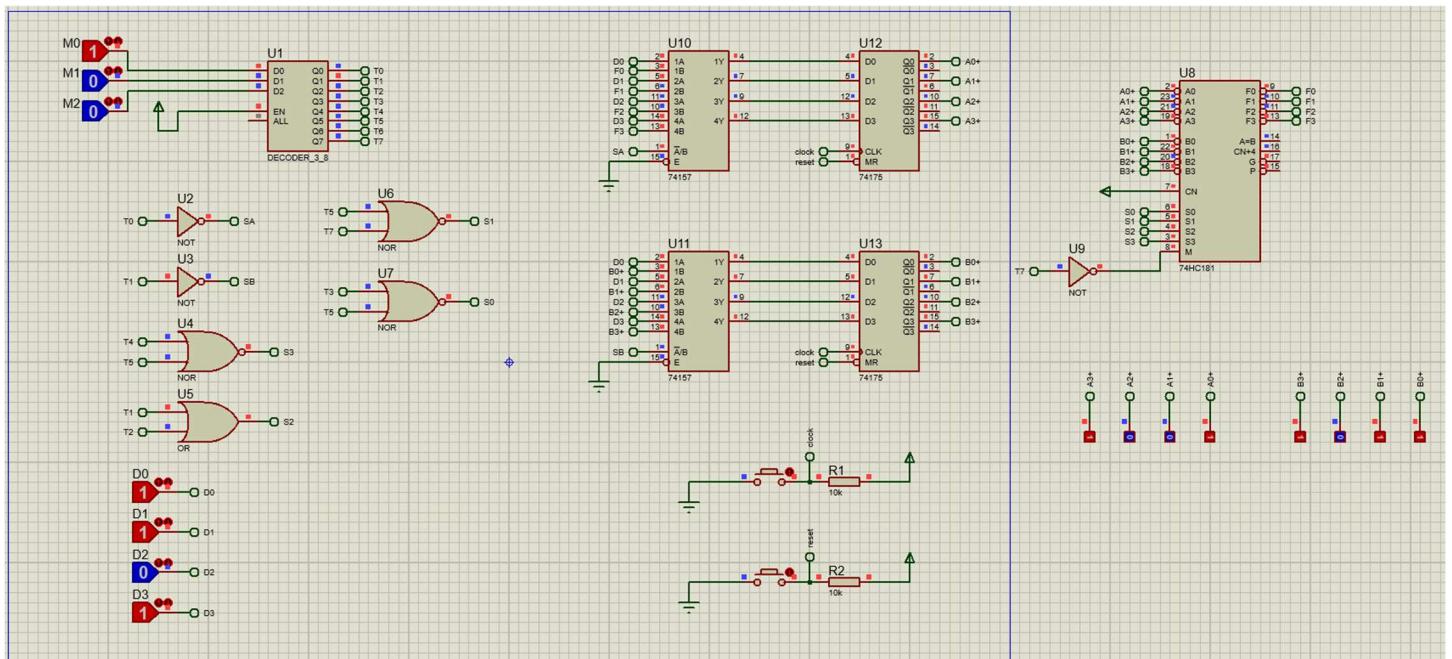
همانطور که در تصویر بالا نیز مشاهده می‌شود دو push button برای ریست و کلک قرار داده شده و به ترتیب به CLK و MR متصل شده‌اند.

حال به بررسی عملکرد مدار با توجه به ورودی‌های M0-M2 می‌پردازیم. مراحل زیر همگی به ترتیب انجام شده‌اند.

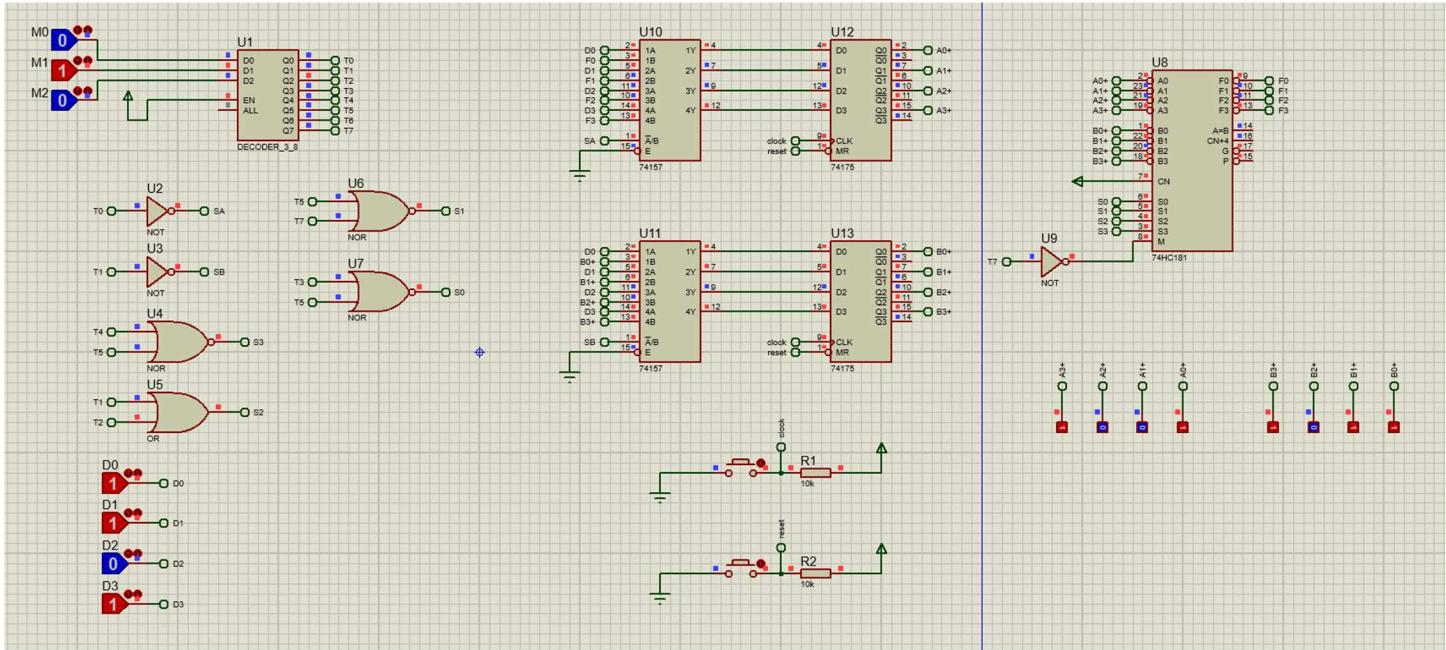
به ازای ورودی ۰۰۰ برای M0-M2 عدد ۱۰۰۱ را در A لود می‌کنیم:



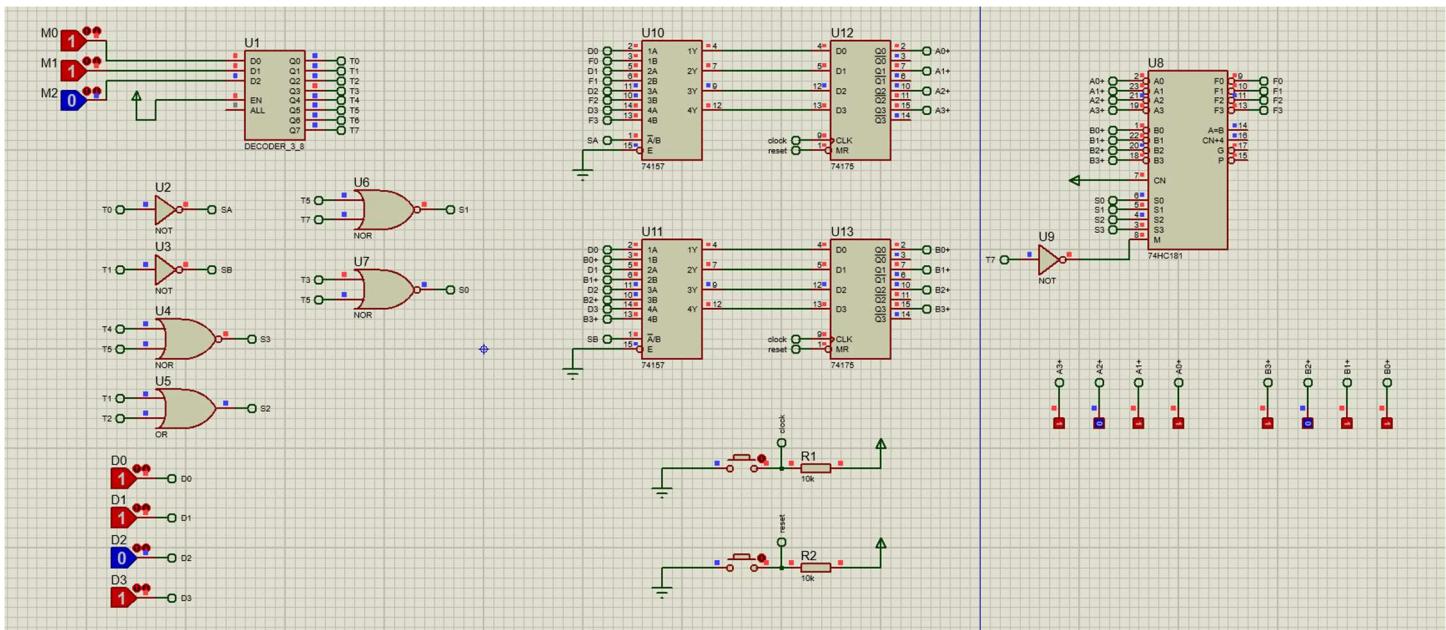
به ازای ورودی ۰۰۱ برای M0-M2 عدد ۱۰۱۱ را در B لود می‌کنیم:



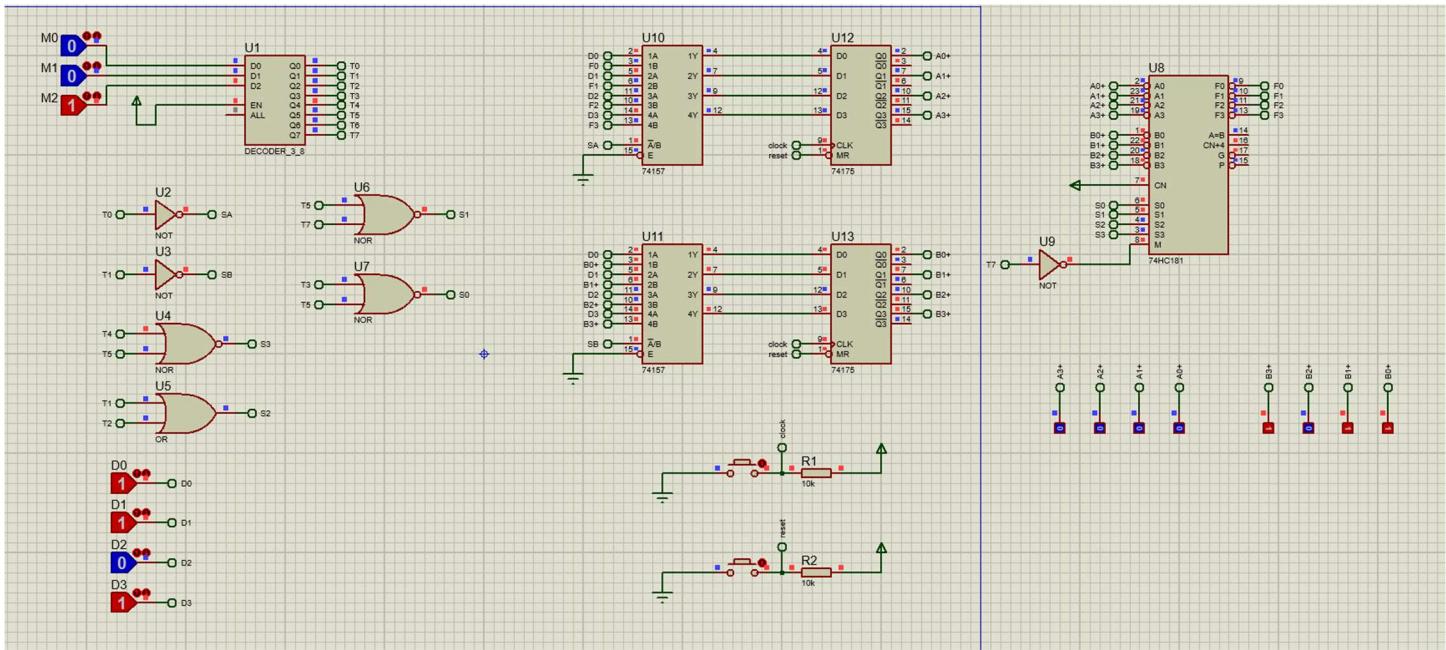
به ازای ورودی ۱۰ برای M0-M2 عدد A و B ثابت می‌مانند:



