هفتمین کنفرانس بین المللی انجمن رمز ایران
۲۴ و ۲۵ شهریورماه ۱۳۸۹، دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشگاه صنعتی خواجه نصیرالدین طوسی
انجمن رمزایران

پیاده‌سازی سیستم‌های رمزنگاری بر اساس زوج‌سازی η_T با استفاده از کدهای قابل سنتز VHDL

محسن جهانبانی[۱]، محمود احمدیان[۲]، محمود گردشی[۳]

تهران، دانشگاه جامع امام حسین (ع)، دانشکده مهندسی برق[۱]
mo.jahanbani2009@gmail.com

تهران، دانشگاه خواجه نصیرالدین طوسی، گروه مهندسی برق[۲]
Mahmoud@eetd.kntu.ac.ir

تهران، دانشگاه جامع امام حسین (ع)، گروه ریاضی و رمز[۳]
gardeshi2000@yahoo.com

چکیده

در این مقاله یک معماری جدید برای محاسبات نوع خاصی از زوج‌سازی تیت، که η_T نامیده می‌شود، روی میدان $F_{2^{283}}$ پیشنهاد شده است. این معماری از ادغام دو بخش الگوریتم زوج‌سازی و توان‌رسانی نهایی با استفاده از تکنیک به اشتراک‌گذاری منابع حاصل شده است. پایه محاسبات این معماری بر اساس واحدهای محاسباتی میدان متناهی شامل جمع، ضرب، مربع، معکوس‌ضربی و توان‌رسانی است. طراحی و سنتز این پیاده‌سازی روی FPGAهای Xilinx انجام شده است. مقایسه نتایج این پیاده‌سازی با دیگران، بهبود ۳۸٪ در زمان محاسبه و بهبود ۱۰٪ برای معیار سطح در زمان نسبت به بهترین نتیجه بدست آمده را نشان می‌دهد. همچنین پیاده‌سازی نرم‌افزاری توسط نرم‌افزار ریاضی SAGE به منظور آزمودن صحت جواب‌های به دست آمده و همچنین تولید نقاط روی خم به کار گرفته شده است.

واژه های کلیدی

زوج‌سازی تیت، زوج‌سازی η_T، محاسبات میدان متناهی دودویی، خم بیضوی، معماری سخت‌افزاری، FPGA.

۱- مقدمه

در سال‌های اخیر سیستم‌های رمزنگاری بر اساس زوج‌سازی[۱] روی خم‌های بیضوی به عنوان جایگزینی برای سیستم‌های رمزنگاری کلید عمومی سنتی معرفی شده‌اند. همچنین 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 تیت و ویل در رمزنگاری در آغاز برای حمله به مسئله لگاریتم گسسته برای دسته خاصی از خم‌های بیضوی روی میدان متناهی، که به ترتیب به عنوان حمله MOV[2] [۱] و FR[3] [۲] شناخته می‌شود، استفاده شد. این حمله‌ها مسئله لگاریتم گسسته روی این نوع خم را به مسئله لگاریتم گسسته روی میدان متناهی کاهش داد. پس از آن از خواص زوج‌سازی‌ها برای ساخت پروتکل‌های رمزنگاری استفاده شد.

[1] Pairing

[2] Menezes, Okamoto and Vanstone

[3] Frey and Ruck

اولین‌بار میلر [۳] در یک دست‌نوشته منتشر نشده در سال ۱۹۸۶، الگوریتم محاسبه زوج‌سازی تیت و ویل را پیشنهاد داد. اگرچه در زوج‌سازی تیت جهت بدست آوردن یک مقدار یکتا، یک توان‌رسانی نهایی روی خروجی الگوریتم میلر لازم است، اما به دلیل اینکه زوج‌سازی ویل به اجرای دو بار الگوریتم میلر نیاز دارد، زوج‌سازی تیت حداقل دو برابر سریع‌تر از زوج‌سازی ویل است [۳]. چندین بهینه‌سازی برای الگوریتم میلر برای زوج سازی تیت پیشنهاد شده است. از جمله بارتو [۴] با توسعه ایده دورسما-لی [۵] زوج‌سازی اِتا کوتاه‌شده[1] (η_T) را معرفی کرد که تعداد تکرارهای الگوریتم میلر را به نصف کاهش داد.

تحقّق عملی تمام پروتکل‌های رمزنگاری بر اساس زوج سازی، وابسته به پیاده‌سازی کارآمد آن است و بسیاری از محققان در تلاش برای بهینه‌سازی الگوریتم‌ها و همچنین پیاده‌سازی‌ها هستند. پردازنده‌های سخت‌افزاری تک‌منظوره بستر ایده‌آلی برای پیاده‌سازی زوج‌سازی است. بستر آرایه‌های دروازه‌ی قابل برنامه‌ریزی (FPGA[2]) یک انتخاب خوب با توجه به قابلیت‌های فراوان آن از جمله ایجاد سریع نمونه، بازپیکربندی، انعطاف‌پذیری و هزینه توسعه کم، است.

