دوفصل‌نامه علمی منادی امنیت فضای تولید و تبادل اطلاعات (افتا)
سال ۱۴۰۲، جلد ۲۱، شماره ۲ (۷۳–۸۰)
https://monadi.isc.org.ir

ارائه‌شده در بیستیمن کنفرانس بین‌المللی انجمن رمز ایران در سازمان پژوهش‌های علمی و صنعتی ایران، ۸ و ۹ شهریور ۱۴۰۲

پیاده‌سازی کارای ضرب چندجمله‌ای NTT مورد استفاده در الگوریتم رمزنگاری پساکوانتومی NTRU Prime بر روی FPGA★

رضا رشیدیان، راضیه سالاری فرد* و علی جهانیان

گروه معماری کامپیوتر، دانشکده مهندسی و علوم کامپیوتر، دانشگاه شهید بهشتی، تهران، ایران

اطلاعات مقاله

کلمات کلیدی:
رمزنگاری پساکوانتومی
ضرب چندجمله‌ای روی حلقه
NTRU Prime
Number
Theoretic
Transform

dor: 20.1001.1.24763047.1402.12.2.8.4

نوع مقاله: پژوهشی

چکیده

افزایش توان پردازشی کامپیوترهای کوانتومی، آسیب‌پذیری الگوریتم‌های کلاسیک را جدی‌تر خواهد کرد. بنابراین به‌کارگیری رمزنگاری پساکوانتومی بسیار ضروری است. الگوریتم‌هایی از گونه‌های مختلف رمزنگاری‌های پساکوانتومی همچون مشبکه‌مبنا و کدمبنا درحال طی مراحل نهایی استانداردسازی رقابت NIST هستند که انتظار می‌رود در برابر کامپیوترهای کوانتومی امن باشند. یکی از الگوریتم‌های KEM مشبکه‌مبنای مورد توجه این رقابت NTRU Prime است. چالش اصلی در پیاده‌سازی این گونه الگوریتم‌ها محاسبات سنگین ضرب چندجمله‌ای روی حلقه می‌باشد. با استفاده از Number Theoretic Transform ضرب چندجمله‌ای را می‌توان در زمان شبه‌خطی $\mathcal{O}(n \log n)$ انجام داد که عموماً از ساختار پروانه‌ای به منظور افزایش کارایی در پیاده‌سازی سخت‌افزاری آن استفاده می‌شود. در این مقاله ، دو ایده‌ی ادغام حافظه‌های قابل یکپارچه‌سازی و حذف تقارن داده‌های پیش‌پردازش شده مطرح و هریک جداگانه پیاده‌سازی گردیدند . کار ما در مقایسه با بهترین پیاده‌سازی های ضرب مورد استفاده در NTRU Prime روی FPGA تا پیش از این نگارش، با بازطراحی و اصلاح شیوه‌ی به‌کارگیری و ساختار ذخیره‌سازی داده‌های پیش‌پردازش شده سبب افزایش ۷ درصدی فرکانس و در عین حال کاهش توان مصرفی و کاهش بیش از ۱۴ درصدی منابع LUT به کار رفته شده است. کاهش تأخیر حاصل شده توأم با کاهش حجم منابع مصرفی افزایش کارایی را نشان می‌دهد.

© ۱۴۰۲ انجمن رمز ایران

۱ مقدمه

با پیشرفت روز افزون کامپیوترهای کوانتومی و ارائه‌ی الگوریتم‌هایی هم‌چون شور[۱] و گرور[۲] بسیاری از الگوریتم‌های مطرح رمزنگاری شامل سامانه‌های رمز کلید عمومی ، متقارن یا چکیده‌ساز[۳] شکسته خواهند شد و یا دست کم نیازمند کلیدها و خروجی‌ای با اندازه‌ی بزرگتر هستند [۱]؛ دلایلی این چنینی سبب می‌شوند که در آینده‌ای نزدیک جنبه‌های مختلفی از زندگی همچون امنیت تراکنش‌های بانکی، تعاملات اجتماعی

[1] Shor's Algorithm [2] Grover's algorithm [3] hashing algorithm

★از کمیته علمی بیستمین کنفرانس بین‌المللی انجمن رمز ایران برای داوری این مقاله تشکر می‌شود.

*نویسنده مسئول.
آدرس‌های رایانامه: RezaRashidian[at]iran.ir (رضا رشیدیان)، R_Salarifard[at]sbu.ac.ir (راضیه سالاری فرد)، Jahanian[at]sbu.ac.ir (علی جهانیان)

© ۱۴۰۲ تمامی حقوق متعلق به انجمن رمز ایران است.

