Electronical & Cyber Defence
Vol. 13, No. 1, Spring 2025, Serial No. 49

Reducing delay of InvBR-LWE PQC algorithm in limited-resources devices

N. Shiri, H. Abdoli*

*Assistant Professor, Bu-Ali Sina University, Hamedan, Iran

Received:2024 /12/08, Revised: 2025/02/19, Accepted: 2025/03/13, Published: 2025/04/21

DOR: https://dor.isc.ac/dor/20.1001.1.23224347.1404.13.1.10.5

ABSTRACT

With the expansion of IoT devices and the emergence of quantum computers, new security challenges have arisen. One significant concern is the vulnerability of IoT infrastructure to malicious attacks, given its integral role in the Internet and digital ecosystems. Quantum computers possess processing power millions of times greater than that of classical computers, rendering traditional cryptographic algorithms susceptible to decryption. Furthermore, resource constraints in IoT and edge devices exacerbate the difficulty of implementing large and complex cryptographic algorithms. Consequently, there is a pressing need for lightweight cryptographic approaches that offer resistance to both quantum and classical attacks. Given that large-scale quantum computers are anticipated to become available within the next 10–15 years, the NIST has initiated the post-quantum cryptography standardization process to identify new public-key algorithms that can withstand quantum attacks. Among the various quantum-resistant cryptographic schemes, lattice-based cryptography has emerged as a promising, cost-effective, and efficient solution. Specifically, lattice-based schemes derived from LWE problem and BR-LWE model are designed to address the constraints of resource-limited devices. These schemes leverage binary errors to minimize key sizes and reduce hardware requirements while maintaining sufficient security for lightweight applications. However, implementing such algorithms presents challenges, including execution time, latency, and resource demands. In this study, an efficient LFSR-based architecture is proposed to facilitate parallel and efficient polynomial multiplication, which is critical for InvBR-LWE scheme. By decomposing polynomial coefficients and into multiple groups and executing them simultaneously in two parallel circuits, the overall execution time of the algorithm is significantly reduced. Synthesis results on an FPGA chip demonstrate that the proposed scheme achieves lower total latency than existing approaches due to a reduced execution cycle. Overall, the ADP criterion of the proposed method is improved by 35%. These findings indicate that the proposed scheme effectively reduces latency in lightweight cryptographic applications.

Keywords: Post-quantum cryptography, InvBR-LWE, Lattice-Based Algorithms, hardware implementation

.

* Corresponding Author Email: abdoli@basu.ac.ir

"پدافند الکترونیکی و سایبری"
سال سیزدهم، شماره ۱،بهار ۱۴۰۴، ص۱۴۳-۱۳۱
شاپا الکترونیکی: ۸۹۷۹-۲۹۸۰ شاپا چاپی: ۴۳۴۷-۲۳۲۲

علمی – پژوهشی

کاهش تأخیر اجرای الگوریتم رمزنگاری پساکوانتوم *InvBR-LWE* در دستگاه‌های با منابع محدود

نادر شیری[۱]، حاتم عبدلی[۲*]

۱- دانشجوی کارشناسی ارشد، ۲- استادیار، دانشگاه بوعلی سینا، همدان، ایران.

(دریافت: ۱۴۰۳/۰۹/۱۲، بازنگری: ۱۴۰۳/۱۲/۰۱، پذیرش: ۱۴۰۳/۱۲/۲۳، انتشار: ۱۴۰۳/۰۲/۰۱)

DOR: https://dor.isc.ac/dor/20.1001.1.23224347.1404.13.1.10.5

* این مقاله یک مقاله با دسترسی آزاد است که تحت شرایط و ضوابط مجوز *Creative Commons Attribution* (*CC BY*) توزیع شده است.

ناشر: دانشگاه جامع امام حسین (ع) © نویسندگان

چکیده

با گسترش دستگاه‌های اینترنت اشیاء و ظهور کامپیوترهای کوانتومی با چالش جدیدی مواجه هستیم که در زیرساخت‌های اینترنت اشیاء به‌عنوان بخشی از اینترنت و دنیای دیجیتال با حملات مخربی روبرو خواهیم بود. قدرت پردازش کامپیوترهای کوانتومی تا میلیون‌ها برابر کامپیوترهای کلاسیک است و درنتیجه الگوریتم‌های رمزنگاری کلاسیک در معرض شکسته شدن قرار می‌گیرند. همچنین محدودیت منابع در دستگاه‌های IoT و لبه، سختی کار را برای پیاده‌سازی الگوریتم‌های بزرگ و پیچیده دوچندان می‌کند؛ بنابراین نیاز به رویکردهای رمزنگاری سبک‌وزن و درعین‌حال مقاوم در برابر حملات کامپیوترهای کوانتومی و طبیعتاً کامپیوترهای کلاسیک داریم. بر این اساس که کامپیوترهای کوانتومی در مقیاس بزرگ در ۱۵-۱۰ سال آینده در دسترس خواهند بود،NIST فرآیند استانداردسازی رمزنگاری پساکوانتومی را به‌منظور یافتن الگوریتم‌های کلید عمومی جدید و مقاوم در برابر رایانه‌های کوانتومی آغاز کرد. در میان انواع مختلف طرح‌های رمزنگاری مقاوم در برابر کوانتوم، رمزنگاری مبتنی بر مشبکه به‌عنوان یک طرح مقرون‌به‌صرفه و کارا، در حال گسترش است. طرح‌های پیشنهادی مبتنی بر مشبکه براساس مسئله LWE و نوع سبک BR-LWE باهدف پوشش دستگاه‌های با منابع محدود، برای کاهش اندازه کلید و دستیابی به مساحت کمتر، خطاهای باینری را به کار می‌گیرند و درعین‌حال امنیت کافی برای برنامه‌های سبک‌وزن را نیز حفظ می‌کنند. پیاده‌سازی این الگوریتم با چالش‌هایی مانند زمان اجرا، تأخیر و منابع موردنیاز روبه‌رو است. در روش پیشنهادی، یک معماری کارآمد مبتنی بر LFSR برای اجرای موازی و مؤثر ضرب چندجمله‌ای و کاربرد آن در طرح InvBR-LWE ارائه شده است. با تجزیه ضرایب چندجمله‌ای A و B به گروه‌های متعدد و اجرای هم‌زمان در دو مدار موازی، زمان اجرای کل الگوریتم کاهش یافته است. نتایج سنتز بر روی تراشه FPGA نشان می‌دهد که طرح پیشنهادی نسبت به کارهای مشابه، به دلیل کاهش سیکل اجرا، تأخیر کل کمتری دارد و به‌طورکلی معیار ADP روش پیشنهادی تا ۳۵٪ کاهش‌یافته است. با توجه به نتایج حاصل‌شده، طرح پیشنهادی می‌تواند باعث کاهش تأخیر در کاربردهای سبک‌وزن شود.

کلید واژه‌ها: رمزنگاری پساکوانتوم، InvBR-LWE، پیاده‌سازی سخت‌افزاری، رمزنگاری مبتنی بر مشبکه.

۱. مقدمه

اینترنت اشیاء *(IoT[۱])* به‌عنوان مجموعه‌ای از چیزها تعریف می‌شود که با تبادل انواع داده‌های ناشی از تعامل حسگرها و اشیاء مختلف ایجاد می‌شود و شامل ساختارهای ناهمگن بزرگ (مدل‌های ارتباطی بین افراد و اشیا) و دستگاه‌های هوشمند (دستگاه‌های تعبیه‌شده‌ای با امکان تبادل داده‌های بلادرنگ از طریق حسگرهای مجازی و فیزیکی) است.

محبوبیت اینترنت اشیاء به‌قدری سریع در حال افزایش است که برخی گزارش‌ها، فعال شدن ۷۵ میلیارد دستگاه اینترنت اشیاء در سال ۲۰۲۵ را دور از دسترس نمی‌دانند. این رشد عظیم مستلزم استانداردسازی پروتکل‌ها و توسعه معماری‌هایی مناسب برای ارائه خدمات به دستگاه‌های اینترنت اشیاء است. به‌طورکلی چالش‌های زیادی برای دستگاه‌های وابسته به باتری، و با قدرت محاسباتی و حافظه محدود وجود دارد [۱].

روش‌های احراز هویت مرسوم برای شبکه‌های اینترنت اشیاء به دلیل ماهیت محدود منابع دستگاه‌های *IoT* به‌طورمعمول قابل‌اجرا نیستند. در اینترنت اشیاء نیاز به رویکردهای احراز هویت سبک‌وزن و نیز حفظ حریم خصوصی اهمیت زیادی دارد [۲].

[1] *Internet of Things*

از طرفی، پیش‌بینی می‌شود با روی کار آمدن و قدرتمند شدن روزافزون کامپیوترهای کوانتومی، حدوداً تا سال ۲۰۳۰ به‌راحتی الگوریتم‌های رمزنگاری کلید عمومی شکسته شوند. برای حل این مشکل باید از الگوریتم‌های سبک‌وزن رمزنگاری پساکوانتوم و امضای دیجیتال[1] مقاوم در مقابل حملات کوانتومی استفاده کرد [۳].

۱-۱. الگوریتم‌های رمزنگاری پساکوانتوم

در این راستا مؤسسه ملی استاندارد و فناوری ایالات‌متحده (*NIST*[2]) فرایند استانداردسازی رمزنگاری پساکوانتومی (*PQC*[3]) را به‌منظور یافتن الگوریتم‌های کلید عمومی جدید و مقاوم در برابر رایانه‌های کوانتومی را در اواخر سال ۲۰۱۶ آغاز کرد [۴].

عملی بودن طرح‌های پساکوانتومی مبتنی بر *Lattice* یا مشبکه (*LBC*[4])، برای امضای دیجیتال و تبادل کلید، بر اساس معیارهایی برای کاربردهای اینترنت اشیاء، قابل‌بررسی است [۵]. در پژوهش‌های این حوزه، دو چالش از اهمیت بیشتری برخوردار است و عمدتاً به‌عنوان معیار ارزیابی و مقایسه مورد استفاده قرار می‌گیرند:

- زمان اجرای الگوریتم: هرچقدر زمان اجرا کمتر باشد، کارایی و عملکرد الگوریتم بهتر است.
- منابع موردنیاز: پیاده‌سازی باید به‌گونه‌ای باشد که مناسب منابع محدود آن دستگاه باشد.

با توجه به این چالش‌ها، در اکثر پژوهش‌های مرتبط، تأکید بر کمتر شدن شاخص *ADP*[5] است که حاصل‌ضرب تعداد اسلایس رجیسترها در تأخیر است (کاهش توأمان مساحت و تأخیر) و از رابطه *(1)* به دست می‌آید:

$$ADP = \#ALMs^{6} * delay \qquad (1)$$

برای بهینه‌سازی پیاده‌سازی سخت‌افزاری الگوریتم‌های رمزنگاری پساکوانتوم دو رویکرد زیر مورد استفاده قرار می‌گیرد:

- بهبود کارایی بخش‌های پیچیده الگوریتم مانند بهینه‌سازی بخش ضرب چندجمله‌ای با استفاده از روش‌های حساب کامپیوتری.
- استفاده از تکنیک‌های مختلف طراحی در سطح معماری کامپیوتر (خط لوله و موازی‌سازی) باهدف افزایش سرعت پردازش و کارایی الگوریتم.