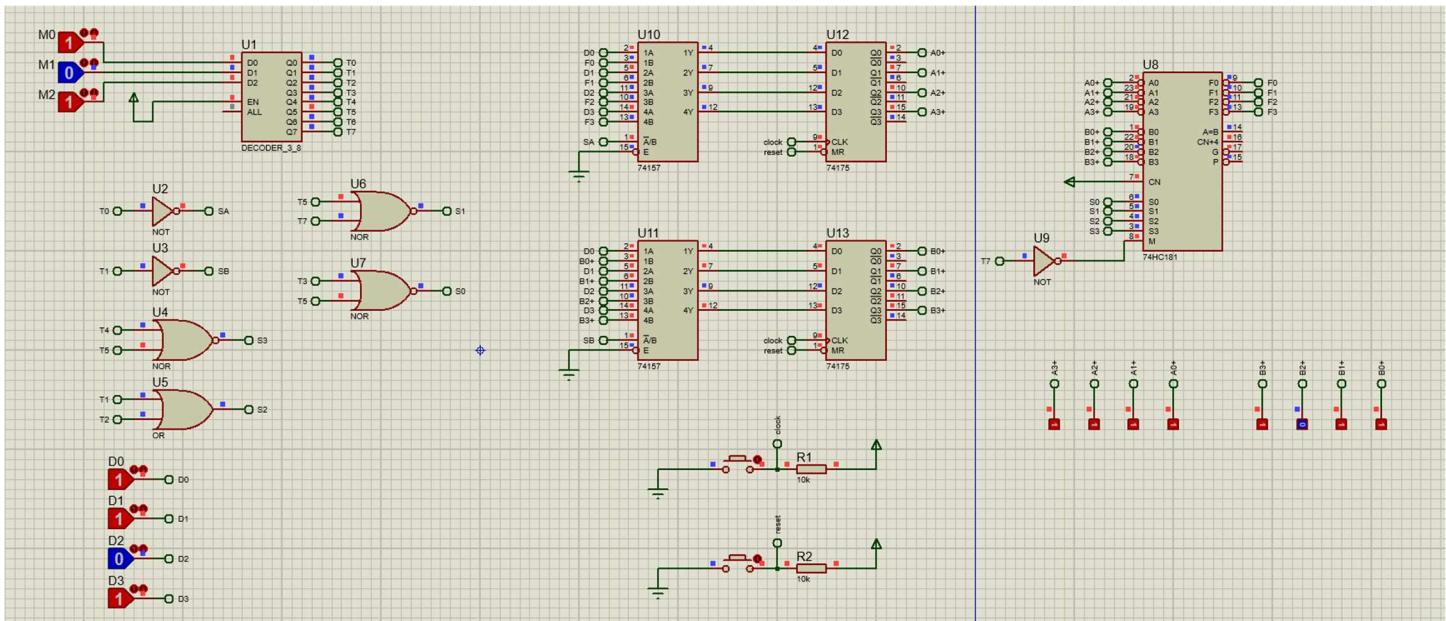
به ازای ورودی ۱۱ برای M0-M2 عدد B در A ریخته می‌شود:



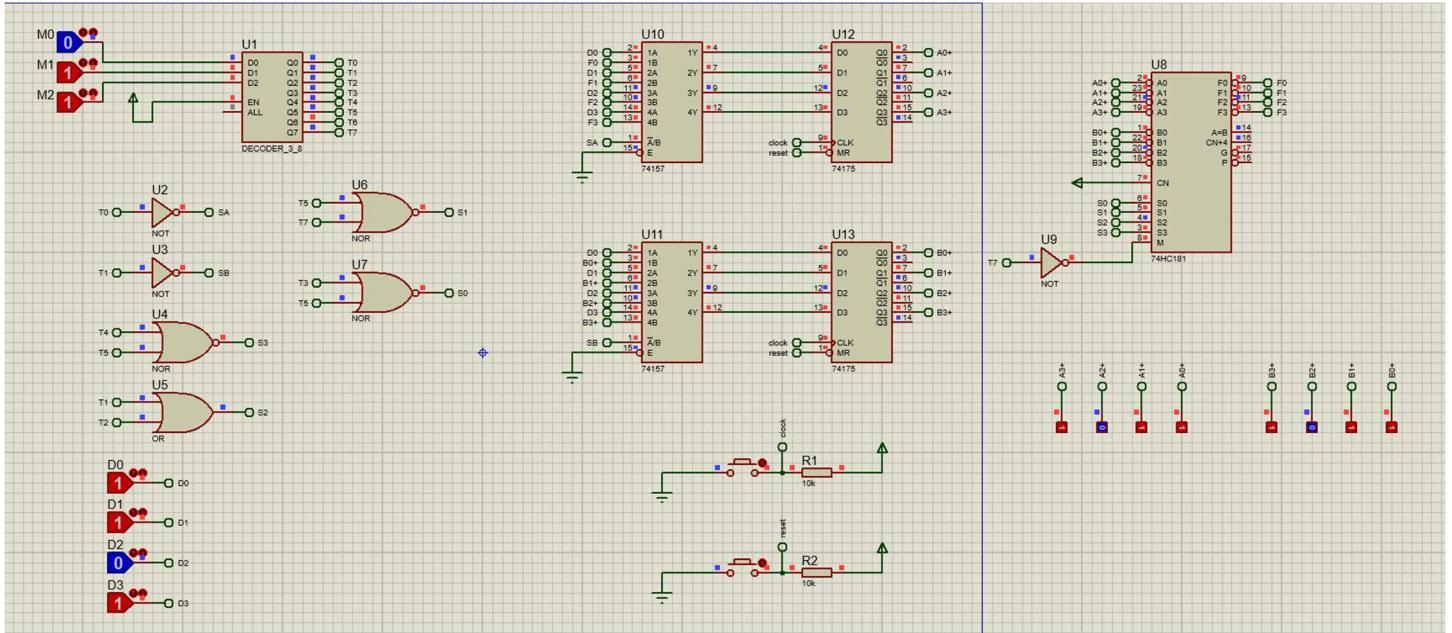
به ازای ورودی M0-M2 عدد A صفر می‌شود:



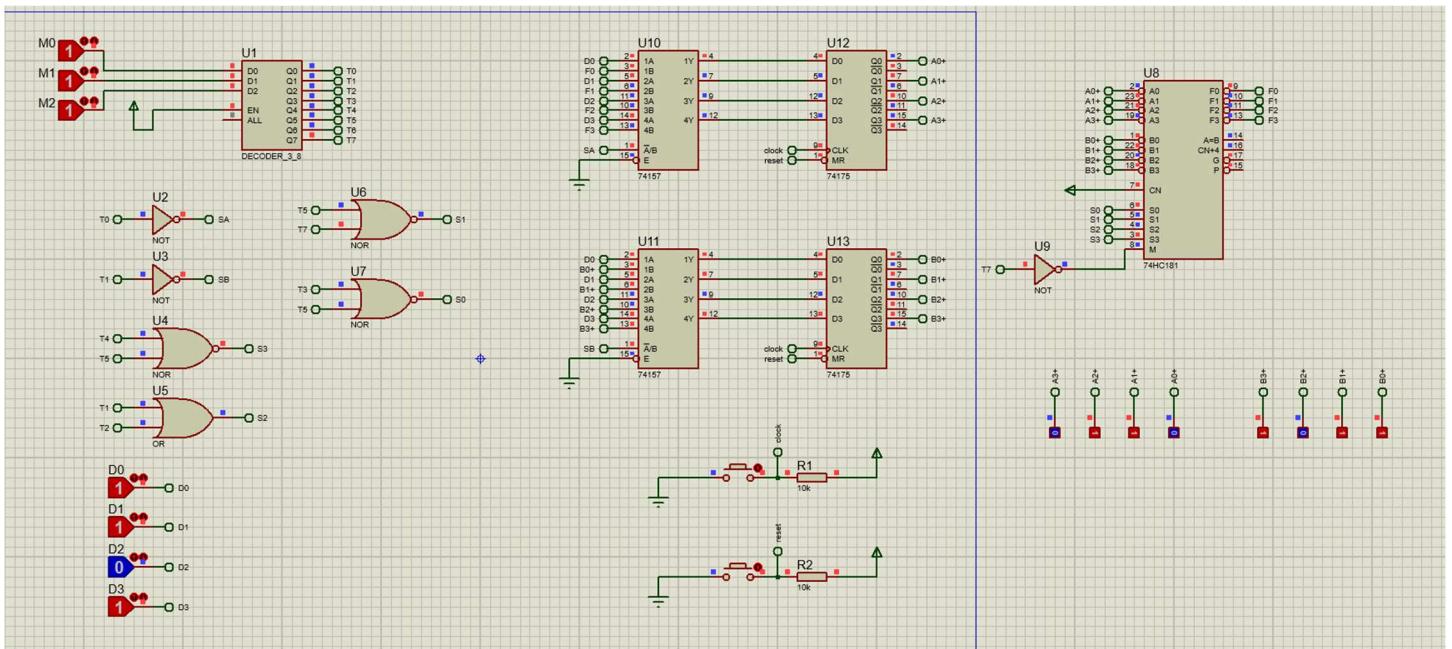
به ازای ورودی M0-M2 معکوس A در آن ریخته می‌شود:

