مجله علمی پژوهشی «علوم و فناوری‌های پدافند نوین»
سال هفتم، شماره ۲، تابستان ۱۳۹۵؛ ص ۱۰۶-۹۵

پیاده‌سازی سخت‌افزاری سیستم‌های رمزنگاری بر اساس زوج‌سازی تیت با استفاده از FPGA روی $F_{2^{283}}$

محسن جهانبانی[1*]، زین‌العابدین نوروزی[2]، نصور باقری[3]

۱- دانشجوی دکتری ریاضی- رمز، ۲- استادیار، دانشگاه جامع امام حسین(ع)، ۳- استادیار، دانشگاه تربیت دبیر شهید رجائی

(دریافت: ۹۲/۰۶/۰۴، پذیرش: ۹۴/۱۱/۱۵)

چکیده

زوج‌سازی در رمزنگاری، یک نگاشت دوخطی از اعضای دو گروه جمعی از خم بیضوی به یک عضو گروه ضربی از میدان متناهی است و به‌منظور ساختن طرح‌های رمزنگاری یا حمله به آن‌ها مورد استفاده قرار می‌گیرد. زوج‌سازی تیت پرکاربردترین نوع زوج‌سازی است که با استفاده از الگوریتم میلر محاسبه می‌شود و نسخه بهبودیافته این الگوریتم برای خم‌های ابرمنفرد زوج‌سازی η_T نامیده می‌شود. به‌دلیل حجیم و زمان‌بربودن محاسبات زوج‌سازی تیت، پیاده‌سازی سخت‌افزاری آن بر پیاده‌سازی نرم‌افزاری ترجیح داده می‌شود. در این مقاله یک معماری جدید برای محاسبات زوج‌سازی تیت روی میدان $F_{2^{283}}$ پیشنهاد شده است. این معماری از ادغام دو بخش الگوریتم شامل زوج‌سازی و توان‌رسانی نهایی با استفاده از تکنیک به اشتراک‌گذاری منابع حاصل شده است. این پیاده‌سازی روی FPGAهای خانواده Xilinx انجام شده است. مقایسه نتایج این پیاده‌سازی بهبود ۳۸ درصد در زمان محاسبه و بهبود ۱۰ درصد برای معیار سطح در زمان را نسبت به کارهای دیگر نشان می‌دهد. در ضمن پیاده‌سازی نرم‌افزاری با استفاده از نرم‌افزار ریاضی SAGE به‌منظور آزمودن صحت جواب‌های به‌دست‌آمده و همچنین تولید نقاط روی خم، انجام شده است.

کلید واژه‌ها: زوج‌سازی تیت، زوج‌سازی η_T، میدان متناهی دودویی، خم بیضوی، معماری سخت‌افزاری، FPGA

Hardware Implementation of Cryptographic Systems Based on Tate Pairing Using FPGA on $F_{2^{283}}$

M. Jahanbani*, Z. Noroozi, N. Bagheri
Imam Hossein University

(Received: 26/08/2013; Accepted: 04/02/2016)

Abstract

Pairing-based cryptography is the use of a pairing between elements of two additive groups on elliptic curve to a third multiplicative group on finite field to construct or analyze cryptographic systems. The tate pairing is a common pairing and computed using Miller`s algorithm. Improved version for this algorithm on supersingular elliptic curve is named η_T. Pairing is quite computationally expensive and time consuming. Therefore, it is more attractive to implement on hardware rather than software. In this paper, a new architecture for computation of η_T pairing on $F_{2^{283}}$ is proposed. This architecture is resulted from merging two parts of algorithm including pairing and final exponentiation using resource sharing technique. The base of tate pairing is finite field arithmetic units. Design and synthesis of this implementation are conducted using Xilinx`s FPGA. Comparison between this result and other reports shows that this implementation gains % 38 improvements on calculation time and %10 improvement on product of area and time. Also software implementation is done by SAGE mathematical software for justifying of the results and generating points on elliptic curve.

Keywords: Tate Pairing, Elliptic Curve, Binary Finite Field, Hardware Architecture, FPGA.

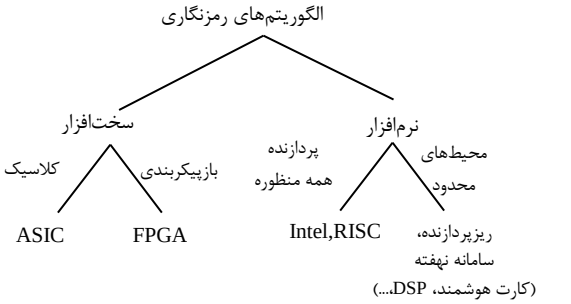
*Corresponding Author E-mail: mjahanbani@ihu.ac.ir

۱. مقدمه

در سال‌های اخیر سیستم‌های رمزنگاری بر اساس زوج‌سازی[1] روی خم‌های بیضوی به‌عنوان جایگزینی برای سیستم‌های رمزنگاری کلید عمومی معرفی شده‌اند. این سیستم‌ها دارای قابلیت ساخت پروتکل‌های جدید رمزنگاری که در گذشته امکان ایجاد آن‌ها نبود، هستند. رمزنگاری بر اساس هویت، تبادل کلید چندبخشی بر اساس هویت، امضای کوتاه، نمونه‌ای از این پروتکل‌ها است. زوج‌سازی روی خم‌های بیضوی به‌صورت یک نگاشت دوخطی از دو نقطه روی خم بیضوی به یک گروه ضربی است. معمول‌ترین روش‌های زوج‌سازی شامل تیت و ویل[2] است. زوج‌سازی تیت و ویل در رمزنگاری در آغاز برای حمله به مسئله لگاریتم گسسته برای دسته خاصی از خم‌های بیضوی روی میدان متناهی که به‌ترتیب به‌عنوان حمله MOV[3] [۱] و FR[4] [۲] شناخته می‌شوند، استفاده شد. این حمله‌ها مسئله لگاریتم گسسته روی این نوع خم را به مسئله لگاریتم گسسته روی میدان متناهی کاهش داد. سپس از خواص زوج‌سازی‌ها برای ساخت پروتکل‌های رمزنگاری استفاده شد. اولین بار میلر [۳] در یک دست‌نوشته منتشرنشده در سال ۱۹۸۶، الگوریتم محاسبه زوج‌سازی تیت و ویل را پیشنهاد داد. در این الگوریتم برای محاسبه زوج‌سازی تیت و به‌دست آوردن یک مقدار یکتا، یک توان‌رسانی نهایی روی خروجی الگوریتم میلر لازم است، اما به‌دلیل این‌که زوج‌سازی ویل به اجرای دوبار الگوریتم میلر نیاز دارد، زوج‌سازی تیت حداقل دو برابر سریع‌تر از زوج‌سازی ویل است. چندین بهینه‌سازی برای الگوریتم میلر برای زوج‌سازی تیت پیشنهاد شده است. ازجمله بارتو [۴] با توسعه ایده دورسما- لی [۵] زوج‌سازی اِتا کوتاه‌شده[5] (η_T) را معرفی کرد که تعداد تکرارهای الگوریتم میلر را به نصف کاهش داد.

تحقّق عملی تمام پروتکل‌های رمزنگاری به پیاده‌سازی کارآمد آن‌ها است و بسیاری از محققان در تلاش برای بهینه‌سازی الگوریتم‌ها و همچنین پیاده‌سازی‌ها هستند. شکل (۱) دسته‌بندی بسترهای مختلف پیاده‌سازی و جدول (۱) مقایسه آن‌ها را نشان می‌دهد.

پیاده‌سازی‌های نرم‌افزاری برای الگوریتم‌های رمزنگاری به‌خصوص زوج‌سازی به‌علت محاسبات سنگین روی خم بیضوی بسیار کند است. نتایج [۶ و ۷] این موضوع را به‌خوبی نشان می‌دهد. پیاده‌سازی بر روی سخت‌افزاری تک‌منظوره مانند مدارات مجتمع خاص‌منظوره (ASIC[6]) و آرایه‌های دروازه قابل برنامه‌ریزی (FPGA[7]) بستر ایده‌آلی برای پیاده‌سازی الگوریتم زوج‌سازی است. اگرچه ASIC دارای سرعت بالاتر و مصرف توان پایین‌تر نسبت بستر است، اما FPGAها به‌دلیل هزینه پایین‌تر، سرعت نمونه‌سازی، قابلیت بازپیکربندی و انعطاف‌پذیری برتری دارند. لذا در این پیاده‌سازی از بستر FPGA استفاده شده است.



