ساختار جدید از یک مقایسه گر تحمل پذیر خطا

سید امین علوی[1] **سید جواد سید مهدوی چابک[2]**

۱- دانشجوی دکتری گروه مهندسی برق- واحد مشهد- دانشگاه آزاد اسلامی - مشهد- ایران

aminalavi@mshdiau.ac.ir

۲-استادیار گروه مهندسی برق- واحد مشهد- دانشگاه آزاد اسلامی - مشهد- ایران

mahdavi@mshdiau.ac.ir

چکیده: در این مقاله یک طراحی جدید از مقایسه گر تحمل پذیر خطا ارائه شده که با افزودن یک یدکی بدون خطا می‌تواند بدون اثرگذاری و ایجاد وقفه بر روی عملکرد عادی مقایسه گر، آن را به یک سیستم تحمل پذیر خطا تبدیل کند. مدار برای راحتی عملیات تست به بخش‌های کوچکتری تقسیم شده و یک یدکی برای پیکربندی جدید به آن اضافه شده است. ما از روند بازیابی پویا و روش hot-standby در عیب یابی و تصحیح استفاده کرده ایم که این قابلیت را دارد که بدون اعمال وقفه به روند عادی سیستم، آن را تست و خطا را آشکار و پیکربندی مجدد انجام دهد. این روش در پارامترهای مساحت، تاخیر و پیچیدگی مدار بسیار کارآمد است.

کلمات کلیدی: مقایسه گر، سیستم تحمل پذیر خطا، Comparator، Fault Tolerant، Fault-free.

تاریخ ارسال مقاله: ۱۳۹۵/۰۹/۲۲

تاریخ پذیرش مشروط مقاله: ۱۳۹۶/۰۵/۰۴

تاریخ پذیرش مقاله: ۱۳۹۶/۱۱/۱۵

نام نویسنده‌ی مسئول: سید جواد سید مهدوی چابک

نشانی نویسنده‌ی مسئول: ایران – مشهد – قاسم آباد- بلوار امامیه- خیابان استاد یوسفی- دانشگاه آزاد اسلامی - دانشکده مهندسی برق

۱- مقدمه

سیستم تحمل پذیر خطا[1]، سیستمی است که بتواند وظیفه خود را به درستی حتی با وجود خطاهای سخت افزاری و اشتباهات نرم افزاری ادامه دهد. تحمل خطا یکی از نگرانی‌های عمده در عملکرد سیستم‌های دیجیتال است و برای اینکه سیستم صحیح کار کند و تاثیرخرابی برروی عملکرد آن کاهش یابد باید ابتدا خطا پیش بینی سپس مدیریت و کنترل شود. روش‌های تحمل خطا برای پیش بینی خرابی و انجام یک اقدام مناسب قبل از خطا می باشد. افزایش قابلیت اطمینان عملکرد سیستم‌های دیجیتال بوسیله پیاده سازی ساختار تحمل پذیر خطا، امکان پذیز است. تحمل پذیری خطا در یک سیستم از طریق افزونگی در سخت افزار، نرم افزار، اطلاعات یا محاسبات بدست می آید. این قبیل افزونگی می تواند در پیکربندی ایستا، پویا یا ترکیبی پیاده سازی شود. افزونگی سخت افزاری بوسیله فراهم کردن دو یا تعداد بیشتری نمونه‌های فیزیکی از یک مولفه سخت افزاری بدست می‌آید. هدف مهم در طراحی سیستم‌ها این است که سیستم بتواند بصورت خودکار در برابر بخشهایی که دچار خرابی شده اند خودش را بازیابی کند بدون اینکه این امر تاثیری در عملکرد کلی سیستم داشته باشد.

مقایسه بین دو مقدار باینری به صورت گسترده ای در سیستم‌های کامپیوتری و رابط‌های ابزارهای ارتباطی برای تشخیص صحت اطلاعات ارسالی مورد استفاده قرار می‌گیرد. به مداری که دو مقدار باینری را با هم مقایسه کند و تشخیص دهد کدام مقدار بزرگتر است یا مقادیر برابر یکدیگر هستند مقایسه گر گفته می‌شود [۱].

مقایسه بین دو مقدار باینری در ریزپردازنده‌ها، سیستم‌های ارتباطی، تجهیزات رمزگذاری، شبکه‌های طبقه بندی و... کاربردهای فراوانی دارد. بنابراین مقایسه گر باینری دارای نقش ضروری در VLSI پیشرفته و نانو تکنولوژی می‌باشد [۲].