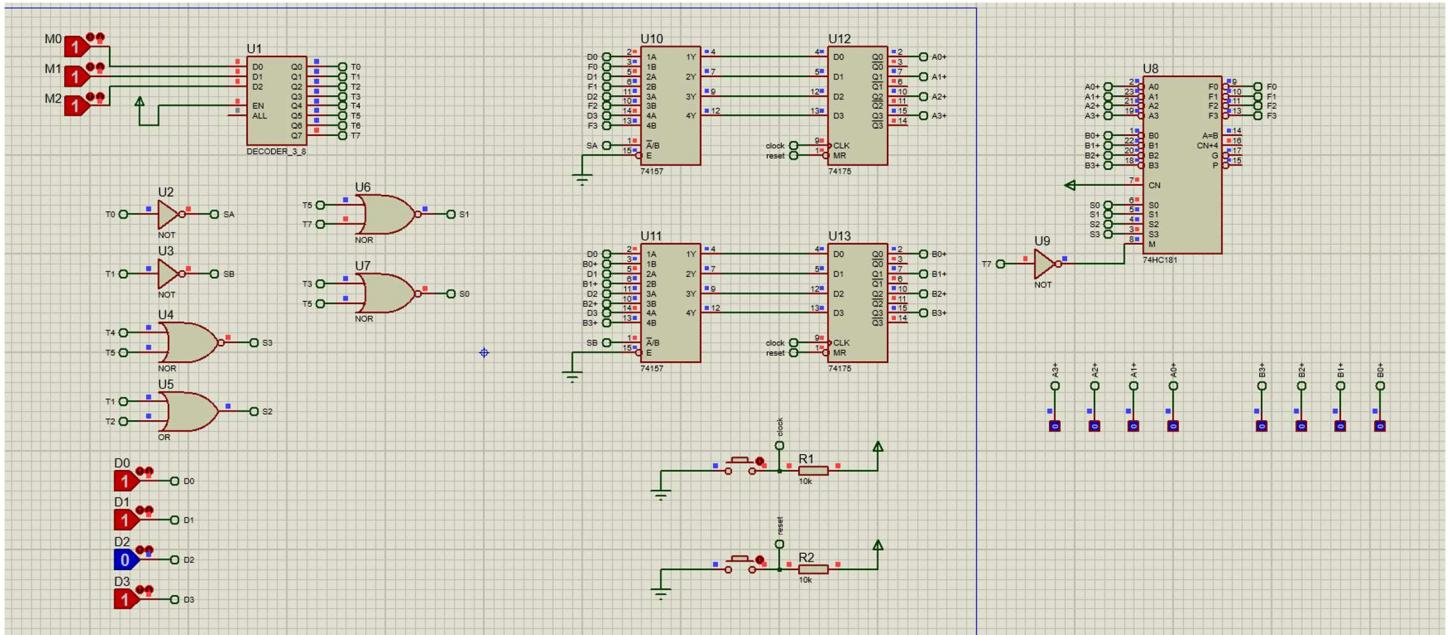


به ازای ورودی ۱۱۰ برای M0-M2 اند شده‌ی A و B در A قرار می‌گیرد:



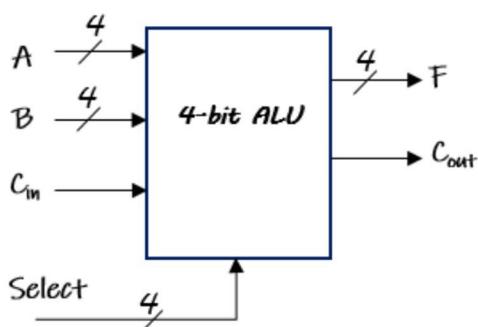
به ازای ورودی ۱۱۱ برای M0-M2 جمع A و B در A قرار می‌گیرد:





ساخت مدار داخلی ALU

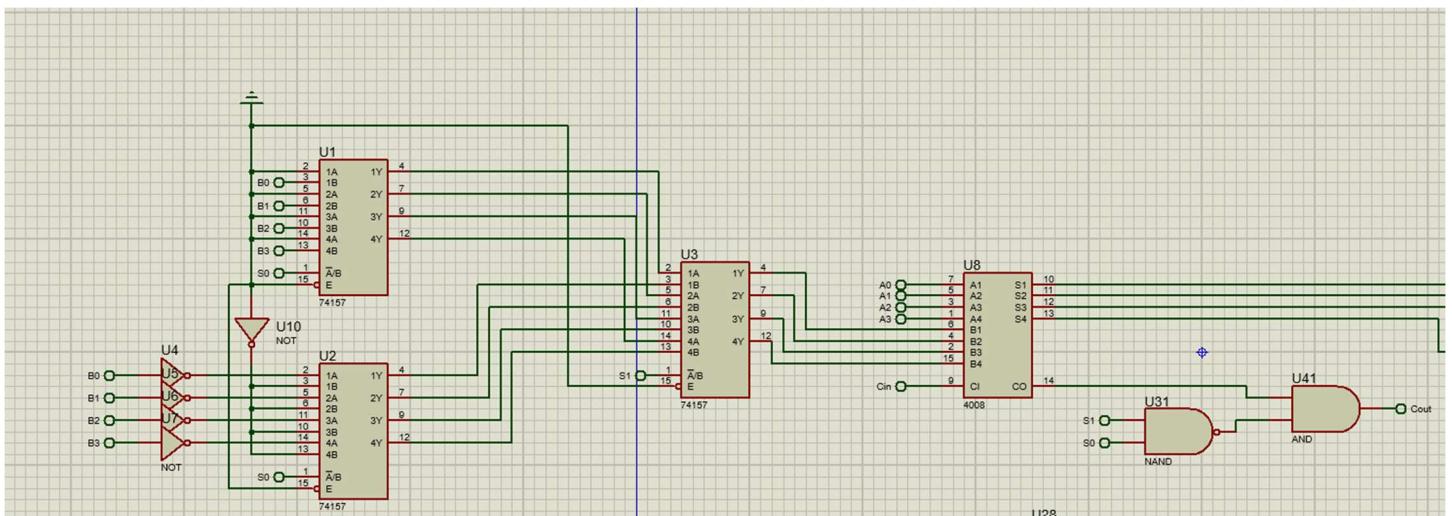
در این قسمت از ما خواسته شده تا مطابق جدول زیر مدار ALU ۴ بیتی را بسازیم. همانطور که در تصویر سمت چپ مشاهده می‌شود این مدار دو عدد ۴ بیتی A و B را ورودی می‌گیرد و با توجه به ورودی‌های Cin و چهار خط آدرس یک عملیات خاص روی آنها انجام می‌دهد.



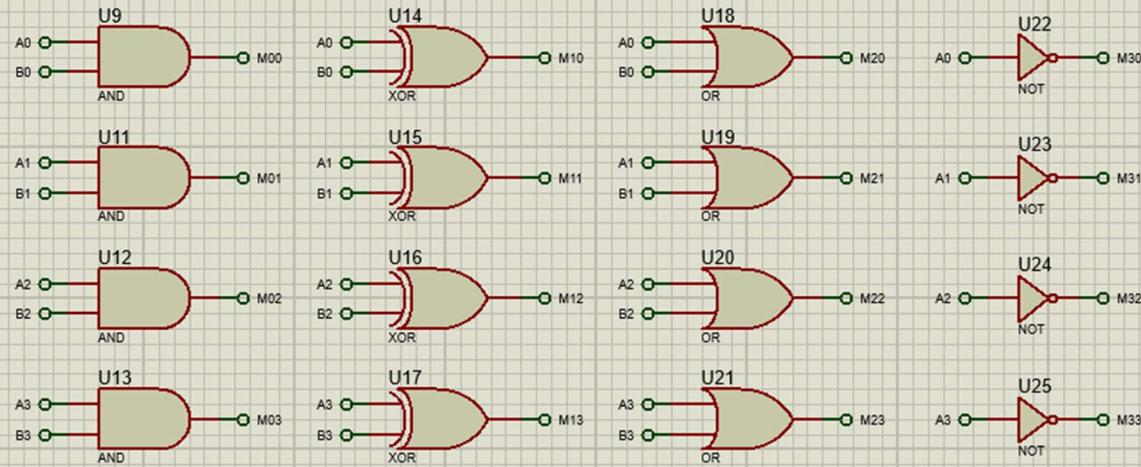
Operation select							
S ₃	S ₂	S ₁	S ₀	C _{in}	Operation	Function	
0	0	0	0	0	F = A	Transfer A	
0	0	0	0	1	F = A + 1	Increment A	
0	0	0	1	0	F = A + B	Addition	
0	0	0	1	1	F = A + B + 1	Add with carry	
0	0	1	0	0	F = A + \bar{B}	Subtract with borrow	
0	0	1	0	1	F = A + \bar{B} + 1	Subtraction	
0	0	1	1	0	F = A - 1	Decrement A	
0	0	1	1	1	F = A	Transfer A	
0	1	0	0	x	F = A \wedge B	AND	
0	1	0	1	x	F = A \vee B	OR	
0	1	1	0	x	F = A \oplus B	XOR	
0	1	1	1	x	F = \bar{A}	Complement A	
1	0	x	x	x	F = shr A	Shift right A into F	
1	1	x	x	x	F = shl A	Shift left A into F	

اگر به جدول صفحه قبل نگاه کنیم متوجه می‌شویم که اعمال مربوط به سطرهای اول تا هشتم با یک جمع کننده ۴ بیتی انجام می‌شود. همانطور که مشخص است ورودی اول جمع کننده باید عدد A و ورودی Cin آن نیز باید به ورودی Cin مدار متصل باشد. برای عدد دومی که به جمع کننده می‌دهیم، ۴ حالت وجود دارد که به ترتیب صفر، B، B' و ۱۱۱ است. در حالت چهارم به جای منهای یک کردن، عدد A را با مکمل عدد ۱ که ۱۱۱ است جمع می‌کنیم. در هشت سطر اول جدول صفحه قبل فقط S0 و S1 تغییر می‌کنند که Cin را همان ورودی Cin جمع کننده در نظر گرفتیم. در نتیجه ۸ سطر اول به همان چهار حالتی که برای عدد دوم جمع کننده گفتیم تبدیل می‌شوند که این حالتها با ورودی‌های S0 و S1 متمایز می‌شوند و سایر خطوط آدرس همگی صفر هستند. بدین منظور از ایسی‌های ۷۴۱۵۷ که جمعی از ۴ مالتی‌پلکسر ۲ به ۱ بودند استفاده می‌کنیم. برای این کار ابتدا دو ایسی ۷۴۱۵۷ قرار می‌دهیم. برای ایسی اول ورودی‌های اول هر جفت ورودی را به زمین و ورودی‌های دوم را به بیت‌های B متصل می‌کنیم. برای ایسی دوم ورودی‌های اول هر جفت ورودی را به یک و ورودی‌های دوم را به معکوس بیت‌های B متصل می‌کنیم. خط آدرس هر دو مالتی‌پلکسر به S0 متصل هستند. درنهایت یک ایسی مالتی‌پلکسر دیگر قرار می‌دهیم و ورودی‌های اول هر جفت ورودی آن را به خروجی‌های مالتی‌پلکسر اول و ورودی‌های دوم آن را به خروجی‌های مالتی‌پلکسر دوم متصل می‌کنیم و در آخر خروجی‌های آخرین مالتی‌پلکسر را به عنوان عدد دوم ورودی می‌دهیم. خط آدرس این ایسی به S1 متصل است. این قسمت از مدار در تصویر زیر قابل مشاهده است.

