بسمه تعالی



گزارش کار دوم آزمایشگاه معماری کامپیوتر

ضرب‌کننده ممیز ثابت

استاد

دکتر سربازی آزاد

نویسندگان

امیررضا سلیمان‌بیگی – آرین احدی‌نیا – سایه جارالهی

دانشگاه صنعتی شریف

تابستان ۱۴۰۱

فهرست مطالب

[1- هدف آزمایش 3](#_Toc110363309)

[1-1 هدف آزمایش 3](#_Toc110363310)

[2- الگوریتم 3](#_Toc110363311)

[3- ماژول‌های استفاده شده 3](#_Toc110363312)

[3-1 جمع کننده تک بیتی(Full Adder) 4](#_Toc110363313)

[3-2 جمع کننده 4 بیتی(4-bit Adder) 4](#_Toc110363314)

[3-3 جمع کننده و تفریق کننده 8 بیتی 5](#_Toc110363315)

[3-4 ماژول‎‌های آماده 6](#_Toc110363316)

[3-5 کنترل یونیت (control unit) 7](#_Toc110363317)

[3-6 ماژول نهایی ضرب کننده ممیز ثابت 8](#_Toc110363318)

[4- نتایج بدست آمده از آزمایش 10](#_Toc110363319)

1. هدف آزمایش
   1. هدف آزمایش

در این آزمایش، ضرب کننده ممیز ثابت طراحی می‌شود. ضرب دو عدد 4 بیتی به نام‌های A و B انجام شده و حاصل پس از چند کلاک، در خروجی نمایش داده می‌شود. همچنین پس از پایان عملیات، سیگنال end نیز مقدار یک را به خود می‌گیرد تا پایان عملیات ضرب و آماده بودن جواب مشخص شود. برای شروع عملیات نیز سیگنال start باید مقدار یک را داشته باشد تا عملیات شروع شود. خروجی نهایی در 8 بیت نمایش داده میشود. عملیات ضرب با درنظر گرفتن علامت انجام می‌شود و خروجی نهایی نیز علامتدار است. علامت همه اعداد در سیستم مکمل دو تعریف شده است.

1. الگوریتم

در این تمرین، الگوریتم shift and add برای ضرب دو عدد استفاده شده است. این الگوریتم مشابه با ضرب ساده است. برای خروجی یک ثبات 8 بیتی در نظر گرفته میشود که در ابتدا مقدار صفر را دارد. در الگوریتم، در لبه بالارونده هر کلاک، مضروب یک واحد به چپ شیفت میخورد و مضروب فیه یک واحد به سمت راست شیفت می‌خورد. در تمام شیفت‌هایی که انجام می‌شود مقدار صفر وارد ثبات می‌شود. سپس در صورتی که بیت خارج شده از سمت راست مضروب فیه مقدار یک را داشت و علامت مضروب فیه مثبت بود و یا در چهارمین شیفت نبودیم، مضروب با مقدار قبلی جمع می شود. در صورتی که علامت مضروب فیه منفی بود، در چهارمین کلاک که آخرین عملیات باید انجام شود، مقدار قبلی از مضروب کم می‌شود تا پاسخ نهایی در سیستم مکمل دو صحیح باشد.

1. ماژول‌های استفاده شده

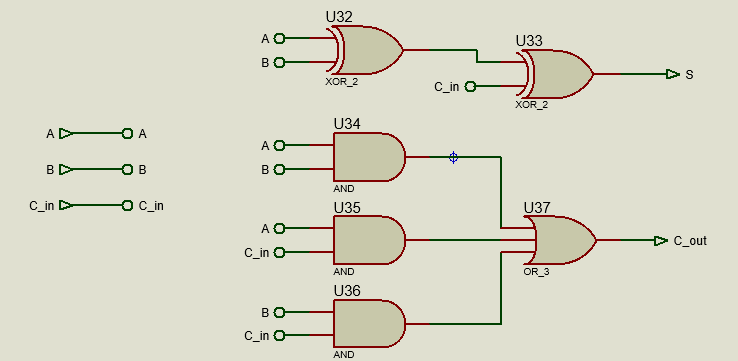
به منظور سادگی در طراحی و همینطور اشکال‌زدایی،‌ این آزمایش را به صورت سلسله مراتبی انجام می‌دهیم. در ادامه تمامی بلوک‌ها را شرح می‌دهیم.