تحقیقات زیادی برای بهبود ساختار مقایسه گرها در طول سالیان اخیر صورت گرفته است که اکثر آن‌ها بر روی کاهش مساحت، تاخیر و توان مصرفی مقایسه گرها متمرکز بوده اند [۳-۱۱]. برای اولین بار در [۱۲] ساختار جدید مقایسه گر باینری n بیتی برگشت پذیر تحمل پذیر خطا ارائه شد. الگوریتم جدیدی برای این ساختار ارائه گردید که شامل دو گیت برگشت پذیر تحمل پذیر خطا به نام‌های AG و FTSEG به منظور بهینه سازی تعداد گیت‌ها، هزینه کوانتوم، ورودی‌های ثابت و تاخیر مدار است همچنین این دو گیت بهبود تحمل پذیری خطا یا حفظ توازن را نیز به همراه دارند. از AG و FTSEG برای محافظت از بیت توازن ورودی و خروجی استفاده شده است.

یک ساختار تحمل پذیر خطا در [۱۳] برای یک جمع کننده ارائه شده است که با افزودن یک بلوک اضافی و کنترل مطلوب آن توانسته علاوه بر عملکرد صحیح مدار در شرایطی که یکی از بلوک‌های آن دچار خطا می‌شود سربار مساحت و تاخیر را نیز تا حد امکان کاهش دهد.

روشی ارائه شده که برای پیاده سازی سیستم تحمل پذیر خطا از توپولوژی cold-standby استفاده کرده است [۱۴]. یعنی قبل از اینکه سیستم وارد حالت تست و پیکربندی مجدد شود باید عملکرد عادی آن متوقف شود. اما روش ارائه شده در این مقاله hot-standby می‌باشد که این قابلیت را دارد که بدون اعمال وقفه به روند عادی سیستم، آن را تست و خطا را آشکار و پیکربندی مجدد انجام دهد.

ساختار مقاله به شرح ذیل است. مدار پیشنهادی جمع کننده تحمل پذیر خطا در بخش ۲ ارائه می‌شود که در آن روش تحمل پذیری خطا، مدلسازی خطا، تست CSC ها و تولید و پیکربندی مجدد سیگنال‌های کنترل مالتی پلکسرها بررسی شده است و نتایج شبیه سازی و مقایسه در بخش ۳ آورده شده است.

۲- مقایسه گر تحمل پذیر خطا

۲-۱- مقایسه گر

درمقایسهٔ دو بیت، اگر فقط قصد نشان دادن تساوی یا عدم تساوی را داشته باشیم، ساده ترین روش استفاده ازگیت XNOR است. در این مقایسه گر اگر دو بیت مساوی باشند خروجی ۱ و در غیر این صورت خروجی صفر می‌شود. جدول(۱)، جدول صحت گیت XNOR را نشان می‌دهد.

جدول(۱): جدول صحت گیت XNOR

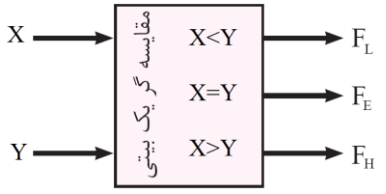
X	Y	F
۰	۰	۱
۰	۱	۰
۱	۰	۰
۱	۱	۱

در مقایسه بین دو بیت، ممکن است بزرگ تر، کوچک تر و مساوی بودن بیتها مورد نظر باشد. جدول(۲) مقایسه بین دو بیت X و Y را نشان می‌دهد.

جدول(۲): مقایسه بین دو بیت X و Y

X	Y	F_L X<Y	F_E X=Y	F_H X>Y
۰	۰	۰	۱	۰
۰	۱	۱	۰	۰
۱	۰	۰	۰	۱
۱	۱	۰	۱	۰

مطابق جدول(۲) سه خروجی داریم. در شکل(۱) بلوک دیاگرام مقایسه‌گر تک بیتی را مشاهده می‌کنید.

