بسمه تعالی

گزارش کار آزمایشگاه معماری کامپیوتر

ضرب کننده ترکیبی

استاد: دکتر سربازی

دستیارآموزشی: مهندس غیبی

نویسنده

مریم شیران 400109446

دانشگاه صنعتی شریف  
تابستان 1403

فهرست مطالب

[1 آزمایش اختیاری: ضرب کننده ترکیبی 1](#_Toc174716947)

[1-1 مقدمه و هدف 1](#_Toc174716948)

[2-1 تجزیه و تحلیل تئوری آزمایش 1](#_Toc174716949)

[3-1 شرح دستگاه ‌ها و وسایل مورد استفاده 5](#_Toc174716950)

[4-1 شرح آزمایش 4های ضرب کننده های4 بیت در 4 بیت 6](#_Toc174716951)

[1 -5 مقایسه دو روش 8](#_Toc174716952)

[1 -6 شرح آزمایش ضرب کننده 8 بیت در 8 بیت 9](#_Toc174716953)

[1 -7 منابع و مراجع 9](#_Toc174716954)

# آزمایش اختیاری: ضرب کننده ترکیبی

## مقدمه و هدف

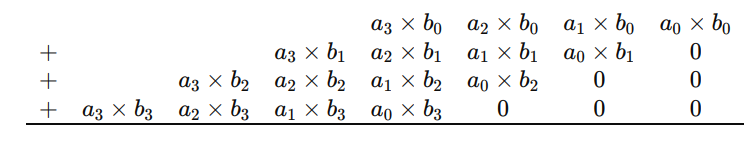
در این آزمایش یک مدار ضرب کننده ترکیبی بدون علامت 4 بیت در 4 بیت در پروتیوس به دو روش پیاده سازی کرده و سپس دو روش را با هم مقایسه میکنیم. روش اول تنها با استفاده از 3 عدد full adder 4 بیتی و گیت های And میباشد.

روش دوم با استفاده از دو عدد full adder 4 بیتی و یک full adder 6 بیتی و گیت های And میباشد.

در ادامه روش اول را گسترش میدهیم و به یک ضرب کننده 8 بیت در 8 بیت تبدیل میکنیم.

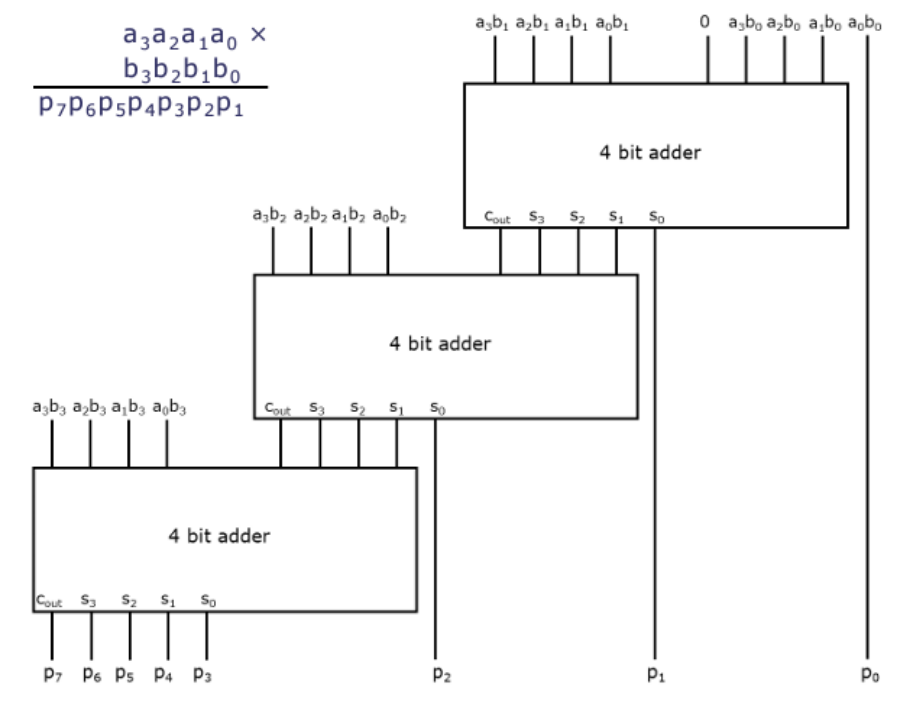
## تجزیه و تحلیل تئوری آزمایش

به صورت عادی هنگام ضرب دو عدد 4 بیتی a3​a2​a1​a0​ و b3​b2​b1​b0 در یک دیگر به شکل زیر فکر کرده و حاصل را محاسبه میکنیم:



