نشریه علمی "پدافند الکترونیکی و سایبری"
سال هفتم، شماره ۴، زمستان ۱۳۹۸، ص ۵۰-۳۷

طراحی و تولید یک کد احراز اصالت سخت‌افزاری با استفاده از تابع فیزیکی کپی‌ناپذیر داور و مدار مولد اعداد تصادفی بر روی تراشه FPGA

مسعود معصومی[1*]، علی دهقان[2] و اقبال مددی[3]

۱- استادیار، دانشکده فنی دانشگاه آزاد اسلامی واحد اسلامشهر، ۲- کارشناس ارشد مخابرات، دانشگاه تربیت مدرس، ۳- کارشناس ارشد الکترونیک

(دریافت: ۹۷/۱۱/۱۲، پذیرش: ۹۷۸/۰۳/۲۸)

چکیده

یکی از چالش‌های مهم در امنیت سخت‌افزار مقابله با کپی‌سازی و استفاده از سخت‌افزارهای جعلی به‌جای سخت‌افزارهای اصلی و واقعی است. یکی از مؤثرترین روش‌های مقابله با این نوع حملات و محافظت از اصالت و امنیت فیزیکی بستر پیاده‌سازی الگوریتم‌های رمزنگاری، استفاده از توابع کپی‌ناپذیر فیزیکی یا پاف است. در این مقاله تحقق عملی یک پاف سیلیکونی مبتنی بر مالتی پلکسر موسوم به پاف داور بر روی تراشه‌های FPGA از خانواده Xilinx و ایجاد یک کد تصادفی سخت‌افزاری برای احراز اصالت تراشه گزارش شده است. ابتدا با استفاده از پاف، یک هسته اولیه ۳۲ بیتی تصادفی تولید شده که از آن به‌عنوان مقدار اولیه یک شیفت رجیستر با بازخورد خطی استفاده شده است. پس از آن با پیاده‌سازی یک مولد اعداد تصادفی مبتنی بر نوسان‌سازهای حلقوی بر روی تراشه FPGA، جمع انحصاری خروجی‌های به‌دست‌آمده از شیفت رجیستر و مولد اعداد تصادفی و تصحیح دنباله خروجی با استفاده از تصحیح‌کننده وان نیومن یک کد ۶۴ بیتی برای شناسایی منحصربه‌فرد تراشه پیاده‌سازی تولید شده است. طرح پیاده‌سازی‌شده به‌گونه‌ای است که کد تولیدشده به‌عنوان امضای پاف را غیرقابل کپی‌سازی، غیرقابل مدل‌سازی و غیرقابل بازسازی می‌سازد. نتایج پیاده‌سازی نشان داد که با استفاده مدار ذکرشده و مصرف تقریباً ۱۵ درصد از سطح تراشه بورد استاندارد حملات کانال جانبی موسوم به ساکورا حاوی تراشه Spartan-6 XC6SLX75، قادر به تولید یک کد ۶۴ بیتی تصادفی برای شناسایی تراشه و استفاده از آن در پروتکل‌های احراز هویت به‌منظور تائید اصالت سخت‌افزار هستیم.

کلیدواژه‌ها: امنیت سخت‌افزار، تابع کپی‌ناپذیر فیزیکی، مولد اعداد تصادفی سخت‌افزاری، پیاده‌سازی FPGA

۱. مقدمه

یکی از چالش‌های مهم در حوزه امنیت سخت‌افزار، مقابله با مهندسی معکوس سخت‌افزار و اطلاعات تراشه‌ها و نیز استفاده از سخت‌افزارهای جعلی به‌جای سخت‌افزارهای اصلی و واقعی است. در حقیقت هدف این نوع حمله کشف کلید یا پارامترهای حساس ابزار رمز نیست بلکه هدف مهاجم بازیابی یا تغییر اطلاعات ذخیره‌شده در تراشه‌های سامانه رمز است. از این‌رو برای مقابله با آن باید تمهیدات ویژه و متفاوت با حملات رمزشکنی نرم‌افزاری و متداول در نظر گرفت. تاکنون روش‌های زیادی برای جلوگیری از تولید غیرمجاز قطعات نیمه‌هادی ارائه شده است. این روش‌ها شامل استفاده از سازوکارهای رمزنگاری نظیر الگوریتم‌های رمزنگاری، امضای دیجیتال، کد کردن داده‌ها و ... هستند. تمامی این روش‌ها مبتنی بر کلیدهای رمزنگاری هستند. این کلیدها در اغلب موارد در حافظه‌های غیر فرار یا فیوزها ذخیره می‌شوند که در معرض حملات مهندسی معکوس بوده و می‌توان توسط فناوری‌های موجود، اطلاعات درون آن‌ها را خوانده و مورد استفاده قرار داد. از این‌رو برای تأمین امنیت سامانه‌ها نمی‌توان تنها به پروتکل‌ها و الگوریتم‌های رمزنگاری متکی بود. روند مقالات و گزارش‌ها به‌روشنی نشان می‌دهد که برای تأمین امنیت، سازوکارهای محافظت نرم‌افزاری به‌تنهایی کافی نیستند. در عوض نیازمند روش‌های جدیدی هستیم که به لحاظ فیزیکی قابل اعتماد بودن و امنیت فیزیکی بستر پیاده‌سازی و ابزار امنیتی را تائید کنند. یکی از مؤثرترین روش‌های پیشنهادشده برای مقابله با این نوع حملات و محافظت از اصالت سخت‌افزار و سامانه‌های رمز، استفاده از توابع کپی‌ناپذیر فیزیکی است [۴-۱]. توابع کپی‌ناپذیر فیزیکی یا پاف‌ها را می‌توان برای استخراج پارامترهای مخفی از خصوصیات فیزیکی مدارهای مجتمع مورد استفاده قرار داد. وقتی این خصوصیت فیزیکی تأخیر یا یک عامل زمانی باشد، این وضعیت شبیه به استخراج مقادیر تصادفی از نویز خواهد بود.

* رایانامه نویسنده مسئول: m_masoumi@iiau.ac.ir

یکی از مناسب‌ترین ویژگی‌ها برای شناسایی منحصربه‌فرد مدارهای مجتمع می‌تواند تغییرات پارامترهای وابسته به فرآیندهای ساخت آن‌ها باشد. در واقع پاف تابعی است که پاسخ آن به هر ورودی، به‌صورت تکرارناپذیری، به فرآیند ساخت آن وابسته است. فرآیندهای وابسته پاف‌ها می‌توانند انواع و اقسام داشته باشند که بافه‌ای موسوم به بافه‌ای غیر الکترونیکی مانند نوری، اکوستیکی یا صوتی و مغناطیسی از آن جمله هستند اما بافه‌ای الکترونیکی و به‌خصوص بافه‌ای سیلیکونی که بر مبنای تأخیرها و زمان‌بندی‌های[1] خاص فرآیندهای ساخت مدارهای مجتمع هستند متداول‌تر هستند. اکنون کاملاً مشخص شده است که زمان‌بندی‌ها و تأخیرهای یک مدار مجتمع از یک بستر[2] بر روی یک ویفر تا یک بستر دیگر حتی بر روی همان ویفر به‌دلیل تغییرات فرایندهای ساخت مانند تأخیر سیم‌ها، سیگنال‌ها و ولتاژ آستانه که در فرآیند ساخت اتفاق می‌افتد، انحنای ویفر و ... تغییر می‌کند. از این‌رو پاف‌ها به‌سادگی قابل ارزیابی[3] ولی به‌سختی قابل پیش‌بینی هستند. علاوه بر آن پاف‌ها به‌سادگی قابل ساخت ولی به لحاظ عملی حتی با داشتن تمام اطلاعات ساخت، غیرقابل کپی‌سازی هستند. اکنون این مسئله به‌خوبی مشخص شده است که ولتاژ آستانه و نیز ضخامت اکسید گیت ترانزیستورها حتی در یک فرآیند ساخت و بر روی یک بستر، یکسان نیستند که این به مفهوم آن است که چنانچه دو تراشه حتی با یک شرایط یکسان ساخته شوند از نظر تأخیر و توان مصرفی یکسان نخواهند بود. با کوچک‌تر شدن ابعاد فناوری و اندازه نمای ترانزیستورها این تغییرات به‌مراتب بیشتر نیز می‌شود. عمده پاف‌ها مجموعه‌ای از چالش‌ها را به مجموعه منحصربه‌فردی از پاسخ‌ها در قالب زوج‌های چالش-پاسخ[4] می‌نگارند به شکلی که این پاسخ‌ها وابسته به خصوصیت فیزیکی آن ابزار خاص است و قابل کپی‌سازی در ابزار دیگری نیست [۶-۵]. در واقع این مجموعه چالش-پاسخ باید دارای ویژگی‌هایی از جمله غیرقابل پیش‌بینی بودن به مفهوم وجود تناظر یک به یک بین هر چالش و پاسخ متناظر آن، منحصربه‌فرد و غیرقابل پیش‌بینی بودن آن‌ها، غیرقابل شکست بودن به مفهوم غیرممکن بودن به‌دست آوردن زوج چالش-پاسخ بدون در اختیار داشتن فیزیکی پاف و غیرقابل تکرار بودن خروجی حتی با وجود اطلاع از تمام پارامترهای ساخت و قابل تشخیص بودن رخنه به مفهوم تخریب پاف و مشخص شدن حمله به آن در صورت انجام حملات مهاجم می‌باشد. به‌دلیل ویژگی‌های منحصربه‌فرد پاف‌ها، از آن‌ها در کاربردهای مختلفی از جمله محافظت از مالکیت معنوی، نگه‌داری امن کلید، تائید ابزار[5] که به‌نوعی مقابله با کپی‌سازی و مهندسی معکوس بشمار می‌رود، برقراری اعتماد در ارتباطات راه دور ... استفاده می‌شود [۱۰-۷].

بر اساس بررسی‌های به‌عمل‌آمده و با وجود اینکه محافظت سامانه‌های رمزنگاری در برابر سخت‌افزارهای جعلی و نیز مهندسی معکوس از موضوعات بسیار مهم و غیرقابل چشم پوشی در حوزه امنیت سخت‌افزاری سامانه‌ها محسوب می‌گردد و با وجود انتشار مقالات متعدد از طرف دانشگاه‌ها و مراکز تحقیقاتی بین‌المللی در این خصوص، تاکنون تحقیقات کمی در مورد این موضوع در داخل کشور صورت گرفته است.

