بخش اول

معرفی

تلاش برای دستیابی به افزایش قابلیت اطمینان طراحی های سخت افزاری همواره در حال رشد بوده است.

از آنجایی که طراحی و ساخت مدار نیازمند شرکت افراد مختلف می باشد تضمین این واقعیت که سخت افزار مدار دچار دستکاری نشود بسیار بعید می باشد.

می توان گفت که امنیت سخت افزار بطور رسمی بعد از ظهور پدیده تروجان [1]های سخت افزاری معرفی شد.

بسیاری از تکنیک های شناسایی تروجان [2]ها در مراحل بعد از ساخت سیلیکونی معرفی شدند.

که تاکید بیشتر این روش ها بر روی تست و تکرار آن در همان مراحل بعد ساخت می باشد.

اما بجای شناسایی این گونه تهدیدات در مدار های آلوده طبق دستورالعمل[3] بهتر است از ورود این گونه مدار ها در دستگاه مربوطه پیشگیری شود چرا که باعث افزایش قابلیت اطمینان و امنیت میشود.

که به همین منظور در جریان طراحی یا design flow روش ها و متد های گوناگونی پیشنهاد شد[4].

به علاوه جلوگیری از دسترسی غیر مجاز افراد مختلف به داده های مربوط به IC۱ نیز امری بسیار ضروری محسوب میشود.

در نتیجه کلیدی برای دسترسی به داده ها تعبیه شد, تنها افرادی قادر به دسترسی این داده ها هستند که این کلید را در اختیار دارند.

این مفهوم در قالب مراحل توسعه دستگاه با نام Physical Random Function معرفی شد[4].

این مفهوم بعد ها در قالب PUF ۱ تکامل پیدا کرد[5].

PUF ها به گونه ای توسعه پیدا کردند که بتوان از وجود بخش های مختلف سخت افزاری در یک مدار بیشترین بهره وری را داشت.

این بخش های مختلف سخت افزاری میتواند شامل ویژگی های سخت افزاری , تاخیر گیت ها , آستانه ی ولتاژ , ویژگی های الکتریکی ) مقاومت , ظرفیت , اندوکتانس( و ویژگی های نوری و بصری بشود.

ممکن است تفاوت هایی در این بخش ها هنگام ساخت سخت افزار رخ بدهد[6].

اینگونه تفاوت ها نمیتوانند برای دو دستگاه یکسان باشند مانند اثر انگشت انسان ها , پس PUF ها تمرکز خود را به گونه ای معطوف این امر کرده اند که بتوانند این تفاوت ها را شناسایی و حتی در مواردی آن ها را تشدید کنند.

این کارکرد به صورت یک تابع با عنوان PUF به مدار مربوطه اضافه شد.

PUF ها در شاخه های گوناگونی بر اساس ساختار فیزیکی آن ها ) سیلیکونی بودن یا نبود ( , توپولوژی و ... دسته بندی شده اند.

میتوان PUF ها را نیز بر اساس قوی یا ضعیف بودن تقسیم کرد

PUF های قوی دارای CRP۱ های بیشتری هستند و بیشتر برای عملیات شناسایی و احراز هویت بکار میروند.

در حالیکه PUF های ضعیف بیشتر در جهت تولید کلید بکار میروند.

بر اساس ساختار, PUF های سیلیکونی را میتوان در قابلیت هایی مانند تاخیری , حافظه ای و آنالوگ دسته بندی کرد.

در طرف مقابل PUF های غیر سیلیکونی را میتوان در قابلیت هایی مانند نوری , مغناطیسی و آکوستیک را دسته بندی کرد[7].

این ساختار ها توسط نرم افزار ها پیادی سازی شده اند. همچنین ساختار های جدایی برای FPGA[8] و ASIC [9] ها ارائه شده است.

پیاده سازی PUF بر روی FPGA خود نیز دارای چالش های فراوانی میباشد.

باید مراقب باشیم تا از تاثیر مسیریابی۱ یکنواخت جلوگیری شود تا خللی در فرآیند پیاده سازی PUF ایجاد نشود.

