ارائه معماری سبک‌وزن برای رمز احراز اصالت‌شده COLM و پیاده‌سازی آن روی بسترهای FPGA و ASIC

محسن جهانبانی[۱]، دانشجوی دکتری ریاضی-رمز؛ زین‌العابدین نوروزی[۲]، استادیار

۱- دانشکده و پژوهشکده مهندسی فن‌آوری اطلاعات و ارتباطات - دانشگاه جامع امام حسین (ع)- تهران- ایران- mjahanbani@ihu.ac.ir
۲- دانشکده و پژوهشکده مهندسی فن‌آوری اطلاعات و ارتباطات- دانشگاه جامع امام حسین (ع)- تهران- ایران- znorozi@ihu.ac.ir

چکیده: طرح‌های رمزنگاری احرازاصالت‌شده دو سرویس محرمانگی و جامعیت را هم‌زمان فراهم می‌کنند. مسابقه سزار با هدف طراحی این رمزها در حال برگزاری است. یکی از معیارهای انتخاب طرح نهایی این مسابقه در کنار امنیت، عملکرد سخت‌افزاری کاندیداها در محیط‌های با منابع محدود است. در این مقاله برای اولین‌بار برای رمز احراز اصالت‌شده COLM از دور نهایی مسابقه سزار، یک معماری سخت‌افزاری سبک‌وزن ۸-بیتی و سازگار با واسط برنامه‌نویسی کاربردی نسخه ۲ ارائه شده‌است. به‌دلیل این‌که طرح COLM از رمز AES به‌عنوان اولیه استفاده می‌کند، از معماری سبک‌وزن Atomic-AES سازگار شده با قوانین مسابقه سزار استفاده شده‌است. برای کاهش منابع سخت‌افزاری مصرفی از تکنیک‌هایی مانند پیاده‌سازی یک هسته AES برای رمزنگاری/رمزگشایی طرح، به اشتراک‌گذاری ثبات‌ها و پیاده‌سازی دوبرابرکردن روی میدان ($GF(2^{128})$) با ساختار ۸-بیت و ساختن ضرب‌های مرتبه بالاتر از آن، استفاده شده‌است. معماری پیشنهادی طرح COLM روی بسترهای ASIC و FPGA پیاده‌سازی شده‌است. این معماری برای دو بستر فوق مشابه بوده ولی از تکنیک‌های بهینه‌سازی نگاشت تکنولوژیکی برای هر بستر استفاده شده‌است. مقایسه نتایج این‌کار با پیاده‌سازی‌های پایه نشان می‌دهد که ناحیه مصرفی در FPGA، ۶۲٪ و در ASIC، ۷۴٪ کاهش داشته‌است. هم‌چنین اختصاصی‌نمودن API v۲ برای عرض داده ۸-بیت ناحیه مصرفی API را به‌ترتیب به میزان ۸٪ و ۶٪ روی بستر FPGA و ASIC کاهش داده است.

واژه‌های کلیدی: طرح‌های رمزنگاری احرازاصالت‌شده، پیاده‌سازی سبک‌وزن، مسابقه سزار، COLM، FPGA، ASIC .

Lightweight Architecture for COLM Authenticated Ciphers and Implementation on FPGA and ASIC

Mohsen Jahanbani[1], PhD student; Zeinolabedin Norouzi [2], Assistant professor

1- Imam Hossein Comprehensive University, Tehran, Iran, Email: mjahanbani@ihu.ac.ir
2- Imam Hossein Comprehensive University, Tehran, Iran, Email: znorozi@ihu.ac.ir

Abstract: Authenticated encryption schemes provide both confidentiality and integrity services, simultaneously. The CAESAR competition is being held with the aim of designing this cipher. An important criterion for selecting the final portfolio, besides security, is the hardware performance of the candidate in the environments with limited resource. In this paper, for the first time for COLM authenticated ciphers from the final round of the CAESAR, an 8-bit lightweight architecture have been presented, which is compatible with API v2. Since COLM scheme uses AES cipher as a primitive, lightweight architecture of Atomic-AES has been selected and adopted according to the API rules. Furthermore, to reduce the area in the hardware implementation, several techniques are used, including implementing one AES core in the datapath, sharing of registers and implementation doubling on the GF (2^{128}) with 8-bit architecture for constructing the higher-order multipliers. Proposed architecture of COLM is implemented on ASIC and FPGA platforms. This architecture is similar in both platforms, but different technology mapping optimization techniques are used for each platform. Comparing the results with 128-bit implementations shows that the area on FPGA and ASIC is reduced by 62% and 74%, respectively. Also, the customized API v2 for 8-bit data width reduced the API area by 8% and 6% on the FPGA and ASIC platforms, respectively.

Keywords: authenticated encryption schemes, lightweight implementation, CAESAR competition, COLM, FPGA, ASIC.

تاریخ ارسال مقاله: ۱۳۹۷/۰۳/۰۹
تاریخ اصلاح مقاله: ۱۳۹۷/۰۵/۲۳
تاریخ پذیرش مقاله: ۱۳۹۷/۰۸/۲۹
نام نویسنده مسئول: زین‌العابدین نوروزی
نشانی نویسنده مسئول: ایران- تهران-اتوبان بابایی- دانشگاه جامع امام حسین (ع) - دانشکده و پژوهشکده مهندسی فن‌آوری اطلاعات و ارتباطات- گروه مخابرات رمز

۱- مقدمه

طرح‌های رمزنگاری احراز اصالت‌شده (AE[1])، دو ویژگی محرمانگی و احراز اصالت را هم‌زمان فراهم می‌کنند. در محرمانگی، دشمن نمی‌تواند متن رمزشده را از یک رشته بیت شبه‌تصادفی تمایز دهد. احراز اصالت پیام نیز شامل دو بخش جامعیت به‌معنی عدم‌تغییر پیام در زمان ارسال و احراز اصالت فرستنده است. طرح‌های AE این دو ویژگی را ترکیب و ارتباط امن و احراز اصالت‌شده را بین دو بخش برقرار می‌کنند. ورودی یک طرح AE، کلید (K)، تک‌شمار[2] (N)، داده‌همراه[3] (AD) و متن اصلی (M) است. خروجی AE اغلب متن رمزشده (C) و برچسب (T) است. رمزگشایی، معکوس فرآیند فوق بوده و مقادیر K، N، AD، C و T را دریافت و مقدار M را ارائه می‌دهد.

در روش سنتی طراحی AE از ترکیب دو الگوریتم مجزا استفاده می‌شد که به‌علت دوگذری‌بودن و نیاز به دو کلید مجزا، دارای کارایی پایین است. روش‌های جدید طراحی AE، مبتنی‌بر مدهای رمز قالبی، جریانی، اسفنجی و طرح‌های اختصاصی هستند. در طرح‌های AE مبتنی‌بر مدهای رمز قالبی، یکی از رمزهای قالبی مانند AES در یک مد خاص استفاده می‌شود. برای مثال استاندارد SP800-38C [۱] مد CCM را با الگوریتم AES-۱۲۸ پیشنهاد می‌دهد. در استاندارد SP800-38D [۲] مد GCM برای نیازهای با گذردهی بالا توصیه شده‌است. در حال حاضر از هر دو این مدها به‌صورت گسترده استفاده‌می‌شود، ولی هیچ‌کدام از این طرح‌ها، مدرن، کارآمد و چندمنظوره نیستند.

یک رقابت جدید برای طراحی طرح‌های AE با حمایت NIST به نام مسابقه سزار[4] [۳] در ژانویه ۲۰۱۳ با دو هدف گستردگی کاربرد و برتری نسبت به طرح AES-GCM آغاز شده‌است. این مسابقه در چهار مرحله طراحی شده‌است که با ۵۷ طرح شروع و اکنون هفت طرح به مرحله نهایی آن راه‌یافته‌اند. از این هفت طرح، چهار طرح از مد رمز قالبی، یک طرح براساس مد اسفنجی و دو طرح اختصاصی هستند.

پیاده‌سازی‌های اولیه برای بسیاری از طرح‌های مسابقه سزار توسط طراحان آن‌ها ارائه شده‌است. اما این پیاده‌سازی‌ها معمولاً بهینه نبوده و برای تمامی نسخه‌های توصیه‌شده یک طرح و بسترهای سخت‌افزاری متفاوت ارائه نشده‌است. یکی از معیارهای سخت‌افزاری که توسط کمیته مسابقه اعلام شد، کارایی طرح در وسایل با منابع محدود حافظه، توان و انرژی مانند حسگرهای بی‌سیم شبکه [۴] و RFID-ها است. بنابراین ارزیابی کارایی این طرح‌ها در محیط‌های محدود یکی از زمینه‌های مناسب برای تحقیق و پژوهش است.

در سال ۲۰۱۶ یک چارچوب کلی در قالب یک واسط برنامه‌نویسی کاربردی سخت‌افزاری (API) [۵] برای پیاده‌سازی طرح‌های مسابقه سزار ارائه شد. این واسط دارای دو هدف ارزیابی عادلانه طرح‌های AE و سازگاری انواع پیاده‌سازی‌های مرتبط با یک طرح خاص است. نسخه اول این واسط [۶] برای پیاده‌سازی‌های سرعت‌بالا و نسخه دوم [۷] شامل دو واسط مجزا برای پیاده‌سازی‌های سرعت‌بالا و سبک‌وزن است.

۱-۱- نوآوری مقاله

تحلیل تمام کاندیداهای دور نهایی مسابقه سزار نشان می‌دهد که سه کاندیدا، از الگوریتم AES-۱۲۸ ده دوری به‌عنوان رمز قالبی پایه استفاده می‌کنند. از میان این سه طرح، طرح COLM به‌دلیل ویژگی‌های خاص امنیتی، برای ارزیابی عملکرد در محیط‌های محدود انتخاب و برای آن یک معماری سبک‌وزن ۸-بیت و سازگار API v۲ ارائه شده‌است. به‌دلیل این‌که طرح COLM نیاز به پیاده‌سازی AES دارد از معماری Atomic-AES [۸] که یکی از کوچک‌ترین معماری‌های ارائه‌شده برای AES می‌باشد، استفاده شده‌است. به‌دلیل غیرمعمول‌بودن ورودی و خروجی این معماری و عدم سازگاری آن با قوانین API سزار، در این‌کار معماری کلی طرح COLM به‌نحوی تغییریافته که این تطابق بدون هزینه سخت‌افزاری انجام گیرد.

