دانشکده فنی و مهندسی
گروه برق - الکترونیک

پایان نامه جهت اخذ درجه کارشناسی ارشد رشته برق
گرایش الکترونیک

عنوان پایان نامه

طراحی و پیاده سازی یک *ALU* دیجیتال ۳۲ بیتی با سرعت *1GHZ* در پروسه *0.18µm CMOS*

استاد راهنما :
دکتر محمد مهدی کارخانه چی

نگارش :
شهاب تیموری

مهر ۱۳۹۲

دانشکده فنی مهندسی
گروه الکترونیک

پایان نامه جهت اخذ درجه کارشناسی ارشد رشته برق گرایش الکترونیک

نام دانشجو:
شهاب تیموری

تحت عنوان :

طراحی و پیاده سازی یک *ALU* دیجیتال ۳۲ بیتی با سرعت *1GHZ* در پروسه *0.18µm CMOS*

در تاریخ ۲۱ مهر ۱۳۹۲ توسط هیأت داوران زیر بررسی و با درجه به تصویب نهایی رسید.

۱- استاد راهنما آقای دکتر محمد مهدی کارخانه چی با مرتبه ی علمی استادیار امضاء

۱- استاد داور داخل گروه آقای دکتر محسن حیاتی با مرتبه ی علمی دانشیار امضاء

۲- استاد داور داخل گروه آقای دکتر آرش احمدی با مرتبه ی علمی استادیار امضاء

تقدیم به:

روح پاک خواهرم

که عالمانه به من آموخت تا چگونه در عرصه زندگی، ایستادگی را تجربه نمایم.

چکیده

با پیشرفت سریع تکنولوژی و تمرکز بر کارهای پیچده علمی ، که نیاز به شبیه سازیهای سنگین و زمان گیر دارد اهمیت پردازنده های سریع در سیستم های کامپیوتری بیش از پیش نمایان می شود. در این میان ضرب کننده های دیجیتال و واحد محاسبات منطقی که به عنوان بخشی از میکروپروسسورها و *DSP* ها که دارای بحرانی ترین تاخیر[1]است اهمیت ویژه ای دارد چون کارایی کل چیپ مستقیماً با کارایی ضرب کننده داخلیش در ارتباط است. یکی از روشهای افزایش سرعت و کارایی در پردازنده ها استفاده از تکنیک خط لوله[2] است که در سالهای اخیر در بسیاری از کاربردها اعمال می شود. عناصر خط لوله اغلب به صورت موازی و یا در مد زمان قطعه قطعه اجرا می شوند که درآن صورت برای ذخیره سازی بین عناصر به المان های حافظه دار نیاز می باشد. هدف ما طراحی و پیاده سازی یک ضرب کننده ی ۳۲ بیتی با سرعت ۱ گیگا هرتز در پروسه ی*0.18µm CMOS* باحداقل تعداد سیکل کلاک ممکن بود، تا علاوه بر تاخیر کلی کم، سخت افزار و توان مصرفی کمی داشته باشد.

