مقاله پژوهشی

الگوریتم جدید ضرب دیجیتال با سرعت بالا بدون خط لوله با قابلیت بسط آسان

ابراهیم حسینی و مرتضی موسی‌زاده

چکیده: در این مقاله یک الگوریتم جدید برای ضرب‌کننده دیجیتال بدون علامت با مشخصات سرعت بالا و توان مصرفی کم بدون خط لوله که به آسانی برای تعداد بیت‌های بیشتر نیز بسط می‌یابد پیشنهاد شده است. بلوک‌های این ضرب‌کننده به صورت موازی کار می‌کنند و این عملکرد موجب افزایش چشم‌گیر سرعت ضرب‌کننده خواهد شد. در این الگوریتم، بیت‌های ورودی به دسته‌های کوچک‌تری تقسیم‌بندی می‌شوند که ضرب این دسته‌ها به صورت موازی و هم‌زمان انجام خواهند گرفت. این تقسیم‌بندی تا رسیدن به کمترین تعداد بیت ورودی یعنی ۲×۲ ادامه می‌یابد. در محاسبه حاصل‌ضرب هر یک از دسته‌ها، از الگوریتم پیشنهادی استفاده گردیده که منجر به تسریع حاصل‌ضرب هر دسته شده است و نتیجه نهایی از حاصل‌جمع این دسته‌های کوچک‌تر به دست خواهد آمد. برای جمع‌کردن دسته‌های کوچک‌تر از جمع‌کننده‌های درختی اصلاح‌شده که بتواند منجر به افزایش سرعت ضرب شود استفاده گردیده است. ضرب‌کننده‌هایی با طول بیت‌های ورودی ۲، ٤، ۸، ١٦، ۳۲ و ٦٤ با استفاده از الگوریتم پیشنهادی در فناوری ۱۸۰ نانومتر و ۹۰ نانومتر پیاده‌سازی شده‌اند که برای طول بیت ورودی ۳۲ بیت در فناوری ۱۸۰ نانومتر، تأخیر ۳/۰۵ نانوثانیه و مصرف توان ٤۰ میلی‌وات و در فناوری ۹۰ نانومتر، تأخیر ۱/۵۳ نانوثانیه و مصرف توان ۹/۷ میلی‌وات می‌باشد. همچنین با استفاده از روش پیشنهادی تخمین زده می‌شود که تأخیر ضرب‌کننده ۱۲۸×۱۲۸ در فناوری ۱۸۰ و ۹۰ نانومتر به ترتیب برابر با ٥/٤ نانوثانیه و ۲/۵ نانوثانیه شود. با توجه به نتایج و در مقایسه با سایر کارهای گزارش‌شده در مقالات و در پروسس یکسان، بدون افزایش توان مصرفی و با مساحت سیلیکون ۱/۵ برابر، سرعت ضرب‌کننده پیشنهادی بیش از ۲ برابر افزایش یافته است.

کلیدواژه: **ضرب‌کننده پرسرعت، جمع‌کننده بدون خط لوله، جمع‌کننده درختی Kogge-Stone اصلاح‌شده، جمع‌کننده پیش‌بینی بیت نقلی.**

۱- مقدمه

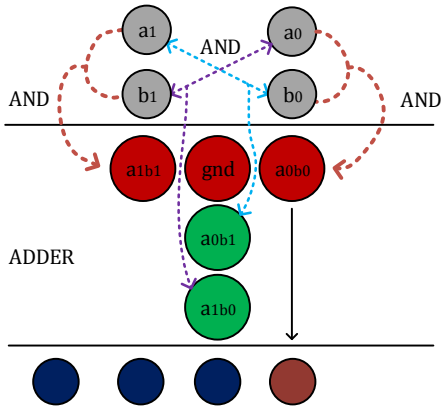
ضرب‌کننده‌ها از بلوک‌های اساسی و ضروری برای ریزپردازنده‌ها، پردازنده سیگنال‌های دیجیتال، پردازنده گرافیکی GPU[1]، فیلترهای دیجیتال و بسیاری دیگر از سیستم‌های الکترونیکی هستند که در بسیاری از موارد منجر به محدودیت‌های اساسی در عملکرد این بلوک‌ها می‌شوند. برای نمونه می‌توان به ساختار ریزپردازنده‌ها اشاره کرد که به دلیل ماهیت کند ضرب در مقایسه با سایر عملیات‌های ریاضی یا در مجموعه دستورالعمل‌های آنها دستور ضرب وجود ندارد و عملیات ضرب باید به صورت نرم‌افزاری و بسیار کند با استفاده از جمع‌های متوالی انجام شود و یا در صورت وجودداشتن دستور ضرب در مجموعه دستورالعمل‌ها، مسیر مجزای سخت‌افزاری فقط برای ضرب در قسمت محاسباتی و منطقی پردازنده (ALU)[2] پیش‌بینی می‌شود که باعث افزایش مساحت سیلیکون پردازنده و در نتیجه بالارفتن قیمت پردازنده خواهد شد. ضرب‌کننده سریع یکی از الزامات طراحی پردازنده‌های سریع می‌باشد که در صورت تحقق، پردازنده‌های بسیار باکیفیت‌تر می‌تواند طراحی شود. الگوریتم‌ها و تکنیک‌های مختلفی برای عمل ضرب در طول زمان پیشنهاد شده که در هر یک از این الگوریتم‌ها یک و یا ترکیبی از پارامترهای تأخیر، توان مصرفی، مساحت و پیچیدگی به عنوان معیار طراحی در نظر گرفته می‌شوند. در ساده‌ترین حالت الگوریتم ضرب، برای ضرب دو عدد M و N بیتی بدون علامت، ابتدا رقم کم‌ارزش عدد دوم N بیتی انتخاب شده و در تک‌تک ارقام عدد M بیتی ضرب می‌شود و اولین حاصل‌ضرب جزئی تولید می‌گردد. این حاصل‌ضرب جزئی یا صفر است در صورتی که رقم کم‌ارزش (LSB)[3] عدد دوم، صفر باشد و یا برابر با عدد M بیتی می‌باشد در صورتی که رقم کم‌ارزش عدد دوم یک باشد. عمل ضرب تک‌بیتی باینری توسط گیت AND انجام می‌گیرد و بنابراین برای تولید هر یک از حاصل‌ضرب‌های جزئی به تعداد M گیت AND دو ورودی نیاز است. روند تولید حاصل‌ضرب‌های جزئی به ترتیب و با در نظر گرفتن ارزش مکانی بیت‌های عدد دوم تا بیت پرارزش (MSB)[4] ادامه پیدا می‌کند. به این ترتیب N حاصل‌ضرب جزئی M بیتی تولید می‌گردد. سرانجام با جمع حاصل‌ضرب‌های جزئی و با رعایت ارزش مکانی بیت‌ها حاصل‌ضرب نهایی تولید می‌شود. بنابراین در حالت کلی می‌توان عملیات ضرب را به دو بخش: ۱) تولید حاصل‌ضرب‌های جزئی و ۲) جمع‌کردن حاصل‌ضرب‌های جزئی تولیدشده تقسیم‌بندی کرد. تولید حاصل‌ضرب‌های جزئی به سادگی و با تعدادی گیت AND امکان‌پذیر می‌باشد. کندبودن عملیات ضرب به دلیل بخش دوم یعنی جمع‌کردن حاصل‌ضرب‌های جزئی مربوط می‌شود. به عبارت دیگر مسأله ضرب در این مرحله به مسأله جمع‌کردن N عدد M بیتی تبدیل شده است. ساده‌ترین جواب این مسأله استفاده از یک جمع‌کننده M بیتی برای جمع‌کردن N عدد در N پالس ساعت متوالی می‌باشد که تحت عنوان ضرب‌کننده سریال شناخته می‌شود و عملیات ضرب با این ضرب‌کننده بسیار کند خواهد بود.

این مقاله در تاریخ ۲۳ فروردین ماه ۱۳۹۹ دریافت و در تاریخ ۲۱ بهمن ماه ۱۳۹۹ بازنگری شد.

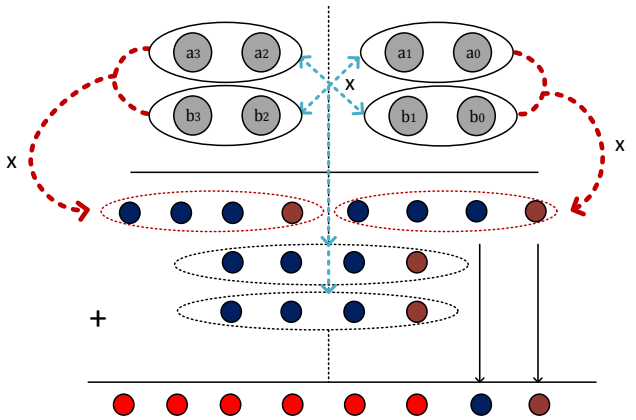
ابراهیم حسینی، گروه الکترونیک، دانشکده مهندسی برق و کامپیوتر، دانشگاه ارومیه، ارومیه، ایران، (email: st_e.hosseini@urmia.ac.ir).