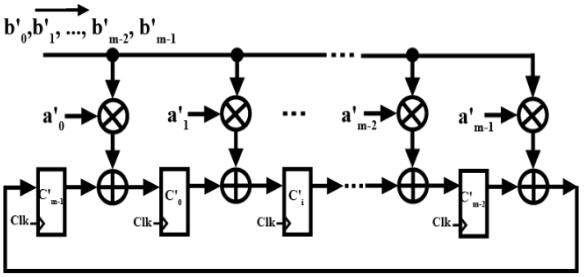
باوجود کارهایی که در این حوزه تحقیقاتی جدید و پر چالش انجام‌شده است، تعداد پیاده‌سازی سخت‌افزاری این الگوریتم‌ها در مقایسه با پیاده‌سازی نرم‌افزاری بسیار کمتر است و در این پژوهش قصد داریم تا در سطح معماری، کارایی الگوریتم *InvBR-LWE* را بهبود دهیم.

۱-۲. معرفی *LFSR*

یک شیفت رجیستر خطی فیدبک مجموعه‌ای از بیت‌ها است که هنگام راه‌اندازی شیفت داده می‌شوند و حالت بعدی تابعی خطی از حالت قبلی آن است و ممکن است از فیلد حسابی محدود بر روی $GF(2^m)$ [۶]، برای نمایش جزئیات *LFSR* استفاده شود. از شیفت به راست به‌عنوان عملیات شیفت و *XOR* به‌عنوان تابع خطی برای ایجاد حالت بعدی رجیستر استفاده می‌شود. این ساختار به‌طور گسترده برای تولید اعداد تصادفی در میکروکنترلرها مورداستفاده قرار می‌گیرد.

تراکم و پیچیدگی روزافزون طراحی‌های دیجیتال امروزی، مصرف انرژی پایینی را می‌طلبد. این برای اجزایی که به‌طور گسترده مورداستفاده قرار می‌گیرند، حیاتی‌تر می‌شود. *LFSR* به‌طور گسترده در مدارهای فشرده‌سازی داده، مدارهای رمزگذاری، خودآزمایی داخلی، مدارهای ارتباطی و مدارهای تصحیح خطا استفاده می‌شود [۷].

شکل (۱) با استفاده غیرمستقیم از *LFSR* در طرح *InvBR-LWE*، حاصل‌ضرب $A' \cdot x \bmod f'(x)$ (x ریشه چندجمله‌ای مشخصه $f'(x)$) را با m رجیستر یک بیتی انجام می‌دهد. ابتدا مقداردهی اولیه رجیسترهای درون مدار با مختصات عناصر A' و ضرایب f_i ($i = 1 \dots m-1$) بارگذاری می‌شوند و بعد از m پالس ساعت، رجیسترها ضرایب حاصل را ذخیره می‌کنند [۸]. ضرب مبتنی بر *LFSR* بر روی $GF(2^m)$ در شکل (۱) ارائه‌شده است [۹].



شکل (۱). معماری مبتنی بر *LFSR* [۱۰]

۱-۲-۱. آشنایی با رمزنگاری *Lattice-based*

مشبکه را می توان به عنوان مجموعه ای از نقاط تقاطع n بعدی بی نهایت، منظم، اما نه لزوماً متعامد تعریف کرد. به عنوان مثال، مجموعه بردارهای عدد صحیح Z^n یک مشبکه است. بردارهای مشبکه آنهایی هستند که کل مشبکه را در بر می‌گیرند. این بردارها ساختار تناوبی مشبکه را مشخص می‌کنند. پایه عالی پایه‌ای است که در آن بردارهای پایه تقریباً عمود باشند. در رمزنگاری مبتنی بر مشبکه، کلید خصوصی از پایه خوب و کلید عمومی از پایه ضعیف (بردارموازی) استفاده می‌کند. تعیین کوتاه

[1] *Digital Signature*
[2] *National Institute of Standards and Technology*
[3] *Post-Quantum Cryptography*
[4] *Lattice Base Cryptography*
[5] *Area Delay Product*
[6] *Adaptive Logic Modules*

ترین یا نزدیک ترین بردار نیز با بردار خوب یا عالی دست یافتنی‌تر است.

با توجه به n بردار مستقل خطی $b_1 , ... , b_n \in R^m$ مشبکه تولید شده توسط آنها مجموعه‌ای از بردارها است. تصور می‌شود که سیستم رمزنگاری مبتنی بر یادگیری با خطا *(LWE)* مؤثرترین سیستم رمزنگاری مبتنی بر مشبکه تا به امروز است [۱۱].

۱-۲-۲. آشنایی با رمزنگاری *LWE*

این سازوکار کپسوله‌سازی کلید *(KEM*[1]*)* که امنیت آن بر اساس سختی حل کردن مسئله یادگیری با خطا در مشبکه است، در مقابل حملات کامپیوترهای کوانتومی مقاوم است و توسط اودد رجو[2] در سال ۲۰۰۵ معرفی‌شده است. طرح‌های پساکوانتومی مبتنی بر *LWE* به دلیل پیچیدگی کمتر در میان طرح‌های رمزگذاری کلید عمومی *(PKE*[3]*)*، توجه زیادی را به خود جلب کرده‌اند. یادگیری با خطاها یک روش کوانتومی قوی رمزنگاری است که می‌تواند برای رمزگذاری هم‌ریختی نیز استفاده شود. مسئله یادگیری با خطا برای ساخت بسیاری از الگوریتم‌های رمزنگاری مبتنی بر مشبکه، برای به دست آوردن پیچیدگی محاسباتی کمتر استفاده‌شده است [۱۲].

Ring-LWE نوع سبکی از رمزنگاری *LWE* است که عملیات اصلی آن به یک ضرب چندجمله‌ای روی حلقه $Zq / (x^n + 1)$ تبدیل می‌شود و اندازه‌های کلید کوچک‌تری نسبت به *LWE* اصلی دارد [۱۳]. بسیاری از طرح‌های مبتنی بر مشبکه بر اساس سختی مسئله *LWE* یا انواع آن در برابر چندین نوع حمله مقاومت نشان داده است.

نوع سبک از *Ring-LWE*، معروف به *BR-LWE*[4]، برای استفاده در کاربردهایی با منابع محدود معرفی شد که نیازی به توزیع گاوسی و ضرب چندجمله‌ای مبتنی بر *NTT* ندارد. این روش از خطاهای باینری به‌جای گاوسی معمولی استفاده می‌کند که منجر به اندازه‌های کلید بسیار کوچک‌تر و اجرای پیچیدگی کم در مقایسه با هر دو الگوریتم *LWE* و *RLWE* می‌شود، درحالی‌که امنیت کافی برای برنامه‌های سبک‌وزن حفظ می‌شود [۱۴].

مطابق شکل (۲)، *PQC* مبتنی بر *BR-LWE* دارای سه فاز اصلی است:

تولید کلید (*Key Generation*)

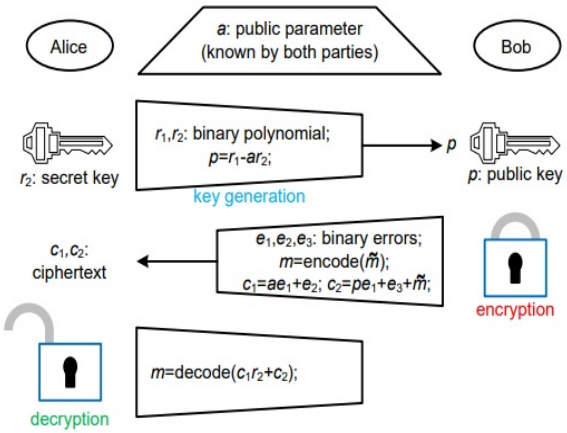
آلیس $p = r_1 - a. r_2$ را محاسبه و باب $p(nlog_2q)$ را به‌عنوان کلید عمومی دریافت می‌کند. (a یک پارامتر عمومی اشتراکی بین آلیس و باب و r_1 و r_2 کلید مخفی چندجمله‌ای باینری است که به‌طور تصادفی انتخاب شدند).

رمزنگاری (*Encryption*)

باب از خطاهای باینری e_1،e_2 و e_3 برای تولید متن رمزی c_1 و c_2 استفاده می‌کند و بعد به آلیس ارسال می‌شوند.

رمزگشایی (*Decryption*)

آلیس از r_2 برای بازیابی پیام اصلی m از طریق فرآیند رمزگشایی $(c_1r_2+ c_2)$ استفاده می‌کند، با توجه به آستانه رمزگشایی و اینکه ضریب در محدوده $(q / 4, 3q / 4)$ باشد.



شکل (۲). سه فاز اصلی طرح رمزنگاری مبتنی بر *BR-LWE* [۱۵]

با فرض اینکه q عددی صحیح $(q \in Z)$ است، مدول حلقه q به‌صورت $Zq = Z / qZ = \{0, 1,..., q - 1\}$ تعریف می‌شود. روش *BR-LWE* به ترتیب به امنیت کوانتومی ۷۳ بیتی و ۱۴۰ بیتی برای پارامترهای $(n, q) = (256,256)$ و $(n, q) = (512,256)$ دست می‌یابد و به‌خوبی برای برنامه‌های سبک‌وزن معمولی مناسب است.

طرح رمزنگاری مبتنی بر *InvBR-LWE* نوعی از *BR-LWE* است که در آن ضرایب چندجمله‌ای‌ها در Rq از Zq با محدوده $\{-\lfloor q/2 \rfloor, ... \lfloor q/2 \rfloor - 1\}$ انتخاب می‌شوند. این محدوده معکوس با نمایش مکمل دو مطابقت دارد. بنابراین ضرایب چندجمله‌ای در Rq را می‌توان بدون کاهش[5] بیشتر محاسبه کرد و هر ضریب پیام ورودی باید در $(-q/2)$ ضرب شود [۱۶].

در مقایسه با سایر الگوریتم‌های *PQC* مبتنی بر مشبکه، پردازنده‌های رمزنگاری مبتنی بر *BR-LWE*، شاخص بهره‌وری *ADP* بهتری دارند [۱۳].

۲. پیشینه پژوهش

تاکنون بهینه‌سازی‌های بسیاری برای *LFSR* ارائه‌شده است که در ادامه برخی از موارد که با الگوریتم *BR-LWE* مرتبط هستند را ذکر می‌کنیم.

[1] *Key Encapsulation Mechanism*
[2] *Oded Regev*
[3] *Public Key Encryption*
[4] *Binary Ring Learning With Error*
[5] *Reduction*

ژانگ [۱۷] یک معماری موازی *LFSR* طراحی کرده که مصرف انرژی پویا را کاهش می‌دهد و می‌تواند بیش از یک بیت خروجی در هر سیکل ساعت تولید کند.