جهت به‌کارگیری حداقل تعداد منابع از راه‌کارهایی مانند کاهش ناحیه مصرفی API v۲ با تبدیل آن به معماری ۸-بیت، پیاده‌سازی یک AES در هسته رمزنگاری/رمزگشایی و به‌اشتراک‌گذاری ثبات‌ها استفاده شده‌است. پیاده‌سازی دوبرابرکردن روی میدان ($GF(2^{128})$) مناسب با ساختار غیرمعمول (ترانهاده) خروجی AES انجام شده و برای ساخت ضرب‌های مرتبه بالاتر به‌کار گرفته شده‌است. جهت کاهش ناحیه مصرفی روی بستر FPGA از پیاده‌سازی S-box به‌صورت ROM به‌جای پیاده‌سازی منطقی ارائه‌شده در [۸] استفاده شده‌است.

۱-۲- ساختار مقاله

در ادامه ساختار مقاله به این شرح است: در بخش ۲ مروری بر کارهای مرتبط و طرح COLM انجام شده‌است. در بخش ۳، روش پیاده‌سازی طرح COLM، نحوه انتخاب و اجرای AES بیان شده‌است. جزییات معماری پیشنهادی برای طرح در بخش ۴ تشریح شده‌است. در بخش ۵، نتایج پیاده‌سازی و مقایسه با کارهای قبلی انجام گرفته‌است. نتیجه‌گیری نهایی و کارهای آتی نیز در بخش ۶ ارائه شده‌است.

۲- ادبیات موضوع

در این بخش تاریخچه پیاده‌سازی‌های سبک‌وزن مرتبط، مرور مفاهیم طرح COLM و تکنیک‌های پیاده‌سازی سبک‌وزن ارائه شده‌است.

۲-۱- کارهای مرتبط

برای پیاده‌سازی طرح‌های مسابقه سزار یک چارچوب API توسط تیم گروه تحقیقات مهندسی رمز (CERG[5]) دانشگاه جرج میسون آمریکا [۵] ارائه شد. این تیم که در مسابقه قبلی SHA-۳ نیز فعال بود برروی وب‌سایت خود [۹] پایگاه داده‌ای از نتایج پیاده‌سازی FPGA طرح‌های مسابقه سزار، گزارش نموده‌است. بهترین نتایج حاصله که با API سزار بوده، برای رتبه‌بندی طرح‌ها انتخاب و در [۱۰] ارائه شده‌است. نتایج گزارش‌شده نشان می‌دهد که به‌جز طرح ACORN، سایر طرح‌ها برروی معماری‌های سرعت‌بالا تمرکز داشته‌اند.

تاکنون تعداد کمی پیاده‌سازی سبک‌وزن برای طرح‌های مسابقه سزار ارائه شده‌است. بنیک و همکاران [۱۱] دو طرح CLOC، SILC از دور سوم مسابقه سزار را به‌صورت فشرده، با دو معماری تهاجمی و محافظه‌کارانه پیاده‌سازی کرده‌اند. معماری‌های ارائه‌شده با واسط سخت‌افزاری API سازگار نبوده و از محدودیت‌هایی مانند محدودسازی تعداد بلوک متن اصلی و پیش‌لایه‌گذاری[۶] ورودی به‌منظور کاهش ناحیه مصرفی استفاده کرده‌است. در معماری تهاجمی با فرض خالی‌بودن داده‌همراه، کامل بودن بلوک‌های متن اصلی و ذخیره‌سازی مقادیر میانی به‌صورت برون‌خط پیاده‌سازی تا حد ممکن فشرده شده‌است. در نسخه محافظه‌کارانه داده‌همراه یک بلوک است اما نیازی به کامل‌بودن بلوک متن اصلی و ذخیره برون‌خط نیست.

تنها تحقیق مرتبط با پیاده‌سازی طرح COLM [۱۲] طرح را از نقطه‌نظر توانایی پردازش موازی بلوک‌های AD، پیام یا متن رمزشده بررسی کرده‌است. هم‌چنین از تکنیک خط-لوله داخل دوری جهت کاهش مسیر بحرانی و افزایش فرکانس کاری و درنتیجه افزایش گذردهی به میزان ۶۰٪ استفاده شده‌است. اما استفاده از این تکنیک منجر به افزایش ۶۲٪ ناحیه نسبت به معماری پایه شده‌است.

اخیراً در [۱۳] یک تجزیه و تحلیل جامع برای کاندیداهای دوره سوم مسابقه سزار برروی بسترهای FPGA و ASIC از نظر عملکرد سخت‌افزاری، ارائه شده‌است. بدین منظور کدهای VHDL ارائه‌شده توسط تیم CERG یا تیم طراحان در بستر ASIC سنتز و تحلیل شده‌است. هم‌چنین برای سه طرح نیز براساس معماری‌های سخت‌افزاری ارائه‌شده توسط تیم CERG به‌صورت دستی بهینه‌سازی انجام و روی دو بستر ASIC و FPGA پیاده‌سازی شده‌است.

۲-۲- رمز احراز اصالت‌شده COLM

از بین کاندیداهای دور نهایی مسابقه سزار سه طرح AEGIS [۱۴] و OCB [۱۵] و COLM [۱۶]، از الگوریتم AES به‌عنوان رمز قالبی پایه استفاده می‌کنند. مقایسه ویژگی‌های این سه طرح در جدول ۱ نشان می‌دهد که طرح COLM دارای امنیت در برابر تکرارپذیری تک‌شمار[۷] تحت کلید واحد، امنیت در برابر انتشار متن اصلی بررسی‌نشده[۸] و پشتیبانی از برچسب‌های میانی است که اغلب طرح‌های AE فاقد چنین سطح امنیتی هستند. بنابراین احتمال انتخاب COLM به‌عنوان برنده نهایی بالاتر است.

جدول ۱: ویژگی‌های سه طرح AES-محور AE از مرحله نهایی مسابقه سزار

نام طرح	عدم نیاز به معکوس رمز قالبی	برچسب‌های میانی	تکرارپذیری تک‌شمار	انتشار متن اصلی بررسی‌نشده	موازی‌پذیر بودن رمزنگاری/رمزگشایی	امنیت اثبات‌پذیر
AEGIS	✓	×	×	×	رمزنگاری	×
OCB	×	×	×	×	کامل	✓
COLM	×	✓	✓	✓	کامل	✓

COLM یک مد رمز قالبی براساس ساختار رمز-مخلوط‌ساز خطی-رمز است که دارای طول کلید و حالت ۱۲۸-بیتی و تک‌شمار ۶۴-بیتی و سطح امنیتی درصورت تکرار یا عدم تکرارتک‌شمار برای محرمانگی و جامعیت، ۶۴ بیت است.

طرح COLM دارای دو نسخه .COLM و $COLM_{۱۲۷}$ است که توصیه اول طرحان، .COLM است. این طرح از تابع خطی مخلوط‌سازی ρ استفاده می‌کند که دارای دو ورودی $x, st \in \{0,1\}^{۱۲۸}$ و دو خروجی $y = x \oplus ۳.st$ و $st' = x \oplus ۲.st$ است. مراحل رمزنگاری .COLM به شرح زیر است (شکل ۱):

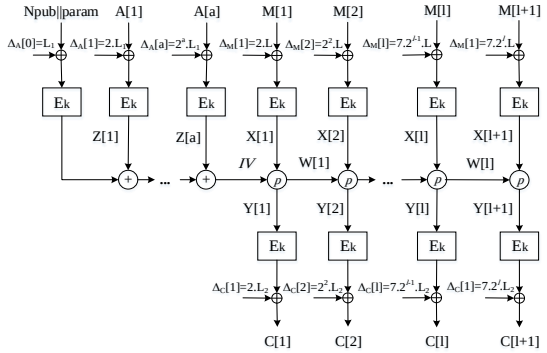
تولید زیرکلید: مقادیر $L = E_k(۰)$، $L_۱ = ۳.L$ و $L_۲ = ۳^۲.L$ به‌عنوان زیرکلید محاسبه می‌شود.

تولید مقدار اولیه (IV): نحوه محاسبه مقدار اولیه از AD (مقادیر پوشانه Δ_A در شکل ۱ نشان داده‌شده‌است):

$$\begin{aligned} &W'[0] = E_K((npub||param) \oplus \Delta_A[0]) \\ &\text{For i=1 to a} \\ &\quad W'[i] = E_K(A[i] \oplus \Delta_A[i]) \oplus W'[i-1] \\ &IV = W'[a] \end{aligned} \quad (۱)$$

تولید متن رمزشده برچسب‌دار: متن رمزشده برچسب‌دار از متن اصلی لایه‌گذاری‌شده و IV محاسبه می‌شود. متن اصلی به l قالب ۱۲۸-بیتی شکسته می‌شود. اگر $M = (M[1], \dots, M[l-1], M^*[l])$ و l-امین قالب برابر $M[l] = (M[1] \oplus \dots \oplus M[l-1] \oplus M^*[l]10^*)$ و $l+1$-امین قالب برابر $M[l+1] = M[l]$ و $W[0] = IV$، آن‌گاه برای i=1 تا $l+1$، متن رمزشده برچسب‌دار به‌صورت زیر محاسبه می‌شود:

$$\begin{aligned} &(Y[i], W[i]) = \rho(E_K(M[i] \oplus \Delta_M[i]), W[i-1]) \\ &C[i] = E_K(Y[i]) \oplus \Delta_C[i] \\ &C = (C[1], .., C[l], \lfloor C[l+1] \rfloor_{|M^*[l]|}) \end{aligned} \quad (۲)$$



شکل ۱: مراحل رمزنگاری طرح COLM [۱۶]

رمزگشایی و وارسی: این مرحله شامل تولید زیرکلید l و IV مانند مرحله رمزنگاری است. برای i=1 تا l:

$$\begin{aligned} &(X[i], W[i]) = \rho^{-1}(E_k^{-1}(C[i] \oplus \Delta_C[i]), W[i-1]) \\ &M[i] = E_k^{-1}(X[i]) \oplus \Delta_M[i] \\ &M^*[l] = M[1] \oplus \dots \oplus M[l],\ M[l+1] = M[l] \\ &X[l+1] = E_K(M[l+1] \oplus \Delta_M[l+1]) \\ &(Y[l+1], W[l+1]) = \rho(X[l+1], W[l]) \\ &C'[l+1] = E_K(Y[l+1]) \oplus \Delta_C[l+1] \end{aligned} \quad (۳)$$

وارسی با یکی از شرایط زیر موفقیت‌آمیز خواهدبود:

۱) اگر $|C[l+1]| = 128$، آنگاه $C[l+1] = C'[l+1]$.

۲) برای $|C[l+1]| < 128$، آنگاه $C[l+1] = \lfloor C'[l+1] \rfloor_{|C[l+1]|}$ و $|C[l+1]| - 128$ بیت آخر از $M^*[l]$، برابر 10^* باشد.