در این مقاله ضمن بررسی مختصر انواع توابع فیزیکی کپی‌ناپذیر موجود و به‌خصوص توابع قابل پیاده‌سازی بر روی تراشه‌های سیلیکونی و تراشه‌های FPGA، یک نمونه از انواع توابع کاربردی و قابل پیاده‌سازی بر روی تراشه FPGA موسوم به پاف داور که مبتنی بر گیت‌های مالتی پلکسر است [۱۴-۱۱] به‌صورت عملی پیاده‌سازی شده و نتایج آن مورد بررسی قرار گرفته است. علاوه بر آن با پیاده‌سازی یک مولد اعداد تصادفی مبتنی بر نوسان‌ساز حلقوی بر روی تراشه مزبور و جمع کردن بیت به بیت خروجی آن با خروجی به‌دست‌آمده از پاف و تصحیح دنباله خروجی با استفاده از تصحیح‌کننده وان نیومن[6]، یک کد منحصربه‌فرد ۶۴ بیتی برای شناسایی و احراز اصالت تراشه تولیدشده که می‌توان از آن در پروتکل‌های احراز اصالت سخت‌افزار استفاده نمود. نتایج به‌دست‌آمده از طرح انجام‌شده امیدوارکننده و قابل اعتماد است به شکلی که می‌توان از آن برای افزایش امنیت و اعتماد فیزیکی بسترهای سخت‌افزاری استفاده نمود زیرا حتی در صورت دسترسی فیزیکی مهاجم به ابزار رمز امکان کپی‌سازی، بازیابی و مهندسی معکوس شماهای پیاده‌سازی شده درون تراشه و کد تولیدشده توسط آن‌ها وجود ندارد.

با توجه به تنوع بالای تراشه‌های برنامه‌پذیر، آزمایش‌های لازم بر روی چند تراشه نمونه از تراشه‌های خانواده Xilinx انجام‌شده و نتایج از حیث غیرقابل تکرار و قابل اعتماد بودن مورد بررسی قرار گرفته است. از نتایج این کار می‌توان برای ارائه و پیاده‌سازی راه‌کار مناسب برای محافظت از تراشه‌ها در مقابل جعل، کپی‌سازی و آسیب‌پذیری‌های مختلف از ناحیه ارتباط سامانه امنیتی با سامانه‌های غیر خودی، تولید اعداد تصادفی و شناسایی ابزار استفاده نمود. در ادامه مقاله ابتدا در بخش ۲ در مورد اهمیت امنیت فیزیکی و ریشه‌های اعتماد و پس از آن در بخش ۳ به‌طور مختصر ساختار بافه‌ای سیلیکونی و کاربردهای آن‌ها را

[1] Timing
[2] Die
[3] Evaluate
[4] Challenge-Response Pair
[5] Device Authentication
[6] Von Neumann

تشریح می‌کنیم. در بخش ۴ به بررسی مفهوم غیرقابل شکست بودن با ارائه برخی روابط اساسی خواهیم پرداخت. در بخش ۵ شمای طرح پیشنهادی، نحوه پیاده‌سازی آن بر روی تراشه‌های هدف و نتایج پیاده‌سازی را ارائه خواهیم داد. در انتها با جمع‌بندی بحث و ارائه نتایج نهایی، مقاله را به پایان می‌بریم.

۲. امنیت فیزیکی و ریشه‌های اعتماد[1]

برای فراهم آوردن امنیت تنها نمی‌توان به الگوریتم‌ها و پروتکل‌های رمزنگاری متکی بود زیرا استفاده از الگوریتم‌ها و پروتکل‌های رمزنگاری آخرین مرحله از فراهم آوردن امنیت یک سامانه است. همان‌طور که گفته شد برای استفاده از هر نوع سازوکار امنیتی و هر الگوریتم رمزنگاری در دنیای واقعی باید آنرا بر روی یک بستر واقعی پیاده‌سازی کرد. بسترهای پیاده‌سازی معمولاً تراشه‌های دیجیتال از قبیل ریزپردازنده‌ها یا تراشه‌های FPGA هستند. در این صورت سه مسئله حتماً باید مورد توجه قرار بگیرد: ۱. تولید کلید به‌صورت امن، ۲. نگهداری از کلید به‌صورت امن و ۳. اجرای الگوریتم به‌صورت امن. بدیهی است که برای فراهم آوردن مفاهیم فوق نمی‌توان تنها به الگوریتم‌ها و پروتکل‌های رمزنگاری متکی بود. در عوض نیازمند روش‌ها و سازوکارهایی هستیم که در ابتدا به‌صورت فیزیکی قابل اعتماد بودن بستر پیاده‌سازی را تائید کنند و پس از آن به پیاده‌سازی الگوریتم‌های امنیتی بپردازیم زیرا در غیر این صورت انواع حملات رمزشکنی تهاجمی و فیزیکی می‌توانند کلید یا سایر پارامترهای امنیتی را از ابزار رمز استخراج کنند و تمام سازوکارهای امنیتی پیش‌بینی‌شده آن را دور بزنند. مهاجم می‌تواند در زمان خاموش بودن ابزار رمز اطلاعات و کلید رمز را به‌طور فیزیکی از حافظه پردازنده استخراج کند. از این‌رو ذخیره و ارسال اطلاعات دیجیتال به یک ابزار رمز امری چالش برانگیز و هزینه‌بر است. به ابزارها و روش‌هایی که باعث محافظت از امنیت فیزیکی ابزار در برابر حملات تهاجمی و فیزیکی می‌شود ریشه‌های امنیت فیزیکی اطلاق می‌شود. متأسفانه تاکنون روش مشخصی برای مقابله با حملات فیزیکی ارائه نشده زیرا مقابله با این حملات چندان ساده نیست. از جمله معدود روش‌هایی که در سال‌های اخیر برای مقابله با حمله‌های فیزیکی ارائه شده است پاف‌ها یا توابع کپی‌ناپذیر فیزیکی هستند. تابع فیزیکی غیر قابل نسخه‌برداری پاف، تابعی است که پاسخ آن به ورودی، به‌صورت غیر قابل کنترلی، به فرآیند ساخت آن وابسته است و همان‌طور که اشاره شد به‌دلیل ویژگی پاف، می‌توان از آن‌ها برای تولید کلید رمزنگاری بدون نیاز به ذخیره‌سازی در حافظه فیزیکی استفاده نمود. مهم‌ترین ویژگی این توابع آن است که اولاً نیازی به فرآیند اضافی برای برنامه‌ریزی موارد یا رموز درون آن وجود ندارد و ثانیاً نمی‌توان با صرف هزینه و زمان محدود از روی آن نسخه‌برداری نمود زیرا تعیین مشخصه آن بسیار پیچیده است. بدین معنی که با تعداد محدودی اندازه‌گیری، نمی‌توان مشخصه تابع را تعیین کرده و سپس از روی آن مشخصه، خروجی تابع را برای ورودی تصادفی دیگری تعیین نمود. از این‌رو مهاجمینی که با استفاده از حملات فیزیکی یا تهاجمی قصد استخراج کلید یا کپی‌سازی مدار داخل تراشه رمزنگار را داشته باشند در اهداف خود ناکام خواهند ماند زیرا خروجی هر تابع بر مبنای خصوصیات منحصربه‌فرد یک تراشه خاص تولید می‌شود که هیچ تراشه مشابه دیگری قادر به تولید آن پاسخ حتی به ازای همان ورودی‌ها نخواهد بود. از آنجا که با استفاده از این توابع نیاز به ذخیره‌سازی کلید در ابزار رمز نداریم و حتی در صورت فاش شدن مدار مولد کلید، به ازا ورودی یکسان خروجی آن متفاوت خواهد بود لذا دو هدف اول برآورده شده و پس از آن لازم است تا الگوریتم به‌صورت امن پیاده‌سازی و اجرا شود بدین مفهوم که الگوریتم به شکلی پیاده‌سازی شود که از نشت اطلاعات فیزیکی و کانال جانبی در حین اجرای الگوریتم جلوگیری شود [۱۶-۱۵].

۳. برخی از انواع پاف‌های سیلیکونی

با توجه به متداول‌تر بودن استفاده از پاف‌های سیلیکونی برای آشنا شدن بهتر با موضوع چند نوع پاف سیلیکونی متداول را که برای پیاده‌سازی بر روی FPGA مناسب هستند را به‌صورت مختصر تشریح می‌کنیم و سایر موارد را به خواننده علاقه‌مند واگذار می‌کنیم.

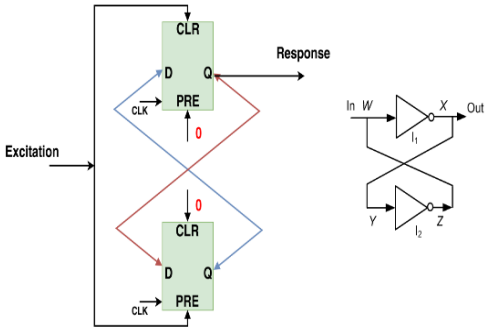
۳-۱. پاف‌های مبتنی بر حافظه[2]

پاف‌های مبتنی‌بر SRAM و پاف‌های پروانه‌ای[3] از جمله پاف‌های متداول مبتنی‌بر حافظه هستند [۱۷]. یک پاف SRAM شامل تعداد زیادی از واحدهای حافظه است. وجود تفاوت جزئی در ولتاژها ترانزیستورها به‌دلیل تفاوت در فرآیندهای ساخت، توسط معکوس‌کننده‌های موجود در ساختار تقویت شده و باعث ایجاد '1' یا '0' تصادفی در خروجی خواهد شد. در واقع چالش، مجموعه‌ای از واحدهای حافظه پس از روشن شدن مدار و پاسخ، مجموعه‌ای از این مقادیر خروجی خواهد بود. از آنجا که همه FPGA ها شامل حافظه‌هایی که نیاز به فرانشانی نداشته باشند نیستند این نوع **پاف‌ها** برای همه FPGAها مناسب نیستند. راه‌کار پیشنهادشده از سوی گواردو[4] [۱۸-۱۷] جایگزینی

[1] Physical Security and Roots of Trust

[2] Memory-Based PUFs
[3] Butterfly
[4] Guajardo

معکوس‌کننده‌های ساختار با فلیپ فلاپ‌های متقاطع است. این مدارها که می‌توانند اطلاعات را در خود ذخیره کنند با ریست شدن اطلاعاتشان پاک می‌شود و نیاز به فرانشانی ندارند. از این‌رو، این ساختار می‌تواند بر روی FPGA پیاده‌سازی شود. شکل (۱) نشان‌دهنده نمونه‌ای از این ساختارهاست که یکی معکوس‌کننده متقاطع و یکی فلیپ- فلاپ متقاطع موسوم به پاف پروانه را نشان می‌دهد.