همچنین در برخی پژوهش‌ها [۱۸-۲۰] پیاده‌سازی سخت‌افزاری با پیچیدگی پایین *PQC* مبتنی بر *BR-LWE* و مناسب بسترهای با منابع محدود بهبود داده شده است. برای مثال در [۱۸] پیاده‌سازی با سرعت بالا، تأخیر کمتری نسبت به طرح‌های اخیراً منتشرشده به دست می‌آورد. در تحقیق [۱۹] با ترکیب بهینه‌سازی‌های محاسبات کارآمد *NTT*، باعث می‌شود عملکرد ۱۹ درصد سریع‌تر از بهترین پیاده‌سازی قبلی شود. شهبازی و همکاران [۲۰] هم ضرب مبتنی بر ستون در بخش کاهش ماجولار در الگوریتم *Ring-BinLWE* ارائه کرده‌اند که یک شتاب‌دهنده رمزنگاری پساکوانتوم برای دستگاه‌های با منابع محدود *IoT* است. ضرب چندجمله‌ای عملیات اصلی شتاب‌دهنده محسوب شده که توسط شیفت رجیستر اجرا می‌شود و تأثیر زیادی بر کاهش مساحت و توان مصرفی مدار دارد.

ابراهیمی و همکاران [۱۶] با استفاده از اجرای طرح بهینه‌سازی شده*InvRB-LWE* متناسب با امنیت کلاسیک ۸۴ و ۱۹۰ بیت، سرعت، مساحت و توان مصرفی را بهبود داده‌اند. بطوریکه زمان رمزگشایی و پیچیدگی *AT* در مقایسه با پیاده‌سازی پژوهش [۲۱] به ترتیب حداقل ۶۸٪ و ۸۷٪ بهبود یافته است.

همچنین ژی و همکارانش [۲۱, ۲۲] با طراحی یک ساختار سخت‌افزاری برای محاسبات فیلد محدود $AB + C$ برای طرح رمزگذاری مبتنی بر *BR-LWE*، باعث تسریع کل فرآیند شده‌اند. معماری [۲۱] از طریق یک روش شبیه *LUT*[1] پیچیدگی *Area-time* طرح پیشنهادی، دارای ۹,۴۴٪ و ۱۶,۲٪ *ADP* کمتر (۲۱,۶٪ و ۲۱,۲٪ مساحت کمتر) نسبت به طرح‌هایی با سرعت‌بالا برای موارد$n = 256$ و $n = 512$ است که برای کاربردهای سبک‌وزن بسیار مطلوب است. معماری ارائه شده در [۲۲] با انعطاف‌پذیری سریع در گروه‌بندی پردازش‌ها به u گروه موازی، فرآیند محاسبات را سرعت می‌بخشد.

پژوهش‌های [۱۰, ۲۳] شامل دو معماری سخت افزاری کارآمد با استفاده از ساختارهای *LFSR* است. معماری اول سرعت پردازش را افزایش می دهد در حالی که معماری دوم به طور موثر پیچیدگی مساحت را کاهش می دهد. این ساختارها *ADP* کمتری دارند. ایمانا و همکاران [۱۰] دو معماری کارآمد مبتنی بر*LFSR* برای محاسبات مورداستفاده در طرح رمزنگاری مبتنی بر*InvBR-LWE* ارائه کرده‌اند که بهترین پیچیدگی‌های *Area-Time* را در بین تمام نتایج گزارش‌شده ارائه می‌کند و شامل *5.80٪* و *9.67٪* *ADP* کمتر نسبت به معماری اخیر [۲۴]، (با سبک‌های ورودی/خروجی پردازشی مشابه) روی دستگاه-*Virtex* 7است.

امیدوارکننده ترین بهینه سازی سخت افزاری برای طرح-*BR LWE* است. در این پژوهش، نویسندگان با استفاده از ترکیب الگوریتم به کارگرفته شده و محاسبات موازی، *ADP* بهتری را با بهبود مساحت و سرعت به طور هم زمان گزارش کرده‌اند [۲۵].

در برخی پژوهش‌ها *ADP* کمتر نسبت به طرح‌های اخیر حاصل شده است که می‌توانند برای استقرار در محیط‌های متنوع و سبک‌وزن توسعه یابند، اما زمان اجرا و تأخیر و تعداد سیکل‌های اجرا افزایش داشته و به منابع نسبتاً بیشتری هم محتاج است.

در روش پیشنهادی می‌خواهیم یک طرحی برای بهبود سرعت عملیات محاسباتی $A \cdot B \ MOD \ (x^n + 1) + C$ که برای *InvBR-LWE* مناسب‌سازی شده است را بر روی تراشه‌های *FPGA* به‌گونه‌ای بهینه کنیم که تأخیر اجرای کمی داشته باشد و درعین‌حال به منابع سخت‌افزاری زیادی احتیاج نداشته باشد. این ویژگی باعث کمتر شدن شاخص *ADP* که نشان‌دهنده کاهش حاصل‌ضرب تأخیر و مساحت است خواهد شد.

۳. روش پیشنهادی

تاکنون در پیاده‌سازی‌هایی که برای الگوریتم‌های پساکوانتوم مبتنی بر *LWE* و به‌ویژه *BR-LWE* جهت محاسبه $W = A \cdot B \ mod \ (x^n + 1) + C$ انجام‌شده است، از چندجمله‌ای با یک ضریب ورودی A و یک ضریب چندجمله‌ای باینری ورودی B استفاده‌شده است. حال اگر بتوان ضرایب، دو ورودی چندجمله‌ای A و ضرایب چندجمله‌ای باینری B را به دو گروه دوتایی تجزیه کرد، می‌توان عملیات محاسبه در دو مدار (زوج و فرد) به‌صورت موازی با تقسیم رجیسترها و سایر منابع مورداستفاده در بخش انباره انجام داد. درنتیجه تعداد سیکل‌های اجرا نصف و زمان اجرای کل الگوریتم کاهش می‌یابد که برای تحقق این امر لازم است تغییرات زیادی در *InvRB-LWE* رخ دهد.

در این معماری باید محاسبه دو ضریب چندجمله‌ای A و B، طبق روالی انجام شود که تغییر و تأثیری در رویکرد و عملکرد خروجی مدار یا همان W_i ها نداشته باشد. همچنین باید ترتیب ورود هریک از ضرایب A و B را بررسی نماییم و در نظر بگیریم که در مدارهای زوج و فرد طبق روال مشخص‌شده، وارد مدار شوند.

در طراحی‌های گذشته با توجه به محاسبات هریک از ضرایب چندجمله‌ای A و B، یک جمع کننده، یک گیت *AND* و یک گیت *Not* برای انجام فرآیند محاسباتی موردنیاز بوده است که برای تجزیه ضرایب چندجمله‌ای یا گروه‌بندی ضرایب پردازشی مناسب نبودند. ضرب دوبه‌دوی ورودی‌ها در هر یک از دو مدار زوج و فرد نیازمند دو جمع کننده، یک گیت *And* و یک گیت *Not* است. بنابراین در معماری و کد تغییراتی انجام شده که این

[1] *LookUp Table*

واحد با تجزیه ضرایب و پردازش در دو خط لوله هم‌زمان سازگار باشد.

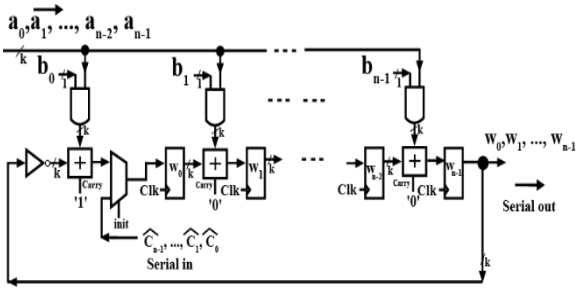
با گروه‌بندی ضرایب A و B و اجرا عملیات حسابی در دو مدار هم‌زمان (موازی) می‌توان انتظار داشت که برای انجام محاسبات به $n/2$ پالس ساعت نیاز داریم. هرچند مساحت مدار به‌طور نسبی کمی افزایش می‌یابد ولیکن با نصف شدن سیکل‌های اجرا، تأخیر کل و شاخص ADP کاهش می‌یابند.

۳-۱. معماری *Serial-In Serial-Out*

در این معماری ضرایب چندجمله‌ای اصلی هنگامی‌که با قالب ورودی سریال تغذیه می‌شوند، عمدتاً یک ساختار *Serial-In Serial-Out* است. این معماری برای کاربردهای با منابع محدود که به استفاده کمتری از مساحت در طرح‌های رمزنگاری PQC نیاز دارند، به کار می‌رود.

اخیرا برای عملیات حسابی ($W = A \cdot B \bmod (x^n + 1) + C$) در روش رمزگذاری مبتنی بر *InvBR-LWE*، مطابق شکل (۳)، طرحی با به‌کارگیری غیرمستقیم معماری مبتنی بر $LFSR$ ارائه‌شده است [۱۰]. این معماری از ثبات‌های k-bit، جمع کننده‌های k-bit، مالتی‌پلکسر k-bit $2:1$ $(k = \log_2 q)$ و همچنین گیت‌های AND برای انجام فرآیند محاسباتی استفاده می‌کند.

با توجه به اینکه بیشترین زمان اجرای الگوریتم‌های پساکوانتوم به بخش ضرب‌های چندجمله‌ای اختصاص دارد، با انجام محاسبات در $n/2$ پالس ساعت و نصف شدن سیکل‌های اجرا، تأخیر کاهش می‌یابد و تأثیر بسزایی در کاهش زمان کل اجرای الگوریتم مربوطه خواهد داشت.



شکل (۳). معماری مبتنی بر $LFSR$ [۱۰]

فرآیند به دست آوردن حاصل‌ضرب $[Ax^i]_j b_i$، با $ai \in Zq$ و $bi \in \{0,1\}$، توسط k گیت AND به‌صورت موازی (درمجموع nk گیت AND) انجام می‌شود. در این ساختار اگر b_j='1' باشد نتیجه $a_i b_j = a_i$ می‌شود و اگر b_j='0' باشد، حاصل $aibj$ = "0 ...00" (k بیت) است. سپس انباره حاصل $[Ax^i]_j b_i$ همراه با ضرایب مربوط به C می‌تواند در حلقه انباره مربوطه، نگاشت شود. نتایج حاصله از حاصل‌ضرب/حاصل‌جمع با مقادیر ثبات‌های k-bit با استفاده از جمع‌کننده‌های k-bit در $(w_i,\ i = 0 \ldots n\text{-}1)$ پس‌از n سیکل، ذخیره می‌شوند [۱۰]. ضرایب A و C برای محاسبه در هسته این معماری در داخل شیفت رجیسترها بارگذاری می‌شوند و در n سیکل ساعت تحویل داده می‌شوند. همچنین ضرایب B به شیفت رجیستر *Serial-In Serial-Out* برای تحویل سریال از $b_i (0 \le i \le n - 1)$، ضرایب مربوط به هسته معماری تغذیه می‌شوند.

نتایج تولیدشده در سیکل ششم که درواقع بخش نهایی از جزئیات عملیات محاسبه $A \cdot B \bmod (x^6 + 1) + C$ مبتنی بر معماری *Serial-In Serial-Out* است در جدول (۱) نمایش داده‌شده است.