شو [۶] اولین پیاده‌سازی سخت‌افزاری زوج‌سازی تیت را بر اساس دو الگوریتم اصلاح‌شده بارتو [۷] و کون [۸] روی خم ابرمنفرد و میدان $F_{2^{283}}$ با سطح امنیت معادل AES-۷۲ بیت انجام داد. کلر و همکاران [۹] یک معماری دو وضعیتی را با استفاده از الگوریتم بارتو [۷] روی خم ابرمنفرد پیاده‌سازی کردند که یک پردازنده زوج‌سازی تیت و خم بیضوی بود و ضرب عددی نقطه روی خم را نیز محاسبه می‌کرد.

لی و همکاران [۱۰] الگوریتم پیشنهادی کون [۸] را روی خم ابرمنفرد و میدان $F_{2^{283}}$ با سطح امنیت معادل AES-۷۲ بیت پیاده‌سازی کردند. رونن و همکاران [۱۱] 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 [۱۲] 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.

این مقاله به پیاده‌سازی الگوریتم زوج‌سازی η_T، که توسط بارتو [۷] ارائه و توسط بیوچت [۱۲] بهینه شده است، می‌پردازد. هدف اصلی این مقاله ارائه و پیاده‌سازی یک معماری سخت‌افزاری کارآمد جهت کاهش زمان محاسبه بدون افزایش چشم‌گیر سطح اشغالی است. این پیاده‌سازی روی خم‌های بیضوی ابرمنفرد و میدان متناهی دودویی در بستر FPGA انجام می‌شود.

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 نتایج پیاده‌سازی در بخش ۴ ارائه می‌شود. بخش ۵ شامل پیاده‌سازی نرم‌افزاری جهت آزمودن صحت جواب و بخش آخر شامل نتیجه‌گیری و پیشنهاداتی برای کارهای آینده است.

۲- تعاریف مقدماتی

فرض کنیم $F_q = F_{2^m}$ میدان متناهی دودویی و گروه نقاط یک خم بیضوی E تعریف شده روی میدان F_q با $E(F_q)$ نشان داده شود. یک زیرگروه $E(F_q)$ از مرتبه اول r دارای درجه تعبیه[3] k است، به طوری‌که رابطه $r \mid q^k - 1$ برای کوچکترین k ممکن برقرار باشد. یک زیرگروه مرتبه r به عنوان گروه r- پیچش[4] 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 $F_{q^k}^*$ به صورت 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 زوج-سازی تیت اصلاح‌شده به صورت زیر می‌شود:

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

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

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

[1] Trunkless Eta

[2] Field Programmable Gate Array

[3] Embedding Degree

[4] Torsion

زوج‌سازی اِتای کوتاه شده (η_T) یک تکنیک بسیار کارآمد برای محاسبه زوج‌سازی تیت و شامل یک حلقه تکرار و توان-رسانی نهایی است.

۲-۱- محاسبات زوج‌سازی η_T

روش η_T برای محاسبه زوج‌سازی روی خم‌های بیضوی ابرمنفرد است. این خم روی میدان با مشخصه دو دارای معادله به صورت $E(F_{2^m}): y^2+y=x^3+x+b$ است که $b\in\{0,1\}$ و m فرد و دارای درجه تعبیه $k=4$ است. تعداد نقاط گویای E روی F_{2^m} به صورت $N=\#E(F_{2^m})=2^m+1+\nu 2^{(m+1)/2}$ است [۴]، به طوری‌که $\nu=(-1)^{\delta}$. اگر $m\equiv 1,7 \bmod 8$ پس $\delta=b$ و در غیر اینصورت $\delta=1-b$ است. اعضای میدان $F_{2^{4m}}$ می‌توانند بر حسب s و t نمایش داده شوند که $s^2=s+1$ و $t^2=t+s$ است. برای نمایش اعضای $F_{2^{4m}}$ به عنوان یک توسیع از F_{2^m} از پایه $\{1,s,t,st\}$ استفاده می‌شود به طوریکه:

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

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

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

محاسبات زوج‌سازی η_T در الگوریتم ۱ آورده شده است.

الگوریتم ۱: محاسبه زوج‌سازی η_T با مشخصه دو [۱۲]

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,2XOR)

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.

در محاسبه‌ی تعداد عملیات الگوریتم، A جمع، M ضرب، S مربع کردن و I معکوس‌ضربی را روی میدان دودویی F_{2^m} و XOR، یای انحصاری را نشان می‌دهد. همچنین $\bar{\delta}=1-\delta$ است. محاسبه خط ۱۹ در بخش ۲ انجام می‌شود.

