تولید سلول استاندارد بهینه در تکنولوژی CMOS استاتیک

شادرخ سماوی*، افسانه ترکیان** و پژمان خدیوی***
دانشکده برق و کامپیوتر، دانشگاه صنعتی اصفهان

(دریافت مقاله: ۷۹/۱۱/۱۱ – دریافت نسخه نهایی: ۸۱/۶/۳)

چکیده : ساخت یک مدار مجتمع با سطح کمتر، علاوه بر کاهش هزینهٔ ساخت لازم، اغلب منجر به اثرات مثبتی در کاهش توان مصرفی و افزایش قدرت پردازش می‌شود. برای ساخت یک مدار مجتمع که قابلیت انجام یک تابع منطقی را داشته باشد و در عین حال مساحت کمتری اشغال کند، باید بتوان آن تابع منطقی را در تکنولوژی CMOS با نفوذ پیوسته ساخت. برای این کار باید مسیر اویلر پیوسته‌ای برای آن تابع منطقی به دست آورد. هر انفصال در ناحیهٔ نفوذ باعث افزایش مساحت و کاهش کارایی می‌شود.
برای طراحی یک تابع منطقی در تکنولوژی CMOS ایستا، روشهای مختلفی ارائه شده که اکثر آنها بر پایه روش اهارا ست. در این مقاله الگوریتمی ارائه شده که برای یک تابع منطقی می‌تواند مسیر اویلر را پیدا کند و ساخت مدار مجتمع را با نفوذ پیوسته و حداقل سطح، امکانپذیر می‌سازد. چنانچه تابع مورد نظر به هیچ عنوان مسیر اویلر پیوسته‌ای نداشته باشد می‌توان، زیرمسیرهای اویلر غیرپیوسته‌ای به دست آورد. علاوه بر آن، الگوریتم پیشنهادی، اطلاعات کاملی از مسیر اویلر پیدا شده ارائه می‌دهد که به انجام جانمایی و ساخت مدار مجتمع آن تابع منطقی، کمک می‌کند.

واژگان کلیدی: طراحی VLSI، روش اهارا ، CMOS ، نفوذ پیوسته، سلول استاندارد.

Optimized Standard Cell Generation for Static CMOS Technology

S. Samavi, A. Torkian, and P. Khadivi
Department of Electrical and Computer Engineering, Isfahan University of Technology

Abstract: *Fabrication of an integrated circuit with smaller area, besides reducing the cost of manufacturing, usually causes a reduction in the power dissipation and propagation delay. Using the static CMOS technology to fabricate a circuit that realizes a specific logic function and occupies a minimum space, it must be implemented with continuous diffusion runs. Therefore, at the design stage, an Eulerian path should be found for the logic function. Every discontinuity causes an increase in the area as well as a reduction in the clock rate and performance.*

The realization of a logic function using the static CMOS technology is done through different methods, most of which are based on the Uehara's method. In this paper, an algorithm is suggested that finds the Eulerian path and allows the implementation of the circuit with continuity in the diffusion region that results in minimum area. In a case where there is no Eulerian path, the possible sub-paths are found. In addition, the algorithm gives information that helps the layout generation.

Keywords: *VLSI , Uehara's method, Static CMOS, Continous diffusion, Standard cell.*

---

* -دانشیار ** -کارشناسی ارشد *** -دانشجوی دکترا

۱ـ مقدمه

تولید جانمایی[۱] به معنی تبدیل توصیف منطقی یک سیستم به ماسکهای فیزیکی به منظور ساخت مداراست. پیچیدگی این فرایند به قدری است که روش سلول استاندارد[۲] در بین طراحان رواج بسیار پیدا کرده است. در این روش طرح را به قطعات کوچکتر تقسیم می‌کنند تا پیچیدگی کاهش یابد. تولید جانمایی به زیر فرایندهای دیگری تقسیم می‌شود که عبارت‌اند از:

تحلیل منطقی[۳] و نگاشت تکنولوژی[۴]، تولید سلول، جایگذاری، مسیریابی کلی و جزئی[۵] و فشرده‌سازی[۶]. تمام زیر فرایندهای بالا بر مبنای سلولهای استاندار قرار دارند. فرضا در فرایند نگاشت تکنولوژی، سلولها از یک کتابخانهٔ استاندار انتخاب و در محل مورد نظر قرار می‌گیرند. به این ترتیب برای کمینه کردن سطح کل جانمایی، باید هر سلولی که در کتابخانه قرار دارد، دارای ساختار کمینه باشد. از طرف دیگر طرح VLSI یک تابع منطقی خود می‌تواند به عنوان یک سلول جدید به کتابخانهٔ روش سلول استاندارد، اضافه شود.

مساحت هر سلول برای ساخت به صورت زیر محاسبه می‌شود:

عرض سلول × ارتفاع سلول = مساحت لازم

= عرض سلول

(۱+ تعداد انفصالات + تعداد ورودیها)× بزرگی واحد پایه[۷]