۳-۲- تکنیک‌های پیاده‌سازی سبک‌وزن الگوریتم‌های رمز AE

دو رویکرد برای پیاده‌سازی‌های سخت‌افزاری الگوریتم‌های رمز روی بسترهای FPGA و ASIC وجود دارد که عبارت‌اند از رویکرد پیاده‌سازی سرعت‌بالا و سبک‌وزن. تکنیک‌های پیاده‌سازی سبک‌وزن برای بخش‌های مسیر داده، کنترلی و منابع منطقی سخت‌افزاری قابل‌اعمال است. برای کاهش ناحیه مسیر داده از تکنیک‌هایی مانند به اشتراک‌گذاری منابع، پیاده‌سازی S-box با مدارات ترکیبی، انتقال محاسبات روی میدان متناهی از یک‌پایه به پایه دیگر و محاسبات در پرواز استفاده می‌شود. بخش کنترلی به‌صورت یک ماشین حالت متناهی (FSM[۹]) پیاده‌سازی می‌شود. ماشین‌های حالت بر اساس ROM ازلحاظ مصرف ناحیه و فرکانس کاری از دیگر ماشین حالت‌ها کارآمدتر هستند. مصرف ناحیه این ماشین به تعداد حالت‌ها و تعداد سیگنال‌های کنترلی وابسته است که کاهش تعداد آن، ناحیه مصرفی را کاهش می‌دهد. جزییات بیشتر برای تکنیک‌های پیاده‌سازی سبک‌وزن در [۱۷] ارائه شده‌است.

پس از سنتز مدار، گیت‌های استاندارد به منابع سخت‌افزاری هدف نگاشته می‌شوند. با شناخت منابع سخت‌افزاری هدف می‌توان این نگاشت را بهینه نمود. برای مثال در کتاب‌خانه سلول‌های استاندارد ASIC، یک نوع فلیپ-فلاپ به نام فلیپ-فلاپ پویشی وجود دارد که ثبات‌های ساخته‌شده از آن‌ها را می‌توان به چند طریق به‌هم متصل نمود و به‌وسیله یک سیگنال انتخاب‌گر، داده‌ها در دو جهت مختلف حرکت داد. این تکنیک‌ها برای پیاده‌سازی سبک‌وزن بسیاری از طرح‌های رمزنگاری استفاده شده‌است. به‌دلیل این‌که یکی از معیارهای ارزیابی سخت‌افزاری طرح‌های مسابقه سزار عملکرد آن‌ها در محیط‌های محدود سخت‌افزاری است از این تکنیک‌ها می‌توان برای پیاده‌سازی سبک‌وزن طرح‌های مسابقه سزار استفاده نمود.

۳- روش پیاده‌سازی

در این بخش نحوه استفاده و بهبود API، انتخاب معماری مناسب برای پیاده‌سازی AES و نحوه سازگار نمودن آن با API شرح داده می‌شود.

۳-۱- API سزار

API سزار [۵] به‌منظور تأمین نیازهای پیاده‌سازی سخت‌افزاری طرح‌های AE مسابقه سزار تعریف‌شده است. نسخه اول API [۶] فقط برای پیاده‌سازی سرعت‌بالا مناسب است. اما در نسخه دوم [۷] پشتیبانی از پیاده‌سازی‌های سبک‌وزن نیز اضافه‌شده است (شکل ۲). تفاوت نسخه سرعت‌بالا و سبک‌وزن، عرض داده ورودی پشتیبانی شده و سیگنال‌های داخلی هسته رمز است. تمام سیگنال‌های نشان داده‌شده به‌جز موارد ذکرشده یک بیتی هستند. بخش‌های مختلف این واسط عبارت‌اند از:

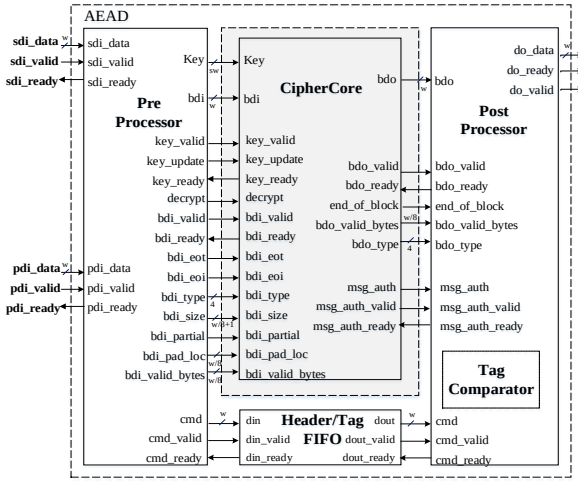
واحد پیش‌پردازنده: این واحد مسئول تجزیه سرآیند، بارگذاری کلید، انتقال قالب‌های ورودی به هسته رمز به‌همراه اطلاعات موردنیاز برای لایه‌گذاری و تعداد بلوک‌های پردازش‌نشده است. این واحد در حالت غیرثباتی کار می‌کند و دارای واحد لایه‌گذاری نیست.

واحد پس‌پردازنده: واحد پس‌پردازش بخش اضافی خروجی که در اثر لایه‌گذاری اضافه شده‌است را حذف می‌کند. هم‌چنین برای بلوک‌های خروجی، مقایسه تگ و تولید بلوک وضعیت باتوجه به نتایج احراز اصالت، سرآیند را تولید می‌کند.

واحد FIFO[۱۰]: از نوع FWFT[۱۱] و ۴ کلمه‌ای است که تمام بخش‌هایی از سرآیندکه به خروجی ارسال می‌شود راذخیره‌می‌کند.

واحد هسته رمز: هسته رمز شامل دو بخش داخلی مسیر داده و بخش کنترلی است که با هم مرتبط هستند. توسعه این هسته رمز برعهده طراحان سخت‌افزاری قرار دارد.

پروتکل ارتباطی: این پروتکل فرمت ورودی و خروجی‌های معمول یک طرح AE را توصیف می‌کند که شامل سه بخش دستورالعمل، سرآیند و داده ورودی است. داده‌های ورودی به بخش‌هایی با طول مشخص شکسته می‌شود. هر بخش از داده دارای یک سرآیند و یک دستورالعمل است که به‌ترتیب اطلاعاتی در مورد آن داده و نوع عملیاتی که روی آن انجام می‌شود را مشخص می‌کند.



شکل ۲: دیاگرام بلوکی معماری سبک‌وزن API v۲ [۶]

۳-۱-۱- بهبود در API v۲ برای معماری ۸-بیت

نسخه سبک‌وزن API v۲ [۷] برای پشتیبانی از عرض داده ۸، ۱۶ و ۳۲ طراحی شده‌است. بنابراین می‌توان با شخصی‌سازی این واسط برای طرح‌های ۸ بیت ناحیه مصرفی را کاهش داد. بهبودهای انجام‌شده بر روی API v۲ عبارت‌اند از:

- سرآیند داده‌های عمومی از ۳۲ بیت به ۱۶ بیت کاهش‌یافته به‌نحوی که ۸ بیت رزرو سرآیند حذف و اندازه بخش طول از ۱۶ بیت به ۸ بیت تغییر یابد. هم‌چنین بیت جزئی به‌دلیل عدم کاربرد، به بیت رزرو تبدیل شده‌است.
- دستورالعمل از ۱۶ بیت به ۸ بیت کاهش‌یافته به‌نحوی‌که ۸ بیت رزرو آن حذف شده‌است.

تأثیر این تغییرات بر ناحیه مصرفی در بخش ۵-۳ موردبررسی قرارگرفته است.

۳-۲- پیاده‌سازی سبک‌وزن رمز پایه AES

در پیاده‌سازی طرح COLM از الگوریتم AES به‌عنوان رمز قالبی پایه استفاده شده‌است. تاکنون چندین پیاده‌سازی سبک‌وزن AES روی بستر ASIC [۸، ۲۱-۱۸] و روی بستر FPGA [۲۴-۲۲] ارائه شده‌است. مقایسه کارایی این پیاده‌سازی‌ها در جدول ۲ نشان می‌دهد که پیاده‌سازی Atomic-AES [۸] یکی از کوچک‌ترین معماری‌های ارائه‌شده در بستر ASIC با عملکرد دوگانه رمزنگاری/رمزگشایی است. در این طرح S-box و معکوس آن در یک مدار پیاده‌سازی شده‌است که با یک انتخاب‌گر نوع عملیات انتخاب می‌شود. هم‌چنین برای معکوس مخلوط‌سازی ستونی مدار جداگانه‌ای طراحی نشده و با اجرای سه مرتبه مخلوط‌سازی ستونی محاسبه می‌شود.

جدول ۲: مقایسه نتایج پیاده‌سازی سبک‌وزن AES بر روی ASIC

مرجع	مسیر داده	عملیات	کتاب‌خانه	ناحیه (GE*)	گذردهی رمزنگاری	(Mbps) رمزگشایی
[۱۸]	۸ بیت	رمزنگاری/ رمزگشایی	Philips ۳۵۰ nm	۳۴۰۰	۹/۹	۸/۸
[۱۹]	۸ بیت	رمزنگاری/ رمزگشایی	۲۲ nm	۴۰۳۷	۴۳۲	۶۷۱
[۲۰]	۸ بیت	رمزنگاری	UMC ۱۸۰ nm	۲۴۰۰	۰/۰۶۱	۰/۰۶۱
[۲۱]	۸ بیت	رمزنگاری/ رمزگشایی	STM ۹۰ nm	۲۶۰۵	۹۴/۴	۹۴/۴
[۸]	۸ بیت	رمزنگاری/ رمزگشایی	STM ۹۰ nm	۲۰۶۰	۸۸/۴	۶۶/۷

*هر یک GE معادل مصرف ناحیه یک گیت NAND دو ورودی است.

تکنیک‌های بهینه‌سازی معماری و مداری برای بهینه‌سازی بستر FPGA متفاوت از ASIC است. یکی از مهم‌ترین تفاوت در تکنیک‌ها، نحوه پیاده‌سازی S-box در AES است به‌طوری‌که پیاده‌سازی منطقی در ASIC مناسب‌تر و پیاده‌سازی بر اساس ROM برای FPGA-ها مناسب‌تر است [۲۵]. برای انتخاب معماری سبک‌وزن AES برای بستر FPGA کدهای ارائه‌شده در [۸] دوباره سنتز شده‌است (جدول ۳). نتایج سنتز برای حالتی که S-box به‌صورت منطقی و ROM است با کارهای دیگر مقایسه شده‌است.