۲-۲- توان‌رسانی نهایی

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

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

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:

$$\eta_T(P,Q)^M= \qquad (۷)$$

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

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

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

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

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

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 $\eta_T\left(P,Q\right)^M$ را جمع‌بندی می‌کند.

الگوریتم ۲: توان‌رسانی نهایی زوج‌سازی کاهش‌یافته η_T [۱۱]	
ورودی: $U=u_0+u_1s+u_2t+u_3st\in F_{2^{4m}}$	
خروجی: $V=U^M\in F_{2^{4m}}$.	
۱. $u_1^2\rightarrow m_1, u_0^2\rightarrow m_0$.	(2 S)
۲. $u_3^2\rightarrow m_3, u_2^2\rightarrow m_2$.	(2 S)
۳. $(m_0+m_1)+m_1s\rightarrow T_0$	(1 A)
۴. $(m_2+m_3)+m_3s\rightarrow T_1$	(1 A)
۵. $m_3+m_2s\rightarrow T_2$	(1 A)
۶. $(u_0+u_1s).(u_2+u_3s)\rightarrow T_3$	(3 M, 3 A)
۷. $T_0+T_2\rightarrow T_4,\ T_3+T_4\rightarrow D$.	(4 A)
۸. $D^{-1}\rightarrow D$.	(1I, 3M, 1S, 2A)
۹. $T_1.D\rightarrow T_5,\ T_4.D\rightarrow T_6$.	(6 M, 8 A)
۱۰. $T_5+T_6\rightarrow V_0$.	(2 A)
ادامه الگوریتم ۲	
۱۱. $W_1,V_1\rightarrow T_5$.	
۱۲. اگر $\nu=-1$ آنگاه:	
۱۳. $V_0\rightarrow W_0$	
۱۴. در غیر اینصورت:	
۱۵. $T_6\rightarrow W_0$.	
۱۶. $W_0+W_1t\rightarrow W,\ V_0+V_1t\rightarrow V$	(5 M, 2 S, 9 A)
۱۷. $V^{2^{m}+1}\rightarrow V$.	(4 S, 4 A)
۱۸. برای $i\rightarrow 1$ تا $(m+1)/2$ انجام بده:	
۱۹. $W^2\rightarrow W$.	(9 M, 20 A)
۲۰. V . W را بازگردان.	

۳- پیاده‌سازی واحدهای محاسباتی میدان دودویی

محاسبات میدان‌های متناهی به دلیل کاربرد گسترده در الگوریتم‌های رمزنگاری بسیار مورد توجه قرار گرفته است. پیاده‌سازی میدان‌های با مشخصه دو بدلیل داشتن محاسبات بدون رقم نقلی، نسبت به میدان با مشخصه بزرگتر آسان‌تر است [۶]. این موضوع نه تنها باعث سادگی معماری شده، بلکه سطح اشغالی سخت‌افزاری مورد نیاز را نیز کاهش می‌دهد. همچنین جمع در این میدان به صورت 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 [۶]. میدان بوسیله چندجمله‌ای $f(x)=x^m+\sum_{i=0}^{m-1}g_ix^i$ روی F_{2^m} تولید می‌شود. فرض کنیم α ریشه $f(x)$ است. پس نتایج محاسبات باید به پیمانه $f(\alpha)$ کاهش یابد. اعضای میدان دودویی F_{2^m} به صورت یک چندجمله‌ای هستند که ضرائب چندجمله‌ای صفر یا یک است. بنابراین اعضای میدان F_{2^m} به صورت یک رشته m-بیتی و اعضای میدان $F_{2^{4m}}$ به صورت یک رشته $4m$-بیتی نشان داده می‌شود که این بیت‌ها معادل ضرایب چندجمله‌ای هستند.

۳-۱-مربع‌کننده

مربع‌کننده روی میدان F_{2^m} برای عضو $A\in F_{2^m}$ به صورت بسط A به اندازه دو برابر طول بیت آن با قرار دادن بیت‌های صفر بین بیت‌های اصلی A، انجام می‌شود و سپس حاصل کاهش می‌یابد. معادله (۱۰) این مراحل را به خوبی نشان می‌دهد [۱۳]:

$$C=A^2 \bmod f(\alpha) \qquad (۱۰)$$

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

مربع‌کننده برای یک عضو از میدان $F_{2^{4m}}$ مانند

$U = u_0 + u_1 s + u_2 t + u_3 st \in F_{2^{4m}}$، به صورت $V = u_0^2 + u_1^2 s^2 + u_2^2 t^2 + u_3^2 s^2 t^2$ بدست می‌آید [۱۲].

۳-۲ ضرب‌کننده

در میان عملیات‌های 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 به وسیله پردازش چندین بیت در هر پالس انجام می‌شود. تعداد بیت‌هایی که به صورت موازی پردازش می شوند، اندازه رقم (D) گویند. اگر بیت‌ها از کم-ارزش‌ترین 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 [۱۳].