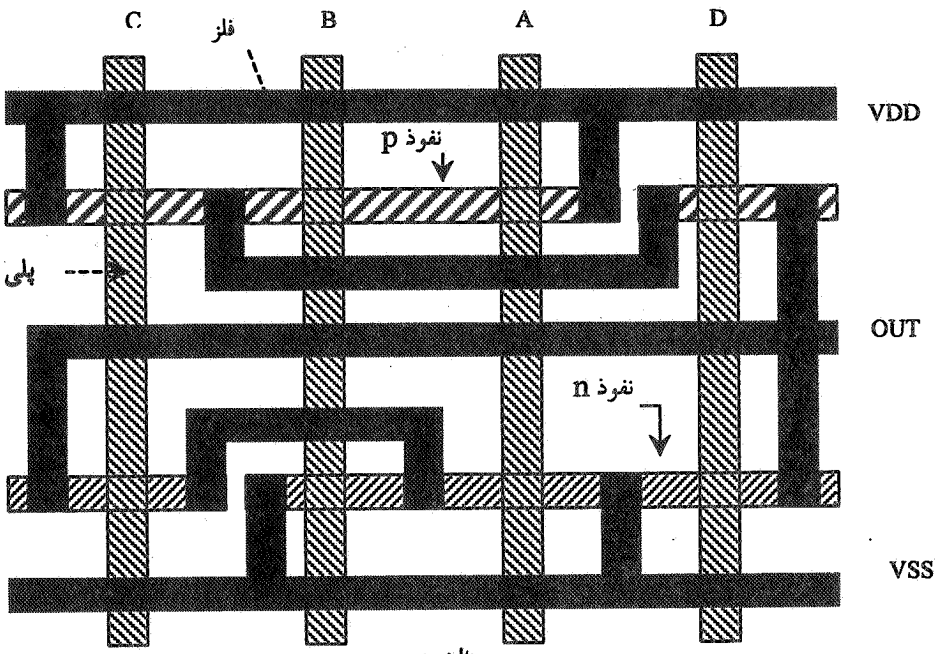
ارتفاع سلول، خود تابع پهنای ترانزیستور و فشردگی سیم‌بندی است. افزایش فشردگی سیم‌بندی باعث کاهش ارتفاع و کاهش مساحت می‌شود که این عامل و بزرگی واحد پایه نیز به تکنولوژی ساخت بستگی دارند. همان‌گونه که ارتفاع سلول در تکنولوژی ساخت با دو لایه فلز نسبت به تکنولوژی ساخت یک لایه فلز کمتر است [۱].

طراحی با روش بالا، نیاز به سلولهای منظم دارد. بنابراین سلولها باید هم ارتفاع باشند، تا در موقع قرار دادن در نقشهٔ ماسکها، خطوط V_{DD} (تغذیه) و V_{SS} (زمین) سلولها با یکدیگر انطباق داشته باشند [۲]. علاوه بر یکسانی ارتفاع سلولها در روش سلول استاندارد برای سلولهای کتابخانه عوامل دیگری نظیر اتلاف توان، تأخیر انتشار و مصونیت در مقابل نویز[۸] نیز مهم‌اند.

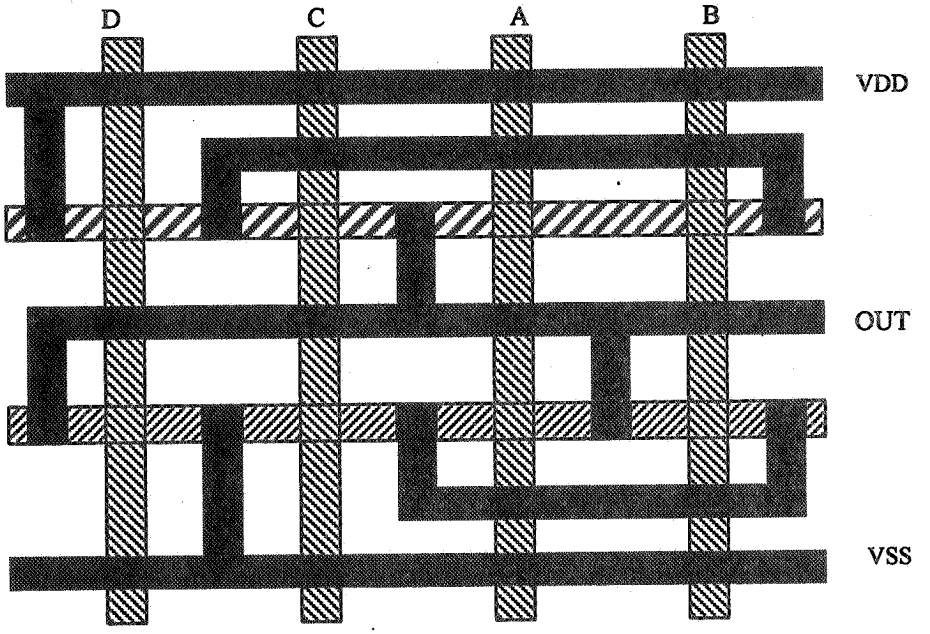
عرض سلول، تابعی از توان ترانزیستورهای سلول و همچنین تعداد ورودیها و انفصالات است. تعداد ورودیها به تابع منطقی وابسته است. بنابراین تنها با کاهش انفصالات می‌توان به حداقل سطح رسید و بهترین حالت، عدم انفصال یا همان نفوذ پیوسته است. تاکنون کارهای زیادی بر روی نحوهٔ جانمایی و جایگذاری ترانزیستورها صورت گرفته که اکثرا بر پایهٔ کاهش انفصالات برای کاهش مساحت بوده است[۳]و[٤]و[۸]. هوانگ [۸] توسط نرم‌افزاری به نام THEDA.P سعی کرده است تعداد انفصالات و همچنین ارتفاع سلول را کاهش دهد. روشی که تعداد انفصالات را با آن کاهش می‌دهند بر الگوریتم اُهارا مبتنی است [۳].

هدف از روش اُهارا و روشهای مبتنی بر آن افزایش اتصالات سورس و درین بین ترانزیستورهای همجواراست. این کار با تبدیل توصیف مدار به یک زوج گراف دو بعدی و یافتن مسیر اویلر[۹] مشابه برای هر دو گراف صورت می‌گیرد. مسیر اویلر به طریقی است که تمام اضلاع گراف را به صورت پیوسته طی می‌کند و از هیچ ضلعی دوبار عبور نمی‌کند. به این ترتیب نیازی به اتصال فلزی بین دوترانزیستور نخواهد بود و می‌توان با امتداد ناحیه نفوذ، مانند آنچه در شکل (۱) دیده می‌شود، دوترانزیستور را به هم وصل کرد.