در بین این ساختار های معرفی شده PUF های که ساختار آن ها بر اساس تاخیر یا delay میباشند نسبتا قدرتمندتر میباشند .

ازین ساختار ها میتوان به FF APUF[12] , FF PUF[11], APUF[10] اشاره کرد که به توضیح آن ها خواهیم پرداخت.

عملکرد این ساختار ها بر اساس معیار هایی مانند یکتایی یا Uniqueness و قابلیت اطمینان آن ها محاسبه می شود.

همچنین با هم ساختار XOR FF APUF را نیز بر روی خانواده ای از FPGA ها بررسی میکنیم.

بخش دوم

ساختار PUF ارائه شده جهت مقایسه

A.Arbiter PUF

APUF نشان داده شده در شکل ۱ دارای دو خط تاخیر می باشد.

که این دو خط با مولتی پلاکسر کنترل می شوند و انتخاب این دو خط تاخیر وابسته به بیت اعمال شده می باشد.درحالت ایده آل برای N بیت ورودی در PUF به N مرحله نیاز است.که هر مرحله شامل دو واحد مالتی پلاکسر می شود.

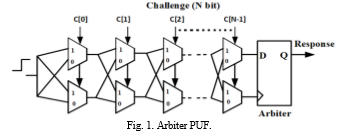
سپس بیت ورودی انتخاب میکند که کدام خط تاخیر در نظر گرفته شود و در نهایت Arbiter موجود در مرحله پایانی سریع ترین خط را انتخاب می کند.

در فاز شبیه سازی این دو خط تاخیر قابل شناسایی اند و هرکدام از آن ها در یک لحظه زمانی پاسخ را ارائه میکنند.

در زمان ساخت دستگاه ممکن است برخی از اختلالات رخ بدهد که در نتیجه این اختلالات تضمین این حقیقت که دریافت سیگنال از این دو خط در یک زمان باشد بسیار بعید است.

PUF های ترکیبی میتوانند از CRP های منحصر به فردی که برای شناسایی هر دستگاه وجود دارد استفاده کرده و نقش بسزایی در تسهیل فرآیند شناسایی این اختلالات حاصل از مرحله ساخت را داشته باشند.

از APUF ها در شناسایی (identification) دستگاه های مختلف استفاده میشود.



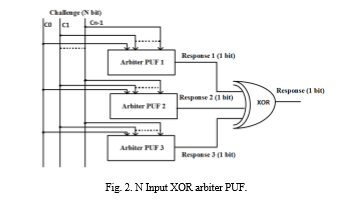
B. N Input XOR Arbiter PUF

در این ساختار, برای اینکه یک بیت خروجی به عنوان پاسخ داشته باشیم ,خروجی های APUF را میتوان با هم XOR کرد. برای ایجاد دگرگونی بیشتر در جهت افزایش یکتایی میتوان ۳ PUF را با هم در نظر گرفت و خروجی آن ها را XOR کرد.

ار مطالعات حاصل از [7] میتوان نتیجه گرفت که XOR کردن تعدادیPUF موجب افزایش خطرات ناشی از Machine Learning میشود.

XOR APUF ارائه شده در [4] از دسترسی غیرمجاز افراد جلوگیری میکند.

در شکل ۲ XOR APUF با N بیت ورودی نشان دادهشده است.



C. FF APUF

در FF APUF ارائه شده در[12] فلیپ فلاپ های نوع D به همراه مالتی پلاکسر نیز به ساختار مربوطه اضافه شده است.

بخاطر این امر مشاهده میشود که حتی دگرگونی بیشتری در هر مرحله ایجاد میشود.

این ساختار بسیار انعطاف پذیر میباشد که در نتیجه آن میتوان به تعداد متعددی از CRP ها تکثیر شود.

این FF APUF شامل دو نوع معماری میشود.

نوع ۱:

در این نوع معماری تمام خروجی های فلیپ فلاپ نوع D به یک خروجی از طریق انتخاب بیت مورد نظر و با کمک تعدادی مالتی پلاکسر تبدیل میشود.