انتخاب اندازه رقم خیلی کوچک به دلیل طولانی شدن زمان محاسبه مناسب نیست. همچنین برای اندازه رقم‌های خیلی بزرگ، هر چند تعداد پالس ساعت کمتری برای ضرب لازم است، اما ضرب‌کننده سطح خیلی بالایی را اشغال کرده و همچنین به علت طولانی شدن مسیر بحرانی فرکانس کار کاهش می‌یابد. انتخاب بقیه اندازه رقم‌ها با توجه به اهمیت معیار زمان محاسبه یا سطح اشغالی یا هر دو صورت می‌پذیرد. این ضرب به صورت زیر محاسبه می‌شود [۱۴]:

$$C \equiv A.B \mod f(\alpha) \equiv \Big[B_0 A + B_1 \left(A\alpha^D \bmod f(\alpha) \right) + B_2 \left(A\alpha^D \alpha^D \bmod f(\alpha) \right) + \ldots + B_{d-1} \left(A\alpha^{D(d-2)} \alpha^D \bmod f(\alpha) \right) \Big] \bmod f(\alpha) \quad (11)$$

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 [۱۳] 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 و از لحاظ عملی بهترین ضرب کننده موازی است. در این ضرب‌کننده در هر تکرار ضرب شونده‌ها به دو نیمه تقسیم می‌شوند و این تقسیم شدن تا زمانی ادامه می‌یابد که حاصل‌ضرب سطح در زمان کمترین شود، آنگاه بیت‌های باقیمانده به صورت ضرب موازی محاسبه می‌شود. تعداد تکرار بهینه با استفاده از آزمون و خطا بدست می‌آید.

اگر $a(\alpha), b(\alpha) \in F_{2^m}$ و $d(\alpha) = a(\alpha) \times b(\alpha)$ باشد ضرب به صورت زیر انجام می‌شود:

$$a(\alpha) = \alpha^{m/2} A_H + A_L,\ b(\alpha) = \alpha^{m/2} B_H + B_L$$
$$d(\alpha) = \alpha^m A_H B_H + \alpha^{m/2} (A_H B_L + A_L B_H) + A_L B_L \quad (12)$$

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

۳-۳- معکوس ضربی

در میان عملیات‌های میدان، محاسبه معکوس‌ضربی زمان‌برترین آن‌ها است. محاسبه معکوس یک عضو $a \in F_{2^m}$ به صورت یافتن عضو یکتایی مثل $a^{-1} \in F_{2^m}$ تعریف می‌شود، به طوری که $a.a^{-1} = 1$ باشد.

برای محاسبه معکوس یک عضو میدان، از الگوریتم معکوس‌ضربی ایتو-سیوجو (ITMIA[1]) که بر اساس قضیه کوچک فرما است، استفاده می‌شود. این الگوریتم از یک دنباله بازگشتی به همراه مفهوم زنجیره جمع[2] جهت یافتن معکوس-ضربی استفاده می‌کند [۱۵]. معماری پیشنهادی شامل ضرب-کننده 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.

معکوس‌ضربی روی میدان $F_{2^{2m}}$ برای عضو $V = v_0 + v_1 s \in F_{2^{2m}}$ به صورت $U = u_0 + u_1 s \in F_{2^{2m}}$، که $u_0, u_1, v_0, v_1 \in F_{2^m}$ است [۱۲]. چون $UV = 1$، $s^2 + s + 1 = 0$ و $t^2 + s + 1 = 0$ است، آنگاه:

$$\begin{cases} u_0 v_0 + u_1 v_1 = 1 \\ u_0 v_0 + u_1 v_0 + u_1 v_1 = 0 \end{cases} \quad (13)$$

حل این دستگاه معادلات به صورت زیر است:

$$v_0 = w^{-1}.(u_0 + u_1),\ v_1 = w^{-1}.u_1,\ w = u_0^2 + (u_0 + u_1).u_1 \quad (14)$$

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

[1] Itoh-Tsujii Multiplicative Inversion Algorithm

[2] Addition Chain

۳-۴- توان‌رسانی

یک عضو از میدان $F_{2^{4m}}$ مانند U را می‌توان به صورت زیر به توان 2^m+1 رساند [۱۲]:

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

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

۳-۵- نتایج پیاده‌سازی واحدهای محاسباتی

در این پیاده‌سازی m برابر ۲۸۳ و چندجمله‌ای کاهش به صورت $f(x)=x^{283}+x^{12}+x^7+x^5+1$ 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 کیلو بیتی است. علت انتخاب این خانواده از FPGAها، داشتن سطح کافی 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 در جدول (۱) ارائه شده است.