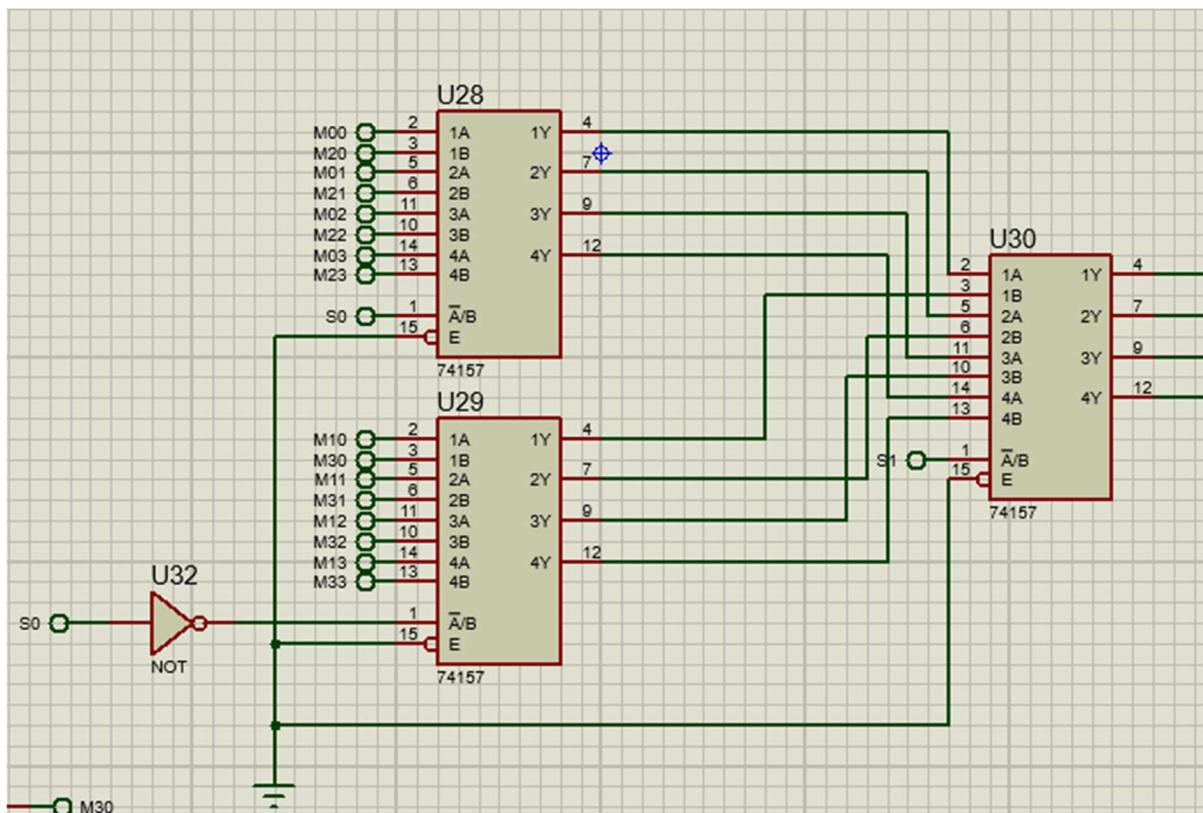
برای خروجی Cout مدار می‌توانیم از Cout جمع کننده استفاده کنیم ولی باید دقت کنیم که در بالا توضیح داده شد عدد A را با ۱۱۱ جمع کردیم به همین دلیل Cout را با نند شده S0 و S1 اند می‌کنیم تا اگر در حالت چهارم بودیم خروجی Cout صفر شود.



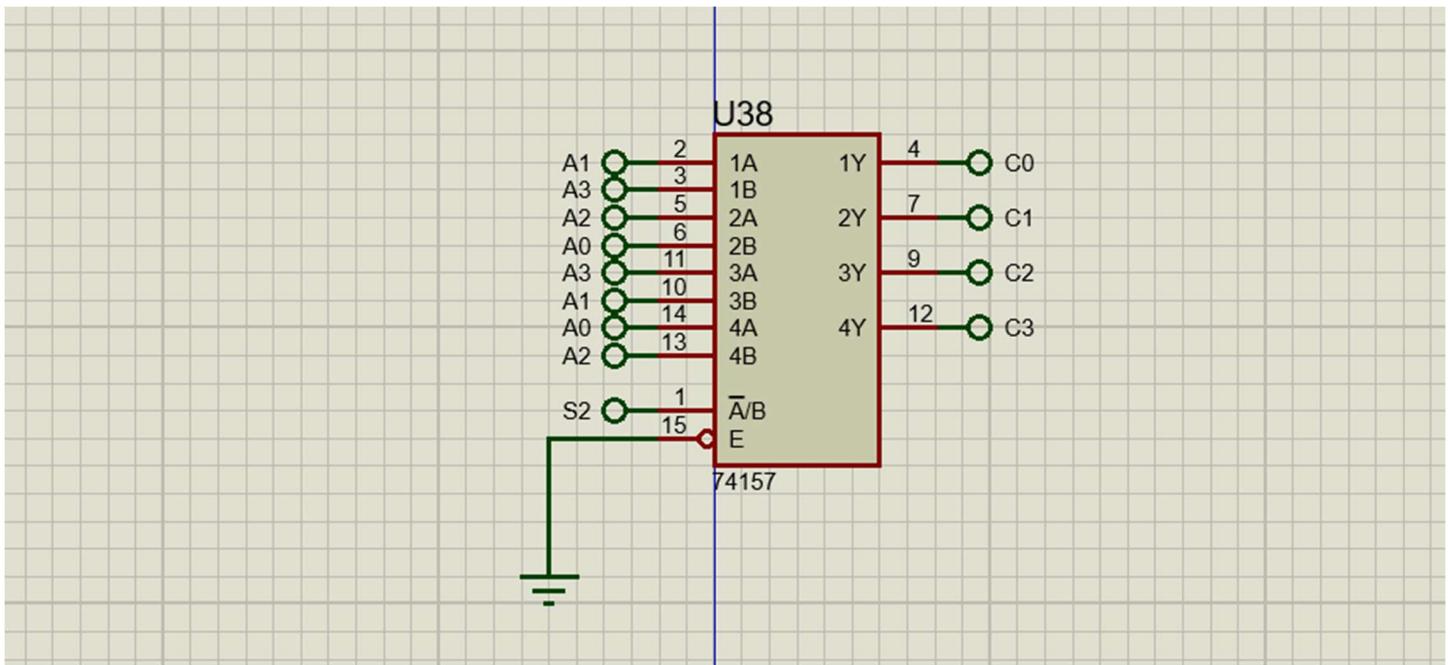
سطر نهم تا دوازدهم جدول صفحه قبل مربوط به عملیات منطقی روی دو عدد A و B است. این ۴ حالت را نیز همانند بالا با ۳ ایسی مالتی‌پلکسر هندل می‌کنیم. ابتدا این ۴ حالت را با گیت‌های ساده می‌سازیم که در تصویر صفحه بعد مشاهده می‌شود.



مالتیپلکسر های مربوط به این ۴ حالت نیز در تصویر زیر مشاهده می شود. حالت های اند و اور را مالتیپلکسر U28 و XOR و نات را مالتیپلکسر U29 انجام می دهد و خطوط آدرس ایسی اول به S0 و دومی به S1 متصل است. در نهایت خروجی های این دو ایسی به یک ایسی مالتیپلکسر دیگر متصل می شوند که خط آدرس آن به S1 به S1 متصل است.

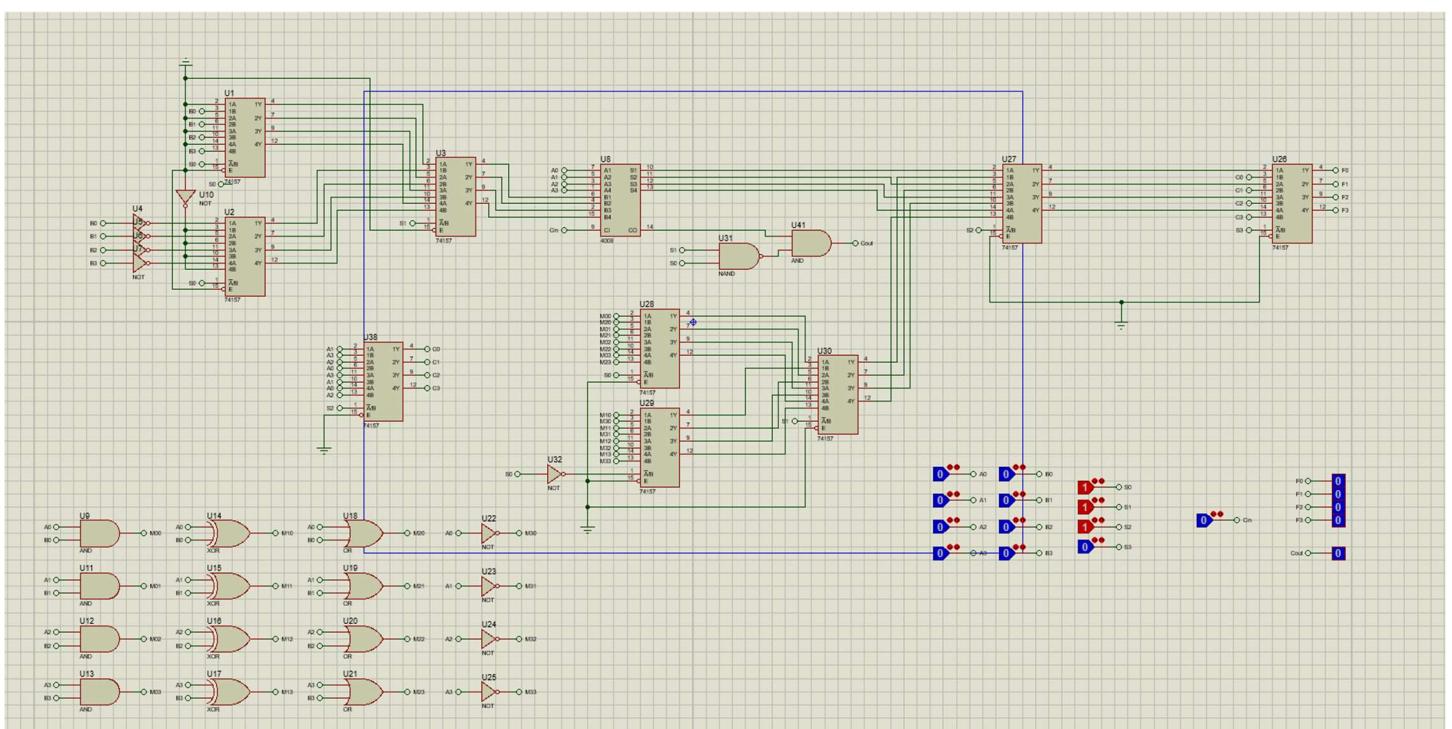


دو سطر باقیمانده از جدول دستورکار مربوط به شیفت به راست و شیفت به چپ عدد A است. کافی است برای این دو حالت نیز یک ایسی مالتیپلکسر قرار دهیم و به صورت زیر به آن ورودی می‌دهیم تا شیفت به راست و شیفت به چپ را هندل کند.



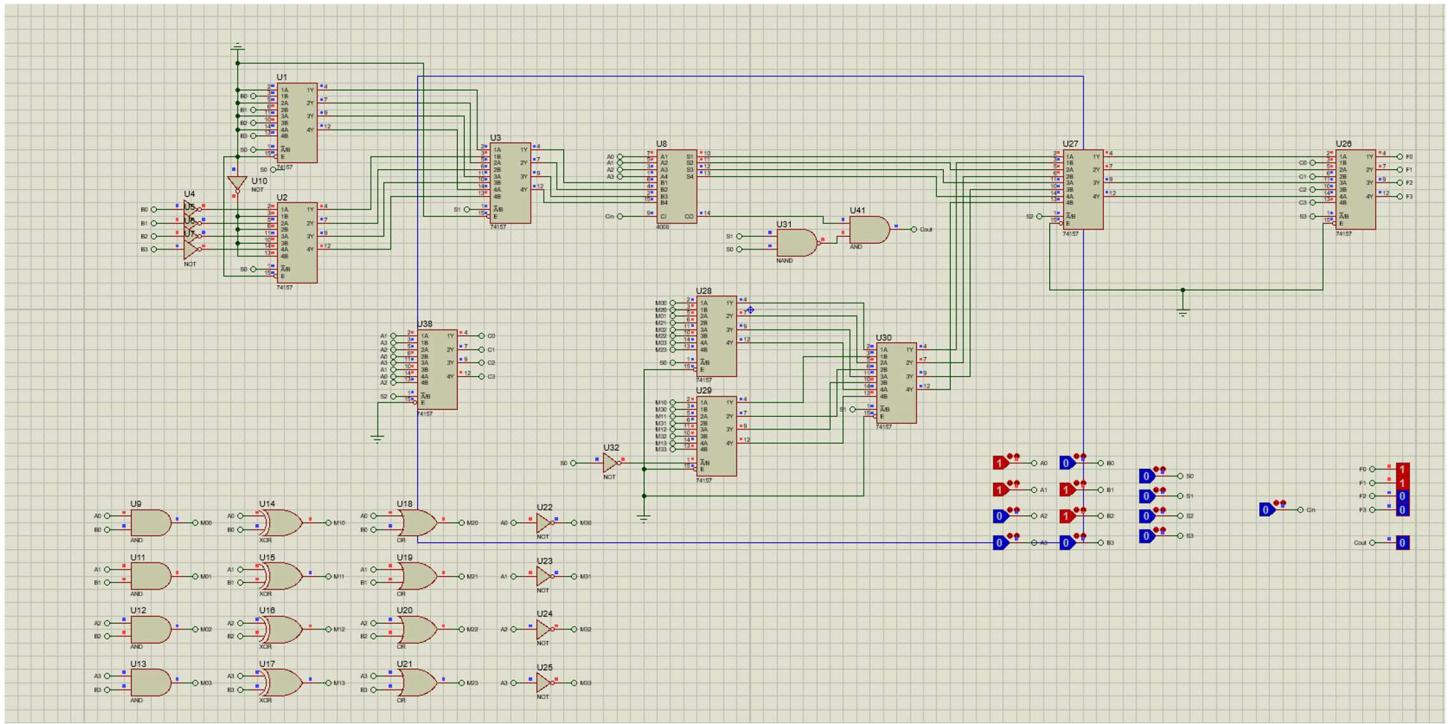
خروجی‌های جمع کننده و مالتیپلکسر نهایی بالا (U30) در تصویر صفحه قبل) باید به یک ایسی مالتیپلکسر دیگر متصل شوند که خط آدرس آن به S2 متصل است. سپس خروجی‌های این ایسی و ایسی‌ای که در تصویر بالا مشاهده می‌شود باید به یک ایسی مالتیپلکسر دیگر ورودی داده شوند که خط آدرس آن به S3 متصل است و خروجی‌های مدار از این ایسی گرفته می‌شود.