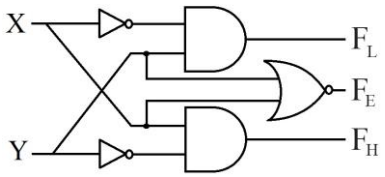


شکل(۱): بلوک دیاگرام مقایسه گر تک بیتی

با توجه به جدول صحت مقایسه گر تک بیتی، عبارت بولی هر یک از خروجی‌های F_L، F_E و F_H را می‌نویسیم:

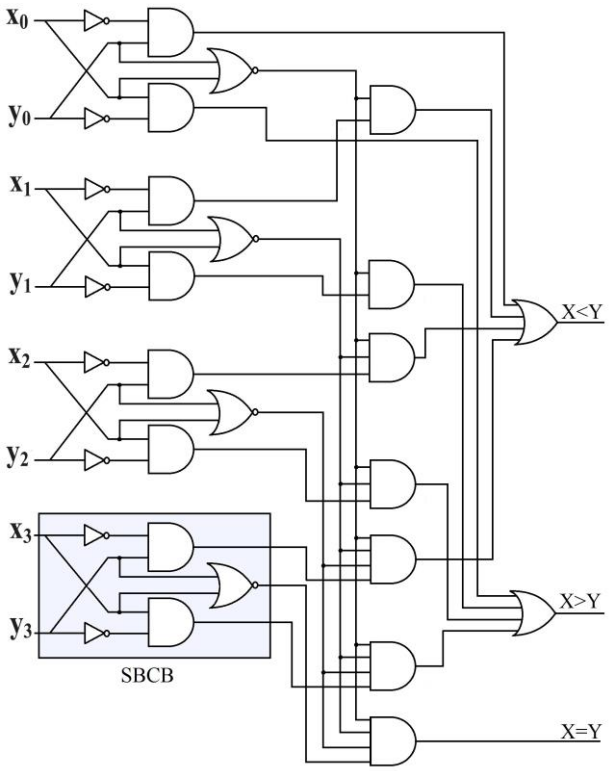
$$F_L = \overline{X}Y$$
$$F_E = \overline{X}\overline{Y} + XY = \overline{X \oplus Y} \qquad (۱)$$
$$F_H = X\overline{Y}$$

با استفاده از توابع به دست آمده از خروجی‌ها می‌توان مدار شکل(۲) را طراحی کرد.



شکل(۲): مدار بلوک مقایسه گر تک بیتی SBCB (Single Bit Comparator Block)

شکل(۳) یک مقایسه گر ۴بیتی که از کنار هم قرار دادن بلوک‌های SBCB تشکیل شده است را نمایش می‌دهد. با استفاده از این معماری آبشاری به ساختاری با آرایه‌های منطقی مکرر[۲] (ILA) و همسان می‌رسیم که توسط آن راحت‌تر می‌توان ساختارC-testable و در نهایت هر مقایسه گر n بیتی را پیاده سازی کرد. با این ساختار منظم، پیاده سازی مقایسه گر با هرتعداد بیت ورودی با تکرار بلوک‌های SBCB و گیت‌های منطقی ساده امکان پذیر می‌شود.