جدول (۱). نمایش سیکل ۶ در محاسبه عملیات $A \cdot B \bmod (x6 + 1)$

Cycle	*Serial*	*Register-* w_5*	*Register-* w_0*	*Register-* w_1*	*Register-* w_2*	*Register-* w_3*	*Register-* w_4*
t_6	a_0	$a_0b_5+a_1b_4+a_2b_3+a_3b_2+a_4b_1+a_5b_0+c_5$	$a_0b_0-a_1b_5-a_2b_4-a_3b_3-a_4b_2-a_5b_1+c_0$	$a_0b_1+a_1b_0-a_2b_5-a_3b_4-a_4b_3-a_5b_2+c_1$	$a_0b_2+a_1b_1+a_2b_0-a_3b_5-a_4b_4-a_5b_3+c_2$	$a_0b_3+a_1b_2+a_2b_1+a_3b_0-a_4b_5-a_5b_4+c_3$	$a_0b_4+a_1b_3+a_2b_2+a_3b_1+a_4b_0-a_5b_5+c_4$

۳-۲. رویکرد *InvBR-LWE* در محاسبه $A.B + C$

ساختارهای رمزنگاری مبتنی بر مشبکه، مبتنی بر سختی مسائل مشبکه است [۲۶, ۲۷] و سیستم رمزنگاری مبتنی بر مشبکه، بر سختی مسئله LWE تکیه دارد [۲۸, ۲۹]. در LWE کلید خصوصی در $s \in Z_q^n = \{(x_0 \ldots . x_n)\}$، با $x_i \in Z_q$ در چندین زوج (a,b) با $b_i = a \cdot s + e_i$، قرار می‌گیرد. طرح مبتنی بر *Ring-LWE* [۲۱]، از مشبکه‌های ایده‌آل در LWE استفاده می‌کند و اندازه‌های کلید کوچک‌تری نسبت به طرح اصلی مبتنی بر LWE دارد.

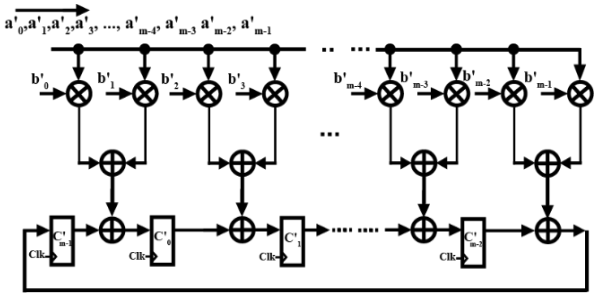
*Ring-LWE*شامل عملیات حسابی روی حلقه چندجمله‌ای $Rq = Zq\ [x] / (x^n + 1)$ است. جمع و تفریق این حلقه با جمع و تفریق ضرایب ماژول q مطابقت دارد و ضرب آن شامل ضرب چندجمله‌ای است که چندجمله‌ای درجه $(2n\text{-}2)$ را با مدول کاهنده $f(x) = x^n + 1$ به چندجمله‌ای درجه $(n\text{-}1)$ تبدیل می‌کند. همچنین ضرایب چندجمله‌ای حاصل نیز باید به مدول q کاهش یابند.

۳-۳. معماری مبتنی بر $LFSR$ برای عبارت $A.B + C$

با استفاده از این ساختار می‌توانیم معماری جدیدی را بر اساس کارکرد معماری شکل (۱) [۱۰]، با تجزیه ورودی‌ها به دو ضریب از A و B به همراه نصف نمودن رجیسترهای محدوده انباره در دو مدار جداگانه، موازی‌سازی نماییم. این رویکرد به بالاتر رفتن سرعت پردازش و کاهش تعداد سیکل‌ها از N به $N/2$ منجر می‌شود.

مطابق شکل (۴)، حاصل‌ضرب $A' \cdot x \bmod f'(x)$ با استفاده از یک $LFSR$با m رجیستر یک بیتی، ابتدا مقداردهی اولیه رجیسترها با مختصات عناصر A' و ضرایب f_i $(i = 1 \ldots m - 1)$بارگذاری شده و بعد از m کلاک، رجیسترها ضرایب حاصل را ذخیره می‌کنند.

همچنین نمادهای $\otimes$، $\oplus$ و $\square$ به ترتیب به گیت *AND*، جمع‌کننده و رجیستر اشاره دارد.



شکل (۴). طرح پیشنهادی مبتنی بر *LFSR* $A' \cdot B' \bmod f'(xm + 1)$

عملیات حسابی درگیر در طرح رمزگذاری مبتنی بر *InvBR-LWE*، $A \cdot B \bmod (x^n + 1) + C$ است که با به‌کارگیری غیرمستقیم معماری مبتنی بر *LFSR*، ابتدا دو گروه از A و B باهم ضرب و بعد از جمع آن‌ها، خروجی وارد چرخه اصلی *LFSR* می‌شود.

ضرب چندجمله‌ای $A \cdot B$ که در آن $A = \sum_{i=0}^{n-1} a_i x^i$ و $B = \sum_{i=0}^{n-1} b_i x^i$ با $a_i \in Zq$ و $b_i \in \{0, 1\}$، چندجمله‌ای درجه $(2n-2)$ تولید می‌کند. این چندجمله‌ای از طریق مدول $f(x) = x^n + 1$ کاهش می‌یابد تا $Xn \equiv -1$، $x^{n+1} = -x$ و ... و $x^{2n-2} = -x^{n-2}$ باشد.

برای محاسبه ضرب پیمانه‌ای: $A \cdot B \bmod (x^6 + 1)$ داریم:

$$= (a_5x^5 + \dots + a_1x + a_0) \cdot (b_5x^5 + \dots + b_1x + b_0) \bmod (x^6+1)$$

استفاده از دو ویژگی محاسباتی:

چندجمله‌ای B مورداستفاده در طرح رمزنگاری مبتنی بر *InvBR-LWE* دارای ضرایب باینری است؛ بنابراین در حاصل‌ضرب a_ib_j فقط اعداد صحیح با فرمت مکمل دو نشان داده می‌شود.

۱- محاسبه تفریق دو عدد صحیح را می‌توان به‌صورت:

$Z - Y = Z + C'_2(Y)$ محاسبه کرد.

۲- در $C'_2(Y) = C'_1(Y) + 1$ نیز $C'_1(Y)$ مکمل یک Y است. (توسط *NOT bit-wise* از k بیت‌های Y محاسبه می‌شود).

دو ویژگی بالا را می‌توان برای محاسبه عبارات حسابی بالا صرفاً با جمع استفاده کرد (تفریق‌ها توسط جمع‌ها، به‌صورت مکمل دو انجام می‌شود). عملیات الگوریتمی پیشنهادی دو ویژگی مهم ذکرشده را باهم ترکیب می‌کند که استفاده از منابع موردنیاز به حداقل می‌رسد.

۳-۴. معماری گسترش الگوریتم پایه

با استفاده از دو ویژگی بالا الگوریتم نهایی برای محاسبه عملیات اصلی طرح مبتنی بر *InvBR-LWE* را می‌توان ارائه کرد؛ بنابراین $W = AB + C = \sum_{i=0}^{n-1} w_i x^i$ و $C = \sum_{i=0}^{n-1} c_i x^i$ تعریف شده (برای $c_i, w_i \in Zq$) تا معادله به‌صورت رابطه (۲) بیان شود:

$$W = AB \bmod (x^n + 1) + C$$
$$= A(b_0 + \dots + b_{n-1}x^{n-1}) \bmod (x^n + 1) + C$$
$$= [Ab_0 + \dots + Ax^{n-1}b_{n-1}] \bmod (x^n + 1) + C, \qquad (2)$$

و درصورتی‌که $(x^n \equiv -1)$ باشد:

$$Ax = -a_{n-1} + a_0x + a_1x^2 + \dots + a_{n-2}x_{n-1},$$
$$Ax^2 = -a_{n-2} - a_{n-1}x + a_0x^2 + \dots + a_{n-3}x^{n-1},$$
$$\dots \dots \dots$$
$$Ax^{n-1} = -a_1 - a_2x - a_3x^2 - \dots + x^{n-1}, \qquad (3)$$

و درصورتی‌که جایگزینی با رابطه (2) به رابطه (4) می‌رسیم:

$$w_0 = a_0b_0 - a_{n-1}b_1 - \dots - a_1b_{n-1} + c_0,$$
$$w_1 = a_1b_0 + a_0b_1 - \dots - a_2b_{n-1} + c_1,$$
$$\dots \dots \dots$$
$$w_{n-1} = a_{n-1}b_0 + a_{n-2}b_1 + \dots + a_0b_{n-1} + c_{n-1}, \qquad (4)$$

Output: W=AB mod (X^n + 1) +C.
Initialization step
1 Make ready the input A, B, and C;
2 $\overline{Wo}$ =($\sum_{i=0}^{n-1}$ $\overline{W_i}X^i$)o; //executed in serial or parallel main step
3 $\overline{We}$ =($\sum_{i=0}^{n-1}$ $\overline{W_i}X^i$)e; //executed in serial or parallel main step
4 for i=0 to n - 1 do
5 $\overline{W_{io}}$ = $C'_2(c_{io})$;
6 $\overline{W_{ie}}$ = $C'_2(c_{ie})$;
7 end
8 for j=0 to n - 1 do
9 for i=0 to n - 1 do
10 $\overline{W}$jo=$\overline{W}$jo + $[AX^i]_{jo}$ b_{io}; //following(3)
11 $\overline{W}$je=$\overline{W}$je + $[AX^i]_{je}$ b_{ie}; //following(3)
12 end
13 end
14 Wo=$\overline{Wo}$;
15 We=$\overline{We}$;
Final step
16 deliver all the coefficients of output W serially;

شکل (۵). الگوریتم روش پیشنهادی برای عملیات $AB + C$ با ترکیب دو ویژگی محاسباتی [۱۰]

عملیات الگوریتم در شکل (۵)، دو ویژگی ذکرشده را باهم ترکیب می‌کند تا استفاده از منابع موردنیاز به حداقل برسد و همچنین اجازه می‌دهد تا بارگذاری ورودی و تحویل خروجی توسط همان متغیر میانی اجرا شود.

۳-۵. معماری پیشنهادی

با تمرکز بر معماری شکل (۳)، برای تسریع عملیات حسابی چندجمله‌ای $(W = A \cdot B \bmod (x^n + 1) + C)$ در طرح رمزنگاری مبتنی بر *InvBR-LWE* و تجزیه ضرایب چندجمله‌ای ورودی A و B (ضرایب چندجمله‌ای باینری) به دو گروه موازی (دو گروه زوج و فرد) جهت پردازش در دو مدار هم‌زمان تقسیم می‌شود و درنتیجه تعداد سیکل‌های اجرا نصف و سرعت پردازش افزایش می‌یابد.