در شکل ۳ FF APUF نوع یک دارای ۴ فلیپ فلاپ نوع D و ۳ واحد مالتی پلاکسر میشود.

در حالی که نوع دوم نمونه ای کوچکتر از نوع اول میباشد که شامل ۲ فلیپ فلاپ نوع D و ۱ واحد مالتی پلاکسر میشود.

بلوک نقطه چین در شکل ۳ نشان دهنده معماری نوع ۲ میباشد.

این دو نوع معماری ساختار های پایه و اولیه برای طراحی یک FF APUF برای تعداد متفاوتی از CRP ها میباشند.

برای مثال برای ساخت یک FF APUF ۸ بیتی به ۴ مرحله نیاز است.

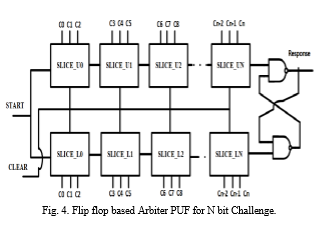
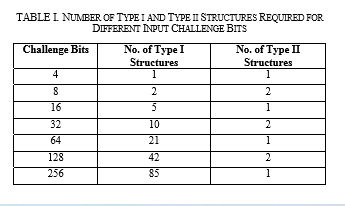
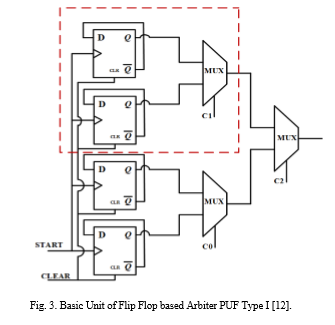
که دو مرحله اول شامل معماری نوع ۱ و دو مرحله نهای شامل معماری نوع ۲ میشود بطور کلی برای ساخت N بیت FF APUF میتوان از شکل ۴ استفاده کرد.

در شکل ۴

Slice\_U نشان دهنده تراشه های بالای خط تاخیر و Slice\_L نشان دهنده تراشه های پایین خط تاخیر میباشند.

هر دو نوع تراشه ها شامل معماری هایی میباشند که در شکل ۳ معرفی شدند.

هر دو نوع U و L میتواند شامل معماری نوع ۱ یا ۲ باشد, در جدول ۱ تعداد معماری های لازم از نوع ۱ و۲ را برای بیت های ورودی متفاوت فراهم کرده است.



بخش ۳

ارائه یک ساختار PUF

این ساختار ارائه شده XOR FF APUF می باشد که حاصل ترکیب FF APUF وXOR APUF که در بخش های قبل مطرح شد میشود.

این نوع ساختار از XOR کردن ۳ تا FF APUF بدست میآید.

این ساختار تاثیر بسزایی در بهبود پایداری حملات Machine Learning دارد.

XOR FF APUF که ارائه کردیم قابلیت پاسخ گویی به ۸ الی ۱۶ بیت ورودی (CRP) را دارد.

۳ نوع از ساختار FF APUF ارائه شده در شکل ۴ باهم XOR شده اند تا بتوانیم XOR FF APUF شکل ۵ را بسازیم. لازم به ذکر است که ما یک بیت خروجی به عنوان پاسخ داریم که حاصل XOR کردن ۳ خروجی FF APUF می باشد.

در شکل ۵ ساختار قرار داده شده در نقطه چین نشان دهنده FF APUF ها قبل از مرحله XOR کردن میباشد که هرکدام از این FF APUF شامل دو خط تاخیر موازی از مجموع ۶ خط میشوند.

