ارائه روش رمزنگاری سریع و ایمن بر اساس الگوریتم رمز منحنی های بیضوی

علی کوهستانی

۱- گروه مهندسی کامپیوتر ، دانشکده مهندسی برق و کامپیوتر ، دانشگاه فنی و حرفه ای ، تهران ، ایران

آموزشکده کوثر گنبد کاووس

چکیده

در سال های اخیر روش های مختلفی جهت رمز نگاری مطرح شده است. روش هایی بیشتر مورد توجه قرار میگیرند که علاوه بر پیچیدگی بالا سرباری سخت افزاری کمتری داشته باشند. یکی از مناسب ترین روش ها برای این مهم، استفاده از رمز نگاری منحی های بیضوی می باشد این الگوریتم در عین حال که از کلیدی با طول ۱۲۸ بیت استفاده می کند دارای پیچیدگی لگاریتمی است. سرعت پایین عملیات یک اشکال بزرگ این روش است. در این مقاله روشی برای ضرب پیمانه ای مونتگمری ارائه میشود که برای محاسبه، فقط نیاز به یکسری شیفت و جمع دارد. جمع کننده ای سریع جایگزین CSA شده است. .در این روش علاوه برکاهش تاخیر (افزایش سرعت پردازش) فضای مصرفی کمتری استفاده می شود و کارایی بیشتری در مقایسه با معماری‌های قبلی دارد. با بکار گیری این معماری در الگوریتم ضرب پیمانه ای با منطق تقسیم می توان با کاهش فضای مصرفی الگوریتم بهبود یافته در حدود ۲۱٪ و در زمان پردازش نیز بهبودی در حدود ۱۵٪ داشته است .

کلمات کلیدی: رمزنگار ی مقارن، الگوریتم رمزنگاری، ECC ، مونتگمری، ضرب پیمانه ای، جمع پیمانه ای

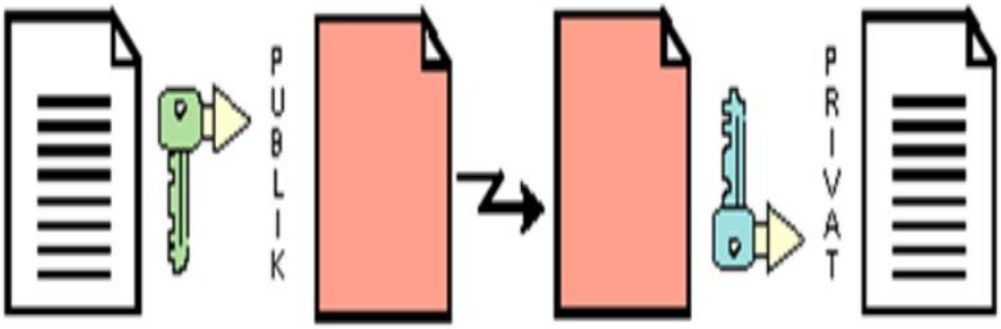
۱. مقدمه

در طول توسعه فناوری اطلاعات در دنیای کامپیوتر، امنیت نقش بسیار مهمی دارد. امنیت نه تنها بر روی محرمانگی داده ها تمرکز می کند بلکه بر چگونگی تبادل آنها نیزتمرکز دارد. یکی از تکنیک هایی است که برای حفظ محرمانه بودن داده ها در سیستم های دیجیتال به کار می رود رمزنگاری است. برای این منظور، الگوریتم های مختلف رمز نگاری مطرح شده اند. ویژگی مهم یک الگوریتم رمز پیچیدگی مناسب برای کاهش احتمال رمزگشایی توسط غیر است.الگوریتم های رمز نگاری مبتنی بر نوع عملکردشان بدو دسته متقارن و نا متقارت تقسیم بندی می شوند.(۱)

الگوریتم متقارن الگوریتمی است که در آن کلید رمزگذاری استفاده شده با کلید رمزگشایی یکسان است به طوری که الگوریتم را الگوریتم تک کلیدی نیز می نامند .کلید باید در طول فرآیند ارتباط برای فرآیند رمزگذاری و رمزگشایی ارائه شود .بر اساس میزان داده های پردازش شده، رمزنگاری کلید متقارن به دو نوع تقسیم می شود: تراشه بلوکی و تراشه جریان .در تراشه بلاک، داده ها به بلوک ها یا گروه های داده با طول داده های مشخص (در چند بایت) تبدیل می شوند، بنابراین در یک فرآیند رمزگذاری یا رمزگشایی، داده های ورودی دارای اندازه یکسانی هستند .در حالی که در تراشه جریان، داده ها به تک بیت ها یا گاهی اوقات در یک بایت تقسیم می شوند، بنابراین فرمت داده جریانی از بیت ها است تا سپس رمزگذاری و رمزگشایی شود.(شکل۱)