ابتدا در دو مدار ورودی‌ها دوبه‌دو باهم *and* شده و نتیجه دو گیت *and* در یک جمع کننده، جمع شده، (فرآیند به دست آوردن حاصل‌ضرب $[Ax^i]_j$ b_i، با $ai \in Zq$ و $bi \in \{0,1\}$، توسط $2k$ گیت *AND* به‌صورت موازی (درمجموع $2nk$ گیت *AND*) انجام می‌شود) و خروجی حاصله هریک در اختیار انباره حاصل $[Ax^i]_j$ همراه با ضرایب مربوط به C نگاشت شود. در ادامه با همان b_i

رویکرد و عملکرد پژوهش [۱۰]، محاسبات خروجی هریک از W_i های زوج و فرد انجام می‌شود.

مثال برای محاسبه $W = A \cdot B \ Mod \ (x^6 + 1) + C$

جدول (۲)، سیرتکاملی محتویات ثبات‌های k-bit که در (w_0، w_1، w_2، w_3، w_4، w_5) ذخیره می‌شوند را نشان می‌دهد. خروجی نهایی حاصل از $[Ax^i]_j b_i$ همراه با ضرایب مربوط به C، در جدول مدار زوج (ذخیره در w_{n-2}) معکوس شده و تحت عنوان w_0 در خروجی قرار می‌گیرد. تمام ضرایب (a_0، a_1، a_2، a_3، a_4، a_5) چندجمله‌ای A در t_3+1 سیکل ساعت به‌صورت موازی در ساختار حاصل می‌شوند.

جدول (۲). نمایش سیرتکاملی محتویات ثبات‌ها در دو مدار موازی

Cycle	*Serial*	*Register-(for w_2*)*	*Register-(for w_4*)*	*Register-(for w_0*)*
init	-	$-c_2$	$-c_4$	$-c_0$
t_1	a_5,a_4	$a_4b_2+a_5b_1+c_0$	$a_4b_4+a_5b_3-c_2$	$-a_4b_0+a_5b_5-c_4$
t_2	a_3,a_2	$a_2b_2+a_3b_1+a_4b_0-a_5b_5+c_4$	$a_2b_4+a_3b_3+a_4b_2+a_5b_1+c_0$	$-a_2b_0+a_3b_5+a_4b_4+a_5b_3-c_2$
t_3	a_1,a_0	$a_0b_2+a_1b_1+a_2b_0-a_3b_5-a_4b_4-a_5b_3+c_2$	$a_0b_4+a_1b_3+a_2b_2+a_3b_1+a_4b_0-a_5b_5+c_4$	$-a_0b_0+a_1b_5+a_2b_4+a_3b_3+a_4b_2+a_5b_1+c_0$

Cycle	*Serial*	*Register-(for w_1*)*	*Register-(for w_3*)*	*Register-(for w_5*)*
init	-	$-c_1$	$-c_3$	$-c_5$
t_1	a_5,a_4	$a_4b_1+a_5b_0+c_5$	$a_4b_3+a_5b_2-c_1$	$-a_4b_5+a_5b_4-c_3$
t_2	a_3,a_2	$a_2b_1+a_3b_0-a_4b_5-a_5b_4+c_3$	$a_2b_3+a_3b_2+a_4b_1+a_5b_0+c_5$	$-a_2b_5+a_3b_4+a_4b_3+a_5b_2-c_1$
t_3	a_1,a_0	$a_0b_1+a_1b_0-a_2b_5-a_3b_4-a_4b_3-a_5b_2+c_1$	$a_0b_3+a_1b_2+a_2b_1+a_3b_0-a_4b_5-a_5b_4+c_3$	$a_0b_5+a_1b_4+a_2b_3+a_3b_2+a_4b_1+a_5b_0+c_5$

مطابق جدول (۲) در طول فرآیند مقداردهی اولیه، زمانی که *init* = '1' است ابتدا $\hat{c_i}$ ها در شش ثبات مربوط به مدارهای زوج و فرد موازی (مکمل دویی از ضرایب $\hat{c_i}$ چندجمله‌ای C) و همچنین ضرایب A در هر دو مدار زوج و فرد، در سه رجیستر k بیتی بارگذاری شده و b_5 از چندجمله‌ای B آماده می‌شود تا در سیکل بعدی به معماری وارد شود. در سیکل بعدی جدول (۲)، t_1، عبارات زیر به‌طور موازی محاسبه و در رجیسترهای مربوطه به ترتیب بارگذاری می‌شود (جدول (۲) - سیکل t_1).

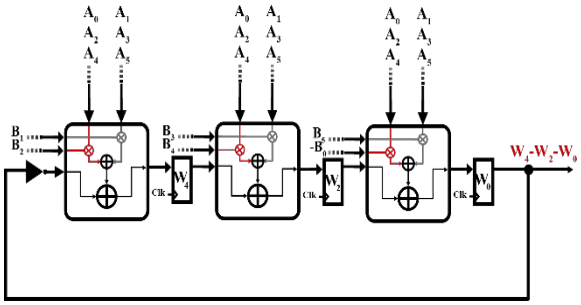
ورود a_4 , a_5	ورود a_4 , a_5	ورود a_4 , a_5
$-a_4b_0 + a_5b_5 - c_4$	$a_4b_4 + a_5b_3 - c_2$	$a_4b_2 + a_5b_1 + c_0$
و		
ورود a_4 , a_5	**ورود a_4 , a_5**	**ورود a_4 , a_5**
$-a_4b_5 + a_5b_4 - c_3$	$a_4b_3 + a_5b_2 - c_1$	$a_4b_1 + a_5b_0 + c_5$

بنابراین پس از فرآیند مقداردهی اولیه، رجیسترها حاوی ضرایب دقیقی از نتایج $W = A \cdot B \ mod \ (x^6 + 1) + C$ پس از سه سیکل ساعت (t_3) هستند که با مقادیر نهایی جدول (۱) و نمایش سیکل ۶ آن مطابقت دارند. همچنین نتایج پایانی در t_3 با مقادیر نهایی جدول (۲) مطابقت دارند و به‌طور دقیق همان خروجی برای w_0 تا w_5 تولید می‌شود(جدول (۲) - سیکل t_3).

نکته: ضرایب چندجمله‌ای باینری B در مدار زوج، با $n = 256$ از ضرایب B_i به ترتیب به‌صورت (b_0، b_{255} ، ...، b_2، $-b_1$) بارگذاری می‌شود، درحالی‌که ضرایب B_i در مدار فرد به ترتیب (b_1، b_0 ، ...، b_{255}، b_{254}) بارگذاری می‌شود و لازم است که مقدار b_0 مکمل شود.

۶-۳. رویکرد توسعه دو مدار زوج و فرد

در دو مدار موازی عملیات $W = A \cdot B \ mod \ (x^6 + 1) + C$ به‌طور هم‌زمان در دو مدار زوج و فرد اجرا می‌شوند، مقادیر مربوط به چندجمله‌ای عدد صحیح A و چندجمله‌ای باینری B از شیفت رجیسترهای مربوطه وارد هر دو مدار می‌شوند. معماری حاصل از رویکرد تجزیه عملیات محاسباتی در دو مدار زوج و فرد در شکل‌های (۶) و (۷) نشان داده‌شده است.



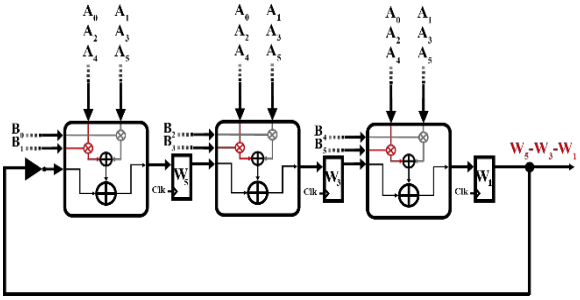
شکل (۶). معماری عملیات محاسباتی مدار زوج

همان‌طور که مشاهده می‌کنید در این مدار ضرایب A از شیفت رجیستر اصلی مدار (*LFSR* چرخشی) در گروه‌هایی دوتایی تجزیه‌شده و به‌صورت موازی و البته به همراه ضرایب C در یک سیکل اضافی، طبق مثال بالا ($W = A \cdot B \ mod \ (x^6 + 1) + C$)، در مدار قرار می‌گیرند یا به‌اصطلاح مقداردهی اولیه می‌شوند.

در سیکل دوم ابتدا عملیات *and* بین ضرایب A_5، A_4 ها و ضرایب B (با بارگذاری سریال به همان ترتیب شیفت رجیستر چرخشی و طبق شکل‌های (۶) و (۷))، انجام می‌شود، سپس باهم جمع می‌شوند. نتیجه با مقادیر رجیستر قبلی در $n/2$ سیکل به‌صورت چرخشی در انباره مدار (شامل جمع کننده و رجیسترها) برای جمع شدن با استفاده از یک *Adder* قرار می‌گیرند.

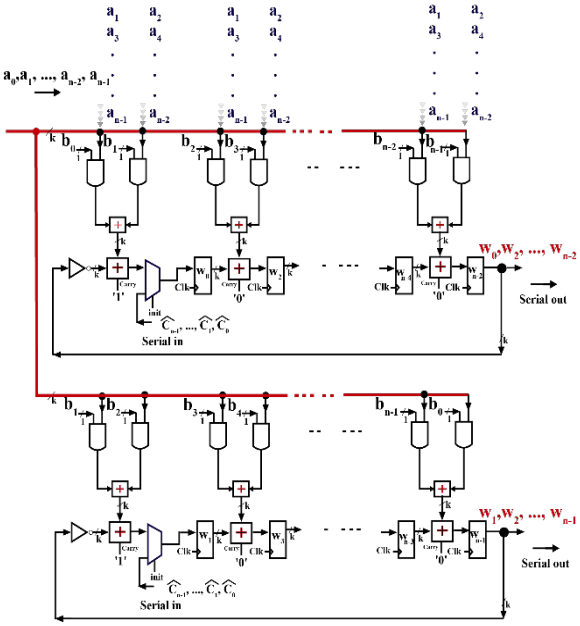
در سیکل سوم و چهارم نیز محاسبات، A_4، A_3 ها و A_2، A_1 ها به ترتیب با ضرایب B همانند سیکل دوم انجام می‌شود و درنهایت دو خروجی (w_i) به‌صورت سریال در هر سیکل در خروجی قرار می‌گیرند. شکل (۷) معماری عملیات محاسباتی مثال فوق را برای مدار فرد نشان می‌دهند.

عملکرد مدار فرد، به‌غیر از ترتیب ورودی ضرایب B از شیفت رجیستر چرخشی، تفاوتی با مدار زوج ندارد و ترتیب ضرایب A و C عیناً در دو مدار تکرار می‌شود.



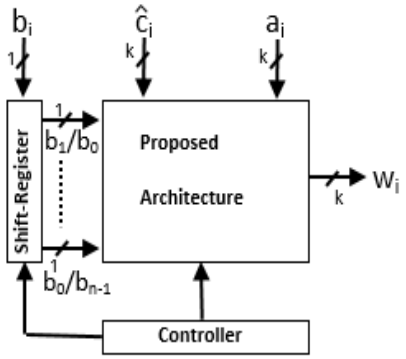
شکل (۷). معماری عملیات محاسباتی مدار فرد

جزییات معماری روش پیشنهادی مبتنی بر معماری *LFSR* در شکل (۸) قابل‌مشاهده است که شامل دو مدار زوج و فرد است.