برای این منظور سعی شده با طراحی مدارات جدید در بعضی از قسمت ها، توازن در تاخیر طبقات برقرار شود تا ماکزیمم بهره برداری از تکنیک *Pipeline* صورت گیرد. بطور مثال مدار *Booth Encoder/Decoder* جدید طوری طراحی شده که حاصلضرب های جزئی[3] در یک سیکل و با تاخیر کوچک (*380 ps*) تولید شوند. از طرفی با استفاده از یک تکنیک بهینه یافته ی ریاضی و مداری، ردیف اضافی ساختار *Modified Booth* بدون استفاده از یک سیکل اضافی و همزمان با تولید *Partial Product* ها در سیکل اول، حذف می شود حذف این ردیف علاوه بر کاهش توان مصرفی، سخت افزار و تاخیر کلی، به منظم تر شدن کل ضرب کننده می انجامد .از آنجاییکه روش های قبلی برای حذف این بیت ، تاخیر بسیار زیادتری داشتند عملاً حذف بیت اضافی امکان پذیر نبود. ولی در این ضرب کننده مدارات سیکل اول طوری بهینه شده اند که ۱۶ ردیف پس از تاخیری حدود *380 ps* تولید می شوند و به دلیل منظم بودن ساختار ایجاد شده برای جمع عمودی حاصلضرب های جزئی فقط کمپرسور ۴ به ۲ کافیست تا پس از ۴ سیکل ورودی های جمع کننده ی نهایی تولید شوند.همچنین جمع کننده ی نهاییِ جدید طوری طراحی شده که برای سرعت های مختلف درساختارهای *Pipeline* شده قابل شکسته شدن بوده و علاوه بر سخت افزار کوچک ، تاخیر کمی دارد. بطوریکه در ۳ سیکل جمع ۵۱ بیت در این ضرب کننده تحقق می یابد . مدار *CLA* سریع و همچنین ارائه ی یک مدار تعیین کننده ی رقم نقلی هر بلوک *(BCG)* از اصلی ترین عللی هستند که سرعت این جمع کننده ی کوچک را قابل مقایسه با جمع کننده های پیچیده و حجیم کرده است در نهایت این ضرب کننده می تواند ۱ میلیارد نمونه ی ۳۲ بیتی را در هر ثانیه از ورودی ها گرفته و پس از ۸ سیکل نتیجه ی ضرب علامت دارشان را تولید کند. نتایج شبیه سازی نشان می دهد باروش *Pipeline* سرعت ضرب کننده به بیش از ۶ برابر حالت عادی میرسد، درحالیکه مساحت سخت افزار (*1.5mm× 1.87mm*) تقریباً ۲/۲ برابر و توان مصرفی آن (*543 mw*) می شود.

---

1 -Critical path delay

2 -Pipeline

3 -Partial Product

فهرست مطالب

عنوان **صفحه**

فهرست اشکال

عنوان **صفحه**

فهرست جداول

عنوان **صفحه**

فصل اول

اهداف و الگوریتم های مختلف ضرب کننده ها

۱-۱- مقدمه

ریزپردازنده[1] ، واحد پردازش مرکزی در هر کامپیوتر است، که پردازش های مختلف در آن انجام می گیرد. با استفاده از تکنولوژی ساخت قطعات نیمه هادی می توان این واحد پردازنده (*CPU*) را کوچک نموده و در یک بسته بندی یا تراشه[2] قرار داد، که به آن ریزپردازنده یا میکروپروسسور گویند. پس در حالت کلی کلمه *CPU* یک مفهوم کلی تری از کلمه ریزپردازنده است. با پیشرفت تکنولوژی، هم اکنون تمام *CPU*ها بصورت ریزپردازنده ساخته می شود، در صورتی که در کامپیوترهای نسل های پایین تر به دلیل پایین بودن تکنولوژی، امکان ساخت *CPU* بصورت متمرکز و در داخل یک بسته بندی (تراشه) وجود نداشت. یک ریزپردازنده مجموعه هایی از دستورات به زبان ماشین را اجرا می کند. براساس این دستورات، ریزپردازنده سه نوع عملیات اصلی را انجام میدهد:

ریزپردازنده با استفاده از واحد *ALU (Arithmetic Logic Unit)* قادر به انجام عملیات محاسباتی نظیر جمع، تفریق، ضرب و ... ، عملیات منطقی نظیر *AND* ، *NAND* و... و می تواند داده ها را از یک حافظه به حافظه دیگر منتقل کند. قادر به تصمیم گیری و جهش به مجموعه جدیدی از دستورات بر اساس این تصمیم گیری است. از معیارهای طبقه بندی میکروپروسسور ، سرعت پردازش اطلاعات در آنهاست، که با واحد مگا هرتز بیان می شود. هر مگا هرتز، معادل انجام یک میلیون دستور در ثانیه می باشد. هرچقدر سرعت ساعت یک ریزپردازنده بیشتر باشد، آن ریزپردازنده سریع تر عمل می کند. *CPU* به طور کلی از چند قسمت تشکیل شده است. مانند واحد محاسبه و منطق ، واحد کنترل ، واحد حافظه ثبات و ...