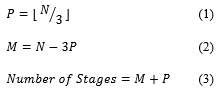
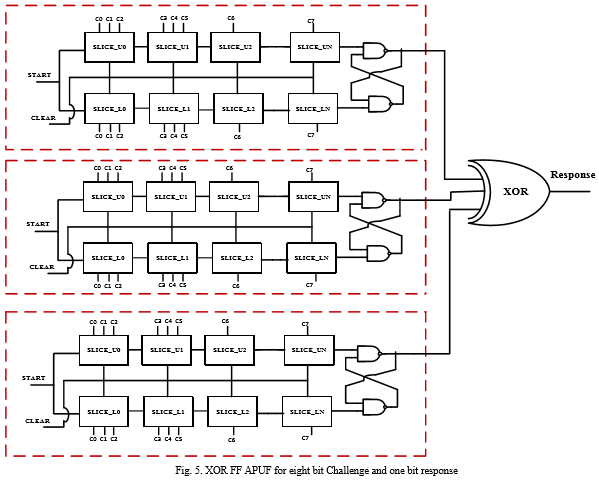
مهم ترین شاخص برای اندازه گیری این ساختار تعداد مراحل(STAGE) میباشد که تعداد این مراحل به بیت های ورودی وابسته است.

منظور از تعداد مراحل, تراشه های موجود در اولین خط تاخیر موازی میباشد.

تعداد PUF های که باهم XOR میشوند ممکن است وابسته به میزان یکتایی مورد نظر و منابع موجود در FPGA و یا مساحت(Area) موجود در ASIC باشند.

در این پروسه مورد بحث ۳ FF APUF باهم XOR شده اند که سیگنال شروع به تراشه U0 داده شده است لازم به ذکر است که معماری این تراشه از نوع ۱ می باشد و خروجی این تراشه به عنوان یک سیگنال شروع به تراشه بعدی داده میشود و این روال به همین گونه ادامه پیدا میکند .

البته انتخاب اینکه چه نوع معماری ای به هر تراشه داده شود وابسته به تعداد بیت های ورودی میباشد که برای محاسبه آن میتوان مانند فرمول های ذیل عمل کرد.



N نشان دهنده بیت های ورودی اعمال شده میباشد.

P نشان دهنده معماری نوع ۱ لازم برای ساخت یک FF APUF می باشد.

M نشان دهنده تعداد معماری نوع ۲ لازم میباشد.

A.Design Of Parrallel Paths

طراحی خطوط موازی

از آنجایی که خطوط تاخیر در FF APUF دارای ساختار یکسانی هستند, میتوان با محاسبه پارامتر های ورودی به یک خط نتایج و ویژگی های آن را به سایر خطوط نیز نسبت داد.

برای N=8 بیت داریم P=2 در نتیجه M برابر 2 میشود و تعداد مراحل برابر ۴ میشود.

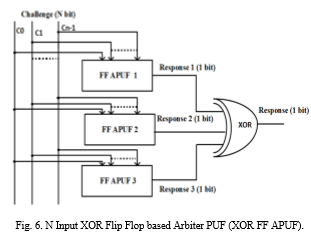
تعداد معماری نوع ۱ و ۲ هر دو ۲ واحد میباشد.

۳ بیت اول به تراشه ی اول با معماری نوع ۱ و ۳ بیت دوم به تراشه ی دوم با همین معماری داده میشود و ۲ بیت باقی مانده به دو تراشه ی نوع ۲ داده میشود(هرکدام یک بیت از دو بیت).

در شکل ۵ تراشه ی U1 و U0 از نوع ۱ و U2وU3 از نوع ۲ میباشد.

برای ۱۶ بیت ورودی P برار پنج و M برابر یک میباشد و تعداد مراحل ۶ خواهد بود که میتوان با این داده ها ۱۶ بیت XOR FF APUF طراحی کرد.

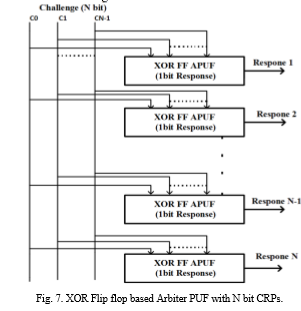
با استفاده از فرمول های ارائه شده و محاسبه معماری نوع ۱ و۲ میتوان XOR FF APUF را بر اساس شکل ۶ درست کرد.