مرتضی موسی‌زاده (نویسنده مسئول)، گروه الکترونیک، دانشکده مهندسی برق و کامپیوتر، دانشگاه ارومیه، ارومیه، ایران، (email: m.mousazadeh@urmia.ac.ir).

1. Graphical Processing Unit
2. Arithmetic and Logic Unit
3. Least Significant Bit
4. Most Significant Bit



شکل ۱: الگوریتم ضرب دوبیتی.



شکل ۲: الگوریتم ضرب چهاربیتی.

روش مرسوم دیگر برای قسمت جمع، استفاده از فشرده‌سازها برای تبدیل N حاصل‌ضرب جزئی به دو عدد و در نهایت استفاده از یک جمع‌کننده برای تولید حاصل‌ضرب نهایی می‌باشد [۱]. استفاده از فشرده‌سازها برای کم‌کردن تعداد حاصل‌ضرب‌های جزئی به دلیل تأخیری که به مدار اضافه می‌کند نمی‌تواند منجر به ضرب‌کننده‌های بسیار سریع گردد. تأخیر فشرده‌سازها نیز با افزایش تعداد بیت‌های ورودی افزایش می‌یابد. با در نظر گرفتن فشرده‌سازی می‌توان عملیات ضرب را به سه بخش تقسیم کرد: ۱) تولید حاصل‌ضرب‌های جزئی، ۲) فشرده‌سازی و تبدیل حاصل‌ضرب‌های جزئی به دو عدد و ۳) جمع دو عدد حاصل از فشرده‌سازی و تولید حاصل‌ضرب نهایی. این نوع از ضرب‌کننده‌ها تحت عنوان ضرب‌کننده‌های Wallace tree نیز شناخته می‌شوند [۲]. یکی از مؤثرترین روش‌هایی که برای افزایش سرعت ضرب‌کننده‌ها پیشنهاد شده است ضرب‌کننده‌های Booth encoding می‌باشد [۱]. ضرب‌کننده Booth در واقع انجام عمل ضرب در مبنای چهار می‌باشد که منجر به نصف‌شدن تعداد حاصل‌ضرب‌های جزئی و همچنین نصف‌شدن تأخیر کلی ضرب‌کننده خواهد شد. با توسعه این کار به مبناهای بالاتر می‌توان تعداد حاصل‌ضرب‌های جزئی را باز هم بیشتر کاهش داد. در حالت کلی ضرب‌کننده در مبنای r^2 تعداد حاصل‌ضرب‌های جزئی را به N/r کاهش می‌دهد [۳]. عیب روش بوث نیز توان مصرفی زیاد به علت قطعی‌های ناخواسته که از مسیرهای متفاوت این روش به وجود می‌آید است [۴] و [۵]. در تمام روش‌های بالا می‌توان از تکنیک خط لوله[1] نیز برای افزایش سرعت ضرب‌کننده استفاده کرد ولی به دلیل افزایش تعداد طبقه‌ها و همچنین تعداد پالس ساعت، توان مصرفی و مساحت کلی ضرب‌کننده افزایش می‌یابد [۶]. در این مقاله یک روش جدید برای الگوریتم ضرب معرفی می‌شود که به شکل ساختاری با تمام روش‌های مرورشده متفاوت است. با استفاده از روش ارائه‌شده، سرعت ضرب به شکل قابل ملاحظه‌ای افزایش می‌یابد بدون این که از ساختار خط لوله، بوث و یا درخت والاس استفاده شود. با استفاده از ساختار پیشنهادی ضرب‌کننده‌هایی با عرض بیت‌های بسیار بالا و همچنین بسیار سریع طراحی و پیاده‌سازی شده است.

ساختار کلی مقاله به قرار زیر است: بخش دوم این مقاله الگوریتم پیشنهادی برای ضرب را توصیف می‌کند. بخش سوم، پیاده‌سازی مداری ضرب‌کننده‌هایی با تعداد بیت‌های مختلف و با استفاده از الگوریتم پیشنهادی را توضیح می‌دهد. در بخش چهارم شبیه‌سازی و مقایسه با کارهای قبلی آورده شده و بخش نهایی به نتیجه‌گیری کلی می‌پردازد.

1. Pipeline

۲- الگوریتم پیشنهادی ضرب

برای تسریع عملیات ضرب، ابتدا N بیت ورودی‌های ضرب‌کننده به دو دسته $N/2$ بیتی تقسیم می‌شوند. سپس هر کدام از گروه‌هایی که $N/2$ بیت هستند به دو دسته کوچک‌تر با طول $N/4$ بیت تقسیم می‌شوند و این تقسیم‌بندی ادامه پیدا می‌کند تا این که به زیرگروه‌هایی با طول ۲ بیت برسیم. مشخص است اگر تعداد بیت‌های ورودی توانی از ۲ باشند، این تقسیم‌بندی بهینه و متقارن خواهد بود. بعد از گروه‌بندی ضرب دسته‌های دوبیتی به طور هم‌زمان طبق الگوریتم جدید حساب می‌شوند. شکل ۱ چگونگی ضرب دسته‌های دوتایی را نشان می‌دهد. ضرب بیت‌هایی که به طور عمودی زیر همدیگر قرار گرفته‌اند، سطر اول را تشکیل می‌دهند و ضرب مورب بیت‌ها، سطرهای دوم و سوم را تشکیل می‌دهند. اگر در نظر بگیریم $A=(a_1\times 2,a_0)$ و $B=(b_1\times 2,b_0)$ در این صورت ضرب آنها برابر است با

$$\begin{aligned} A\times B &= (a_1\times 2,a_0)\times(b_1\times 2,b_0)\\ &= a_0b_0,(a_0b_1,a_1b_0)\times 2,a_1b_1\times 2^2 \end{aligned} \quad (۱)$$

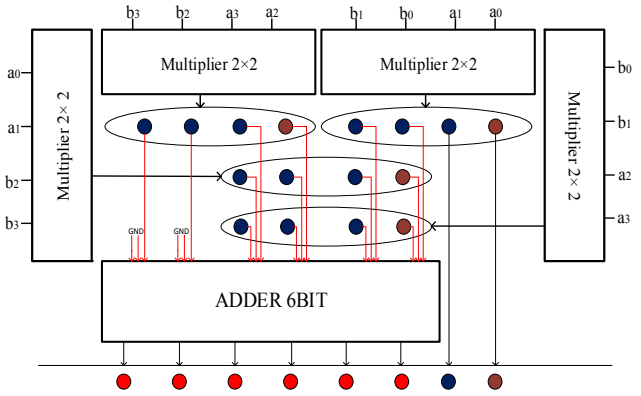
این ایده برای تعداد بیت‌های بالاتر نیز کاربرد دارد. در حالتی که ضرب‌کننده دو ورودی ۴بیتی داشته باشد، این دو ورودی به این صورت نوشته خواهند شد

$$\begin{aligned} A &= (a_3\times 2^3,a_2\times 2^2,a_1\times 2,a_0)\\ B &= (b_3\times 2^3,b_2\times 2^2,b_1\times 2,b_0) \end{aligned} \quad (۲)$$

و ضرب آنها طبق (۳) بیان خواهد شد

$$\begin{aligned} A\times B &= (a_3\times 2^3,a_2\times 2^2,a_1\times 2,a_0)\\ &\times(b_3\times 2^3,b_2\times 2^2,b_1\times 2,b_0)\\ &= (a_0b_0,(a_0b_1,a_1b_0)\times 2,a_1b_1\times 2^2)\\ &,(a_0b_2,(a_0b_3,a_1b_2)\times 2,a_1b_3\times 2^2)\times 2^2\\ &,(a_2b_0,(a_2b_1,a_3b_0)\times 2,a_3b_1\times 2^2)\times 2^2\\ &,(a_2b_2,(a_2b_3,a_3b_2)\times 2,a_3b_3\times 2^2)\times 2^4\\ &= M_1,(M_2,M_3)\times 2^2,M_4\times 2^4 \end{aligned} \quad (۳)$$

همان طور که (۳) نشان می‌دهد مشابه حالت قبل در ضرب ۴بیتی که به دو دسته ۲بیتی تقسیم شده‌اند، نتیجه خروجی در سه سطر چینش می‌یابد و سرانجام توسط یک جمع‌کننده ۶بیتی نتیجه نهایی آماده می‌شود. نمودار نقطه‌ای این ضرب در شکل ۲ نمایش داده شده است. برای ضرب ۴بیتی



شکل ۳: الگوریتم پیشنهادی برای ضرب چهاربیتی.

چهار بلوک از ضرب‌کننده‌های ۲بیتی و یک جمع‌کننده ۶بیتی نیاز است و شکل ۳ این موضوع را نشان می‌دهد.