ثباتها، حافظه های ناپایدار برای ذخیره سریع و موقتی داده هایی هستند که باید پردازش شوند و در *CPU* قرار گرفته اند. این نوع حافظه ها می توانند داده ها و دستورالعمل های در حال پردازش را به سرعت دریافت، ذخیره و منتقل کنند. انباره[3] ، ثباتی است که نتایج عملیات ریاضی و منطقی انجام شده به وسیله واحد *ALU* را در خود نگه می دارد. وظیفه ی واحد کنترل[4] ، کنترل دریافت داده ها از واحد ورودی، کنترل عملیات داخلی *CPU* و کنترل ارسال اطلاعات به واحد خروجی می باشد. این واحد مشابه یک سیستم عصبی برای کنترل سایر بخشهای کامپیوتر عمل می کند.

---

1 - CPU
2 -IC
3 -Accumulator
4 -CU- Control Unit

در علم کامپیوتر، واحد حساب و منطق (به اختصار *ALU*)، یک مدار دیجیتالی است که عملیات محاسباتی و منطقی را انجام می دهد. *ALU* بخش بنیادی واحد پردازش مرکزی *(CPU)* یا هر ریز پردازنده ی دیگری است. پردازنده های مرکزی یا پردازنده های گرافیکی مدرن، شامل *ALU* های بسیار قدرتمند و پیچیده هستند و ممکن است بیش از یک *ALU* داشته باشند. اکثر عملیات پردازنده به وسیله یک یا چند *ALU* انجام می شود. *ALU* دیتا را از رجیسترهای ورودی بارگذاری می کند، سپس یک واحد کنترل *(Control unit)* به *ALU* می گوید که چه عملیاتی را بر آن اطلاعات انجام دهد. و درآخر *ALU* نتایج را بر روی یک رجیستر خروجی ذخیره می کند.

اکثر *ALU* ها می توانند این عملیات را انجام دهند:

- عملیات محاسباتی صحیح (جمع ، تفریق و بعضی ضرب و ...)
- عملیات بیتی منطقی (*NAND*، *NOR* ، *XOR*، *XNOR* و ...)
- عملیات انتقال بیتی[5] انتقال یا چرخش یک کلمه با تعداد مشخصی بیت به راست یا چپ با/بدون گسترش علامت.

۱-۲- اهداف :

با پیشرفت سریع تکنولوژی و تمرکز بر کارهای پیچده علمی ، که نیاز به شبیه سازی های سنگین و زمان گیر دارد اهمیت پردازنده های سریع در سیستم های کامپیوتری بیش از پیش نمایان می شود. در این میان ضرب کننده های دیجیتال و واحد محاسبات منطقی که به عنوان بخشی از میکروپروسسور ها و *DSP* ها که دارای بحرانی ترین تاخیر (*Critical path delay*) است اهمیت ویژه ای دارد چون کارایی کل چیپ مستقیماً با کارایی ضرب کننده داخلیش در ارتباط است. یکی از روش های افزایش سرعت و کارایی در پردازنده ها استفاده از تکنیک *Pipeline* (خط لوله) است که درسالهای اخیر دربسیاری ازکاربردها اعمال می شود. عناصرخط لوله اغلب به صورت موازی و یا درمد زمان قطعه قطعه اجرا می شوند که درآن صورت برای ذخیره سازی بین عناصر به المان های حافظه دار نیاز می باشد. هدف ما طراحی و پیاده سازی یک ضرب کننده ی۳۲ بیتی با سرعت ۱ گیگا هرتز در پروسه ی *0.18µm CMOS* باحداقل تعداد سیکل کلاک ممکن بود،تا علاوه برتاخیرکلی کم، سخت افزار و توان مصرفی کمی داشته باشد .

همچنین جمع کننده ی نهاییِ جدید طوری طراحی شده که برای سرعت های مختلف در ساختارهای *Pipeline* قابل شکسته شدن بوده وعلاوه برسخت افزارکوچک، تاخیرکمی دارد. بطوریکه در ۴ سیکل جمع ۶۴ بیت در این ضرب کننده تحقق می یابد. مدار *CLA* سریع و همچنین ارائه ی یک مدار تعیین کننده ی رقم نقلی هر بلوک *(BCG)* از اصلی ترین عللی هستندکه سرعت این جمع کننده ی کوچک را قابل

5 -*Bit-Shifting*

مقایسه با جمع کننده های پیچیده و حجیم کرده است در نهایت این ضرب کننده می تواند ۱ میلیارد نمونه ی۳۲ بیتی رادرهرثانیه از ورودی ها گرفته و پس از ۸ سیکل نتیجه ی ضرب علامت دارشان را تولیدکند. نتایج شبیه سازی نشان می دهد با روش *Pipeline* سرعت ضرب کننده به بیش از ۶ برابر حالت عادی می رسد، در حالیکه مساحت سخت افزار ($1.5mm \times 1.87mm$) تقریباً ۲/۲ برابر و توان مصرفی آن ($543\ m$) می شود.