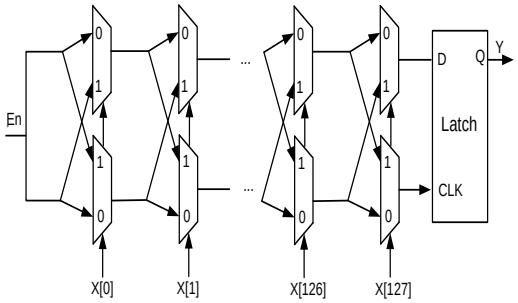
شکل (۱): دو فلیپ فلاپ متقاطع موسوم به پاف پروانه (چپ) و معکوس کننده متقاطع (راست) [۱].

۳-۲. پاف‌های مبتنی بر تأخیر[1]

این گونه پاف نیز انواع و اقسام دارند اما پاف‌های داور[2] و پاف‌های مبتنی بر نوسان‌ساز از جمله پرکاربردترین آن‌ها هستند. پاف‌های داور در خانواده پاف‌های قوی قرار می‌گیرند [۱۹]، بدین مفهوم که می‌توانند تعداد زیادی زوج چالش-پاسخ[3] را فراهم آورند و از این حیث برای شناسایی ابزارهای کم قیمت مناسب هستند. ساختار پایه چنین مداراتی در شکل (۲) نشان داده شده است. این ساختار یک زنجیره ۱۲۸ تایی از مالتی پلکسرهای با ورودی مشترک است که یکی از خروجی‌های آخرین طبقه به ورودی D یک فلیپ فلاپ و خروجی دیگر به پالس ساعت آن وصل می‌شود. ورودی مدار سیگنال‌های پله است. ایده اصلی پس این نوع پاف بر مبنای برقراری یک شرط مسابقه[4] بین دو مسیر دیجیتال درون یک تراشه است. بیت‌های چالش رشته بیت ورودی $X[0]$~$X[127]$ است که به ورودی‌های انتخاب‌گر مالتی پلکسرها وارد می‌شوند. سیگنال $X[i]$ نشان‌دهنده آن است که سیگنال ورودی در طبقه iام به کدام مالتی پلکسر وارد می‌شود.

سیگنال‌های چالش مختلف ورودی و تأخیرهای مختلف بین زنجیره مالتی پلکسرهای موازی تعیین‌کننده آن است که آیا سیگنال پله زودتر به ورودی D فلیپ فلاپ یا به پایه پالس ساعت می‌رسد. در حالت اول منطق '1' و در حالت دوم منطق '0' در خروجی فلیپ فلاپ قفل (لچ) خواهد شد. هر بیت خروجی به‌عنوان یک بیت امضا در پاسخ به چالش ورودی در نظر گرفته می‌شود.

در سناریوهای مرسوم کاربردی، از پاف در دو مرحله استفاده می‌شود. در مرحله اول که نام‌نویسی[5] نام دارد، تعدادی زوج ورودی - خروجی با استفاده از پاف مورد نظر جمع آوری شده و درون یک پایگاه داده که پایگاه داده زوج ورودی-خروجی نام دارد، ذخیره می‌شود. در فاز دوم که فاز شناسایی نام دارد، یک ورودی از پایگاه داده انتخاب شده و به پاف اعمال می‌شود. خروجی تولیدشده توسط پاف با خروجی مربوطه ذخیره‌شده در پایگاه داده مقایسه می‌شود. در صورتی که شباهت پاسخ پاف به ازای ورودی مشخص، با پاسخ ذخیره‌شده در پایگاه داده از حد معینی بیشتر بود، عمل شناسایی مثبت بوده و این پاسخ، پاسخ همان پاف مورد نظر تشخیص داده می‌شود [۲۰].

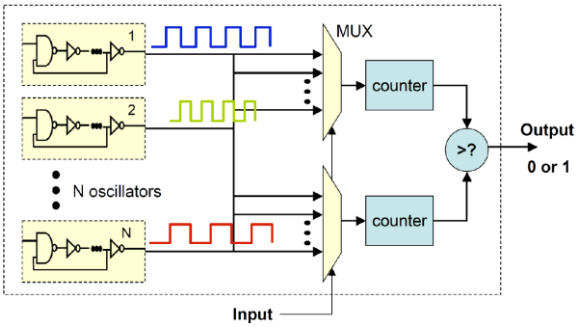


شکل (۲): ساختار یک پاف داور که زنجیره‌ای از مالتی پلکسرها و چالش‌ها یک بیت تصادفی را تولید می‌کنند [۱].

پاف‌های مبتنی بر نوسان‌سازهای حلقوی اولین بار توسط سوه[6] و همکارانش برای تولید رشته بیت‌های تصادفی است را پیشنهاد شد [۲۱]. یک نوسان‌ساز حلقوی یک مدار ساده متشکل از تعدادی معکوس‌کننده است که به‌صورت حلقوی به یکدیگر متصل شده‌اند و با فرکانس مشخصی نوسان می‌کند. فرکانس نوسان نوسان‌ساز بستگی به تعداد و تأخیر معکوس‌کننده‌ها و نیز سیم‌های بین آن‌ها دارد. از آنجا که این تأخیرها به پارامترهای ساخت و نیز برخی فاکتورهای غیرقطعی بستگی دارد فرکانس نوسان نوسان‌ساز به‌طور قطعی قابل پیش‌بینی نیست. ساده‌ترین فرم این پاف‌ها رشته‌ای از بیت‌های '0' و '1' را در خروجی با مقایسه فرکانس یک زوج یا چند نوسان‌ساز تولید می‌کند. از آنجا که پاف‌های مبتنی بر نوسان‌سازهای حلقوی نیاز به تقارن ندارند پیاده‌سازی آن‌ها در FPGA نسبتاً ساده است. شکل (۳) ساختار

[1] Delay-Based PUFs
[2] Arbiter PUF
[3] Challenge-Response Pair
[4] Race Condition
[5] Enrollment
[6] Suh

یک پاف مبتنی‌بر N نوسان‌ساز حلقوی را نشان می‌دهد. در منابع مربوطه نشان داده شده که یک پاف دارای N نوسان‌ساز حلقوی دارای آنتروپی $N \times \log N$ بیت اطلاعات است [۵]. این توابع، در مقایسه با پاف‌هایی که بر اساس تأخیر گیت ها و داوری طراحی شده‌اند از لحاظ پیاده‌سازی ساده‌تر هستند اما از لحاظ منابع و توان مصرفی نیاز به منابع سخت‌افزاری و توان مصرفی بیشتری نسبت به روش تأخیر و داوری (روش مبتنی بر مالتی پلکسر) دارند. مطابق با آنچه در منابع مربوطه گزارش شده است پاف‌های مبتنی‌بر SRAM دارای بالاترین امنیت (آنتروپی)، پس از آن پاف‌های پروانه و پس از آن پاف‌های مبتنی بر تأخیر قرار دارند [۵].



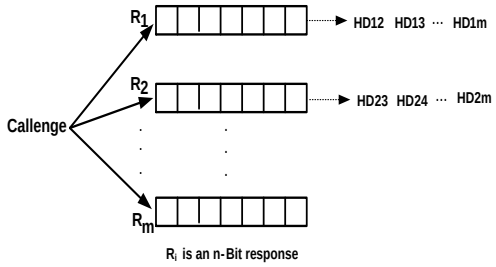
شکل (۳): ساختار پایه یک پاف مبتنی بر نوسان‌ساز حلقوی [۵].

۴. مفهوم غیرقابل شکست بودن از نظر فیزیکی[1]

قبل از تعریف و بیان مفهوم غیرقابل شکست بودن از نظر فیزیکی دو مفهوم مهم که در منابع پاف به‌طور گسترده در منابع مربوطه مورد استفاده قرار می‌گیرد را بیان می‌کنیم. مفهوم فاصله همینگ بین تراشه‌ای[2] و فاصله همینگ درون تراشه‌ای[3] از مفاهیم بسیار مهم در زمینه پاف است که به شکل زیر تعریف می‌شود. مجموعه‌ای از بیت‌های پاسخ را مطابق شکل (۴) در نظر می‌گیریم. وقتی یک چالش به z تراشه دارای یک پاف یکسان ارسال می‌شود، فرض کنیم m پاسخ به‌دست‌آمده R_1، R_2 ... R_m باشد که هرکدام n بیتی هستند. در این‌صورت فاصله همینگ با مقایسه هر پاسخ با سایر پاسخ‌های مجموعه و متوسط‌گیری از تمام فاصله‌های همینگ به‌دست می‌آید که فاصله همینگ بین تراشه‌ای نامیده می‌شود. رابطه (۱) این روال را در قالب یک رابطه بیان می‌کند و شکل (۴) بیان‌کننده مفهوم آن است [۱، ۲۳-۲۲].

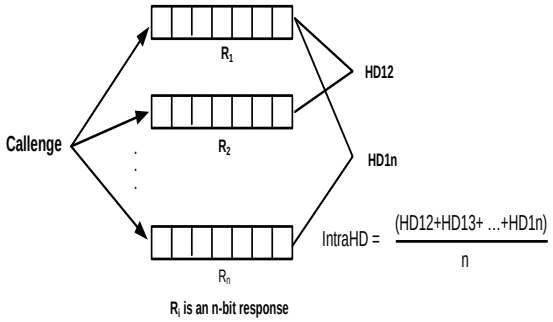
$$\text{Inter} - \text{HD} = \frac{2}{z(z-1)} \sum_{x=1}^{z} \frac{HD(R_x, R_y)}{n} \times 100\% \qquad (۱)$$

جایی‌که R_x و R_y متناظراً پاسخ‌های n بیتی تراشه‌های x و y و z نشان‌دهنده تعداد تراشه‌هاست.