شکل (۸). معماری روش پیشنهادی مبتنی بر *LFSR*

نمایش بلوک دیاگرام رویکرد پیشنهادی در شکل (۹)، جهت محاسبه $W = A \cdot B \bmod (x^n + 1) + C$ در R_q با $q = 2^k$ ارائه‌شده است.



شکل (۹). بلوک دیاگرام روش پیشنهادی

۴. ارزیابی نتایج پیاده‌سازی

با استفاده از روش پیشنهادی، تجزیه ضرایب چندجمله‌ای و اجرای هم‌زمان در دو مدار موازی روی *FPGA* های *Kintex-7* *(XC7K325t)* و *Virtex-7* *(XC7V2000t)* توسط *Vivado 2021.2* و *Stratix-V* *(5SGXMABN1F45C)* در *Quartus Prime 17.0* پیاده‌سازی شده است. نتایج به‌دست‌آمده براساس گزارش‌های نتایج سنتز ابزارهای *Vivado* و *Quartus* برای تراشه‌های فوق است.

طبق گزارش‌های به‌دست‌آمده، برای $n=256$ در فاز رمزگشایی روی *Virtex-7* و *Kintex-7* به ترتیب ۳۰۹۱ و ۴۱۱۲ فلیپ فلاپ و ۴۱۲۴ و ۴۰۵۳ *LUT* روی فرکانس ۲۹۵،۱۸ مگاهرتز نیاز است و توان مصرفی آن به ترتیب ۱۴۵ و ۱۸۵ *mW* است و برای $n=512$ نیز در فاز رمزگشایی روی *Virtex-7*، نیازمند ۵۵۹۱ فلیپ فلاپ، ۶۶۲۴ *LUT* روی فرکانس ۲۹۹،۷۴ مگاهرتز و توان مصرفی ۲۸۵ *mW* است.

جدول (۳) نتایج به‌دست‌آمده از *Vivado* را نشان می‌دهد. مطابق جدول، در ازای افزایش منابع مصرفی، زمان اجرا و تأخیر و تعداد سیکل‌های اجرا، نسبت به سایر پیاده‌سازی‌ها کاهش‌یافته است.

جدول (۳). نتایج پیاده‌سازی طرح پیشنهادی

Design	Platform	Parameter(n)	LUTs	FFs	Slice(ALMs)	Fmax	Latency	Delay	ADP	Power	EPC
روش پیشنهادی	*Virtex-7*	256	4124	3091	1166	295.18	128	495	**601**	145	**0.491**
	Kintex-7	256	4053	4112	1309	295.18	128	434	**568**	186	**0.630**
	Stratix-V	256	-	-	2975	228.35	128	434	**1291**	195	**0.855**
	Virtex-7	256	6624	5591	1776	295.18	256	865	**1668**	285	**0.966**

*Units of Fmax (Max. freq): MHz, Latency: refers to the computation time in clock cycles, Units for delay: ns. Delay=Critical path * Latency, where critical path=1/ (Max. freq), ADP = Number of Slices (ALMs)*Delay* 10^3, Power = represents dynamic power in mW, EPC=dynamic power / (Fmax * #output coefficient per cycle)[10].*

۴-۱. مقایسه پیچیدگی *Area-time* معماری پیشنهادی

در جدول (۴) زمان اجرای و تأخیر و تعداد سیکل‌های اجرا درروش پیشنهادی روی *FPGA* مدنظر با دیگر طرح‌ها مقایسه شده است. با توجه به پیچیدگی زمانی، می‌توان مشاهده کرد که حداکثر تاخیر مسیر بحرانی این معماری $T_{AND} + T_{ADD} + T_{MUX}$ است که در آن T_{AND}، T_{ADD}و T_{MUX} به ترتیب نشان‌دهنده تأخیر گیت *AND* ۲ ورودی، جمع کننده و مالتی پلکسر ۲:۱ است. محاسبات کل $W = A \cdot B \bmod (x^n + 1) + C$ به n سیکل ساعت (به‌علاوه یک سیکل اولیه اضافی برای *Initiate* ضرایب C در

ثبات‌ها)، بدون احتساب بارگذاری ورودی و خروجی چندجمله‌ای‌ها، نیاز دارد.

برای کاهش تأخیر این روش به‌جای استفاده از جمع کننده مواج[1] [۱۰]، از جمع کننده سریع با پیش‌بینی نقلی[2] استفاده‌شده است؛ بنابراین سرعت عملیات جمع بهتر شده است و برای انجام محاسبات به *n/2* پالس ساعت نیاز دارد.

جدول (۴). مقایسه پیچیدگی‌های *Area-time* معماری پیشنهادی

Design	*#AND*	*#INV*	*#Adder(k-bit)*	*#Mux(k-bit)*	*#Mux(2-bit)*	*#FF*	*#Clk*	*Critical Path*
[۲۶]	-	-	*2n*	*n*	-	*nk*	*n*	$T_{SUB}+T_{ADD}+T_{MUX3}$
[۱۶]	*nk*	*k+1*	*n*	*n+1*	-	*nk*	*n+1*	$\geq T_{AND}+T_{ADD}+T_{MUX}$
[۲۲] *(u=1)*	*(k+1)n*	*2n*	*n*	*1*	*n*	*nk*	*n*	$\approx T_{AND}+T_{ADD}+T_{MUX}$
[۱۰]	*nk*	*K*	*n*	*1*	-	*nk*	*n*	$T_{AND}+T_{ADD}+T_{MUX}$
روش پیشنهادی	***2nk***	***2k***	***2n***	***1***	***-***	***nk***	***n/2***	$\boldsymbol{T_{AND}+T_{ADD(CLA)}+T_{MUX}}$

در جدول (۴)، معماری پیشنهادی *ADP* کمتری نسبت به طرح [۱۰] دارد و با توجه به دو برابر شده تعداد *AND*، *Inv* و *Adder* نسبت به [۱۰]، مساحت این بخش بیشتر شده است، درحالی‌که تأخیر مسیر بحرانی افزایش کمی را دارد. به‌طورکلی، می‌توان نتیجه گرفت که این معماری ازنظر معیار *ADP* کارایی بالاتری نسبت به طرح‌های اخیر دارد. این شکل حاوی دو خط لوله رجیسترهای زوج و فرد است و باوجود زیادشدن نسبی مساحت مدار و بلوک‌های منطقی، تأخیر کل مدار *(delay)* به دلیل نصف شدن سیکل‌های اجرا *(Latency)* کاهش چشمگیری را در پی خواهد داشت.

در شکل (۸) به دلیل استفاده از دو واحد جمع کننده سریع در هر مرحله از خط لوله، تأخیر هر مرحله خط لوله نسبت به طرح‌های مشابه، کمی بیشتر شده است. مطابق جدول (۵) و (۶) طرح پیشنهادی در حالت *n=512* دارای ۳۵٪ *ADP* کمتر نسبت به معماری اخیر روی تراشه *Virtex-7* است.

برای *n = 256* در دستگاه *Kintex-7 FPGA*، معماری شکل (۸) دارای *10.58*٪ *ADP* کمتر از معماری [۱۰] است. همچنین توان مصرفی پویا معماری پیشنهادی نیز بهتر از طرح‌های موجود است. معیارهای توان و *EPC* در جدول (۵) نشان داده‌شده است. در جدول (۶)، نتایج پیاده‌سازی‌های مختلف روی *STRATIX-V* نشان داده‌شده است.

[1] *Ripple Carry Adder*
[2] *Carry Look-ahead Adder*

جدول (۵). مقایسه عملکرد پیاده‌سازی *FPGA* (*XILINX DEVICES*)

Design	*n*	*Phase**	*Device*	*LUT*	*FF*	*Slice*	*Fmax*	*Latency*	*Delay*	*ADP*	*Power*	*EPC*
[۲۶]	256	Dec.	Spartan 6	6728	6813	1874	101	262	2594	4861	-	-
[۲۲]	256	Dec.	Virtex-7	3600	2568	1146	415	256	617	707	233	0.561
[۱۶]	256	Dec.	Virtex-7	5153	2151	1701	261	257	985	1675	921	3.529
[۱۰]	256	Dec.	Virtex-7	2580	2340	907	349	256	734	666	140	0.401
روش پیشنهادی	**256**	**Dec.**	**Virtex-7**	**3091**	**4124**	**1166**	**285**	**128**	**495**	**601**	**145**	**0.491**
[۲۲]	512	Dec.	Virtex-7	7184	5128	2208	399	512	1283	2833	456	1.143
[۱۶]	512	Dec.	Virtex-7	10285	4249	3289	263	513	1951	6417	1871	7.114
[۱۰]	512	Dec.	Virtex-7	5650	4656	2230	379	512	1351	2559	420	1.108
روش پیشنهادی	**512**	**Dec.**	**Virtex-7**	**6163**	**8220**	**2299**	**296**	**256**	**865**	**1668**	**186**	**0.966**
[۲۲]	256	Dec.	Kintex7	3600	2568	1134	394	256	650	737	237	0.602
[۱۰]	256	Dec.	Kintex7	2836	2340	960	373	256	686	659	193	0.517
روش پیشنهادی	**256**	**Dec.**	**Kintex 7**	**4112**	**4053**	**1309**	**295**	**128**	**434**	**568**	**285**	**0.630**

*ADP=Slice*Delay, Delay=critical path*latency, EPC=power /(Fmax * #output coefficient per cycle)[10].*

جدول (۶). مقایسه عملکرد پیاده‌سازی *FPGA* (*STRATIX-V* (*DEVICES*

Design	*Alms*	*Fmax*	*Latency*	*Delay*	*ADP*
n=256 (Decryption Phase)					
[۱۶]	5734	369.14	257	696	3991
[۲۱]	4495	321.03	258	804	3691
[۲۸]	4446	379.22	257	678	3014
[۱۰]	2315	569.8	256	678	1039
روش پیشنهادی	**2975**	**228**	**128**	**434(869)**	**1291**
n=512 (Decryption Phase)					
[۱۶]	11470	336.36	513	1525	17492
[۲۱]	9038	317.06	514	1621	14651
[۲۸]	8864	327.98	513	1564	13863
[۱۰]	4620	437.25	512	1171	5410
روش پیشنهادی	**6169**	**165.24**	**257**	**865**	**5336**

*ADP=ALMs*Delay, Delay=critical path*latency.*

نتایج پیاده‌سازی مبتنی بر *FPGA* اینتل *(Stratix-V)* معماری پیشنهادی کارایی مشابه جدول (۵) دارند. برای *(n, q) =(256,256)*، این معماری شاخص *ADP* بسیار بهتری نسبت به [۱۶, ۲۱] دارد. بطوریکه حداقل *71.23*٪ *ADP* کمتر نسبت به [۲۱] و به‌طور مشابه، برای *(q ،n) =(256 ،512)* *60.98*٪ *ADP* کمتر از طراحی [۱۶] است.

در جدول (۷) زمان اجرای روش پیشنهادی با روش پیاده‌سازی [۲۲]، *(u=2)* روی *Virtex-7* و *Kintex-7*، مقایسه شده است.

