بسمه تعالی

عنوان طرح: عیب‌یابی و تصحیح اتوماتیک ایرادهای طراحی در مدارهای RTL و سطح گیت

مجری اصلی: بیژن علیزاده

آدرس محل کار: تهران، خیابان کارگر شمالی، دانشکدگان فنی دانشگاه تهران، دانشکده مهندسی برق و کامپیوتر، ساختمان جدید، اتاق ۳۲۰

چکیده طرح: پیچیدگی روزافزون مدارهای دیجیتال موجب شد که فرآیند درستی‌سنجی[1] و عیب‌یابی[2] مدت زمان زیادی از فاز طراحی را بخود اختصاص دهند. بر اساس آمار منتشر شده، حدود ۵۵ الی ۶۰ درصد زمان طراحان صرف درستی‌سنجی و عیب‌یابی طرح‌شان می‌شود. مقیاس‌پذیری کم روش‌های درستی‌سنجی و فشار برای آماده‌سازی هرچه سریعتر محصول نهایی، درستی‌سنجی و عیب‌یابی را به مساله‌ای ضروری و چالشی تبدیل کرده است. در صورت نبودن روش سیستماتیکی برای حل این مساله، هزینه‌ی طراحی و ساخت محصول نهایی بصورت چشمگیری افزایش می‌یابد. دلیل این امر تکرار فازهای طراحی و ساخت سیستم از ابتدا می‌باشد. از این رو، ارایه‌ی راهی سیستماتیک برای درستی‌سنجی و تصحیح ایرادهای طراحی که منجر به کاهش زمان طراحی و ساخت شود از جمله نیازهای طراحان سخت‌افزار می‌باشد، طوری که مجبور به تکرار فازهای طراحی و ساخت نشوند. هدف از این کار پژوهشی، ارایه روشی کارا و اتوماتیک برای عیب‌یابی و تصحیح ایرادهای طراحی در مدارهای RTL و سطح گیت است.

[1] Verification
[2] Debug

کلید واژه ها:

درستی‌سنجی: فرآیند تصدیق درستی عملکرد یک مدار دیجیتال توصیف شده در سطوح مختلف تجرد (مثلا در سطح انتقال ثبات (RTL)[3]). دو روش کلی درستی‌سنجی عبارتند از: ۱) روش‌های مبتنی بر شبیه‌سازی[4]، ۲) روش-های صوری[5]

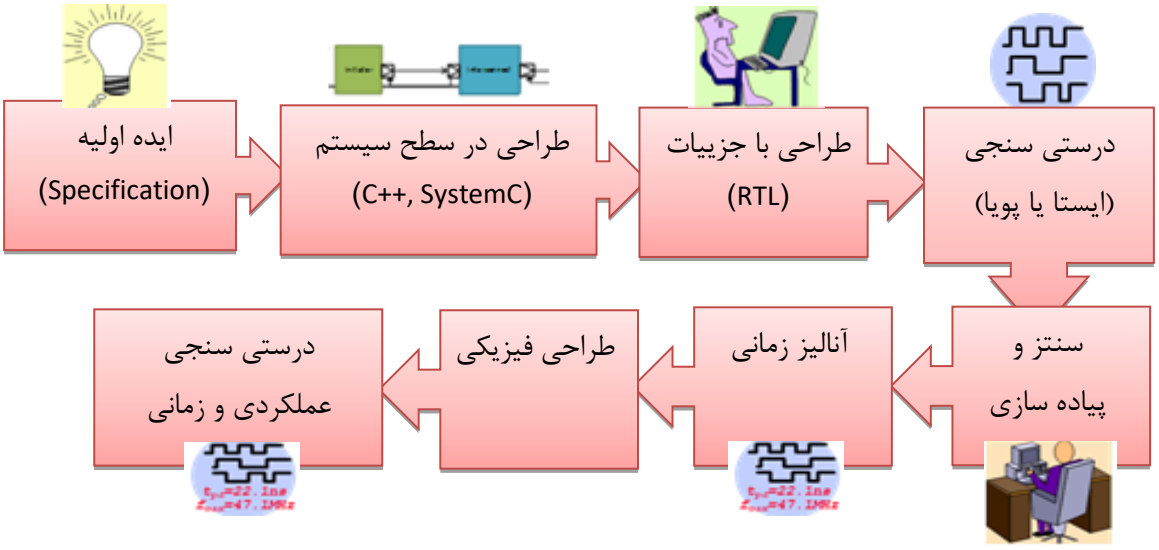
عیب‌یابی: فرآیند یافتن محل ایرادهای طراحی و ارایه اطلاعات لازم به طراح جهت رفع ایرادها.

تصحیح: فرآیند رفع ایرادهای طراحی که منجر به طراحی بدون ایراد خواهد شد.

[3] Register Transfer level (RTL)
[4] Simulation-based Verification
[5] Formal Verification

۱- مقدمه

روند طراحی و ساخت مدارهای دیجیتال بر اساس استفاده از ابزار طراحی به کمک کامپیوتر (CAD)[۶] می‌باشد که در شکل ۱ نشان داده شده است. در چنین روندی، ابتدا طراحان یک توصیف در سطح سیستم، عمدتا الگوریتم موردنظر، از آنچه که می‌بایست ساخته شود، ارایه می‌کنند. سپس با استفاده از ابزار سنتز سطح بالا این توصیف را به یک مدل در سطح انتقال ثبات (RTL) می‌رسانند. همانطور که در شکل ۱ نشان داده شده است، پس از رسیدن به یک توصیف در سطح انتقال ثبات، روند سنتی طراحی مدارهای دیجیتال، یعنی انجام مراحل سنتز سطح انتقال ثبات به سطح گیت و سپس طراحی فیزیکی[۷] بهمراه درستی‌سنجی (در این ارتباط از ابزار شبیه‌سازی و یا ابزار درستی‌سنجی صوری استفاده می‌شود) طی خواهد شد تا در نهایت سیستم مورد نظر ساخته شود. پس از انجام سنتز در سطوح مختلف، مهم‌ترین سوال این خواهد بود که آیا مدار RTL، مدار سطح گیت و در نهایت Layout تولید شده به لحاظ عملکردی با کد سطح بالا یکسان است؟ فرآیند درستی‌سنجی پاسخ‌گوی این سوال است.



شکل ۱. روند طراحی کامل مدارهای دیجیتال

[6] Computer Aided Design
[7] Physical Design

درستی‌سنجی طرح به معنای حصول اطمینان از صحت طراحی انجام شده است. عیب‌یابی، تشخیص[8] و تصحیح[9] خطای طراحی به معنای بررسی یک طرح، پیدا کردن خطاهای موجود در طراحی و رفع آن‌ها می‌باشد [۱]. در آغاز طراحی یک سیستم دیجیتال وظیفه درستی‌سنجی سیستم پیاده‌سازی شده بر عهده خود طراح بود. اما به‌تدریج با بزرگ‌شدن و پیچیده‌شدن طراحی‌های سخت‌افزاری طبق قانون مور [۲] و پیشرفت‌های انجام شده در زمینه روش‌های طراحی، از یک طرف تعداد ویژگی‌های خاص هر طرح روزبه‌روز افزایش یافت و از طرف دیگر نقاط گوشه و حالات کمتر قابل دسترس رو به فزونی نهاد. به همین جهت احتیاج به مهندسان درستی‌سنج[10] و روش‌های درستی‌سنجی کارا بیش از پیش آشکار شد. به‌علاوه به‌دلیل این‌که به‌طور معمول تاکید بیش‌تری بر روی روش‌های طراحی در مقابل روش‌های درستی‌سنجی و عیب‌یابی بوده است، ابزارهای زیادی دراین زمینه به‌وجود نیامده‌اند.

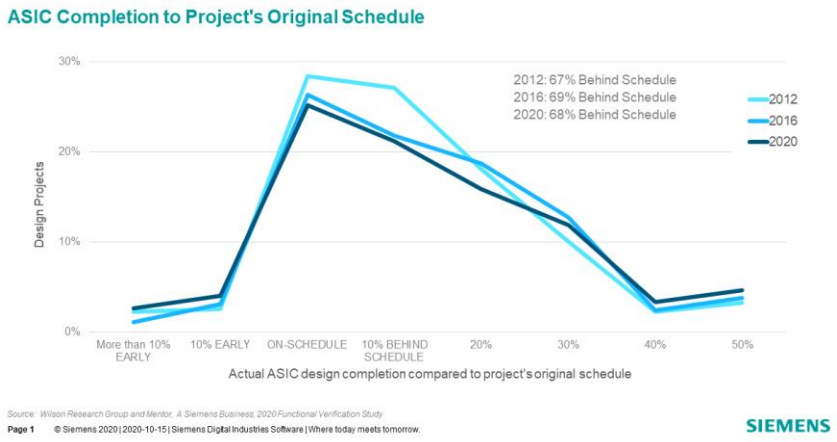
امروزه همراه با افزایش اندازه و پیچیدگی مدارهای مجتمع[11] و سیستم‌های روی تراشه[12]، درستی‌سنجی طراحی‌ها به یک عامل تعیین‌کننده در روند طراحی تبدیل شده است. مقایسه‌ی تکمیل پروژه‌ی ASIC/IC با زمان‌بندی اصلی، معیاری است که اغلب برای ارزیابی کارایی[13] بررسی می‌شود. همان‌طور که در شکل ۲ مشخص است، برای مثال در سال ۲۰۲۰، ۶۸٪ پروژه‌های ASIC/IC از زمان‌بندی عقب مانده‌اند [۳] که یکی از دلایل مهم آن، تاخیر در درستی‌سنجی است.