به هر یک از سطر های بالا حاصل ضرب جزیی میگویند. ما میخواهیم این 4 حاصل ضرب های جزیی را با تنها 3 جمع کننده ای که داریم با هم جمع کنیم. شاید ناکافی به نظر برسد چون ما 8 ستون و 4 سطر داریم، ولی نکته اینجاست که تنها یکی از ستون ها نیاز به جمع 4 عملوند دارد و باقی با تعداد کمتر و حتی صفر، حاصل آن ستون را برای ما آماده میکنند.   
بدین منظور میتوان مانند زیر عمل کرد:

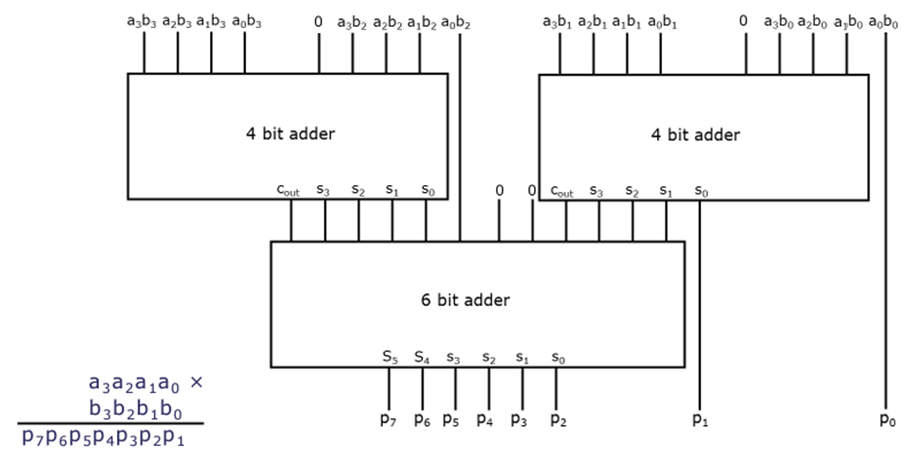
روش اول:  
با استفاده از سه عدد full adder 4 بیتی و گیت های And مدار مربوطه را میسازیم. در این روش معادلا داریم حاصل ضرب را بدین صورت حساب میکنیم که ابتدا دو حاصل ضرب جزیی اول را باهم جمع کرده و سپس حاصل را حاصل ضرب جزیی سومی جمع میکنیم و حاصل این جمع را با حاصل ضرب جزیی چهارم جمع میکنیم.



شکل 1. نمای کلی مدار روش اول

روش دوم:

با استفاده از دو عدد full adder 4 بیتی و یک full adder 6 بیتی و گیت های And مدار مربوطه را میسازیم. در این روش معادلا داریم حاصل ضرب را بدین صورت حساب میکنیم که دو حاصل ضرب جزیی اول و دوم را باهم جمع کرده و سپس دو حاصل ضرب جزیی سوم و چهارم را نیز باهم جمع کرده و بعد با رعایت تفاوت در مرتبه ها این دو حاصل را باهم جمع میکنیم.



شکل 2. نمای کلی مدار روش دوم

در ادامه ما هر دو روش را در پروتیوس پیاده سازی خواهیم نمود.

برای گسترش ضرب کننده 4 بیتی در 4 بیتی به 8 بیتی در 8 بیتی از الگوریتم کاراتسوبا-آفمن استفاده میکنیم که توضیح آن به شرح زیر میباشد:

الگوریتم کاراتسوبا-آفمن روشی برای ضرب اعداد بزرگ است که به طور کارآمدتری نسبت به الگوریتم استاندارد مدرسه‌ای عمل می‌کند. این روش ضرب اعداد بزرگ را به اعداد کوچکتر تقسیم می‌کند و تعداد ضرب‌های تک‌رقمی مورد نیاز را کاهش می‌دهد. بیایید نحوه عملکرد آن را به‌طور خاص برای ضرب دو عدد باینری ۸ بیتی با استفاده از ضرب‌های ۴ بیتی توضیح دهیم.

توضیح مرحله به مرحله:

۱. تقسیم اعداد:

با فرض دو عدد باینری ۸ بیتی A و B، هر عدد به دو نیمه ۴ بیتی تقسیم می‌شود.

فرض کنید A = A1 \* 2^4 + A0 و B = B1 \* 2^4 + B0 باشد، که در آن A1 و A0 به ترتیب نشان‌دهنده بیت‌های بالاتر و پایین‌تر A هستند و B1 و B0 بیت‌های بالاتر و پایین‌تر B را نشان می‌دهند.