پیاده‌سازی معماری پیشنهادی توانسته ازلحاظ تاخیر مسیر بحرانی و تعداد ثبات عملکرد بهتری داشته باشد.

جدول (۷). مقایسه عملکرد پیاده‌سازی *FPGA* (*STRATIX-V DEVICES*)

Design	*And*	*Adder (log_2q-bit)*	*Register*	*MUX (log_2q-bit)*	*Critical-Path*	*Latency*
[۲۲] *(u=1)*	$2nlog_2q$	$2n$	$n+nlog_2q$	*1*	$\approx T_A+T_{Mux2}+2T_{AD}$	$n/2$
روش پیشنهادی	$\mathbf{2nlog_2q}$	$\mathbf{2n}$	$\mathbf{n}$	**1**	$\approx \mathbf{T_A+T_{Mux}+2T_{AD}}$	$\mathbf{n/2}$

مقدار تأخیر روش پیشنهادی روی *Virtex-7* و *Kintex-7* نسبت به *n=256* [۲۲]، ۲۸,۷۴ درصد سریع‌تر و نسبت به [۲۲]، *(u=2)*، ۱۲,۳۴ درصد سریع‌تر است.

۴-۲. مقایسه کلی روش پیشنهادی با سایر روش‌ها

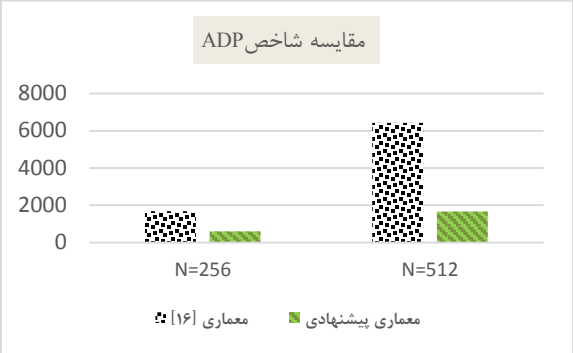
با وجود افزایش قابل‌قبول مساحت مدار نسبت به طرح مشابه، تعداد سیکل‌های اجرای الگوریتم نصف و درنتیجه زمان اجرای کل الگوریتم کاهش‌یافته و در بهترین حالت، معیار *ADP* ٪۳۵ کاهش‌یافته است. در کل نتایج به‌دست‌آمده روی *Virtex-7* و *Kintex-7* و *STRATIX-V* نشان می‌دهد که معماری پیشنهادی عملکرد مناسبی دارد.

۴-۳. شاخص بهره‌وری *ADP* (*Slice*Delay*)

این شاخص از حاصل‌ضرب *Slice Registers* ها (*ALM* در دستگاه اینتل)، در تأخیر به دست می‌آید و مقدار تأخیر براساس تأخیر مسیر بحرانی[1] محاسبه می‌شود. شاخص *ADP* هرچقدر کوچک‌تر باشد، بهتر است.

- **مقدار شاخص *ADP* در *Artix-7***

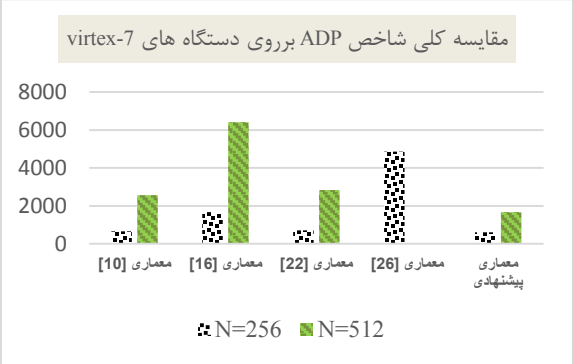
شکل (۱۰) که مقایسه مقدار شاخص *ADP* طرح پیشنهادی در *n=256* و *n=512* روی *Artix-7*، با معماری [۱۶] را نشان می‌دهد. طبق نتایج به‌دست‌آمده، معماری پیشنهادی بهترین نتیجه را به دست آورده و در *n=256* شاخص *ADP* کمتر از نصف شده و در *n=512* نیز *ADP* معماری پیشنهادی به کمتر از یک چهارم کاهش یافته است.



شکل (۱۰). مقایسه *ADP* معماری پیشنهادی و [۱۶] روی *Artix-7*

- **مقدار شاخص *ADP* در *Virtex-7***

شکل (۱۱) مقایسه مقدار شاخص *ADP* طرح پیشنهادی در *n=256* و *n=512* روی دستگاه‌های *Virtex-7*، با معماری‌های [۱۰, ۱۶, ۲۲, ۲۶] را نشان می‌دهد. طبق نتایج به‌دست‌آمده پیاده‌سازی پیشنهادی نتایج خوبی را در این پلتفرم به دست آورده است. در پارامتر *n=256* مقدار شاخص نسبت به معماری‌های مورد مقایسه، عملکرد بهتری دارد و در مقایسه با [۱۰]، ٪۱۲,۰۳ *ADP* کمتری دارد. کاهش شاخص در معماری‌های[۱۶, ۲۲]، به‌وضوح قابل‌مشاهده است. در *n=512* نیز شاخص *ADP* کاهش‌یافته وبرتری محسوسی نسبت به معماری [۱۶] دارد.

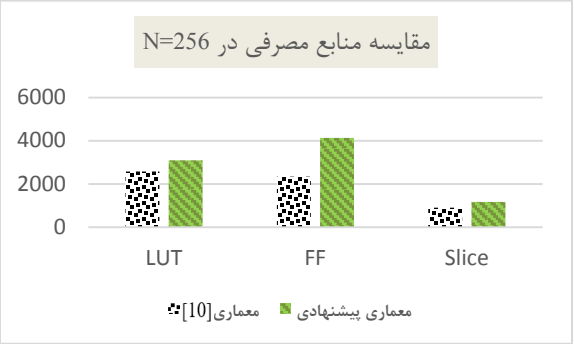


شکل (۱۱). مقایسه شاخص *ADP* معماری پیشنهادی و کارهای مشابه روی *Virtex-7*

- **مقایسه منابع مصرفی معماری پیشنهادی با [۱۰]**

شکل (۱۲) مقایسه مقدار منابع مصرفی در معماری [۱۰]، با معماری پیشنهادی در *n=256* روی *Virtex-7* را نمایش می‌دهد. پیاده‌سازی معماری پیشنهادی در قطعات مورد استفاده از حافظه (*Slice*) فاصله نزدیکی با طرح دیگر دارد. با توجه به موازی‌سازی و تجزیه ضرایب تفاوت نسبی کمی در تعداد رجیسترها و جدول‌های جستجو وجود دارد.

[1] *Critical Path Delay*



شکل (۱۲). مقایسه منابع مصرفی معماری پیشنهادی و [۱۰]، در *n=256* روی *Virtex-7*

۴-۴. شاخص بهره‌وری *EPC*[1]

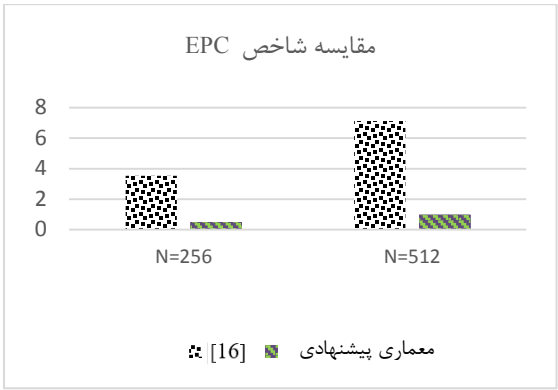
*(Power / (Fmax * #output Coefficient per Cycle))*

این شاخص از تقسیم توان مصرفی بر حاصل‌ضرب حداکثر فرکانس مورد استفاده در ضریب خروجی هر سیکل که در روش پیشنهادی برابر یک است به دست می‌آید. این شاخص مقدار انرژی مصرفی در محاسبات را براساس *n* به دست می‌آورد و عدد این شاخص هرچقدر کوچک‌تر باشد، بهتر است.

مقدار شاخص *EPC* در پلتفرم *FPGA*

- مقایسه *EPC* معماری پیشنهادی با معماری [۱۶]:

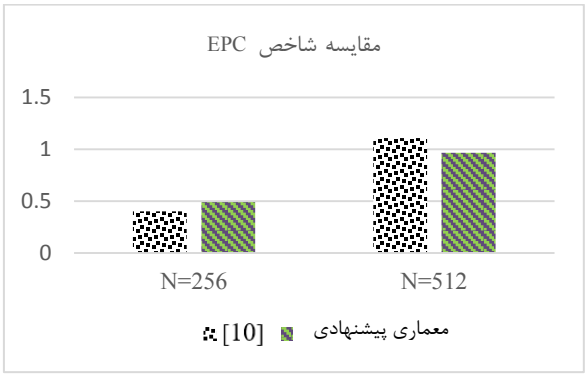
شکل (۱۳) مقدار شاخص *EPC* را در *n=256* و *n=512* نشان می‌دهد و معماری پیشنهادی بهترین مقدار را به دست آورده است. معماری [۱۶]، در *n=256* و نیز *n=512* نتایج ضعیفی دارد و چند برابر انرژی در محاسبات مصرف می‌کند.



شکل (۱۳). مقایسه شاخص *EPC* معماری پیشنهادی و [۱۶]، روی *Virtex-7*

- مقایسه *EPC* معماری پیشنهادی با معماری [۱۰]:

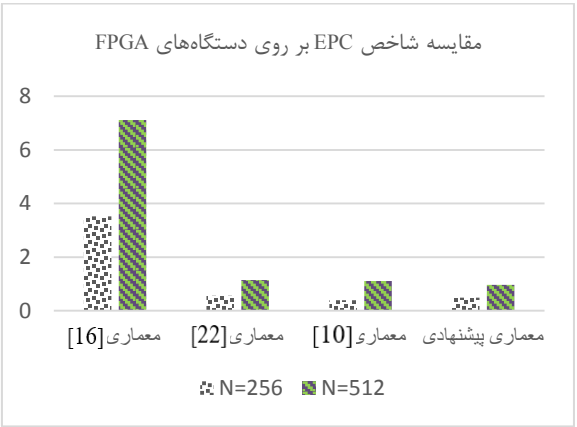
شکل (۱۴) مقدار شاخص *EPC* را در *n=256* و *n=512* نشان می‌دهد. معماری پیشنهادی در حالت *n=512* مقدار انرژی محاسباتی کمتری دارد ولیکن در *n=256* انرژی بیشتری را نسبتاً مصرف می‌کند.



شکل (۱۴). مقایسه شاخص *EPC* معماری پیشنهادی و [۱۰]، روی *Virtex-7*

- مقایسه کلی *EPC* معماری پینهادی با معماری‌های [۱۰, ۱۶, ۲۲]:

شکل (۱۵) مقایسه مقدار شاخص*EPC* طرح پیشنهادی در *n=256* و *n=512* روی دستگاه‌های *FPGA*، با معماری‌های [۱۰, ۱۶, ۲۲] را نشان می‌دهد. طبق نتایج به‌دست‌آمده معماری پیشنهادی نتایج خوبی را در این پلتفرم نشان می‌دهد. در *n=256* به‌غیراز معماری [۱۰]، کاهش شاخص به‌ویژه نسبت به معماری [۱۶]، به وضوح قابل‌مشاهده است. در*n=512* نیز کاهش محسوسی نسبت به معماری‌های مورد مقایسه وجود دارد، به‌طورکلی در هر دو پارامتر شاخص *EPC* کاهش داشته است.