در شکل (۱) دیده می‌شود که برای جانمایی یک تابع منطقی، می‌توان همبندیهای متفاوتی داشت. همبندی‌ای که ایجاد انفصالات در ناحیهٔ نفوذ می‌کند (شکل (۱ـالف)) پهنای سلول بیشتری خواهد داشت. البته در نمودارهای شکل (۱) قوانین طراحی مانند آنچه در ماسکها باید باشد، رعایت نشده و صرفاً همبندی جانمایی مد نظر بوده است.



(الف)



(ب)

شکل ۱ - مقایسهٔ دو روش ساخت برای تابع منطقی $F=\overline{(A+B)C+D}$

الف- ناپیوستگی در ناحیهٔ نفوذ ب- پیوستگی در ناحیهٔ نفوذ

[DOR: 20.1001.1.2251600.1381.21.2.2.6]

شیباتانی [۹] اگر چه روش ا هارا را برای سلولهای ASIC معمولی مناسب می‌داند ولی برای حافظه‌های خاص DRAM مانند EDO-DRAM و SDRAM روشی دوبعدی را پیشنهاد می‌کند. کارلسون [۱۰] روشی برای بهینه‌سازی سلولها مبتنی بر مسیر اویلر ارائه داده است که به آن عدم وابستگی دوگانه[10] می‌گوید. در این روش بهینه‌سازی را برای ناحیهٔ ترانزیستورهای p جدا از آن چه برای ناحیهٔ n انجام می‌گیرد، صورت می‌دهد. در واقع روش اوهارا را به طور جداگانه دوبار استفاده می‌کند. به این ترتیب درجهٔ آزادی در طراحی افزایش می‌یابد ولی انفصالاتی در خطوط پلیسیلیکان ایجاد خواهد شد.

کانیکو [۱۱] با استفاده از شبکه‌ای از سوئیچ‌ها به طراحی سلول و تبدیل رابطهٔ منطقی به مدار ترانزیستوری پرداخته است. در این راه از ترانزیستورهای اضافی برای به دست آوردن سلولی با نفوذ پیوسته استفاده کرده است که به نظر می‌رسد این کار با هدف حداقل کردن مساحت سلول متناقض است.

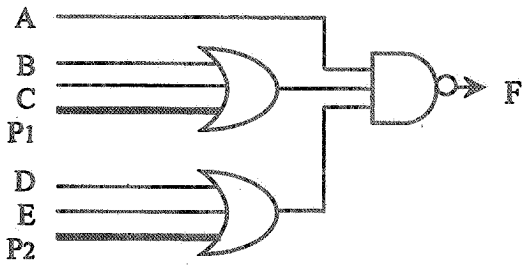
در مرجع [۱۲] سعی شده است با استفاده از شبکه عصبی هاپفلید[11] مسیر اویلر برای یک تابع منطقی به دست آید، ولی مواقعی وجود دارد که این روش قادر نیست مسیر اویلر را پیدا کند. در اینجا الگوریتمی را پیشنهاد می‌کنیم که مسیر اویلر پیوسته را برای یک تابع منطقی پیدا می‌کند. چنانچه مسیر مذکور وجود نداشته باشد مسیرهایی ارائه خواهد شد که حداقل ناپیوستگی را در ناحیه نفوذ ایجاد می‌کنند.

۲ـ الگوریتم اهارا

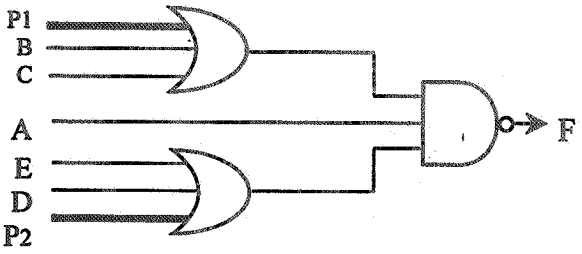
در طراحی جانمایی یک تابع منطقی با تکنولوژی CMOS، خطوط پلیسیلیکان به طور ممتد از هر دو ناحیهٔ نفوذ p و n عبور کرده و دو نوع ترانزیستور را به وجود می‌آورند. بنابراین برای نفوذ پیوسته، جانمایی ترانزیستوری هر دو ناحیه p و n باید یکسان باشد و این بدین معناست که باید هر دو ناحیه مسیر اویلر یکسانی داشته باشند.

اگر تعداد ورودیهای همهٔ گیت‌های OR و AND تابع منطقی، فرد باشد آن تابع منطقی حتما مسیر اویلر پیوسته‌ای خواهد داشت [۲]. بر همین اساس، اهارا در الگوریتم خود ابتدا تعداد ورودهای کلیهٔ گیت‌ها با تعداد زوجی ورودی را با اضافه کردن ورودی کاذب[12]، فرد می‌کند. سپس با این ورویهای کاذب همانند ورودی‌های اصلی رفتار می‌شود و مسیر اویلر به دست می‌آید. مسلما هر ورودی کاذب در مسیر، معادل یک انفصال در ناحیهٔ نفوذ خواهد بود. سپس اهارا الگوریتم دیگری به نام الگوریتم حداقل تداخل[13] برای به دست آوردن گرافی که بتواند حداقل انفصال را بدهد، ارائه کرده است. روش بالا الزاما بهترین جواب را نمی‌دهد [۱]و[۲]. بدیهی است که اضافه شدن هر ورودی کاذب بر تعداد همبندیهای گراف و نیز مسیرها می‌افزاید و پیدا کردن مسیر اویلر را مشکلتر می‌کند. در ضمن برای برخی توابع که به راحتی می‌توان مسیر اویلر پیوسته‌ای را بدون نیاز به اضافه کردن ورودی کاذب پیدا کرد، مسیر اویلر غیرپیوسته تولید می‌کند.