شکل ۳، نسبت مدت زمان درستی‌سنجی به مدت زمان کل پروژه‌ی ASIC/IC را بر حسب درصد نمایش داده است. باتوجه به شکل برای منحنی‌ها دو کران وجود دارد. اولین کران مربوط به پروژه‌هایی است که مدت زمان کم‌تری را صرف درستی‌سنجی می‌کنند. این پروژه‌ها معمولا روی مدارهایی کار می‌کنند که از یکپارچه‌سازی IPهایی تشکیل شده‌اند که درستی‌سنجی آنها از پیش انجام‌شده و مشخص است، بنابراین مدت زمان لازم برای درستی‌سنجی مدار یکپارچه کاهش خواهد یافت. در مقابل کران دوم مربوط به پروژه‌هایی است که مدت زمان

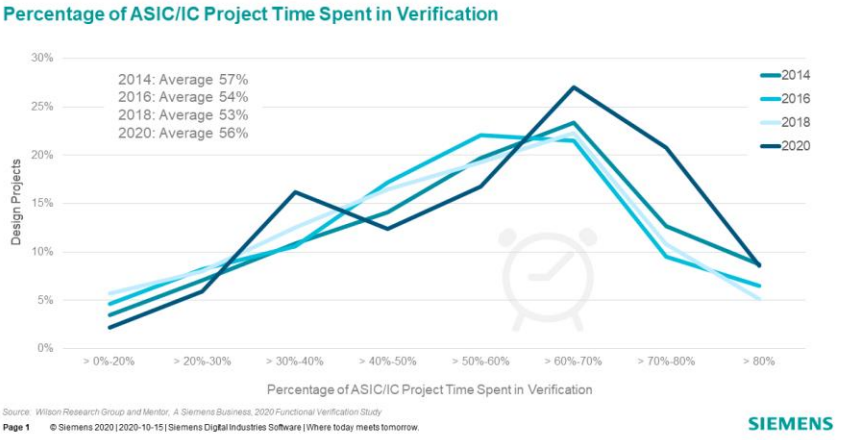
---

[8] Diagnosis
[9] Correction or Repair
[10] Verification Engineers
[11] Integrated Circuits (IC)
[12] System on Chip (SoC)
[13] Efficiency

بیش‌تری را صرف درستی‌سنجی می‌کنند. این پروژه‌ها معمولا روی مدارهایی کار می‌کنند که از یکپارچه‌سازی IPهایی تشکیل شده‌اند که جدید هستند و درستی‌سنجی آنها از پیش انجام نشده است. در این میان، یافتن محل ایراد اهمیتی دوچندان پیدا کرده است، به‌طوری که عیب‌یابی یک مدار امروزه بیش از ۴۱ درصد زمان درستی‌سنجی را به خود اختصاص می‌دهد [۳]. شکل ۴ درصدهای اختصاص‌یافته به بخش‌های مختلف درستی سنجی را نشان می‌دهد.



شکل ۲. تاخیر در زمان‌بندی پروژه‌های ASIC/IC [۳]

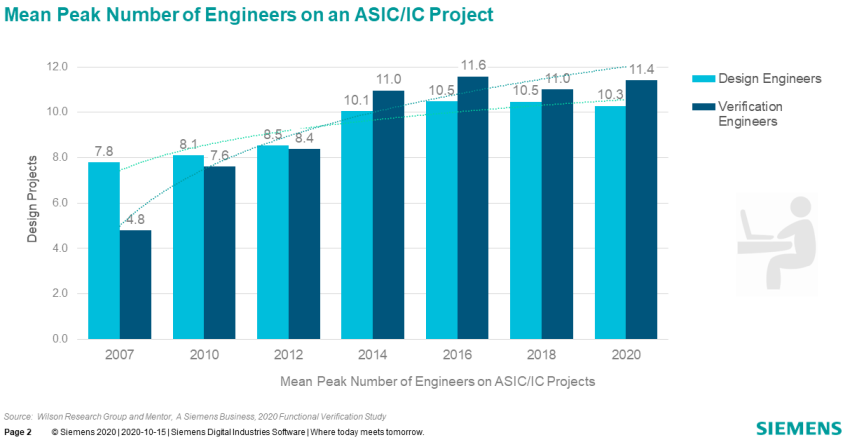


شکل ۳. نسبت مدت زمان درستی‌سنجی به مدت زمان کل پروژه ASIC/IC [۳]

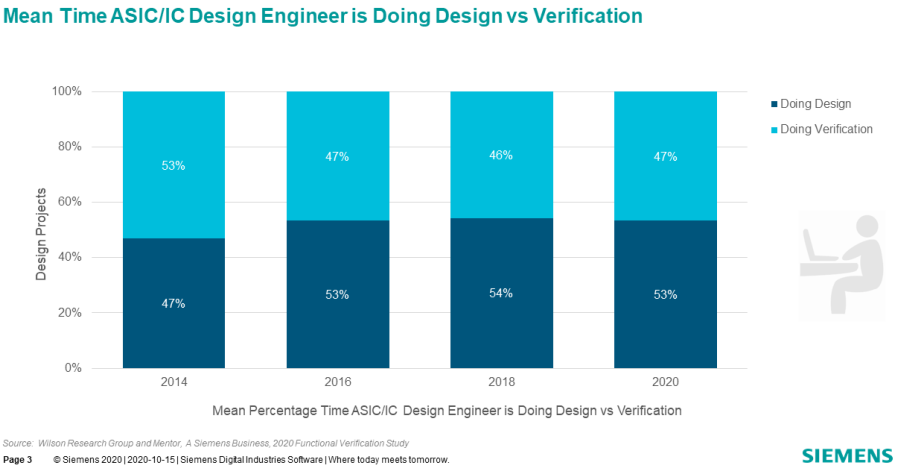
شکل ۴. درصد زمان تخصیص‌یافته به درستی‌سنجی و عیب‌یابی در پروژه‌های ASIC/IC [۳]

امروزه مدیریت کردن هزینه و تعداد کارمندان، یکی از بزرگترین چالش‌های پیشروی شرکت‌ها محسوب می‌شود. بنابراین تشخیص روش‌های طراحی و درستی‌سنجی ASIC/IC که توانایی افزایش بهره‌وری را دارند، از اهمیت ویژه‌ای برخوردار است. نیاز به مدیریت بهره‌وری در شکل ۵ با استفاده از پارامتر متوسط ماکزیمم [۱۴] تعداد مهندسان ASIC/IC به تصویر کشیده شده است. همان‌طور که در این شکل مشخص است، نیاز به مهندسان طراحیASIC/IC بین سال‌های ۲۰۰۷ تا ۲۰۲۰، تقریبا ۳٪ افزایش یافته است. درحالی که نیاز به مهندسان درستی‌سنجی در مدت زمان مشابه ۶,۸٪ افزایش داشته است. امروزه در تمام بخش‌های بازار به طور متوسط به ازای هر مهندس طراحی، یک مهندس درستی‌سنجی مشغول به کار است. اگرچه در برخی از بخش‌ها مثل پردازنده‌ها، این نسبت ۱ به ۵ خواهد بود. یعنی به ازای هر مهندس طراحی، پنج مهندس درستی‌سنجی مشغول هستند. توجه شود که فرآیند درستی‌سنجی، تنها توسط مهندسان درستی‌سنجی انجام نمی‌شود، بلکه مهندسان طراحی نیز بخش قابل توجهی از زمان خود را صرف درستی‌سنجی می‌کنند. باتوجه به شکل ۶، برای مثال در سال ۲۰۲۰، مهندسان طراحی به ترتیب ۵۳٪ و ۴۷٪ از زمان خود را صرف طراحی و درستی‌سنجی کرده‌اند. اگرچه به نظر می‌رسد در سال ۲۰۱۴، مهندسان طراحی بخش بیشتری از زمان خود را (۵۳٪) برای درستی‌سنجی صرف کرده‌اند.

[14] Mean Peak



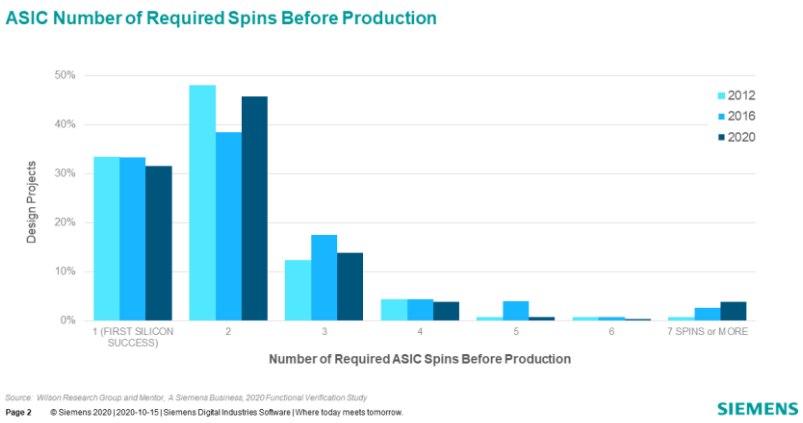
شکل ۵. تعداد مهندسین درستی‌سنجی در مقایسه با تعداد مهندسین طراح در پروژه‌های ASIC/IC [۳]



شکل ۶. درصد زمان اختصاص داده شده توسط طراحان به درستی‌سنجی در پروژه‌های ASIC/IC [۳]

تعداد spinها قبل از تولید، یکی دیگر از معیارهایی است که اغلب برای ارزیابی کارایی بررسی می‌شود. شکل ۷، تعداد spinهای قبل از تولید مورد نیاز را بین سال‌های ۲۰۱۲ تا ۲۰۲۰ نشان می‌دهد. باتوجه به شکل، اگرچه در این مدت زمان، مدارها پیچیده‌تر شده‌اند، تعداد spinهای قبل از تولید مورد نیاز، افزایش نیافته است. با این وجود امروزه، تقریبا فقط ۳۲٪ از پروژه‌ها دفعه‌ی اول شانس ساخت موفق [۱۵] را خواهند داشت.