مطابق جدول ۳، اگرچه [۲۲،۲۳] دارای حداقل ناحیه مصرفی می‌باشند اما تنها از عملیات رمزنگاری پشتیبانی کرده و دارای گذردهی بسیار پایینی هستند. بنابراین پیاده‌سازی معماری [۸] به‌صورت ROM یکی از مناسب‌ترین معماری‌های سبک‌وزن مناسب AES برای FPGA است.

جدول ۳: مقایسه نتایج پیاده‌سازی سبک‌وزن AES بر روی FPGA

مرجع	مسیر داده	عملیات	نوع FPGA	Area (Slice)	F_{max} (MHz)	Tp (Mbps)
[۲۲]	۸ بیت	رمزنگاری	Virtex 6	۸۰	۴۵/۶	۳۶
[۲۳]	۸ بیت	رمزنگاری	Virtex 6	۲۱	۱۰۵	۹
سنتز مجدد [۸] (S-box منطقی)	۸ بیت	رمزنگاری/ رمزگشایی	Virtex 6	۱۵۹	۱۳۶	۷۰/۷
سنتز مجدد [۸] (S-box ROM)	۸ بیت	رمزنگاری/ رمزگشایی	Virtex 6	۱۳۶	۲۸۹	۱۵۰/۴

۳-۳- نحوه سازگاری Atomic-AES با API

معماری AES ارائه‌شده در [۸] به‌جای دریافت داده با ترتیب معمول، آن را به‌صورت ترانهاده دریافت و خروجی هر مرحله از AES نیز به‌صورت ترانهاده خارج می‌شود (شکل ۳).

۱	۲	۳	۴
۵	۶	۷	۸
۹	۱۰	۱۱	۱۲
۱۳	۱۴	۱۵	۱۶

الف) ترتیب ورودی معمولی

۱	۵	۹	۱۳
۲	۶	۱۰	۱۴
۳	۷	۱۱	۱۵
۴	۸	۱۲	۱۶

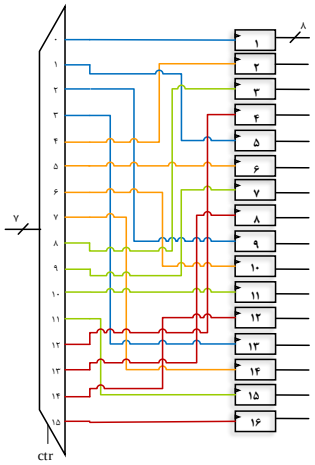
ب) ترتیب ورودی ترانهاده‌شده

شکل ۳: نحوه ورود داده Atomic-AES [۸]

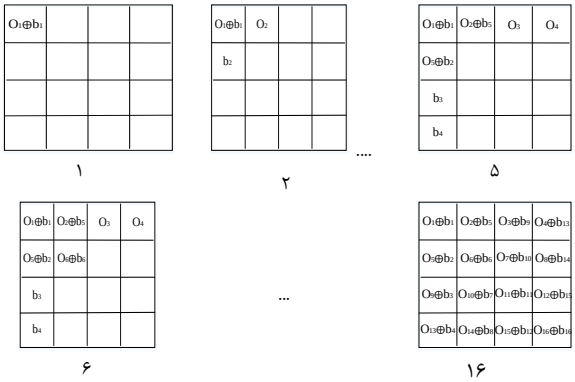
در [۱۳] ادعا شده‌است که پیاده‌سازی Atomic-AES [۸] به خاطر ورودی و خروجی غیرمعمول آن و قوانین API سزار مبنی بر عدم چرخش داده‌های ورودی، قابل‌استفاده نیست. بااین‌وجود دو راه‌کار ممکن برای حل این مشکل عبارت‌اند از: ۱) تعریف ثبات اضافی ۱۲۸-بیتی، ۲) تغییر در زمان‌بندی فراخوانی ورودی. راه‌کار اول منجر به مصرف حجم بیش‌تر به‌اندازه یک ثبات ۱۲۸-بیتی و ۳۲ کلاک (۱۶ کلاک صرف چرخش ورودی و ۱۶ کلاک صرف چرخش خروجی AES) اضافه به‌ازای هر AESمی‌شود. بنابراین باتوجه به استراتژی پیاده‌سازی سبک‌وزن، راه‌کار دوم انتخاب شده‌است. نحوه اجرای راه‌کار دوم به شرح زیر است:

اولین داده ورودی: با استفاده از یکی از ثبات‌های آزاد ۱۲۸-بیتی در معماری پیشنهادی، اولین داده ورودی که به‌صورت ۸ بیت از پیش‌پردازنده دریافت می‌شود، توسط یک دی‌مالتی‌پلکسر ۱:۱۶ در ۱۶ کلاک، در ۱۶ بایت این ثبات به‌ترتیب شکل ۴ ذخیره می‌شود.

سایر داده‌های ورودی: فراخوانی داده ورودی هر مرحله هم‌زمان با ذخیره‌سازی خروجی AES مرحله قبل از آن انجام می‌گیرد و از ثبات‌های آزاد خروجی AES برای چرخش داده استفاده می‌شود. اگر فرض شود که نحوه چیدمان بایت‌ها مشابه شکل ۳-ب باشد، نحوه ذخیره‌سازی داده ورودی، ذخیره‌سازی خروجی AES و زمان‌بندی XOR شدن آن‌ها به‌صورت شکل ۵ است. O_i و b_i، به‌ترتیب i-امین بایت خروجی AES و i-امین بایت ورودی هستند.



شکل ۴: نحوه ذخیره‌سازی ورودی اول در ۱۶ ثبات



شکل ۵: XOR کردن خروجی AES با ورودی و مرتب‌سازی دوباره

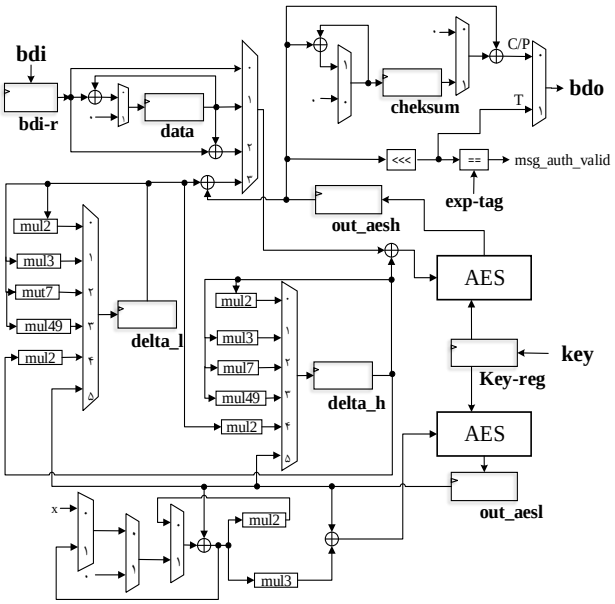
باتوجه به زمان‌بندی ارائه‌شده در طرح، این راه‌کار بدون هیچ هزینه سخت‌افزاری و زمانی انجام می‌شود. هم‌چنین سایر محاسبات موردنیاز بر روی خروجی‌های AES مانند XOR با ورودی، متناسب با حالت ترانهاده انجام‌شده و به همان صورت وارد AES مرحله بعد شده‌است که بدون هیچ‌گونه هزینه سخت‌افزاری انجام می‌شود.

۴- معماری پیشنهادی سبک‌وزن .COLM

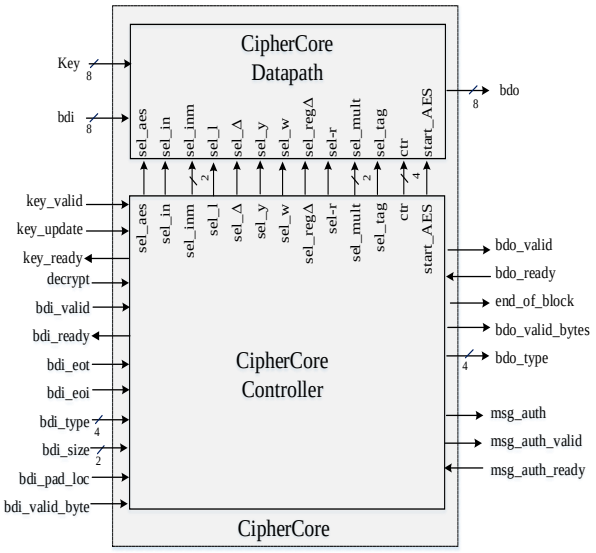
مرور ادبیات موضوع نشان می‌دهد که تاکنون معماری‌های پیاده‌سازی‌شده برای .COLM از نوع سرعت‌بالا بوده است (شکل ۶). معماری ارائه‌شده در شکل ۶ بر اساس کدهای پایه‌ای که طراحان اصلی [۲۶] ارائه کرده‌اند، رسم شده‌است. معماری سرعت‌بالای ارائه‌شده از دو هسته AES، ۸ ثبات ۱۲۸-بیتی و ۱۲ ضرب با ثابت روی میدان $GF(2^{128})$ استفاده کرده و ضرب‌های مرتبه بالاتر را بر اساس دوبرابرکردن ساخته‌است. این نوع معماری برای کاربردهای سبک‌وزن مناسب نیست.

۴-۱- معماری سطح بالای هسته رمز پیشنهادی

معماری سطح بالای هسته رمز و سیگنال‌های داخلی بین بخش مسیر داده و بخش کنترلی هسته رمز برای معماری سخت‌افزاری سبک‌وزن ۸-بیتی پیشنهادی در شکل ۷ نمایش داده‌شده است. ۱۳ سیگنال از بخش کنترلی، پردازش داده در بخش مسیر داده را هدایت می‌کند.