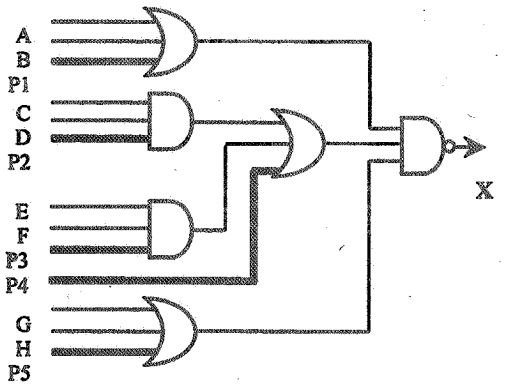
با استفاده از روش اهارا در شکل (۲) مسیر اویلر برای تابع بولی $F=\overline{A(B+C)(D+E)}$ به دست می‌آید. مسیر مذکور به صورت BCADE می‌باشد. این مسیر را روی گرافهای pd[14] و pu[15] می‌توان دنبال کرد. الگوریتم اهارا در یافتن مسیر اویلر در شکل (۲) موفق بوده است. در شکل (۳) دیده می‌شود که اگر چه مسیر اویلر پیوسته‌ای به صورت CDGHEFAB وجود دارد ولی روش اهارا با اضافه کردن ورودیهای کاذب نمی‌تواند مسیر مذکور را پیدا کند. به این ترتیب ابتدا مسیر اویلر به صورت $P_5GHP_1BAP_3EFP_2CDP_4$ است و بعد از حذف ورودیهای کاذب به مسیر غیرپیوستهٔ GHBA-EFDC تبدیل می‌شود. واضح است که علی‌رغم وجود یک مسیر پیوسته روش اوهارا قادر به یافتن مسیر صحیح نیست. در ادامه الگوریتم جدیدی را بر مبنای روش اهارا ارائه می‌دهیم که کاستیهای روش مذکور را مرتفع می‌کند.



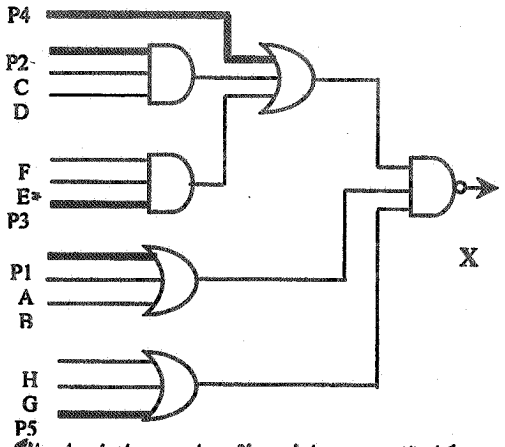
شکل۲ – الف- پیاده سازی تابع $F=\overline{A(B+C)(D+E)}$ با ورودیهای اصلی و کاذب.



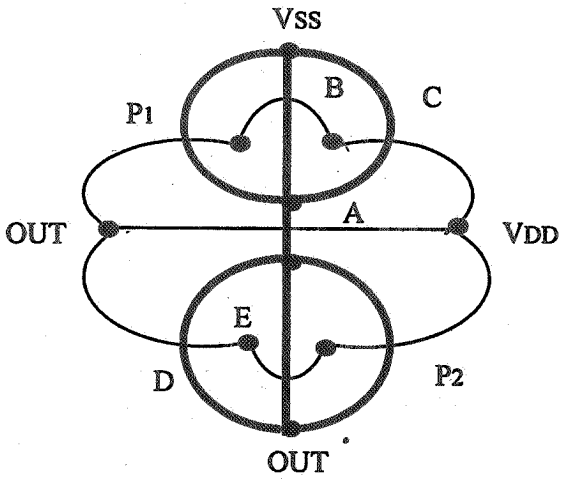
شکل ۲- ب- معادل منطقی تابع بعد از اجرای الگوریتم حداقل تداخل.



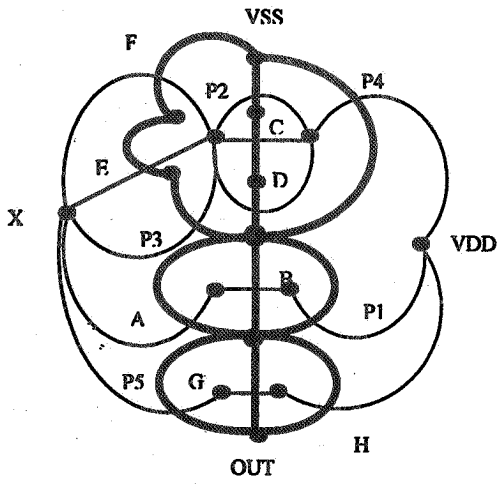
شکل۳- الف- پیاده سازی تابع $X=\overline{(A+B)(CD+EF)(G+H)}$



شکل ۳- ب- معادل منطقی تابع بعد از اجرای الگوریتم حداقل تداخل



شکل ۲-ج- گراف بخشهای PU و PD



شکل ۳-ج- گراف بخشهای PU و PD

۳ـ الگوریتم پیشنهادی برای پیدا کردن مسیر اُویلر

توابع منطقی را از نظر داشتن مسیر اویلر به گروههای زیر تقسیم می‌کنیم:

۱ـ توابعی که تحت هیچ شرایطی مسیراویلر پیوسته ندارند.

۲ـ توابعی که فقط یک مسیر اویلر پیوسته دارند.

۳ـ توابعی که چندین مسیر اویلر پیوسته دارند.