[15] The first silicon success

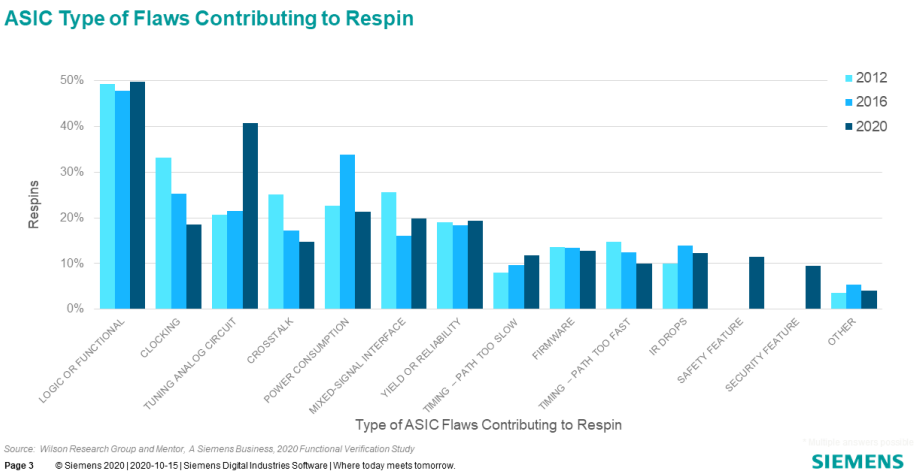


شکل ۷. تعداد Spinها قبل از تولید محصول نهایی در پروژه‌های ASIC/IC [۳]

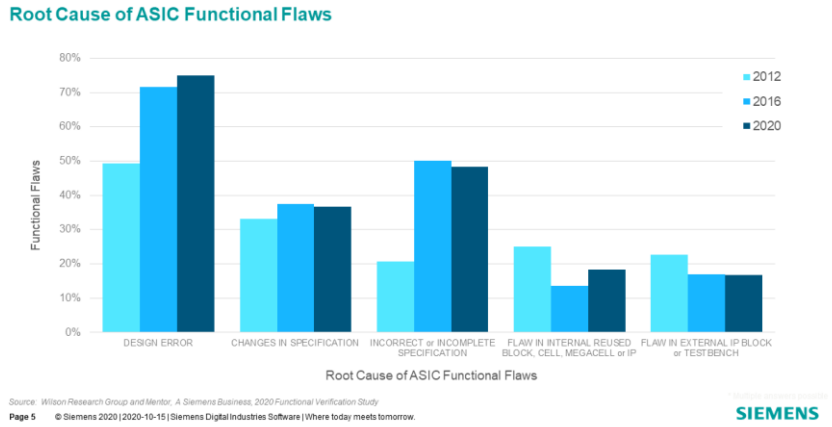
انواع مختلف ایرادهای طراحی که منجر به ساخت دوباره‌ی ASIC/IC [۱۶] می‌شوند در شکل ۸ به تصویر کشیده شده است. باتوجه به این شکل، ایرادهای Logic or Functional سهم بیش‌تری در ایجاد عیب دارند. درسال ۲۰۲۰ برای اولین بار سهم ایرادهای مربوط به Safety و Security بررسی شدند که به ترتیب برابر با ۱۱٪ و ۱۰٪ می‌باشند. اما میزان سهم این دو دسته برای سال‌های قبل در دسترس نیست. بدیهی است که چون چندین نقص[۱۷]می‌تواند به پوشانده شدن عیب منجر شود، جمع درصدهای دسته‌های مختلف بیش‌تر از ۱۰۰٪ خواهد بود. باتوجه به شکل ۸، سهم دسته‌ی سوم، یعنی میزان‌سازی مدار آنالوگ[۱۸] در سال ۲۰۲۰، برابر با ۴۱٪ بوده و افزایش چشمگیری داشته است. ممکن است در نگاه اول اینطور به نظر برسد که این افزایش به دلیل کاهش ابعاد تکنولوژی است، اما بررسی‌ها نشان می‌دهد که بیش‌تر این نقص‌ها مربوط به تکنولوژی‌های ۱۵۰ نانومتر و بزرگ‌تر هستند تا تکنولوژی ۷ نانومتر. دسته‌های مختلف ریشه‌ی ایرادهای Logic or Functional در شکل ۹ نشان داده شده است. باتوجه به این شکل، نقص‌های طراحی[۱۹] بیش‌ترین سهم را در بین دسته‌های مختلف داشته و تاثیر آن‌ها بین سال‌های ۲۰۱۲ تا ۲۰۲۰ افزایش یافته است. به علاوه نقص‌های مربوط به دسته‌های دوم و سوم، یعنی

[16] Re-spin
[17] Multiple flaws
[18] Tuning analog circuit
[19] Design Error

تغییر، نادرستی یا کامل نبودن توصیف مساله رایج هستند که اغلب توسط مهندسان درستی‌سنجی و مدیران پروژه گزارش می‌شوند.



شکل ۸. انواع مختلف ایرادهای طراحی منجر به ساخت دوباره‌ی ASIC/IC [۳]



شکل ۹. دسته‌های مختلف ریشه‌ی ایرادهای Logic or Functional [۳]

با این مقدمه، مهم‌ترین دلیل ظاهر شدن ایرادها در مرحله پیش از ساخت مدارهای مجتمع را می‌توان به‌صورت کلی به عوامل زیر تقسیم کرد.

- **پیچیدگی روزافزون طراحی‌های امروزی نسبت به گذشته**: امروزه با گسترش صنعت نیمه‌هادی و تقاضای بسیار زیاد مصرف‌کنندگان برای دستگاه‌هایی با قابلیت پیچیده و عمل‌کرد برتر، پیچیدگی

طرح‌ها روز به روز افزایش یافته است. در ۲۵ سال گذشته این صنعت، دستگاه‌هایی با ویژگی‌های بیش‌تر، عملکرد بهتر، اندازه فیزیکی کوچک‌تر و مصرف انرژی کم‌تر تولید کرده است. طرح‌های پیچیده احتمال بروزخطا در مرحله طراحی و نادیده گرفته‌شدن در مرحله درستی‌سنجی را دشوار می‌نماید [۱].

- **زمان محدود عرضه به بازار**: به‌دلیل فشار برای آماده سازی سریع‌تر یک محصول و عرضه آن به بازار و رقابت با سایر شرکت‌های سازنده، ممکن است فاز درستی‌سنجی به طور کامل انجام نشده باشد [۴].
- **سرعت پایین روش‌های شبیه‌سازی جهت پوشش کامل مدار**: در مراحل اولیه‌ی طراحی به کمک کامپیوتر برای سنتز و درستی‌سنجی، شبیه‌سازی یکی از مهم‌ترین روش‌های درستی‌سنجی طراحی مدار بوده است. شبیه‌سازی مبتنی بر محاسبه‌ی مقادیر خروجی در ازای یک رشته از الگوهای ورودی[۲۰] می‌باشد. همان‌طور که مشخص است، شبیه‌سازی روشی زمان‌بر است و موفقیت این روش وابسته به پاسخ محاسبه شده برای تمام الگوهای ورودی می‌باشد. در نتیجه احتمال مشخص نشدن خطاهای طراحی در طول شبیه‌سازی بسیار بالا است. علاوه بر این، با بالا رفتن اندازه و پیچیدگی مدار، تعداد الگوهای ورودی به صورت نمایی بالا می‌رود. این محدودیت باعث طولانی شدن شبیه‌سازی‌های واقعی می‌شود. با توجه به زمان محدود درستی‌سنجی، بسیاری از عیب‌ها، کشف[۲۱] نمی‌شوند. به‌عنوان مثال کل مدت زمان قابل انجام در مرحله پیش از ساخت پردازنده پنتیوم چهار معادل کمتر از دو دقیقه از اجرای آن با فرکانس یک گیگاهرتز می‌باشد [۵].
- **عدم مقیاس‌پذیری روش‌های درستی‌سنجی صوری**: روش‌های درستی‌سنجی صوری در مقایسه با روش‌های شبیه‌سازی، می‌توانند با اثبات درستی عملکرد طراحی، پاسخی قطعی برای

[20] Input patterns
[21] Detect

درستی‌سنجی داشته باشند. با این‌حال نیاز به جستجوی کل فضای حالت باعث رشد نمایی فضای جستجو می‌شود (انفجار حالت) که مقیاس‌پذیری این روش‌ها را محدود می‌کند. بنابراین نمی‌توان در طراحی‌های بزرگ از این شیوه استفاده نمود. به‌عنوان مثال در پردازنده پنتیوم چهار، برای تنها سه واحد ممیز شناور[۲۲]، رمزگشای دستور[۲۳] و زمان‌بندی پویا[۲۴] از این روش استفاده شده است [۴].

[22] Floating-point Execution Unit
[23] Instruction Decoder
[24] Dynamic Scheduler

۲- مروری بر ادبیات و پیشینه تحقیق

اگرچه تلاش‌های فراوانی در جهت توسعه و بهبود ابزارهای درستی‌سنجی و عیب‌یابی صورت گرفته است، ابزار موجود نیازمند بهبودهایی در راستای سرعت بیشتر و مقیاس‌پذیری بهتر می‌باشند. در این بخش برخی از کارهای انجام شده است با اهداف ذکر شده، را بررسی می‌کنیم.

۲-۱ درستی‌سنجی و عیب‌یابی در سطح گیت

درستی‌سنجی مدار می‌تواند به صورت پویا[۲۵] و یا ایستا[۲۶] انجام شود [۶] [۷]. روش‌های پویا در واقع مبتنی بر شبیه‌سازی[۲۷] هستند، که با تولید بردار آزمون[۲۸] و دادن ورودی[۲۹] (PI) به مدار، خروجی[۳۰](PO) آن را به‌دست آورده و با مقدار خروجی مطلوب[۳۱] (DPO) مقایسه می‌کنند. درصورتی که خروجی مدار با خروجی مطلوب برابر باشد، مدار بدون خطا تلقی می‌شود. در غیر این‌صورت، مدار دارای خطاست. توجه شود که ممکن است خطایی در مدار باشد و به وسیله‌ی بردارهای آزمون مورد استفاده، تشخیص داده نشود. بنابراین روش‌های مبتنی بر شبیه‌سازی تضمین نمی‌کنند که مدار بدون خطا است. اما در عوض این روش‌ها در مقایسه با سایر روش‌ها، سریع‌تر هستند. در مقابل، روش‌های ایستا که به روش‌های صوری[۳۲] نیز معروفند، با استفاده از الگوریتم‌های ریاضی، درستی مدار را اثبات می‌کنند. این روش‌ها از ابزارهای صوری[۳۳] مانند دیاگرام‌های تصمیم‌گیری[۳۴] [۸] و قابلیت تصدیق بولی[۳۵] [۹]

---

25 Dynamic
26 Static
27 Simulation-based
28 Test Vector
29 Primary Input
30 Primary Output
31 Desired Primary Output
32 Formal
33 Formal Engine
34 Decision Diagram
35 Boolean Satisfiability

استفاده می‌کنند. روش‌های عیب‌یابی براساس سطح مدار مورد بررسی به دو دسته تقسیم می‌شوند: عیب‌یابی در سطح گیت[۳۶] [۹] و عیب‌یابی در سطح انتقال ثبات[۳۷] [۱۰] و [۱۱].