در نهایت تصویر کلی مدار به صورت زیر می‌شود که خروجی‌های مدار در سمت راست مشاهده می‌شوند.

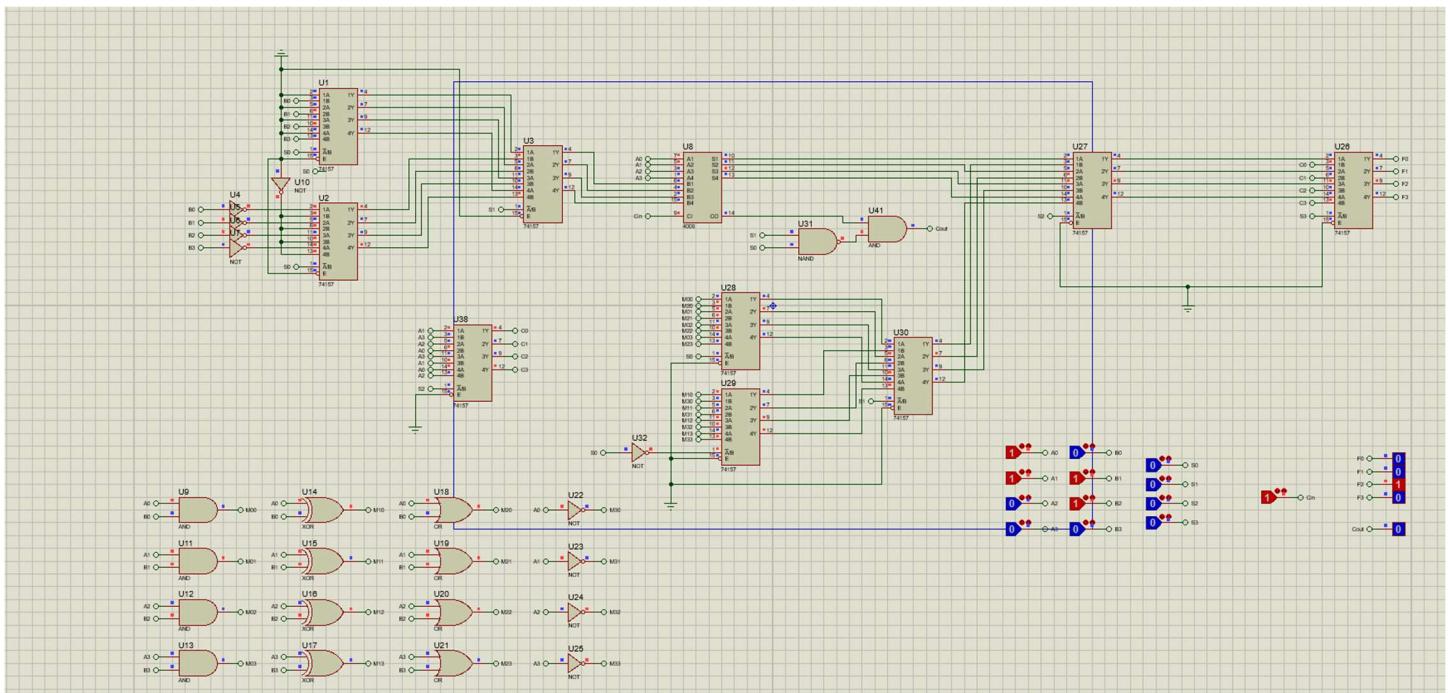


حال به بررسی عملکرد مدار به ترتیب سطرهای جدول دستورکار و به ازای عدد ۱۱۰ برای A و ۱۱۰ برای B میپردازیم.

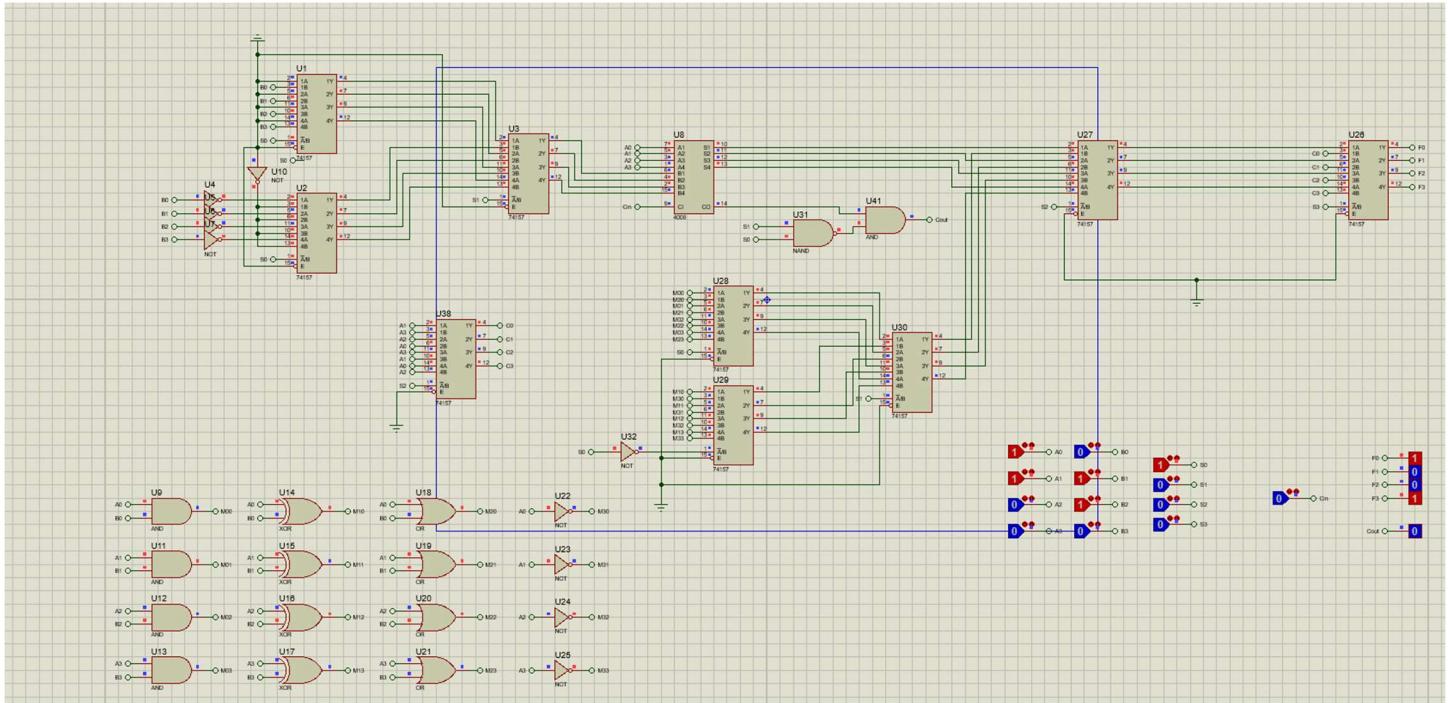
:Cin در خروجی به ازای ۰۰۰۰ برای S0-S3 و ۰ برای Cin شدن A



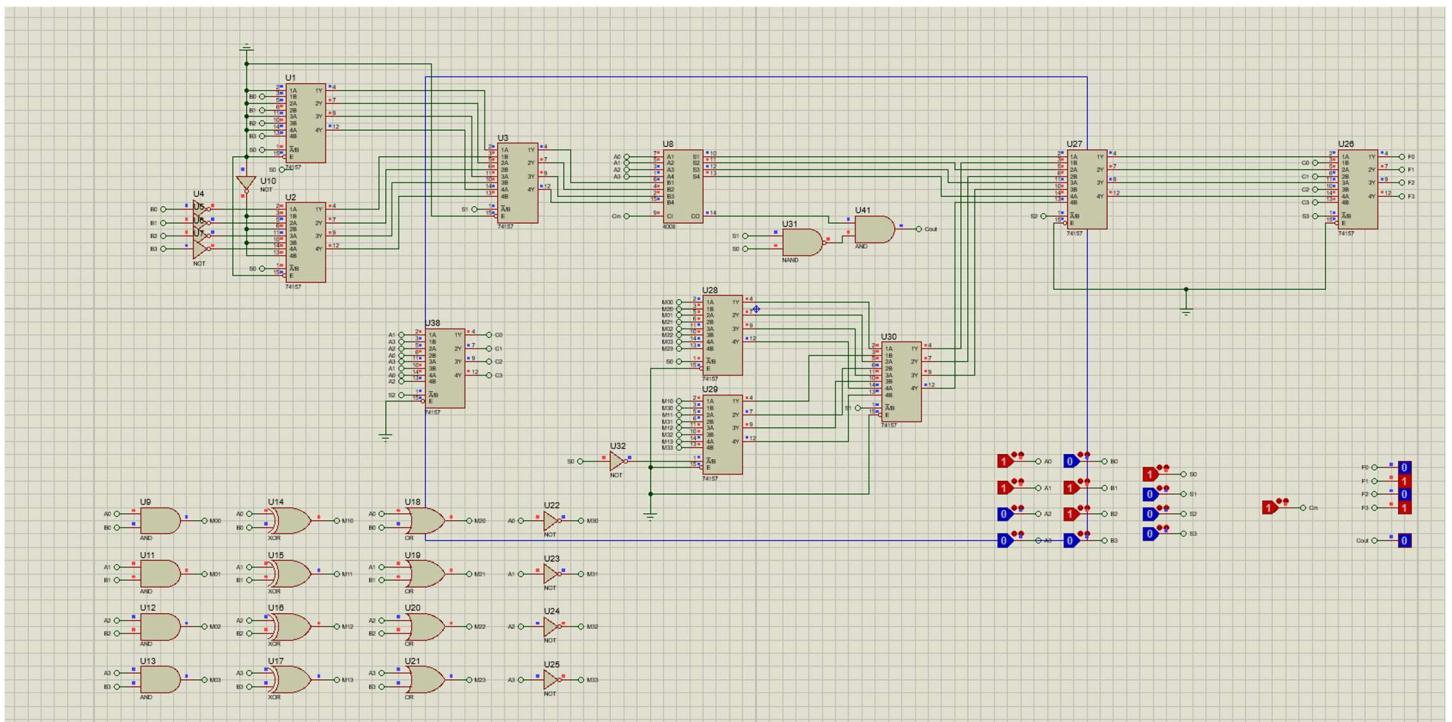
:Cin در خروجی به ازای ۰۰۰۰ برای S0-S3 و ۱ برای Cin شدن A