شکل (۴): مفهوم فاصله همینگ بین تراشه‌ای [۲۳].

شکل (۵) نشان‌دهنده مفهوم فاصله همینگ درون تراشه‌ای است جایی‌که R_1 ... R_n پاسخ پاف به چالش یکسان در شرایط متفاوت مثلاً در دماهای مختلف است. رابطه (۲) مفهوم وزن همینگ درون تراشه‌ای و رابطه (۳) قابل اعتماد بودن پاف را در قالب یک رابطه ساده بیان می‌کند.



شکل (۵): مفهوم فاصله همینگ درون تراشه‌ای، جایی‌که R_i ها پاسخ پاف در آزمایش‌های مختلف است [۲۳].

$$Intra - HD = \frac{HD(R_x, R_{x'})}{n} \times 100\% \qquad (۲)$$

$$Reliability = 100\% - Intra\,HD \qquad (۳)$$

فرض کنیم که مهاجم بر روی یک فرآیند خاص از یک کلاس پاف کنترل کامل داشته باشد به این مفهوم که می‌تواند شرایط، پارامترها و تصادفی بودن منبع $P.Create$ را در یک محدوده مشخص تحت تأثیر قرار دهد. هنگامی‌که تائید هویت مبتنی پاف با فرض وجود چنین مهاجمی در نظر گرفته می‌شود، یک بحث و نظریه قوی‌تر برای اطمینان از منحصربه‌فرد ماندن هویت‌ها لازم است. دلیل این امر آن است که مهاجم می‌تواند از کنترل خود بر روی فرآیند استفاده کرده و دو پاف مشابه‌تر به یکدیگر در مقایسه با پافی که برای تولید امضای منحصربه‌فرد به‌کار می‌رود تولید کند. برای اجتناب از این مسئله کاربر مایل است تا پاف

[1] Physically Unclonability
[2] Inter-Hamming Distance
[3] Intra-Hamming Distance

ویژگی منحصربه‌فرد بودن[1] داشته باشد تا حتی در حضور چنین مهاجمی، منحصربه‌فرد بودن هویت‌ها به مخاطره نیافتد. این همان چیزی است که به‌عنوان غیرقابل شکست بودن از نظر فیزیکی[2] شناخته می‌شود.

مطابق با تعریف یک پاف کلاس P از نظر فیزیکی غیرقابل شکست است اگر اولاً قابل ارزیابی باشد، ثانیاً برای مهاجم بسیار سخت یا غیرممکن باشد که فرآیند $P.Create$ را به نحوی تحت تأثیر قرار دهد که

$$Pr\left(dist[Y \leftarrow PUF(X);\ Y' \leftarrow PUF'(X)] < D_p^{inter}(X)\right) is\ high \quad for\ X \leftarrow X_p \qquad (۴)$$

بدین مفهوم که فاصله همینگ دو پاسخ Y و Y' به چالش X از دو پاف مختلف بیشتر از فاصله همینگ بین تراشه‌ای یک پاف مشخص است.

همچنین غیرقابل شکست بودن از نظر فیزیکی بدین مفهوم است که پیدا کردن دو پاف مختلف که برای آن‌ها رابطه (۵) برقرار باشد بسیار مشکل باشد.

$$\Pr\left(dist[Y \leftarrow PUF(X);\ Y' \leftarrow PUF'(X)] > D_p^{intra}(X)\right) is\ low \qquad (۵)$$

در اینجا "سخت" بودن بسته به کاربرد و امکانات فنی مهاجم دارد و سخت بودن به مفهوم میزان هزینه و پیچیدگی احتمالی محاسباتی یا کاری است که مهاجم متحمل خواهد شد. این مشکلات مربوط به زمانی است که مهاجم بخواهد یک پاف مشابه با پاف مورد نظر بسازد. هنگامی‌که این ویژگی با ویژگی قابل ساخت بودن ترکیب می‌شود این مفهوم را پیدا می‌کند که "ساخت یک پاف تصادفی آسان اما ساخت یک نمونه مشخص آن بسیار مشکل است". آن کلاس از پاف‌ها که ویژگی غیرقابل شکست بودن از نظر فیزیکی را دارا هستند از نظر امنیتی دارای این مزیت هستند که حتی کارخانه سازنده با در دست داشتن مشخصات آن‌ها قادر به باز تولید آن‌ها نیست و از این‌رو ویژگی منحصربه‌فرد بودن آن‌ها خدشه‌دار نخواهد شد. از این‌رو نیاز به سازنده مورد اعتماد برای ساخت پاف نیست و این کلاس از پاف‌ها مقاوم در برابر سازنده[3] نامیده می‌شوند.

۴-۱. غیرقابل پیش‌بینی بودن[4]

بسیاری از کاربردهای پاف مبتنی‌بر عملکرد مناسب سامانه چالش-پاسخ هستند بدین مفهوم که در پاسخ به یک چالش یک پاسخ تصادفی دریافت شده و به‌عنوان امضای پاف مورد استفاده قرار می‌گیرد. اما برای تضمین امنیت پاف تنها غیرقابل شکست بودن و منحصربه‌فرد بودن کافی نیستند بلکه ویژگی غیرقابل پیش‌بینی بودن نیز باید مورد توجه قرار داده شود بدین مفهوم که پاسخ‌های مشاهده‌نشده به‌اندازه کافی تصادفی و غیرقابل پیش‌بینی حتی بعد از مشاهده پاسخ سایر چالش‌ها باشد. مطابق با تعریف، یک پاف کلاس P غیرقابل پیش‌بینی است اگر قابل ارزیابی بوده و برآورده کردن موارد زیر برای یک پاف تصادفی $PUF \in P$ مشکل و سخت باشد.

- در فاز آموزش، شخص مجاز است تا پاف را با تعداد محدودی پاسخ- چالش ارزیابی کند. مجموعه چالش‌های ارزیابی‌شده X'_p یا به‌صورت تصادفی (غیرقابل پیش‌بینی بودن ضعیف[5]) یا به‌صورت تطبیقی (غیرقابل پیش‌بینی بودن قوی[6]) انتخاب می‌شوند.

- در فاز چالش، چالش $X \leftarrow \frac{X_p}{X'_p}$ به شخص معرفی می‌شود. در این صورت لازم است تا شخص Y_{pred} را برای این چالش در هنگام ارزیابی پاف بسازد. شخص به پاف دسترسی ندارد ولی الگوریتم پیش‌بینی $predict$ در فاز قبل به شکلی آموزش داده شده که بتواند از چالش ورودی، خروجی را حدس بزند یا به عبارت دیگر:

$$Y_{pred} \leftarrow predict\ (X) \qquad (۶)$$

در این صورت چنانچه رابطه (۷) برقرار باشد مهاجم برنده بازی خواهد بود.

$$\Pr\left(dist[Y_{pred} \leftarrow predict(X);\ Y \leftarrow PUF(X)] > D_p^{inter}(X)\right) is\ high \qquad (۷)$$

نکته مهم مشابهت این رابطه با توزیع بین تراشه‌ای در تعریف غیرقابل شکست بودن فیزیکی پاف است. اما به‌جای در نظر گرفتن فاصله با پاف دوم PUF'، فاصله با الگوریتم پیش‌بینی $predict$ که در فاز قبل با استفاده از پاسخ‌های همین پاف آموزش داده‌شده در نظر گرفته شده است. در بهترین حالت می‌توان نشان داد که پاسخ به چالش‌های مختلف کاملاً مستقل از یکدیگر است که این به این معنی است که توسط هیچ الگوریتم پیش‌بینی قابل پیش‌بینی نیست. البته چنین کیفیت بالایی به‌سختی برای پاف قابل اثبات است و بیشتر انگیزه‌ها و توجیهات فیزیکی می‌تواند این امر را اثبات کند. اثبات امنیت پاف در برابر حملات به غیرقابل پیش‌بینی بودن آن بستگی به میزان دقت

[1] Uniqueness
[2] Physical Unclonability
[3] Manufacturer Resistance
[4] Unpredictability
[5] Weak Unpredictability
[6] Strong Unpredictability

تابع پیش‌بینی، پیچیدگی تابع پیش‌بینی و ارزیابی و پیدا کردن تعداد مورد نیاز زوج چالش- پاسخ برای آموزش دادن پاف دارد. البته مشابه با ارزیابی امنیت الگوریتم‌های رمزنگاری مانند الگوریتم پیشرفته رمز استاندارد که مهاجم نوع الگوریتم و حتی نوع پیاده‌سازی آن را می‌داند، فرض بر آن است که مهاجم همه چیز را در مورد نوع پاف و نحوه پیاده‌سازی آن می‌داند و مشخصات وابسته به ابزار[1] هر پاف آن را در مقابل حملات مختلف محافظت می‌کند.

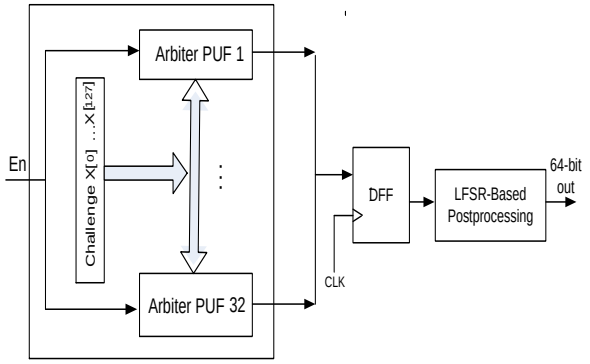
۵. پیاده‌سازی طرح پاف مورد نظر

همان‌طور که ذکر شد از بین تمام شماهای پیشنهادشده برای تحقق پاف‌ها بر روی FPGA، طرح‌های پروانه، نوسان‌ساز حلقوی و داور مناسب‌تر از بقیه برای پیاده‌سازی بر روی FPGA هستند که در این تحقیق طرح پاف داور پیاده‌سازی شد.