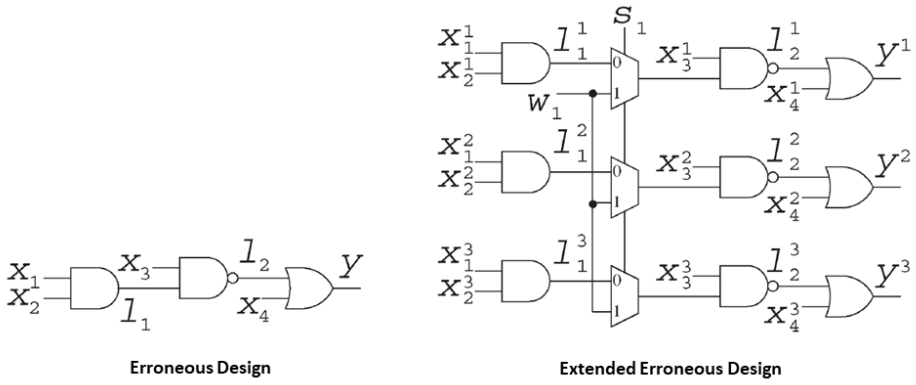
مهم‌ترین چالش در حوزه‌ی درستی‌سنجی و عیب‌یابی سیستم‌های دیجیتال، مقیاس‌پذیری[۳۸] است. از آنجا که انجام شبیه‌سازی برای مدارهای بزرگ فرآیندی زمان بر است، روش‌های صوری نسبت به روش‌های مبتنی بر شبیه‌سازی، مقیاس‌پذیرتر هستند. در بین روش‌های صوری نیز، روش‌های مبتنی بر قابلیت تصدیق‌پذیری بولی نسبت به روش‌های مبتنی بر دیاگرام‌های تصمیم‌گیری مقیاس‌پذیرتر هستند. زیرا با افزایش اندازه‌ی مدار دیجیتال، نمایش دیاگرام‌های تصمیم‌گیری عملا امکان‌پذیر نیست و بنابراین نمی‌توان از آن‌ها برای درستی‌سنجی و عیب‌یابی استفاده کرد.

به طور کلی الگوریتم‌های عیب‌یابی به دو دسته‌ی اثر-علت[۳۹] و علت-اثر[۴۰] تقسیم می‌شوند [012]. آنالیز اثر-علت ابتدا به خروجی‌های اشتباه مدار توجه می‌کند و با استفاده از تکنیک‌های ساختاری پیمایش آن، مکان‌هایی را که امکان وجود اشکال در آن‌ها می‌باشد را تشخیص می‌دهد. این روش‌ها مبتنی بر شبیه‌سازی یا شبیه‌سازی نمادین می‌باشند. آنالیز علت-اثر به طور عمده بر پایه ساخت فرهنگ اشکال‌ها[۴۱] می‌باشند [۱۳]. دراین روش برای هر خطا یک الگوی شبیه‌سازی انجام و نتایج آن در یک فرهنگ خطا ذخیره می‌شود. چنین روشی می‌تواند موجب تسریع عملیات پیدا کردن محل عیب یا اشکال شود، اما سایز فرهنگ خطای نهایی ممکن است بسیار زیاد شود [۱۰]. این روش ها به مدل خطا وابسته می‌باشند. یک مدل کلی از خطاهای طراحی وجود ندارد و مدل‌های ارایه شده کنونی فقط یک بخش کوچک از خطاهای ممکن طراحی، که ممکن است در فرآیند عیب یابی پیدا شوند، را منعکس می‌کنند [۱۴].

---

36 Gate Level
37 Register Transfer Level
38 Scalability
39 Effect-Cause
40 Cause-Effect
41 Fault Dictionary

در سال‌های اخیر، روش‌های نوینی (مانند روش های مبتنی بر SAT) برای غلبه بر معایب مربوط به تکنیک‌های درستی‌سنجی ایستا ارایه شده است و تکنیک‌های قبلی را عملی‌تر کرده‌اند. نویسندگان [۹] یک روش جدید ارایه داده‌اند که مساله‌ی عیب‌یابی را به صورت CNF کد می‌کند و محدودیت‌هایی مانند بردارهای آزمون ورودی، پاسخ-های طلایی و عدد تعداد خطاها را به آن اضافه می‌کند و آن را به یک حل‌کننده‌ی SAT می‌دهد تا عبارت‌های تصدیق‌کننده را که مطابق با مکان‌های بالقوه عیب هستند، بیابد. شیوه کار به این صورت است که، یک مدار در سطح گیت و تعدادی بردار آرمون در اختیار داریم. این بردارهای آزمون نشان می‌دهند که مدار خطا دارد. مطابق شکل ۱۰، بعد از هر گیت یک مالتی‌پلکسر ۲ به۱، قرار می‌دهیم و مدار را به ازای تعداد بردارهای آزمون تکرار می‌کنیم. توجه شود که همانطور که در این شکل مشخص است، ورودی انتخاب[۴۲] مالتی‌پلکسرهای متناظر به یکدیگر متصل می‌شود.



شکل ۱۰. چگونگی تکرار مدارخطادار به ازای بردارهای آزمون [۹]

درمرحله‌ی بعد، این مدار گسترش یافته را به فرم CNF[۴۳] تبدیل می‌کنیم. در این مرحله PI و DPOهای مربوط به بردارهای آزمون را به صورت عبارت تک جمله‌ای[۴۴] به CNF اضافه می‌کنیم. ابزار حل‌کننده‌ی قابلیت تصدیق بولی، CNF را دریافت کرده، مساله را حل و در انتها، مجموعه‌ای از ورودی‌های انتخاب مالتی‌پلکسر را یک می‌کند. در واقع، گیت‌هایی که ورودی انتخاب مالتی‌پلکسر متناظرشان، یک شده است، محل وقوع خطا هستند. توجه شود که

42 Select
43 Conjunctive Normal Form
44 Unit Clause

برای سادگی، در شکل ۱۰، فقط بعد از گیت اول، مالتی‌پلکسر قرار داده شده است. این روش برای مدارهای ترکیبی[۴۵] ارائه شده است. برای مدارهای ترتیبی[۴۶] تنها کافی است از تکنیک گسترش زمانی[۴۷] استفاده کرده و مدار اولیه را به ازای هر سیکل، تکرار کنیم.

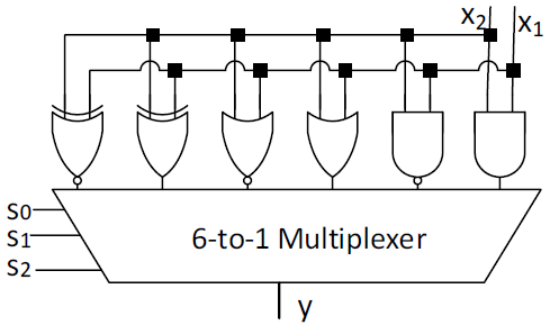
در [۱۵]، به منظور تسریع بخشیدن به حل مساله‌های عیب‌یابی، به خصوص وقتی چندین اشکال وجود دارد، یک مجموعه از هسته‌های غیر تصدیق‌شونده از CNF استخراج می‌شود. به منظور نمایش کارآمد مدارات ترتیبی، تکنیک‌های مبتنی بر فرمول بولی اندازه‌گیری شده[۴۸] (QBF) [۱۶]، به‌عنوان جایگزین تکنیک‌های آرایه‌ی منطقی تکرار شونده[۴۹] (ILA)، توصیه شده‌اند. در این تکنیک‌ها، فقط یک فریم تک زمانی مقداردهی می‌شود. بنابراین، یک مساله‌ی کوچکتر برای عیب‌یابی بدست می‌آید. برای مغلوب کردن مساله‌ی آرایه‌ی منطقی تکرار شونده و کاهش ابعاد مساله عیب یابی، روشهای مختلفی مانند تکنیک فشرده‌سازی ردیاب[۵۰] [۱۷] پیشنهاد شده است.

روش مبتنی بر قابلیت تصدیق بولی، جزء روش‌های صوری است که دو عیب اساسی دارد. این روش فقط می‌تواند خطاهایی را مکان‌یابی کند که توسط مجموعه بردارهای آزمون تشخیص داده شده‌اند. به عبارت دیگر، ممکن است خطایی در مدار وجود داشته باشد، اما چون بردارهای آزمون آن را تشخیص نداده‌اند، خطا اصطلاحا پوشانده[۵۱] شده است. بنابراین این روش تضمین نمی‌کند که مدار به طور کامل عیب‌یابی شده است. ایراد دوم روش، ضعف آن در مقابله با چالش مقیاس‌پذیری است. از آنجاکه بر سر راه هر گیت در مدار، یک مالتی‌پلکسر قرار داده و مدار باید به ازای هر بردار آزمون و هر گسترش زمانی، تکرار شود، تعداد عبارت‌ها درCNF نهایی برای مدارهای دیجیتال بزرگ، بسیار زیاد بوده و حل مساله‌ی عیب‌یابی یا به مدت زمان زیادی احتیاج دارد و یا امکان‌پذیر نیست. به همین دلیل، روش‌هایی ارائه شده است که هر کدام به نوعی تلاش کرده اند، ایرادهای این روش را برطرف سازند.