به همین ترتیب یک ضرب‌کننده ۸بیتی از ۴ بلوک ضرب‌کننده ۴بیتی و یک جمع‌کننده ۱۲بیتی تشکیل شده است. این روش برای تعداد بیت‌های بالاتر قابل بسط است، یعنی این که برای ضرب‌کننده ۱۶بیتی ۴ بلوک از ضرب‌کننده ۸بیتی و یک جمع‌کننده ۲۴بیتی و برای ضرب‌کننده ۳۲بیتی ۴ بلوک از ضرب‌کننده ۱۶بیتی و یک جمع‌کننده ۴۸بیتی مورد نیاز است. شکل ۴ این الگوریتم را برای یک ضرب‌کننده ۶۴بیتی نشان می‌دهد. از مزیت‌های مهم و برجسته این روش عملکرد موازی و هم‌زمان بلوک‌های تشکیل‌دهنده است. با دو برابر شدن تعداد بیت‌های ورودی تنها تأخیر یک جمع‌کننده جدید به مسیر بحرانی اضافه می‌شود. رابطه (۴) تأخیر مسیر بحرانی یک ضرب‌کننده ۶۴بیتی را نشان می‌دهد. بر طبق الگوریتم پیشنهادی در ضرب ۶۴ بیت در طبقه نخست دو بیت اول و در طبقه دوم ۴ بیت اول و در طبقه سوم ۸ بیت اول و در طبقه چهارم ۱۶ بیت و در طبقه پنجم ۳۲ بیت از نتیجه نهایی خیلی سریع‌تر تولید خواهد شد. این بیت‌ها وارد جمع‌کننده طبقه بعدی نمی‌شوند و به همین دلیل زودتر از حاصل‌ضرب نهایی آماده خواهند شد. شکل ۵ نشان می‌دهد که چگونه این بیت‌ها برای یک ضرب‌کننده ۸بیتی تولید می‌شوند. شکل ۶ این الگوریتم و تعداد بیت‌های مورد نیاز را برای یک ضرب‌کننده $N \times N$ بیت نشان می‌دهد

$$\tau_{total} = \tau_{(۲\times۲ multiplier)} \times \tau_{(۶bit\_adder)} \times \tau_{(۱۲bit\_adder)} + \tau_{(۲۴bit\_adder)} + \tau_{(۴۸bit\_adder)} + \tau_{(۹۶bit\_adder)} \quad (۴)$$

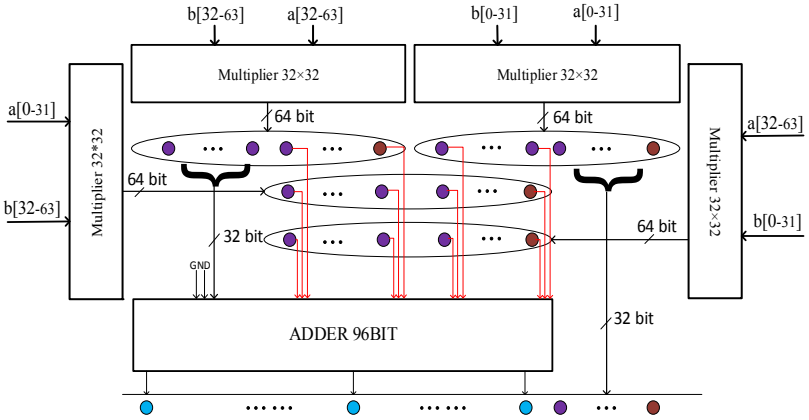
۳- پیاده‌سازی در سطح مداری

۳-۱ ضرب‌کننده ۲×۲

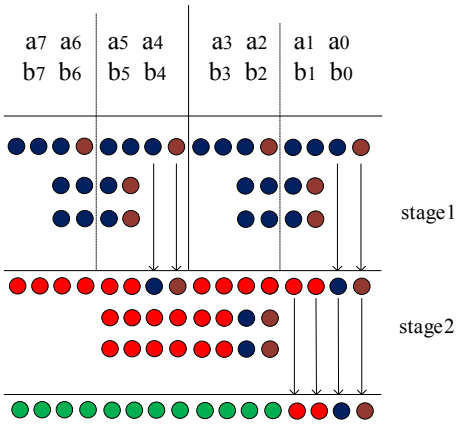
همان طور که در بخش‌های قبل هم اشاره شد، گروه‌بندی تا جایی ادامه می‌یابد که زیرگروه‌هایی با طول ۲ بیت به دست آیند. در مرحله اول ضرب دسته‌های دوتایی طبق الگوریتم معرفی‌شده به دست خواهد آمد. مدار شکل ۷ به این منظور استفاده شده است. برای ضرب‌کننده‌های ۶۴بیتی به تعداد ۴۵ بلوک از ضرب‌کننده‌های ۲بیتی مورد نیاز است. شکل ۸- الف، پیاده‌سازی گیت‌های AND در سطح ترانزیستوری و به صورت معماری گیت‌های انتقال به منظور بهبود سرعت را نشان می‌دهد. در شکل ۸- ب یک ساختار جدید و سریع بدون استفاده از معکوس‌کننده برای گیت XOR دو ورودی پیشنهاد شده است. این ساختار برای XOR بسیار ساده است و کار اصلی را ترانزیستورهای M۱-۴ انجام می‌دهند و برخلاف ساختارهای معمول از گیت NOT استفاده نشده که باعث سریع عمل کردن شده است. در این ساختار ورودی‌ها باید قدرت درایو داشته باشند. برای حالتی که دو ورودی صفر هستند اگر تنها از ترانزیستورهای M۱-۴ استفاده می‌شد، مشکل یک ولتاژ ترشولد در خروجی به وجود می‌آمد که اگر از طبقه‌ها و گیت‌های دیگر پشت سر آن استفاده شود نتیجه به حالت نرمال برمی‌گردد. ولی برای حل مشکل از ترانزیستورهای M۵-۹ استفاده می‌شود که سایز کوچکی دارند و برای صفرکردن یک ولتاژ ترشولد به کار رفته‌اند. ساختار ارائه‌شده برای ضرب، ساختار سریعی می‌باشد ولی برای افزایش بیشتر سرعت سعی شده که در پیاده‌سازی مداری هم ساختارهای سریعی طراحی و یا پیشنهاد گردد. گیت XOR پیشنهادی به دلیل تعداد دفعات استفاده در ساختار ضرب‌کننده می‌تواند تأخیر کلی را کاهش دهد.

۳-۲ ضرب‌کننده ٤×٤

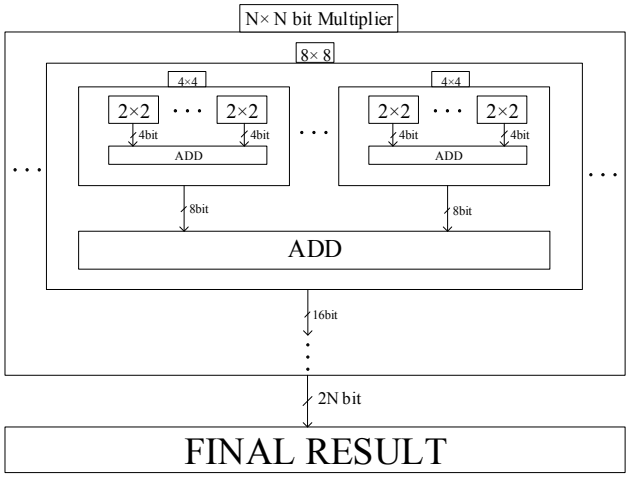
بعد از ضرب دسته‌های دوتایی، ضرب دسته‌های ۴تایی باید محاسبه شود. در این مرحله ۴ بلوک از ضرب دسته‌های دوتایی باید استفاده شود. ضرب‌های پاره‌ای برای دسته‌های ۴تایی توسط این بلوک‌ها تولید می‌شوند و سرانجام توسط یک جمع‌کننده ۶بیتی حاصل نهایی آماده می‌شود. باید سعی شود در هر طبقه جمع‌کننده‌ای مناسب انتخاب شود تا عملکرد بهینه گردد. جمع‌کننده‌هایی که دو عدد باینری با هم جمع می‌کنند همواره مورد تحقیق و بررسی طراحان قرار گرفته است. جمع‌کننده‌ها مجموعه‌ای گسترده و بزرگ هستند که هر کدام سرعت، توان مصرفی و مساحت اشغالی متفاوتی دارند. نیم‌جمع‌کننده تنها دو تک‌بیتی A و B را با هم جمع می‌کند و نتیجه صفر، ۱ یا ۲ خواهد بود. در نتیجه دو بیت برای نمایش خروجی لازم است که آنها نتیجه جمع S و بیت نقلی خروجی COUT نامیده می‌شوند. بیت نقلی خروجی به عنوان ورودی ستون بعدی در جمع‌کننده‌های چندبیتی استفاده می‌شود و در نتیجه ارزش مکانی دو برابری خواهد داشت. جمع‌کننده‌های چندبیتی باید امکان گرفتن بیت نقلی ستون قبلی را داشته باشند، یعنی این که به صورت جمع‌کننده کامل استفاده شوند. در جمع‌کننده‌ها تعریف دو سیگنال تولید (G) و انتشار (P) می‌تواند مفید باشد. در جمع‌کننده زمانی بیت نقلی تولید می‌شود که هر دو ورودی یک باشند و در نتیجه $G = A.B$ و جمع‌کننده زمانی بیت نقلی منتقل می‌کند یا به عبارت دیگر دوباره رقم نقلی تولید می‌کند که Cin دریافت کند و یکی از ورودی‌ها یک باشد و در نتیجه $P = A \oplus B$. با توجه به تعریف‌های ارائه‌شده دو سیگنال S و COUT به این شیوه تعریف می‌شوند: $S = P \oplus Cin$ و $Cout = G + PCin$. ساده‌ترین شیوه برای جمع‌کننده‌های چندبیتی، ساختار نقلی نردبانی (Carry-ripple) است به این شیوه که نقلی خروجی به عنوان ورودی طبقه بعدی استفاده می‌شود. در این شیوه برای تعداد بیت بالا به علت طولانی‌شدن مسیر بحرانی، تأخیر جمع‌کننده بسیار زیاد خواهد شد و عمل جمع‌کردن به کندی انجام می‌گیرد. روش‌های سریع‌تر پیش‌بینی رقم نقلی (Carry-lookAhead) [۷] و پرش نقلی (Carry-skip) [۸] و [۹] تولید نقلی را در گروه‌های چندبیتی پیش‌بینی می‌کنند و این کار به وسیله محاسبه گروهی سیگنال‌های P و G انجام می‌گیرد و نشان خواهد داد چه زمانی بیت نقلی منتشر یا تولید می‌شود. جمع‌کننده‌های بزرگ‌تر از چند سطح ساختار CLA برای افزایش سرعت استفاده می‌کنند. الگوریتم پیشنهادی برای ضرب به گونه‌ای است که طبقه اول سه سطر دارد، پس باید جمع‌کننده‌ای که استفاده می‌شود سه سطر را با هم جمع کند. در روش‌های معمول می‌توان این سه سطر را به دو سطر فشرده کرد و سپس عمل جمع را انجام داد. در این صورت باید