شکل(۳): مقایسه گر ۴ بیتی

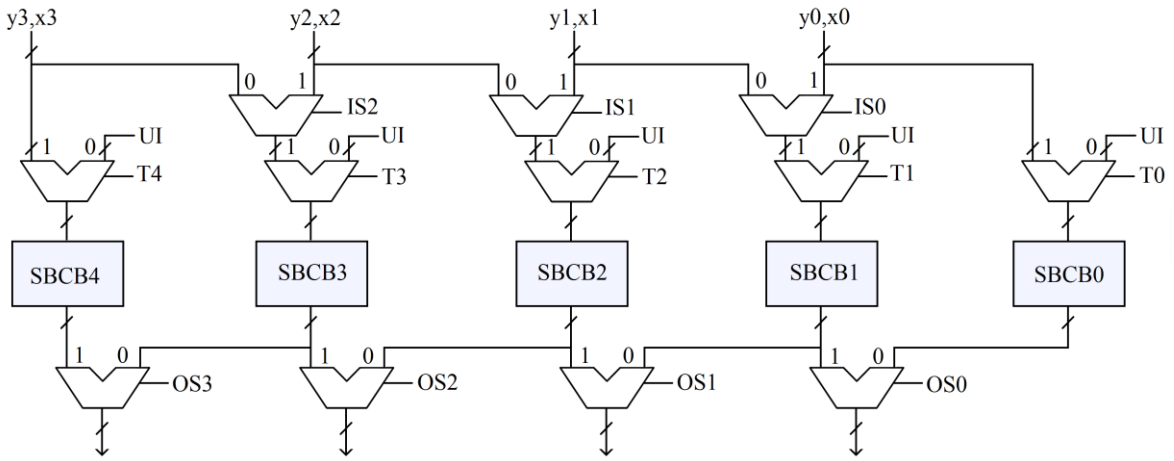
۲-۲- روش تحمل پذیری و مدلسازی خطا

بهترین روش جبران خطا در کاربردهایی که دغدغه اصلی آن‌ها افزایش حداکثر بار[۳] است، روش بازیابی پویا است. برای پیچیدگی کمتر ساختار تست، هر آرایه منطقی مکرر باید C-testable باشد. این در حالی است که الگوهای تست مورد نیاز مستقل از تعداد آرایه‌های منطقی مکرر هستند. برای ساختن یک مقایسه گر C-testable و همچنین روش بازیابی پویا برای ساخت مقایسه گر تحمل پذیر خطا، باید مدار به واحدهای مستقل کوچکتری تبدیل شود. یک مقایسه گر ۴بیتی شامل ۴بلوک SBCB و گیت‌های منطقی سطوح بعدی آن می‌باشد. برای داشتن یک مقایسه گر تحمل پذیر خطا یک بلوک SBCB اضافی[۴] با کارکرد فعال به ۴بلوک SBCB مدار اضافه می‌شود. بلوک‌های SBCB از یکدیگر مستقل هستند و ساختار C-testable باعث شده است که الگوهای تست هم بصورت مستقل به بلوک‌های SBCB اعمال شوند و اگر یکی از بلوک‌های SBCB دچار مشکل شد مقایسه گر با ۴بلوک SBCB باقیمانده به عملکرد بدون خطای خود ادامه دهد.

سیستم تحمل پذیر خطا ارائه شده قادر به تحمل تمامی خطاها از جمله stuck-at، stuck-open، stuck-close و خطاهای عملکردی به شرطی که در یک زمان فقط برای یک بلوک SBCB اتفاق بیافتد می‌باشد. علاوه بر این خطای پل که به علت اتصال کوتاه بین دو نقطه از مدار رخ می‌دهد در این نمونه قابل تحمل است.

ساختار بلوک‌های SBCB تحمل پذیر خطا با تمام ورودی‌ها و خروجی‌ها در شکل(۴) نشان داده شده است. دو سطح از مالتی پلکسرها در طرف ورودی وجود دارند. یک سطح برای مسیر یابی صحیح ورودی به بلوک‌های SBCB با انتخاب درست سیگنال‌های ISi و سطح دیگر برای انتخاب حالت تست توسط سیگنال Ti برای بلوک SBCBi می‌باشد. زمانیکه سیگنال Ti صفر باشد بردار تست Ui به بلوک SBCBi وارد شده و خروجی‌های تولید شده در هر پالس ساعت با خروجی‌های مطلوب محاسبه شده با یک مداربندی ساده یا ذخیره شده در جداول مراجعه ای[۵] مقایسه می‌شوند. اگر در یک SBCB خطایی رخ دهد علائم آن در خروجی همان ورودی تست معین ظاهر می‌شود و خروجی مقایسه گر یک می‌شود و مشخص می‌کند SBCBi معیوب است. سپس عملیات تست بلوک‌های SBCB بعدی متوقف و مقدار سیگنال T در همین حالت باقی می‌ماند و جمع کننده با چهار SBCB بدون خطای باقیمانده به کار عادی خود ادامه می‌دهد.

بطور کلی در روش‌های بازیابی پویا به علت زمان تست و پیکربندی مجدد پیچیدگی زمانی[۶] بیشتر است. در [۱۵] هنگامیکه تایمر واچداگ[۷] یک ناهنجاری و عمل غیرعادی را تشخیص دهد سیستم وارد حالت تست و پیکربندی مجدد می‌شود. اما در این مقاله از روش hot-standby استفاده شده که به سیستم اجازه تست و پیکربندی مجدد را بدون اعمال وقفه در عملکرد عادی مقایسه گر می‌دهد که نهایتا منجر به اضافه نشدن زمان اضافه به سیستم می‌شود. عملیات تست در بین بلوک‌های SBCB اصلی به ترتیب و بصورت

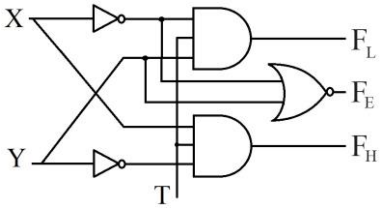


شکل(۴): بلوک‌های SBCB تحمل پذیر خطا

چرخشی انجام می‌شود و SBCB یدکی تا زمانیکه یکی از بلوک‌های SBCB معیوب نشود وارد مدار نمی‌شود.