---

45 Combinational Circuits
46 Sequential Circuits
47 Time Expansion
48 Quantified Boolean Formula
49 Iterative Logic Array
50 Trace Compaction
51 Mask

در [۱۸] روشی ارائه شده است که عدم توانایی روش عیب‌یابی مبتنی بر تصدیق بولی در تضمین عیب‌یابی کامل مدار را برطرف می‌کند. روش معرفی شده در [۱۸] نه تنها قادر است، مدار را به طور کامل عیب‌یابی کند بلکه توانایی تصحیح مدار را هم دارد. روند به این صورت است که، یک مدار درسطح گیت و تعدادی بردار آزمون در اختیار داریم. بردارهای آزمون نشان می‌دهند که مدار دارای خطاست. فرض بر این است که تعدادی از گیت‌های مدار، که با احتمال بیشتری محل وقوع خطا هستند را به عنوان ورودی در اختیار داریم. بنابراین مساله‌ی عیب‌یابی، در واقع یافتن محل دقیق خطا از بین این گیت‌های مظنون[۵۲] است. برسر راه هر کدام از این گیت‌ها، یک مالتی‌پلکسر ۶ به ۱ مطابق شکل ۱۱، قرار داده می‌شود. این مالتی‌پلکسر، شش گیت پایه یعنی گیت‌های AND، NAND، OR، NOR، XOR و XNOR را پوشش می‌دهد. بعد از اینکه مدار را به ازای تعداد برادرهای آزمون و گسترش زمانی، تکرار کردیم، مدار را به CNF تبدیل کرده و PI و DPOها را به عنوان عبارت‌های تک جمله‌ای به CNF اضافه می‌کنیم. ابزار حل‌کننده‌ی قابلیت ارضای بولی، CNF را دریافت کرده، مسئله را حل و در انتها مجموعه‌ای از ورودی‌های انتخاب مالتی‌پلکسرها را یک می‌کند. مقدار ورودی‌های انتخاب مالتی‌پلکسر متناظر با هر گیت، مشخص می‌کنند که در صورت معیوب بودن گیت، آن را باید با کدام یک از گیت‌های پایه جایگزین کرد. بنابراین نه تنها مدار عیب‌یابی شده بلکه برای تصحیح آن نیز، یک راه حل به دست آمده است.

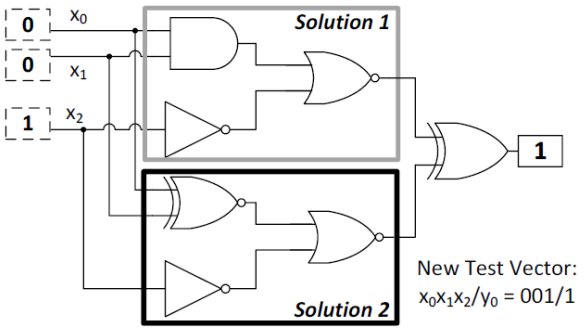


شکل ۱۱. مالتی‌پلکسر ۶ به ۱ استفاده شده در [۱۸]

در این مرحله، مکمل راه‌حل یافته‌شده را به CNF مدار اضافه کرده و از ابزار حل‌کننده‌ی قابلیت تصدیق بولی خواسته می‌شود که مساله را دوباره حل کند. سپس با پیاده‌سازی مدارهای تصحیح شده توسط این دو راه حل،

[52] Suspect

سعی می‌شود که بردار آزمون جدیدی تولید شود، به طوریکه رفتار این دو مدار برای این بردار آزمون، متفاوت باشد. بدین منظور مطابق شکل ۱۲، خروجی دو مدار توسط یک گیت XOR، متصل شده و بعد از تبدیل آن‌ها به CNF از ابزار حل‌کننده‌ی قابلیت تصدیق بولی خواسته می‌شود که مساله را حل کند. این ابزار یک بردار آزمون جدید را به عنوان خروجی برمی‌گرداند. در این مرحله، این بردار آزمون جدید به مجموعه بردارهای آزمون اولیه اضافه شده و روند قبل برای یافتن یک راه‌حل، جهت تصحیح مدار، مجددا تکرار می‌شود. این روند تاجایی ادامه می‌یابد که دیگر نتوان بردار آزمون جدیدی تولید کرد. به این ترتیب، با تولید بردارهای آزمون جدید، امکان عیب‌یابی جامع مدار میسر شده و یکی از مشکلات روش اولیه مبتنی بر تصدیق بولی بر طرف می‌شود. در مرجع [۱۹] نیز روشی پیشنهاد شده که سعی در بهبود روش معرفی شده در مرجع [۱۸]، دارد. در واقع در هر مرحله تکرار یکی از راه حل‌های پیدا شده به ازای بردار آزمون جدید، صحیح و راه‌حل دیگر نادرست است. بنابراین راه‌حل صحیح را به عنوان راه‌حل معتبر شناخته و در مرحله بعد، به جای تولید دو راه‌حل جدید، از راه‌حل معتبر مرحله‌ی قبلی استفاده می‌شود. به این ترتیب مدت زمان اجرا و میزان حافظه‌ی مصرفی بهبود می‌یابد.



شکل ۱۲. چگونگی تولید بردار آزمون جدید در [۱۸]

در مرجع [۱۰]، روشی مبتنی بر قابلیت تصدیق بولی ارائه شده است که سعی در کاهش زمان مکان‌یابی و بهبود مقیاس‌پذیری دارد. در این روش سعی می‌شود به جای انجام کامل مساله توسط ابزار حل‌کننده‌ی قابلیت تصدیق بولی، از اطلاعات مکان‌یابی قبلی استفاده شود. به این ترتیب که فرض می‌شود دو دسته آزمون ورودی داریم. برای اولین دسته از آزمون‌های ورودی، عیب‌یابی توسط ابزارحل‌کننده‌ی قابلیت تصدیق بولی انجام شده است. در دسته

دوم، برای تعدادی از بردارهای آزمون، از ابزار استفاده کرده و برای مابقی سعی می‌شود با استفاده از اطلاعات به دست‌آمده از عیب‌یابی دسته اول و الگوریتم‌های ابتکاری[۵۳] محل وقوع خطا پیش‌بینی شود. از آنجا که در این روش از الگوریتم‌های ابتکاری استفاده می‌شود، احتمال پیش‌بینی نادرست وجود دارد. به همین دلیل در مرجع [۱۱] سعی شده است که از شبکه عصبی برای پیش‌بینی استفاده شود.

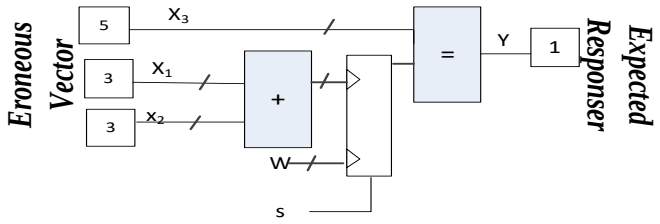
۲-۲ درستی‌سنجی و عیب‌یابی در سطح انتقال ثبات

روش‌های درستی‌سنجی و عیب‌یابی در سطح گیت، در برخورد با طراحی‌های بزرگ که با چندین عیب همراه می‌باشند، با مشکلاتی از قبیل بیرون‌ریزی زمان و فضا مواجه می‌شوند. امروزه تغییرات سریع‌تر طراحی‌ها، پیچیدگی و هم‌چنین مدت زمان لازم برای شبیه‌سازی آن‌ها، طراحان را بر آن داشته تا به سطوح بالاترِ تجرید مانند سطح انتقال ثبات یا سطح سیستم الکترونیکی سوق داده شوند. از طرف دیگر، نبود ابزارهای قوی و مقیاس‌پذیر برای عیب‌یابی و تصحیح خطاهای سطح انتقال ثبات به شدت زمان عرضه به بازار محصول را افزایش داده و در نتیجه میزان بهره‌وری طرح کاهش می‌یابد. برای رفع این مشکل، برخی روش‌ها پیشنهاد شده‌اند که به طور مستقیم در سطح انتقال ثبات کار می‌کنند. تکنیک ارایه شده در [۲۰] یک رویکرد تجزیه و تحلیل نرم‌افزاری را به کار می‌گیرد که به طور ضمنی از مالتی‌پلکسرها برای شناسایی جملات داخل کُدِ سطح انتقالِ ثبات که مسبب عیب می‌باشند، استفاده می‌کند. هم‌چنین این روش، مکان‌های وسیعی که امکان عیب در آن‌ها می‌باشد)، را باز می‌گرداند. یک راه برای غلبه بر این مشکل آن است که مالتی‌پلکسرها را به طور صریح داخل خود کُد اضافه شوند. مقاله [۲۱] از این رویکرد بهره می‌جوید. در این روش از تکنیک‌های آنالیز سخت‌افزاری بهره برده و تا حد زیادی دقت عیب‌یابی خطا را بهبود می‌بخشد.

در مرجع [۲۲]، یک روش عیب‌یابی ارایه گردیده است که از آنالیز صوری یک توصیف سخت افزاری و هم‌چنین مشخصه‌های ناموفق استفاده می‌کند تا جملات معیوب را شناسایی می‌کند. این تکنیک تنها می‌تواند در یک

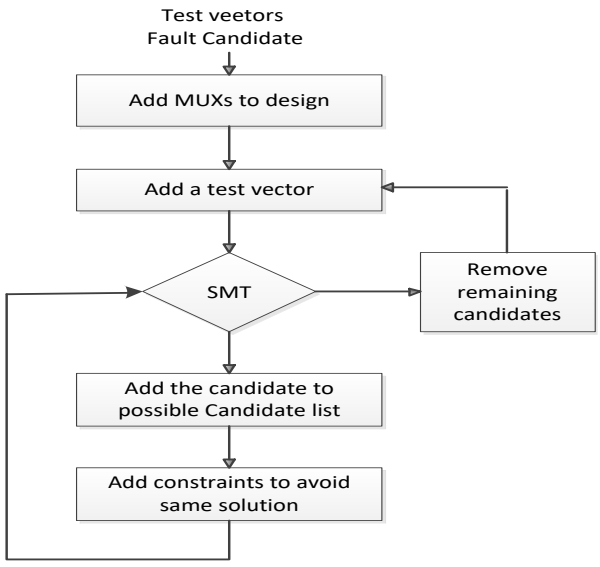
[53] Heuristic

چارچوب رسمی استفاده شود. رویکردهایی مبنی بر حل‌کننده‌های SMT نیز وجود دارند که امکان عیب‌یابی طراحی‌های سطح انتقال ثبات را فراهم می‌کند. این روش‌ها به وجود ردیابی‌های نادرست و هم‌چنین نتایج به دست آمده‌ی درستِ متناظر با آن‌ها وابسته می‌باشند. در [۲۳] یک روش عیب‌یابی معرفی گردیده است که عیب‌یابی مبتنی بر SAT را برای طراحی‌های سطح انتقال ثبات توسعه می‌دهد. توصیف طراحی و نیز سیگنال‌های نامزد خطا در سطح کلمه مشخص گردیده و مالتی‌پلکسرهای سطح کلمه به سیگنال‌های خطا اضافه می‌شوند. در نهایت برای حل فرمول به دست آمده، یک حل‌کننده‌ی سطح کلمه استفاده می‌شود. مالتی‌پلکسرهای اضافه شده در واقع مالتی پلکسرهای کلمه‌ای می‌باشد و محل اضافه شده برای هر مالتی‌پلکسر پس از هر عبارت در یک بلاک است. نحوه‌ی عملکرد مالتی پلکسر کلمه‌ای مطابق شکل ۱۳ است.