* 1. جمع کننده تک بیتی(Full Adder)

بلوک Full Adder مطابق شکل 1 طراحی شده است. باتوجه به درس مدار منطقی، در صورتی که سه ورودی a, b , c\_in داشته باشیم، مقدار خروجی s و carry که همان c\_out است مطابق روابط زیر محاسبه می‌شود.

S = c\_in xor (a xor b)

C\_out = (a.b) + (a.c\_in) + (b.c\_in)

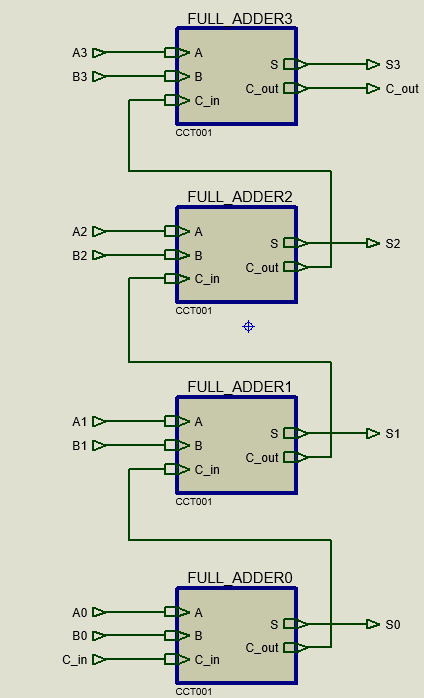


شکل 1

* 1. جمع کننده 4 بیتی(4-bit Adder)

بلوک 4-bit Adder را مطابق شکل 2 طراحی شده است. این بلوک با cascade کردن 4 بلوک

Full Adder ساخته شده است. ورودی‌های این بلوک، دو عدد 4 بیتی a3a2a1a0 و b3b2b1b0، و مقدار carry ورودی c\_in است. خروجی‌های این بلوک نیز مقدار حاصل جمع 4 بیتی s3s2s1s0 و carry خروجی c\_out است. در این بلوک از چهار Full Adder استفاده شده است، به طوری که ورودی‌های Full\_Adder\_0، carry ورودی c\_in و بیت‌های a0 و b0 است. به همین ترتیب ورودی‌های Full\_Adder\_i مقدار بیت‌های ai و bi و خروجی c\_out مربوط به Full\_Adder\_i-1 است. این جمع کننده مشابه با آنچه در تمرین قبل بود طراحی شده‌است.