شکل ۱. دسته‌بندی بسترهای پیاده‌سازی الگوریتم‌های رمزنگاری

جدول۱. مقایسه بسترهای پیاده‌سازی الگوریتم رمزنگاری

مشخصه/بستر	نرم‌افزار	ASIC	FPGAها
اندازه	کوچک (وابسته)	بزرگ	کوچک
هزینه	پایین	بالا	پایین
سرعت	پایین	خیلی بالا	بالا
انعطاف‌پذیری	انعطاف‌پذیری بالا	غیر انعطاف‌پذیر	انعطاف‌پذیری بالا
توان مصرفی	وابسته	پایین	بالا
خطایابی	آسان	سخت	آسان
زمان آماده‌سازی	کوتاه	خیلی طولانی	کوتاه
بازپیکربندی	ندارد	ندارد	دارد

شو [۸] اولین پیاده‌سازی سخت‌افزاری زوج‌سازی تیت را بر اساس دو الگوریتم تغییریافته بارتو و کون [۹] روی خم ابرمنفرد و میدان $F_{2^{283}}$ با سطح امنیت معادل AES-۷۲ بیت انجام داد. کلر و همکاران [۱۰] یک معماری دووضعیتی را با استفاده از الگوریتم بارتو روی خم بیضوی ابرمنفرد پیاده‌سازی کردند که یک پردازنده

[1] Pairing
[2] Tate and Weil
[3] Menezes, Okamoto and Vanstone
[4] Frey and Ruck
[5] Trunk Less Eta
[6] Application-Specific Integrated Circuit
[7] Field Programmable Gate Array

زوج‌سازی تیت و خم بیضوی بود و ضرب عددی نقطه روی خم را نیز محاسبه می‌کرد. لی و همکاران [۱۱] الگوریتم پیشنهادی کون را روی خم ابرمنفرد و میدان $F_{2^{283}}$ با سطح امنیت معادل AES-۷۲ بیت پیاده‌سازی کردند. رونن و همکاران [۱۲]، زوج‌سازی η_T را برای میدان $F_{2^{313}}$ با سطح امنیت معادل AES-۸۰ بیت پیاده‌سازی کردند. ویژگی اصلی کار آن‌ها استفاده از تکنیک خط- لوله، موازی‌سازی محاسبات و ساده‌سازی مسیر داده و کنترل، جهت افزایش سرعت بود. بیوچت و همکاران [۱۳]، زوج‌سازی η_T را بر اساس معماری عملگر یک‌پارچه پیاده‌سازی کردند. این معماری بدون استفاده از موازی‌سازی باعث افزایش فرکانس کار و در نتیجه کاهش زمان محاسبه شد. استیبلاس[1] [۱۴] یک پیاده‌سازی فشرده زوج‌سازی η_T را روی میدان سه‌ای و خم بیضوی ابرمنفرد $E(F_{3^{485}})$ با امنیت معادل AES-128 انجام داد. ایده پیاده‌سازی ایشان بر مبنای سریال‌سازی محاسبات جهت کاهش سطح مصرفی است که در مقابل منجر به افزایش بسیار زیاد زمان محاسبه شده است.

هدف اصلی این مقاله ارائه و پیاده‌سازی یک معماری سخت‌افزاری کارآمد جهت کاهش زمان محاسبه بدون افزایش چشم‌گیر سطح اشغالی است. رویکرد ما ارائه یک پیاده‌سازی بهبودیافته زوج‌سازی تیت برای خم‌های ابرمنفرد روی میدان دودویی است. این رویکرد شامل طراحی و پیاده‌سازی کارآمد محاسبات میدان به‌خصوص ضرب‌کننده‌ها است. برای تحقّق این موضوع استفاده هم‌زمان از ضرب‌کننده‌های سریال- رقمی LSD[2] [۱۵] و کاراتسوبا-اُفمن[3] [۱۶] پیشنهاد شده است. ضرب‌کننده LSD امکان برقراری موازنه بین سطح و زمان را با تغییر اندازه رقم ایجاد کرده و همچنین، استفاده از ضرب‌کننده کاراتسوبا درون حلقه‌های تکرار الگوریتم زوج‌سازی، باعث صرفه‌جویی زیادی در زمان شده است. چندین عامل در افزایش عملکرد این پیاده‌سازی نسبت به سایر پیاده‌سازی‌های مشابه مؤثر بوده که عبارت‌اند از:

- معماری مناسب سطح بالا به‌صورت ادغام دو بخش الگوریتم زوج‌سازی.
- انتخاب میدان دودویی که نسبت به میدان سه‌ای برتری دارد. اگرچه میدان با مشخصه سه، نیاز به اندازه میدان کوچک‌تری دارد، ولی نمایش اعضای این میدان نیاز به ۳ بیت دارد. به‌علاوه عملیات در میدان دودویی ساده‌تر از میدان سه‌ای است [۸].
- اشتراک‌گذاری منابع به‌صورت استفاده از ضرب‌کننده‌ها و مربع‌کننده‌ها در مواقعی که بیکار هستند.
- استفاده از ضرب‌کننده LSD با دو انباشته‌گر[4] جهت افزایش فرکانس کار.
- استفاده از ضرب‌کننده LSD و کاراتسوبا به‌صورت هم‌زمان جهت کاهش زمان محاسبه.

در بخش ۲ تعاریف مقدماتی و الگوریتم‌های زوج‌سازی ارائه می‌شود. الگوریتم‌ها و معماری‌های پیشنهادی و نتایج پیاده‌سازی واحدهای محاسباتی لازم در بخش ۳ و معماری سخت‌افزاری زوج‌سازی η_T و در بخش ۴ ارائه می‌شود. بخش ۵ شامل نتایج پیاده‌سازی سخت‌افزاری واحدهای محاسباتی و زوج‌سازی، مقایسه با دیگر کارها و پیاده‌سازی نرم‌افزاری جهت آزمودن صحت جواب است. بخش آخر شامل نتیجه‌گیری و پیشنهادهایی برای کارهای آینده است.

۲. تعاریف مقدماتی

فرض کنیم $F_q = F_{2^m}$ میدان متناهی دودویی و گروه نقاط یک خم بیضوی E تعریف‌شده روی میدان F_q با $E(F_q)$ نشان داده شود. یک زیرگروه $E(F_q)$ از مرتبه اول r زیرگروهی است که مرتبه گروه، یک عدد اول باشد. این زیرگروه دارای درجه تعبیه[5] k است، هرگاه رابطه $r \mid q^k - 1$ برای کوچک‌ترین k ممکن برقرار باشد. یک زیرگروه مرتبه r به‌عنوان گروه r- پیچش[6] شناخته می‌شود که با $E(F_q)[r]$ نشان داده می‌شود.

زوج‌سازی تیت از مرتبه r یک نگاشت دوخطی بین $E(F_q)[r]$ و $E(F_{q^k})[r]$ به یک عضو گروه ضربی $F_{q^k}^*$ به‌صورت زیر است:

$$e_r(P,Q): E(F_q)[r] \times E(F_{q^k})[r] \rightarrow F_{q^k}^* \qquad (۱)$$

ورودی دوم نگاشت می‌تواند روی $E(F_q)[r]$ تولید و سپس با استفاده از نگاشت اعوجاج (که با ψ نمایش داده می‌شود) به $E(F_q)[r]$ تبدیل شود. این موضوع منجر به تعریف زوج‌سازی

[1] Estibals
[2] Least Significant Digit-Serial
[3] Karatsuba-Ofman
[4] Accumulator
[5] Embedding Degree
[6] Torsion

تیت اصلاح‌شده به‌صورت زیر می‌شود:

$$\hat{e}_r(P,Q)=e_r(P,\psi(Q)) \qquad (۲)$$

که، $P,Q\in E(F_q)[r]$ و مقدار $\hat{e}_r(P,Q)$ به‌صورت یک کلاس هم‌ارزی است. چون در رمزنگاری به یک مقدار یکتا برای نمایش این کلاس نیاز است، باید مقدار زوج‌سازی تیت به توان $(q^k-1)/r$ برسد درنتیجه بخش‌های دارای توان r حذف می‌شوند و باقی‌مانده، r-اُمین ریشه واحد در F_{q^k} است. زوج‌سازی تیت کاهش‌یافته به‌صورت زیر تعریف می‌شود [۵]:

$$\hat{e}(P,Q)=\hat{e}_r(P,Q)^{(q^k-1)/r} \qquad (۳)$$

زوج‌سازی اِتای کوتاه‌شده (η_T) یک تکنیک بسیار کارآمد برای محاسبه زوج‌سازی تیت بوده و شامل یک حلقه تکرار و توان‌رسانی نهایی است.