شکل ۴: الگوریتم پیشنهادی برای ضرب ۶۴بیتی.

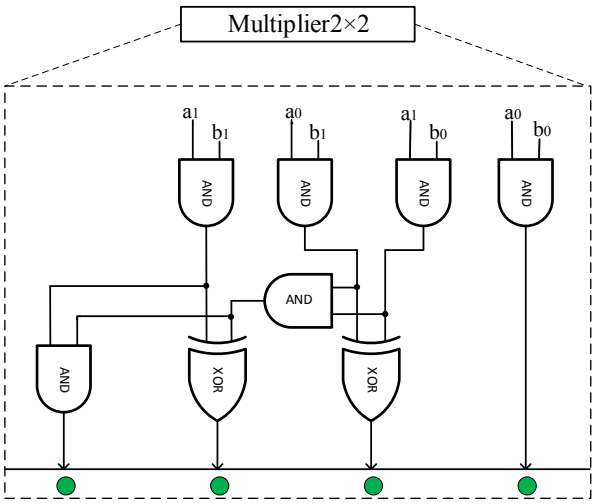


شکل ۵: الگوریتم ضرب هشت‌بیتی.

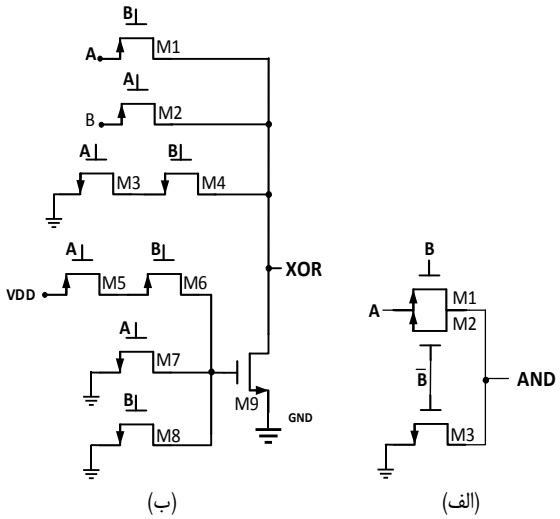


شکل ۶: ساختار پیشنهادی برای ضرب $N \times N$ بیتی.

یک فشرده‌ساز ۳ به ۲ استفاده شود که آن هم تأخیر اضافی وارد خواهد کرد. در این روش که پیشنهاد شده است فشرده‌ساز با گیت‌های ورودی جمع‌کننده یکی شده و تأخیر آن حذف شده است. این سه سطر ورودی مستقیماً و بدون فشرده‌شدن وارد جمع‌کننده می‌شوند. بر خلاف ساختارهای معمول برای جمع‌کننده از گیت‌های XOR و MAJ سه ورودی به جای دو ورودی استفاده شده است: $P = A \oplus B \oplus C$ و $G = MAJORITY(A, B, C)$. شکل ۹ ساختار کلی جمع‌کننده ۶بیتی به کار رفته را نشان می‌دهد. در مرحله اول به وسیله XOR و MAJ سه ورودی سیگنال‌های P و G تولید می‌شوند و به وسیله بلوک پیش‌بینی نقلی، بیت‌های نقلی مورد نظر تولید می‌شوند و سرانجام نتیجه حاصل‌جمع به وسیله XOR دو ورودی تولید خواهد شد.

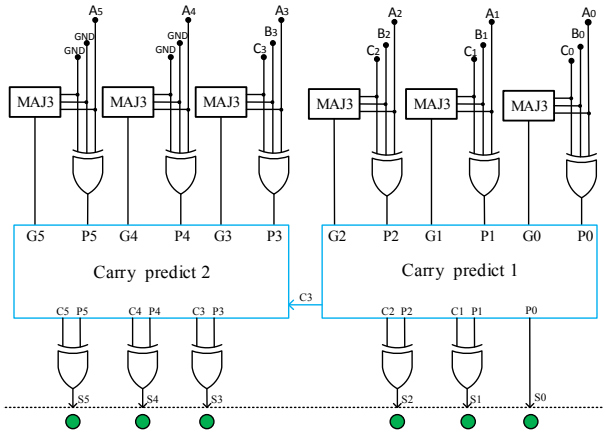


شکل ۷: پیاده‌سازی در سطح گیت ضرب‌کننده ۲×۲.

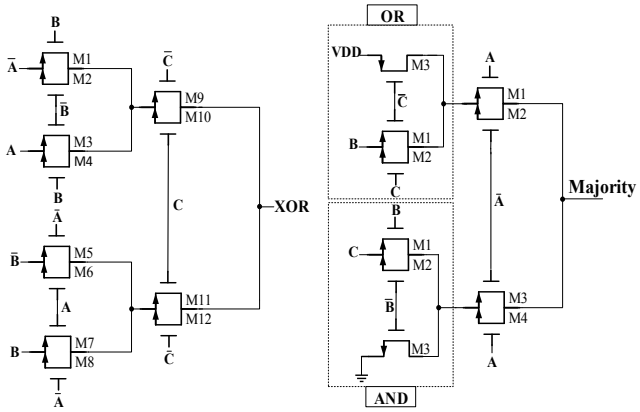


شکل ۸: پیاده‌سازی سطح ترانزیستور، (الف) AND و (ب) XOR پیشنهادی.

خروجی MAJ۳ زمانی یک خواهد شد که حداقل ۲ تا از ورودی‌های آن یک باشد و خروجی XOR۳ زمانی یک است که تعداد یک‌های ورودی فرد باشد. شکل‌های ۱۰-الف و ۱۰-ب پیاده‌سازی در سطح ترانزیستوری این دو گیت را نشان می‌دهد. برای پیش‌بینی و تولید نقلی از دو بلوک پیش‌بینی نقلی (Carry_predict۱,۲) استفاده شده است. به طوری که در بلوک اول $C_{1,2,3}$ تولید می‌شوند که $C_{1,2}$ برای تولید حاصل‌جمع $S_{1,2}$ و C_3 به عنوان ورودی طبقه بعدی استفاده شده است. رابطه‌های (۵) تا (۷) چگونگی محاسبه هر کدام از بیت‌های



شکل ۹: جمع‌کننده ۶بیتی.



(ب) (الف)

شکل ۱۰: (الف) MAJ۳ و (ب) XOR۳.