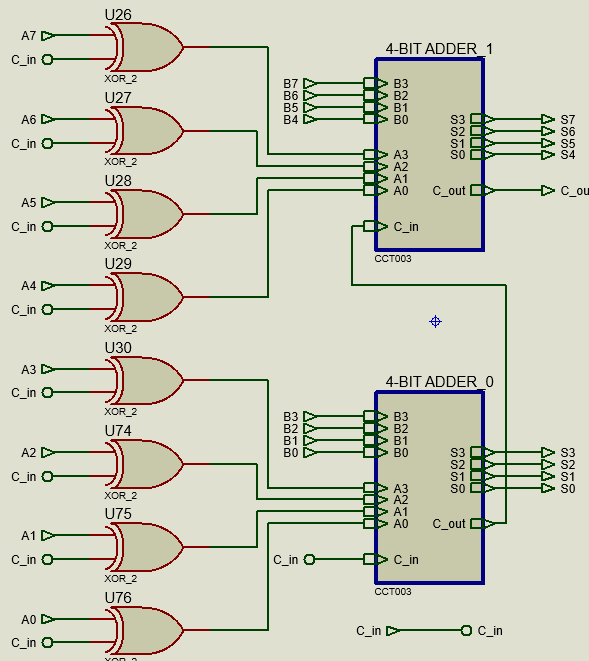
شکل 2

* 1. جمع کننده و تفریق کننده 8 بیتی

برای انجام عملیات‌های جمع و تفریق در کلاک‌های متوالی، یک ماژول جمع‌کننده و یک ماژول تفریق‌کننده مورد نیاز است. برای آسان‌تر شدن پیاده سازی این دو ماژول ادغام شده و یک ماژول جمع-تفریق کننده ۸ بیتی مطابق شکل ۳ پیاده‌سازی شده است. این ماژول دو ورودی هشت بیتی A=A7A6…A0 و B=B7B6…B0 دریافت کرده و بیت کنترلی c\_in را نیز دریافت می‌کند. خروجی این ماژول S=S7…S0 و c\_out است. درصورتی که c\_in=1 باشد عملیات تفریق انجام شده و حاصل A-B در S ریخته می‌شود. در غیر اینصورت حاصل A+B در S ریخته می‌شود. خروجی c\_out نیز carry خروجی است.

در این ماژول از دو جمع کننده ۴ بیتی استفاده شده است. جمع‌کننده 4-bit-adder-0 برای محاسبه ۴ بیت کم ارزشتر و جمع کننده 4-bit-adder-1 برای محاسبه چهار بیت پرارزش است. کری حاصل از جمع بیت‌های کم ارزش به عنوان کری ماژول جمع‌کننده پرارزش وارد میشود.

درصورتی که عملیات تفریق باشد و c\_in=1 ، تمامی بیت‌های B باید نقیض شوند و درغیر اینصورت خود این بیت‌ها باید به عنوان ورودی به جمع کننده ها داده شود. اینکار، با استفاده از گیت‌های xor و مطابق شکل ۳ انجام شده‌است



شکل 3

* 1. ماژول‎‌های آماده

در این تمرین، از دو نوع ثبات برای نگه‌داری و شیفت دادن مقادیر مضروب، مضروب‌فیه و خروجی استفاده شده است. ماژول ۷۴۱۹۴ یک ثبات چهار بیتی با قابلیت شیفت به راست و چپ است که برای نگه‌داری مضروب‌فیه استفاده می‌شود. تعیین مقادیر بیت‌های ورودی این ماژول از روی دیتاشیت و از این [لینک](https://pdf1.alldatasheet.com/datasheet-pdf/view/119310/ETC1/74198.html) به دست آمده است.

از ثبات 8 بیتی 74198 برای نگه‌داری و شیفت مقدار مضروب استفاده شده است. همچنین از این ثبات برای نگه‌داری خروجی P نیز استفاده شده‌است.

* 1. کنترل یونیت (control unit)

این ماژول، وظیفه کنترل کردن عملیات‌های شیفت و جمع و تفریق را بر عهده دارد و حساس به لبه بالارونده ساعت است. علاوه بر ورودی کلاک، مقدار کم ارزش‌ترین بیت مضروب‌فیه که همان QB0 است را نیز ورودی می‌گیرد. همچنین دو ورودی start و or\_B را نیز دریافت می‌کند. Or\_B همان حاصل اور شدن تمام بیت‌های مضروب فیه است. start نیز همان ورودی اصلی مدار است.

خروجی های مدار به طورت زیر است.

* S0A,S1A: مشخص کننده وضعیت دو بیت کنترلی شیفت ثبات مضروب فیه است.
* S0B, S1B: مشخص کننده وضعیت دو بیت کنترلی شیفت ثبات مضروب است.
* S0P,S1P: مشخص کننده وضعیت دو بیت کنترلی شیفت ثبات خروجی نهایی است.
* End: خروجی کلی مدار است که بیانگر تمام شدن عملیات ضرب و آماده بودن خروجی است.
* Reset: برای ضفر کردن مقادیر ثبات خروجی و برگرداندن مدار به حالت اول است.