با توجه به اینکه هر گیت OR (AND) تبدیل به اتصالات موازی (سری) و هر ورودی اصلی نیز معادل یک ترانزیستور به همان نام است، ابتدا نمایش مداری تابع منطقی را به دست می‌آوریم. سپس از روی این نمایش مداری، گراف مربوطه با توجه به تعاریف زیر حاصل می‌شود:

الف) هر ضلع گراف معادل کانال یک ترانزیستور است.

ب) نام هر ضلع گراف نام سیگنال ورودی متصل به گیت ترانزیستور است.

ج) اتصالات سری (موازی) ترانزیستورها درنمایش مداری نیز عینا به اتصالات سری (موازی) اضلاع گراف تبدیل می‌شود.

د) نقاط تغذیه (V_{DD}) و زمین (V_{ss}) و خروجی (out) در نمایش مداری و گراف یکسان هستند.

چنانچه مشخص است المانهای سری می‌توانند با یکدیگر جا به جا شوند. بنابراین برای یک تابع منطقی می‌توان نمایش‌های مداری و گراف مختلفی داشت. به این ترتیب یک تابع منطقی الزاما یک نمایش مداری و گرافی منحصر به فرد ندارد. از همین خاصیت برای به دست آوردن مسیر اویلر یک تابع منطقی استفاده می‌کنیم.

براساس آنچه بیان شد در طراحی CMOS ساختار ترانزیستوری و گرافهای دو ناحیه نفوذ p و n دوگان یکدیگرند. هر گاه گراف n (یا p) در دسترس باشد گراف دوگان به راحتی قابل حصول است. حال به چند تعریف می‌پردازیم:

- برخی از توابع منطقی در کلیه گراف‌هایشان گرهی دارند که همیشه تعداد انشعاباتش فرد است. ما این نوع گره را فرد دائمی می‌گوییم. اگر تعداد این گره‌ها بیش از دو باشد آن تابع تحت هیچ شرایطی مسیر اویلر پیوسته نخواهد داشت.
- در صورتی‌که تعداد ورودیهای یک گیت OR ، فرد (زوج) بوده و همگی آنها ورودیهای اصلی سلول باشند، به مجموعه اضلاعی که این گیت در گراف تولید می‌کند، یک گروه فرد (زوج) خالص می‌گوییم. در صورتی‌که تعداد گره‌های زوج خالص سری با هم بیش از دو باشد، آن تابع تحت هیچ شرایطی مسیر اویلر پیوسته نخواهد داشت.
- در صورتی که تعداد گره‌های فرد، دو باشد، باید یکی از این دو گره، نقطه شروع مسیر و دیگری نقطه پایان مسیر باشد.

با استفاده از تعاریف بالا در شروع اجرای الگوریتم، می‌توان توابعی راکه دارای مسیر اویلر پیوسته نیستند تشخیص داد و به این ترتیب سرعت کار الگوریتم را افزایش دهیم .

در شکل (٤) شبه کد الگوریتم پیشنهادی ارائه می‌شود.

جزئیات تابع Find-Eulerian-Path (t) را که برای به دست آوردن مسیر اویلر طراحی شده است، به علت پیچیدگی و اهمیت خاص، در شکل (٥) نشان می‌دهیم.

در تابع بالا از آرایه‌ای به نام path-list(i) استفاده شده است که عناصر آن دارای دو بخش link و node هستند که به ترتیب ضلع و گره بعدی که باید پیموده شود را در خود ذخیره می‌کند. همچنین s گره شروع و current گره جاری و next گره بعدی است.

اگر چه یافتن یک مسیر اویلر برای یک تابع منطقی کافی است، اما تمام مسیرهای اویلرِ ممکن در اینجا پیدا می‌شوند. با این کار می‌توان مسیری را انتخاب کرد که کمترین تعداد اتصالات[16] به خروجی‌ها را داراست. شکل (٦) سه مثال از اجرای گام به گام الگوریتم را نشان می‌دهد.

```
Standard Cell Generation Algorithm:
Begin
   read boolean_equation( );
   n:= count_permanent_odd_nodes;
   m:=count_pure_even_nodes;
   if ( m<= 2 and n<= 2) then
   begin
      while ( there exists new topology) do
      begin
         t:=find-topology ( );
         insert_in_queue(t);
      end;
      while ( queue_not_empty) do
      begin
         t:=take_from_queue( );
            repeat
               x = Find-Eulerian-Path (t );
            until (x is also true for dual graph);
               print path(x);
      end;
   end;
end;
```

شکل ٤- شبه کد الگوریتم پیشنهادی

٤ ـ نتایج شبیه‌سازی

برای اطمینان از درستی الگوریتم، نرم‌افزاری به زبان C^{++} نوشته و مثالهای متعددی توسط این نرم‌افزار اجرا شد. شکل (۷) یک نمونه از نتایج به دست آمده را نشان می‌دهد.

چنانچه تابع بولی X به صورت $\overline{X} = A(B+C+D)(E+F)+GH+IJ$ تعریف شده باشد می‌توان آن را توسط سلولی که دارای ۱۰ ورودی A تا J است طراحی کرد. واضح است که ترتیب قرار گرفتن ورودیها نوع همبندی را مشخص می‌کند. با وارد کردن رابطهٔ بولی به عنوان ورودی نرم‌افزار چهار مسیر اویلر به طریقی که در جداول (۱) تا (٤) دیده می‌شود به دست می‌آید.

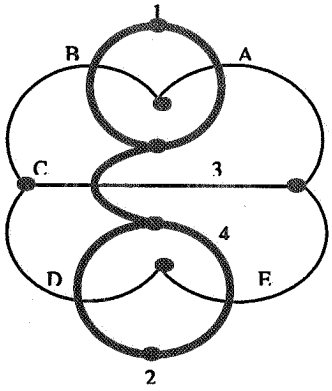
شکل (۷) پیاده‌سازی تابع منطقی را نشان می‌دهد. جدول (۱) خروجی نرم‌افزار است که توسط این اطلاعات جانمایی شکل (۷) ساخته شده است. همان گونه که مشهود است نواحی نفوذ دارای پیوستگی هستند.