مجازی و یا حتی اسرار و ارتباطات نظامی به طور جدی مورد تهدید قرار بگیرند [۲]. از این رو دو رویکرد کلی در مواجهه با این تهاجم قطعی و قریب‌الوقوع وجود خواهد داشت . نخست اینکه رمزنگاری مبتنی بر اصول مکانیک کوانتوم باشد که قابل شکستن نیست اما نیازمند سخت افزار و زیرساخت های ارتباطی جدید بوده که بسیار پر هزینه است . رویکرد دوم روی آوردن به رمزنگاری پساکوانتومی است . هرچند که این نوع رمزنگاری امنیت را به طور قطعی تأمین نمی‌کند اما صرفاً منوط به ایجاد تغییرات نرم‌افزاری بر روی بسترهای موجود است . مؤسسات و سازمان‌های مختلفی در خصوص پیشبرد اهداف رمزنگاری پساکوانتومی در حال فعالیت هستند که موسسه ملی فناوری و استانداردها [۱] (آمریکا) در حال انجام رویه‌ای برای انتخاب الگوریتم‌های رمزنگاری کلید عمومی از طریق یک رویه مسابقه مانند عمومی است [۳]. الگوریتم NTRU Prime یکی از موارد راه یافته به دورهای نهایی این رقابت هاست که در دسته‌ی الگوریتم های مبتنی بر مشبکه است . الگوریتم‌های رمزنگاری اغلب محاسبات پیچیده و سنگینی را تحمیل می‌کنند .در این الگوریتم نیز درصد قابل توجهی از پردازش‌ها مربوط به عملیات ضرب چندجمله‌ای‌ها روی حلقه می‌باشد. از این حیث بهبود کارایی ضرب‌کننده‌ها سهم بالایی در کارا کردن الگوریتم‌های رمز دارد . دو نوع پیاده‌سازی از NTRU Prime وجود دارد که در این مقاله Streamlined NTRU Prime مد نظر است . پیاده‌سازی‌های سخت‌افزاری و نرم‌افزاری محدودی بر بسترهای مختلف انجام گرفته است که کاراترین پیاده‌سازی سخت‌افزاری تا پیش از این مقاله مربوط به کار یانگ و همکارانش [۴] است که شرح داده خواهد شد . این نگارش شامل بخش‌های الگوریتم NTRU Prime، ضرب Number Theoretic Transform، کارهای پیشین، ایده‌ها و قابلیت‌های بهبود، پیاده‌سازی و اصلاح، مقایسه نتایج پیاده‌سازی، پیشنهادات و کارهای آتی و همچنین بخش نتیجه است.

۲ الگوریتم NTRU Prime

به منظور طراحی مکانیزم کپسوله‌سازی کلید [۲] به طور کلی سه الگوریتم تولید کلید، کپسوله‌سازی و کپسوله‌زدایی مورد نیاز است. در این الگوریتم عملیات ضرب چندجمله‌ای‌ها تابع قوانین محاسبات در فضای حلقه با پیمانه‌ی مشخص صورت می‌گیرد. الگوریتم‌های رمزنگاری مشبکه‌مبنای زیادی توسط حملات جدید شکسته شده‌اند و دلیل آن ساختار حلقه‌های استفاده شده در این الگوریتم‌ها است؛ الگوریتم رمزنگاری NTRU Prime با حذف این گونه ساختارها در برابر این تهدیدات مقاوم است [۵]. حلقه‌ی چندجمله‌ای تعریف شده در Streamlined NTRU Prime [۶] به صورت زیر است:

$$R/q = \mathbb{Z}/q[x]/(xp - x - ۱) \qquad (۱)$$

همچنین پارامترهای p، q، ω می‌بایست در شرایط زیر صدق کنند :

$$\omega > ۰;\ \omega \in \mathbb{Z};\ p, q \in \mathbb{P};$$

$$۲p \geq ۳\omega;\ q \geq ۱۶\omega + ۱.$$

پارامترهای پیشنهادی مد نظر برای sntrup761 نیز به صورت $p =$ ۷۶۱، $\omega =$ ۲۸۶ و $q =$ ۴۵۹۱ خواهد بود. در KEM ضرب‌های حائز اهمیت $R/۳ \cdot R/q$ و $R/q \cdot R/q$ انجام می پذیرد. روش‌های مختلفی برای پیاده‌سازی این نوع ضرب وجود دارد از جمله کاراتسوبا، ان‌تی‌تی و ضرب کلاسیک. به طور کلی ضرب دو چندجمله‌ای $A(x)$ و $B(x)$ در این حلقه به شکل ذیل خواهد بود :

$$A(x) \boxtimes B(x) \triangleq (A(x)B(x) \mod q) \mod x^p - x - ۱ \qquad (۲)$$

بیشترین هزینه محاسبات مربوط به عمل ضرب چندجمله‌ای‌ها با دو پیمانه‌ی q و $x^p - x - ۱$ است. جهت ضرب دو چندجمله‌ای ضرایب x می‌بایست به پیمانه q کاهش داده شده و ضرایب k حاصل از ضرب در بازه‌ای محدود به $۰ \leq k \leq q - ۱$ قرار گیرند. در NTRU Prime کاهش به پیمانه q ضرایب k را به دو قسمت مثبت و منفی تقسیم کرده و در بازه‌ی $-\lfloor \frac{q}{۲} \rfloor \leq k \leq \lfloor \frac{q}{۲} \rfloor$ قرار می‌دهد. چندجمله‌ای حاصل نیز در انتها به پیمانه‌ی $x^p - x - ۱$ کاهش می‌یابد.

۳ ضرب Number Theoretic Transform

تبدیل نظری اعداد (NTT) تعمیم‌یافته‌ی تبدیل فوریه گسسته (DFT) است که از ریشه اولیه‌ی واحد و یک حلقه به جای میدان اعداد مختلط استفاده می‌کند.