شکل 1: رمزنگاری کلید متقارن

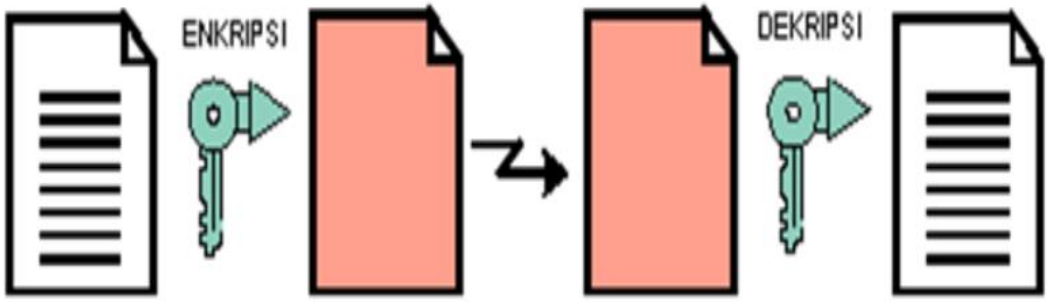
الگوریتم نامتقارن الگوریتمی است که در آن کلید رمزگذاری استفاده شده با کلید رمزگشایی یکسان نیست. در این الگوریتم



از دو کلید کلید عمومی (کلید عمومی) و کلید خصوصی (کلید خصوصی) استفاده شده است. کلیدهای عمومی به صورت عمومی توزیع می شوند، در حالی که کلیدهای خصوصی به صورت محرمانه توسط کاربر نگهداری می شوند. اگرچه کلید عمومی شناخته شده است، اما دانستن اینکه از کدام کلید خصوصی استفاده می شود بسیار دشوار خواهد بود.(شکل۲) لگوریتم های نامتقارن امنیت بالاتری نسبت الگوریتم های رمز متقارن دارند. در آن کلیدها به طور متفاوتی در فرآیندهای رمزگذاری و رمزگشایی استفاده می شوند و کلیدهای مورد استفاده طولانی تر از الگوریتم های متقارن هستند. در حالی که ضعف الگوریتم نامتقارن برای کلید با طول بزرگ و سپس زمان پردازش و فضای مصرفی افزایش یافته و منجر به کاهش سرعت عملیاتی می شود. نمونه هایی از الگوریتم های نامتقارن عبارتند از: RSA، DSA،ECC.

شکل ۲:رمزنگاری کلید نامتقارن

به دلیل محدودیت منابع (فضا و توان مصرفی) و نیاز به سرعت بالای محاسبات سیستم‌های قابل حمل، از قبیل تلفن‌های



همراه کارت‌های هوشمندو... از دیگر ویژگی مورد توجه در الگوریتم های رمز کاهش سرباری سخت افزاری است. بنابراین کوتاه بودن طول کلید در سیستم‌های رمزنگاری دارای اهمیت بسزایی می‌باشد و سبب کاهش توان مصرفی، پهنای باند موردنیاز سیستم، میزان پردازش و حافظه مصرفی می‌شود. سیستم رمزنگاری مبتنی بر منحنی های بیضوی به عنوان یک راهکار مناسب جهت رفع محدودیت‌های مذکور، استفاده می شود (۲و۳و۴). البته مشکل این سیستم، زمان‌بر بودن محاسبات بدلیل پیچیدگی بالای عملیات می‌باشد که بهتر است، رفع شود. یکی از روش‌های مطرح برای سرعت بخشیدن به محاسبات در این الگوریتم، استفاده از ضرب پیمانه‌ای مونتگمری است(۷و۸). معماری های مختلفی برای پیاده سازی ضرب پیمانه‌ای

مونتگمری پیشنهاد شده است. منطق تقسیم پیمانه‌ای یکی از این روش‌ها می باشد . روش پیشنهادی این مقاله، با استفاده از ضرب پیمانه ای مونتگمری و بهبود آن در عین حال که پیچیدگی رمزگشایی را با طول کلید ۱۲۸ بیت حفظ میکند سرعت پردازش را بالا می برد..

در ادامه و دربخش۲، برای درک بهتر عملکرد الگوریتم بهبود یافته به منطق الگوریتم تقسیم پیمانه‌ای که یکی از موثرترین روش‌ها جهت بهبود ضرب پیمانه‌ای مونتگمری است اشاره ای داشته، در بخش۳، به بررسی الگوریتم ضرب پیمانه‌ای مونتگمری با منطق تقسیم پیمانه‌ای پرداخته شده و سپس در بخش۴، به ارائه راه کاری برای سرعت بخشیدن به جمع مورد استفاده در آن، با کدینگ جدید خواهد پرداخت و در آخرخروجی الگوریتم بهبود یافته را بررسی و تحلیل نتیاج خواهد داشت.

۲.پیشینه تحقیق

ضرب نقطه مهم ترین بخش رمزنگاری منحنی بیضوی است که زمان قابل توجهی از زمان اجرا را به خود اختصاص میدهد. بنابراین افزایش راندمان کل سیستم به کارایی این قسمت بستگی دارد. افزایش کارایی ضرب مدولار باعث بهبود عملکرد کلی سیستم رمزنگاری می شود. با استفاده از روش تقسیم پیمانه ای در ضرب مونتگومری ، تاخیر ضرب مدولار کاهش می یابد. در ادامه این روش و روش های بهبود یافته آن بیان می شود.