شکل ۶: معماری سرعت‌بالای .COLM



شکل ۷: سیگنال‌های داخلی بین مسیر داده و بخش کنترلی در هسته رمز معماری پیشنهادی COLM

۴-۲- معماری مسیر داده

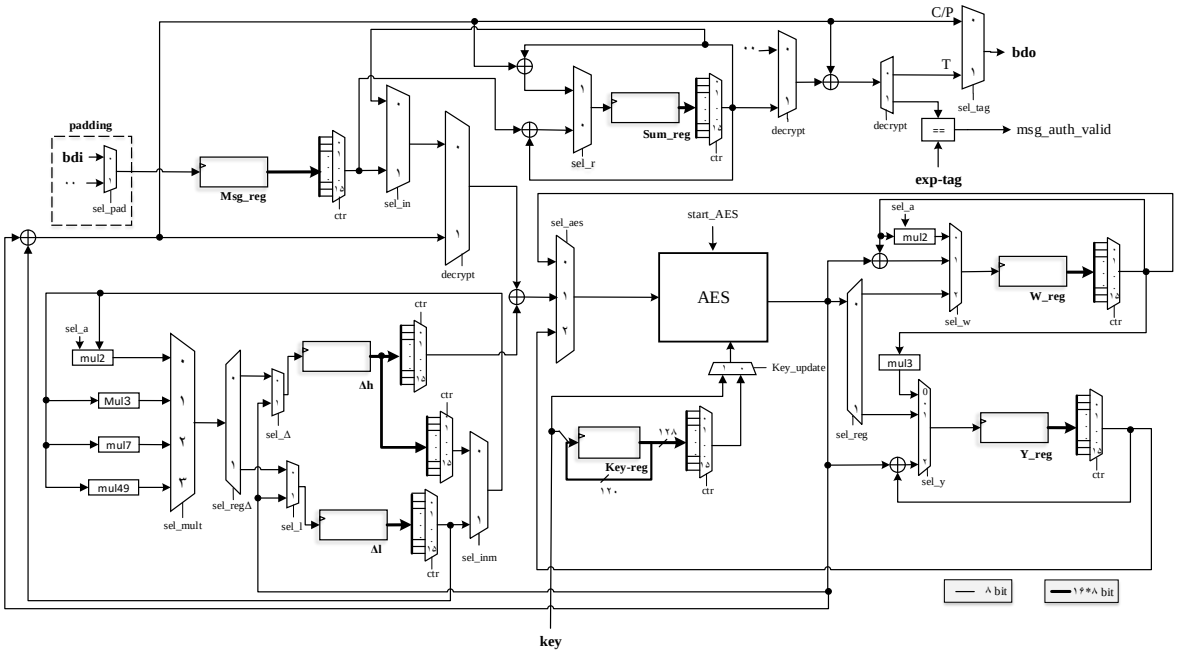
شکل ۸ معماری مسیر داده پیشنهادی .COLM را نشان می‌دهد. این طرح شامل سه مرحله تولید زیرکلید (L)، تولید مقدار اولیه (IV)، تولید متن رمزشده برچسب‌دار است. این معماری به دو بخش بالایی و پایینی تقسیم شده‌است. در بخش بالایی طرح مقادیر AD و متن اصلی/رمزی توسط AES پردازش شده و در بخش پایینی متن رمزشده برچسب‌دار توسط AES تولید می‌گردد. در معماری پیشنهادی برای پیاده‌سازی سبک‌وزن مدار از یک AES برای هر دو بخش استفاده شده‌است و به‌صورت یک‌درمیان بخش بالایی و پایینی اجرا می‌شود. مراحل رمزنگاری در معماری پیشنهادی به‌شرح زیر است:

تولید زیرکلید L و پوشانه‌ها: در اولین گام، کلید فراخوانی‌شده، دوباره مرتب‌شده و در Key_reg ذخیره می‌شود. سپس یک AES با ورودی صفر (0^{128}) جهت تولید زیرکلید L اجرا می‌شود. خروجی AES دوباره مرتب‌شده و با یک مالتی‌پلکسر با خط انتخاب *sel_l* در ثبات‌های ۱۲۸-بیتی Δ_L و Δ_H ذخیره می‌شود. ثبات Δ_H و Δ_L به‌ترتیب برای ذخیره‌سازی نتایج پوشانه‌ها در بخش بالایی و نتایج پوشانه‌ها در بخش پایینی است. باتوجه به این‌که تولید پوشانه یک عملیات بیتی است، ثبات Δ_L و Δ_H به‌صورت ثبات‌های ۱۲۸-بیتی تعریف شده‌است.

برای تولید پوشانه‌های طرح، نیاز به پیاده‌سازی ضرب در ثابت‌های ۲، ۳، ۷ و ۴۹ روی میدان $GF(2^{128})$ است. اگرچه عملیات ضرب روی میدان نیاز به کاهش در پیمانه $GF(2^{128})$ دارد و دارای پیچیدگی مداری بالایی است اما عملیات ضرب در ثابت ۲ (دوبرابرکردن) روی میدان می‌تواند با یک جابجایی یک-بیتی و سه XOR انجام شود. و به‌طور مشابه بقیه ضرب‌ها از دوبرابرکردن ساخته می‌شود. معماری ۸-بیت پیشنهادی برای دوبرابرکردن یک عضو میدان $GF(2^{128})$ که از نوع ساده‌شده ضرب سریال‌رقمی (MSD^{12}) [۲۷] در شکل ۹ نمایش داده‌شده است.

معماری پیشنهادی دوبرابرکردن به‌دلیل نامنظم بودن مقدار ورودی و خروجی ضرب متفاوت با معماری دوبرابرکردن معمولی روی میدان است. این معماری دارای دو قسمت اصلی ۱) جابجایی سیمی برای عملیات شیفت، ۲) جابه‌جایی سیمی برای حفظ داده به‌دلیل ذخیره‌شدن خروجی ضرب در همان ثبات ورودی است. محاسبه آخرین ۸ بیت دوبرابرکردن نیاز به دانستن مقدار باارزش‌ترین بیت ورودی (a_{127}) بوده که به سیگنال کنترلی وارد می‌شود.

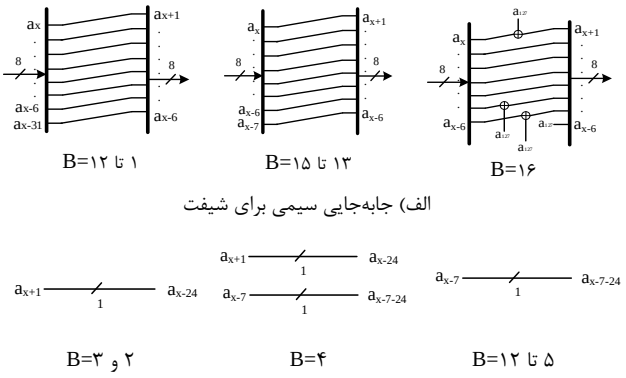
ساختن سایر ضرب‌ها بر اساس دوبرابرکردن $(mul2(x))$ با روابط $mul3(x) = mul2(x) \oplus x$، $mul7(x) = mul3(mul2(x)) \oplus x$ و $mul49(x) = mul7(mul7(x))$ محاسبه شده‌است. زیرا ساختن مستقیم ضرب نسبت به ساختار تکراری به هزینه بیشتری دارد. حداکثر هزینه سخت‌افزاری ازنظر ناحیه مصرفی برای ضرب در ثابت ۴۹ و برابر ۶۰ XOR یک-بیتی در معماری ۸-بیتی و ۷۸۰ XOR یک-بیتی در معماری ۱۲۸-بیتی است. هم‌چنین حداکثر هزینه ازنظر مدت‌زمان اجرا در معماری ۸-بیتی، ۶۴ کلاک و در معماری ۱۲۸-بیتی، ۴ کلاک است. زمان‌بندی معماری پیشنهادی به‌نحوی است که امکان اجرای هم‌زمان ضرب در ثابت‌ها با اجرای AES مرحله قبلی وجود دارد و طول زمان اجرای یک AES ۲۴۶ کلاک است. بنابراین زمان کافی برای محاسبه ضرب با ثابت برای معماری ۸-بیت پیشنهادی وجود دارد و می‌توان از این معماری به‌دلیل مصرف ناحیه کم‌تر نسبت به معماری ۱۲۸-بیت استفاده کرد.



شکل ۸: معماری پیشنهادی ۸ بیت COLM.

پردازش تک‌شمار (N): هم‌زمان با ذخیره‌شدن L در مرحله تولید زیرکلید، مقدار N||param در ثبات Msg_reg ذخیره می‌شود. یک کلاک بعد از اجرای $\Delta_H \times 3$، AES با مقدار ورودی Msg_reg $\oplus \Delta_H$ اجرا می‌شود. خروجی AES با مالتی‌پلکسر با خط انتخاب *sel_w* در W-reg ذخیره می‌شود.

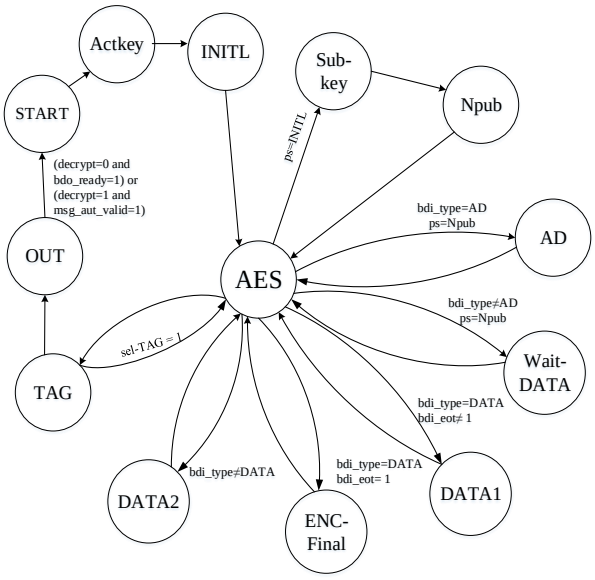
پردازش داده همراه (AD): در هر مرحله از پردازش بلوک AD، مقدار $(2 \times \Delta_H) \oplus bdi$، به‌عنوان ورودی AES پردازش می‌شود و به‌صورت هم‌زمان $\Delta_H \times 2$ در Δ_H ذخیره می‌شود. خروجی ۸-بیتی AES با مقدار W-reg، XOR و مجدد به W-reg منتقل می‌شود. پس از پردازش آخرین بلوک AD مقدار ذخیره‌شده در W-reg مقدار IV است. هم‌زمان با پرشدن W-reg مقدار Δ_L در Δ_H ذخیره می‌گردد.



الف) جابه‌جایی سیمی برای شیفت

ب) جابه‌جایی سیمی برای حفظ داده

شکل ۹: دوبرابرکردن در $GF(2^{128})$، B=شماره بایت، B=۱ باارزش‌ترین بایت ورودی، x=اندیس تعداد بیت و x=۱۳۴ - ۸*B



شکل ۱۰: ماشین حالت بخش کنترلی طرح (ps= حالت قبل)

پردازش متن اصلی-بخش بالایی: برای پردازش هر بلوک متن اصلی، ورودی AES برابر $\text{Msg\_reg} \oplus (2 \times \Delta_H)$ است. به‌صورت هم‌زمان مقدار $2 \times \Delta_H$ در Δ_H ذخیره می‌شود. جمع مقادیر متن اصلی در ثبات Sum_reg ذخیره می‌شود. هم‌زمان با محاسبه AES، $\text{W\_reg} \times 2$ و $\text{W\_reg} \times 3$ محاسبه و با استفاده از دو مالتی‌پلکسر با خط‌های انتخاب *sel_w* و *sel_y* و به‌ترتیب در ثبات‌های W-reg و Y-reg ذخیره می‌شود. خروجی AES هر مرحله با W-reg و Y-reg، XOR و مجدد در آن‌ها ذخیره می‌شود.