این ماژول طبق جدول زیر عمل می‌کند و بیت‌های کنترلی را تولید می‌کند.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | Next state | | inputs | | | Present state | |
| S0P | S1P | S0A | S1A | S0B | S1B | End | Q0 | Q1 | start | Or\_B | QB0 | Q0 | Q1 |
| x | x | x | x | x | x | x | 0 | 0 | 0 | x | x | 0 | 0 |
| X | x | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | x | x | 0 | 0 |
| 0 | 0 | x | x | x | x | 1 | 1 | 1 | x | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | x | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | X | 1 | 1 | 1 | 0 |
| x | x | x | x | x | x | X | X | X | X | X | X | 0 | 1 |
| 0 | 0 | x | x | x | x | 1 | 1 | 1 | 0 | X | X | 1 | 1 |
| x | x | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | X | X | 1 | 1 |
| X | x | x | x | x | x | X | X | X | X | 0 | 1 | 1 | 0 |

باتوجه به جدول بالا، روابط زیر برای هریک از بیت‌های خروجی محاسبه شده‌است.

S1P = S0P = Q1’.Q0.QB0

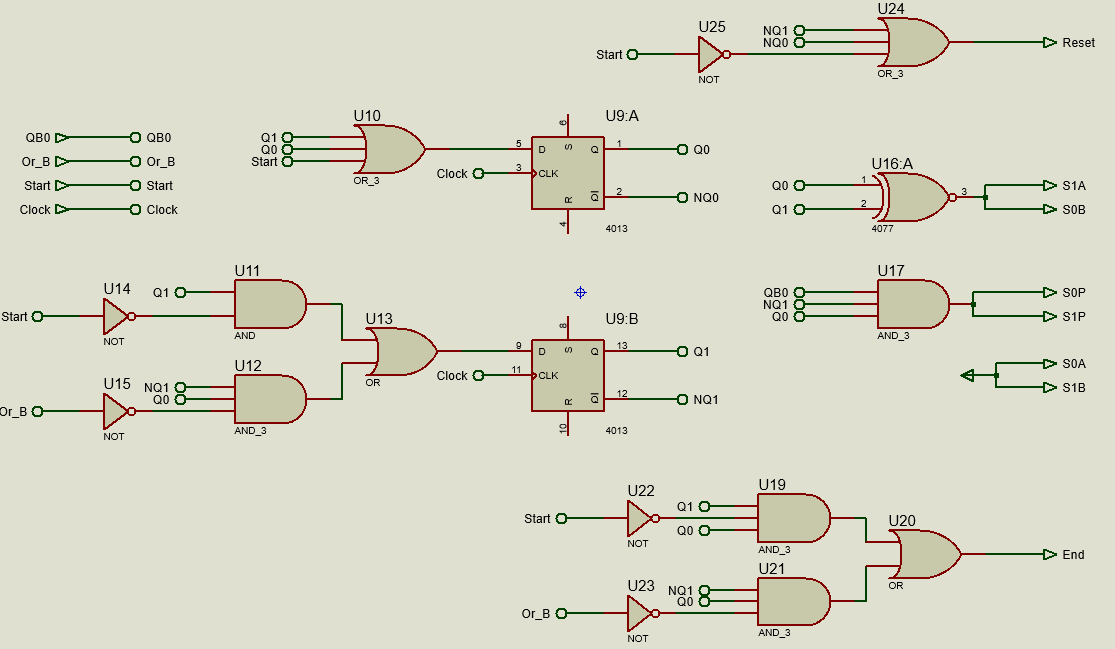
S1B = S0A = 1

Reset = (Q1.Q0.start)’

S0B = Q1’.Q0’+Q1.Q0 = S1A

End = Q1.Q0.start’ + Q1’.Q0.Or\_B’

این ماژول مشابه زیر طراحی شده است.