۲-۳- تست بلوک‌های SBCB

برای کاهش تاخیر سیستم، مالتی پلکسرهای سطح دوم با بلوک‌های SBCB ادغام شده اند و ساختار SBCB با اضافه کردن یک سیگنال ورودی T (Ti برای بلوک SBCBi) بهبود یافته است(شکل(۵)).



شکل(۵): بلوک SBCB اصلاح شده

زمانیکه Ti یک باشد SBCBi در حالت نرمال خود کار می‌کند و اگر Ti صفر شود به حالت تست تغییر وضعیت می‌دهد. جدول صحت SBCB بهبود یافته با T=1 و T=0 در جدول(۳) نشان داده شده است.

جدول(۳): جدول صحت SBCB بهبود یافته با T=1 و T=0

		T=1			T=0		
Xi	Yi	F_L	F_E	F_H	F_L	F_E	F_H
۰	۰	۰	۱	۰	۰	۱	۰
۰	۱	۱	۰	۰	۰	۰	۰
۱	۰	۰	۰	۱	۰	۰	۰
۱	۱	۰	۱	۰	۰	۱	۰

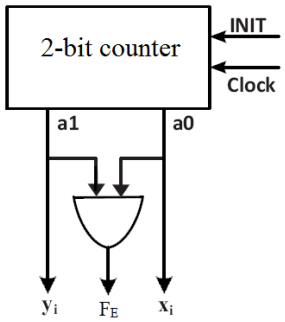
با یک مرتب سازی مناسب الگوهای ورودی، الگوهای تستی که برای بلوک CSC طراحی کردیم در جدول(۴) آمده است که فقط ۳ الگوی تست برای تست بلوک SBCB کافی است.

جدول(۴): الگوی تست برای SBCB اصلاح شده و خروجی‌های آن (T=0)

بردار تست	الگوی تست		خروجی مطلوب		
	Yi	Xi	F_L	F_E	F_H
۰	۰	۱	۰	۰	۰
۱	۱	۱	۰	۱	۰
۲	۱	۰	۰	۰	۰

تست بلوک‌های SBCB از بیت‌های با ارزش شروع و به سمت بیت‌های کم ارزش تر حرکت می‌کند. از این رو اول ورودی تست T4 برای SBCB4 در با ارزشترین بیت مقدار صفر و بقیه Ti ها یک می‌شوند و تست بلوک SBCB4 در با ارزشترین بیت شروع می‌شود. اگر بلوک SBCB4 معیوب باشد عملیات تست متوقف و مقدار سیگنال T4 در همین حالت با مقدار صفر باقی می‌ماند تا مسیریابی ورودی‌ها به دیگر بلوک‌های SBCB که بدون خطا کار می‌کنند بصورت مطلوب انجام گیرد. اما اگر SBCB4 معیوب نباشد ورودی تست به بلوک بعدی یعنی SBCB3 اعمال می‌شود و T3 صفر و T4 یک می‌شود. اگر بلوک SBCB3 معیوب باشد عملیات تست متوقف و جمع کننده به کار عادی خود با چهار SBCB بدون خطا باقیمانده ادامه می‌دهد و اگر SBCB3 معیوب نباشد ورودی تست به ترتیب به بلوک‌های بعدی SBCB2، SBCB1 و SBCB0 اعمال می‌شود.

یک شمارنده دو بیتی به همراه تعدادی گیت ترکیبی برای ساختن الگوهای تست مطلوب برای تست بلوک‌های SBCB بهبودیافته مورد استفاده قرار گرفته که در شکل(۶) نشان داده شده است. همین مداربندی برای تولید خروجی‌های مطلوب الگوهای تست اعمال شده به مدار مورد استفاده قرار گرفته است.



شکل(۶) : تولیدکننده الگوهای تست و خروجی‌های مطلوب

جدول(۵): سیگنال‌های کنترل مالتی پلکسرها (IS و OS) و سیگنال کنترل تست T

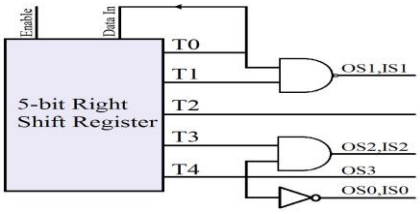
T4	T3	T2	T1	T0	IS2	IS1	IS0	OS3	OS2	OS1	OS0
۰	۱	۱	۱	۱	۰	۰	۰	۰	۰	۰	۰
۱	۰	۱	۱	۱	X	۰	۰	۱	۰	۰	۰
۱	۱	۰	۱	۱	۱	X	۰	۱	۱	۰	۰
۱	۱	۱	۰	۱	۱	۱	X	۱	۱	۱	۰
۱	۱	۱	۱	۰	۱	۱	۱	۱	۱	۱	۱
					T4.T3	$\overline{T0.T1}$	$\overline{T4}$	T4	Is2	Is1	Is0