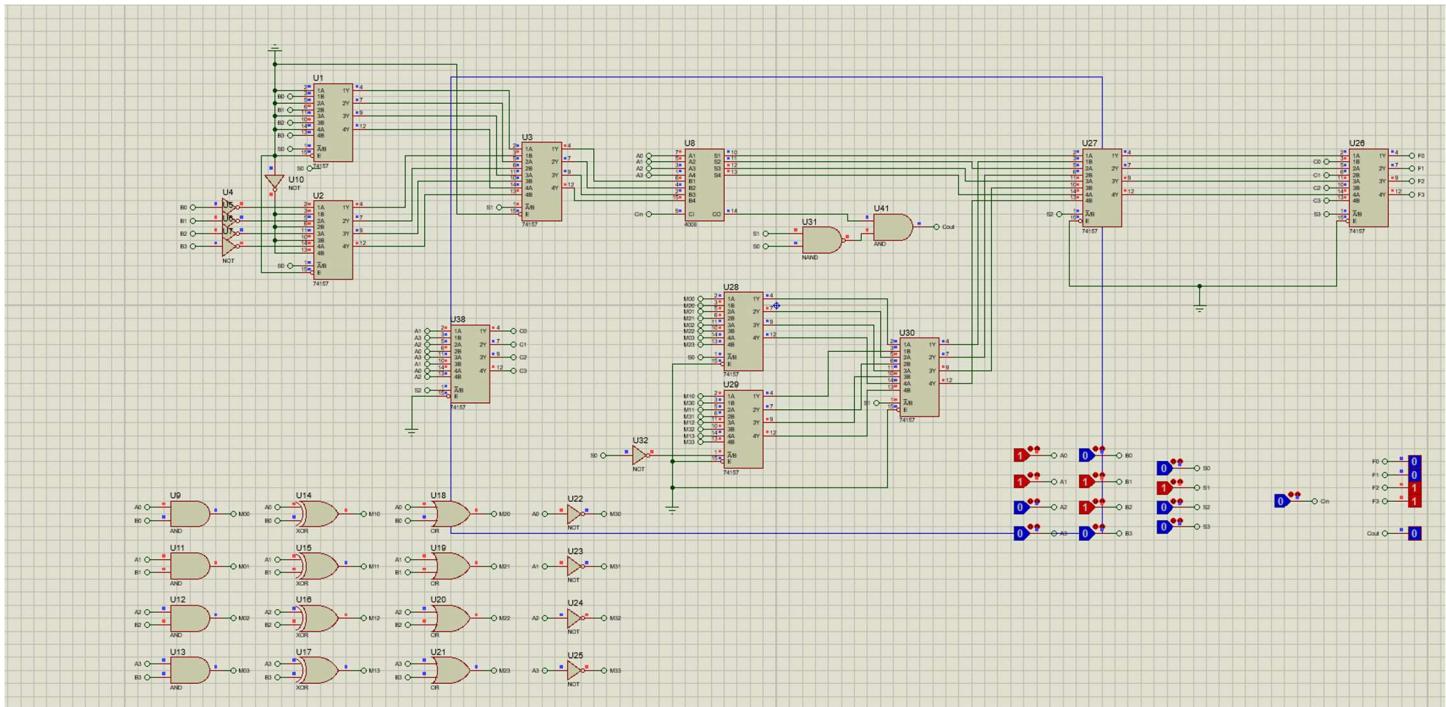
جمع شدن A با Cin و B در خروجی به ازای ۱۰۰۰ برای S0-S3 و ۰ برای Cin



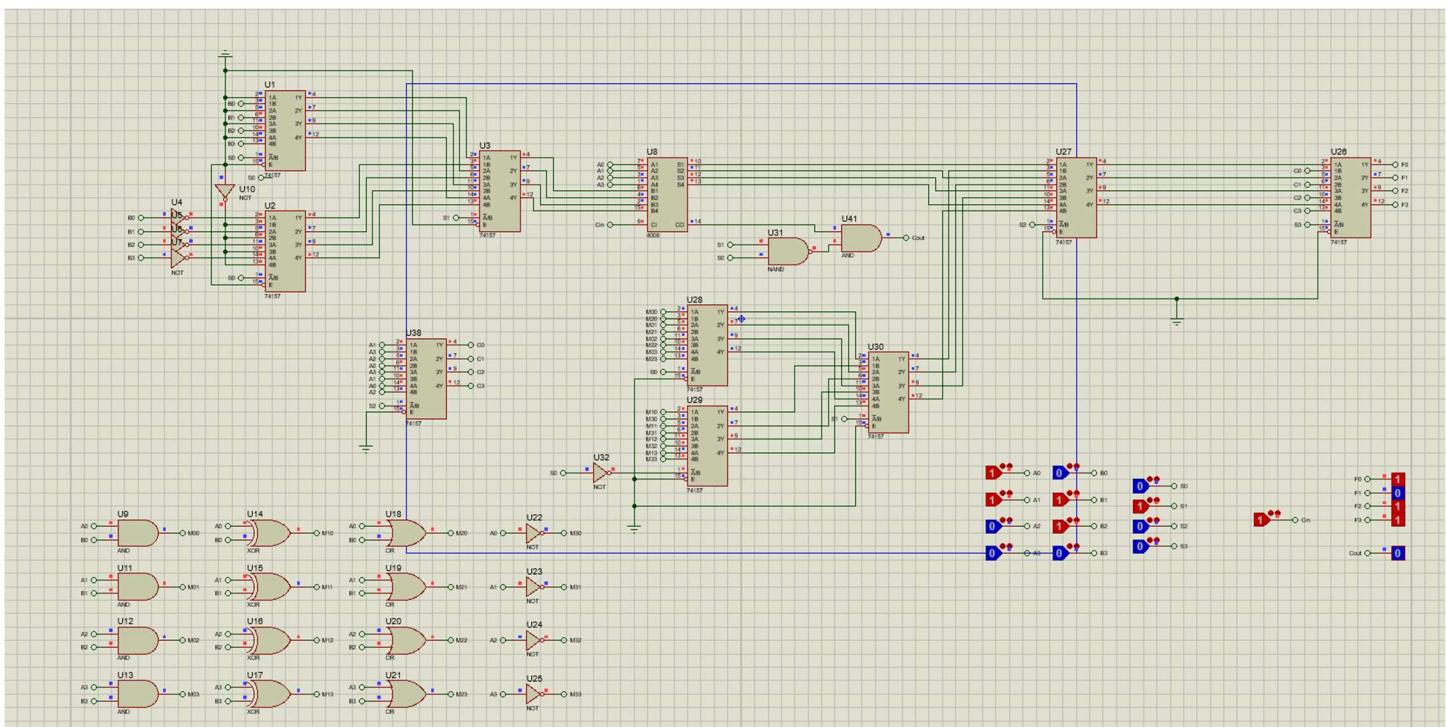
جمع شدن A با Cin و B به ازای ۱۰۰۰ برای S0-S3 و ۱ برای Cin



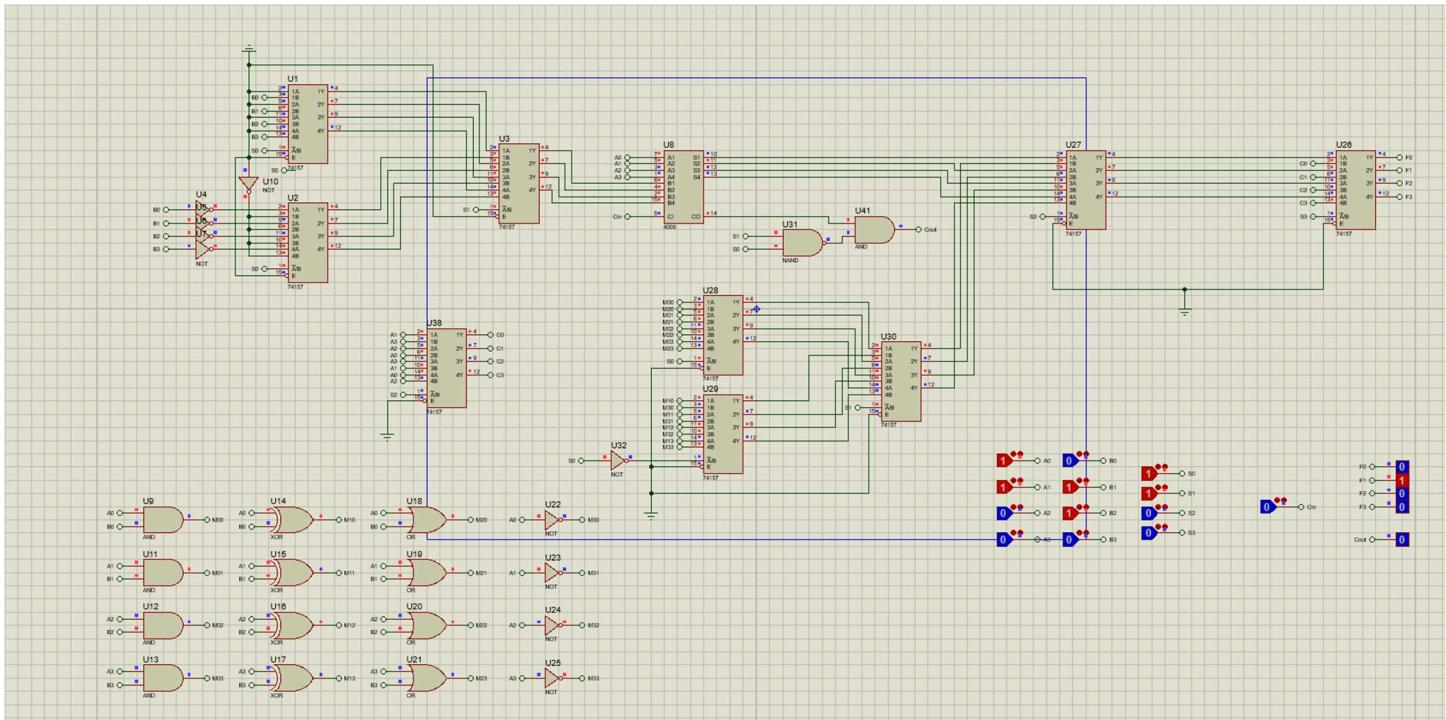
جمع شدن A با Cin و B' به ازای S0-S3 و ۰ برای Cin A