سطر اول جدول (۱) نام ورودیهای سلول را نشان می‌دهد که ترتیب آنها همان مسیر اویلر است. در این مثال چهار مسیر اویلر به دست می‌آید. اولین مسیر، GHFEIJADCB است که در جدول ذکر شده است. جداول (۲) تا (٤) به ترتیب مسیرهای ABCDGHFEIJ ، GHFEIJBCDA و DCBAGHFEIJ را بیان می‌کنند. حال مختصرا نحوهٔ تعبیر جدول (۱) را برای طراحی جانمایی توضیح می‌دهیم.

ستون یازدهم این جدول که در این مثال u در بالای آن دیده می‌شود بیانگر این است که دو ردیف اول جدول، گرههای pu جانمایی هستند و دو ردیف بعد، گره‌های جانمایی pd اند. چنانچه در بالای این ستون حرف d آمده بود دو ردیف اول

```
Find_Eulerian_Path(t)
Begin
   n:=count_odd_nodes(t);
   If ( n = 2) then
   Begin
      while ( n>0 ) do
      begin
         s := an_unmarked_node(t);
         mark_odd_node(s);
         n := n-1
         i := 1
         current := s;
         repeat
            path_list(i).node := current;
            link_to_next := an_unmarked_link(t, current);
            mark_link(link_to_next);
            current :=  next_node_of_current(t,current,link_to_next);
            while (current is a dead_end) and (path is not complete) and (current ≠ s) do
               begin
                  unmark_all_links_connected_to(current, t);
                  current := path_list(i).node;
                  if ( there exists unmarked link connected to current) then
                     begin
                        link_to_next := an_unmarked_link(t, current);
                        mark_link (link_to_next);
                        current := next_node_of_current(t, current, link_to_next);
                     end;
                  else
                     begin
                        i := i-1
                     end;
               end;
            i :=i+1;
            path_list(i).link := link_to_next:
         until (current = s) or (path is complete);
         if (path is complete) then
            print (path_list);
            n := -1;
         end;
      end;
   end;
end;
```

شکل ۵- جزئیات تابع ***Find-Eulearian-Path (t)***



$$F = \overline{(A+B)C(D+E)}$$

۲= تعداد گرههای فرد

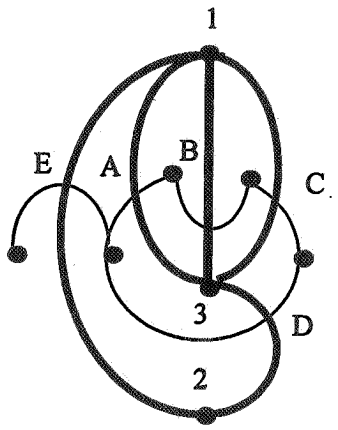
۳= گره شروع

مسیر ناموفق (به علت توقف در ٤) : ٤ E D ٤ C ٣

مسیر ناموفق (به علت عدم ادامه مسیر):٤ C ٣

مسیر اویلر : ٤ D E ٤ C ٣ AB ٣

(این مسیر برای گراف دوگان هم صادق است.)



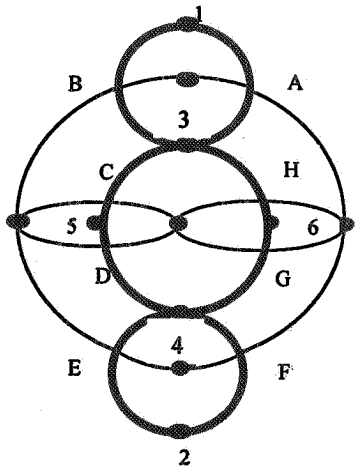
$$F = \overline{(A+B+C)D+E}$$

۰ = تعداد گرههای فرد

۱= گره شروع

مسیر اویلر : ١ E ٢ D ٣ C AB ١

(این مسیر برای گراف دوگان هم صادق است.)



$$F = \overline{(A+B)(HG+CD)(E+F)}$$

۰ = تعداد گرههای فرد

۱= گره شروع

مسیر ناموفق (به علت توقف در ۱) : ١AB١

تغییر گره شروع:

مسیر ناموفق (توقف در ۳) : ٣ AB ٣ C٥ D ٤ G٦H٣

مسیر ناموفق (به علت عدم انتخاب جدید):

٣ AB ٣ C ٥ D٤G٣

مسیر اویلر: ٣ AB ٣ C٥ D ٤ G٦H٣

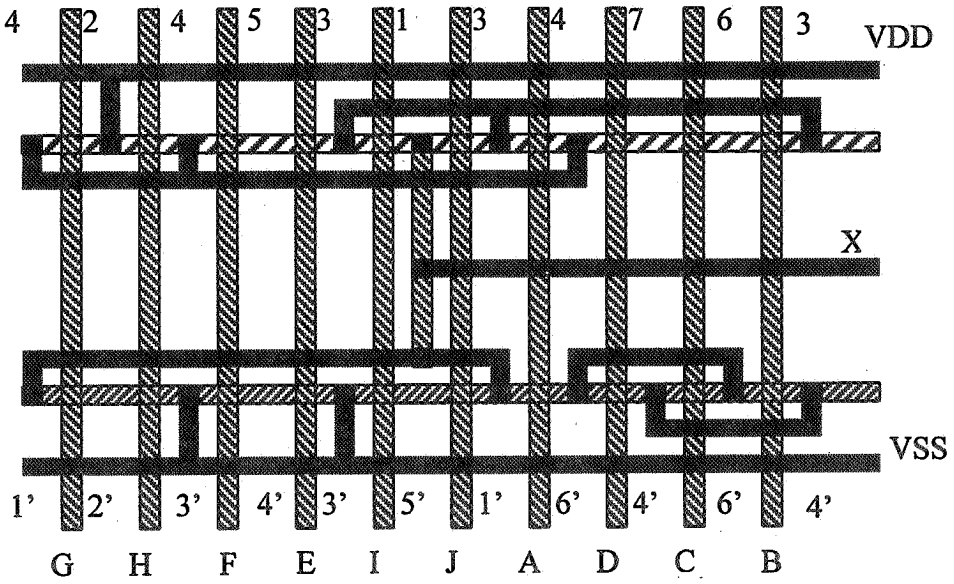
(این مسیر برای گراف دوگان هم صادق است)