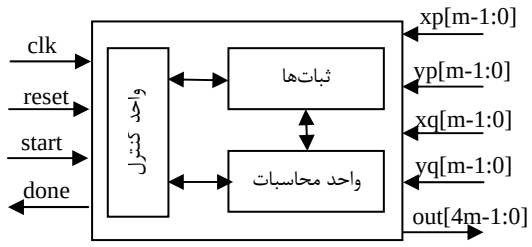
جدول (۱) نتایج پیاده‌سازی سخت‌افزاری واحدهای محاسباتی

واحد محاسباتی	میدان	تعداد اسلایس	حداکثر فرکانس (MHz)	زمان محاسبه (ns)	سطح×زمان (Slice.µs)
مربع‌کننده	F_{2^m}	۱۶۴	۱۸۲/۱۵	۵/۴۹	۰/۹
	$F_{2^{4m}}$	۹۶۶	۱۶۱/۵۵	۶/۱۹	۵/۹۸
LSD (D=8)	F_{2^m}	۱۴۵۲	۳۵۸/۰۸	۱۱۰	۱۵۹
کاراتسوبا (۵ تکرار)	F_{2^m}	۱۱۶۵۱	۸۱/۳۶	۱۲/۲۹	۱۴۳
ضرب‌کننده	$F_{2^{4m}}$	۱۴۱۸۲	۳۱۲/۵۷	۱۱۰	۱۵۶۰
معکوس ضربی	F_{2^m}	۴۴۸۲	۱۵۳/۹۲	۱۹۶۰	۸۷۸۴
	$F_{2^{2m}}$	۶۸۸۱	۱۱۹/۶۲	۵۲۵۰	۳۶۱۲۵
توان‌رسانی	$F_{2^{4m}}$	۷۶۵۶	۳۷۲/۷۹	۱۰۴	۷۹۸

نتایج جدول (۱) نشان می‌دهد که مربع‌کننده ساده‌ترین عملیات در بین واحدهای محاسباتی است و در یک پالس ساعت انجام می‌شود. چون ضرب‌کننده روی $F_{2^{4m}}$ از ۹ ضرب LSD به صورت موازی و ۲۲ روی F_{2^m} استفاده می‌کند، دارای سطح اشغالی حدوداً ۹ برابر یک LSD و زمان برابر آن است. سطح اشغالی ضرب‌کننده کاراتسوبا حدوداً ۸ برابر LSD است، در حالی‌که ۹ برابر سریع‌تر از آن انجام می‌شود. معکوس ضربی روی $F_{2^{4m}}$ در میان تمام واحدهای محاسباتی میدان به دلیل وجود حلقه‌های تکرار در محاسبات آن، زمان‌برترین است. توان‌رسانی نیز به علت استفاده از ۵ ضرب موازی LSD، سطح اشغالی تقریباً ۵ برابر LSD و زمانی برابر محاسبه یک LSD دارد.

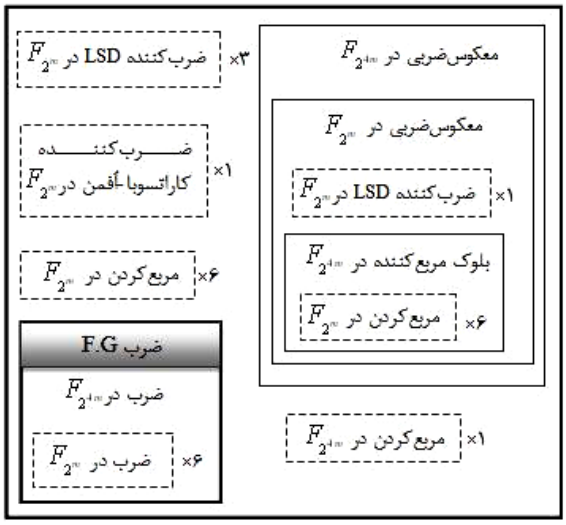
۴- معماری سخت‌افزاری زوج‌سازی η_T

شکل (۱) یک معماری سطح بالا را برای محاسبه زوج‌سازی η_T نشان می‌دهد. این معماری شامل واحد کنترل، ثبات‌ها و واحدهای محاسباتی روی میدان است. واحد کنترل با استفاده از ماشین حالت برای کنترل و انتقال حالت‌ها پیاده‌سازی شده است. ثبات‌ها با استفاده از اسلایس‌های موجود در تراشه FPGA پیاده‌سازی شده است. ورودی‌های این سیستم شامل سیگنال‌های کنترلی، سیگنال‌های داده و سیگنال‌های سیستمی است. سیگنال‌های کنترلی شامل start و done، سیگنال‌های داده شامل دو نقطه خم ورودی با مختصات (xp,yp) و (xq,yq) و خروجی out است. سیگنال‌های سیستمی شامل reset و clk است.



شکل (۱) معماری سطح بالا برای محاسبه زوج‌سازی η_T

شکل (۲) معماری پیشنهادی برای محاسبه زوج‌سازی η_T را نشان می‌دهد.