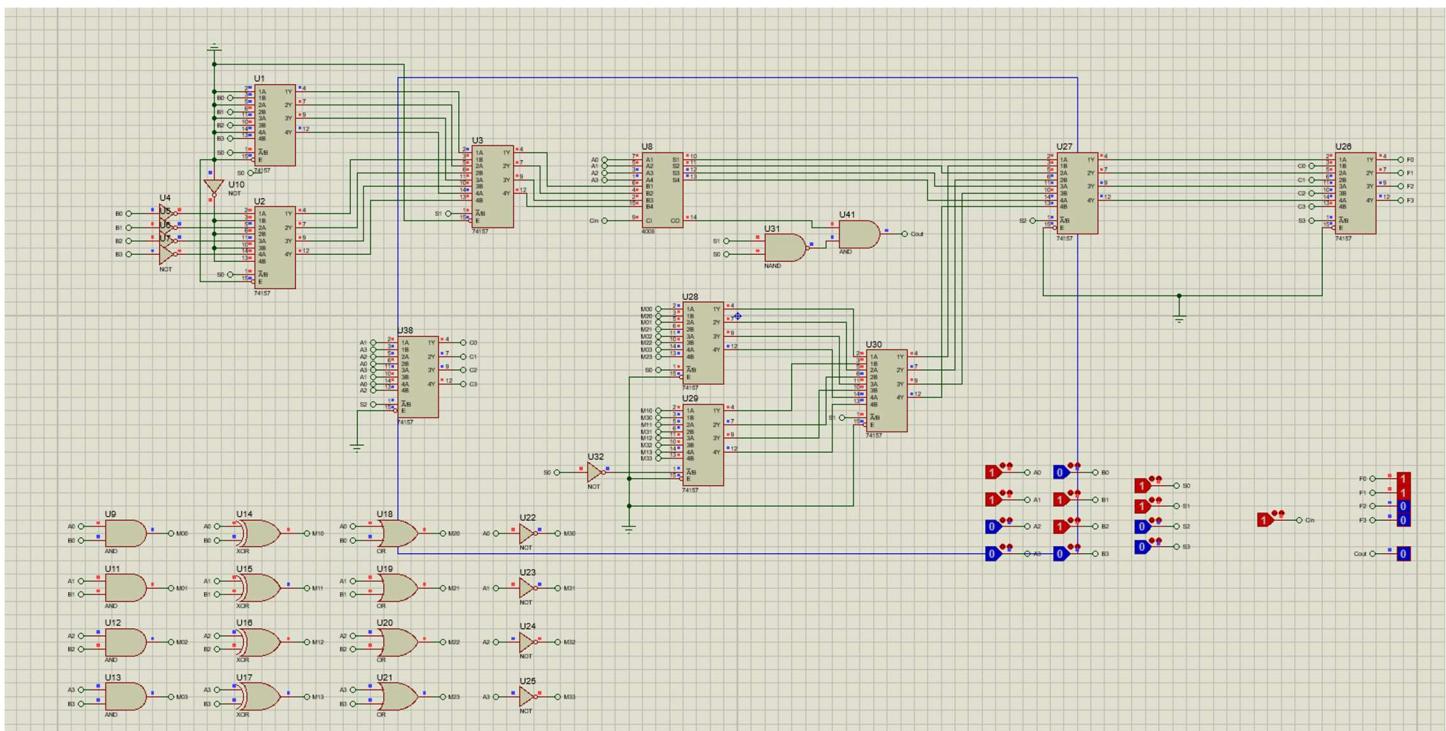
جمع شدن A با Cin و B' به ازای S0-S3 و ۱ برای Cin A



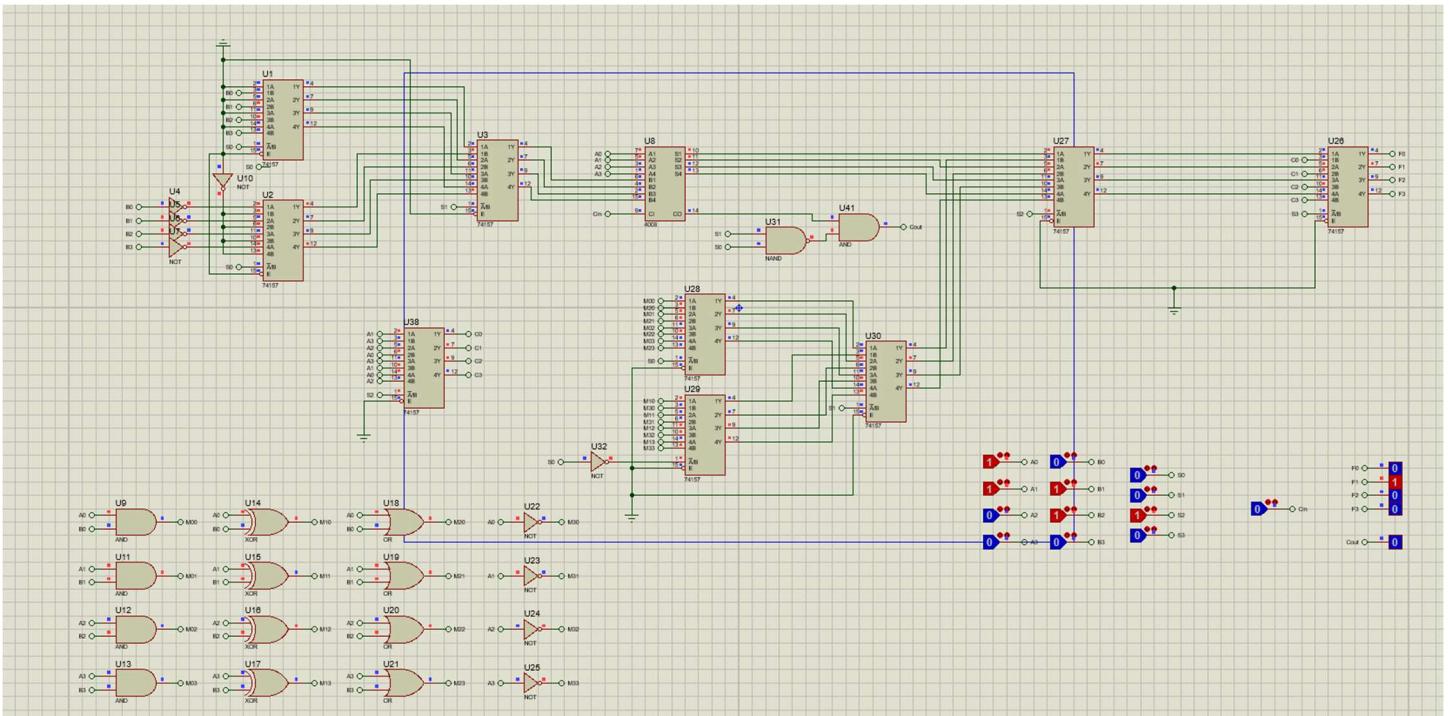
:Cin در خروجی به ازای ۱۱۰۰ برای Cin-S3 + A-1 و برای



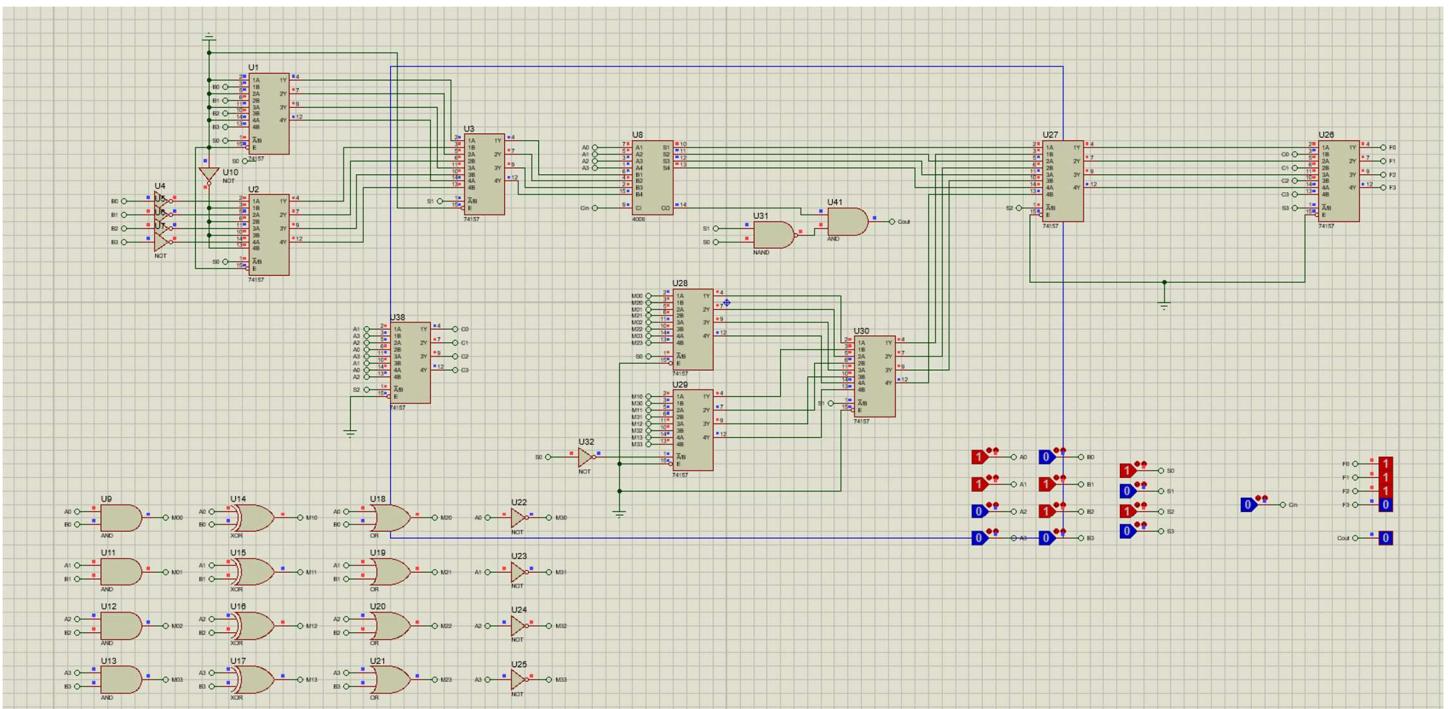
:Cin در خروجی به ازای ۱۱۰۰ برای Cin-S3 + A-1 و برای