شکل ٦- اجرای سه مثال با استفاده از الگوریتم پیشنهادی

[DOR: 20.1001.1.2251600.1381.21.2.2.6]

جدول ۱- اعلام مسیر اویلر توسط خروجی نرم‌افزار برای تابع $\overline{X} = A(B+C+D)(E+F)+GH+IJ$

	اتصال به تغذیه و خروجی	ورودیهای مدار (مسیر اویلر)									
	U	B	C	D	A	J	I	E	F	H	G
اتصالات PULLUP	1	3	6	7	3	1	1	3	5	4	4
	2	6	7	4	4	3	3	5	4	2	2
اتصالات PULLDOWN	3	4'	4'	4'	1'	5'	5'	3'	3'	2'	1'
	1	6'	6'	6'	6'	1'	1'	4'	2'	3'	2'



شکل ۷- پیاده‌سازی تابع منطقی $\overline{X} = A(B+C+D)(E+F)+GH+IJ$

متعلق به pd و دو ردیف بعد گرههای pu را مشخص می‌کـرد. به این ترتیب گره‌های دو طرف ورودی G در pu گره‌هـای 4 و 4 است و در pd گرههای '1 و '2 است. به این ترتیب سورس و درین در ترانزیستور PMOS و NMOS کـه گیت آنها توسط سیگنال G تحریک می‌شود شماره گذاری شدند. البته باید توجه کرد که شماره گذاری گرهها در pd و pu مستقل از هم اسـت به همین دلیل شمارهٔ گرههای ناحیهٔ نفوذ n یعنی pd را با پریـم (') تفکیک کرده‌ایم

چنانچه مسیر اویلر پیوسته‌ای وجود داشته باشد باید بین دو ترانزیستور که در جدول، مجاور هم هستند چه در pd و چه در pu، حداقل یک گره مشترک وجود داشته باشد. این بدان معنـی است که ناحیهٔ نفوذ بین دو ترانزیسـتور مجـاور پیوسـته اسـت. اگر گره مشترکی بین دو ورودی که در جدول در مجاورت هـم نیستند وجود داشته باشد باید با اتصال فلز آن دو ترانزیسـتور را به هم متصل کرد. این وضعیت برای درین ترانزیسـتورهای G و J در ناحیهٔ pd دیده می‌شود که در گره '1 مشترک هستند. حال اگر دو ورودی مجاور دارای دو گره مشترک باشند به این معنـی است کـه دو ترانزیسـتور مـوازی هسـتند. ایـن وضعیـت بـرای

جدول ۲- همبندی شمارهٔ ۲ برای تابع $\overline{X} = A(B+C+D)(E+F)+GH+IJ$

	اتصال به تغذیه و خروجی	ورودیهای مدار (مسیر اویلر)									
	u	J	I	E	F	H	G	D	C	B	A
اتصالات PULLUP	1	1	1	3	5	4	4	7	6	3	3
	2	3	3	5	4	2	2	4	7	6	4
اتصالات PULLDOWN	5	6'	5'	5'	5'	4'	3'	2'	2'	2'	1'
	3	3'	6'	1'	1'	5'	4'	3'	3'	3'	2'

جدول ۳- همبندی شمارهٔ ۳ برای تابع $\overline{X} = A(B+C+D)(E+F)+GH+IJ$

	اتصال به تغذیه و خروجی	ورودیهای مدار(مسیر اویلر)									
	U	A	D	C	B	J	I	E	F	H	G
اتصالات PULLUP	1	3	7	6	3	1	1	3	5	4	4
	2	4	4	7	6	3	3	5	4	2	2
اتصالات PULLDOWN	3	4'	1'	1'	1'	5'	3'	3'	3'	2'	1'
	1	6'	6'	6'	6'	1'	5'	4'	4'	3'	2'

جدول ٤- همبندی شمارهٔ ٤ برای تابع $\overline{X} = A(B+C+D)(E+F)+GH+IJ$

	اتصال به تغذیه و خروجی	ورودیهای مدار(مسیر اویلر)									
	U	J	I	E	F	H	G	A	B	C	D
اتصالات PULLUP	1	1	1	3	5	4	4	3	3	6	7
	2	3	3	5	4	2	2	4	4	7	4
اتصالات PULLDOWN	5	6'	5'	5'	5'	4'	3'	2'	2'	1'	1'
	3	3'	6'	1'	1'	5'	4'	3'	3'	2'	2'

ترانزیستورهای I و J در ناحیهٔ pu وجود دارد کـه هـر دو دارای گره‌های 1 و 3 هستند.

جدول (۱) محل اتصال ناحیهٔ نفوذ به تغذیه و خروجـی را نیز نمایش می‌دهد. ستون یازدهم جدول این مثال در زیر حرف u، برای ناحیهٔ pu ، اتصال به خط تغذیـه را گره 2 و اتصـال بـه خروجی را گره 1 معرفی کرده است. این اتصالات در ناحیهٔ pd گرههای 3 و 1 هستند. واضح است که این گرهها قـابل جـا بـه جایی‌اند. با جا به جا کـردن گره تغذیـه و خروجـی تغییری در مسیر اویلر داده نمی‌شود ولی تاثیرات زیادی روی سرعت مـدار می‌تواند ایجاد شود. نرم افزار گره‌ای را به عنوان خروجی اعـلام می‌کند که تعداد اتصالات کمتری به ناحیهٔ نفوذ دارد.