۲-۱. محاسبات زوج‌سازی η_T

روش η_T برای محاسبه زوج‌سازی تیت روی خم‌های بیضوی ابرمنفرد است که توسط بارتو [۶] معرفی شد. خم‌های بیضوی ابرمنفرد روی میدان با مشخصه دو، دارای معادله به‌صورت $E(F_{2^m}): y^2+y=x^3+x+b$ است که $b\in\{0,1\}$ و m فرد و دارای درجه تعبیه $k=4$ است. تعداد نقاط گویای E روی F_{2^m} به‌صورت $N=\#E(F_{2^m})=2^m+1+\nu 2^{(m+1)/2}$ محاسبه می‌شود [۴] که در آن، $\nu=(-1)^{\delta}$. اگر $m\equiv 1 \bmod 8$ یا $m\equiv 7 \bmod 8$ باشد آن‌گاه $\delta=b$ و در غیر این‌صورت $\delta=1-b$ است. اعضای میدان $F_{2^{4m}}$ می‌توانند برحسب s و t نمایش داده شوند که $s^2=s+1$ و $t^2=t+s$ است. برای نمایش اعضای $F_{2^{4m}}$ با استفاده از یک توسیع F_{2^m} از پایه $\{1,s,t,st\}$ استفاده می‌شود به‌طوری‌که:

$$F_{2^{4m}}=F_{2^m}[s,t]\cong$$
$$F_{2^{4m}}[X,Y]\Big/\left\langle X^2+X+1,\ Y^2+Y+X\right\rangle \qquad (۴)$$

ψ یک نگاشت اعوجاج از $E(F_{2^m})[r]$ به $E(F_{2^{4m}})[r]$ برای هر $(x,y)\in E(F_{2^m})[r]$ به‌صورت زیر تعریف می‌شود:

$$\psi(x,y)=(x+s^2,\ y+sx+t) \qquad (۵)$$

محاسبات زوج‌سازی η_T در الگوریتم ۱ آورده شده است.

در محاسبه تعداد عملیات الگوریتم، A جمع، M ضرب، S مربع‌کردن و I معکوس ضربی را روی میدان دودویی F_{2^m} و XOR، یای انحصاری را نشان می‌دهند. همچنین $\bar{\delta}=1-\delta$ است. محاسبه خط شماره ۱۹ الگوریتم ۱ نیاز به الگوریتم جداگانه دارد که در بخش ۲-۲ آورده شده است.

الگوریتم ۱: محاسبه زوج‌سازی η_T با مشخصه دو [۱۳]

ورودی: $P,Q\in E(F_{2^m})[l]$.

خروجی: $\eta_T(P,Q)\in F_{2^{4m}}^*$.

۱. $y_p+\bar{\delta}\to y_p$ ($\bar{\delta}$ XOR)

۲. $x_p^2\to x_p,\ y_p^2\to y_p$ ($2S$)

۳. $y_p+b\to y_p,\ x_p+1\to u$ ($b+1$ XOR)

۴. $u+x_Q\to g_1$ (1 A)

۵. $x_p.x_Q+y_p+y_Q+g_1\to g_0$ (1 M, 3 A)

۶. $x_Q+1\to x_Q$ (1 XOR)

۷. $x_p^2+x_Q\to g_2$ (1 S, 1 A)

۸. $g_0+g_1s+t\to G$

۹. $(g_0+g_2)+(g_1+1)s+t\to L$ (1 A, 1 XOR)

۱۰. $L.G\to F$ ($2M,1S,5A,2$ XOR)

۱۱. برای $j=1$ تا $(m-1)/2$ انجام بده:

۱۲. $F^2\to F$ (4 S, 4 A)

۱۳. $x_Q^4\to x_Q,\ y_Q^4\to y_Q$ ($4S$)

۱۴. $x_Q+1\to x_Q,\ y_Q+x_Q\to y_Q$ (1 A, 1 XOR)

۱۵. $u.x_Q+y_P+y_Q\to g_0$ (1 M, 2 A)

۱۶. $x_P+x_Q\to g_1$ (1 A)

۱۷. $g_0+g_1s+t\to G$

۱۸. $F.G\to F$ (6 M, 14 A)

۱۹. F^M را بازگردان.

۲-۲. توان‌رسانی نهایی

همان‌گونه در بخش ۲-۱ بیان شد زوج‌سازی η_T به‌منظور تعیین مقدار یکتا باید کاهش یابد. بدین منظور $\eta_T(P,Q)$ را به M -اُمین توان می‌رسانیم که:

$$M=\frac{2^{4m}-1}{N}=\left(2^{2m}-1\right)\left(2^m+1-\nu 2^{(m+1)/2}\right) \quad (۶)$$

با بسط (۶)، $M=\left(2^{2m}-1\right)\left(2^m+1\right)+\nu\left(1-2^{2m}\right)2^{(m+1/2)}$ است، آن‌گاه:

$$\eta_T(P,Q)^M = \left(\eta_T(P,Q)^{2^{2^m}-1}\right)^{2^m+1}.\left(\eta_T(P,Q)^{\nu(1-2^{2m})}\right)^{2^{\frac{m+1}{2}}} \quad (۷)$$

به‌صورت U باشد. اگر $U=\eta_T(P,Q)\in F_{2^{4m}}^*$ فرض کنیم باشد، $t^{2^{2m}}=t+1$ که $U_0,\ U_1\in F_{2^{2m}}$ و $U=U_0+U_1t$ است. بنابراین: $U^{2^{2m}}=U_0+U_1+U_1t$ آن‌گاه

$$U^{2^{2m}-1}=\frac{U_0+U_1+U_1t}{U_0+U_t}=\frac{\left(U_0+U_1+U_1t\right)^2}{\left(U_0+U_t\right).\left(U_0+U_1+U_1t\right)}$$

$$=\frac{U_0^2+U_1^2+U_1^2s+U_1^2t}{U_0^2+U_0U_1+U_1^2s} \quad (۸)$$

$$U^{1-2^{2m}}=\frac{U_0+U_1t}{U_0+U_1+U_1t}=\frac{U_0^2+U_1^2s+U_1^2t}{U_0^2+U_0U_1+U_1^2s} \quad (۹)$$

که، $U_0^2+U_0U_1+U_1^2s\in F_{2^{2m}}$ است. توجه شود که توان‌رسانی نهایی همیشه به یک معکوس‌ضربی در $F_{2^{2m}}$ نیاز دارد. الگوریتم ۲ محاسبات $\eta_T(P,Q)^M$ را جمع‌بندی می‌کند.

۳. پیاده‌سازی زوج‌سازی η_T

شکل (۲) سلسه‌مراتب پیاده‌سازی زوج‌سازی را در حالت کلی نشان می‌دهد. در بالاترین سطح این سلسه‌مراتب، زوج‌سازی انجام می‌شود. سطح دوم شامل دو بخش ضرب‌های جمع‌شونده و توان‌رسانی نهایی و سطح سوم شامل عملیات روی میدان توسیع است. در پایین‌ترین سطح محاسبات میدان پایه شامل معکوس ضربی، ضرب، مربع و جمع انجام می‌شود.

الگوریتم ۲: توان‌رسانی نهایی زوج‌سازی کاهش‌یافته η_T [۱۳]

ورودی: $U=u_0+u_1s+u_2t+u_3st\in F_{2^{4m}}$

خروجی: $V=U^M\in F_{2^{4m}}$.

۱. $u_1^2\to m_1, u_0^2\to m_0$. (2 S)

۲. $u_3^2\to m_3, u_2^2\to m_2$. (2 S)

۳. $(m_0+m_1)+m_1s\to T_0$ (1 A)

۴. $(m_2+m_3)+m_3s\to T_1$ (1 A)

۵. $m_3+m_2s\to T_2$ (1 A)

۶. $(u_0+u_1s).(u_2+u_3s)\to T_3$ (3 M, 3 A)

۷. $T_0+T_2\to T_4,\ T_3+T_4\to D$. (4 A)

۸. $D^{-1}\to D$. (1I, 3M, 1S, 2A)

۹. $T_1.D\to T_5,\ T_4.D\to T_6$. (6 M, 8 A)

۱۰. $T_5+T_6\to V_0$. (2 A)

۱۱. $W_1, V_1\to T_5$.

۱۲. اگر $\nu=-1$ آنگاه:

۱۳. $V_0\to W_0$.

۱۴. در غیر این‌صورت:

۱۵. $T_6\to W_0$.