تولید متن رمزشده-بخش پایینی: Y-reg ورودی AES بخش پایینی است. هم‌زمان با شروع دریافت ورودی توسط AES، محاسبه مقدار $2 \times 9 \times \Delta_L$ شروع و در ثبات Δ_L ذخیره می‌شود. خروجی AES این مرحله با مقدار Δ_L، XOR شده و بلوک متن رمزشده را می‌سازد. در این معماری تا زمان پردازش آخرین بلوک متن اصلی، AES در بخش بالایی پایینی به‌صورت یک‌درمیان اجرا می‌شود.

تولید برچسب: تولید برچسب نیز دارای دو بخش بالایی و پایینی است. پس از پردازش آخرین بلوک متن اصلی، مقدار ذخیره‌شده در Msg_reg با $2 \times \Delta_H$ یا $7 \times 2 \times \Delta_H$، XOR شده و ورودی AES در بخش بالایی است. خروجی AES با $\text{W\_reg} \times 3$، XOR شده و به‌عنوان ورودی بخش پایینی در W-reg ذخیره می‌شود. برچسب برابر XOR خروجی AES در بخش پایینی با $2 \times \Delta_L$ است.

رمزگشایی و وارسی: این مرحله شبیه رمزنگاری به‌جز این‌که ثبات Sum_reg برای تولید متن اصلی استفاده شده‌است. این مرحله با سیگنال *decrypt* فعال می‌شود.

۴-۳- بخش کنترلی

بخش‌کنترلی با استفاده از یک ماشین حالت ۱۳ حالتی پیاده‌سازی شده‌است (شکل ۱۰). در حالت START مقادیر اولیه مشخص می‌شود. در حالت Actkey کلید بارگذاری‌شده و در حالت INITL ورودی اولین AES برای ساخت پوشانه L_1 آماده می‌شود. بعد از اجرای حالت AES باتوجه به اینکه شرط ورود به کدام حالت محقق شده باشد یکی از حالت‌های Sub_key، AD، Wait-DATA، DATA1، DATA2، ENC_Final، TAG اجرا می‌شود. شرط ورود به این حالت‌ها به موارد مختلفی مانند نوع ورودی در حال پردازش (bdi_type)، شماره بیت در حال پردازش (bdi_eot)، حالت قبل از AES (ps) وابسته است. بعد از اجرای هر یک از این ۷ حالت به‌جز حالت Sub_key دوباره حالت AES اجراشده و درنهایت پس از اجرای حالت OUT و برقراری شروط خاتمه، ماشین حالت به حالت START بازمی‌گردد.

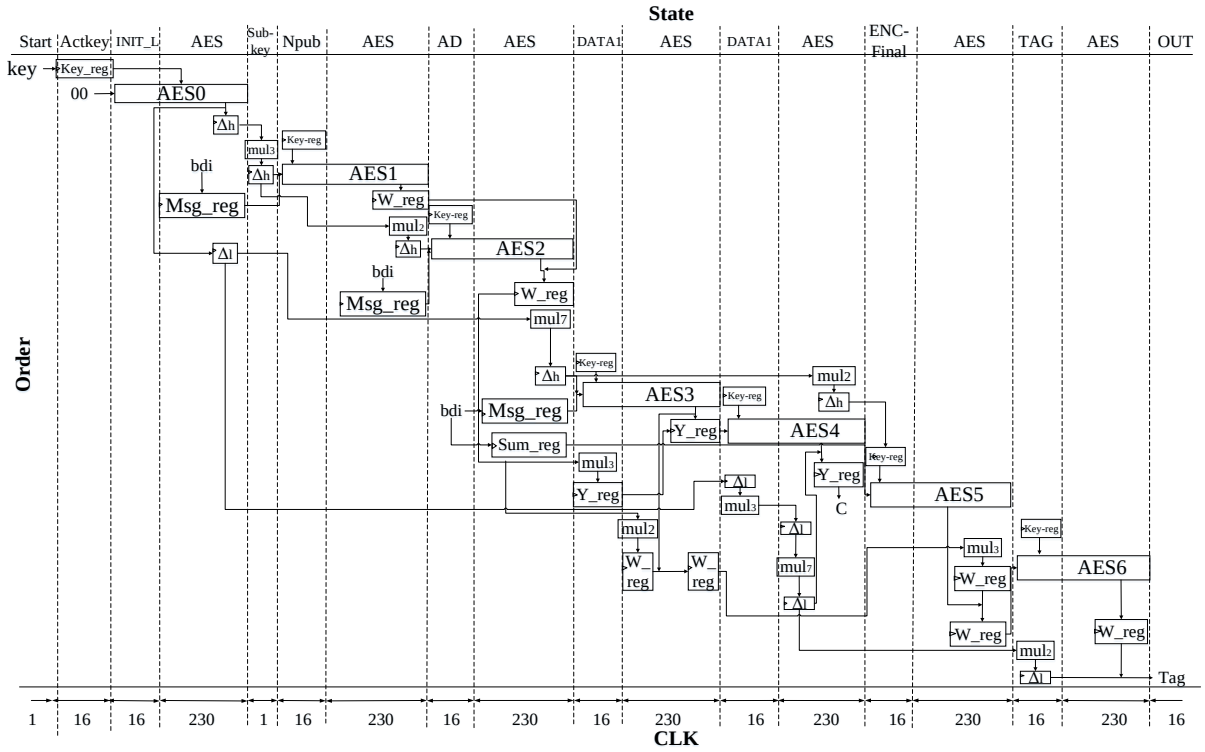
۴-۴- نمودار زمان‌بندی

نمودار زمان‌بندی در مقابل FSM برای پیام با یک بلوک AD و یک بلوک متن اصلی در شکل ۱۱ نشان داده‌شده است. این پیام به ۷ بار اجرای AES نیاز دارد. زمان‌بندی به‌نحوی است که بین اجرای هر AES به‌جز در حالت Sub-key هیچ فاصله‌ای وجود ندارد. هر عملیات Atomic-AES ۲۴۶ کلاک [۸] طول می‌کشد که ۱۶ کلاک اول برای خواندن ورودی و ۱۶ کلاک آخر برای ذخیره‌سازی خروجی است. در کل ۱۷۵۶ کلاک برای پردازش این پیام لازم است.

در حالت کلی تعداد کلاک موردنیاز برای اجرای طرح .COLM با N_{clk} نمایش داده شده و برابر:

$$N_{clk} = \left(2 + N_{|AD|} + 2 \times N_{|M/C|} + 2\right) \times 246 + 16 + 16 \quad (۴)$$

به‌طوری‌که $N_{|AD|}$ و $N_{|M/C|}$ برابر تعداد بلوک AD و متن اصلی/رمز شده‌است. اولین مقدار ثابت ۲ مربوط به اجرای AES برای محاسبه پوشانه L و پردازش تک‌شمار است و مقدار ثابت ۲ بعدی برای محاسبه برچسب است. اعداد ثابت ۱۶ نیز برابر تعدادی کلاکی است که برای بارگذاری کلید و ارسال برچسب به پس‌پردازنده مصرف می‌شود.



شکل ۱۱ : برنامه زمان‌بندی .COLM برای پیام با یک بلوک AD و یک بلوک متن اصلی/رمزشده

۵- نتایج پیاده‌سازی و مقایسه با کارهای قبلی

برای ارائه نتایج، هر معماری در سطح RTL با زبان VHDL کدنویسی و سپس برروی دو بستر FPGA و ASIC سنتز و پیاده‌سازی شده‌است.

۵-۱- نتایج پیاده‌سازی بر روی FPGA

سنتز و پیاده‌سازی برروی FPGA-های Xilinx Virtex-6 مدل xc6vlx760t و Xilinx Virtex-7 مدل xc7vx485t انجام‌شده است که بدین منظور از دو ابزار Xilinx ISE v14.7 و Xilinx Vivado2017.2 استفاده شده‌است. هم‌چنین از ابزار Minerva [۲۸] به‌منظور بهینه‌سازی سخت‌افزاری خودکار استفاده شده‌است. این ابزار با کمک نرم‌افزار Vivado حدود ۲۵ الگوریتم بهینه‌سازی سخت‌افزاری را روی طرح اجرا می‌کند. این ابزار دارای دو مد بهینه‌سازی گذردهی و بهینه‌سازی نسبت گذردهی به ناحیه است. در مد اولی فرکانس کاری بیشینه شده و در مد دوم نسبت فرکانس به LUT بیشینه‌سازی می‌گردد که در اینجا مد دوم انتخاب شده‌است.

صحت عملکرد مدار با کمک ابزار Mentor Graphics ModelSim نسخه 10.1.c بررسی شده‌است. این ابزار یکی از قوی‌ترین نرم‌افزارهای طراحی و شبیه‌سازی برنامه‌های VHDL است. طراحان به‌وسیله این نرم‌افزار می‌توانند برنامه‌های خود را قبل از تست سخت‌افزاری، شبیه‌سازی کرده و از عملکرد برنامه خود اطمینان حاصل نمایند.

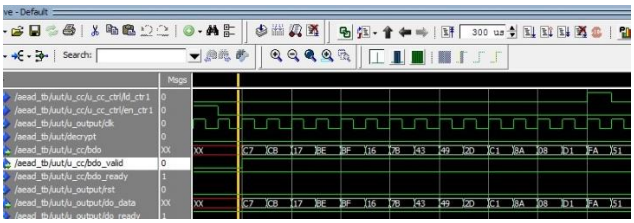
شبیه‌سازی طرح سبک‌وزن پیشنهادی بر اساس کلیه بردارهای تست که در [۲۶] ارائه شده‌است ، توسط این نرم‌افزار انجام‌شده است. شکل ۱۲ خروجی این ابزار را برای مقادیر ورودی (جدول ۴) نشان می‌دهد.

در شکل ۱۲ مقدار متن رمزشده با bdo، برچسب با dout و اعتبار هرکدام با متغیر با پسوند valid نشان داده‌شده است. مقدار E0 برای متغیر dout بیانگر اتمام رمزنگاری/رمزگشایی است.