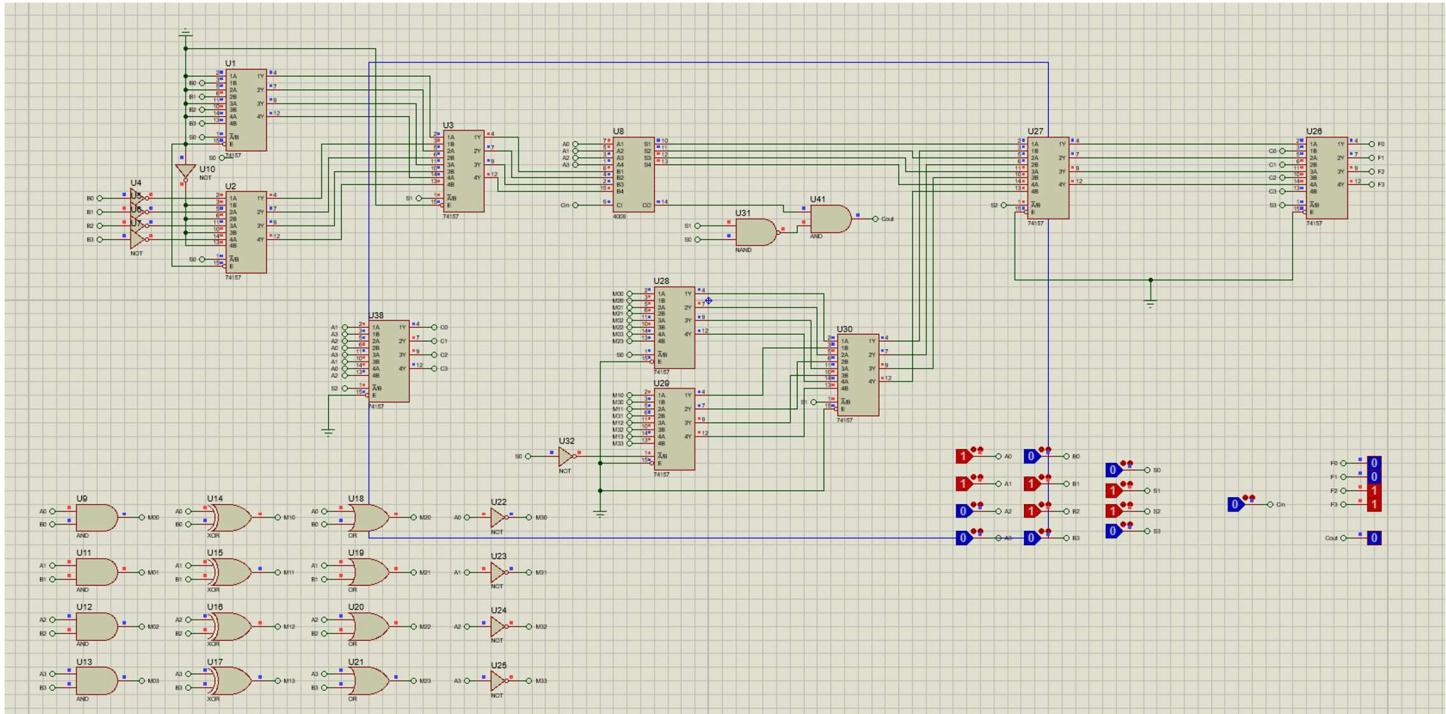
:S0-S3 به ازای ۱۰۰ برای A and B



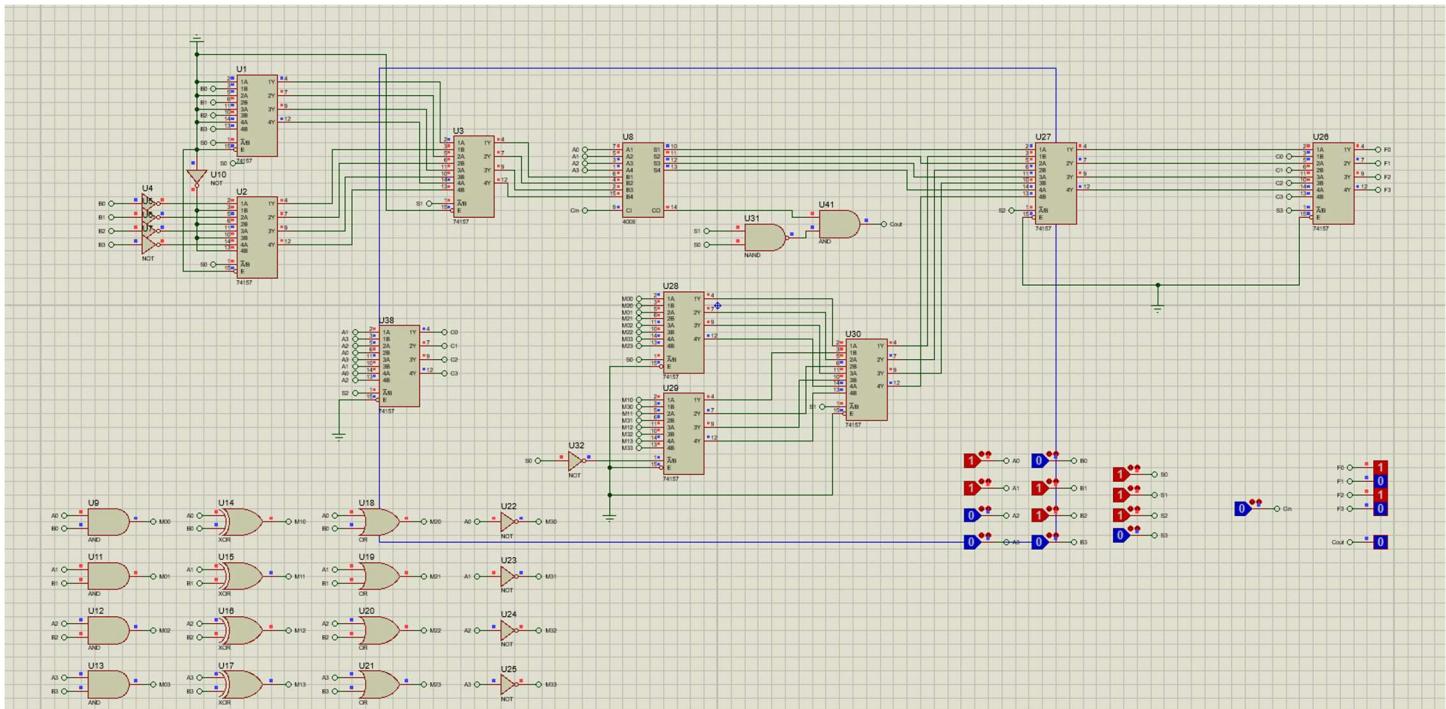
:S0-S3 به ازای ۱۰۱ برای A or B



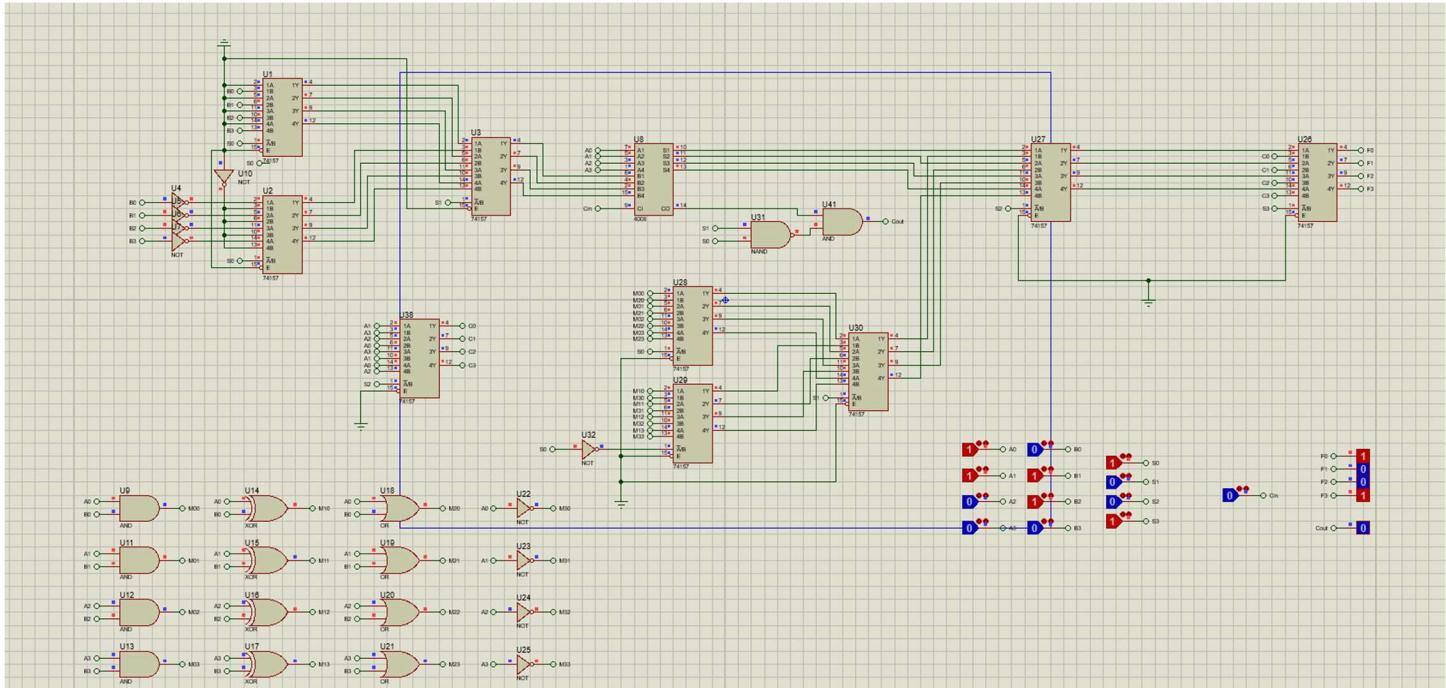
S0-S3 به ازای A xor B



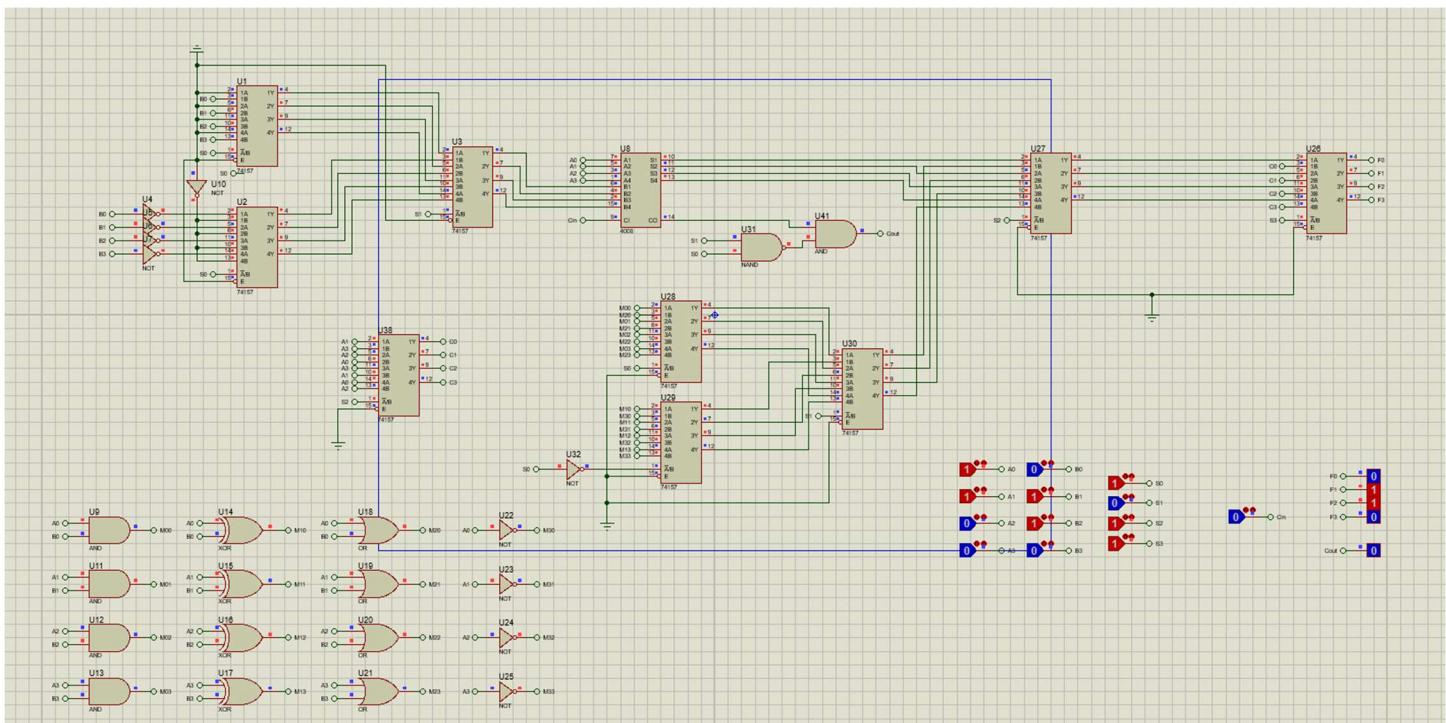
S0-S3 به ازای A'



شیفت A به راست به ازای 10XX برای S0-S3



شیفت A به چپ به ازای 11XX برای S0-S3



نتیجه‌گیری

در این آزمایش با ایسی ALU آشنا شدیم و توانستیم با استفاده از این ایسی اعمال حسابی و منطقی را انجام دهیم. همانطور که می‌دانیم CPU کامپیوتر یک واحد محاسبات و منطق دارد که این آزمایش در واقع مقدمه‌ای برای آن بود. در قسمت دوم این آزمایش نیز با استفاده از ایسی‌های ۷۴۱۵۷ توانستیم حالت بندی‌های زیادی را هندل کنیم. از این قسمت می‌توان نتیجه گرفت که زمانی که یک مسئله حالت‌های زیادی دارد می‌توان با توجه به تغییرات ورودی‌ها و با استفاده از ایسی‌های مالتی‌پلکسر این حالت‌بندی‌ها را هندل کرد.