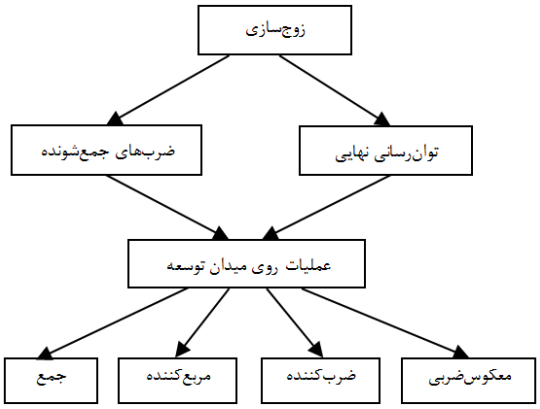
۱۶. $W_0+W_1t\to W,\ V_0+V_1t\to V$ (5 M, 2 S, 9 A)

۱۷. $V^{2^m+1}\to V$ (4 S, 4 A)

۱۸. برای $i\to 1$ تا $(m+1)/2$ انجام بده:

۱۹. $W^2\to W$ (9 M, 20 A)

۲۰. W . V را بازگردان.



شکل ۲. سلسله مراتب پیاده‌سازی زوج‌سازی

۳-۱. پیاده‌سازی واحدهای محاسباتی میدان دودویی

محاسبات میدان‌های متناهی به‌دلیل کاربرد گسترده در الگوریتم‌های رمزنگاری بسیار مورد توجه قرار گرفته است. پیاده‌سازی میدان‌های با مشخصه دو، به دلیل داشتن محاسبات بدون رقم نقلی، نسبت به میدان با مشخصه بزرگ‌تر آسان‌تر است. این موضوع نه‌تنها باعث سادگی معماری شده، بلکه سطح اشغالی سخت‌افزاری مورد نیاز را نیز کاهش می‌دهد. همچنین، جمع در این میدان به‌صورت XOR بیتی انجام می‌شود. در اینجا پیاده‌سازی سخت‌افزاری محاسبات میدان، بر اساس نمایش چندجمله‌ای است و برای ساخت میدان از یک پنج‌جمله‌ای استفاده شده که منجر به پیاده‌سازی کارآمد می‌شود [۸]. میدان به‌وسیله چندجمله‌ای $f(x)=x^m+\sum_{i=0}^{m-1} g_i x^i$ روی F_{2^m} تولید می‌شود. اگر α ریشه $f(x)$ باشد آن‌گاه نتایج محاسبات باید به پیمانه $f(\alpha)$ کاهش یابد. اعضای میدان دودویی F_{2^m} به‌صورت یک چندجمله‌ای هستند که ضرایب چندجمله‌ای صفر یا یک است. بنابراین اعضای میدان F_{2^m} به‌صورت یک رشته m بیتی و اعضای میدان $F_{2^{4m}}$ به‌صورت یک رشته $4m$ بیتی نشان داده می‌شوند که این بیت‌ها معادل ضرایب چندجمله‌ای هستند.

۳-۲. مربع‌کننده

مربع‌کننده روی میدان F_{2^m} برای عضو $A\in F_{2^m}$ به‌صورت بسط A به اندازه دو برابر طول بیت آن با قراردادن بیت‌های صفر بین بیت‌های اصلی A انجام می‌شود و سپس حاصل کاهش می‌یابد. معادله (۱۰) این مراحل را به‌خوبی نشان می‌دهد [۱۷]:

$$C=A^2 \bmod f(\alpha)$$

$$=(a_{m-1}\alpha^{2(m-1)}+a_{m-2}\alpha^{2(m-2)}..+a_1\alpha^2+a_0) \bmod f(\alpha) \quad (۱۰)$$

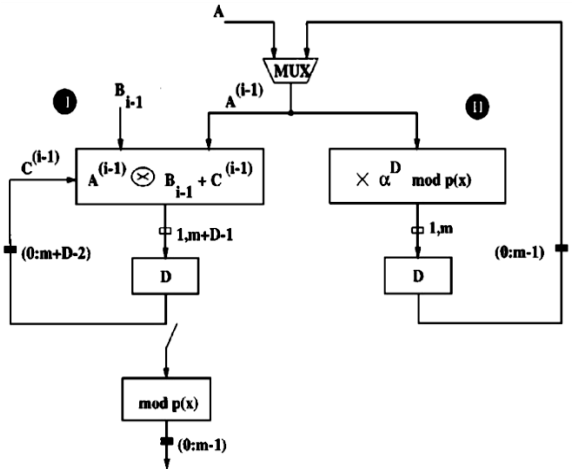
مربع‌کننده برای یک عضو از میدان $F_{2^{4m}}$ مانند $U=u_0+u_1 s+u_2 t+u_3 st\in F_{2^{4m}}$، به‌صورت $V=u_0^2+u_1^2 s^2+u_2^2 t^2+u_3^2 s^2 t^2\in F_{2^{4m}}$ به‌دست می‌آید [۱۳].

۳-۳. ضرب‌کننده

در میان عملیات میدان، ضرب پرکاربردترین و مهم‌ترین بخش است که کارایی سامانه را تعیین می‌کند. ضرب‌کننده سریال- رقمی که امکان موازنه بین سرعت و سطح اشغالی را فراهم می‌آورد، برای کاربردهای رمزنگاری با اندازه عملوندهای بزرگ مناسب است. این ضرب به‌وسیله پردازش چندین بیت در هر پالس انجام می‌شود. تعداد بیت‌هایی که به‌صورت موازی پردازش می‌شوند، اندازه رقم (D) گویند. اگر بیت‌ها از کم‌ارزش‌ترین به باارزش‌ترین پردازش شوند، ضرب‌کننده را LSD و پردازش معکوس را MSD گویند. چون ضرب‌کننده LSD دارای مسیر بحرانی کوتاه‌تری نسبت به MSD است، کارآمدی بالاتری دارد [۱۷]. انتخاب اندازه رقم خیلی کوچک به‌دلیل طولانی‌شدن زمان محاسبه مناسب نیست. همچنین، برای اندازه رقم‌های خیلی بزرگ، هرچند تعداد پالس ساعت کم‌تری برای ضرب لازم است، اما ضرب‌کننده سطح خیلی بالایی را اشغال کرده و همچنین به‌علت طولانی‌شدن مسیر بحرانی فرکانس کار کاهش می‌یابد. انتخاب بقیه اندازه رقم‌ها باتوجه به اهمیت معیار زمان محاسبه یا سطح اشغالی یا هردو صورت می‌پذیرد. این ضرب به‌صورت زیر محاسبه می‌شود [۱۵]:

$$C\equiv A.B \bmod f(\alpha)\equiv\Big[B_0 A+B_1\left(A\alpha^D \bmod f(\alpha)\right)$$
$$+B_2\left(A\alpha^D\alpha^D \bmod f(\alpha)\right)+...$$
$$+B_{d-1}\left(A\alpha^{D(d-2)}\alpha^D \bmod f(\alpha)\right)\Big]\bmod f(\alpha) \quad (۱۱)$$

شکل (۳) معماری ضرب‌کننده LSD را طبق فرمول (۱۱) نشان می‌دهد.



شکل ۳. معماری ضرب‌کننده LSD [۱۵]

دو حلقه در این شکل وجود دارد. ضرب جزئی در تکرار i-اُم، در حلقه ۱ محاسبه می‌شود و حلقه ۲ محاسبه $A^{(i-1)}.\alpha^D \bmod f(\alpha)$ را انجام می‌دهد. یک انباشته‌گر که از دروازه‌های XOR تشکیل شده است، ذخیره‌سازی ضرب‌های

جزئی و مقادیر میانی $C^{(i-1)}$ را انجام می‌دهد. واحد $\bmod f(\alpha)$ نتیجه نهایی را از $m+D-2$ به کم‌تر از m بیت کاهش می‌دهد. اگر در این ضرب‌کننده برای ذخیره‌سازی ضرب‌های جزئی از دو یا چند انباشته‌گر استفاده شود، مسیر بحرانی هسته ضرب‌کننده کاهش می‌یابد که باعث بهبود عملکرد آن خواهد شد.