پیش‌بینی شده را نشان می‌دهد و شکل‌های ۱۱-الف و ۱۱-ب پیاده‌سازی در سطح ترانزیستوری آنها را نشان می‌دهد. دو بلوک پیش‌بینی بیت‌های نقلی دقیقاً مشابه هم هستند و در هر دوی آنها مدارهای شکل ۱۱ به کار گرفته شده‌اند

$$C_1 = G_0 \quad (۵)$$

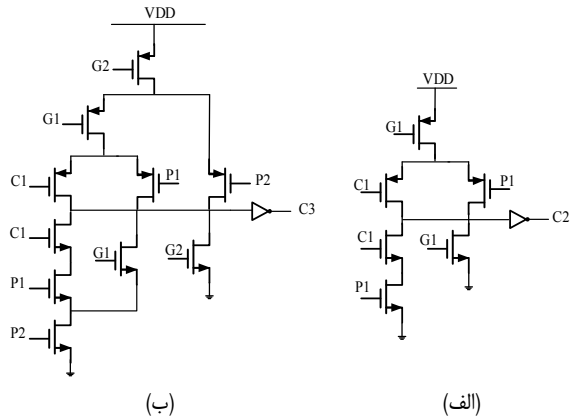
$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 \quad (۶)$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 \quad (۷)$$

۳-۳ ضرب‌کننده ۸×۸

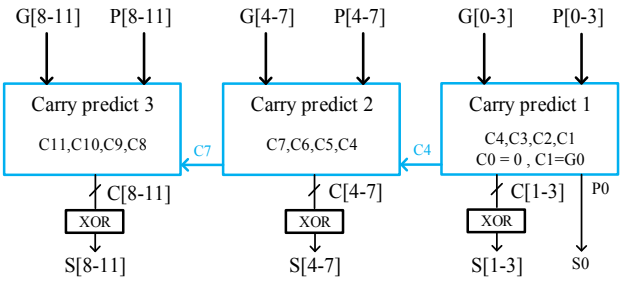
بر طبق الگوریتم معرفی‌شده ۴ بلوک از ضرب‌کننده ۴×۴ و یک جمع‌کننده ۱۲بیتی برای انجام ضرب ۸×۸ لازم است به این معنی که تنها تأخیر یک جمع‌کننده ۱۲بیتی به ضرب‌کننده جدید اضافه می‌شود. ساختار جمع‌کننده این طبقه شبیه طبقه قبل است با این تفاوت که به دلیل افزایش تعداد بیت‌ها ۳ بلوک پیش‌بینی نقلی استفاده شده و هر کدام از بلوک‌ها ۴ رقم نقلی تولیدی را پیش‌بینی می‌کنند. شکل ۱۲ ساختار این جمع‌کننده را نشان می‌دهد. هر کدام از سیگنال‌های C_2، C_3، C_5، C_6، C_8 و C_9 به وسیله مدارهای معرفی‌شده در شکل ۱۱- الف و ۱۱- ب تولید می‌شوند و سیگنال‌های $C_{4,7,10}$ به وسیله (۸) تعریف می‌شوند و توسط مدار شکل ۱۳- الف تولید خواهند شد و سیگنال C_{11} به وسیله (۹) تعریف خواهد شد و توسط مدار شکل ۱۳- ب پیش‌بینی می‌شود

$$C_{4,7,10} = G_{3,6,9} + P_{3,6,9} G_{2,5,8} + P_{3,6,9} P_{2,5,8} G_{1,4,7} + P_{3,6,9} P_{2,5,8} P_{1,4,7} C_{1,4,7} \quad (۸)$$



(ب) (الف)

شکل ۱۱: بلوک پیش‌بینی نقلی در سطح ترانزیستور (الف) C_2 و (ب) C_3.

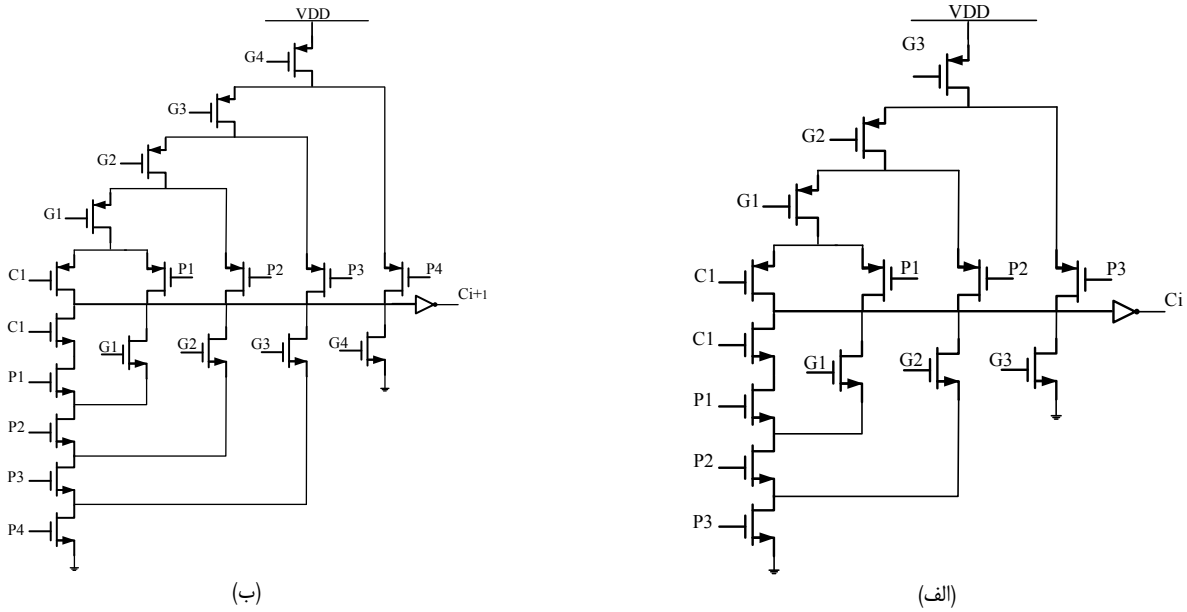


شکل ۱۲: جمع‌کننده ۱۲بیتی.

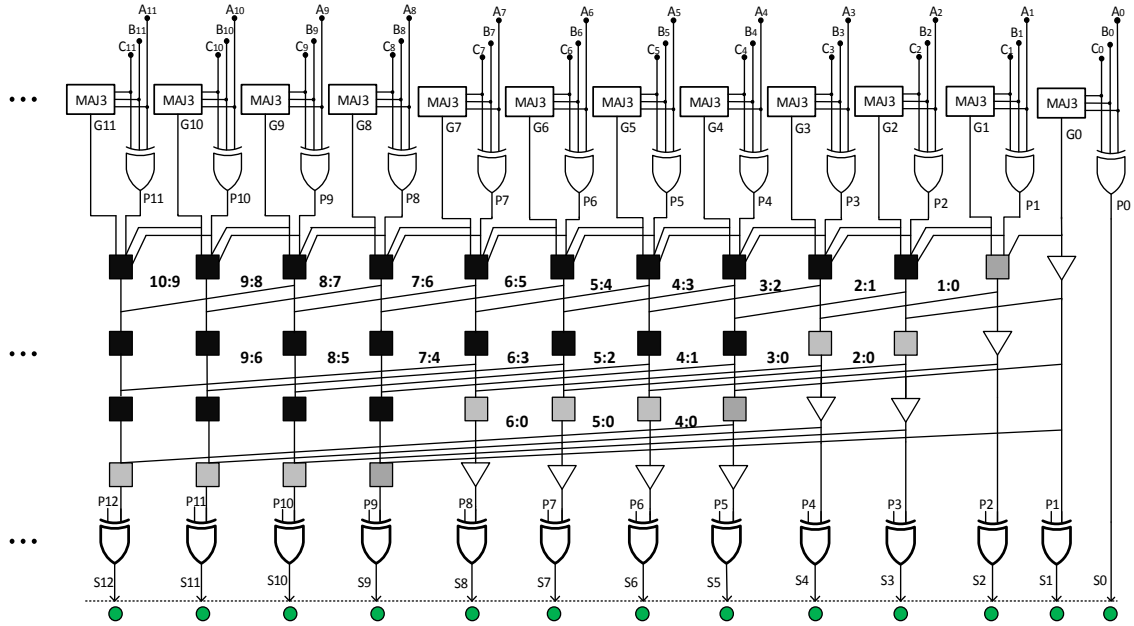
$$C_{11} = G_{10} + P_{10} G_9 + P_{10} P_9 G_8 + P_{10} P_9 P_8 G_7 + P_{10} P_9 P_8 P_7 C_7 \quad (۹)$$