شکل ۱۳. روش عیب‌یابی مبتنی بر مالتی‌پلکسرهای کلمه‌ای [۱۹]

شیوه عیب‌یابی مطابق شیوه عیب یابی بخش قبل می باشد. به این معنا که ابتدا مالتی‌پلکسرهای کلمه‌ای به مدار اضافه می‌شود. سپس مجموعه بردارهای ورودی و پاسخ آنها به عنوان یک محدودیت به مدار حاصل اضافه می‌شود. مدار استخراج شده در سطح انتقال ثبات به یک حل‌کننده SMT داده شده و به این صورت کاندیداهای خطا به دست می‌آیند. برای جلوگیری از تکرار این جواب، متمم این کاندیدها به عنوان محدودیت به CNF مدار اضافه می شود. این پروسه تا اتمام تمام بردارهای آزمون ادامه می یابد. شکل ۱۴ این روند را نشان می دهد.



شکل ۱۴. روند عیب‌یابی مبتنی بر SMT-solver ها [۱۹]

به دست‌آوردن یک مجموعه کاهش‌یافته از کاندیدهای عیب[۵۴]، می‌تواند به فرآیند عیب‌یابی طراحی‌های سخت‌افزاری در سطح انتقال ثبات کمک شایانی نماید. چنانچه مجموعه کاندیداهای خطای به دست آمده خیلی بزرگ باشد، تشخیص صحیح خطای طراحی با آزمودن یک به یک این کاندیدها زمان و تلاش زیادی را می‌طلبد. در [۲۴] روشی به‌نام عیب‌یابی با اولویت‌بندی[۵۵] برای سرعت بخشیدن به فرآیند جستجوی عیب در مجموعه کاندیدهای خطاها ارایه شده است. در این کار، یک معیار که امتیاز اطمینان [۵۶](CS) نامیده می‌شود، برای تعیین احتمال درست بودن هر کاندیدای خطا استفاده می‌شود. سپس کاندیدهای خطا بر اساس CS، مرتب می‌شوند تا آنها به ترتیب الویت نمایش داده می‌شوند. وقتی که این ترتیب مشخص شد، خطای طراحی صحیح در چند خط اول ترتیبِ داده شده قرار خواهد گرفت. اگر طراح بر اساس ترتیب داده شده، کاندیداهای خطا را بیازماید، خطای واقعی می‌تواند راحت‌تر پیدا شود. این روش به صورت ضمنی شرایط پوشش خطا را نادیده می‌گیرد. بدون در نظر گرفتن این شرایط، معیار CS ممکن است که خطای واقعی را در الویت‌های اولیه قرار ندهد و در نتیجه کارایی روش محدود می‌شود. بنابراین در [۱۵] تلاش شده است تا یک معیار احتمالی جدید، که احتمال امتیاز اطمینان

[54] Reduced Potentail Error location
[55] Debugging priority
[56] Confidence Score

(PCS)[57] نامیده می‌شود، ارایه شود. در این معیار شرایط پوشش خطا نیز در نظر گرفته شده است تا بتواند تخمین دقیق‌تری نسبت به معیار CS بدهد. در این مرجع به جای ارایه یک روش جدید، برای بدست آوردن یک مجموعه‌ی کاهش یافته از کاندیدهای عیب، بر روی یک اولویت‌بندی صحیح‌تر بین کاندیدهای عیب بدست آمده از سایر روش‌ها تمرکز دارد. این روش بر اساس مجموعه آزمون است که از طریق شبیه‌سازی تولید می‌شوند. اگر مجموعه آزمون تولید شده به میزان کافی پوشش نداشته باشد، عیب طراحی پیدا نخواهد شد و بنابراین کارایی به میزان قابل توجهی کاهش خواهد یافت. ضمن این که این روش تنها در صورتی کار می‌کند که تنها یک عیب در مدار وجود داشته باشد. اگرچه هر نوع عیبی، اعم از عیب های مسیر داده یا کنترلی، در این مدارات می تواند وجود داشته باشد.

پس از مکان‌یابی عیب، تصحیح آن نیز امری ضروری است. بسیاری از روش‌های عیب‌یابی فاقد فرآیند تصحیح اتوماتیک هستند. اما با این وجود روش‌هایی برای تصحیح خطای طراحی برای مدارات (به خصوص ترکیبی) در طول سال‌های اخیر پیشنهاد شده است که بر دو دسته‌ی کلی مبتنی بر تطبیق-خطا[58] [۲۵] و مبتنی بر سنتز مجدد[59] [۲۶] می‌باشند. نویسندگان [۲۷] به منظور یافتن تصحیحات ممکن، روشی پیشنهاد کرده‌اند که بر اساس مثال‌های نقضی است که در نظر گرفته می‌شود. این روش برای طراحی‌های سطح انتقال ثبات ارایه شده است و با استفاده از تکنیک برش برنامه و محدودیت‌های تولید شده از مثال‌های نقض، اصلاحات لازم را پیدا می‌کند. در نهایت با بکارگیری حل‌کننده‌های SMT، بررسی می‌شود که آیا چنین جایگزینی در راستای اصلاح مثال‌های به دست آمده وجود دارد یا خیر. اما این روش تنها قادر به تصحیح یک عیب در طراحی می‌باشد. سنتز مجدد روشی است که بر مبنای یک جدول درستی رفتار می‌کند و مبتنی بر ورودی های طراحی تحت درستی‌سنجی می‌باشد. اما استفاده از آن برای تصحیح خطا در سطح انتقال ثبات یا بالاتر از آن معایبی را به دنبال خواهد داشت. از آن جا که امکان خواندن تصحیح وجود ندارد، بررسی آن توسط طراح میسر نخواهد بود؛ به علاوه تصحیح ایجادشده بر

[57] Probabilistic Confidence Score
[58] Error-Matching
[59] Resynthesis

اساس مجموعه‌ی محدودی از مثال‌های نقض می‌باشد. بنابراین نمی‌توانیم از تطبیق طراحی اصلاح‌شده با توصیف متناظر آن اطمینان حاصل کنیم. به همین دلیل هنگامی که نیاز به افزودن مثال‌های نقض جدیدتری است که پیشتر در طی فرایند تصحیح در نظر گرفته نشده‌اند، امکان شکست وجود خواهد داشت. در [۲۸] از آزمون جهش برای عیب‌یابی و تصحیح طراحی‌هایی با مسیر داده‌ی چندجمله‌ای بزرگ استفاده شد. این روش می‌تواند به طور خودکار عیب‌ها را پیدا و تصحیح کند، اما فقط می‌تواند به طراحی‌های مسیر داده چندجمله‌ای اعمال شود و نمی-تواند به طور کارآمدی مساله‌ی اشکال‌یابی را وقتی چندین عیب در طراحی وجود دارد بررسی کند. این مشکل به دلیل استفاده از روش فراگیر[۶۰] برای یافتن محل اشکال و تصحیح طراحی داری عیب است.

۲-۳ درستی‌سنجی و عیب‌یابی در سطح سیستم

با افزایش پیچیدگی طراحی‌ها، روند تکاملی سطوح انتزاع طراحی اکنون به سطح بالاتری به نام سطح سیستم رسیده است. روش طراحی در این سطح عموما به صورت توصیف واحدهای محاسباتی بسیار بزرگ (که خود سیستم‌های پیچیده‌ای دارند)، کانال‌های ارتباطی میان این واحدها، سوییچ‌ها و گذرگاه‌های داده با عملکرد پیچیده است که توسط زبان های سطح بالا همچون C، C++ و SystemC و یا گاهی توسط روش مدل‌سازی تراکنشی (TLM)[۶۱] توصیف می‌شوند.

یک روش اجرایی نمادین که بر پایه‌ی حل‌کننده‌ی SMT می‌باشد در [۲۹] معرفی شده است که مختص برنامه‌های نرم‌افزاری و نیز سیستم توصیف‌شده به زبان C است. روش تصحیح آن بر اساس الگوهایی است که تنها برای سطح تجرید بالای طراحی مناسب می‌باشد. شاید بتوان با اندکی تغییر، از آن در سطح توصیفات سیستمی سخت افزاری نیز بهره جست.

60 Exhastive
61 Transaction Level Modeling

در [۳۰] روشی برای عیب‌یابی این طراحی‌ها ارایه شده است که می‌تواند مکان عیب‌ها را بر اساس برش‌های پویا[۶۲] پیدا کرده و بر مبنای تکنیک جهش، آنها را تصحیح نماید. به دلیل وابسته بودن این روش به ورودی‌های آزمایش‌شده، نمی‌توان از تطابق طراحی اصلاح‌شده و توصیف آن اطمینان حاصل کرد.