ضرب‌کننده کاراتسوبا- اُفمن [۱۶] یک ضرب‌کننده موازی است و از یک الگوریتم بازگشتی استفاده می‌کند که نسبت به دیگر ضرب‌کننده‌های موازی از جمله ضرب‌کننده‌های کلاسیک، پیچیدگی مداری کم‌تری دارد. در این ضرب‌کننده در هر تکرار ضرب‌شونده‌ها به دو نیمه تقسیم می‌شوند و این تقسیم‌شدن تا زمانی ادامه پیدا می‌کند که حاصل‌ضرب سطح در زمان کم‌ترین شود. آن‌گاه بیت‌های باقی‌مانده به‌صورت ضرب موازی محاسبه می‌شوند. تعداد تکرار بهینه با استفاده از آزمون و خطا به‌دست می‌آید. اگر $a(\alpha),b(\alpha)\in F_{2^m}$ و $d(\alpha)=a(\alpha)\times b(\alpha)$ باشد ضرب به‌صورت زیر انجام می‌شوند:

$$a(\alpha)=\alpha^{m/2}A_H+A_L,\ b(\alpha)=\alpha^{m/2}B_H+B_L$$

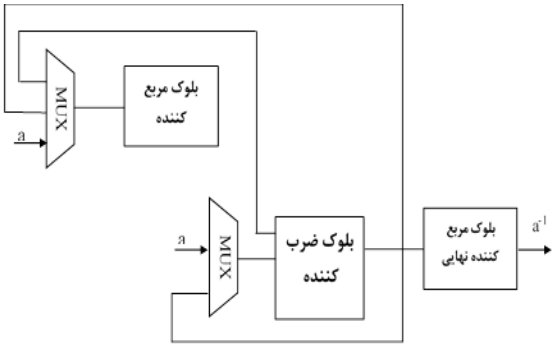
$$d(\alpha)=\alpha^{m}A_HB_H+\alpha^{m/2}(A_HB_L+A_LB_H)+A_LB_L \quad (۱۲)$$

ضرب روی $F_{2^{4m}}$ بر طبق تکنیک کاراتسوبا- اُفمن با ۹ ضرب LSD و ۲۲ جمع روی F_{2^m} انجام می‌شود.

۳-۴. معکوس ضربی

در میان عملیات میدان، محاسبه معکوس‌ضربی بیش‌ترین زمان را به خود اختصاص می‌دهد. محاسبه معکوس یک عضو $a\in F_{2^m}$ به‌صورت یافتن عضو یکتایی مثل $a^{-1}\in F_{2^m}$ تعریف می‌شود به‌طوری‌که $a.a^{-1}=1$ شود.

برای محاسبه معکوس یک عضو میدان، از الگوریتم معکوس ضربی ایتو- تسوجو (ITMIA[1]) که بر اساس قضیه کوچک‌فرما است، استفاده می‌شود. این الگوریتم از یک دنباله بازگشتی به‌همراه مفهوم زنجیره جمع[2] جهت یافتن معکوس ضربی استفاده می‌کند [۱۷]. معماری پیشنهادی در شکل (۴) شامل ضرب‌کننده LSD، یک بلوک مربع‌کننده با قابلیت انتخاب تعداد مربع‌کننده‌های داخلی و مربع‌کننده نهایی است. در ساخت بلوک مربع‌کننده سعی شده تا حد ممکن از کم‌ترین پالس ساعت برای ساخت هر جمله زنجیره جمع استفاده شود.



شکل ۴. معماری ارائه‌شده جهت محاسبه معکوس‌ضربی

معکوس‌ضربی روی میدان $F_{2^{2m}}$ برای عضو $U=u_0+u_1s\in F_{2^{2m}}$ به‌صورت $V=v_0+v_1s\in F_{2^{2m}}$ که $u_0,u_1,v_0,v_1\in F_{2^m}$ است [۱۳]. چون $UV=1$، $s^2+s+1=0$ و $t^2+s+1=0$ است، آن‌گاه:

$$\begin{cases} u_0v_0+u_1v_1=1 \\ u_0v_0+u_1v_0+u_1v_1=0 \end{cases} \quad (۱۳)$$

حل این دستگاه معادلات به‌صورت زیر است:

$$v_0=w^{-1}.(u_0+u_1),\ v_1=w^{-1}.u_1,\ w=u_0^2+(u_0+u_1).u_1 \quad (۱۴)$$

بنابراین معکوس روی $F_{2^{2m}}$ شامل ۳ ضرب، ۲ جمع، ۱ مربع‌کننده و ۱ معکوس‌ضربی روی F_{2^m} است.

۳-۵. توان‌رسانی

یک عضو از میدان $F_{2^{4m}}$ مانند U را می‌توان به‌صورت زیر به توان 2^m+1 رساند [۱۳].

$$\begin{aligned} U^{2^m+1}&=((u_0+u_1)(u_2+u_3)+u_0u_1+u_0u_3+(u_0+u_1)^2) \\ &+((u_0+u_1)(u_2+u_3)+u_1u_2+u_2u_3+(u_2+u_3)^2)s \\ &+(u_0u_3+u_1u_2)t+(u_2u_3+(u_2+u_3)^2)st \end{aligned} \quad (۱۵)$$

توان‌رسانی شامل ۵ ضرب، ۲ مربع و ۱۴ جمع روی F_{2^m} است.

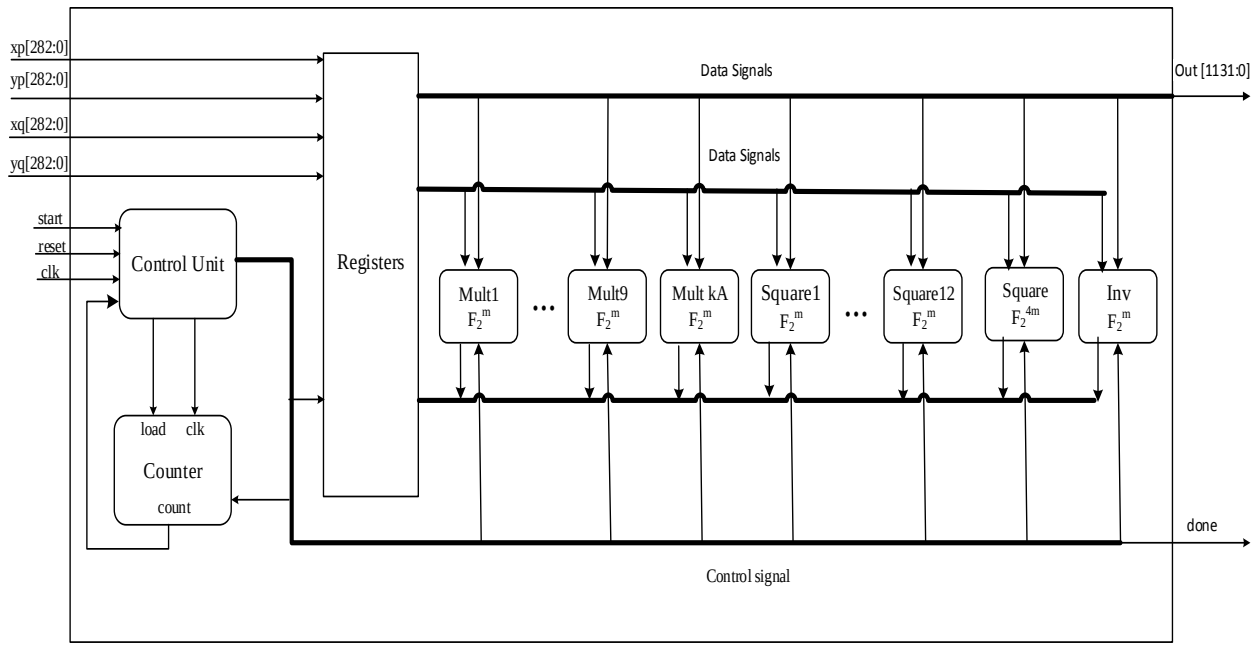
۴. معماری پیشنهادی سخت‌افزاری زوج‌سازی η_T

شکل (۵) یک معماری پیشنهادی را برای محاسبه زوج‌سازی η_T نشان می‌دهد. این معماری شامل واحد کنترل، ثبات‌ها و واحدهای محاسباتی روی میدان است. واحد محاسبات شامل بخش‌های محاسباتی مورد نیاز در الگوریتم ۱ و ۲ است. عملیات زوج‌سازی به‌صورت کارآمدی در بین این واحدهای محاسباتی زمان‌بندی شده است. با تغییر تعداد واحدهای محاسباتی موازی

[1] Itoh-Tsujii Multiplicative Inversion Algorithm
[2] Addition Chain

به معماری با پیاده‌سازی با گذردهی بالا، فشرده (کم‌حجم) یا تعادل بین این دو دست پیدا می‌کنیم. بنابراین با هدف ایجاد تعادل تعداد بلوک‌های موازی محاسباتی انتخاب شده است. این بلوک‌های محاسباتی شامل ضرب‌کننده LSD و کاراتسوبا- افمن در F_{2^m}، مربع‌کننده در میدان F_{2^m}، مربع‌کننده در میدان $F_{2^{4m}}$ و معکوس‌گر در میدان F_{2^m} است. واحد کنترل با استفاده از ماشین حالت متناهی (FSM[1]) و یک شمارنده برای کنترل زمان‌بندی عملیات و انتقال حالت‌ها پیاده‌سازی شده است. ضروری است زمان‌بندی عملیات تا حد ممکن کارآمد باشد تا مطمئن شویم کلاک‌های سامانه به هدر نمی‌روند. همچنین به‌طور خاص زمان‌بندی عملیات درون حلقه‌ها به‌دلیل تکرار زیاد بسیار ضروری است. ثبات‌ها که نقش ذخیره‌سازی مقادیر میانی را به‌عهده دارند، با استفاده از اسلایس‌های موجود در تراشه FPGA پیاده‌سازی شده است. ورودی‌های این معماری سیگنال‌های کنترلی، سیگنال‌های داده و سیگنال‌های سیستمی است. سیگنال‌های کنترلی شامل start و done، سیگنال‌های داده شامل دو نقطه خم با مختصات (xp,yp) و (xq,yq) و خروجی out است. سیگنال‌های سامانه‌ای شامل reset و clk است.