۵-۱. پیاده‌سازی پاف داور

برای پیاده‌سازی هسته اولیه مدار از یک پاف داور ۳۲ بیتی مطابق با شکل (۶) با ساختار یک زنجیره ۱۲۸ تایی از مالتی پلکسرهای با ورودی مشترک که یکی از خروجی‌های آخرین طبقه به ورودی D یک فلیپ فلاپ و خروجی دیگر به پالس ساعت آن وصل می‌شود استفاده شد. ورودی مدار سیگنال‌های پله است که به تمام ورودی‌ها به‌طور هم‌زمان می‌رسد. همان‌طور که ذکر شد برای امنیت بیشتر و مقابله با حملاتی که با استفاده از زوج چالش- پاسخ، پاف را مدل‌سازی می‌کنند بیت‌های انتخاب‌گر مالتی پلکسرها که در ساختار عادی پاف داور به‌عنوان چالش در نظر گرفته می‌شود درون تراشه قرار داده و پیاده‌سازی شد که این امر باعث مشکل‌تر شدن مهاجم در حمله به ساختار طرح شده است. زیرا از آنجا که پاسخ یک پاف سیلیکونی مبتنی بر تأخیر را می‌توان توسط یک تابع خطی چالش‌ها مدل کرد، چنانچه مهاجم چالش‌های ورودی که همان بیت‌های انتخاب‌گر مالتی پلکسرها هستند را بداند و بتواند تأخیر هر طبقه را تخمین بزند ممکن است بتواند با جمع کردن تأخیر مسیرها خروجی را حدس بزند. از این‌رو استفاده از چند مسیر برای تعیین پاسخ خروجی کار مهاجم را به‌مراتب سخت‌تر خواهد کرد. برای صرفه‌جویی در منابع تراشه و سخت‌تر کردن مهاجم به‌جای استفاده از چند مسیر، بیت‌های انتخاب‌گر مالتی پلکسرها را درون تراشه قرار دادیم که در این صورت کلید تولیدشده توسط پاف درون تراشه تولید و ذخیره خواهد شد و امکان دسترسی به آن برای مهاجم وجود نخواهد داشت.

پس از آن از یک شیفت رجیستر ۳۲ بیتی با بازخورد خطی با چند جمله‌ای اولیه $f(X) = X^{32} + X^{22} + X^2 + X + 1$ برای تولید ۶۴ بیت تصادفی استفاده شد به این شکل که ۱۰۲۴ بیت اولیه دور ریخته شد و پس از آن بیت‌های خروجی شیفت رجیستر مورد نظر قرار گرفتند. لازم به ذکر است که با استفاده از ساختار مزبور، تعداد بافه‌ای بیشتر و شیفت رجیستر با طول بلندتر قادر به تولید کدهای قوی‌تری هستیم اما به‌منظور صرفه‌جویی در منابع تراشه از رجیستر و هسته اولیه کوتاه‌تر استفاده کردیم. از این کد می‌توان به‌عنوان کلید خصوصی ابزار در پروتکل‌های شناسایی و احراز اصالت ابزار استفاده نمود [۲۴-۲۳].



شکل (۶): شمای طرح اولیه پیاده‌سازی شده بر روی تراشه FPGA شامل ۳۲ پاف داور که یک هسته اولیه ۳۲ بیتی تصادفی تولید می‌کنند.

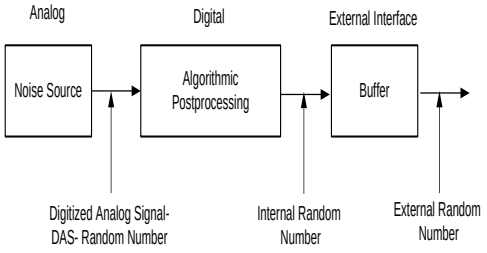
۵-۲. پیاده‌سازی مولد اعداد تصادفی بر روی تراشه FPGA

به‌منظور افزایش میزان تصادفی‌بودن کد تولیدشده و غیرقابل کپی‌برداری یا مدل‌سازی شدن طرح از سوی مهاجم از یک مدار مولد کد تصادفی مبتنی‌بر نوسان‌ساز حلقوی بر روی تراشه FPGA استفاده کردیم و دنباله تولیدشده را با دنباله‌ای که از تابع پاف به‌دست آمد XOR نمودیم. لازم به ذکر است که هدف این کار تولید دنباله تصادفی نامتناهی برای خروجی نیست بلکه هدف استخراج یک دنباله ۶۴ بیتی تصادفی به‌عنوان کلید پاف است که برای مهاجم غیرقابل کپی‌برداری، مدل‌سازی یا بازسازی باشد. در عمل برای ایجاد یک دنباله تصادفی بر روی تراشه‌های دیجیتال نیاز به یک منبع نویز دیجیتال داریم. معمولاً پس از ایجاد یک نویز اولیه، یک پس پردازش ریاضی[2] نسبتاً ساده مانند استفاده از یک تابع XOR یا تابع چکیده‌ساز[3] و یک تصحیح‌کننده وان نیومن[4] که باعث افزایش تصادفی بودن دنباله خروجی بدون کاهش آنتروپی آن می‌شود انجام‌شده و از خروجی آن به‌عنوان

[1] Instance Specific

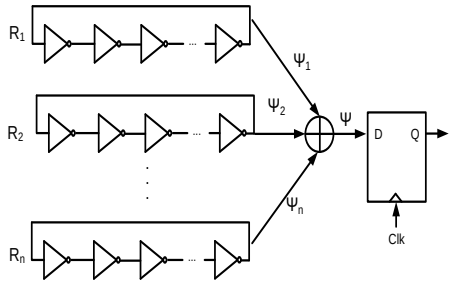
[2] Mathematical Post Processing
[3] Hash Function
[4] Von Neumann

رشته بیت تصادفی استفاده می‌شود. منبع نویز اولیه می‌تواند یک پالس ساعت دارای جیتر[1]، یک تابع آشوب و یا نویز حرارتی استخراج‌شده از یک مقاومت بیرون تراشه باشد. شکل (۷) یک مولد تصادفی نوعی را نشان می‌دهد.

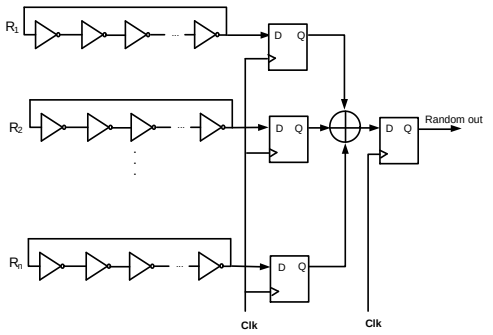


شکل (۷): بلوک دیاگرام یک مولد اعداد تصادفی نوعی بر روی تراشه FPGA.

برای پیاده‌سازی مولد اعداد تصادفی بر روی FPGA شماهای مختلفی پیشنهادشده که از آن جمله می‌توان به روش کاسیک[2] [۲۶]، اپستاین[3] [۲۷]، گولیک فیگارو[4] [۲۸] و کولبرنر گاج[5] [۲۹] اشاره کرد اما یکی از روش‌های مؤثر و ساده استفاده از نوسان‌سازهای حلقوی است. این ترکیب برای اولین بار توسط سونار، مارتین و استینسون [۳۰] ارائه شد. این طرح، طرح ساده‌ای است که از نوسان‌سازهایی که به‌صورت آزاد در حال نوسان هستند استفاده می‌کند. شکل (۸) شمای ساختار تولید بیت با استفاده از نوسان‌ساز حلقوی را نشان می‌دهد. شکل (۹) نوع بهبودیافته‌ای از این مولد اعداد تصادفی را نشان می‌دهد که در آن از فلیپ-فلاپ نوع D در خروجی شیفت رجیسترها استفاده شده است. استفاده از این ساختار باعث می‌شود تا حتی با تعداد طبقات کمتر به درجه امنیت بهتری برسیم. در این طرح، ما از پنج نوسان‌ساز حلقوی هرکدام با ۱۷۵ گیت معکوس‌کننده استفاده کردیم. نکته مهم این است که استفاده از همین نوسان‌سازهای حلقوی نیز کار را برای مهاجم هر چه دشوارتر و یا غیرممکن می‌کند زیرا چنانچه قبلاً توضیح داده شد ساختار نوسان‌ساز حلقوی نیز نوع دیگری از تابع کپی‌ناپذیر فیزیکی است که در صورت بزرگ بودن تعداد عناصر حلقه حتی با معلوم بودن ساختار و مشخص بودن تعداد گیت‌ها و نوسان‌سازها امکان کپی‌سازی از آن برای مهاجم به‌سادگی وجود ندارد. با تعداد نوسان‌سازهای مشخص‌شده، خروجی با فرکانسی حدود ۱/۶ مگاهرتز بر روی Spartan3AN XC3S700AN نوسان می‌کند. برای تولید اعداد تصادفی از خروجی با فرکانس ۱/۵ مگاهرتز نمونه‌برداری و ۱۰۲۴ بیت اولیه تولیدشده را دور ریختیم.



شکل (۸): طراحی یک مولد اعداد تصادفی با استفاده از نوسان سار حلقوی [۳۰].



شکل (۹): مولد اعداد تصادفی بهبودیافته مبتنی بر نوسان‌سازهای حلقوی.