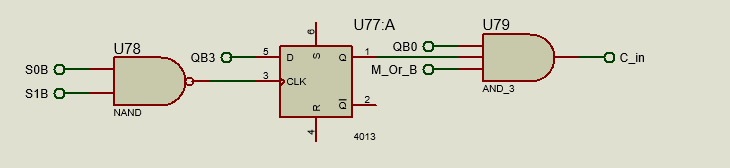


شکل 4

* 1. ماژول نهایی ضرب کننده ممیز ثابت

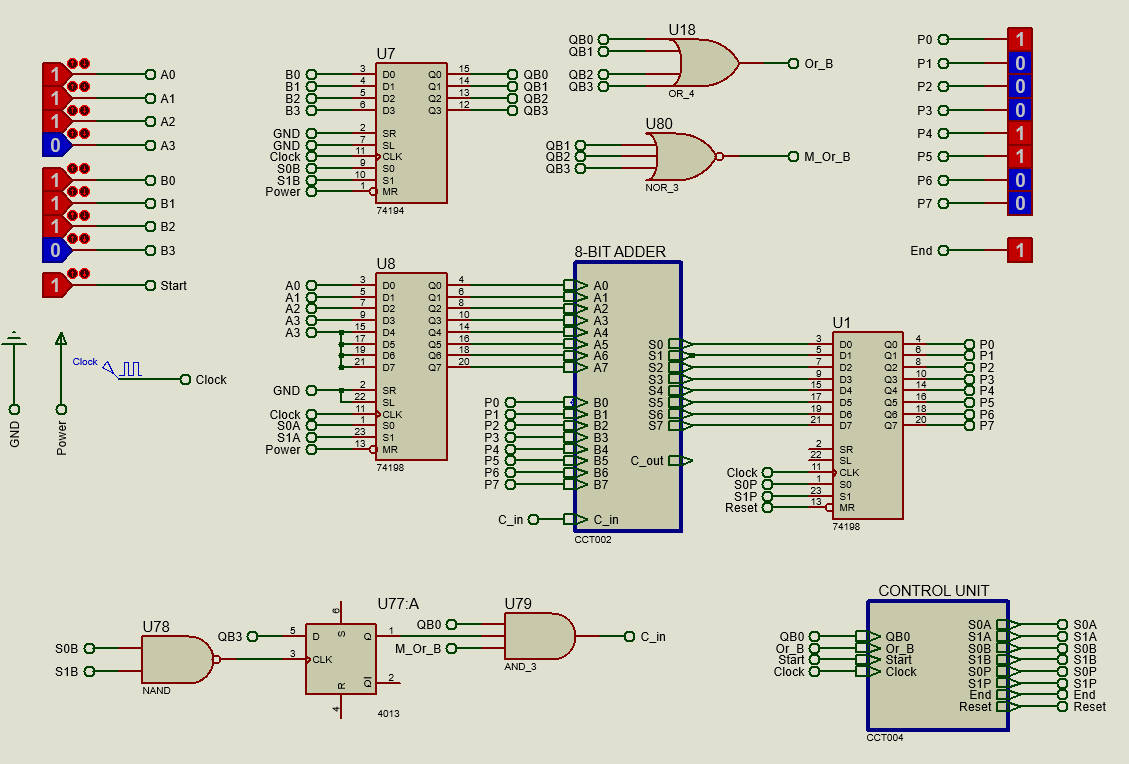
این ماژول دو ورودی چهار بیتی A و B را میگیرد. هنگامی که start =1 شود عملیات ضرب شروع شده و پس از حداکثر چهار کلاک خروجی نهایی نمایش داده می‌شود و سیگنال end مقدار یک را به خود میگیرد.

در ابتدا، مضروب sign extend میشود تا امکان محاسبه ضرب برای اعداد منفی نیز ممکن باشد. همانطور که پیشتر نیز اشاره شد، علاوه بر گسترش علامت مضروب، بیت علامت مضروب فیه نیز تا انتهای عملیات ضرب باید نگهداری شود. در صورتی که در آخرین کلاک، بیت مضروب یک بود، باید به جای جمع کردن مقدار قبلی خروجی و مضروب‌فیه، مضروب فیه از مقدار قبلی تفریق شود تا پاسخ نهایی در سیستم مکمل دو درست باشد. این کار با استفاده از مدار شکل 5 انجام میشود.



شکل 5

نمای کلی مدار در شکل 6 قابل مشاهده است.