۲-۴- تولید و پیکربندی مجدد سیگنال‌های کنترل مالتی پلکسرها

برای پیکربندی مطلوب سیگنال‌های ورودی و خروجی در آرایه‌های SBCB، سیگنال‌های کنترل مالتی پلکسرها باید بصورت صحیح با تغییر در ورودی‌های Ti تولید شوند. به نحوی که مقایسه گر بدون مختل شدن روند عملکرد اصلی حتی در صورت بروز خطا، عملکرد یکنواختی داشته باشد. جدول(۵) تغییرات مقادیر سیگنال‌های کنترل ورودی IS و کنترل خروجی OS مالتی پلکسرها را با توجه به سیگنال کنترل تست T نشان می‌دهد.

در ابتدا بلوک‌های SBCB0 تا SBCB3 فعال هستند و SBCB4 در حالت تست قرار دارد. سپس SBCB3 وارد حالت تست شده، ورودی‌های جمع کننده وارد آن نمی‌شوند و خروجی‌ها با صفر شدن OS3 به جای SBCB3 از SBCB4 گرفته می‌شوند. برای تست SBCB2 تمام ورودی‌های آن با یک شدن IS2 به SBCB3 منتقل می‌شوند و در نهایت خروجی‌ها برای عملکرد نرمال جمع کننده از بلوک‌های SBCB فعال گرفته می‌شوند. اگر در یکی از بلوک‌های SBCB خطایی کشف شود همه سیگنال‌های کنترلی مقادیر خود را ثابت نگه می‌دارند. ورودی‌ها به بلوک‌های بدون خطا منتقل می‌شوند و خروجی‌ها نیز از همان بلوک‌ها گرفته می‌شوند تا مقایسه گر به عملکرد عادی خود ادامه دهد. در صورتیکه خطایی در سیستم کشف نشود عملیات تست پس از یک دوره کامل متوقف می‌شود و این عملیات تا زمانیکه سیستم بدون خطا به روند عادی خود ادامه دهد تا پیکربندی مجدد سیستم بصورت دوره ای تکرار می‌شود. چنین روندی از پیکربندی مجدد که سیستم بدون وقفه ای در عملکرد عادی از خطا بازیابی می‌شود مبادله داغ[۸] نام دارد. در این روش عملیات تست، پیکربندی مجدد و عملکرد عادی سیستم همزمان انجام می‌شوند.

جدول(۵) نشان می‌دهد سیگنال‌های ISi و OSi مقادیر یکسان دارند پس می‌توان از یک سخت افزار یکسان برای تولید آن‌ها بهره جست. Ti ها با استفاده از یک شیفت رجیستر ۵ بیتی تولید می‌شوند و سیگنال‌های ISi و OSi را نیز می‌توان با استفاده از Ti تولید کرد(شکل(۷)). برای سیگنال‌های کنترل مالتی پلکسرها در طراحی‌های مشابه بزرگتر نیز می‌توان از چنین مدار تولید کننده ای استفاده کرد.



شکل(۷): تولید کننده سیگنال‌های کنترل مالتی پلکسرها و سیگنال کنترل تست T

۲-۵- محاسبه مساحت

برای محاسبه و مقایسه پیچیدگی‌های سخت افزاری طراحی، سربار مساحت را بصورت زیر تعریف می‌کنیم

$$AO = \frac{Area_{FT} - Area}{Area} \times 100\% \qquad (۲)$$

در این رابطه Area مساحت مدار اصلی(غیر تحمل پذیر خطا) و $Area_{FT}$ مساحت مدار تحمل پذیر خطا است. مساحت یک مقایسه گر ۴بیتی برای داشتن مساحت پایه و برای راحتتر شدن محاسبات از شکل(۳) محاسبه شده است. حال مساحت یک مقایسه گر تحمل پذیر خطا n بیتی با یک SBCB یدکی بصورت زیر محاسبه می‌شود