۱-۳- سازماندهی :

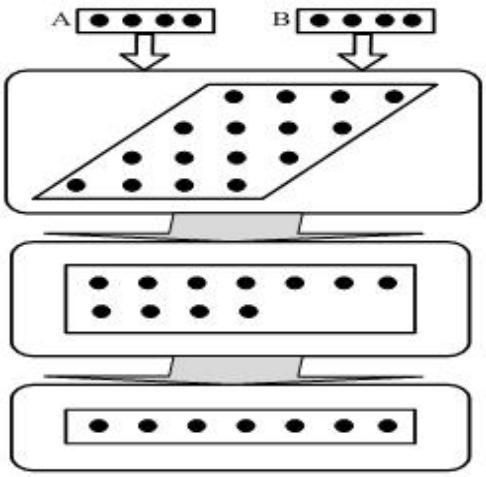
در بخش های این پایان نامه که شامل شش فصل می باشد قسمت های مختلف پایان نامه مورد بررسی و تجزیه وتحلیل قرار می گیرد. فصل ۱ مروری بر مفاهیم و اجزاء یک ضرب کننده ی دیجیتال و ساختارهای متداول دارد. فصل ۲ شامل بررسی ساختارهای جمع کننده های دیجیتال و ویژگیهای هریک است. فصل ۳ به توضیح ساختار ضرب کننده ی ارائه شده و طراحی مدارات قسمت های مختلف اختصاص یافته است. در فصل ۴ ساختار *ALU* مورد بررسی قرار می گیرد. و در فصل ۵ نتایج شبیه سازی آورده شده به همراه جمع بندی نتایج کلی پایان نامه و مقایسه با کار های انجام شده در زمینه مشابه، نیز در پایان فصل گنجانده شده است.

در این بخش توصیف مختصری از ضرب کننده های دیجیتالی شامل ساختار و اجزای وابسته آنها نمایش خواهیم داد. همچنین بعضی تکنیک ها که برای افزایش سرعت ضرب کننده ها ارائه شده اند، مورد بحث قرار خواهد گرفت. سپس مروری بر توان مصرفی در مدار های *CMOS* به همراه تکنیک های اولیه برای کاهش توان مصرفی خواهیم داشت.

۱-٤- ساختار ضرب کننده ی دیجیتال:

ضرب دیجیتالی شیفت و جمع یافتن یک سری از بیت ها است که در آن دو عدد، ضرب کننده و ضرب شونده، برای تولید خروجی ترکیب می شوند، با فرض اینکه نمایش بیتی ضرب شونده بصورت:

$X = X_{n-1} \ldots X_1 X_0$ و ضرب کننده بصورت $Y = Y_{n-1} \ldots Y_1 Y_0$ باشد. برای شکل گیری حاصلضرب، حداکثر N ردیف از ضرب شونده بصورت شیفت یافته (برای ضرب بدون علامت) جمع می شوند. کل فرآیند شامل ۳ مرحله می باشد، تولید حاصلضرب جزئی(*Partial Product*)، کاهش حاصلضرب جزئی و جمع نهایی. ساختار کلی ضرب دیجیتال در شکل (۱-۱) نشان داده شده است.