شکل (۱۵). مقایسه شاخص *EPC* معماری پیشنهادی و کارهای مشابه

۵. نتیجه‌گیری

به علت قدرت بالای محاسباتی کامپیوترهای کوانتومی، در آینده این کامپیوترها می‌توانند الگوریتم‌های رمزنگاری کلید عمومی فعلی را بشکنند. برای مقابله با این موضوع الگوریتم‌های رمزنگاری پساکوانتوم به وجود آمدند. پیاده‌سازی الگوریتم‌های رمزنگاری پساکوانتوم با چالش‌های مختلفی، همچون منابع موردنیاز، زمان اجرا و میزان تأخیر روبه‌رو هستند. علاوه بر استفاده از سیستم‌های رمزنگاری امن در محیط‌های *IoT*، دستگاه‌های اینترنت اشیاء عموماً از باتری‌ها استفاده می‌کنند و محدودیت‌های فیزیکی مختلفی در ذخیره‌سازی، هزینه، قدرت و

[1] *Energy Per Computation*

انرژی دارند. با توجه به‌اندازه بزرگ کلید موردنیاز الگوریتم‌های پساکوانتومی و تمایل بیشتر به استفاده از منابع، سیستم رمزنگاری مناسبی با در نظر گرفتن رابطه بین اندازه کلید، سطح امنیت و عملکرد، با عنوان رمزنگاری مبتنی بر مشبکه، مدنظر قرار گرفته است. بلوک چالش‌برانگیز اصلی در رمزنگاری مبتنی بر مشبکه، ضرب چندجمله‌ای است که اندازه‌های کلید کوچک‌تری نسبت به طرح اصلی مبتنی بر *LWE* دارد.

برای حل این چالش‌ها روش‌های مختلفی پیشنهادشده است. از تغییر در بخش‌های محاسباتی و استفاده از تکنیک‌هایی در سطح معماری کامپیوتر، برای بهبود پیاده‌سازی و کوچک‌تر کردن شاخص *ADP* و کاهش تأخیر و زمان الگوریتم‌ها استفاده‌شده است.

در این پژوهش، روشی برای توسعه نوع سبک پیاده‌سازی سخت‌افزاری الگوریتم رمزنگاری *BR-LWE* باهدف کاهش تأخیر اجرای الگوریتم در دستگاه‌های با منابع محدود در اینترنت اشیاء ارائه شد تا عملیات حسابی به صورت موازی در دو مدار هم‌زمان محاسبه گردد. بنابراین زمان اجرا به *N/2* در مقایسه با سایر پیاده‌سازی‌ها کمتر شده و شاخص بهره‌وری *ADP* معماری پیشنهادی تا ۳۵٪ کاهش یافته است.

جهت توسعه کار در آینده، استفاده از گروه‌های ۴، ۸ و ۱۶ تایی از ضرایب چندجمله‌ای *A* و *B* در عملیات محاسباتی و پردازش موازی در مدارات هم‌زمان می‌تواند در زمان اجرا و کاهش تأخیر تأثیرگذار باشد، ولیکن پیچیدگی محاسبات بیشتر خواهد شد.

۶. مراجع

[1] T. M. Fernández-Caramés, "From pre-quantum to post-quantum IoT security: A survey on quantum-resistant cryptosystems for the Internet of Things," IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6457-6480, 2019, doi: https://doi.org/10.1109/JIOT.2019.2958788.

[2] S. Kumari, M. Singh, R. Singh, and H. Tewari, "A post-quantum lattice based lightweight authentication and code-based hybrid encryption scheme for IoT devices," Computer Networks, vol. 217, p. 109327, 2022, doi: https://doi.org/10.1016/j.comnet.2022.109327.

[3] H. Cheng, D. Dinu, J. Großschädl, P. B. Rønne, and P. Y. Ryan, "A lightweight implementation of NTRU Prime for the post-quantum internet of things," in Information Security Theory and Practice: 13th IFIP WG 11.2 International Conference, WISTP 2019, Paris, France, December 11–12, 2019, Proceedings 13, 2020: Springer, pp. 103-119, doi: https://doi.org/10.1007/978-3-030-41702-4_7.

[4] M. Schöffel, F. Lauer, C. C. Rheinländer, and N. Wehn, "Secure IoT in the era of quantum computers—Where are the bottlenecks?," Sensors, vol. 22, no. 7, p. 2484, 2022, doi: https://doi.org/10.3390/s22072484.

[5] A. Khalid, S. McCarthy, M. O'Neill, and W. Liu, "Lattice-based cryptography for IoT in a quantum world: Are we ready?," in 2019 IEEE 8th international workshop on advances in sensors and interfaces (IWASI), 2019: IEEE, pp. 194-199, doi: https://doi.org/10.1109/IWASI.2019.8791343.

[6] B. Liu and H. Wu, "Efficient architecture and implementation for NTRUEncrypt system," in 2015 IEEE 58th international Midwest symposium on circuits and systems (MWSCAS), 2015: IEEE, pp. 1-4, doi: https://doi:10.1109/MWSCAS.2015.7282143.

[7] M. Lowy, "Parallel implementation of linear feedback shift registers for low power applications," IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 43, no. 6, pp. 458-466, 1996, doi: https://doi: 10.1109/82.502318.

[8] F. Rodriguez, N. Saqib, A. D. Pérez, and C. Koc, "Cryptographic Algorithms on Reconfigurable Hardware," Springer, 2006.

[9] J. L. Imaña Pascual, "LFSR-based bit-serial GF (^ 2m) multipliers using irreducible trinomials," 2021, doi: http://dx.doi.org/10.1109/TC.2020.2980259.

[10] J. L. Imana, P. He, T. Bao, Y. Tu, and J. Xie, "Efficient hardware arithmetic for inverted binary ring-lwe based post-quantum cryptography," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 8, pp. 3297-3307, 2022, doi: http://doi:10.1109/TCSI.2022.3169471.

[11] B. S. Rawal and A. Biswas, "A comprehensive survey of post-quantum cryptography and its implications," Engineering Science & Technology, pp. 256-269, 2024, doi: https://doi.org/10.1145/3569457.

[12] J.-P. Aumasson, Serious cryptography: a practical introduction to modern encryption. No Starch Press, Inc, 2024.

[13] P. He, T. Bao, J. Xie, and M. Amin, "FPGA implementation of compact hardware accelerators for ring-binary-LWE-based post-quantum cryptography," ACM Transactions on Reconfigurable Technology and Systems, vol. 16, no. 3, pp. 1-23, 2023, doi: https://doi.org/10.1145/3569457.

[14] S. Ahmadunnisa and S. E. Mathe, "Multi-LFSR Architectures for BRLWE-Based Post Quantum Cryptography," IEEE Access, 2024, doi: https://doi:10.1109/ACCESS.2024.3426990.

[15] J. Xie, K. Basu, K. Gaj, and U. Guin, "Special session: The recent advance in hardware implementation of post-quantum cryptography," in 2020 IEEE 38th VLSI Test Symposium (VTS), 2020: IEEE, pp. 1-10, doi: https://doi:10.1109/VTS48691.2020.9107585.

[16] S. Ebrahimi, S. Bayat-Sarmadi, and H. Mosanaei-Boorani, "Post-quantum cryptoprocessors optimized for edge and resource-constrained devices in IoT," IEEE

Internet of Things Journal, vol. 6, no. 3, pp. 5500-5507, 2019, doi: https://doi:10.1109/JIOT.2019.2903082.

[17] J. Jung, H. Yoo, Y. Lee, and I.-C. Park, "Efficient parallel architecture for linear feedback shift registers," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 62, no. 11, pp. 1068-1072, 2015.

[18] Y. Tu, P. He, U. Guin, and J. Xie, "Low-Complexity Implementation of Lightweight Ring-LWE based Post-Quantum Cryptography."

[19] Z. Liu, R. Azarderakhsh, H. Kim, and H. Seo, "Efficient software implementation of ring-LWE encryption on IoT processors," IEEE Transactions on Computers, vol. 69, no. 10, pp. 1424-1433, 2017, doi: https://DOI10.1109/TC.2017.2750146.

[20] K. Shahbazi and S.-B. Ko, "Area and power efficient post-quantum cryptosystem for IoT resource-constrained devices," Microprocessors and Microsystems, vol. 84, p. 104280, 2021, doi: https://doi.org/10.1016/j.micpro.2021.104280.

[21] J. Xie, P. He, and W. Wen, "Efficient implementation of finite field arithmetic for binary ring-LWE post-quantum cryptography through a novel lookup-table-like method," in 2021 58th ACM/IEEE Design Automation Conference (DAC), 2021: IEEE, pp. 1279-1284, doi: https://doi:10.1109/TETC.2021.3091982.

[22] J. Xie, P. He, X. Wang, and J. L. Imana, "Efficient Hardware Implementation of Finite Field Arithmetic $ AB+ C $ A B+ C for Binary Ring-LWE Based Post-Quantum Cryptography," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 2, pp. 1222-1228, 2021, doi: https://doi:10.1109/TETC.2021.3091982.

[23] T. Bao, J. L. Imaña, P. He, and J. Xie, "Work-in-progress: High-performance systolic hardware accelerator for rblwe-based post-quantum cryptography," in 2022 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2022: IEEE, pp. 5-6.

[24] K. Seyhan, T. N. Nguyen, S. Akleylek, and K. Cengiz, "Lattice-based cryptosystems for the security of resource-constrained IoT devices in post-quantum world: a survey," Cluster Computing, vol. 25, no. 3, pp. 1729-1748, 2022, doi: https://doi.org/10.1007/s10586-021-03380-7.

[25] T. Liu, G. Ramachandran, and R. Jurdak, "Post-quantum cryptography for internet of things: a survey on performance and optimization," arXiv preprint arXiv:2401.17538, 2024, doi: https://doi.org/10.48550/arXiv.2401.17538.

[26] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," Journal of the ACM (JACM), vol. 60, no. 6, pp. 1-35, 2013, doi: https://doi.org/10.1007/978-3-642-13190-5_1.

[27] M. Ghafari, H. Abdoli, and M. Abbasi, "Speeding up the execution-time of Crystals-Kyber PQC Algorithm on FPGA," Electronic and Cyber Defense, vol. 10, no. 4, pp. 101-110, 2023, (in Persian). https://dor.isc.ac/dor/20.1001.1.23224347.1401.10.4.11.1

[28] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," Journal of the ACM (JACM), vol. 56, no. 6, pp. 1-40, 2009, doi: https://doi.org/10.1145/1568318.1568324.

[29] B. J. Lucas et al., "Lightweight hardware implementation of binary ring-LWE PQC accelerator," IEEE Computer Architecture Letters, vol. 21, no. 1, pp. 17-20, 2022, doi: https://doi:10.1109/LCA.2022.3160394.