برای مثال:

A = (A1, A0) = (A1 \* 2^4 + A0) و B = (B1, B0) = (B1 \* 2^4 + B0)

که در آن:

A1 = ۴ بیت بالای A

A0 = ۴ بیت پایین A

B1 = ۴ بیت بالای B

B0 = ۴ بیت پایین B

۲. سه ضرب:

سه حاصل‌ضرب زیر را محاسبه کنید:

P1 = A1 \* B1 (حاصل‌ضرب بیت‌های بالاتر)

P2 = A0 \* B0 (حاصل‌ضرب بیت‌های پایین‌تر)

P3 = A0 \* B1 (ضرب متقابل مجموع بخش‌ها)

P4 = A1 \* B0 (ضرب متقابل مجموع بخش‌ها)

۳. ترکیب نتایج:

با استفاده از نتایج این ضرب‌ها، می‌توان حاصل‌ضرب نهایی را با استفاده از فرمول زیر محاسبه کرد:

A \* B = P1 \* 2^8 + (P3 + P4) \* 2^4 + P2

شیفت به سمت چپ توسط 2^4 و 2^8 به معنی هم‌تراز کردن این حاصل‌ضرب‌های جزئی در موقعیت صحیح آن‌ها در حاصل‌ضرب نهایی ۱۶ بیتی است.

پس با استفاده از 4 ضرب کننده 4 بیتی در 4 بیتی و 2 جمع کننده 16 بیتی میتوانیم ضرب 8 بیت در 8 بیت را انجام دهیم.

## شرح دستگاه ‌ها و وسایل مورد استفاده

برای شبیه سازی تیوری نرم افزار Proteus استفاده میشود. در زیر نام و توضیح مختصری از قطعات استفاده شده قابل مشاهده است.

**7483 IC (FULL ADDER 4 BIT):**

یک آی‌سی که به عنوان یک جمع‌کننده‌ی کامل ۴ بیتی استفاده می‌شود. این آی‌سی می‌تواند عملیات جمع ۴ بیت را به صورت موازی انجام دهد و بیت نقلی (Carry) تولید کند. 7483 برای اجرای عملیات جمع با بیت نقلی (Carry-in) و بیت نقلی خروجی (Carry-out) به کار می‌رود، که در نهایت یک جمع‌کننده‌ی چند بیتی را می‌سازد.

**:AND**

یک گیت AND پایه که برای عملیات منطقی AND استفاده می‌شود. فقط زمانی خروجی درست می‌دهد که همه ورودی‌های آن درست باشند.

**:LOGICPROBE**

یک پروب منطقی که برای تست و نشان دادن وضعیت منطقی (بالا، پایین یا نوسانی) یک مدار دیجیتال استفاده می‌شود.

**:LOGICSTATE**

این قطعه وضعیت منطقی یک نقطه در مدار را نشان می‌دهد. برای نمایش پاسخ از آن استفاده میکنیم.

**:OR**

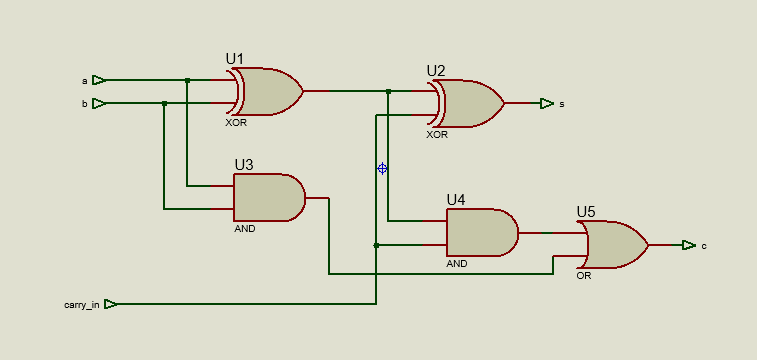
یک گیت OR پایه که برای عملیات منطقی OR استفاده می‌شود. اگر حداقل یکی از ورودی‌های آن درست باشد، خروجی درست می‌دهد.

**XOR:**

یک گیت XOR پایه که برای عملیات منطقی XOR استفاده می‌شود. اگر تعداد ورودی‌های درست فرد باشد، خروجی درست می‌دهد.