برای حالتی‌که دو الگوریتم ۱ و ۲ به‌صورت مجزا پیاده‌سازی و تمام بلوک‌های داخلی به‌صورت موازی فراخوانی شوند، در کل به ۳۶ ضرب‌کننده، ۲۳ مربع‌کننده و یک معکوس‌ضربی روی میدان F_{2^m} نیاز است. در این‌صورت سطح اشغالی بالا است و همچنین منابع استفاده‌شده در بلوک‌های داخلی در طول اجرای الگوریتم فقط یک‌بار به‌کار گرفته شده و بقیه زمان‌ها بدون استفاده هستند. با ادغام دو الگوریتم و بلوک‌های داخلی (به‌جز بلوک ضرب F.G به دلیل قرارداشتن در حلقه الگوریتم ۱و زمان-بربودن) و استفاده از تکنیک به اشتراک‌گذاری منابع، به ۱۵ ضرب‌کننده موازی (۹ تا در الگوریتم اصلی و ۶ تا در بلوک ضرب F.G)، ۱۲ مربع‌کننده، یک معکوس‌ضربی روی میدان F_{2^m} و یک مربع‌کننده روی میدان $F_{2^{4m}}$ نیاز است. جهت پیاده‌سازی ضرب در این معماری، از دو نوع ضرب‌کننده LSD و کاراتسوبا استفاده می‌شود. ضرب‌کننده LSD در خارج از حلقه و ضرب‌کننده موازی کاراتسوبا در حلقه الگوریتم ۱، برای کاهش زمان محاسبه استفاده می‌شود. به‌دلیل زیادبودن سطح اشغالی ضرب‌کننده کاراتسوبا، از تعداد ضرب‌کننده‌های LSD که هم‌زمان فراخوانی شده‌اند، کاسته می‌شود.



شکل ۵. معماری پیشنهادی محاسبه زوج‌سازی η_T

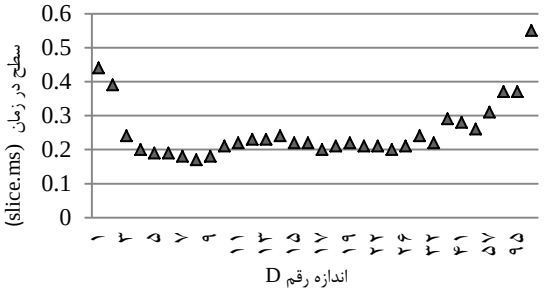
[1]Finite State Machine

۵. نتایج پیاده‌سازی و مقایسه‌ها

در این پیاده‌سازی m برابر ۲۸۳ و چندجمله‌ای کاهش به‌صورت $f(x)=x^{283}+x^{12}+x^{7}+x^{5}+1$ انتخاب شده است. همچنین، FPGA شرکت Xilinx از خانواده Virtex-4 مدل XC4VLX160 انتخاب شده که دارای ۶۷۵۸۴ اسلایس و ۲۸۸ قطعه رَم بلوکی ۱۸ کیلوبیتی است. علت انتخاب این خانواده از FPGAها، داشتن سطح کافی و قیمت پایین‌تر برای پیاده‌سازی در این کار است. نتایج محاسبات روی FPGA با استفاده از نرم‌افزار ISE Foundation 9.2i با زبان VHDL برنامه‌نویسی و سپس نتایج سنتز آن ارائه شده است.

۵-۱. نتایج پیاده‌سازی واحدهای محاسباتی

الف) نتایج واحدهای ضرب‌کننده: باتوجه به توضیحات بخش ۳-۳، دو نوع ضرب‌کننده به‌صورت ترکیبی در پیاده‌سازی استفاده شده است. نوع اول ضرب‌کننده LSD است که با انتخاب مناسب دو پارامتر اندازه رقم و تعداد انباشته‌گر می‌توان نتایج بهینه‌تری کسب کرد. شکل (۶) نتایج پیاده‌سازی ضرب‌کننده LSD با اندازه رقم‌های مختلف و برحسب معیار سطح در زمان را نشان می‌دهد. اندازه رقم هشت نسبت به بقیه اندازه رقم‌ها بهینه است.



شکل ۶. مقایسه نتایج ضرب‌کننده LSD برای Dهای مختلف

همچنین، نتایج برای تعداد ۱ تا ۴ انباشته‌گر مقایسه شده و باتوجه به نتایج، تعداد۲ انباشته‌گر دارای کم‌ترین مقدار حاصل‌ضرب سطح در توان داشته است. جدول (۲) مقایسه نتایج پیاده‌سازی ضرب کننده LSD با اندازه رقم ۸ و دو انباشته‌گر با کارهای دیگر را نشان می‌دهد.

برای پیاده‌سازی ضرب روی میدان $F_{2^{4\times283}}$ از نُه ضرب‌کننده LSD با اندازه رقم هشت به‌صورت موازی استفاده شده است. مقایسه نتایج پیاده‌سازی این ضرب در جدول (۳) نشان داده شده است. به‌دلیل انتخاب پایه متفاوت جهت نمایش اعضای میدان فوق و کارآمدبودن ضرب‌کننده LSD و همچنین کاهش دو عملیات جمع در محاسبات نسبت به دیگر کارها نتیجه بهتری به‌دست آمده است.

جدول ۲. مقایسه نتایج ضرب‌کننده LSD در میدان $F_{2^{283}}$ با دیگران

مرجع	اندازه رقم	تعداد اسلایس	حداکثر فرکانس کاری(MHz)	زمان محاسبه (µs)	سطح×زمان (Slice.ms)
[۸]	۴	۳۰۴۶	۱۰۰	۱/۲۶	۳/۸۴
[۸]	۸	۴۴۸۳	۱۰۰	۰/۷۲	۳/۲۳
[۸]	۱۶	۶۳۰۶	۱۱۲	۰/۳۶	۲/۲۷
[۱۰]	۸	۲۱۳۴	۶۳	۰/۶۰	۱/۲۸
[۱۱]	۳۲	۳۵۶۲	۲۲۹	۰/۰۴۳	۰/۱۵
این تحقیق	۸	۱۴۵۲	۳۵۸/۰۸	۰/۱۱	۰/۱۵

ضرب‌کننده کاراتسوبا- اُفمن شامل دو بخش تابع کاهش و بدنه اصلی است. چون ضرب‌کننده کاراتسوبا یک ضرب‌کننده بازگشتی است، در هر تکرار ضرب‌شونده‌ها به دو نیمه تقسیم می‌شوند و این تقسیم‌شدن تا زمانی ادامه می‌یابد که حاصل‌ضرب سطح در زمان کم‌ترین شود، آن‌گاه بیت‌های باقی‌مانده به‌صورت ضرب موازی کلاسیک محاسبه می‌شود. در هر تکرار ۲۵٪ از سطح اشغالی توسط ضرب‌کننده کم می‌شود. البته این کاهش از لحاظ عملی با محدودیت‌هایی مواجه است و پس از تعداد تکرار مشخصی سطح اشغالی دوباره افزایش می‌یابد. مسیر بحرانی با افزایش تکرار به‌دلیل افزایش تعداد ضرب‌های موازی زیاد می‌شود. در اینجا با ۵ تکرار نتیجه بهینه حاصل شده است. جدول (۴) نتایج این پیاده‌سازی را نشان می‌دهد.