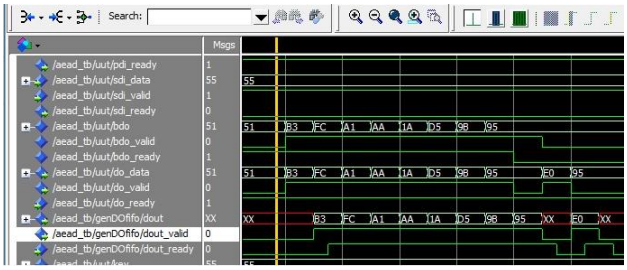
در جدول ۵ و شکل ۱۳ نتایج پیاده‌سازی سخت‌افزاری معماری سبک‌وزن پیشنهادی با [۱۰] و [۱۲] مقایسه شده‌است. معماری ۸-بیت پیشنهادی نسبت به معماری پایه سرعت‌بالای ۱۲۸-بیتی [۱۰]، ۶۲٪ کاهش مصرف ناحیه داشته است. ازآنجاکه برای طرح‌های سرعت‌بالا مصرف ناحیه API v۱ و API v۲ تفاوت زیادی ندارند، نتایج کارهای دیگران که با API v۱ گزارش‌شده برای مقایسه با این کار استفاده شده‌است. ناحیه مصرفی در FPGA-ها برحسب LUT و Slice گزارش شده‌است و از دیگر منابع موجود در FPGA-ها مانند رم بلوکی[۱۳] استفاده‌نشده است.

جدول ۴: یک نمونه از بردارهای تست COLM

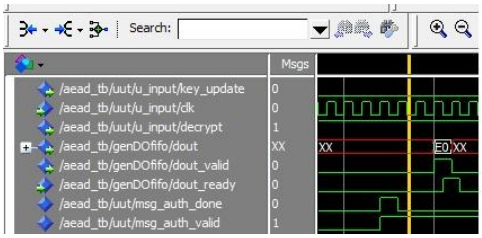
عنوان داده	مقدار
متن اصلی	FF000102030405060708090A0B0C0D0E
کلید	55565758595A5B5C5D5E5F6061626364
داده همراه	A0A1A2A3A4A5A6A7A8A9AAABACADAEAF
تک‌شمار	000000C0B0B1B2B3B4B5B6B7B8B9BABB
متن رمزشده	C7CB17BEBF167B43492DC18A08D1FA51
برچسب	B3FCA1AA1AD59B9



الف) حالت رمزنگاری- تولید متن رمزشده



ب) حالت رمزنگاری-تولید برچسب

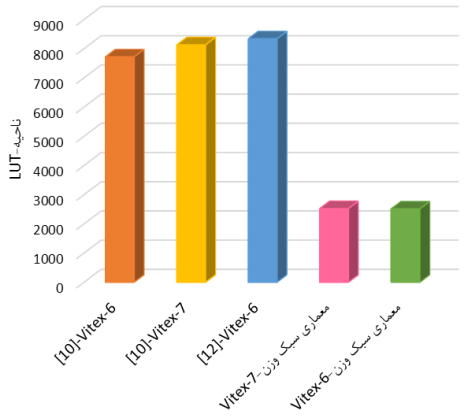


ج) حالت رمزگشایی

شکل ۱۲: نتایج شبیه‌سازی برای طرح پیشنهادی

جدول ۵: مقایسه نتایج پیاده‌سازی COLM با دیگران در FPGA

مرجع	نوع FPGA	معماری	ناحیه مصرفی (LUTs)	ناحیه مصرفی (Slices)	حداکثر فرکانس (MHz)	گذردهی (Mbps)
[۱۰]	Vitex-6	۱۲۸بیت	۷۷۱۸	۲۰۶۰	۲۴۱/۸	۳۰۹۵
	Vitex-7		۸۱۳۱	۲۱۶۶	۲۶۳	۳۰۶۰
[۱۲]	Vitex-6	خط-لوله (۱۲۸ بیت)	۸۳۳۷	۳۹۶۲	۱۸۰	۲۰۹۳
معماری سبک‌وزن	Vitex-7	۸ بیت	۲۵۱۱	۸۱۶	۱۴۲/۸	۳۷/۱۶
	Vitex-6		۲۵۲۱	۱۱۱۳	۱۴۹/۶	۳۸/۹۴



شکل ۱۳: مقایسه نتایج پیاده‌سازی COLM با دیگران در FPGA

طرح پیشنهادی به‌گونه‌ای است که هیچ کلاک اضافه‌ای بین اجرای AES-ها وجود ندارد بنابراین گذردهی کلی طرح به گذردهی AES و فرکانس کاری مدار وابسته است. هم‌چنین به دلیل اینکه برای پردازش متن اصلی/رمزشده به اجرای دو AES نیاز است و در طرح پیشنهادی سبک‌وزن یک AES پیاده‌سازی شده‌است گذردهی مدار از گذردهی AES انتخابی کمتر شده‌است. طرح‌های سبک‌وزن و سرعت‌بالا ازلحاظ معیار گذردهی با یکدیگر قابل‌مقایسه نیستند و معیار اصلی در این پیاده‌سازی‌های سبک‌وزن مصرف کم منابع است [۸]. بنابراین علی‌رغم کاهش گذردهی، به‌دلیل کاهش شدید ناحیه مصرفی، طرح پیشنهادی دارای عملکرد قابل‌قبولی است.

۵-۲- نتایج پیاده‌سازی بر روی ASIC

برای پیاده‌سازی ASIC از ابزار Synopsys Design Compiler نسخه C-2009.06-SP5 و کتاب‌خانه سلول استاندارد open-cell NANGATE ۴۵ nm [۲۹] استفاده شده‌است و صحت عملکرد مدار با کمک ابزار Mentor Graphics ModelSim بررسی شده‌است. هم‌چنین در سنتز محدودیت کاهش ناحیه اعمال‌شده است و دوره کلاک آن‌قدر تغییر داده‌شده تا مقدار اسلک[۱۴] صفر و بیش‌ترین فرکانس محاسبه گردد.

به‌منظور بهینه‌سازی بیش‌تر بر روی بستر AISC در معماری پیشنهادی دو تغییر نسبت به FPGA وجود دارد. نخست S-box پیاده‌سازی شده AES از نوع منطقی است و دوم این‌که برای پیاده‌سازی مالتی‌پلکسر ۱۶:۱ از مالتی‌پلکسرهای اولیه ۲:۱ موجود در کتاب‌خانه استفاده‌شده که باعث کاهش ناحیه GE ۰/۵ به‌ازای هر بیت شده‌است. همان‌طور که در بخش ۳-۲ بیان‌شده برای پیاده‌سازی AES از کدهای [۸] استفاده شده‌است. به این منظور این کدها با کمک کتاب‌خانه NANGATE ۴۵ nm سنتز شده و نتیجه در جدول ۶ آورده شده‌است.

نتایج جدول ۶ نشان می‌دهد که نتیجه سنتز متفاوت از نتایج گزارش‌شده در [۸] است. کتاب‌خانه STM ۹۰ nm استفاده‌شده در [۸] دارای ویژگی‌های برتری نسبت به کتاب‌خانه NANGATE ۴۵ nm استفاده‌شده در این کار است. برای مثال کتاب‌خانه STM ۹۰ nm دارای سلول‌های فلیپ-فلاپ چندبیتی[۱۵] برای پیاده‌سازی ثبات‌ها است که عمده حجم AES را اشغال می‌کنند.

جدول ۶: مقایسه نتایج پیاده‌سازی AES با کتاب‌خانه‌های مختلف

مرجع	عملیات	کتاب‌خانه	ناحیه مصرفی (KGE)	گذردهی (Mbps)	
				رمزنگاری	رمزگشایی
[۸]	رمزنگاری/ رمزگشایی	STM ۹۰nm	۲/۰۶	۸۸/۴	۶۶/۷
سنتز مجدد [۸]	رمزنگاری/ رمزگشایی	NANGATE ۴۵ nm	۳/۰۹۱	۲۳۱/۲	۱۷۴/۵

این ویژگی منجر به کاهش ناحیه مصرفی از GE ۶/۶ به GE ۴/۵ به‌ازای هر بیت فلیپ-فلاپ معمولی می‌شود. هم‌چنین اختلاف GE ۲ به‌ازای هر بیت برای اسکن فلیپ-فلاپ برای دو کتاب‌خانه وجود دارد. نتایج سنتز

دوباره کدهای [۸] منجر به افزایش ۵۰ درصدی ناحیه مصرفی شده‌است. بااین‌وجود، به علت افزایش فرکانس کاری، نرخ گذردهی حدود ۲/۶ برابر افزایش داشته است.

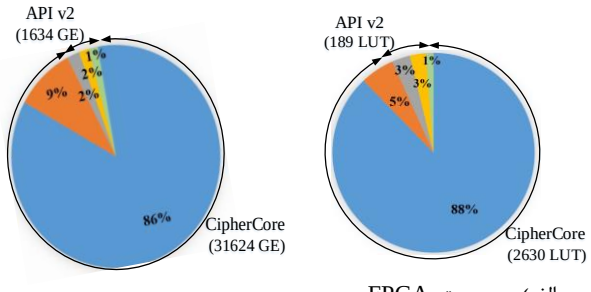
تیم GMU [۲۶] تاکنون نتایجی از پیاده‌سازی طرح COLM را بر روی ASIC گزارش نکرده است. در [۱۳] کدهای ارائه‌شده در [۲۶] را با کتاب‌خانه TSMC ۶۵nm سنتز و نتایج را گزارش کرده است. برای مقایسه عادلانه بین نتایج معماری سبک‌وزن پیشنهادی و کدهای ارائه‌شده در [۲۶] هر دو معماری با کتاب‌خانه یکسان ۴۵nm NANGATE، سنتز شده‌است. مقایسه این نتایج در جدول ۷ نشان می‌دهد که معماری پیشنهادی ۸-بیتی نسبت به معماری ۱۲۸-بیتی پایه دارای ۷۴٪ کاهش مصرف ناحیه بوده است. بنابراین اگرچه معماری سبک‌وزن نسبت به سایرین دارای گذردهی کمتری است اما ناحیه مصرفی آن به‌شدت کاهش‌یافته است. هم‌چنین توان مصرفی نیز کاهش داشته است.

جدول ۷: مقایسه نتایج پیاده‌سازی COLM با دیگران در ASIC

مرجع	کتاب‌خانه	ناحیه مصرفی (KGE)	حداکثر فرکانس (MHz)	گذردهی (Mbps)	توان (mW)
[۱۳]	TSMC ۶۵nm	۱۲۳	۵۰۵	۵۸۸۰	-
سنتز [۲۶]	NANGATE ۴۵nm	۱۰۴	۳۷۰	۴۷۴۰	۱۴/۴
معماری سبک‌وزن	NANGATE ۴۵nm	**۲۷**	۳۸۴/۶۲	۱۰۰/۰۶	۶/۱۲

۵-۳- تأثیر API بر ناحیه مصرفی

بررسی نتایج ازنظر ناحیه مصرفی نشان می‌دهد که تغییرات اشاره‌شده در بخش ۳-۱-۱ جهت بهبود API، ناحیه مصرفی را در FPGA و ASIC به‌ترتیب به میزان ۸٪ و ۶٪ کاهش داده است. تحلیل مصرف منابع سخت‌افزاری به تفکیک اجزاء API v۲ و هسته رمز COLM در نشان می‌دهد که افزونگی API در طرح COLM بر روی بستر FPGA و ASIC به‌ترتیب برابر ۷٪ و ۵٪ است (شکل ۱۴).