لازم به ذکر است که یکی از چالش‌های مهم در تحقق عملی نوسان‌سازهای حلقوی، پیاده‌سازی آن با زبان توصیف سخت‌افزار است. مشکل اصلی در استفاده از گیت‌های معکوس‌کننده به‌صورت سری است که ابزار سنتز آن‌ها را به‌عنوان یک منطق بی‌اثر در نظر گرفته و حذف می‌کند زیرا ترکیب سری دو معکوس‌کننده از نظر منطقی یک ترکیب بی‌اثر است که در پیاده‌سازی از سوی ابزار سنتز حذف می‌شود. استفاده از دستور 'KEEP' در زبان توصیف سخت‌افزار باعث می‌شود تا ابزار سنتز از حذف گیت‌هایی که در ادامه این دستور قرار می‌گیرند خودداری کند و از این‌رو، برای پیاده‌سازی ساختار نوسان‌ساز حلقوی استفاده از این دستور حتماً لازم است. شکل (۱۰) کد پیاده‌سازی یک نوسان‌ساز حلقوی با استفاده از این دستور را نشان می‌دهد که در آن از ۱۷۵ گیت معکوس‌کننده در یک ساختار حلقوی استفاده شده است. علاوه‌بر آن، از یک گیت AND برای کنترل شروع نوسانات استفاده شده است. نتایج به‌دست‌آمده پس از سنتز مدار بر روی تراشه نشان داد که چنین نوسان‌سازی موج

[1] Jittery Clock Pulse
[2] Tkacik
[3] Epstein
[4] Golic-Figaro
[5] Kohlbrenner-Gaj

مربعی با پریود تقریبی ۶۲۰ نانوثانیه تولید خواهد کرد. در تمام شبیه‌سازی‌های انجام‌شده پایه Enable تعریف شده است. با یک شدن پایه Enable، نوسان‌کننده‌های طراحی‌شده هم‌زمان با هم شروع به نوسان می‌کنند که این کار برای هم‌زمانی شروع نوسان‌کننده‌ها بسیار مهم می‌باشد.

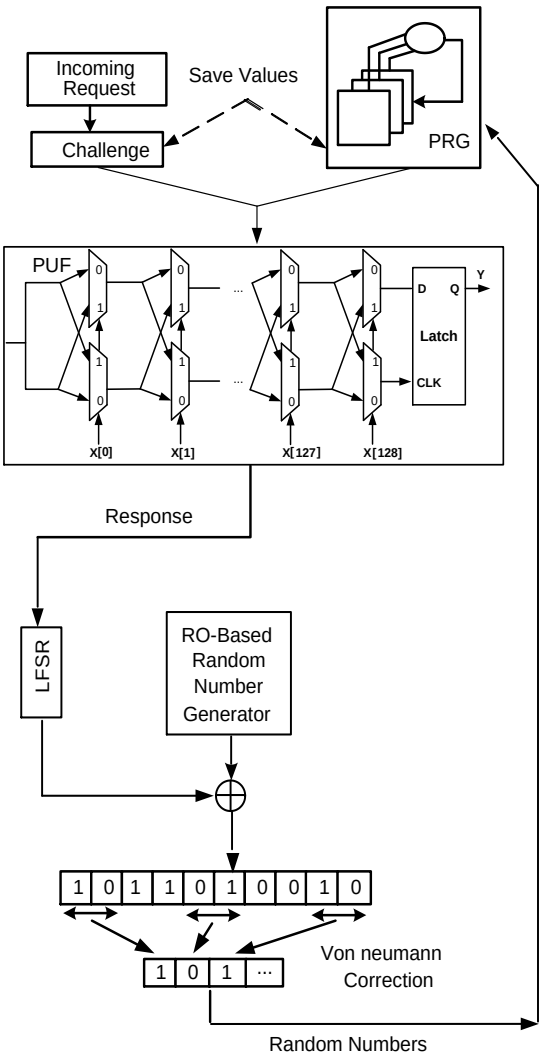
```
module oscillator(
input enable,
output osc
);
(*KEEP="TRUE"*) wire [0:174] im;
and g1(im[0], im[174], enable);
not g2(im[1], im[0]);
.....
not g73(im[174], im[173]);
not g74(osc, im[174]);
endmodule
```

شکل (۱۰): پیاده‌سازی یک نوسان‌ساز حلقوی متشکل از ۱۷۵ گیت معکوس‌کننده با استفاده از زبان توصیف سخت‌افزار ویلاگ.

پس از پیاده‌سازی طرح مولد اعداد تصادفی مورد نظر، بیت‌های تولیدشده را از تصحیح‌کننده وان نیومن عبور داده و ۶۴ بیت خروجی را ذخیره کردیم. شکل (۱۱) نحوه استفاده از تصحیح کننده وان نیومان برای تصحیح دنباله خروجی را نشان می‌دهد. این تصحیح‌کننده بیت‌های تکراری را از بین می‌برد و وقتی بیت‌ها متفاوت هستند تنها یکی از بیت‌ها را نگه می‌دارد. درحالی‌که عملگر XOR نصف بیت‌ها را حذف می‌کند تصحیح‌کننده وان نیومن تقریباً سه‌چهارم بیت‌ها را دور می‌ریزد. جدول (۱) عملکرد این تصحیح‌کننده را در مقایسه به عملگر XOR نشان می‌دهد.

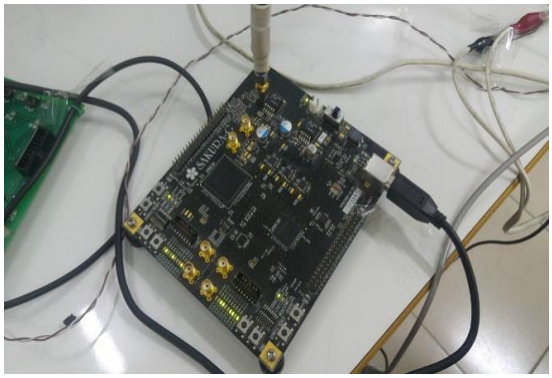
جدول (۱): مثالی از نحوه عملکرد تصحیح‌کننده‌های XOR و وان نیومن.

Sequence	10	11	00	10	10	00	01	10	01	01
XOR	1	0	0	1	1	0	1	1	1	1
Von Neumann	1			1	1		0	1	0	0

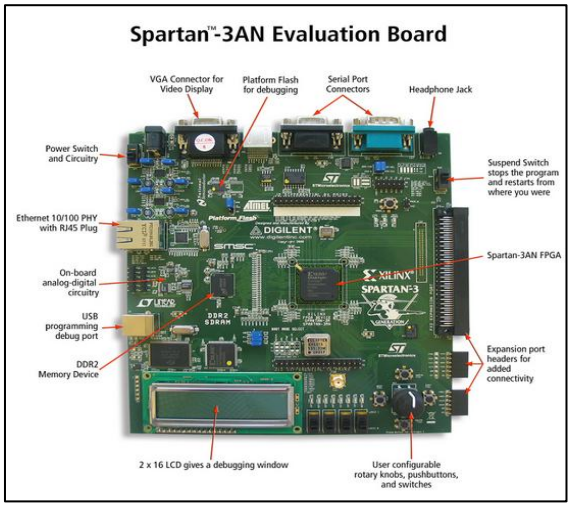


شکل (۱۱): شمای کلی طرح پیاده‌سازی شده شامل پاف، مولد اعداد تصادفی و تصحیح کننده وان نیومن.

بعد از بررسی اجمالی پاف و کاربردهای آن، نوبت به نحوه پیاده‌سازی و نتایج به‌دست‌آمده از اجرای طرح‌ها بر روی تراشه FPGA می‌رسد. ما در این تحقیق از سه بورد FPGA با تراشه‌های Spartan-6 XC6SLX75 و Spartan3 XC3S400 و Spartan3AN XC3S700AN برای پیاده‌سازی طرح استفاده نمودیم. شکل (۱۲) شمای بورد استاندارد حملات کانال جانبی موسوم به بورد ساکورا که تراشه Spartan-6 XC6SLX75 با فناوری ۴۵ نانومتر از شرکت Xilinx بر روی قرار گرفته و شکل (۱۳) بورد Spartan3A XC3S700AN را نشان می‌دهد که دارای تراشه XC3S700AN و دارای حافظه XCF02/04S برای برنامه‌ریزی FPGA است. ضمن این‌که این بورد مجهز به نوسان‌ساز، کانکتورها و سوئیچ‌های مختلف برای برنامه‌ریزی عملکرد آن در کاربردهای مختلف است.



شکل (۱۲): تصویر بورد استاندارد حملات کانال جانبی موسوم به بورد ساکورا حاوی تراشه Spartan-6 XC6SLX75 که در آزمایش‌های مورد استفاده قرار گرفت.

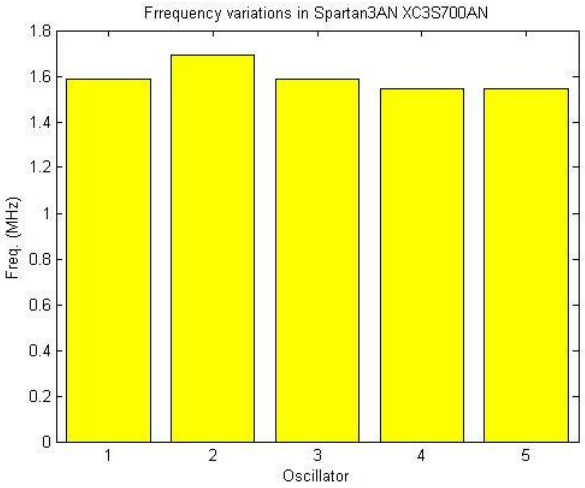


شکل (۱۳): تصویر بورد Spartan3A XC3S700AN از شرکت Xilinx.