دلیل XOR کردن خروجی های FF APUF به این خاطر است که احتمال رخداد بین ۱و۰ به یک میزان میباشد با توجه به وجود گیت XOR

این امر سبب افزایش قدرت دگرگونی لازم برای یکتایی مورد نظر با افزایش بیت ورودی میشود.

برای گرفتن N بیت جواب به عنوان خروجی میتوان مانند شکل ۷ عمل کرد.



بخش ۴

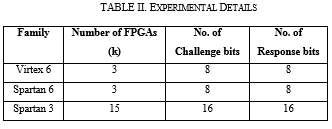
پیاده سازی و روش های تست

XOR FF APUF ارائه شده در بخش قبل بر روی برد های اسپارتان ۶ , ورتکس ۶ , و اسپارتان ۳ در ISE مربوط به Xilinx پیاده سازی شدند.

برای اینکه بتوان مقایسه دقیق و کاملی داشته باشیم FF APUF که در [10] اشاره شده است بر روی این ۳ برد نیز نصب شده است.

از آنجایی که تاثیر XOR APUF و APUF در میزان یکتایی چیزی کمتر از 9% می باشد در مقایسه پیشه رو در نظر گرفته نشده اند.

XOR FF APUF برای CRP های ۸ تا ۱۶ بیت تست شدند و جدول شماره ۲ نتایج حاصل از مقایسه بین این ۳ خانواده از FPGA را نشان میدهد.



همانطور که مشاهده میشود , ورتکس ۶ و اسپارتان ۶ قادرند تا تاخیرات مختلفی را شناسایی کنند اما از آنجایی که که اسپارتان ۳ قادر نیست با ۸ بیت CRP تاخیرات را شناسایی کند در نتیجه این امر برای FPGA هایی با فرکانس کم مانند اسپارتان ۳ (280Hz) بیت های CRP به ۱۶ بیت افزایش پیدا کردند.

برای اینکه بتوان این مقایسه را کامل تر کرد میتوان مرحله جاسازی یا Placement را بصورت دستی انجام داد.

که این فرآیند در ورتکس ۶ و اسپارتان ۶ شامل ۴ فلیپ فلاپ و ۳ واحد مالتی پلاکسر متصل بهم در یک تراشه میشود.

میتوان واحد منطق هر واحد مالتی پلاکسر را به مالتی پلاکسر حاضر در FPGA مربوطه متصل کرد اما در این آزمایش این واحد ها به LUT های حاضر در هر تراشه متصل میشود.

برای هر اسپارتان ۳ دو واحد فلیپ فلاپ و یک واحد مالتی پلاکسر در هر تراشه متصل شده اند.

میزان یکتایی , قابلیت اطمینان و یکنواختی به عنوان معیارهایی برای مقایسه این ۳ برد قرار گرفته اند.

B.Methodology

روش های تست

با استفاده از ابزار ChipScope شرکت Xilinx [14] می توان نتایج این آزمایش را آنالیز کرد.

این ابزار شامل یک آنالیزور منطقی یکپارچه(ILA) , هسته کنترل یکپارچه(ICON) و واحد ورودی و خروجی مجازی(VIO) میباشد.

از ILA برای تحلیل و کاوش ورودی و خروجی و حتی سیگنال های داخلی نیز بکار گرفته میشود.

و از ICON نیز برای کنترل ILA و VIO استفاده میشود.

VIO نیز برای اضافه کردن ورودی ها و خروجی ها بصورت مجازی به منابع یک HDL یا زبان توصیف سخت افزار استفاده میشود.

باید هسته ILA بصورت هستی به سورس کد HDL اضافه شود که این امر توسط ابزار Chipscope Insert Core عملی میشود.

سپس تمام سیگنال های مورپ نظر به هسته های ILA متصل میشوند.

پس از تولید فایل BIT مربوطه , نتایج پیکربندی با کلیک بروی Chipscope در صفحه Design مربوط به ISE شرکت Xilinx آنالیز و تحلیل میشوند.

پس از انجام مراحل فوق نتایج تحلیل برای ما قابل مشاهده می باشد.

بخش ۴

نتایج و آنالیز