با توجه به شکل شماره ۱ در ضرب ان‌تی‌تی ابتدا باید تبدیل ان‌تی‌تی ضرایب چندجمله ای را محاسبه کرده و سپس با ضرب نقطه‌ای آنها و در نهایت محاسبه معکوس ان‌تی‌تی به حاصلضرب مد نظر رسید. این روش در مقایسه با دیگر روش‌های ضرب سریع مانند کاراتسوبا به طور مجانبی سریع‌تر است و پیچیدگی شبه‌خطی برابر $\mathcal{O}(n \log n)$ دارد. pاُمین ریشه‌ی واحد اولیه با ω نام‌گذاری می‌شود. برای محاسبه $\mathrm{NTT}(A[i])$ داریم:

$$\mathrm{NTT}(A[i]) = \sum_{j=۰}^{p-۱} A_j \omega^{ij} \mod q, \quad i = [۰, p-۱];$$

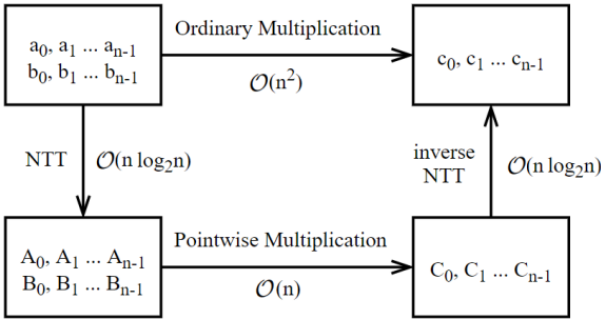
۱.۳ معماری پروانه‌ای

یک ساختار شبه تبدیل فوریه سریع [۳] کارا به نام معماری پروانه‌ای برای محاسبه‌ی ضرب ان‌تی‌تی وجود دارد . الگوریتم‌های تبدیل فوریه سریع با استفاده‌ی مجدد از نتایج محاسبات کوچکتر و متوسط سرعت بالاتری را نسبت به محاسبات کلاسیک ماتریسی فراهم می‌آورند . در این ساختار اگر که طول ورودی n و توانی از دو باشد به تعداد $\log_۲(n)$ لایه‌ی متشکل از واحدهای Cooley-Tukey مبنای دو برای محاسبه ان‌تی‌تی و

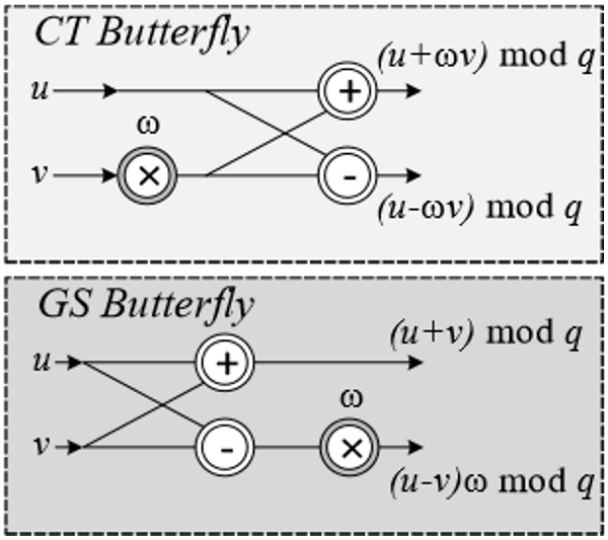
[1] National Institute of Standards and Technology

[2] key-encapsulation mechanism (KEM)

[3] Fast Fourier transform



شکل ۱. ضرب معمولی در مقایسه با ضرب NTT



شکل ۲. ساختار واحدهای محاسبه Cooley-Tukey (بالا) و ساختار Gentleman-Sande (پایین)

به همان تعداد واحد Gentleman-Sande برای محاسبه معکوس ان‌تی‌تی خواهیم داشت [۷]. در صورت نیاز مبناهای دیگری از این ساختار نیز قابل به‌کارگیری است [۸]. در شکل ۲ می‌بینیم که هر واحد پروانه [1] دارای سه عملیات جمع، ضرب و تفریق است. شبکه‌ای از این واحدهای پروانه‌ای‌شکل یک بلوک محاسبه ان‌تی‌تی و معکوس آن را تشکیل می‌دهند که با توجه به جایگاه و لایه‌ی آنها توان‌های مختلفی از ω در محاسباتشان نقش خواهد داشت.

در روند محاسبه ان‌تی‌تی و معکوس آن یک جایگشت [2] مطرح است؛ بدین معنی که ترتیب دنباله‌ی ورودی از ضرایب چندجمله‌ای ها بر اساس الگویی تغییر می‌کند. از آنجایی که در زمان انجام ضرب نقطه‌ای [3]، می‌بایست مؤلفه‌ها نظیر به نظیر در هم ضرب شوند این الگو اهمیت می‌یابد. در شکل ۳ این الگوی جابه‌جایی برای یک دنباله از ضرایب با طول ۱۶ قابل مشاهده است. در این شکل در هر زیرشاخه‌ی چپ از درخت مقادیر فرد از گره والد و در هر زیر شاخه‌ی راست اندیس‌های زوج گره والد انتخاب می‌شوند. در نهایت در پایین ترین سطح از درخت، اندیس‌ها با چیدمان جدید به دست می‌آیند.

[1] Butterfly unit [2] Permutation [3] Pointwise multiplication

۴ کارهای پیشین

یانگ، ماروتسکه و همکاران در مقاله‌ای [۴] پیاده‌سازی سخت‌افزاری الگوریتم sntrup761 را با استفاده از ضرب NTT، تکنیک گود [۹]، قضیه‌ی باقیمانده‌ی چینی، معماری پروانه‌ای و رویکرد استفاده از مقادیر پیش‌پردازش شده انجام داده‌اند.