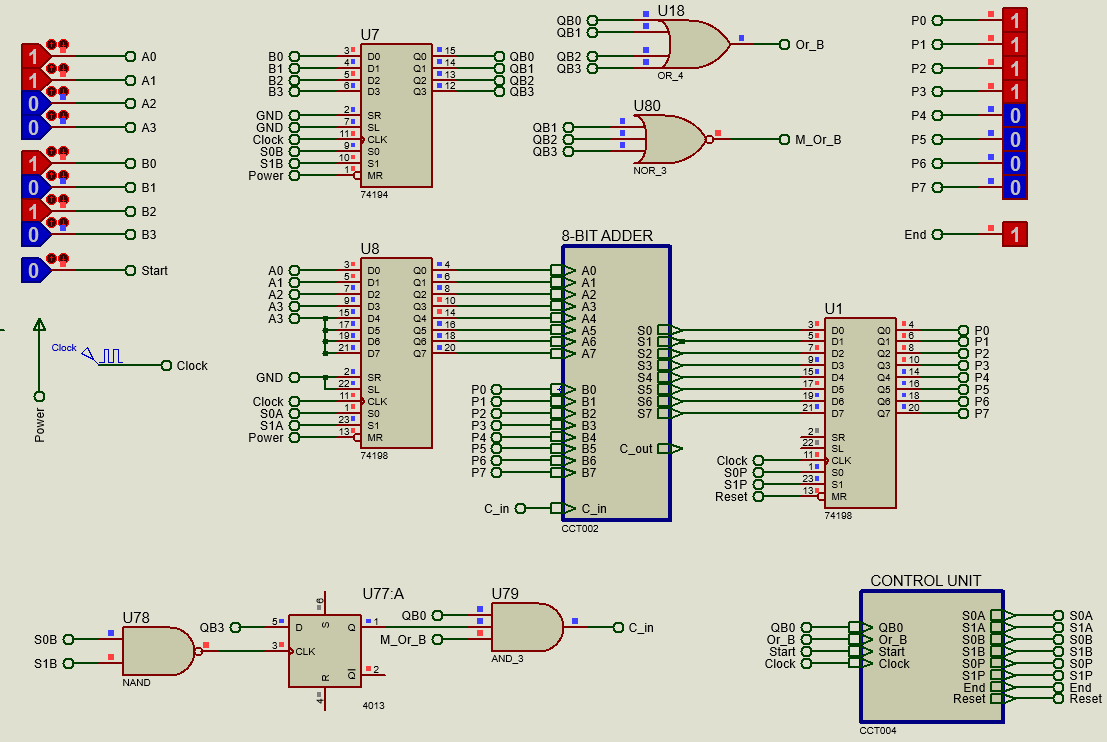


شکل 6

در ابتدا مقادیر A و B در ثبات‌ها ریخته میشود. سپس در چهار کلاک متوالی هربار عملیات جمع یا تفرریق اتفاق می افتد و در انتها سیگنال end به اندازه یک کلاک فعال میشود.