۴-۳ ضرب‌کننده $N \times N$

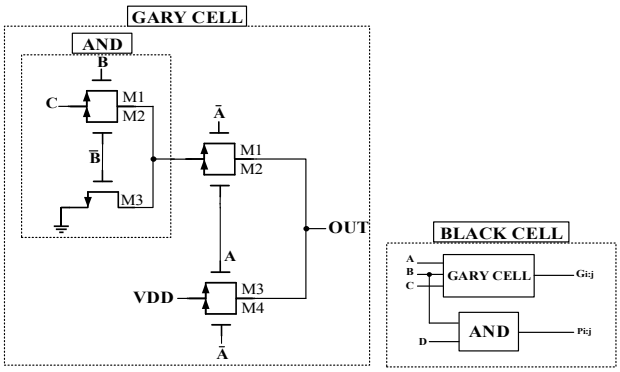
در حالت کلی برای ضرب‌کننده $N \times N$ بیتی به ۴ بلوک از ضرب‌کننده $N/2 \times N/2$ بیتی و یک جمع‌کننده $3N/2$ بیتی نیاز است. با افزایش تعداد بیت‌های ورودی، جمع‌کننده بزرگ‌تر خواهد شد پس باید جمع‌کننده متناسب انتخاب گردد تا تأخیری بهینه داشته باشیم. برای جمع‌کننده‌های بزرگ (معمولاً بیشتر از ۱۶ بیت) با ساختار CLA تأخیر جمع‌کننده بستگی به مسیر گذر نقلی از بین بلوک‌های پیش‌بینی آن دارد. برای کاهش تأخیر می‌توان از ساختارهای درختی پیش‌بینی نقلی استفاده کرد که در آنها تأخیر به صورت $\log_2 N$ رشد خواهد کرد. روش‌های مختلفی برای این کار وجود دارد که از لحاظ تعداد گیت‌ها و طبقه‌های به کار رفته و حداکثر گیت‌هایی که درایو می‌کند و سیم‌بندی بین طبقه‌ها متفاوت هستند. از معروف‌ترین این ساختارها می‌توان به Brent-kung [۱۰]، Sklansky [۱۱] و Kogge-stone [۱۲] اشاره کرد. در ساختار اول (Brent-kung) رقم نقلی ابتدا در گروه‌های ۲بیتی پیش‌بینی می‌شوند و از نتیجه آن برای محاسبه گروه‌های ۴بیتی و به دنبال آن برای گروه‌های ۸بیتی و به این شیوه ادامه خواهد یافت. این ساختار $2\log_2 N - 1$ طبقه خواهد داشت و در هر طبقه ۲ بلوک درایو خواهند شد (فن اوت ۲). در ساختار دوم با محاسبه گروه‌های بزرگ‌تر تعداد طبقه‌ها به $\log_2 N$ کاهش خواهد یافت ولی این کار با هزینه دو برابر شدن تعداد بلوک‌هایی که در هر طبقه درایو می‌شوند انجام خواهد گرفت و این برای جمع‌کننده‌های بزرگ، تأخیر بیشتر و استفاده از بافرهای بزرگ‌تر را به همراه خواهد داشت. در ساختار سوم (kogge-stone) تعداد طبقه‌ها $\log_2 N$ و بلوک‌هایی که باید درایو شوند ۲ است ولی با هزینه سیم‌بندی طولانی همراه است. به طور خلاصه در ساختار اول تعداد طبقه‌های زیاد، در ساختار دوم تعداد بلوک‌های درایوشده بیشتر و در ساختار سوم سیم‌بندی‌های طولانی خواهیم داشت. در این مقاله به علت



شکل ۱۳: پیاده‌سازی مدارهای پیش‌بینی بیت نقلی در سطح ترانزیستوری، (الف) C_i و (ب) C_{i+1}.

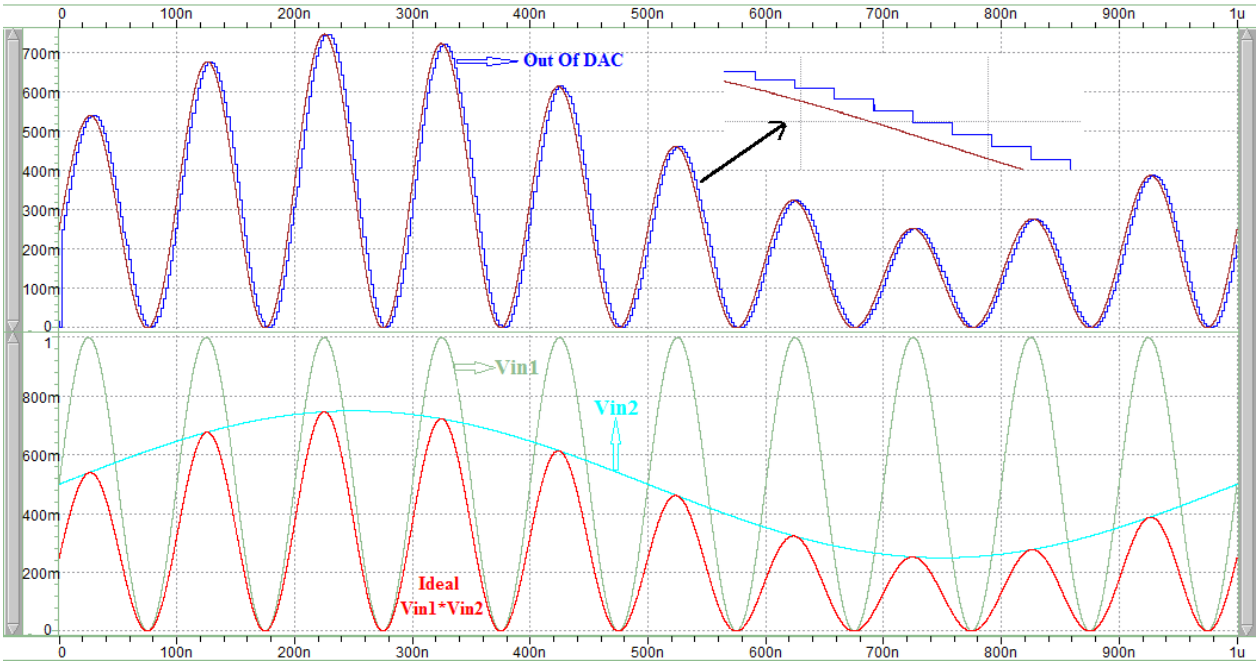


شکل ۱۴: جمع‌کننده پیشنهادی Kogge-Stone تغییریافته.

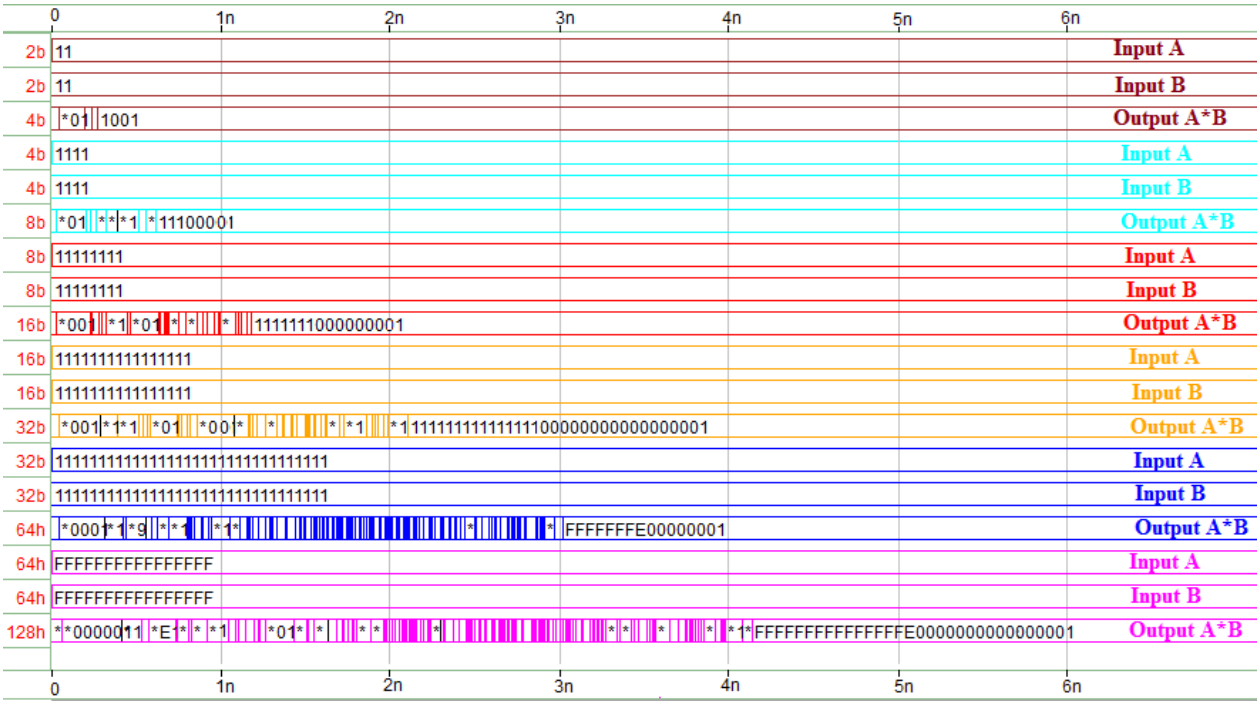


شکل ۱۵: بلوک‌های سیاه و خاکستری جمع‌کننده.