شکل (۲) معماری پیشنهادی برای محاسبه زوج‌سازی η_T

بلوک‌های محاسباتی الگوریتم ۱ شامل مربع‌کننده (خط ۲، ۷ و ۱۳)، ضرب‌کننده F_{2^m} (خط ۵ و ۱۵)، ضرب L.G (خط ۱۰)، مربع‌کننده F^2 (خط ۱۲)، ضرب F.G (خط ۱۸) و واحدهای محاسباتی الگوریتم ۲، شامل مربع‌کننده (خط ۱)، ضرب‌کننده F_{2^m} (خط۶ و۹)، معکوس D (خط۵)، توان‌رسانی $V^{2^{m}+1}$ (خط ۱۵)، مربع‌کننده W^2 (خط ۱۷)، ضرب V.W (خط ۱۹) است.

برای حالتی که دو الگوریتم ۱ و ۲ به صورت مجزا پیاده سازی و تمام بلوک‌های داخلی به صورت موازی فراخوانی شوند، در کل به ۳۶ ضرب‌کننده، ۲۳ مربع‌کننده و یک معکوس‌ضربی روی میدان F_{2^m} نیاز است. در این صورت سطح اشغالی بالا است و همچنین منابع استفاده شده در بلوک‌های داخلی در طول اجرای الگوریتم فقط یک بار به کار گرفته شده و بقیه زمان‌ها بیکار هستند. با ادغام دو الگوریتم و بلوک‌های داخلی (به جز بلوک ضرب F.G بدلیل قرار داشتن در حلقه الگوریتم ۱ و زمان‌بر بودن) 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.

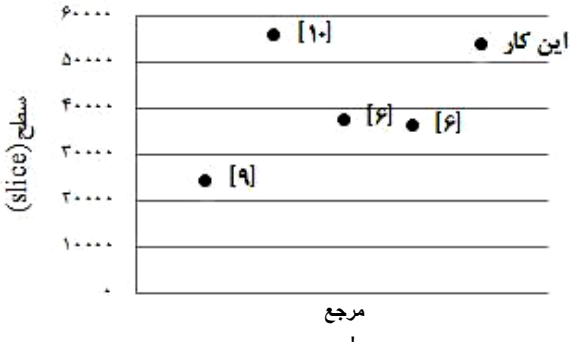
برای پیاده‌سازی ضرب در این معماری از دو نوع ضرب کننده LSD و کاراتسوبا استفاده می‌شود. ضرب‌کننده LSD در جاهایی از الگوریتم که زمان اهمیت چندانی نداشته و ضرب کننده موازی کاراتسوبا در حلقه‌های الگوریتم ۱، برای کاهش زمان محاسبه استفاده می‌شود. به دلیل زیاد بودن سطح اشغالی ضرب‌کننده کاراتسوبا، از تعداد ضرب‌کننده‌های LSD که هم زمان فراخوانی شده‌اند، کاسته می‌شود.

۴-۱- نتیجه پیاده‌سازی زوج‌سازی η_T

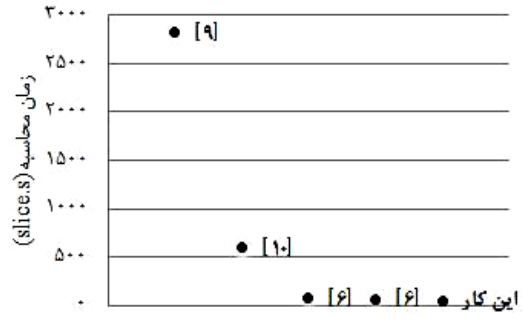
جدول (۲) مقایسه نتایج نهایی پیاده‌سازی زوج‌سازی η_T در این مقاله را با دیگران و نمودار (۱)، (۲) و (۳) مقایسه نتایج پیاده‌سازی زوج‌سازی با دیگران را بر حسب سه معیار سطح، زمان محاسبه و سطح در زمان نشان می‌دهد.

جدول (۲) مقایسه نتایج نهایی پیاده‌سازی زوج‌سازی η_T با دیگران

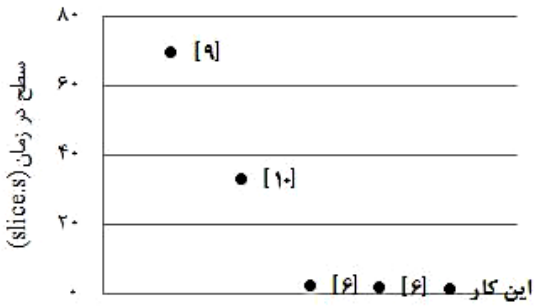
مرجع	FPGA	تعداد اسلایس	فرکانس (MHz)	زمان محاسبه (µs)	سطح×زمان (Slice.s)
[۷]	xc2v6000	۲۴۶۵۵	۴۷	۲۸۱۰	۶۹/۲۸
[۸]	xc4vfx140	۵۵۸۴۴	۱۶۰	۵۹۰	۳۲/۹
[۶]	xc2vp100	۳۷۸۰۳	۷۲	۶۱	۲/۳
[۶]	xc2vp100	۳۶۴۸۱	۱۰۰	۴۶/۱	۱/۶۸
این کار	**xc4vlx160**	**۵۴۱۰۸**	**۱۳۴/۷۲**	**۲۸/۵۳**	**۱/۵۴**