1. نتایج بدست آمده از آزمایش

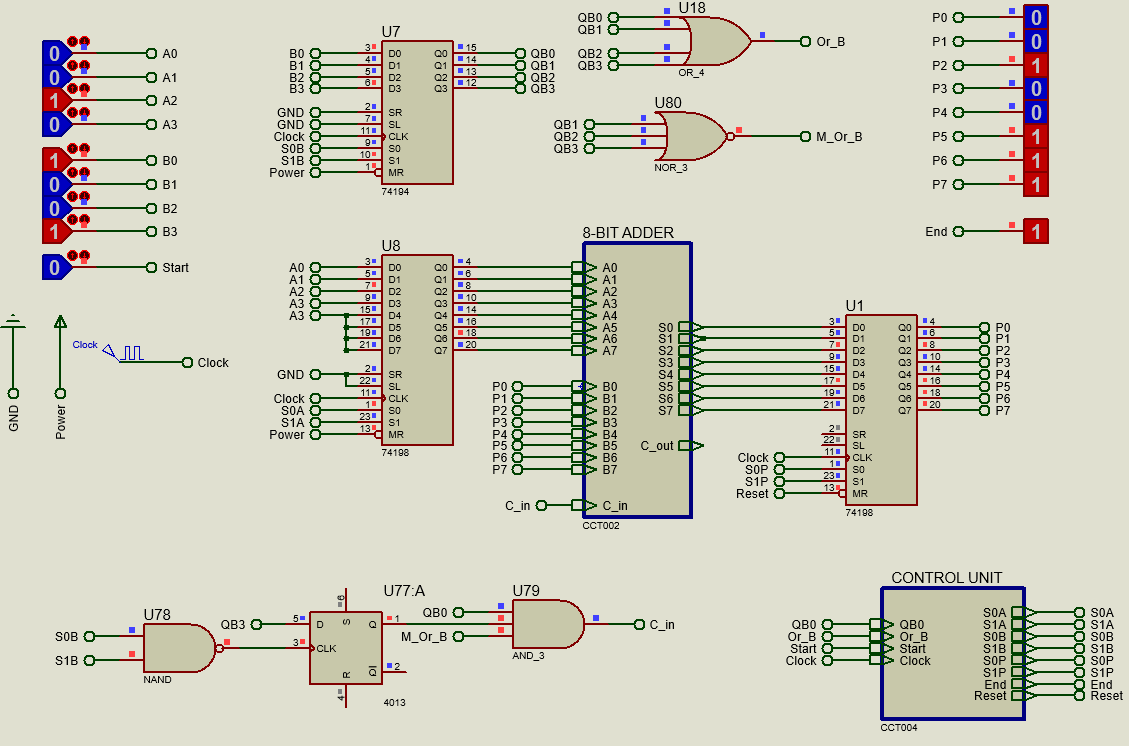
جهت صحت سنجی عملکرد مدار، ورودی‌های متفاوتی تست می‌شود. این ورودی‌ها باتوجه به منفی و مثبت بودن مضروب و مضروب فیه در نظر گرفته شده اند.

* مقدار A=3 و B=5:

شکل 7

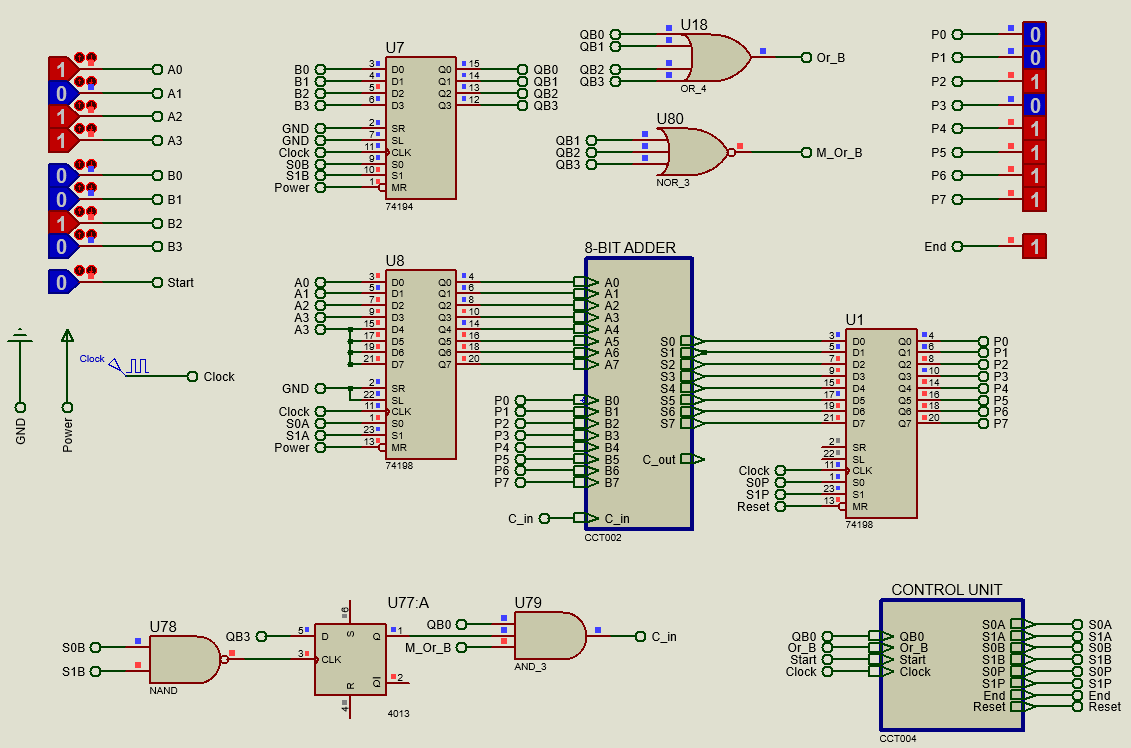
همانطور که دیده می‌شود خروجی نهایی عدد 15=1111 است.