با توجه به اینکه NTT بطور معمول بر روی چندجمله‌ای‌هایی به طول توان‌هایی از ۲ انجام می‌شود با استفاده از تکنیک گود و تغییر متغیر $x = yz$ رابطه‌ی (۲) به شکل رابطه‌ی (۳) در می‌آید:

$$A(x) \times B(x) = A(yz)B(yz) \mod (y^{۳}۱)(z^{۵۱۲} - ۱). \quad (۳)$$

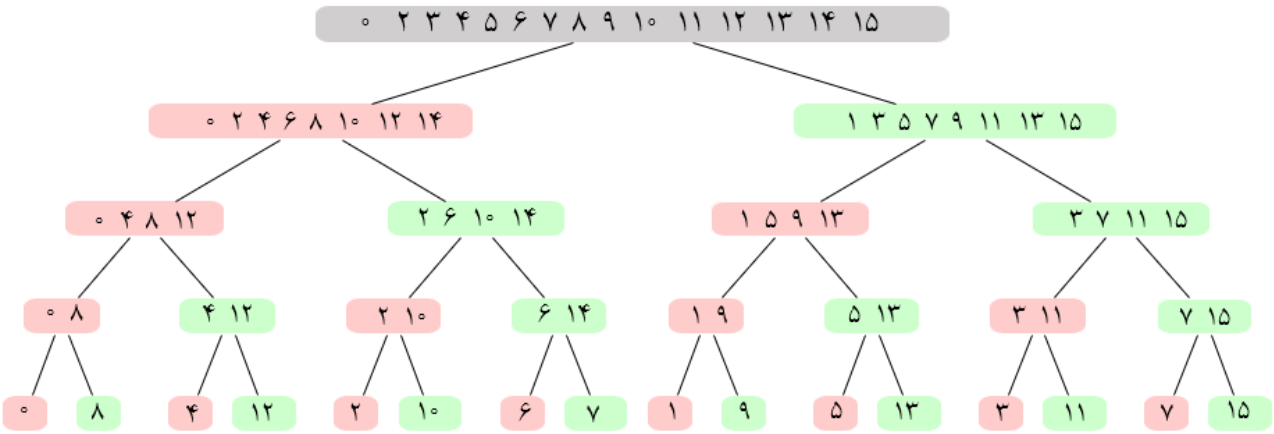
همچنین با استفاده از قضیه‌ی باقیمانده‌ی چینی سه پیمانه‌ی $q۱, q۲, q۳$ به ترتیب با مقادیر ۷۶۸۱, ۱۲۲۸۹, ۱۵۳۶۱ به دست می‌آیند که در سه بلوک پروانه‌ای مختلف استفاده می‌شوند. در مقاله [۳] نحوه‌ی محاسبه‌ی f_{y^j} برای تک تک ضرایب چندجمله‌ای‌های A, B شرح داده شده است.

شمای معماری کلی ضرب‌کننده پیاده‌سازی شده [۴] به صورت شکل ۴ می‌باشد.

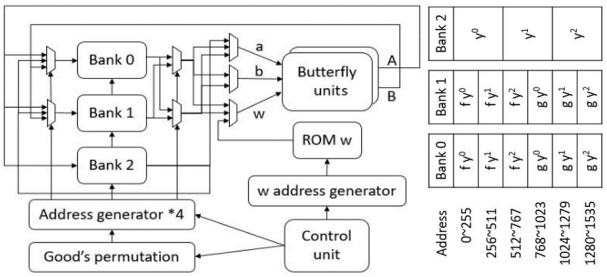
واحد کنترلی در این ساختار مراحل کلی ذیل را دنبال می‌کند که هر یک به طور مفصل شرح داده شده‌اند. بارگذاری، محاسبه ان‌تی‌تی، ضرب نقطه‌ای، بارگذاری مجدد، معکوس ان‌تی‌تی، اعمال قضیه‌ی باقیمانده‌ی چینی و مرحله‌ی کاهش. در این معماری از چندین ساختار حافظه‌ای فقط خواندنی و دسترسی تصادفی استفاده شده است. سه واحد پروانه‌ای مبنای دو نیز عملیات ان‌تی‌تی را به صورت تسریع شده انجام می‌دهند. سه بانک داده با استفاده از BRAMها در فرآیند محاسبات نقش ایفا کرده و چندین حافظه‌ی دیگر هم توان‌هایی از ω را به صورت پیش‌پردازش شده در خود ذخیره دارند. این پیاده‌سازی بر روی FPGAهای خانواده Xilinx Zynq Ultrascale+ انجام شده که نتایج آن را در ادامه مرور خواهیم کرد.