نمودار (۱) مقایسه نتایج پیاده‌سازی زوج‌سازی در این کار با دیگران بر حسب سطح



نمودار (۲) مقایسه نتایج پیاده‌سازی زوج‌سازی در این کار با دیگران بر حسب زمان



نمودار (۳) مقایسه نتایج پیاده‌سازی زوج‌سازی در این کار با دیگران بر حسب معیار سطح در زمان

!

مقایسه نتایج این پیاده‌سازی با دیگران، ۱۰٪ بهبود برای معیار سطح در زمان (نمودار (۱)) و ۳۸٪ بهبود در زمان محاسبه (نمودار (۲)) نسبت به بهترین نتیجه بدست آمده تاکنون [۶] نشان می‌دهد. بهبود نتیجه این کار در مقایسه با دیگران وابسته به دلائل مختلفی است:

ادغام دو بخش محاسبه زوج‌سازی η_T به منظور به اشتراک گذاری منابع.

استفاده از میدان پایه مناسب: انتخاب میدان F_{2^m} نسبت به میدان F_{3^m} برتری دارد [۶].

استفاده از ضرب‌کننده LSD با دو انباشته‌گر جهت افزایش فرکانس کار.

استفاده از ضرب‌کننده LSD و کاراتسوبا به صورت همزمان جهت کاهش زمان محاسبه.

موازی‌سازی در محاسبات جهت افزایش سرعت.

به اشتراک‌گذاری منابع: استفاده از ضرب‌کننده‌ها و مربع‌کننده‌ها بصورت اشتراکی برای محاسبات مختلف.

البته مقایسه نتایج بر حسب معیار سطح اشغالی (نمودار (۳))، افزایش ۳۸٪ نسبت به [۶] را نشان می‌دهد که ۸۰٪ از سطح FPGA انتخابی را پر می‌کند.

۵- پیاده‌سازی نرم‌افزاری

چندین بسته نرم‌افزاری برای پیاده‌سازی محاسبات جبری و نظریه اعداد وجود دارند. SAGE[1] [۱۶] یک بسته نرم‌افزاری رایگانِ متن‌بازِ[2] ریاضی، تحت مجوز گنو[3] است که زمینه‌های جبر، هندسه، نظریه اعداد، رمزنگاری و غیره را پشتیبانی می‌کند.

در اینجا برای تولید نقاط روی خم بیضوی به عنوان 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 را برای ورودی زوج سازی η_T تولید می‌شود. الگوریتم زوج‌سازی η_T بر اساس الگوریتم ۱ و ۲ برنامه نویسی شده است. زمان محاسبه برای زوج‌سازی η_T در این نرم‌فزار در حدود دو ثانیه است که نسبت به سخت‌افزار چند هزار بار کندتر است.

۶- نتیجه‌گیری و کارهای آینده

در این مقاله زوج‌سازی تیت بهبود یافته 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 به خصوص 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 صرفه جویی زیادی در زمان شده است.

مقایسه نتیجه پیاده‌سازی زوج‌سازی برای m=283 با دیگران، بهبود ۳۸٪ در زمان محاسبه و بهبود ۱۰٪ برای معیار سطح در زمان نسبت به بهترین نتیجه بدست آمده تاکنون [۶] را نشان می‌دهد. چندین عامل در افزایش عملکرد این پیاده سازی 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 به صورت همزمان و غیره است.

همچنین پیاده‌سازی نرم‌افزاری توسط نرم‌افزار ریاضی SAGE انجام و مشاهده شد که پردازنده سخت‌افزاری چند هزار برابر در این پیاده‌سازی سریعتر عمل می‌کند. این پیاده‌سازی نرم‌افزاری همچنین برای تولید نقاط روی خم و آزمایش صحت جواب‌های به دست آمده به کار گرفته شده است.

پیشنهاداتی برای کارهای آینده در ادامه آمده است:

۱) طراحی یک سخت‌افزار برای رسیدن به سطح امنیت معادل AES-۱۲۸ برای زوج‌سازی نیاز به اندازه میدان بزرگ و زمان خیلی طولانی برای محاسبه یا سطح سخت‌افزاری بسیار بالا

[1] Software for Algebra and Geometry Experimentation (SAGE)

[2] Open Source

[3] GNU

است. پس رسیدن به این سطح امنیت در یک زمان و هزینه منطقی می‌تواند یکی از کارهای آینده در این زمینه باشد.