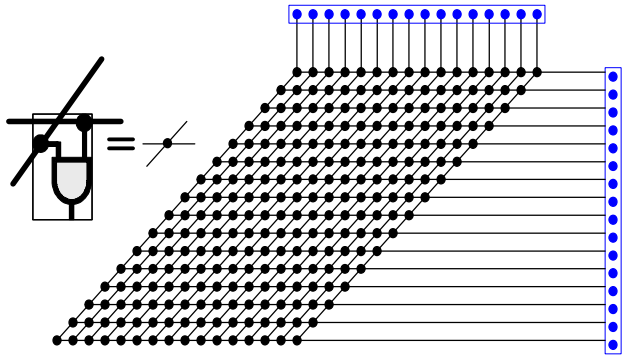


شکل ۱-۱: ساختار کلی ضرب کننده ی دیجیتال (۳-بیتی)

۱-۵- تولید حاصلضرب های جزئی(*Partial Product*) :

به عنوان مرحله اولیه در ضرب دیجیتال، نیاز به تولید N کپی شیفت یافته از ضرب شونده می باشد، که باید در مراحل بعدی جمع شوند. ارزش بیت ضرب کننده مشخص می کند که کپی شیفت یافته از ضرب شونده باید جمع شود یا نه. به این صورت که اگر بیت i-اُم $(0 \leq i \ll n-1)$ از ضرب کننده "*1*" باشد، کپی شیفت یافته از ضرب شونده جمع می شود و اگر"*0*" باشد کپی شیفت یافته از ضرب شونده جمع نمی شود. یک گیت *AND* می تواند این عملیات را بصورت $(X_i\ Y_j)\ AND$ $(0 \leq j \ll n-1$ و $0 \leq i \ll n-1)$ انجام دهد و نتیجه ی این عملیات حاصلضرب جزئی (*Partial Product*) نامیده می شود. شکل(۱-۲)، یک ساختار ذوزنقه ای[۶] با نام (*PPA*) به نمایش می گذارد که درآن بیت های حاصلضرب جزئی در ستون هایی برای جمع شدن جهت تشکیل حاصلضرب نهایی مرتب شده اند. این فرآیند *Partial Product Generation* نامیده می شود. همان طور که اشاره شد تمام بیت ها به صورت موازی در *PPA* شکل گرفته اند، بنابراین تأخیر استاتیک تمام بیت ها برابر است. عرض *PPA* متناسب با اندازه ضرب شونده است، در حالی که ارتفاع آرایه متناسب با اندازه ضرب کننده می باشد. بعضی ستون ها بیت های بیشتری نسبت به بقیه دارند و بنابراین عمل جمعِ بیشتری باید بصورت عمودی بیت های این ستون ها را به دو ردیف کاهش دهد. همانطور که در شکل زیر مشخص است این ستون ها در بیت های میانی قرار دارند.

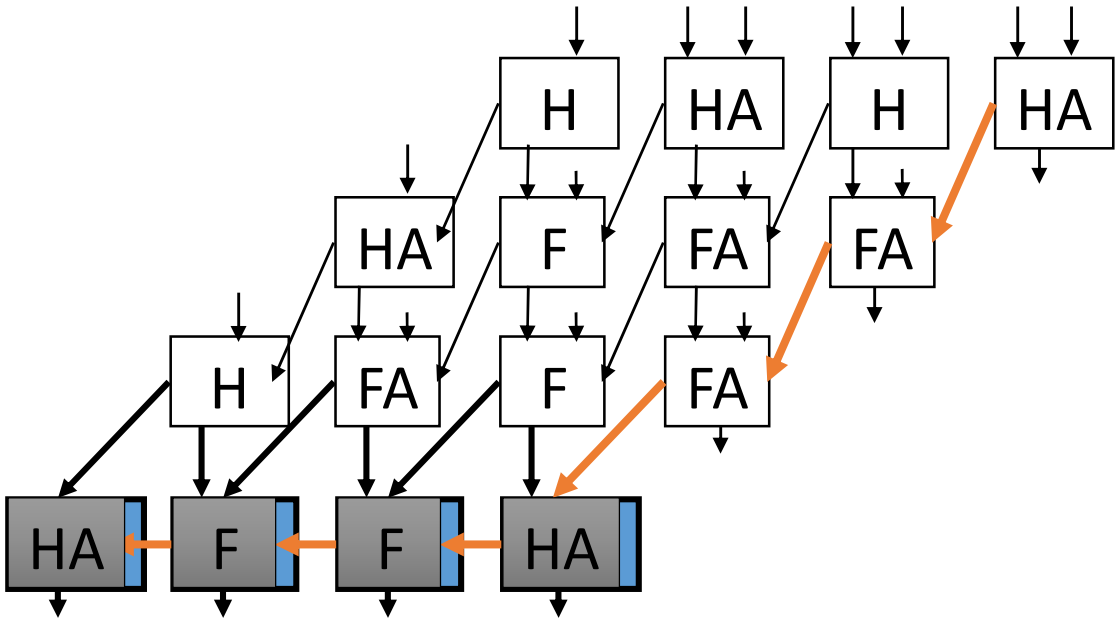
6 -*Partial product Array*



شکل ۱-۲: تولید حاصلضرب های جزئی در یک ضرب کننده ی۱۶بیتی

۱-۶- کاهش حاصلضرب های جزئی:

پیاده سازی یک ضرب کننده ی دیجیتالی با قابلیت بالا، تا حدود زیادی بستگی به روش جمع کردن بیتِ حاصلضرب های جزئی آن ضرب کننده دارد. از آنجایی که تولید هر ردیف شیفت یافته از ضرب شونده تأخیری متناسب با عرض ضرب شونده خواهد داشت بلوک ضرب کننده نیاز به مقدار زیادی زمان برای انجام عملیات خواهد داشت، که اگر برای کاهش این ردیف ها از جمع کننده های معمولی استفاده شود، تاخیر زیادی ایجاد خواهد شد. به خصوص زمانی که تعداد بیت ضرب کننده زیاد باشد. به همین خاطر حاصلضرب های جزئی اغلب توسط یک تکنیک به نام *Carry Save Addition* کاهش داده می شوند(شکل ۱-۳).



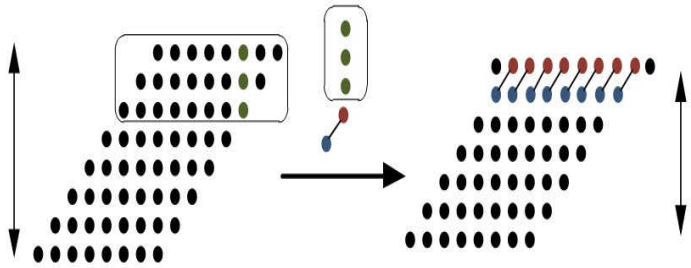
شکل ۱-۳: کاهش حاصلضرب های جزئی با روش ***Carry Save Addition***

در این روش جمع شدن پیاپی در یک مرحله کلی اجازه می دهد تا جمع دو بیت از هر دو ردیف ضرب شونده های شیفت یافته، براحتی توسط تمام جمع کننده ی معمولی تحقق یابد. همان طور که در شکل ۱-۳

مشخص شده است با این روش در یک ضرب کننده ی N بیتی، تبدیل حاصلضرب های جزئی به نتیجه ی نهایی کل ضرب کننده تقریباً تاخیری معادل $2N$ برابر تاخیر یک تمام جمع کننده می باشد و با این حال اگر N عدد بزرگی باشد، تاخیر ضرب کننده نسبتاً خیلی زیاد می شود و برای سرعت های بالا (و بخصوص ضرب کننده های با بیت بالا) باید دنبال روش های دیگری باشیم. در ادامه به چند روش سریعتر و متداول برای کاهش حاصلضرب های جزئی اشاره می کنیم.

۱-٦-۱- کاهش به روش آرایه ای[۷] :

روش های مختلفی برای کاهش حاصلضرب های جزئی ذوزنقه ای شکل وجود دارد. اصلی ترین مزیتی که این روش ها نسبت به روش *Carry Save* دارند این است که حاصلضرب های جزئی با تاخیر کمی به دو ردیف نهایی تبدیل می شوند و چون بیت های این دو ردیف همزمان تولید می شوند، امکان استفاده از جمع کننده های سریعی مثل *Carry Select Adder (CSA)* یا *Carry Lookahead Adder (CLA)* به عنوان جمع کننده ی نهایی وجود دارد. در این بخش ساده ترین روش، به نام کاهش به روش آرایه ای، توصیف خواهد شد. در شکل (۱-۴) *PPA* ذوزنقه ای برای ضرب کننده ی ۸×۸ بیتی نشان داده شده است (هر نقطه نمایش دهنده یک بیت حاصلضرب است). سه ردیف اول توسط یک سری از تمام جمع کننده ها به ۲ ردیف تبدیل می شوند. بدین ترتیب دو ردیف تولید شده با پنج ردیف باقیمانده ترکیب شده و ۷ ردیف باقی می ماند. این روال ادامه می یابد تا حاصلضرب های جزئی در نهایت به ۲ ردیف کاهش یابند. این طراحی با این که ساده است، ولی سرعت کندی دارد و شاید تنها مزیتش وقتی است که قرار باشد تنها سیم های کوتاه برای اتصال عمودی-افقی و قطری سلول های تمام جمع کننده بکار برده شود *[۱۸]*. در ادامه روش های مکمل یافته ی این روش با سرعت بالاتر توضیح داده می شود.