۵ ایده‌ها و قابلیت‌های بهبود

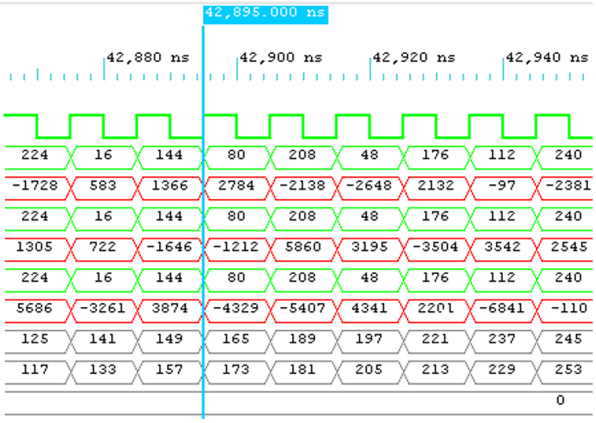
در ساختارهای پروانه‌ای به‌کار گرفته شده که محاسبه‌ی NTT را نسبت به محاسبات به شیوه‌ی ماتریسی و کلاسیک سرعت می‌بخشند توان‌هایی از ω نقش دارند. این ωهای به توان رسیده به صورت پیش‌پردازش شده در سه حافظه‌ی جداگانه ذخیره شده‌اند. با بررسی دقیق معماری دو نکته‌ی اساسی مشخص شد. نخست اینکه در برخی از حافظه‌های جدا از هم در یک کلاک واحد به آدرس‌های یکسانی از حافظه رجوع می‌شود. در شکل ۵ که نمایی از سیگنال‌های آدرس و مقادیر حافظه‌های پیاده‌سازی شده می‌باشد مشخص است که سه حافظه در زمان و کلاک ساعت مشخص مقادیری با آدرس‌های یکسان را به خروجی می‌برند.



شکل ۳. جایگشت اندیس‌ها در ساختار پروانه‌ای با $n = ۱۶$



شکل ۴. دیاگرام ضرب NTT با استفاده از تکنیک گود [۹]



شکل ۵. دیاگرام ضربسیگنال های آدرس و مقادیر خروجی حافظه

دوم اینکه در همین حافظه‌ها با مقادیر پیشفرض ω^n با n از صفر تا ۵۱۱ که از قبل محاسبه شده‌اند دو نوع تقارن وجود دارد به طوری که داریم $\mathrm{ROM}[i] = -\mathrm{ROM}[i + ۲۵۶]$ به این معنی که مقادیر مکان iام از حافظه با مقادیر مکان حافظه $i + ۲۵۶$ام با علامت مخالف برابری می‌کند.

۶ پیاده‌سازی و اصلاح

همانطور که در بخش ۴ بدان اشاره شد پیاده‌سازی یانگ و همکاران [۴] که به زبان Verilog با استفاده از Vivado انجام شده است امکان بهبود کارایی ضرب‌کننده وجود خواهد داشت. مقادیر پیش‌پردازش شده توان‌هایی از ω که در سه واحد محاسبه پروانه‌ای استفاده می‌شوند در سه حافظه‌ی جدا از هم ذخیره شده‌اند. در هر حافظه ۵۱۲ مقدار ۱۴ بیتی علامتدار قرار دارد. با توجه به اینکه واحدهای پروانه‌ای به طور موازی اجرا شده و هر یک به صورت جدا اما در کلاک یکسان مقادیر را از آدرس‌های یکسان حافظه خوانده و به کار می‌بندند می‌توان با ادغام و تجمیع این سه حافظه‌ی منفک از هم یک حافظه ROM با طول بیت ۴۲ تدارک دید که یک سیگنال آدرس مشترک خواهد داشت.

مقادیر ωهای به توان رسیده اعدادی علامت‌دار به فرم علامت مقدار هستند. با الحاق رشته بیت‌های به فرم نمایش مکمل دو آنها و ذخیره در حافظه با هر سیگنال آدرس ورودی در واقع سه مقدار مختلف مربوط به سه واحد پروانه‌ای به خروجی می‌رود . دومین ایده استفاده از خاصیت تقارن در داده‌های پیش‌پردازش و ذخیره شده در سه ماژول حافظه‌ای w7681، w12289 و w15361 بود. دیدیم که نیمی از هر سه حافظه یاد شده مقادیری مشابه نیمه‌ی دیگر دارند. به عنوان مثال در پیاده‌سازی ماژول w_12289 که مقادیر پیش‌پردازش شده‌ی توان‌هایی از ω برابر با ۳ را به پیمانه‌ی ۱۲۲۸۹ در خود نگه می‌دارد داریم که: در سطر اندیس صفر حافظه مقدار ω به توان صفر به پیمانه‌ی ۱۲۲۸۹ برابر یک است و همچنین در سطر ۲۵۶ حافظه مقدار ω به توان ۲۵۶ به پیمانه‌ی ۱۲۲۸۹ برابر ۱۲۲۸۸ یا همان ۱− از دید حلقه است . همین نکته عامل وجود تقارن یاد شده در هر سه ماژول می‌باشد . برای پیاده‌سازی ایده‌ی حذف تقارن به منظور کاهش منابع سخت‌افزاری اشغال شده می‌بایستی که با اعمال چند شرط کنترلی و دستور ، نیمه‌ی حذف شده‌ی حافظه ها را جبران کرد؛ به این صورت که ۲۵۶ سطر انتهایی و حذف شده‌ی حافظه ها با استفاده از مقادیر موجود در ۲۵۶ سطر ابتدایی قابل تولید و بازیابی

باشند. در قسمت پیوست نیز برای مثال مدار RTL یک حافظه رام کوچک‌شده فرضی با ۸ سطر حافظه ۶ بیتی که میتواند ۱۶ سطر مقدار درست را به خروجی ببرد آورده‌ایم؛ شکل ۶ نیز شبیه‌سازی آن را نشان می‌دهد که شروط کنترلی اعمال‌شده روی باس آدرس امکان تولید نیمه‌ی متقارن حذف شده را فراهم می‌کند.سیگنال‌ها به‌ترتیب Clk، Address و Output هستند.