الف) روی بستر FPGA ب) روی بستر ASIC

شکل ۱۴: ناحیه مصرفی به تفکیک واحدها

۶- نتیجه‌گیری و کارهای آینده

در این تحقیق برای اولین بار برای طرح COLM معماری سبک ۸-بیتی و سازگار با API v۲ ارائه شده‌است. این معماری دارای ۷ ثبات ۱۲۸-بیتی برای ذخیره‌سازی کلید، متن اصلی، خروجی AES و پوشانه‌ها است. رمزنگاری و رمزگشایی در یک هسته طراحی و الگوریتم AES به‌عنوان رمز قالبی پایه تنها یک‌بار پیاده‌سازی شده‌است. کلیه عملیات طرح متناسب با ورودی و خروجی غیرمعمول (ترانهاده) AES طراحی شده‌است. برای پیاده‌سازی دوبرابرکردن روی میدان $GF(2^{128})$ به‌دلیل غیرمعمول بودن داده ورودی، معماری جدیدی ارائه شده‌است و سایر ضرب‌ها بر اساس آن پیاده‌سازی شده‌است. زمان‌بندی طرح به‌نحوی است که فاصله بین اجرای دو AES حداکثر یک کلاک است.

پیاده‌سازی این معماری‌ها بر روی دو بستر ASIC و FPGA انجام‌شده است. تکنیک‌های متفاوتی برای بهینه‌سازی در هر بستر استفاده شده‌است. پیاده‌سازی S-box به‌صورت ROM در FPGA، ۹٪ کاهش ناحیه مصرفی نسبت به منطقی و پیاده‌سازی S-box به‌صورت منطقی در ASIC ، ۱۱٪ کاهش ناحیه مصرفی را نسبت به ROM دارد. هم‌چنین در ASIC برای پیاده‌سازی مالتی‌پلکسر ۱۶:۱، استفاده از مالتی‌پلکسرهای اولیه ۲:۱ موجود در کتاب‌خانه، باعث ۰/۵ GE کاهش ناحیه به ازای هر بیت شده‌است. مقایسه نتایج این کار با پیاده‌سازی‌های ۱۲۸-بیتی نشان می‌دهد که ناحیه مصرفی روی بستر FPGA و ASIC به‌ترتیب برابر ۶۲٪ و ۷۴٪ کاهش داشته است.

باتوجه به این‌که طرح COLM به‌صورت ذاتی در برابر حملات کانال جانبی آسیب‌پذیر است و طراحان الگوریتم تاکنون برای امن‌سازی در برابر این حملات هیچ بهبودی در این طرح ارائه نداده‌اند، مقاوم‌سازی این طرح در برابر حملات کانال جانبی به‌عنوان کارهای آینده پیشنهاد می‌گردد.

مراجع

[1] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*, NIST Publications, SP.800-38C, 2004, https://csrc.nist.gov/publications/detail/sp/800-38c/final.

[2] M. Dworkin, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, NIST Special Publications, NIST.SP.800-38D, 2007, https://csrc.nist.gov/publications/detail/sp/800-38d/final.

[3] https://competitions.cr.yp.to/2014-03-15.

[۴] ح. پروین، م. محمدپور، ر. ا. امیدوار، «ارائه روشی مبتنی‌بر پوشش سراسری و تخمین اتفاق آرا برای بهبود کارایی در شبکه حسگر بی‌سیم»، مجله مهندسی برق دانشگاه تبریز، جلد ۴۷، شماره ۳، صفحه ۸۹۱-۸۷۷، ۱۳۹۶.

[5] E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, P. Yalla, J. P. Kaps and K. Gaj, "CAESAR Hardware API," Cryptology ePrint Archive, Report 2016/626. 2016 [Online]. Available: https://eprint.iacr.org/2016/626.

[6] Cryptographic Engineering Research Group (CERG) at GMU. *Development Package for Hardware Implementations Compliant with the CAESAR Hardware API v1*, [Online]. https://cryptography.gmu.edu/athena /index.php?id=CAESAR/2016.

[7] Cryptographic Engineering Research Group (CERG) at GMU, *Development Package for Hardware Implementations Compliant with the CAESAR Hardware API v2*, [Online]. https://cryptography.gmu.edu/athena /index.php?id=CAESAR/2017.

[8] S. Banik, A. Bogdanov and F. Regazzoni, "Atomic-AES v2.0," Cryptology ePrint Archive, Report 2016/1005. 2016 [Online]. Available: http://eprint.iacr.org/2016/1005.

[9] *Database of FPGA Results for Authenticated Ciphers*, George Mason University, U.S.A. Available at https:// cryptography.gmu.edu/athenadb/fpga_auth_cipher/table/ 2018-03-12.

[10] *Authenticated Encryption FPGA Ranking*, George Mason University, U.S.A. Available at https://cryptography. gmu.edu/athenadb/fpga_auth_cipher/rankings_view/ 2018-03-12.

[11] S. Banik and A. Bogdanov, "Low-area hardware implementations of CLOC, SILC and AES-OTR," IEEE International Symposium on Hardware Oriented Security and Trust (HOST), McLean, VA, pp. 71-74, 2016.

[12] D. Sanjay and K. Gaj, "Analysis and Inner-Round Pipelined Implementation of Selected Parallelizable CAESAR Competition Candidates," Euromicro Conference on Digital System Design (DSD), Vienna, Austria, IEEE, pp. 274-282, 2017.

[13] S. Kumar, J. Haj-Yahya, M. Khairallah, and A. Chattopadhyay, "A Comprehensive Performance Analysis of Hardware Implementations of CAESAR Candidates," Cryptology ePrint Archive, Report 2017/1261. 2017 [Online]. Available: http://eprint.iacr.org/2017/1261.

[14] W. u. Hongjun and B. Preneel, *AEGIS: A fast authenticated encryption algorithm*, CAESAR competition proposal, https://competitions.cr.yp.to/round3 /aegisv11/2016.

[15] T. Krovetz and P. Rogaway, *OCB v1.1*, CAESAR competition proposal, Available at: https://competitions. cr.yp.to/round3/ ocbv11.pdf/2016.

[16] A. Elena, A. Bogdanov, N. Datta, A. Luykx, B. Mennink, M. Nandi, E. Tischhauser and K. Yasuda, *COLM v1*, CAESAR competition proposal, Available at https://competitions.cr.yp.to/round3/colmv1.pdf/2016.

[۱۷] م. جهانبانی، ن. باقری، ز.ا. نوروزی، «مروری بر پیاده‌سازی‌های سخت‌افزاری سبک‌وزن رمز AES»، دو فصلنامه علمی و ترویجی منادی، دوره ۶، شماره ۲، صفحه ۳-۲۰، ۱۳۹۶.

[18] M. Feldhofer and J. Wolkerstorfer "AES Implementation on a Grain of Sand," In IEEE Proceedings of Information Security, vol. 152, no.1, pp. 13-20, 2005.

[19] S. Mathew, S. Satpathy, V. Suresh, M. Anders, H. Kaul, A. Agarwal, S. Hsu, and R. K. Krishnamurthy, "340 mV-1.1V, 289 Gbps/W, 2090-gate nanoAES hardware accelerator with area-optimized encrypt/decrypt GF(2^4)^2 polynomials in 22 nm tri-gate CMOS," In IEEE Journal of Solid-State Circuits, vol. 50, no. 1, pp. 1048-1058, 2015.

[20] A. Moradi, A. Poschmann, S. Ling, C. Paar, H. Wang, "Pushing the Limits: A Very Compact and a Threshold Implementation of AES," In Eurocrypt, LNCS, Springer, vol. 6632, pp. 69-88, 2011.

[21] S. Banik, A. Bogdanov and F. Regazzoni, "Atomic-AES: A compact implementation of the AES Encryption/Decryption core," Cryptology ePrint Archive, Report 2016/927. 2016 [Online]. Available: http://eprint.iacr.org/2016/927.

[22] J. Chu and M. Benaissa, "Low area memory-free FPGA implementation of the AES algorithm," In: 2012 22nd international conference on, field programmable logic and applications (FPL), IEEE, pp. 623–626, 2012.

[23] P. Sasdrich and T. Güneysu, Pushing the limits: ultra-lightweight AES on reconfigurable hardware. In: Workshop on trustworthy manufacturing and utilization of secure devices. TRUDEVICE, 2015.

[۲۴] پ. دری، ع. قیاسیان، ح. سعیدی، «طراحی و پیاده‌سازی رمزنگار AES در بستر FPGA برای خطوط پرسرعت»، مجله مهندسی برق دانشگاه تبریز، دوره ۴۶، شماره ۱، صفحه ۱۶۷-۱۵۳، ۱۳۹۵.

[25] M. Khairallah and A. Chattopadhyay, "Looting the LUTs: FPGA Optimization of AES and AES-like Ciphers for Authenticated Encryption," In International Conference in Cryptology in India, Springer, Cham, pp. 282-301, 2017.

[26] *GMU Implementations of Authenticated Ciphers*, George Mason University, U.S.A, available https:// cryptography. gmu.edu/athena/index.php?id=download/2018-03-12.

[۲۷] م. جهانبانی، ن. باقری، ز. ا. نوروزی، «پیاده‌سازی سخت‌افزاری سیستم های رمزنگاری براساس زوج سازی تیت با استفاده از FPGA روی F_2^{283}»، مجله علوم و فناوری‌های پدافند نوین، دوره ۷، شماره ۲، صفحه ۹۵-۱۰۶، ۱۳۹۵.

[28] F. Farahmand, A. Ferozpuri, W. Diehl and K. Gaj, https://cryptography.gmu.edu/athena/index.php?id=Minerva/2018-01-25.

[29] *NANGATE: The NanGate 45 nm Open Cell Library*, http://www.nangate.com/2018-02-14.

زیرنویس‌ها

[1] Authenticated Encryption

[2] Nonce

[3] Associated Data

[4] Competition for Authenticated Encryption: Security, Applicability, and Robustness

[5] Cryptographic Engineering Research Group

[6] Padding

[7] Nonce Misuse Resistant

[8] Release Unverified Plaintext

[9] Finite State Machine

[10] First Input-First Output

[11] First-Word-Fall-Through

[12] Most significant Digit-Serial

[13] Block RAM

[14] Slack

[15] Multi-bit-flip-flop cells