## شرح آزمایش 4های ضرب کننده های4 بیت در 4 بیت

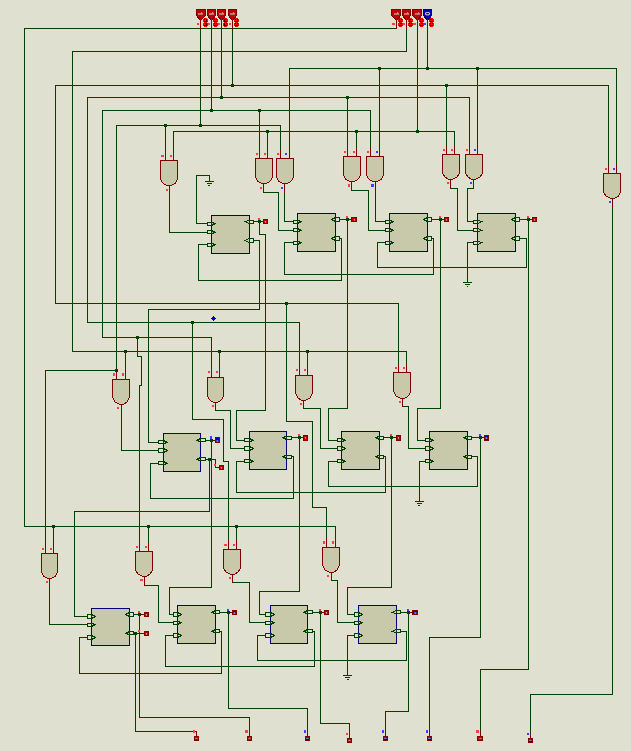
در زیر طراحی داخلی full adder تک بیتی را میبینیم که در هر دو آزمایش از آن استفاده خواهیم نمود.



شکل 3. نمای داخلی جمع کننده تک بیتی

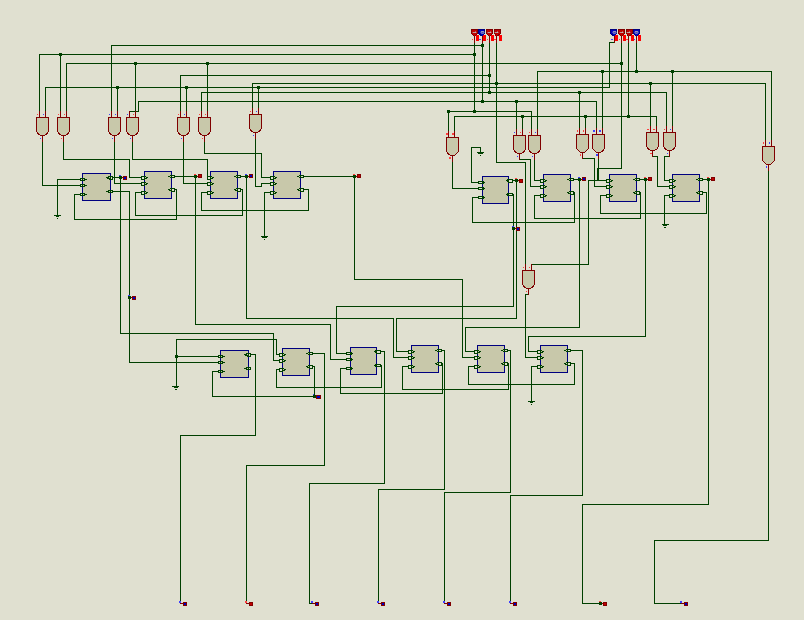
حال مطابق شکل های داده شده در قسمت قبل مدار های مربوطه را در پروتیوس پیاده سازی میکنیم.

روش اول:



شکل 4. پیاده سازی مدار روش اول

روش دوم:



شکل 5. پیاده سازی مدار روش دوم

# -5 مقایسه دو روش

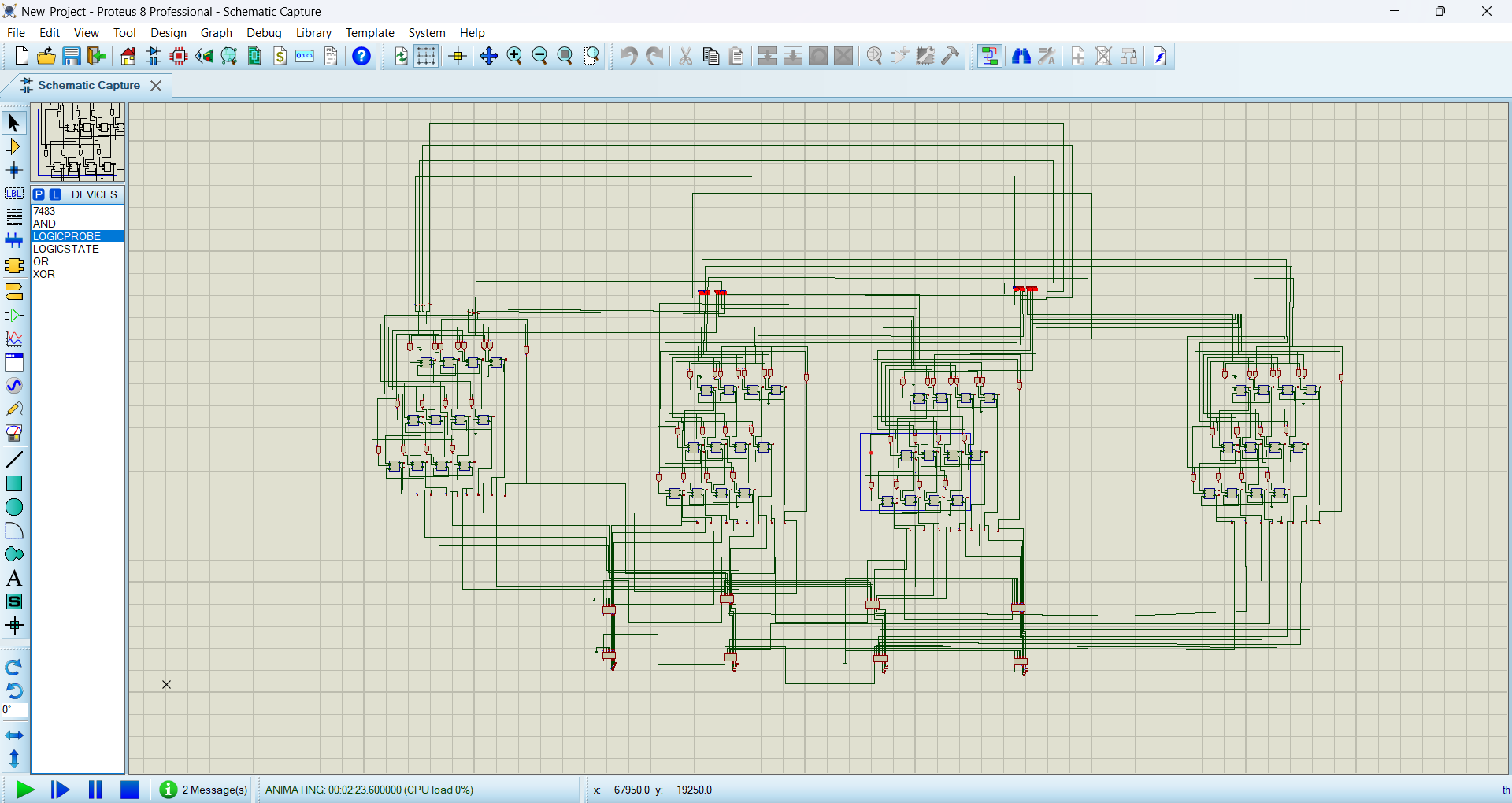
احتمالا فکر میکنیم که روش دوم تاخیر کمتری دارد، زیرا به صورت موازی سعی در محاسبه حاصل شده است. در زیر آن را بررسی میکنیم:

در هر دو روش پس از یک تاخیر گیت And تمام حاصل ضرب های جزیی آماده میشوند. در روش اول آماده شدن حاصل به اندازه 8 برابر تاخیر adder تک بیتی نیز در ادامه طول میکشد. در روش دوم آماده شدن حاصل به اندازه 10 برابر تاخیر adder تک بیتی نیز در ادامه طول میکشد. از این رو روش اول سریع تر است. و نیز نیاز به سخت افزار کمتری برای پیاده سازی دارد.

اما در اسکیل بالا تر، روش دوم مناسب تر خواهد بود و موازی سازی تاخیر را به میزان زیادی کاهش خواهد داد.

## 1 -6 شرح آزمایش ضرب کننده 8 بیت در 8 بیت

اکنون دقیقا طبق توضیحاتی که درباره الگوریتم کاراتسوبا-آفمن دادیم مدار زیر را پیاده سازی میکنیم:



شکل 6. پیاده سازی مدار ضرب کننده ترکیبی 8 بیت در 8 بیت

مشاهده میکنیم که دقیقا طبق توضیحات بخش 1.3 داریم که 4 حاصل ضرب 4 بیت در 4 بیت محاسبه میشود و سپس توسط جمع کننده های 16 بیتی(4 جمع کننده 4 بیتی که سری در کنار هم بسته شوند) باهم جمع میشوند.

## 1 -7 منابع و مراجع

* وبسایت گیکزفورگیکز
* <https://www.researchgate.net/publication/328841479_A_4-bit_array_multiplier_design_by_reversible_logic>
* اسلاید های درس معماری دکتر ارشدی
* ویکی پدیا الگوریتم کاراتسوبا-آفمن