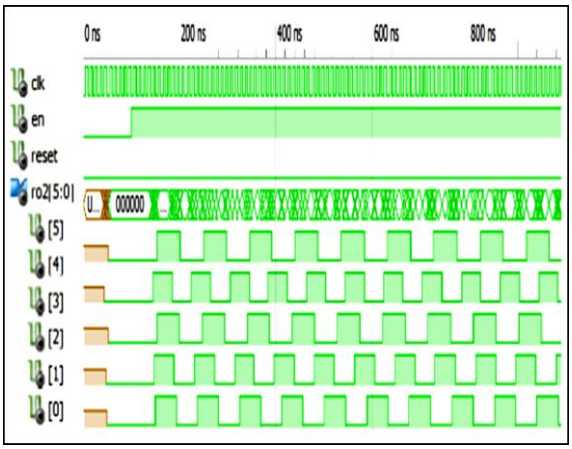
برای پیاده‌سازی مولد اعداد تصادفی ابتدا طرح نوسان‌ساز حلقوی بر روی FPGA پیاده‌سازی شد. پیاده‌سازی این طرح بر روی تراشه Spartan3AN XC3S700AN و فرکانس ۵ نوسان‌کننده در شکل (۱۴) به نمایش گذاشته شده است. این شبیه‌سازی در محیط ISIM در نرم‌افزار ISE 14.7 و با استفاده از یک رایانه شخصی بعد از مرحله جایابی و مسیریابی[1] انجام‌شده است. فرکانس نوسان‌کننده‌ها به ترتیب برابر با 1.54251MHz، 1.54449MHz، 1.58686MHz، 1.69231MHz ، 1.58862MHz می‌باشد. این اختلاف فرکانس در نوسان‌کننده‌ها به خاطر برابر نبودن تأخیر و تعداد گیت‌ها و مسیرهای سیم‌کشی می‌باشد. نکته مهم اینجاست که مهاجم حتی با دانستن تعداد دروازه‌ها و ساختار نمی‌تواند طرح را مجدداً بازسازی کند زیرا از خصوصیّات ذاتی تراشه که در حین فرآیند ساخت اتّفاق افتاده در پیاده‌سازی طرح استفاده شده است. دستور KEEP را باید تحت شرایط نحوه نوسان حلقه‌ها برای هر حلقه منظور نمود که این

[1] Place&Route

مسئله باعث می‌شود تأخیر دروازه‌ها و مسیرها به‌خوبی در شبیه‌سازی و پیاده‌سازی مورد استفاده قرار گیرد.



شکل (۱۴): فرکانس نوسان‌سازهای پیاده‌سازی‌شده بر روی بورد Spartan3AN XC3S700AN و اختلاف آن‌ها با یکدیگر.



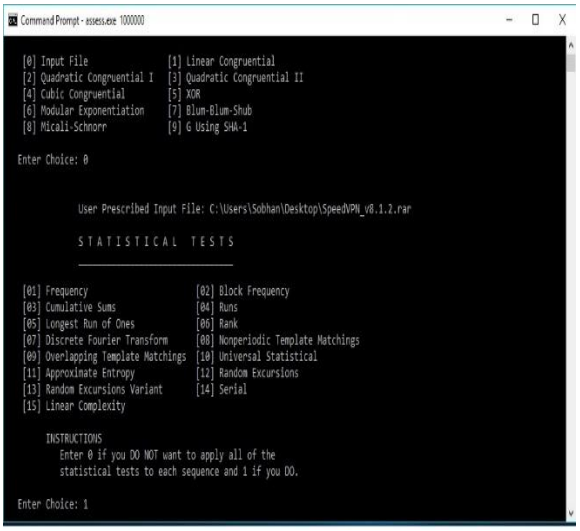
شکل (۱۵): اجرای نوسان‌ساز حلقوی در تراشه Spartan3AN XC3S700AN و اختلاف فرکانس نوسان‌سازها با یکدیگر.

۵-۳. نتایج آزمون‌های آماری NIST

کیفیت اعداد تصادفی، میزان امنیت و غیرقابل پیش‌بینی بودن آن‌ها را می‌توان با انجام برخی آزمون‌های آماری ارزیابی کرد. معمول‌ترین این آزمون‌ها، آزمون‌های آماری منتشرشده از طرف موسسه استاندارد و فناوری ایالات متحده[2] است که در مستندات FIPS-PUB-140-1 و FIPS-PUB-140-2 آمده است. ۲۲-۸۰۰ آخرین نسخه این آزمون‌ها بعد از چند بار اصلاح است که شامل ۱۵ آزمون اصلی است [۲۹]. مهم‌ترین این آزمون‌ها، آزمون فرکانس، آزمون سریال، آزمون ران‌ها[3]، آزمون رانه‌ای بلند[1] و

[2] National Institute of Standard and Technology
[3] Run Tests

آزمون تک بیت است. آزمون فرکانس مشخص‌کننده متعادل بودن تعداد صفرها و یک‌ها در دنباله است. آزمون ران مربوط به تعداد بیت‌های یکسان در کنار یکدیگر است و تعداد زیر دنباله‌های تمام صفر یا تمام یک در دنباله اصلی بررسی می‌کند. آزمون سریال، فرکانس تمام دنباله‌های m بیتی دارای هم‌پوشانی در دنباله اصلی را بررسی می‌کند به شکلی که تعداد 2n دنباله دارای m بیت هم‌پوشانی را تخمین می‌زند. برای انجام این آزمون لازم است تا طول دنباله ورودی $m < [\log 2n] - 2$ باشد. ما برای بررسی تصادفی بودن دنباله خروجی از نرم‌افزار استاندارد آزمون‌های NIST که پنجره آن در شکل (۱۶) نشان داده شده استفاده کردیم [۳۰]. جدول (۲) بخشی از نتیجه آزمایش‌های انجام‌شده بر روی دنباله خروجی را نشان می‌دهد.



شکل (۱۶): پنجره برنامه آزمون‌های آماری تصادفی بودن دنباله خروجی که از سوی موسسه استاندارد و فن‌آوری تهیه شده و در معرض دسترس است.

جدول (۲): بخشی از نتایج به‌دست آمده آزمون‌های NIST برای دنباله تولیدشده.

آزمون	مقدار-P
فرکانس	۰,۴۴۸۴۲۴
فرکانس بلاک ها	۰,۵۲۲۱۰۰
ران	۰,۷۹۴۳۹۱
بلندترین ران	۰,۷۴۳۹۱۵
سریال	۰,۴۹۸۳۱۳
پیچیدگی خطی	۰,۱۵۰۳۴۰
یونیورسال	۰,۹۱۴۰۲۵
تبدیل فوریه گسسته	۰,۰۲۳۲۲۸

[1] Long Run Tests

۵-۴. نتایج پیاده‌سازی سخت‌افزاری

جدول (۳) نتایج پیاده‌سازی طرح مورد نظر بر روی تراشه‌های مورد نظر از حیث مساحت اشغالی و منابع سخت‌افزاری مورد استفاده را نشان می‌دهد. منابع سخت‌افزاری بر حسب تعداد LUTهای چهار ورودی و نیز تعداد بلوک‌های منطقی آرایش‌پذیر[۲] بیان شده‌اند. از آنجا که در جستجوهای انجام‌شده موردی دقیقاً مشابه با طرح پیشنهادی یافت نشد لذا نتوانستیم طرح خود را به‌طور مستقیم با سایر کارها مقایسه کنیم اما خوشبختانه گزارش ابزار سنتز نشان داد که طرح مورد نظر مساحت معقول و قابل قبولی را روی تراشه‌ها اشغال می‌کند که از این حیث پیاده‌سازی آن کاملاً منطقی و معقول به نظر می‌رسد.

جدول (۳): منابع سخت‌افزاری استفاده‌شده در پیاده‌سازی سخت‌افزاری طرح پاف مورد نظر بر روی FPGA.

FPGA	Area (Slices)	Available	Area (LUTs)	Available
Spartan 6	۱۷۴۹	۱۱۶۶۲	۶۷۰۰	۴۶۶۴۸
Spartan3A	۵۱۴	۷۰۴	۱۰۲۶	۱۴۰۸
Spartan 3	۵۱۰	۷۶۸	۹۸۵	۱۵۳۶

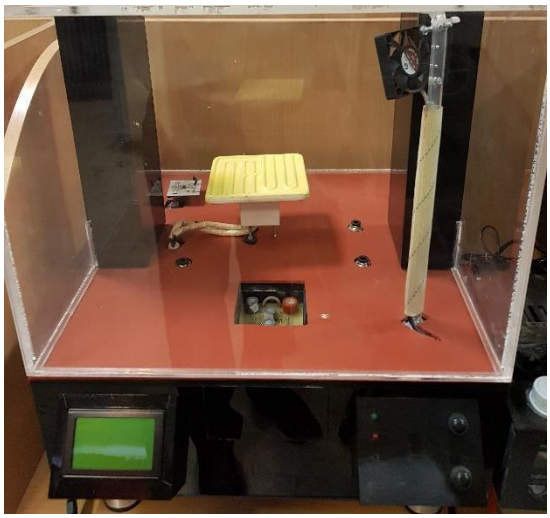
۵-۵. مشخصات شناسه و اثر دما بر عملکرد پاف

همان‌طور که گفته شد در مرحله آخر پیاده‌سازی‌های انجام‌شده بر روی بوردهای مختلف از نظر منحصربه‌فرد بودن و قابل اعتماد بودن مورد مقایسه قرار گرفت. شناسه ایجادشده در هر کدام از تراشه‌های مورد آزمایش با تراشه دیگر متفاوت بوده و این تفاوت بر اساس خصوصیّات فیزیکی تراشه می‌باشد. آزمایش‌های ما نشان داد که چنانچه همین طرح بر روی یک تراشه با شماره دقیقاً یکسان و از همان کارخانه سازنده پیاده‌سازی شود باز هم کد شناسه به‌دست‌آمده متفاوت و منحصربه‌فرد خواهد بود. در واقع با این طرح می‌توان برای هر تراشه یک کد شناسه منحصربه‌فرد تعریف کرده و آنرا از سایر تراشه‌ها متمایز نمود. از این مفهوم می‌توان برای ایجاد اعتماد در محاسبات راه دور، محاسبات مطمئن[۳]، کنترل دسترسی، احراز اصالت تراشه‌ها و یا تولید کلید خصوصی منحصربه‌فرد برای هر تراشه در محاسبات کلید عمومی و تسهیم راز استفاده کرد. نکته مهم دیگر بررسی اثر دما بر عملکرد پاف است که در این تحقیق پس از ۲۵ بار اندازه‌گیری شناسه با ۱۰ درجه سانتی‌گراد اختلاف دما، حداکثر ۱۰٪ خطا در بیت‌های خروجی نسبت به دمای اتاق به دست آمد. در صورت در اختیار داشتن فضای لازم روی تراشه استفاده از کدهای تصحیح خطا می‌تواند به کاهش احتمال خطا کمک کند زیرا در کاربردهایی که نیاز به بازتولید شناسه وجود دارد پایداری و قابلیت اطمینان پاف امر مهمی است که نمی‌تواند نادیده گرفته

[2] CLB Slices
[3] Trusted Computing

شود حال آنکه ممکن است این مسئله در تولید اعداد تصادفی مسئله چندان مهمی نباشد. لازم به ذکر است که قابلیت اطمینان و امنیت از جمله مهم‌ترین معیارهای ارزیابی کیفیت پاف‌ها هستند. دما و تشعشعات الکترومغناطیسی از جمله تأثیرگذارترین عوامل بر عملکرد پاف هستند [۲۳].