در سال‌های اخیر استفاده از ابزارهای سنتز سطح بالا نیز رونق یافته است. این ابزار به طراح کمک می کند تا بتواند یک طراحی که در سطح سیستم با استفاده از زبان‌های سطح بالا همچون C، C++ و SystemC و یا گاهی هم توسط ساختار مدل سازی تراکنشی[۶۳] توصیف شده است را به کد Verilog یا VHDL در سطوح مختلف تبدیل نماید. ابزارهای سنتز سطح بالای کنونی، در استخراج گراف جریان داده-کنترل (CDFG[۶۴])ها از توصیف‌های سطح بالای داده شده، بسیار توانا هستند. مراحل بهینه‌سازی که پس از استخراج CDFG مطرح می‌شوند، اعمال زمان‌بندی[۶۵]، اختصاص منابع[۶۶]، اشتراک منابع[۶۷]، خط‌لوله[۶۸]، زمان‌بندی مجدد[۶۹]و سنتز کنترل[۷۰] هستند. بعد از این مراحل بهینه‌سازی، طراحی‌ها با استفاده از ابزارهای سنتز منطقی و سنتز فیزیکی به سخت‌افزار سنتز می‌شوند. در این ابزارها، بررسی کردن برابری بین سطح سیستم و کد HDL توصیف شده، امری ضروری به نظر می‌رسد.

در [۳۱] از مجموعه‌ای از بهینه‌سازی‌ها نظیر بهینه‌سازی قطع نقطه[۷۱]، بهینه‌سازی قطع حلقه[۷۲] و قطع صفحه[۷۳] برای کاهش پیچیدگی عملیات بررسی برابری استفاده شده است. موتور اصلی این روش بر مبنای روش‌های بسیار پیچیده اثبات قضیه و بازنویسی می‌باشد. نتایج تجربی به دست آمده حاکی از عملکرد موفق این روش دارد. اما هنگامی یک طراحی سطح بالا قرار است به صورت خط‌لوله سنتز شود، پدیده‌ای متعددی هنگام خط‌لوله کردن در

---

62 Dynamic Slicing
63 Transaction Level modeling
64 Control Data Flow Graph
65 Scheduling
66 Binding
67 Resource Sharing
68 Pipelining
69 Re-synthesis
70 Control Synthesis
71 Cut-point
72 Cut loop
73 Cut plane

طول مراحل زمان‌بندی و اختصاص منابع اتفاق می‌افتد که استفاده از فنون بهینه‌سازی بررسیِ کردن برابری نظیر بهینه سازیِ قطعِ نقطه و بهینه سازیِ قطعِ حلقه را بسیار سخت ویا غیر ممکن می‌نماید. به همین جهت ایده به کار رفته در مقاله [۳۱] به طور مستقیم به معماری خط‌لوله قابل اضافه شدن نیست. بنابراین در [۳۲] به جای مقایسه مستقیم، یک مدل مرجع خط لوله ای معرفی شده است که از توصیف بدست می‌آید. این مدل به صورت کاملاً اثبات‌پذیری توصیفِ مدار را نگه‌داری می‌کند. آماده کردن این گراف به صورت برون خط[۷۴] صورت می‌گیرد. در اینجا فرض می‌شود یک گراف جریان داده-کنترل به نام G تولید شده است. با خط‌لوله کردن حلقه‌ی آن، یک گراف جریان داده دیگر به نام G’ تولید می‌شود. الگوریتم پیشنهادی اطلاعاتی نظیر تعداد تکرارها برای خط‌لوله‌ای کردن هنگام حباب خوردن[۷۵] و نظایر آن را به عنوان ورودی می‌گیرد و سعی در خط‌لوله کردن حلقه می‌نماید. تا بدین وسیله طراحی سطح بالا و پایین را کمی به یکدیگر نزدیک نماید. در این صورت نگاشت یک‌به‌یک بین مدل ایجاد شده و کدسنتز شده می‌تواند به وجود آید. در این صورت روش پیشنهادی قادر به استفاده از تکنیک‌های قطع نقطه و حلقه است ولی در عین حال احتیاج به اطلاعات استخراجی از ابزار سنتز سطح بالا دارد و برای حلقه های تودرتو نیز فاقد کارایی است.

[74] Off-line
[75] Stall

۳- دلایل انجام طرح پژوهشی

همانطور که قبلا بحث شد، اگرچه تلاشهای فراوانی در جهت بهبود ابزار درستی‌سنجی و عیب‌یابی در سطوح مختلفی صورت گرفته است، ابزار فعلی پاسخ‌گوی نیاز فعلی بازار نمی‌باشد که دلایل زیر برای آن قابل بیان هستند:

- پیچیدگی روزافزون طراحی‌های امروزی نسبت به گذشته
- زمان محدود عرضه به بازار
- سرعت پایین روش‌های شبیه‌سازی جهت پوشش کل مدار
- عدم مقیاس‌پذیری روش‌های درستی‌سنجی صوری

بنابراین ارایه ابزاری که یک یا چند مورد از موارد بالا را بهبود دهد، می‌تواند کمک شایانی در روند طراحی مدارهای دیجیتال بزرگ نماید. بر این اساس، تمرکز اصلی این پژوهش ارایه‌ی ایده‌هایی جهت حل مشکلات روشهای درستی‌سنجی و عیب‌یابی موجود و بطور خاص پرداختن به موضوع تصحیح ایرادهای طراحی می‌باشد. لازم به ذکر است که از جمله کارهایی که این گروه پژوهشی در گذشته انجام داده است می‌توان به مراجع [۱۸]، [۱۹]، [۲۸]، [۳۳] الی [۵۳] اشاره نمود که در آنها از روش‌های صوری، دیاگرام‌های تصمیم‌گیری و تکنیکهای جبری استفاده شده تا برخی از مشکلات ذکر شده در بخشهای قبلی در ارتباط با روش‌های درستی‌سنجی و عیب‌یابی حل شوند. بهبود این روش‌ها در سطوح انتقال ثبات و گیت، در این پژوهش مورد بحث و بررسی قرار خواهد گرفت.

۴- برنامه و نحوه اجرای طرح

روش تحقیق درنظر گرفته شده، بر اساس تقسیم طرح به مراحل: ۱) بهبود روش‌های درستی‌سنجی، ۲) بهبود روش‌های عیب‌یابی و ۳) تجمیع دو روش در یک قالب واحد جهت تصحیح ایرادهای طراحی، می‌باشد. در این راستا ضمن بررسی منابع مختلف و کارهای انجام شده در هر یک از مراحل فوق، به ارایه روشهای جدید درستی سنجی و عیب یابی برای مدارات دیجیتال خواهیم پرداخت.

اگرچه روش جمع‌آوری اطلاعات موردنیاز مبتنی بر بررسی مقالات، کتب و منابع اینترنتی است، به منظور نشان دادن کارایی تکنیکهای درستی‌سنجی و عیب‌یابی پیشنهادی، مدارهای مختلفی که از سیستم‌های نهفته واقعی مانند کاربردهای DSP، پردازش تصویر و مخابرات استخراج شده‌اند، به کار برده خواهند شد که نمونه‌هایی از آنها در مقالات این گروه (مراجع [۱۸]، [۱۹]، [۲۸]، [۳۳] الی [۵۳]) آورده شده‌اند.

۵- برنامه زمانی طرح

نمودار زمان‌بندی این طرح که در مدت ۱۲ ماه قابل اجرا است به شرح جدول ۱ می‌باشد.

جدول ۱. نمودار زمان‌بندی اجرای طرح پژوهشی

شماره فعالیت	نام فعالیت	مدت زمان اجرا (بر حسب ماه)	نمودار زمان‌بندی (بر حسب ماه)											
			۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲
۱	مطالعه جدیدتریم روش‌های درستی‌سنجی و عیب‌یابی صوری	۲												
۲	پیشنهاد روش جدید تصحیح ایرادهای طراحی در سطوح انتقال ثبات و گیت	۶												
۳	توسعه ابزار	۸												
۴	آماده‌سازی و نگارش مستندات	۲												

۶- مراجع

[۱] P. Rashinkar, P. Paterson, and L. Singh, System-on-a-Chip Verication: Methodology and Techniques. Boston, MA: Kluwer, 2000.

[۲] Intel Corporation, Moore's Law, Available: http://www.intel.com/Technology/ Moreslaw/index.htm, 2010.

[۳] Wilson Research Group and Mentor, 2020 Functional Verification Study, https://go.mentor.com/5ffxz.

[۴] I. Wagner, V. Bertacco, and T. Austin, "Shielding against design flaws with field repairable control logic," in *Proceedings of the 43rd ACM/IEEE Design Automation Conference*, 2006, pp. 344-347.

[۵] J. J. W. Kuan and T. M. Aamodt, "Progressive-BackSpace: Efficient Predecessor Computation for Post-Silicon Debug," in *Proceedings of the IEEE International Workshop on Microprocessor Test and Verification (MTV'11)*, 2011.

[۶] R. Drechcler, Advanced Formal Verification, Second Edition, 2004.

[۷] B. Wile, J. Goss, W. Roesner, Comprehensive Functional Verification: the Complete Industry Cycle, Morgan Kaufmann 2005.

[۸] O. Sarbishei, M. Tabandeh, B. Alizadeh and M. Fujita, "A Formal Approach for Debugging Arithmetic Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 5, pp. 742 - 754, 2009.

[۹] A. Smith, A. Veneris, M. Fahim Ali, and A. Viglas, "Fault diagnosis and logic debugging using Boolean satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 1606-1620, 2005.

[۱۰] N. Veira, Z. Poulos and A. Veneris, "Suspect Set Prediction in RTL Bug Hunting," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018.

[۱۱] N. Veira, Z. Poulos and A. Veneris, "Suspect2vec: A Suspect Prediction Model for Directed RTL Debugging," in *the 24th Asia and South Pacific Design Automation Conference*, 2019.

[۱۲] N. Jha, et. al. , Testing of Digital Systems, New York: Cambridge Univ. Press, 2003.

[۱۳] V. Boppana, I. Hartanto, and W. K. Fuchs, "Full fault dictionary storage based on labeled tree encoding," in 14th IEEE VLSI Test Symposium (VTS'96), April 28 - May 1, 1996, Princeton, NJ, USA, 996, pp. 174–179.