در FPGAهای سری Zynq UlstaScale + شرکت زایلینکس حافظه های بلوکی RAMB36E2 در دسترس است که پیکربندی‌های متنوع با عرض بیت‌های مختلفی را شامل می‌شوند. از دیگر منابع مختلف در دسترس شامل DSP، URAM و LUT هاست. حافظه‌ها را می‌توان با اعمال قیدها و دستورات مربوطه بر بسترهای مختلف شکل داد و مشخص کرد که به صورت حافظه‌ای توزیع شده باشد یا حافظه‌ای یکپارچه. همچنین تعیین نوع ثباتی یا غیرثباتی بودن خروجی آنها روی زمان‌بندی تاثیر گذاشته و تک سیکل یا دو سیکل بودن حافظه را مشخص می‌سازد. در بخش بعدی نتایج به دست آمده‌ی کار ما در مقایسه با بهترین پیاده‌سازی قبلی آمده است.

۷ مقایسه نتایج پیاده‌سازی

همانطور که در بخش‌های قبل بیان شد دو ایده‌ی کلی مطرح کردیم. نخست ایده‌ی حذف تقارن مقادیر درونی حافظه‌های Rom و دیگری ایده‌ی اصلاح و ادغام حافظه‌های با کاربرد یکسان اما منفک از هم. در پیاده‌سازی سخت‌افزاری ایده‌ی حذف تقارن مقادیر پیش‌پردازش شده‌ی موجود در حافظه‌ها روی FPGA و تراشه‌ی ZUS+ دریافتیم که در مقایسه با بهترین پیاده‌سازی قبلی، این تغییر سبب کاهش ۸ درصدی منابع LUT و کاهش ۱۹ درصدی LUT RAMها گشته است. میزان سایر منابع سخت‌افزاری به کار گرفته شده همچون DSPها تقریباً بدون تغییر بود. از طرفی اما فرکانس به دست آمده برابر ۲۸۸٫۵ مگاهرتز بوده که در مقایسه با بهترین کار به میزان ۲٫۵۳٪ افت فرکانس و افزایش تأخیر مداری را شاهد بودیم. به نظر می‌رسد این افزایش تأخیر به علت شرط کنترلی و دستورات اجرایی هم زمان با لبه‌ی بالا رونده‌ی کلاک اعمال شده توسط ما در حافظه‌ها است.

از طرفی اما با اعمال ایده‌ی اصلاح ساختار و ادغام حافظه‌های هم‌شکل که با کلاک واحد مورد استفاده قرار می‌گیرند روی کار یانگ و همکاران پس از سنتز و پیاده‌سازی، نتایج کار ما در مقایسه با بهترین پیاده‌سازی پیش از این [۴] بهبود درخور توجه زمان و مساحت و به طور کلی کارایی را نشان می‌دهد. پیش‌تر یک پیاده‌سازی به روش کاراتسوبا [۶] نیز انجام گرفته است. نتایج حاصل شده در جدول ۱ آمده است. نتایج زیر حاصل گزارش پیاده‌سازی به زبان وریلاگ در نرم افزار Vivado 2019 می‌باشد.

نتایج به دست آمده نشان می‌دهد که کار ما باعث کاهش تأخیر مدار و در نتیجه افزایش ۷ درصدی فرکانس و همچنین کاهش مساحت و منابع سخت‌افزاری به کار رفته در FPGA شده است؛ به طوری که اشغال LUTها حدود بیش از ۱۴ درصد کاهش داشته است. این کاهش تأخیر

جدول ۱. نتایج پیاده‌سازی و مقایسه با کارهای پیشین

	Our best work	Our other work	[۴]	[۶]
Frequency (MHz)	۳۱۷٫۹۶۵	۲۸۸٫۵	۲۹۶٫۲۹	۲۷۹
LUT	۱۶۱۶	۱۷۴۳	۱۸۸۳	۶۲۴۰
LUT Ram	۳۴	۳۴	۴۲	–
FF	۱۳۴۷	۱۳۸۰	۱۳۴۱	۶۲۲۳
BRAM	۷٫۵	۷٫۵	۷٫۵	۹
DSP	۳	۳	۳	۳

حاصل‌شده توأم با کاهش حجم منابع مصرفی افزایش کارایی را نشان می‌دهد. علی‌رغم افزایش فرکانس و کارایی، توان مصرفی نیز کاهش یافته است. بر اساس دانش ما پیاده‌سازی ما کاراترین پیاده‌سازی تا قبل از این نگارش است.

۸ پیشنهادات و کارهای آتی

در بخش‌های پیشین گفتیم که نیمی از داده‌های موجود در حافظه‌های رام مربوط به نگهداری مقادیر ω^n تکراری البته با علامت مخالف هم می‌باشند؛ بنابراین انتظار می‌رود با ایجاد ساختاری کنترلی به شیوه‌ی درست بتوان با چک کردن اینکه آدرس ورودی به نیمه‌ی ابتدایی حافظه اشاره می‌کند یا نیمه‌ی انتهایی آن، مقادیر سطرهای ۲۵۶ تا ۵۱۱ام حافظه را هم بر اساس مقادیر موجود در سطرهای ۰ تا ۲۵۵ام حافظه تولید کرده و به خروجی برد. به طور کلی باید دید که با چه شیوه‌ای می‌توان این ایده را به کار گرفت در حالی که تأخیر زمانی چندان افزایش نیابد. دو ایده‌ی کاهش فضای حافظه با حذف تقارن و تجمیع حافظه‌های جدا از هم هر یک به طور جداگانه پیاده‌سازی و مقایسه شدند. ترکیب و اعمال هر دو ایده با هم نیز به نظر می‌رسد می‌تواند یک پیاده‌سازی قابل قبول از لحاظ بهینگی منابع مصرفی باشد.