* مقدار A = 4 و B = -7:

طبق تصویر خروجی نهایی مقدار 28- است که در سیستم مکمل دو و به صورت 11100100 نمایش داده میشود.

شکل 8

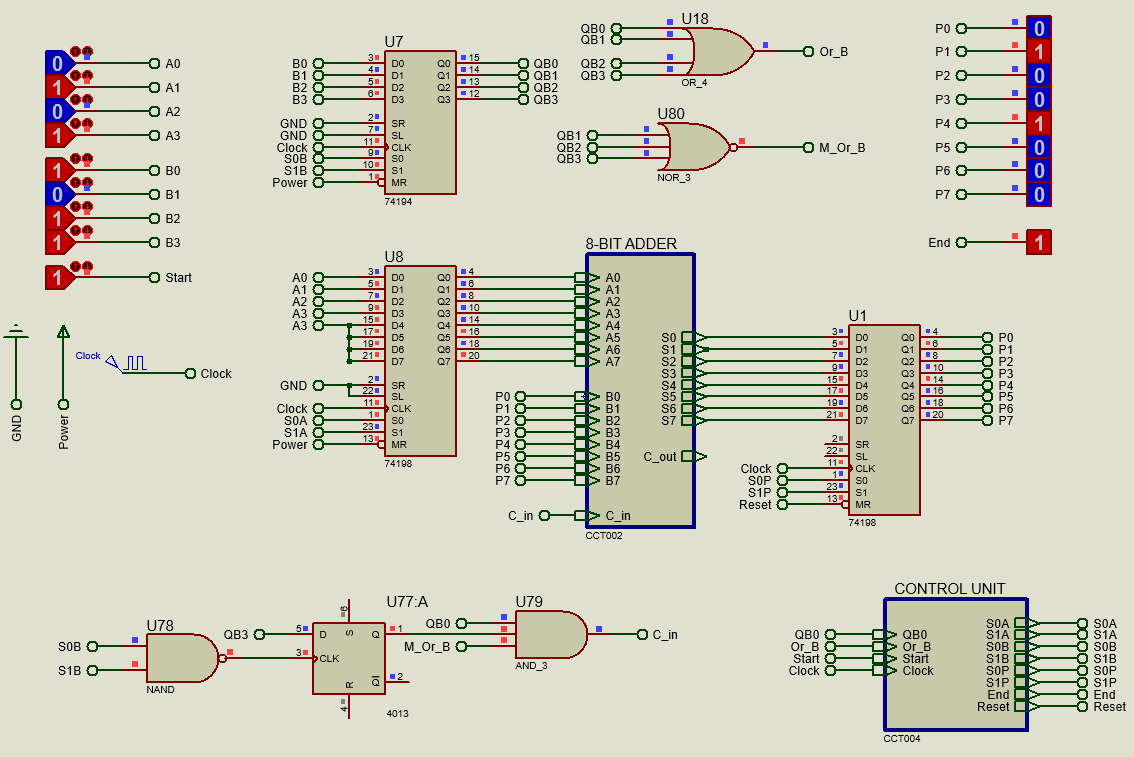
* مقدار A = -3 و B = 4

مطابق تصویر، عدد 12- در سیستم مکمل دو و به صورت 11110100 نمایش داده شده است.

شکل 9

* مقدار A= -6 و B = -3

شکل 10



همانطور که در شکل مشخص است، پاسخ نهایی 18 است که معادل با 00010010 است.