[۱۴] Holst, et. al.,"Adaptive Debug and Diagnosis without Fault Dictionaries,", in *Proc. of (ETS '07)*, pp. 20-24, 2007.

[۱۵] A. Sülflow, G. Fey, R. Bloem, and R. Drechsler, "Using Unsatisable Cores to Debug Multiple Design Errors," *in Proc. Great Lakes Symposium on VLSI (GLSVLSI'08)*, pp. 77-82, 2008.

[۱۶] H. Mangassarian, A. Veneris, S. Safarpour, M. Benedetti, and D. Smith," A Performance-Driven QBF-Based Iterative Logic Array Representation with Applications to Verification, Debug and Test," in *Proc. International Conference on Computer-Aided Design (ICCAD'07)*,pp. 240-245, 2007.

[۱۷] Y. Chen, S. Safarpour, and A. Veneris, "Optimal Trace Compaction with Property Preservation," *in Proc. International Midwest Symposium on Circuits and System (MWSCAS'09)*, pp.1207-1210, 2009.

[۱۸] B. Alizadeh and S. R. Sharafinejad, "Incremental SAT-based Accurate Autocorrection of Sequential Circuits through Automatic Test Pattern Generation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 38 , no. 2, pp. 245 - 252, 2019.

[۱۹] B. Alizadeh and Y. Abadi, "Incremental SAT-based Correction of Gate Level Circuits by Reusing Partially Corrected Circuits," *IEEE Transactions on Circuits and Systems II: Express Briefs,* 2020.

[۲۰] R. Konighofer and R. Bloem, "Automated Error Localization and Correction for Imperative Programs," in *Proc. of Formal Methods in Computer Aided Design (FMCAD'11),* pp. 91-100, 2011.

[۲۱] K. Chang, I. Wagner, V. Bertacco, and I. L. Markov," Automatic Error Diagnosis and Correction for RTL Designs," in *Proc, of .High Level Design, Verification and Test Conference (HLDVT'07),* pp.683-688, 2007.

[۲۲] R. Bloem, and F. Wotawa, "Verification and Fault Localization for VHDL Programs", in *Proc. of Journal of the Telematics Engineering Society, (TIV'02),* vol. 2, pp. 30-33, 2002.

[۲۳] S. Mirzaeian, F. Zheng, and K.-T.T. Cheng, "RTL Error Diagnosis Using a Word-Level SAT-Solver," *in Proc. of Internationa*l *Test Conference (ITC'08),* pp. 1-8, 2008.

[۲۴] T. Y. Jiang, C. N. Liu, and J. Y. Jou, "Effective error diagnosis for RTL designs in HDLs," in *Proc. 11th Asia Test Symp., 2002, pp. 362–367.*

[۲۵] U. Repinski, H. Hantson, M. Jenihhin, J. Raik, R. Ubar, G. Di Guglielmo, G. Pravadelli, F. Fummi," Combining Dynamic Slicing And Mutation Operators For ESL Correction," *in Proc. of European Test Symposium, (ETS'12),* pp. 1-6, 2012.

[۲۶] K. Chang, I. Markov, and V. Bertacco, "Fixing Design Errors With Counterexamples and Resynthesis, " in IEEE Transactions on Computer Aided design (TCAD'08), vol. 27, no. 1, pp. 184-188, 2008.

[۲۷] T. Matsumoto, S. Ono, M. Fujita: "An Efficient Method To Localize And Correct Bugs In High-Level Designs Using Counterexamples and Potential Dependence," in *Proc. of Very Large Scale Integration System on Chip (VLSI-SoC'12),* pp. 291-294, 2012.

[۲۸] B. Alizadeh, "A Formal Approach to Debug Polynomial Datapath Designs," in *Proc. of Asia and South Pacific Design Automation Conference (ASP-DAC'12),* pp.683-688, 2012.

[۲۹] R. Konighofer and R. Bloem, "Automated Error Localization and Correction for Imperative Programs," in *Proc. of Formal Methods in Computer Aided Design (FMCAD'11),* pp. 91-100, 2011.

[۳۰] U. Repinski, H. Hantson, M. Jenihhin, J. Raik, R. Ubar, G. Di Guglielmo, G. Pravadelli, F. Fummi," Combining Dynamic Slicing And Mutation Operators For ESL Correction," *in Proc. of European Test Symposium, (ETS'12),* pp. 1-6, 2012.

[۳۱] K. Hao, F. Xie, S. Ray, and J. Yang. "Optimizing Equivalence Checking for Behavioral Synthesis," in *Proc. of Design Automation & Test in Europe , (DATE'10),*pp. 1500-1505, 2010.

[۳۲] K. Hao, Sandip Ray, Fei Xie, "Equivalence Checking for Behaviorally Synthesized Pipelines," in *Proc. of Design Automation Conference (DAC'12),* pp 344-349. 2012.

[۳۳] B. Alizadeh, P. Behnam, and S. Sadeghi-kohan, *A Scalable Formal Debugging Approach with Auto-correction Capability based on Static Slicing and Dynamic Ranking for RTL Datapath Designs*, in IEEE Transactions on Computers (TC), Vol. 64. No. 6, June 2015, pages 1564-1578.

[۳۴] B. Alizadeh, and P. Behnam, *Formal Equivalence Verification and Debugging Techniques with Auto-correction Mechanism for RTL Designs*, in Microprocessors and Microsystems – Embedded Hardware Design, Vol. 37, No. 8-D, November 2013, pages 1108-1121.

[۳۵] B. Alizadeh, *Formal Verification and Debugging of Precise Interrupts on High Performance Microprocessors*, in ACM Transactions on Design Automation of Electronic Systems (TODAES), Vol. 17, No. 4, October 2012, pages 37-1:37-8.

[۳۶] B. Alizadeh, and M. Fujita, *Modular Data-path Optimization and Verification Based on Modular-HED*, in IEEE Transactions on Computer-Aided-Design of Integrated Circuits and Systems (TCAD), Vol. 29, No. 9, September 2010, pages 1422-1435.

[۳۷] B. Alizadeh and M. Fujita, *A Unified Framework for Equivalence Verification of Datapath-oriented Applications*, in IEICE TRANS. INF. & SYST., Vol. E92-D, No. 5, May 2009, pages 985-994.

[۳۸] H. Haghbayan, B. Alizadeh, A. Rahmani, P. Liljeberg and H. Tenhunen, *Automated Formal Approach for Debugging Dividers Using Dynamic Specification*, DFTS 2014, Netherlands, pages 264-269.

[۳۹] F. Farahmandi, B. Alizadeh, and Z. Navabi, *Effective Combination of Algebraic Techniques and Decision Diagrams to Formally Verify Large Arithmetic Circuits*, ISVLSI 2014, USA, pages 338-343.

[۴۰] S. Sadeghi-kohan, P. Behnam, B. Alizadeh, M. Fujita, and Z. Navabi, *Improving Polynomial Datapath Debugging with HEDs*, ETS 2014, Germany, pages 1-4.

[۴۱] P. Behnam, B. Alizadeh, and Z. Navabi, *Automatic Correction of Certain Design Errors using Mutation Technique*, ETS 2014, Germany, pages 1-2.

[۴۲] M. H. Haghbayan, B. Alizadeh, P. Behnam, and S. Safari, *Formal Verification and Debugging of Array Dividers with Auto-correction Mechanism*, VLSI Design 2014, India, pages 80-85.

[۴۳] F. Farahmandi, and B. Alizadeh, *Groebner basis based formal verification of large arithmetic circuits using Gaussian elimination and cone-based polynomial extraction*, in Microprocessors and Microsystems – Embedded Hardware Design, Vol. 39, No. 2, March 2015, pages 83-96.

[۴۴] P. Behnam, and B. Alizadeh, *In-circuit mutation-based automatic correction of certain design errors using SAT mechanisms*, ATS 2015, India, pages 199-204.

[۴۵] P. Behnam, B. Alizadeh, S. Taheri, and M. Fujita, *Formally Analyzing Fault Tolerance in Datapath Designs using Equivalence Checking*, ASPDAC 2016, Macao, pages 133-138.

[۴۶] S. Beigmohammadi, and B. Alizadeh, *Combinational Trace Signal Selection with Improved State Restoration for Post-silicon Debug*, DATE 2016, Germany, pages 1369-1374.

[۴۷] M.R. Azarbad, and B. Alizadeh, *Scalable SMT-based Equivalence Checking of Nested Loop Pipelining in Behavioral Synthesis*, in ACM TODAES 2017, Vol. 22, No. 2, March 2017, Article No. 22.

[۴۸] F. Refan, B. Alizadeh, and Z. Navabi, *Bridging Pre- and Post-silicon Debugging by Instruction-based Trace Signal Selection in Modern Processors*, in IEEE Transactions on VLSI (TVLSI), Vol. 25, No. 7, July 2017, pages 2059-2070.

[۴۹] S. Salamat, M. Ahmadi, B. Alizadeh, and M. Fujita, *Systematic Approximate Logic Optimization using Don't Care Conditions*, ISQED 2017, USA, pages 419-425.

[۵۰] A.R. Mahzoon, and B. Alizadeh, *Systematic Design Space Exploration of Floating Point Expressions on FPGA*, in IEEE Transactions on Circuits and Systems II (TCAS-II), Vol. 64, No. 3, March 2017, pages 274-278.

[۵۱] M. Grailoo, B. Alizadeh, and B. Forouzandeh, *Improved Range Analysis in Fixed-point Polynomial Datapath*, in IEEE Transactions on Computer-Aided-Design of Integrated Circuits and Systems (TCAD), Vol. 36, No. 11, November 2017, pages 1925-1929.

[۵۲] M. H. Haghbayan, and B. Alizadeh, *A Dynamic Specification to Automatically Debug and Correct Various Divider Circuits*, in Integration the VLSI Journal (INTEGRATION), January 2016, Vol. 53, pages 100-114.

[۵۳] R. Sharafinejad, B. Alizadeh, and Z. Navabi, *Automatic Correction of Dynamic Power Management Architecture in Modern Processors*, in IEEE Transactions on VLSI (TVLSI), Vol. 26, No. 2, February 2018, pages 308-318.