تعداد بیت‌های زیاد به خصوص در طبقه آخر و مقایسه نتایج شبیه‌سازی‌ها ساختار سوم انتخاب شده است. در ساختار سوم در ورودی، سلول‌های $G = AB$ و $P = A \oplus B$ وجود دارند که نتیجه آنها وارد بلوک‌های پیش‌بینی نقلی می‌شود و بعد از تولید نقلی‌های مطلوب، در آخر توسط XORکردن نقلی و P حاصل آماده خواهد شد. در حالت معمول دو سیگنال P و G توسط گیت‌های دو ورودی AND و XOR تولید می‌شوند ولی در این ساختار در هر طبقه سه سطر دیتا وجود دارد. در روشی ساده می‌توان این سه سطر را توسط فشرده‌ساز فشرده کرد و سرانجام وارد جمع‌کننده شوند که با این کار تأخیر فشرده‌ساز نیز اضافه می‌گردد. در ساختار پیشنهادی، فشرده‌ساز حذف شده و سه سطر مستقیماً وارد جمع‌کننده می‌شوند و بلوک‌های XOR و MAJ سه ورودی برای این منظور در ورودی جمع‌کننده در نظر گرفته شده‌اند و این کار باعث سریع‌ترشدن عمل جمع می‌شود. شکل ۱۴ جمع‌کننده تغییریافته را نشان می‌دهد. در این شکل بلوک‌های سیاه و خاکستری به کار گرفته شده‌اند که مدار داخلی آنها در سطح ترانزیستوری در شکل ۱۵ نمایش داده شده و ساختار آنها در این مقاله پیشنهاد گردیده است. بلوک‌های مثلثی این ساختار بافر هستند که از دو اینورتر پشت سر هم تشکیل شده است.



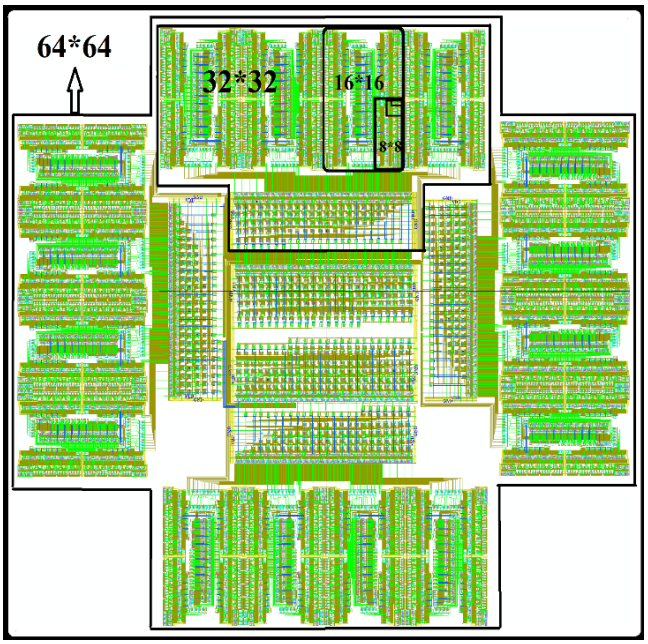
شکل ۱۶: تست ضرب‌کننده با HSPICE.



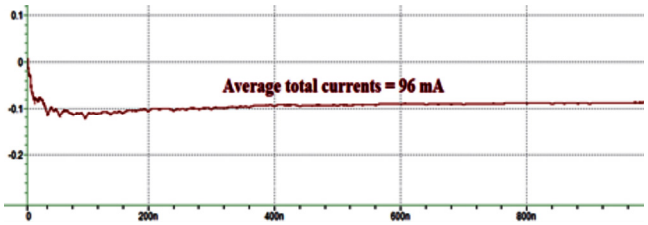
شکل ۱۷: تأخیر ضرب‌کننده برای تعداد بیت‌های مختلف.

٤- نتایج شبیه‌سازی‌های POST-LAYOUT

برای شبیه‌سازی ضرب‌کننده پیشنهادی دو سیگنال آنالوگ سینوسی با فرکانس و دامنه متفاوت در نظر گرفته شده است. این دو سیگنال آنالوگ به وسیله مبدل آنالوگ به دیجیتال ADC با فرکانس نمونه‌برداری مشخص دیجیتالی می‌شوند و به عنوان ورودی به ضرب‌کننده پیشنهادی اعمال می‌شوند. خروجی ضرب‌کننده به وسیله مبدل دیجیتال به آنالوگ DAC با فرکانس نمونه‌برداری برابر با ADC ورودی به حالت آنالوگ تبدیل می‌شود و با ضرب ایده‌آل دو سیگنال آنالوگ سینوسی مقایسه می‌گردد. شکل ۱۶ عملکرد صحیح این ضرب‌کننده را نشان می‌دهد. اگر پرش در خروجی DAC وجود داشته باشد نشان از عملکرد نادرست ضرب‌کننده است. همان طور که دیده می‌شود خروجی دقیقاً منطبق با ضرب ایده‌آل آنالوگی دو سیگنال است و هیچ خطایی رخ نداده است. با این الگوریتم ضرب‌کننده‌هایی با تعداد بیت‌های متفاوت شبیه‌سازی شده‌اند و نتایج برای ورودی با تعداد بیت‌های ورودی ۲، ۴، ۸، ۱۶، ۳۲ و ۶۴ آورده شده است. تأخیر مسیر بحرانی هر کدام از آنها زمانی که همه بیت‌های ورودی یک هستند در شکل ۱۷ نشان داده شده است. با توجه به این شکل مادامی که نتیجه حاصل‌ضرب به درستی حاصل نشده است، بیت‌های خروجی تغییر می‌کنند. با توجه به این شکل مقدار تأخیر برای ضرب‌کننده‌های ۲، ۴، ۸، ۱۶، ۳۲ و ۶۴ در پروسس ۱۸۰ نانومتر به ترتیب برابر با ۰/۳۲، ۰/۶۷، ۱/۲۵، ۲/۱ و ۴/۲ نانوثانیه می‌باشد. این نتایج مستخرج از شبیه‌سازی بعد از Layout می‌باشند. بعضی از مقادیر در این



شکل ۱۸: layout ضرب‌کننده پیشنهادی ۶۴بیتی.



شکل ۱۹: متوسط جریان کشیده‌شده از منبع تغذیه.

شکل به صورت باینری و بعضی به شکل هگزادسیمال نمایش داده شده‌اند. شکل ۱۸، layout ضرب‌کننده ۶۴بیتی را نشان می‌دهد که با استفاده از نرم‌افزار Virtuoso Cadence و در پروسس ۱۸۰ نانومتر پیاده‌سازی شده است. برای اندازه‌گیری توان مصرفی، متوسط جریان کشیده‌شده از منبع تغذیه ۱٫۸ ولتی محاسبه شده است. شکل ۱۹ متوسط جریان کشیده‌شده از منبع تغذیه و در هنگام تست ضرب‌کننده ۶۴بیتی را نشان می‌دهد. نرم‌افزار Hspice جریان واردشده به منبع تغذیه را نمایش می‌دهد که به صورت منفی می‌باشد. بعد از این که مدار به حالت پایدار خود رسید متوسط جریان کشیده‌شده از منبع تغذیه، ۹۶ میلی‌آمپر می‌باشد که با ضرب در منبع تغذیه ۱٫۸ ولتی توان مصرفی برابر با ۱۷۲٫۸ میلی‌وات خواهد شد. جدول ۱ مقایسه عملکرد این ضرب‌کننده را با روش‌ها و کارهای قبلی در پروسس‌هایی که ضرب‌کننده پیشنهادی پیاده‌سازی شده است نشان می‌دهد که گویای بهبود عملیات توسط روش پیشنهادی می‌باشد. در صورت پیاده‌سازی در پروسس‌های جدیدتر سرعت باز هم افزایش خواهد یافت. الگوریتم پیشنهادی در تمام پروسس‌ها، مزیت خود را نسبت به سایر الگوریتم‌ها به لحاظ سرعت و توان مصرفی نشان می‌دهد. برای این که بتوان به شکل منصفانه‌ای مقایسه را انجام داد، مواردی که انتخاب شده‌اند با ساختارهای متفاوت با ساختار پیشنهادی، از پروسس‌هایی هستند که ضرب‌کننده پیشنهادی در آن پروسس پیاده‌سازی شده است. در تمام عرض بیت‌های ورودی، سرعت ضرب‌کننده پیشنهادی بیشتر است ولی در مورد توان مصرفی و مساحت موارد کمتر و بیشتر از ضرب‌کننده پیشنهادی در جدول مقایسه دیده می‌شود. در ساختار پیشنهادی، پارامتر اصلی که دنبال رسیدن به آن بودیم، تأخیر بسیار کمتر بود. هرچه عرض بیت‌های ورودی بیشتر افزایش یابد، به دلیل افزایش تعداد بلوک‌های موازی، درصد بهبود سرعت الگوریتم پیشنهادی بیشتر می‌شود. در جدول مقایسه برای این که مقدار توان مصرفی و مساحت ساختار پیشنهادی هم محک زده شود بعضی از ساختارهای با توان مصرفی پایین هم آورده شده است.