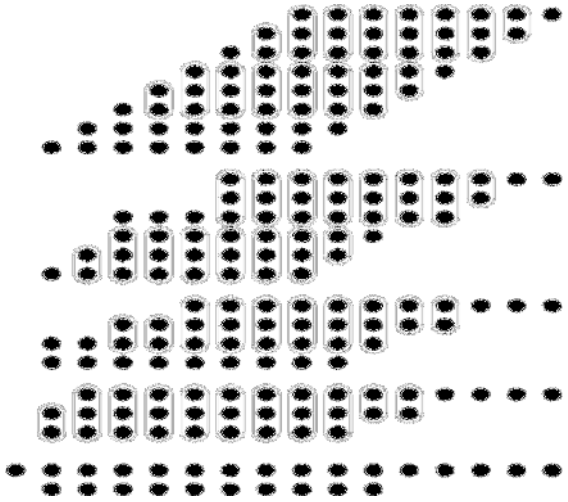


شکل ۱-٤: کاهش حاصلضرب های جزئی به روشِ *Array style*

7 -*Array style reduction*

۱-۶-۲- کاهش به روش درخت والاس[8]:

در سال ۱۹۶۶، *C.S Wallace* به این نتیجه رسید که می توان ساختاری یافت تا عملیات جمعِ حاصلضرب های جزئی را به طور موازی انجام دهد. بنابراین تأخیر بطور چشم گیری کاهش می یابد. در این مقاله ی تاریخی *Wallace* ، روشی متفاوت از جمع موازی بیت های حاصلضرب جزئی با استفاده از یک درخت جمع کننده ی ذخیره ی (*Carry Save*) معرفی کرد، که به درخت والاس (*Wallace Tree*) معروف شده است. در شکل (۱-۵) *PPA* ذوزنقه ای برای ضرب کننده ی ۸×۸ بیتی نشان داده شده است. سه ردیف اول توسط یک سری از تمام جمع کننده ها به ۲ ردیف تبدیل می شوند. به همین ترتیب سه ردیف دوم نیز همزمان با سه ردیف اول به ۲ ردیف کاهش می یابند. بنابراین در یک ضرب کننده ی ۸×۸ بیتی حاصلضرب های جزئی از ۸ ردیف به ۶ ردیف و با تاخیر یک تمام جمع کننده کاهش می یابد. این روال ادامه می یابد. یعنی ۶ ردیف باقیمانده توسط تمام جمع کننده ها به ۴ ردیف و سپس ۳ و در نهایت به ۲ ردیف کاهش می یابند. این طراحی به دلیل موازی صورت گرفتن عمل جمع سرعت بالاتری نسبت به روش های ذکر شده دارد و تاخیر کاهش حاصلضرب های جزئی آن بصورت لگاریتمی با تعداد بیت ضرب کننده متناسب است. بنابراین برای ضرب کننده های با بیت بالا مناسب تر است.



شکل ۱-۵: کاهش حاصلضرب های جزئی یک ضرب کننده ی ۸×۸ بیتی به روشِ *Wallace Tree*

با این حال اصلی ترین عیب درخت *Wallace* چینش[9] نامنظم آن نسبت به ساختار آرایه است. بی نظمی در چینش این ساختار اثر بار (*Load*) را هم زیاد می کند(به خاطر وجود سیم ها ی بیشتر). همچنین ذکر این نکته حائز اهمیت است که عرض (تعداد بیت) جمع کننده ی نهایی در درخت *Wallace* تقریباً $N - N^{\log_{3/2}2}$بوده، در حالیکه این عرض در ساختار (*Carry Save*) برابر با N می باشد *[۲۳]*. ولی با همه ی

8 -Wallace Tree

9 -Layout