۲) پیاده‌سازی سخت‌افزاری زوج‌سازی‌های جدید روی خم‌های معمولی[1] شامل زوج‌سازی‌های اِیت[2]، اِیت-آی و آر-اِیت تاکنون وجود ندارد.

۳) توسعه یک پردازنده دومنظوره بصورت هم‌زمان برای انواع پروتکل‌های رمزنگاری بر اساس خم‌های بیضوی و زوج‌سازی تیت.

۴) بررسی امنیت الگوریتم‌های زوج‌سازی از قبیل تیت، اِتا و اِیت از لحاظ آسیب‌پذیری در برابر حملات جانبی.

۵) ارائه یک معماری برای محاسبه زوج‌سازی تیت با توان مصرفی کم در کاربردهای خاص با محدودیت توان مثل شبکه-های حسگر بی‌سیم ضروری است.

مراجع

[1] Menezes, A., Okamoto, T. & Vanstone, S., "Reducing elliptic curve logarithms to logarithms on a finite field", IEEE Transactions on Information Theory 39, pp. 1639–1646, 1993.

[2] Frey, G., Ruck, G., "A remark concerning m–divisibility and the discrete logarithm problem in the divisor class group of curves", Math. Comp., Vol. 52, pp. 865–874, 1994.

[3] Miller, V.S., "Short programs for functions on curves", Unpublished Manuscript, http://crypto.stanford.edu/ miller, 1986.

[4] Barreto, P.S.L.M., Kim, H.Y., Lynn, B., & Scott, M., "Efficient algorithms for pairing–based cryptosystem" , in Advances in Cryptology Crypto'2002, ser. LNCS 2442, pp. 354–368, Springer–Verlag, 2002.

[5] Duursma, I., Lee, H.S., "Tate pairing implementation for hyperelliptic curves $y^2 = x^p – x+d$", in Advances in Cryptology –Asiacrypt'2003, ser. LNCS 2894, pp. 111–12, Springer–Verlag, 2003.

[6] Shu, C., "Hardware Architectures of Elliptic Curve Based Cryptosystems over Binary Fields", Thesis for degree of Doctor of Philosophy, George Mason University, Spring, Semester 2007.

[7] Barreto, P.S.L.M, "Efficient pairing computation on supersingular Abelian varieties", Cryptology ePrint Archive, Report 2004/375, http://eprint.iacr.org, 2004.

[8] Kwon, S., "Efficient Tate Pairing Computation for Elliptic Curves over Binary Fields", LNCS 3574, pp. 134–145, Springer–Verlag, 2005.

[9] Keller, M., Kerins, T., Crowe, F. and Marnane, W.P., "FPGA implementation of a GF(2^m) Tate pairing architecture", In K. Bertels, J.M.P. Cardoso, and S. Vassiliadis, editors, International Workshop on Applied Reconfigurable Computing, LNCS 3985, pp. 358–369, Springer, 2006.

[10] 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, Vol. 54, pp. 1077–1088, 2008.

[11] Ronan, R., OhEigeartaigh, C., Murphy, C., Scott, M. & Kerins, T., "FPGA acceleration of the Tate pairing in characteristic 2", In Proceedings of the IEEE International Conference on Field Programmable Technology – FPT 2006, pp. 213–220, IEEE, 2006.

[12] Beuchat, J.L., Brisebarre, N., Detrey, J., Okamoto, E. & Rodrguez–Henrquez, F., "A comparison between hardware accelerators for the modified tate pairing over F_2^m and F_3^m", In S.D. Galbraith and K.G. Paterson, editors, Pairing–Based Cryptography–Pairing 2008, LNCS 5209, pp. 297–315, Springer, 2008.

[13] Rodriguez-Henrquez, F., Saqib, N.A., Díaz-Pèrez, A. & Koc, C.K, "Cryptographic Algorithms on Reconfigurable Hardware", Signals and Communication Technology, pp. 35–62, 139-186, Springer–Verlag, 2007.

[14] Song, L., Parhi, K.K., "Low energy digit-serial/ parallel finite field multipliers", J. VLSI Signal Process,Vol. 19, No. 2, pp. 149–166, 1998.

[15] Rodriguez-Henriquez, F., Morales-Luna, G., Saqib, N., & Cruz-Cortes, N., "Parallel Itoh Tsujii Multiplicative Inversion Algorithm for a Special Class of Trinomials", Cryptology ePrint Archive, Report /2006/035, http://eprint.iacr.org/, 2006.

[16] Software for Algebra and Geometry Experimentation (SAGE) Version 4.2 available at: www.sagemath.org.

[1] Ordinary Curve

[2] Ate