۵-نتیجه گیری

همان‌گونه که مشاهده می شـود، در ایـن مقالـه، کاستیهای

روش اُهارا مشخص شد. عمده‌ترین این کاستی‌ها قادر نبودن الگوریتم در یافتن مسیر در مواردی است که مسیر وجود دارد. الگوریتمهای زیادی براساس روش اُهارا در مقالات ارائه شده که در هیچ کدام این کاستیها رفع نشده است. در این مقاله بعد از رفع اشکالات ، الگوریتمی ارائه شد که برای هر تابع منطقی ترکیبی چنانچه مسیر اویلری وجود داشته باشد، مسیر پیوسته‌ای را ارائه می‌دهد. در صورت عدم وجود مسیر مذکور، بهترین جانمایی محتمل را با ناپیوستگی در ناحیهٔ نفوذ، ایجاد می‌کند. برخلاف روشی که در [۱۲] ارائه شده است و فقط هفتاد درصد مواقع به جواب می‌رسد، الگوریتم حاضر برای تمام سیصد تابع آزمایشی جواب صحیح را تولید کرد. توابع آزمایشی همگی به صورت تصادفی با پیچیدگیهای مختلف تولید شده‌اند. حداکثر تعداد متغیرهای هر تابع ۲۸ (تعداد حروف بزرگ لاتین) بود که این محدودیت صرفاً در پیاده سازی نرم‌افزار اعمال شد. از جانب الگوریتم هیچ محدودیتی از نظر تعداد متغیرها وجود ندارد. از خروجی نرم‌افزار می‌توان اتصالات فلزی مورد نظر در جانمایی را نیز استخراج کرد. این اطلاعات برای رسم نقشه ماسکها به صورت خودکار قابل استفاده است.

از الگوریتم حاضر برای یافتن مسیر اویلر روی هر گراف بدون جهت نیز می‌توان استفاده کرد و کارایی آن محدود به جانمایی VLSI نیست.

الگوریتم پیاده‌سازی شده تمام مسیرهای اویلرِ ممکن را تولید می‌کند به این ترتیب غیر از قابلیت انتخاب مسیری با کمترین تعداد اتصالات خروجی، مسیری را می‌توان انتخاب کرد که با سلولهای دیگر تراشه از نظر ترتیب ورودیها دارای سازگاری بیشتری باشد.

واژه نامه

1. layout generation
2. standard cell
3. logic synthesis
4. technology mapping
5. global and detailed routing
6. compaction
7. basic grid size
8. noise immunity
9. Eulerian path
10. dual independece
11. Hopfeild
12. pseduo input
13. minimal interlace algorithm
14. Pull down
15. pull up
16. contacts

مراجع

1. Chi Yi Hwang, Yung Ching Hsieh, Youn-Long Lin, and Yu-Chin Hsu , "An Efficient Layout Style for Two-Metal CMOS Leaf Cells and its Automatic Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* , Vol. 12, No. 3 , pp. 410-424, March 1993.
2. Wolf, W. *Modern VLSI Design: A System Approach*, Prentice-Hall, 1994.
3. Uehara, T. and Van, W. M. Cleemput, "Optimal Layout of Functional Arrays," *IEEE Transaction On Computers*, Vol. C-30 No. 5, pp. 305-314, May 1981.
4. Maziasz, R.L. and Hayes, J. P. "Layout Optimization of Static CMOS Functional Cells," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol. 9, No. 7, PP. 708-719, July, 1990.
5. Wimer, S. and I. Koren "Analysis of Strategies for Constructive General Block Placement" *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 7, No. 3 , pp. 371 –377, March 1988.
6. Bar-Yehuda, R., Feldman, J. A., Pinter, R. Y. and Wimer, S. "Depth First Search and Dynamic Programming Algorithm for Efficient CMOS Cell Generation," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 8, No. 7, PP. 737-743, July 1989.
7. Tong Li; Ramaswamy, and S., Rosenbaum, E., Sung-Mo Kang, "Circuit-Level Simulation and Layout Optimization for Deep Submicron EOS/ESD Output Protection Device," *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 159-162, 1997.
8. Hwang, C.Y., and Hsieh, Y.C., and Hsu, Y.C., "A Fast Transistor-Chaining Algorithm for CMOS Cell Layout," *IEEE Trans. on Computer -Aided Design*

of Integrated Circuits and Systems, Vol. 9, No. 7, PP. 781-786, July 1990.

9. Shibatani, S., Sadakane, T., Nakao, H., Terai, M., and Okazaki, K., "A CMOS Cell Generation System for Two-Dimensional Transistor Placement," *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp 325-328, 1998.

10. Carlson, B.S., Chen, C.Y.R., and Singh, U., "Optimal Cell Generation for Dual Independent Layout," *Transaction on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 10, No. 6, pp. 770-782, June 1991.

11. Kaneko, M., and Tian, J., "Concurrent Cell Generation and Mapping for CMOS Logic Circuits," *Proceedings of the Asia and South Pacific Design Automation* Conf., PP. 247, 252, 1997.

۱۲. سماوی، ش. و ترکیان، الف. "ارائه یک تابع انـرژی جدیـد بـرای یـافتن مسـیر اویلـر در طراحـی مدارهـای منطقـی CMOS"، مجموعـه مقـالات هشـتمین کنفرانـس مهندسـی برق،جلد اول، ص ۱۹۰-۱۹۷، ۱۳۷۹ .

[DOR: 20.1001.1.2251600.1381.21.2.2.6]