برای محاسبه قابلیت اطمینان طرح مورد نظر، بورد FPGA شامل مدار پاف پیاده‌سازی‌شده را در محفظه حرارتی نشان داده شده در شکل (۱۷) که دارای قابلیت کنترل دما به‌صورت دیجیتال است قرار دادیم. این محفظه به‌طور کامل در داخل کشور توسط دانشجویان طراحی و ساخته شده است.



شکل (۱۷): محفظه حرارتی با قابلیت کنترل دما به‌صورت دیجیتال با حساسیت یک درجه سانتی‌گراد.

مطابق با آزمایش‌های انجام‌شده، با تغییر دما از ۲۰ تا ۶۰ درجه سانتی‌گراد تنها یک بیت خطا در خروجی اتفاق افتاد و از این‌رو قابلیت اطمینان طرح حدود ۹۰ درصد به‌دست می‌آید. لازم به ذکر است که مطابق با آنچه در منابع مرتبط ذکرشده پاف داور معمولاً از قابلیت اطمینان کاملاً قابل قبولی برخوردار است و در برابر تغییر دما مقاوم است اما برای استفاده از یک پاف در شرایط عملیاتی لازم است تا آزمایش مورد نظر به شکلی که توضیح داده شد با دستگاهی مجهزتر تکرار شود.

۶. نتیجه‌گیری

وقایع و گزارش‌های اخیر در حوزه امنیت به‌روشنی نشان می‌دهد که تراشه‌ها و ابزارهای محاسباتی به‌صورت عمده، گسترده، بدون نظارت و در معرض انواع حملات فیزیکی هستند. ذخیره اطلاعات دیجیتال به شکل مطمئن در یک ابزار کار ساده‌ای نیست و از این‌رو، توابع کپی‌ناپذیر فیزیکی و سازوکارهای مشابه مهم هستند. در این تحقیق چند هدف مورد توجه قرار گرفت. اول آنکه دو نوع از کاربردی‌ترین انواع توابع کپی‌ناپذیر فیزیکی موسوم به پاف داور و پاف نوسان‌ساز حلقوی به‌صورت عملی پیاده‌سازی شده و مورد آزمایش قرار گرفتند. علاوه‌بر آن، یک کد تصادفی منحصربه‌فرد و غیرقابل کپی‌سازی برای شناسایی تراشه تولید شد که می‌توان از آن در کاربردهای احراز هویت و اصالت سخت‌افزار استفاده کرد و سوم آنکه از شمای پیاده‌سازی شده می‌توان برای تولید اعداد تصادفی و ایجاد کلید بدون ذخیره‌سازی در ابزار نیز استفاده نمود. ادامه این تحقیق حول موضوعاتی که به آن اشاره شد می‌تواند از موضوعات جذاب تحقیقاتی برای محققین مرتبط با حوزه امنیت و سخت‌افزار باشد.

۷. مراجع

[1] R. Maes, "Physically Unclonable Functions: Constructions, Properties and Applications," Ph. D. thesis, Dissertation, University of KU Leuven, 2012.

[2] H. Handschuh, S. Geert-Jan, and P. Tuyls, "Hardware Intrinsic Security from Physically Unclonable Functions," Parts of Towards Hardware-Intrinsic Security, Springer Berlin Heidelberg, pp. 39-53, 2010.

[3] M. Platonov, "SRAM-Based Physical Unclonable Function on an Atmel ATmega Microcontroller," Master's thesis, Czech Technical University in Prague, Faculty of Information Technology, 2013.

[4] V. Van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls, "Hardware Intrinsic Security from D Flip-Flops," In ACM Workshop on Scalable Trusted Computing—STC 2010, New York: ACM, pp. 53–62, 2010.

[5] J.-L. Zhang, "A Survey on Silicon PUFs and Recent Advances in Ring Oscillator PUFs," Journal of Computer Science and Technology, vol. 29, no. 4, pp. 664–678, July 2014. DOI 10.1007/s11390-014-1458-1.

[6] C.-H. Chang, Y. Zheng, and L. Zhang, "A Retrospective and a Look Forward: Fifteen Years Feature Abstract of Physical Unclonable Function Advancement," IEEE Circuits and Systems Magazine, vol. 17, Issue 3, pp. 32-62, 2017.

[7] J. Guajardo, S. S. Kumar, G. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection," CHES 2007, LNCS 4727, pp. 63–80, 2007.

[8] N. Beckmann and M. Potkonjak, "Hardware-Based Public-Key Cryptography with Public Physically Unclonable Functions," In Lecture notes in computer science (LNCS): vol. 5806, International workshop on information hiding—IH 2009, Berlin: Springer, pp. 206–220, 2009.

[9] L. Bolotny and G. Robins, "Physically Unclonable Function-Based Security and Privacy in RFID Systems," In IEEE international conference on pervasive computing and communications—PERCOM 2007, New York: IEEE, pp. 211–220, 2007.

[10] F. Armknecht, R. Maes, A.-R. Sadeghi, B. Sunar, and P. Tuyls, "Memory Leakage Resilient Encryption Based on Physically Unclonable Functions," In Lecture Notes In Computer Science (LNCS): vol. 5912, Advances in Cryptology—ASIACRYPT 2009, Berlin: Springer, pp. 685–702, 2009.

[11] L. Lin, S. Srivathsa, D. K. Krishnappa, P. Shabadi, and W. Burleson, "Design and Validation of Arbiter-Based PUFs for Sub-45-nm Low-Power Security Aapplications. IEEE Trans. Inf. Forensics and Security, vol. 7, no. 4, pp. 1394-1403, 2012.

[12] M. Z. Shahrak, "Secure and Lightweight Hardware Authentication Using Isolated Physical Unclonable Function," MS Thesis, University of Nebraska-Lincoln, 2016.

[13] J. Delvaux and I. Verbauwhede, "Side Channel Modeling Attacks on 65nm Arbiter PUFs Exploiting CMOS Device Noise," In Proc. IEEE Int. Symposium on Hardware-Oriented Security and Trust, pp. 137-142, 2013.

[14] M. Usmani, "Applications of Physical Unclonable Functions on ASICS and FPGAs," MS Thesis, University of Massachusetts Amherst, 2018.

[15] W. Stallings, "Cryptography and Network Security," 5th Ed., Pearson, 2014.

[16] S. Katzenbeisser, U. Kocaba, V. Rozic, A. R. Sadeghi, and I. Verbauwhede, and C. Wachsmann, "PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast In Silicon," In Proceedings of the 14th Int. Conference on Cryptographic Hardware and Embedded Systems (Berlin, Heidelberg, 2012), CHES'12, Springer-Verlag, pp. 283-301, 2012.

[17] J. Guajardo, S. S. Kumar, G. J. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP protection," In Lecture Notes In Computer Science (LNCS), vol. 4727, Workshop on Cryptographic Hardware and Embedded Systems—CHES 2007, Berlin: Springer, pp. 63–80, 2007.

[18] S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "Extended abstract: The Butterfly PUF Protecting IP on Every FPGA," In IEEE International Symposium on Hardware Oriented Security and Trust—HOST 2008, New York: IEEE, pp. 67–70, 2008.

[19] T. Machida, D. Yamamoto, M. Iwamoto, and K. Sakiyama, "A New Mode of Operation for Arbiter PUF to Improve Uniqueness on FPGA. In 2014 Federated Conference on Computer Science and Information Systems, pp. 871-878, Sept. 2014.

[20] R. Maes, A. Van Herrewege, and I. Verbauwhede, "PUFKY: A Fully Functional PUF-Based Cryptographic Key Generator," In Lecture notes in computer science (LNCS): vol. 7428, Workshop on Cryptographic Hardware and Embedded Systems—CHES 2012, Berlin: Springer, 2012.

[21] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," In Proc. The 44th ACM/IEEE Design Automation Conference, Jun. 2007, pp. 9-14, 2007.

[22] A. Cherkaoui, V. Fischer, A. Aubert, and L. Fesquet, "Comparison of Self-Timed Ring and Inverter Ring Oscillators as Entropy Sources in FPGAs," In Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1325-1330, 2012.

[23] S. Gujja, "Temperature Variation Effects on Asynchronous PUF Design Using FPGAs," PhD Thesis, University of Toledo, http://utdr-toledo.edu/theses-dissersions, 2014.

[24] K. B. Frikken, M. Blanton, and M. J. Atallah, "Robust Authentication Using Physically Unclonable Functions, P. Samarati et al. (Eds.): ISC 2009, LNCS 5735, pp. 262–277, 2009.

[25] M. Barbareschi, et al., "A PUF-based Hardware Mutual Authentication Protocol," J. Parrallel and Distributed Computing, 2018. https://doi.org/10.1016/j.jpdc.2018.04.007.

[26] T. E. Tkacik, "A Hardware Random Number Generator," Proc. of CHES 2002, pp. 450- 453, 2002.

[27] M. Epstein, L. K. Hars, H. Z. Raymond, "Design and Implementation of a True Random Number Generator Based on Digital Circuit Artifacts," CHES 2003, pp. 152-165, 2003.

[28] M. Dichtl and D. J. Golic, "High-Speed True Random Number Generation with Logic Gates Only," CHES 2007, 2007. https://iacr.org/archive/ches2007/47270045/47270045.pdf.

[29] W. P. Kohlbrenner, "The Design and Analysis of a True Random Number Generator in a Field Programmable Gate Array," Proc. of International Symposium on FPGAs, 2004.

[30] B. Sunar, W. J., Martin, and D. R. Stinson, "A Provably Secure True Random Number Generator with Built-in Tolerance to Active Attacks," IEEE Transactions on Computers, vol. 56, Issue 1, pp. 109-119, 2007.

[31] A. Rukhin, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," US National Institute of Standards and Technology (NIST), 2001.

[32] https://sourceforge.net/projects/randomanalysis/