پیشنهاد دیگر پیاده‌سازی و بکارگیری یک ساختار بهینه از نظر تأخیر و منابع در ضرب‌های داخل واحدهای پروانه‌ای است. در پیاده‌سازی ما در هر واحد پروانه‌ای یک ضرب اعداد علامتدار ۱۹ در ۳۳ بیتی وجود دارد که ممکن است قابل بهینه‌سازی باشد. این بخش شامل ضرب ورودی دوم واحد پروانه‌ای در ωهای مختلف است که به صورت پیش‌فرض توسط DSP انجام می‌شود. از طرفی استفاده از روش ضرب کاراتسوبا به دلیل اینکه مستلزم هم اندازه کردن دو عدد به وسیله‌ی بسط عدد کوچکتر است از نظر کارایی بهینه نبود.

همچنین پیاده‌سازی و اجرای این الگوریتم رمزنگاری با ضرب‌کننده‌ی کارای مطرح و پیاده‌سازی شده‌ی ما دراین نگارش بر بستر پردازنده‌های RISC با معماری آرم از قبیل ARM Cortex-M است.

از دیگر پیشنهاد ها برای کارهای آینده می‌توان به کاراتر کردن کاهنده‌های پیمانه، پژوهش در خصوص تعیین پارامترهای مناسب‌تر به منظور بهبود کارایی توام با تامین الزامات امنیتی و تحقیق در خصوص

ساختار و نوع منابع موجود در FPGA برای ذخیره‌سازی بهینه‌تر داده‌های پیش‌پردازش شده است.

۹ نتیجه

در این مقاله، مفاهیم و شیوه‌ی پیاده‌سازی ضرب کارای مورد استفاده در الگوریتم رمزنگاری پساکوانتومی NTRU Prime که یک الگوریتم رمز مشبکه‌مبنا است به اختصار بیان شد. همچنین ایده هایمان را به منظور بهبود کارایی بهترین پیاده‌سازی تا پیش از این مقاله مطرح کرده و با مقایسه نتایج حاصل شده ۷ درصد بهبود فرکانس و بیش از ۱۴ درصد کاهش منابع LUT گزارش شد. در بخش ۸ نیز چند پیشنهاد برای کارهای آتی مطرح گردید.

ضمایم

- سورس‌کد انتشار یافته‌ی پیاده‌سازی ما در مخزن گیت‌هاب:

https://github.com/hamidrezarashidian/Polynomial-Multiplication-Used-in-NTRU-Prime

- مجموعه پارامترها و سطوح امنیتی sntrup:

Scheme	security level	p	q	w
sntrup653	2	653	4621	288
sntrup761	3	761	4591	286
sntrup857	4	857	5167	322

- مدار RTL یک ROM کاهش‌یافته به روش حذف تقارن داده‌ها: شکل ۶.

مراجع

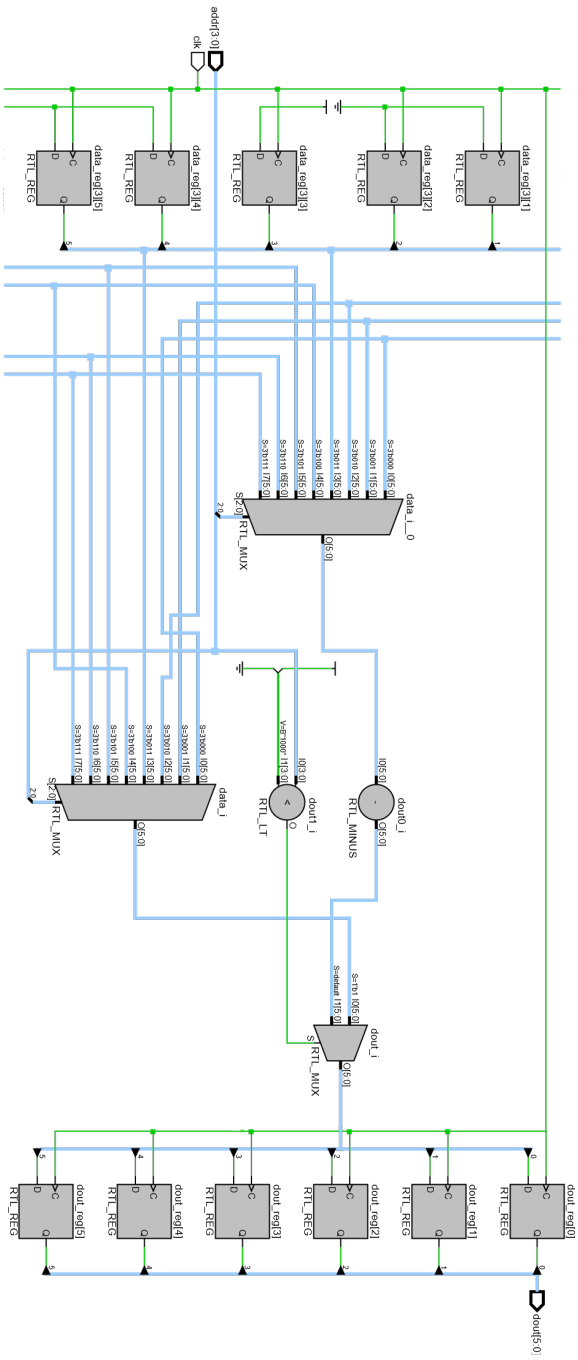
[1] Chen, Lily, Chen, Lily, Jordan, Stephen, Liu, Yi-Kai, Moody, Dustin, Peralta, Rene, Perlner, Ray A, and Smith-Tone, Daniel. *Report on post-quantum cryptography*, vol. 12. US Department of Commerce, National Institute of Standards and Technology …, 2016.