جدول ۳. مقایسه نتایج پیاده‌سازی ضرب‌کننده در میدان $F_{2^{4\times283}}$

مرجع	تعداد اسلایس	حداکثر فرکانس کاری(MHz)	زمان محاسبه (µs)	سطح×زمان (Slice.ms)
[۱۰]	۱۴۸۸۰	۱۳۵	۰/۶۴	۹/۵
[۱۱]	۲۵۹۵۵	۲۵۰	۰/۲۵	۶/۴۸
این تحقیق	۱۴۱۸۲	۳۱۲	۰/۱۱	۱/۵۶

جدول ۴. مقایسه نتایج ضرب‌کننده کاراتسوبا با تعداد تکرارهای مختلف در میدان $F_{2^{283}}$

تعداد تکرار	تعداد ضرب موازی	تعداد اسلایس	زمان محاسبه (ns)	سطح×زمان (Slice.ms)
۱	۳	۲۷۳۶۵	۱۰/۵۰	۰/۲۸
۲	۹	۲۲۳۰۶	۱۱/۰۶	۰/۲۴
۳	۲۷	۱۷۰۰۰	۱۲/۰۷	۰/۲۰
۴	۸۱	۱۴۰۰۰	۱۲/۱۰	۰/۱۷
۵	۲۴۳	۱۱۶۵۱	۱۲/۲۹	۰/۱۴
۶	۷۲۹	۱۲۱۶۴	۱۳/۴۵	۰/۱۶
۷	۲۱۸۷	۱۳۶۱۵	۱۳/۶۷	۰/۱۸

ب) نتیجه واحد محاسبه معکوس ضربی: پیاده‌سازی معکوس‌ضربی روی میدان $F_{2^{283}}$ بر اساس معماری ارائه‌شده در بخش ۳-۴ و شامل یک بدنه اصلی شامل، ضرب‌کننده LSD با اندازه رقم هشت، یک بلوک مربع‌کننده با قابلیت انتخاب تعداد مربع‌کننده‌های داخلی و مربع‌کننده نهایی است. مقایسه نتایج پیاده‌سازی معکوس‌ضربی با دیگران در جدول (۵) آورده شده است. به‌دلیل کارآمدبودن ضرب‌کننده LSD با اندازه رقم هشت در این کار، و همچنین ساخت بلوک مربع‌کننده به‌طوری‌که تا حد ممکن از کم‌ترین پالس ساعت برای ساخت هر جمله زنجیره جمع استفاده شود، نتیجه این کار بهبود یافته است.

جدول ۵. مقایسه نتایج معکوس‌کننده در میدان $F_{2^{283}}$

مرجع	تعداد اسلایس	حداکثر فرکانس کاری(MHz)	زمان محاسبه (µs)	سطح×زمان (Slice.ms)
[۱۱]	۷۳۵۴	۱۶۰	۵/۷۶	۴۲
این تحقیق	۴۴۸۲	۱۵۳	۱/۹۶	۸/۷۸

ج) نتایج پیاده‌سازی واحد مربع‌کننده و توان‌رسانی: مربع‌کننده روی میدان $F_{2^{283}}$ بر اساس بخش ۳-۲ پیاده‌سازی شده و ساده ترین عملیات در بین واحدهای محاسباتی است که در یک پالس ساعت انجام می شود. همچنین، مربع‌کننده روی میدان $F_{2^{4\times 283}}$ با ۴ مربع‌کننده به‌صورت موازی پیاده‌سازی شده است. نتایج در جدول (۶) آمده است که به‌دلیل نبودن نتایج کارهای دیگر در این واحد، مقایسه انجام نشده است.

جدول ۶. نتایج پیاده‌سازی واحدهای مربع‌کننده

میدان	تعداد اسلایس	حداکثر فرکانس (MHz)	زمان محاسبه (ns)	سطح×زمان (Slice.µs)
$F_{2^{283}}$	۱۶۴	۱۸۲/۱۵	۵/۴۹	۰/۹
$F_{2^{283}}$	۹۶۶	۱۶۱/۵۵	۶/۱۹	۵/۹۸

د) نتایج پیاده‌سازی واحد توان‌رسانی: پیاده‌سازی این واحد بر اساس روش ارائه‌شده در بخش ۳-۵ است. توان‌رسانی از ۵ ضرب موازی LSD، ۲ واحد مربع و ۱۴ جمع صورت گرفته است. نتیجه در جدول (۷) آمده است که به‌دلیل نبودن نتایج کارهای دیگر در این واحد، مقایسه انجام نشده است.

جدول ۷. نتایج پیاده‌سازی واحد توان‌رسانی در میدان $F_{2^{4\times 283}}$

واحد محاسباتی	میدان	تعداد اسلایس	حداکثر فرکانس (MHz)	زمان محاسبه (ns)	سطح×زمان (Slice.µs)
توان‌رسانی	$F_{2^{4\times 283}}$	۷۶۵۶	۳۷۲/۷۹	۱۰۴	۷۹۸

۵-۲. نتایج پیاده‌سازی سخت‌افزاری زوج‌سازی η_T

جدول (۸) مقایسه نتایج نهایی پیاده‌سازی زوج‌سازی η_T در این مقاله را با کارهای دیگر برحسب سه معیار سطح، زمان محاسبه و سطح در زمان را نشان می‌دهد. مقایسه نتایج این پیاده‌سازی با دیگر کارها، ۱۰درصد بهبود برای معیار سطح در زمان و ۳۸ درصد بهبود در زمان محاسبه نسبت به بهترین نتیجه به‌دست‌آمده تاکنون [۱۱] نشان می‌دهد. بهبود نتیجه این کار در مقایسه با دیگران وابسته به دلایل مختلفی است:

- ادغام دو بخش محاسبه زوج‌سازی η_T به‌منظور به‌اشتراک‌گذاری منابع.
- استفاده از میدان پایه مناسب با انتخاب میدان F_{2^m} که نسبت به میدان F_{3^m} برتری دارد.
- استفاده از ضرب‌کننده LSD با دو انباشته‌گر جهت افزایش فرکانس کار.
- استفاده از ضرب‌کننده LSD و کاراتسوبا به‌صورت هم‌زمان جهت کاهش زمان محاسبه.
- موازی‌سازی در محاسبات جهت افزایش سرعت.
- به اشتراک‌گذاری منابع: استفاده از ضرب‌کننده‌ها و مربع‌کننده‌ها به‌صورت اشتراکی برای محاسبات مختلف.

جدول ۸. مقایسه نتایج پیاده‌سازی زوج‌سازی η_T با دیگر کارها

مرجع	خم بیضوی	FPGA	تعداد اسلایس	فرکانس (MHz)	زمان محاسبه (µs)	سطح×زمان (Slice.s)
[۱۰]	$E(F_{2^{251}})$	xc2v6000	۲۴۶۵۵	۴۷	۲۸۱۰	۶۹
[۱۱]	$E(F_{2^{283}})$	xc4vfx140	۵۵۸۴۴	۱۶۰	۵۹۰	۳۳
[۱۸]	$E(F_{2^{283}})$	xc2v6000	۱۸۶۹۳	۱۰۸	۳۰۰	۶
[۸]	$E(F_{2^{283}})$	xc2vp100	۳۶۴۸۱	۱۰۰	۴۶	۱/۶۸
این تحقیق	$E(F_{2^{283}})$	xc4vlx160	۵۴۱۰۸	۱۳۴	۲۸	۱/۵۴
مرجع	خم بیضوی	ASIC Tech.	سطح KGE	فرکانس (MHz)	زمان محاسبه (µs)	سطح×زمان (GE.s)
[۱۹]	$E(F_{2^{283}})$	130 nm CMOS	۹۷	۳۳۸	۱۵۸۰۰	۱۵۳۲
[۲۰]	$E(F_{p^{256}})$	130 nm CMOS	۱۸۳	۲۰۴	۲۹۰۰	۵۳۰
[۲۱]	$E(F_{2^{1223}})$	65 nm CMOS	۵۴۸	۵۰۰	۵۵	۳۰
[۲۲]	$E(F_{3^{97}})$	180 nm CMOS	۱۹۳	۲۰۰	۴۶	۹

۵-۳. پیاده‌سازی نرم‌افزاری

چندین بسته نرم‌افزاری برای پیاده‌سازی محاسبات جبری و نظریه اعداد وجود دارند. SAGE [۲۳] یک بسته نرم‌افزاری رایگانِ متن‌بازِ ریاضی، تحت مجوز گنو است که زمینه‌های جبر، هندسه، نظریه اعداد، رمزنگاری و غیره را پشتیبانی می‌کند. در اینجا برای تولید نقاط روی خم بیضوی برای ورودی زوج‌سازی و

همچنین، اطمینان از صحت جواب نهایی و تطبیق آن با جواب پیاده‌سازی سخت‌افزاری زوج‌سازی η_T، از نرم‌افزار SAGE نگارش ۴/۲ استفاده شده است. در محیط برنامه‌نویسی این نرم‌افزار میدان متناهی $F_{2^{283}}$ با استفاده از چندجمله‌ای تحویل‌ناپذیر $f(x)=x^{283}+x^{12}+x^{7}+x^{5}+1$ روی F_2 قابل تعریف است. پارامترهای خم بیضوی ابرمنفرد روی این میدان به نرم‌افزار داده می‌شود و سپس با استفاده از تابع تصادفی تولید نقطه روی خم، دو نقطه برای ورودی زوج‌سازی η_T تولید می‌شود. الگوریتم زوج‌سازی η_T بر اساس الگوریتم ۱ و ۲ برنامه‌نویسی شده است. جدول (۹) برخی نتایج پیاده‌سازی نرم‌افزاری را نشان می‌دهد. اگرچه هدف این پیاده‌سازی بستر نرم‌افزار نبوده، ولی ذکر نتایج فقط برای نشان‌دادن تفاوت بسیار زیاد زمانی در پیاده‌سازی نرم‌افزاری در مقایسه با پیاده‌سازی سخت‌افزاری این تحقیق است.