۵- نتیجه‌گیری

در این مقاله یک الگوریتم ضرب با سرعت بالا، توان مصرفی پایین، بدون خط لوله و با مساحت سیلیکون مصرفی مناسب ارائه شده است. همچنین در بلوک‌های استفاده‌شده ساختارهای جدیدی با سرعت بالا و توان مصرفی کم پیشنهاد شده‌اند که شامل گیت XOR، یکی‌کردن فشرده‌ساز و جمع‌کننده و همچنین ساختار اصلاح‌شده درختی جمع‌کننده می‌باشد. این الگوریتم به آسانی قابل بسط برای تعداد بیت‌های بالاتری نیز است و با دو برابر شدن تعداد بیت‌های ورودی به علت موازی کارکردن بلوک‌ها تنها تأخیر اندکی به مسیر بحرانی اضافه می‌شود. در حقیقت هرچه تعداد بیت‌های ورودی افزایش یابند، الگوریتم پیشنهادی کارایی خود را بیشتر نشان می‌دهد.

مراجع

[1] A. D. Booth, "A signed binary multiplication technique," *the Quarterly J. of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236-240, 1951.

[2] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. on Electronic Computers*, vol. 13, no. 1, pp. 14-17, Feb. 1964.

[3] O. L. MacSorley, "High-speed arithmetic in binary computers," in *Proc. of the IRE*, vol. 49, no. 1, pp. 67-91, Jan. 1961.

[4] D. A. Pucknell and K. Eshraghian, *Basic VLSI Design: Systems and Circuits*, Prentice Hall Englewood Cliffs, New Jersey, USA, 1988.

[5] R. Fried, "Minimizing energy dissipation in high-speed multipliers," in *Proc. of the Int. Symp. on Low Power Electronics and Design*, pp. 214-219, Monterey, CA, USA, 8-20 Aug. 1997.

[6] H. Ghasemizadeh, E. Azadi, K. Hadidi, and A. Khoei, "A 1.6 GHz 16×16-bit low-latency pipelined booth multiplier," in *Proc. 19th Iranian Conf. on Electrical Engineering, ICEE'11,* 6 pp., Tehran, Iran, 17-19 May 2011.

[7] A. Weinberger and J. Smith, "A logic for high-speed addition," *Nat. Bur. Stand. Circ*, vol. 591, pp. 3-12, 1958.

[8] L. Morgan and D. Jarvis, "Transistor logic using current switching and routing techniques and its application to a fast 'carry' propagation adder," *Proc. of the IEE-Part B: Electronic and Communication Engineering*, vol. 106, pp. 467-468, 1959.

[9] M. Lehman and N. Burla, "Skip techniques for high-speed carry-propagation in binary arithmetic units," *IRE Trans. on Electronic Computers*, vol. 10, no. 4, pp. 691-698, Dec. 1961.

[10] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," *IEEE Trans. on Computers*, vol. 31, no. 3, pp. 260-264, Mar. 1982.

[11] J. Sklansky, "Conditional-sum addition logic," *IRE Trans. on Electronic Computers*, vol. 9, no. 2, pp. 226-231, Jun. 1960.

[12] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Trans. on Computers*, vol. 22, no. 8, pp. 786-793, Aug. 1973.

[13] H. C. Chow and I. C. Wey, "A 3.3 V 1 GHz high speed pipelined booth multiplier," in *Proc. IEEE Int. Symp. on Circuits and Systems, ISCAS'02,* pp. 1-1, Phoenix-Scottsdale, AZ, USA, 26-29 May 2002.

[14] K. H. Chen and Y. S. Chu, "A low-power multiplier with the spurious power suppression technique," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 7, pp. 846-850, Jul. 2007.

[15] A. Saha, D. Pal, and M. Chandra, "Low-power 6-GHz wave-pipelined 8b×8b multiplier," *IET Circuits, Devices & Systems*, vol. 7, no. 3, pp. 124-140, May 2013.

[16] Z. Huang and M. D. Ercegovac, "High-performance low-power left-to-right array multiplier design," *IEEE Trans. on Computers*, vol. 54, no. 3, pp. 272-283, 2005.

[17] S. R. Kuang, J. P. Wang, and C. Y. Guo, "Modified booth multipliers with a regular partial product array," *IEEE Trans. on Circuits and Systems II: Express Briefs*, vol. 56, no. 5, pp. 404-408, May 2009.

جدول ۱: مقایسه عملکرد ضرب‌کننده پیشنهادی در پروسس‌های مشابه.

طراحی	فناوری (نانومتر)	بیت‌های ورودی	تأخیر (نانوثانیه)	مساحت (میکرومتر مربع)	توان مصرفی (میلی‌وات)
[۱۳]	۳۵۰	۸×۸	۱	بیان‌نشده	۱۰۰
[۱۴]	۱۸۰	۱۶×۱۶	۵	۱۱۰۲۸	۱٫۲۱
[۱۵]	۱۸۰	۸×۸	۳٫۲۴	بیان‌نشده	۱۸٫۵۴
[۱۶]	۱۸۰	۳۲×۳۲	۶٫۹۹	۷۴۵۹۰	۴۱٫۷۲
[۱۷]	۱۸۰	۳۲×۳۲	۶٫۸۸	۹۸۷۹۴	۱۵٫۷۹۴
[۱۸]	۱۸۰	۶۴×۶۴	۲۰	۵۲۷۶۲۰	۲۱٫۳۸
[۱۹]	۱۸۰	۶۴×۶۴	۹	۴۴۳۳۲۰	۷۵۳
این کار (بعد از Layout)	۱۸۰	۲×۲	۰٫۳۲	۱۶۹	۰٫۱
		۴×۴	۰٫۶۷	۱۳۷۰	۰٫۴۸
		۸×۸	۱٫۲۵	۶۵۱۰	۲٫۱
		۱۶×۱۶	۲٫۱	۳۳۵۶۰	۹٫۲
		۳۲×۳۲	۳٫۰۵	۱۴۹۲۵۰	۴۰٫۲
		۶۴×۶۴	۴٫۲	۶۲۷۰۴۰	۱۷۲٫۸
تخمین	۱۸۰	۱۲۸×۱۲۸	۵٫۴ (حدودی)	۲۵۴۰۰۰ (حدودی)	۷۱۳ (حدودی)
این کار (شبیه‌سازی)	۹۰	۲×۲	۰٫۱۵۰	–	۰٫۰۲۵
		۴×۴	۰٫۲۹۵	–	۰٫۱۱۵
		۸×۸	۰٫۶۲۵	–	۰٫۵۰
		۱۶×۱۶	۱٫۰۶	–	۲٫۲۵
		۳۲×۳۲	۱٫۵۳	–	۹٫۷
		۶۴×۶۴	۲٫۰۳	–	۴۱٫۵
تخمین	۹۰	۱۲۸×۱۲۸	۲٫۵ (حدودی)	–	۱۶۸ (حدودی)

[18] V. S. Dimitrov, K. U. Jarvinen, and J. Adikari, "Area-efficient multipliers based on multiple-radix representations," *IEEE Trans. on Computers*, vol. 60, no. 2, pp. 189-201, Feb. 2011.

[19] P. Mokrian, M. Ahmadi, G. Jullien, and W. Miller, "A reconfigurable digital multiplier architecture," in *Proc. Canadian Conf. on Electrical and Computer Engineering. Toward a Caring and Humane Technology, CCECE'03,* vol. 1, pp. 125-128, Montreal, Canada, 4-7 May 2003.

ابراهیم حسینی تحصیلات خود را در مقاطع کارشناسی و کارشناسی ارشد مهندسی برق گرایش الکترونیک به‌ترتیب در سال‌های ۱۳۹۵ و ۱۳۹۸ از دانشگاه ارومیه دریافت کرد. در دوران کارشناسی ارشد بر روی طراحی انواع مدارهای دیجیتال سریع و با توان کم تحقیق نمود.

مرتضی موسی‌زاده در سال ۱۳۸۲ مدرک کارشناسی مهندسی برق گرایش الکترونیک خود را از دانشگاه علم و صنعت ایران و در سال ۱۳۸۵ مدرک کارشناسی ارشد مهندسی برق گرایش مدارهای مجتمع خود را از دانشگاه ارومیه دریافت نمود. پس از آن به دوره دکترای مهندسی برق طراحی مدارهای مجتمع در دانشگاه ارومیه وارد گردید و در سال ۱۳۹۲ موفق به اخذ درجه دکترا در مهندسی برق از دانشگاه مذکور گردید. دکتر موسی‌زاده از سال ۱۳۸۳ تا ۱۳۹۲ به عنوان مهندس طراح مدارهای مجتمع در شرکت نیمه‌هادی ارومیه مشغول به کار بود. از سال ۱۳۹۳ در دانشکده مهندسی برق و کامپیوتر دانشگاه ارومیه مشغول به فعالیت گردید و اینک نیز عضو هیأت علمی این دانشکده می‌باشد. زمینه‌های علمی مورد علاقه نام‌برده متنوع بوده و شامل موضوعاتی مانند طراحی مبدل‌های داده بسیار سریع، مدارهای دیجیتال و آنالوگ و پردازنده‌های شبکه‌های عصبی می‌باشد.