$$Area_{C_n_FT} = Area_{C_n} + Area_{SBCB} + Area_{TR} \qquad (۳)$$

که در آن $Area_{C_n_FT}$ مساحت مدار مقایسه گر n بیتی تحمل پذیر خطا، $Area_{C_n}$ مساحت مقایسه گر n بیتی، $Area_{SBCB}$ مساحت SBCB یدکی و $Area_{TR}$ مساحت مداراتی است که برای کنترل عملیات تست و پیکربندی مجدد مورد استفاده قرار گرفته اند. از این رو برآورد تقریبی برای محاسبه سربار مساحت یک مقایسه گر تحمل‌پذیر خطا n بیتی با رابطه زیر مشخص می‌شود

$$AO = \frac{Area_{FT} - Area}{Area} \times 100\%$$

$$AO = \frac{Area_{C_n_FT} - Area_{C_n}}{Area_{C_n}} \times 100\% \qquad (۴)$$

$$AO = \frac{Area_{SBCB} + (10n-4)Area_{MUX} + Area_{TR}}{Area_{C_n}} \times 100\%$$

۳- نتایج شبیه سازی

ساختار تحمل پذیر خطای پیشنهادی در تکنولوژی UMC 180 nm نرم افزار Synopsis design_vision برای یافتن مساحت و تاخیر پیاده سازی شد. همانطور که در جدول(۶) نشان داده شده است مقایسه گر تحمل پذیر نسبت به مقایسه گر معمول دارای افزایش مساحت و تاخیر به ترتیب ۱۵۶/۳٪ و ۴۱٪ می‌باشد.

جدول(۶): سربار مساحت و تاخیر در مقایسه گر معمول و تحمل پذیر خطا

سربار تاخیر (٪)	تاخیر (ps)	سربار مساحت (٪)	مساحت (µm2)	نوع مقایسه گر
--	۴۸۵/۶۵	--	۳۸۵/۲	مقایسه گر معمول
۴۱	۶۸۴/۷۷	۱۵۶/۳	۹۸۷/۲۷	مقایسه گر تحمل پذیر خطا

از آنجا که کارهای صورت گرفته در سال‌های اخیر در حوزه مقایسه گر بیشتر برروی مساحت، تاخیر و توان مصرفی متمرکز بوده‌اند مقایسه ای بین مدار پیشنهاد شده و دیگر مدارات تحمل پذیر خطا در جدول(۷) نشان داده شده است.

یک واحد تولید کننده سیگنال کنترل مالتی پلکسرها کافی است و می‌تواند بین واحدهای مختلف تحمل پذیر خطا برای کاهش سربار مساحت بصورت اشتراکی استفاده شود. همچنین یک واحد تولیدکننده الگوی تست برای تست همه بلوک‌های SBCB مورد نیاز است. فرض ما بر این است که مدارات کنترلی بدون خطا هستند. برای داشتن ویژگی‌های تحمل پذیر خطا در سیستم، باید از شمارنده‌ها، مالتی پلکسرها و گیت‌های صرفا تحمل پذیر خطا استفاده کنیم [۱۶].

جدول(۷): مقایسه سربار مساحت و تاخیر در مدارات مختلف تحمل‌پذیر خطا

سربار تاخیر (٪)	سربار مساحت (٪)	مدارات مختلف
۲۳/۵۳	۲۴۴/۸	[17]
۷۸/۴۲	۱۸۶/۲۶	RCA[18]
۵۰/۷۳	۱۷۸/۱۷	CSA[18]
۴۱	۱۵۶/۳	مقایسه گر تحمل پذیر خطا

۴- نتیجه‌گیری

روش جدیدی به منظور بهینه سازی تحمل پذیری خطا در مدار مقایسه گر ارائه شد. در این روش با از کار افتادن یکی از بلوک‌های SBCB، که در حالت عادی موجب از کار افتادگی کل سیستم می‌شود، روند عادی سیستم مختل نمی‌شود و می‌تواند اطلاعات از دست رفته را بازیابی نماید و این در حالی است که سیستم از بازیابی پویا و روش hot-standby بهره می‌برد و به همین منظور عملیات تست، بدون وقفه در عملکرد عادی سیستم صورت می‌پذیرد.

مراجع

[1] Jaiswal, S. K., Verma, K., Verma, K., Pratihar, N., "Design of CMOS 8-BIT Comparator for Low Power Application", Fourth International Conference on Computational Intelligence and Communication Networks, pp. 480 – 482, 2012.

[2] Sun, Y., Ye, Y., Lai, F., "Power Efficient High Speed Switched Current Comparator", 7th International Conference on ASIC,ASICON '07, pp.581 – 584, 2007.