جدول ۹. نتایج پیاده‌سازی نرم‌افزاری زوج‌سازی η_T

مرجع	خم بیضوی	پردازنده	زمان محاسبه (µs)
[۶]	$E(F_{2^{307}})$	Pentium IV 3 GHz	۳۵۰۰
[۶]	$E(F_{3^{127}})$	Pentium IV 3 GHz	۵۳۶۰
[۷]	$E(F_{3^{97}})$	Intel Core2	۱۵۰
[۷]	$E(F_{3^{193}})$	Intel Core2	۹۸۰

۶. نتیجه‌گیری

در این مقاله زوج‌سازی تیت بهبودیافته برای خم‌های ابرمنفرد روی میدان دودویی که η_T نامیده می‌شود، روی FPGA پیاده‌سازی شده است. این کار شامل طراحی و پیاده‌سازی کارآمد محاسبات میدان و به‌خصوص ضرب‌کننده‌ها است. برای تحقّق این موضوع استفاده هم‌زمان از ضرب‌کننده‌های LSD و کاراتسوبا پیشنهاد شده است. ضرب‌کننده LSD امکان برقراری موازنه بین سطح و زمان را با تغییر اندازه رقم ایجاد کرده و همچنین استفاده از ضرب‌کننده کاراتسوبا درون حلقه‌های تکرار الگوریتم زوج‌سازی η_T، باعث کاهش زیادی در زمان محاسبه شده است. مقایسه نتیجه پیاده‌سازی زوج‌سازی برای m=283 با دیگران، بهبود ۳۸ درصد در زمان محاسبه و بهبود ۱۰ درصد برای معیار سطح در زمان نسبت به بهترین نتیجه به‌دست‌آمده تاکنون [۸] را نشان می‌دهد. چندین عامل در افزایش عملکرد این پیاده‌سازی نسبت به سایر پیاده‌سازی‌های مشابه مؤثر بوده که شامل: معماری مناسب سطح بالا، کارآمدی محاسبات میدان پایه، استفاده از میدان پایه مناسب، به اشتراک‌گذاری منابع، استفاده از ضرب‌کننده LSD با دو انباشته‌گر، استفاده از ضرب‌کننده LSD و کاراتسوبا به‌صورت هم‌زمان و غیره است. همچنین، پیاده‌سازی نرم‌افزاری توسط نرم‌افزار ریاضی SAGE برای تولید نقاط روی خم و آزمایش صحت جواب‌های به‌دست‌آمده به‌کار گرفته شده است.

۷. مراجع

[1] Menezes, A. J.; Okamoto, T.; Vanstone, S. "Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field"; IEEE Transactions on Information Theory 1993, 39, 1639–1646.

[2] Frey, G.; Rück, H.-G. "A Remark Concerning-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves"; Mathematics of Computation 1994, 62, 865–874.

[3] Miller, V. S. "Short Programs for Functions on Curves"; (Unpublished manuscript), http://crypto.stanford.edu. 1986.

[4] Barreto, P. S.; Kim, H. Y.; Lynn, B.; Scott, M. "Efficient Algorithms for Pairing-Based Cryptosystems"; Advances in Cryptology - CRYPTO 2002, 2442, 354–369.

[5] Duursma, I.; Lee, H.-S. "Tate Pairing Implementation for Hyperelliptic Curves $y^2 = x^p - x + d$"; Advances in Cryptology - ASIACRYPT 2003, 2894, 111–123.

[6] Barreto, P. S.; Galbraith, S. D.; Ó'hÉigeartaigh, C.; Scott, M. "Efficient Pairing Computation on Supersingular Abelian Varieties"; Designs, Codes and Cryptography 2007, 42, 239–271.

[7] Beuchat, J.-L.; López-Trejo, E.; Martínez-Ramos, L.; Mitsunari, S.; Rodríguez-Henríquez, F. "Multi-Core Implementation of the Tate Pairing Over Supersingular Elliptic Curves"; Cryptology and Network Security 2009, 413–432.

[8] Shu, C. "Hardware Architectures of Elliptic Curve Based Cryptosystems over Binary Fields"; Ph.D. Thesis, George Mason University, USA, 2007.

[9] Kwon, S. "Efficient Tate Pairing Computation for Elliptic Curves over Binary Fields"; Lecture Notes in Computer Science, Information Security and Privacy 2005, 134–145.

[10] Keller, M.; Kerins, T.; Crowe, F.; Marnane, W. "FPGA Implementation of a GF (2 m) Tate Pairing Architecture"; Reconfigurable Computing: Architectures and Applications 2006, 358–369.

[11] Li, H.; Huang, J.; Sweany, P.; Huang, D. "FPGA Implementations of Elliptic Curve Cryptography and Tate Pairing over a Binary Field"; Journal of Systems Architecture 2008, 54, 1077–1088.

[12] Ronan, R.; Eigeartaigh, C. O.; Murphy, C.; Scott, M.; Kerins, T. "FPGA Acceleration of the Tate Pairing in Characteristic 2"; Proc. International Conference on Field Programmable Technology, 2006, 213–220.

[13] Beuchat, J.-L.; Brisebarre, N.; Detrey, J.; Okamoto, E.; Rodríguez-Henríquez, F. "A Comparison between Hardware Accelerators for the Modified Tate Pairing over $\mathbb{F}_2^m$ and $\mathbb{F}_3^m$"; Pairing-Based Cryptography–Pairing 2008, 5209, 297–315.

[14] Estibals, N. "Compact Hardware for Computing the Tate Pairing over 128-Bit-Security Supersingular Curves"; Pairing-Based Cryptography-Pairing 2010, 6487, 397–416.

[15] Song, L.; Parhi, K. K. "Low-Energy Digit-Serial/Parallel Finite Field Multipliers"; Journal of VLSI Signal Processing

Systems for Signal, Image and Video Technology 1998, 19, 149–166.

[16] Rodriguez-Henriquez, F.; Saqib, N. A.; Perez, A. D.; Koc, C. K. "Cryptographic Algorithms on Reconfigurable Hardware"; Springer Science & Business Media, 2007.

[17] Rodriguez-Henriquez, F.; Morales-Luna, G.; Saqib, N. A.; Cruz-Cortés, N. "Parallel Itoh–Tsujii Multiplicative Inversion Algorithm for a Special Class of Trinomials"; Designs, Codes and Cryptography 2007, 45, 19–37.

[18] Pan, W.; Marnane, W. "A Reconfigurable Implementation of the Tate Pairing Computation over GF(2^m)"; Reconfigurable Computing: Architectures, Tools and Applications 2010, 5992, 80–91.

[19] Kammler, D.; Zhang, D.; Schwabe, P.; Scharwaechter, H.; Langenberg, M.; Auras, D.; Ascheid, G.; Mathar, R. "Designing an ASIP for Cryptographic Pairings Over Barreto-Naehrig Curves"; Cryptographic Hardware and Embedded Systems - CHES 2009, 5747, 254–271.

[20] Fan, J.; Vercauteren, F.; Verbauwhede, I. "Faster F_p-Arithmetic for Cryptographic Pairings on Barreto-Naehrig Curves"; Cryptographic Hardware and Embedded Systems - CHES 2009, 5747, 240–253.

[21] Adikari, J., Hasan, M. A.; Negre, C. "Towards Faster and Greener Cryptoprocessor for Eta Pairing on Supersingular Elliptic Curve over $\mathbb{F}_2{}^{1223}$"; Selected Areas in Cryptography 2013, 7707, 166-183

[22] Beuchat, J. -L.; Doi, H.; Fujita, K.; Inomata, A.; Ith, P.; Kanaoka, A.; Katouno, M.; Mambo, M.; Okamoto, E.; Okamoto, T. "FPGA and ASIC Implementations of the η_T Pairing in Characteristic Three"; Computers & Electrical Engineering 2010, 36, 73–87.

[23] "Software for Algebra and Geometry Experimentation (SAGE) Version 4.2"; http://www.sagemath.org