[۲] دوستی مطلق، نصیب اله و فاضلی، مهران. برنامه ریزی راهبردی رمزنگاری پساکوانتومی برای مقابله با تاثیرات رایانش کوانتومی بر روی رمزنگاری ها. فصلنامه فرماندهی و کنترل، ۴ (۴)، ۱۴۰۰.

[3] Alagic, Gorjan, Apon, Daniel, Cooper, David, Dang, Quynh, Dang, Thinh, Kelsey, John, Lichtinger, Jacob, Miller, Carl, Moody, Dustin, Peralta, Rene, et al. Status report on the third round of the nist post-quantum cryptography standardization process. *US Department of Commerce, NIST*, 2022.

[4] Peng, Bo-Yuan, Marotzke, Adrian, Tsai, Ming-Han, Yang, Bo-Yin, and Chen, Ho-Lin. Streamlined ntru prime on fpga. *Journal of cryptographic engineering*, 13(2):167–186, 2023.

[5] Bernstein, Daniel J, Chuengsatiansup, Chitchanok, Lange, Tanja, and van Vredendaal, Christine. Ntru prime:



شکل ۶. مدار RTL یک ROM کاهش‌یافته به روش حذف تقارن داده‌ها

reducing attack surface at low cost. in *Selected Areas in Cryptography–SAC 2017: 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers 24*, pp. 235–260. Springer, 2018.

[6] Marotzke, Adrian. A constant time full hardware implementation of streamlined ntru prime. in *International Conference on Smart Card Research and Advanced Applications*, pp. 3–17. Springer, 2020.

[7] Keersmaekers, Kris. A compact and flexible hardware accelerator for ntru polynomial multiplication using ntt. 2021.

[8] Alkim, Erdem, Cheng, Dean Yun-Li, Chung, Chi-Ming Marvin, Evkan, Hülya, Huang, Leo Wei-Lun, Hwang, Vincent, Li, Ching-Lin Trista, Niederhagen, Ruben, Shih, Cheng-Jhih, Wälde, Julian, et al. Polynomial multiplication in ntru prime: Comparison of optimization strategies on cortex-m4. *Cryptology ePrint Archive*, 2020.

[9] Good, Irving J. Random motion on a finite abelian group. in *Mathematical proceedings of the cambridge philosophical society*, vol. 47, pp. 756–762. Cambridge University Press, 1951.

[10] Bernstein, Daniel J., Chuengsatiansup, Chitchanok, Lange, Tanja, and van Vredendaal, Christine. NTRU prime. *IACR Cryptol. ePrint Arch.*, p. 461, 2016.

MonadiAfta-Bianual Scientific Journal for Cyberspace Security
2023, Volume 21, Number 2 (pp. 73–80)
https://monadi.isc.org.ir

Presented at the ISCISC 2023 in Iranian Research Organization for Science & Technology, Tehran, Iran

FPGA Implementation of Polynomial Multiplication in NTRU Prime PQC Algorithms★

Reza Rashidian, Raziyeh Salarifard * and Ali Jahanian

Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran

ARTICLE INFO.

Keywords:
Post Qountom Cryptography
Latticed-based Cryptography
NTRU Prime
Polynomial Multiplication

dor: 20.1001.1.24763047.1402.12.2.8.4

Type: Research paper

ABSTRACT

The adoption of post-quantum encryption algorithms to replace older asymmetric algorithms is of paramount importance. Diverse categories of post-quantum encryption, including lattice-based and code-based cryptography, are currently in the final stages of NIST's standardization competition, with the aim of providing security against quantum computers. Among the lattice-based key encapsulation mechanisms (KEM) garnering attention in this competition, the NTRU Prime algorithm stands out. The primary challenge in implementing such algorithms revolves around executing resource-intensive polynomial multiplications within a ring structure. Leveraging the Number Theoretic Transform (NTT) allows us to achieve polynomial multiplication with near-linear efficiency ($\mathcal{O}(n \log n)$). To enhance hardware efficiency, butterfly structures are frequently employed in NTT multipliers. Our research centers on comparing our approach with the best multiplication implementations utilized in NTRU Prime on FPGA up to the present version. This involves the redesign and modification of data preprocessing methods and storage structures, resulting in an increase in frequency and a reduction in the utilization of LUT resources.

© 2023 ISC

★ The ISCISC 2023 Program Committee effort is highly acknowledged for reviewing this paper.
* Corresponding author
Email addresses: r.rashidian@mail.sbu.ac.ir (Reza Rashidian), r_salarifard@sbu.ac.ir (Raziyeh Salarifard), jahanian@sbu.ac.ir (Ali Jahanian)
© 2023 ISC. All rights reserved.