[3] Belayet Ali, Md., Mizanur Rahman, Md., Rahman, H. A., "Design and Optimization of Nanometric Reversible 4 Bit Numerical Comparator", International Conference on Informatics, Electronics & Vision (ICIEV), pp.577 – 581, 2012.

[4] Khavari A F, Mafinezhad K, Maymandi Nejad M. A, "Broadband Low Power CMOS LNA for 3.1–10.6 GHz UWB Receivers", Journal of Iranian Association of Electrical and Electronics Engineers, Vol. 14, No. 4, pp. 1-13, 2018.

[5] Babayan-Mashhadi, S., Lotfi, R., "Analysis and Design of a Low-Voltage Low-Power Double-Tail Comparator" IEEE Transactions on Very Larg Scale Integration (VLSI) Systems, Vol. 22, No. 2, pp.343 – 352, 2014.

[6] Prajapat, G., Joshi, A., Jain, A., Verma, K., Sanjay Kr. Jaiswal, "Design of Low power and high speed 4-bit Comparator using Transmission Gate" International Conference on Machine Intelligence Research and Advancement, pp.379 – 382, 2013.

[7] Sharma, A., Sharma, P., "Area and Power Efficient 4 – Bit Comparator Design by Using 1- Bit Full Adder Module" International Conference on Parallel, Distributed and Grid Computing, pp.1 – 6, 2014.

[8] Abdul Halim, I. S., Abidin, N. A., Rahim, A., " Low Power CMOS Charge Sharing Dynamic Latch Comparator using 0.18µm Technology", IEEE Regional Symposium on Micro and Nanoelectronics (RSM), pp.156 – 160, 2011.

[9] Soni, U., Vidyarthi, A., Akashe, S., "Design of High Speed and leakage Tolerant CMOS Comparator in UDSM Technology" Third International Conference on Advanced Computing & Communication Technologies, pp.326 – 329, 2013.

[10] Mehri H, Alizadeh B., "Analytical Performance Model for FPGA-based Reconfigurable Computing", Journal of Iranian Association of Electrical and Electronics Engineers, Vol. 13, No. 4, pp. 1-13, 2017.

[11] Chuang, P. I., Sachdev, M., Gaudet, V. C., "A 167-ps 2.34-mW Single-Cycle 64-Bit Binary Tree Comparator With Constant-Delay Logic in 65-nm CMOS", IEEE Transactions on Circuits and Systems—I: Regular Papers, Vol. 61, No. 1, pp.160 – 171, 2014.

[12] Avishek Bose, Ankur Sarker, "A Novel Approach for Constructing Reversible Fault Tolerant n-Bit Binary Comparator", 3rd International Conference on Informatics, Electronics & Vision, pp.1–6, 2014.

[13] Mukherjee, A., Dhar, A. S., "Real-time fault-tolerance with hot-standby topology for conditional sum Adder", Microelectronics Reliability, Vol. 55, Issues 3–4, pp. 704–712, 2015.

[14] Mukherjee A., Dhar AS. "Design of a fault-tolerant conditional sum adder". In: Progress in VLSI design and test (VDAT), LNCS 7373. Springer Berlin Heidelberg; pp. 217–222, 2012.

[15] Parhi, R., Chris, H., Kim, Keshab, K., Parhi, "Fault-Tolerant Ripple-Carry Binary Adder using Partial Triple Modular Redundancy (PTMR) ", IEEE International Symposium on Circuits and Systems (ISCAS), pp.41– 44, 2015.

[16] Anghel, L., Nicolaidis, M., Defects tolerant logic gates for unreliable future nanotechnologies. In: Int. work-Conf. artif. neural networks, San Sebastin, Spain; p. 422–429, 2007.

[17] Teifel, J., "Self-Voting Dual-Modular-Redundancy Circuits for Single-Event-Transient Mitigation", IEEE Transactions on Nuclear Science, Vol. 55 , No: 6 , pp.3435 – 3439, 2009.

[18] Mukherjee, A., Dhar,A. S., "Double-Fault Tolerant Architecture Design for Digital Adder", IEEE, Students' Technology Symposium (TechSym) ,pp. 154 – 158, 2014.

زیرنویس‌ها

[1] Fault Tolerant
[2] Iterative Logic Array
[3] payload
[4] hot spare
[5] look-up table
[6] time complexity
[7] watchdog timer
[8] hot-swapping