۲.۱ الگوریتم تقسیم پیمانه ای میدان دودویی

با استفاده از تقسیم پیمانه ای با کمک محاسبه بزرگترین مقسوم علیه مشترک اعداد، ضرب پیمانه‌ای که در اکثر محاسبات رمزنگاری کاربرد دارد را بهبود می‌بخشد.. همانطور که مطرح است، دربیشتر الگوریتم‌های رمز نگاری، پیمانه عددی اول و فرد در نظر گرفته می‌شود. برای محاسبه حاصل تقسیم پیمانه‌ای دو عدد مخالف صفر در میدان دودویی بزرگترین مقسوم علیه مشترک بین مقسوم علیه و پیمانه را محاسبه می کند.

در این روش با استفاده از چهار متغیر کمکی A,B,U و V و با توجه به فرد و اول بودن پیمانه، اگر Y عددی زوج باشد بزرگترین مقسوم علیه مشترک بین مقسوم علیه و پیمانه برابر بزرگترین مقسوم علیه مشترک بین نصف مقسوم علیه و پیمانه است و اگر عددی فرد باشد حاصل جمع یا حاصل تفریق آن با پیمانه بر ۴ بخش پذیر خواهد بود و داریم:

$$\begin{aligned} &\text{if } (A-B)(\text{mod } 4)=0 \text{ then} \\ &gcd(((A-B))/(4,M)=gcd(A,M)) \\ &\text{if } (A+B)(\text{mod } 4)=0 \text{ then} \\ &gcd(((A+B))/(4,M)=gcd(A,M)) \end{aligned} \quad (۱)$$

این دستورات تا زمانی که A=0 شود ادامه پیدا می‌کند. برای این منظور ρ را برابر تعداد بیت‌های عدد A تعریف کرده و تا زمانی‌که مخالف صفر باشد دستورات بالا تکرار می‌شوند. ضرب نقطه مهم ترین بخش رمزنگاری منحنی بیضوی است که زمان قابل توجهی را برای اجرا صرف می کند .بنابراین افزایش راندمان کل سیستم به کارایی این قسمت بستگی دارد .افزایش کارایی ضرب مدولار باعث بهبود عملکرد کلی سیستم رمزنگاری می شود زیرا فرکانس آن در برخی کاربردها مانند رمزنگاری منحنی بیضی استفاده می شود .با اعمال سیستم اعداد باقیمانده (RNS) در ضرب مونتگومری به عنوان روشی برای ضرب مدولار، تاخیر ضرب مدولار کاهش می یابد .مجموعه‌های مدول RNS مناسب، عملیات زمان‌بر ضرب را با عملیات‌های کوچک‌تر جایگزین می‌کنند .در این مقاله دو مجموعه مدول متعادل با محدوده دینامیکی مناسب ارائه شده است و راندمان تبدیل از RNS به RNS که زمان‌برترین بخش ضرب مدولار مونتگومری است افزایش می‌یابد.(۵و۶)

۲.۲ بهبود الگوریتم ضرب مونتگمری با کمک روش تقسیم

برای سرعت بخشیدن به محاسبات به صورت $Z=X.Y(\text{mod } M)$ از روش مونتگمری استفاده می شود (۶و۷). در این روش به جای محاسبه در پیمانه ی M یک عدد مثبت و حقیقی R بزرگتر از M را در نظر می گیرند، طوریکه R نسبت به M اول باشد. مقدارمثبت R معمولا به صورت عددی از 2^kدرگرفته می شود وبا توجه به اینکه می¬توان ضرب و تقسیم برآن را با کمک جا به جایی یا انتقال بیتی و عملیات منطقی به راحتی انجام داد، محاسبات سریع و ساده خواهد شد.(۶و۷)

با دانستن gcd (R,M)=1 و با کمک گرفتن از دو عدد R^{-1} معکوس R در پیمانه‌ی (M و) M' یک پیمانه ی کمکی می توان فرمول زیر را محاسبه کرد:

$$0 \le R^{-1} \le ,0 \le \text{و } M' \le R, R \times R^{-1} - M \text{و } M' = 1 \quad (۲)$$

اگر عدد صحیح X کوچکتر از M انتخاب شود، آن‌گاه تبدیل یافته ی مونتگمری آن به صورت زیرمی‌شود:

$$\acute{X} = X \times (R \bmod M), X < M \quad (۳)$$

می‌توان ثابت کرد که مجموعه‌ی$\{i.R(\text{modM}) | 0 \le i \le M-1\}$ مجموعه کامل باقیمانده‌ها است. بنابراین یک رابطه‌ی یک‌به‌یک بین عددی‌که بین ۰ تا1-M است و مجموعه فوق وجود دارد. الگوریتم‌مونتگمری از این‌خاصیت استفاده‌کرده و روش سریعتری را برای ضرب پیمانه‌ای ارائه می‌کند. اگر$\bar{X}$ و $\bar{Y}$ را در نظر بگیریم ، رابطه ضرب مونتگمری به‌صورت زیر بیان می‌شود:

$$\bar{R} = \bar{X}.\bar{Y}.R^{-1}(\text{mod}M) \Rightarrow$$
$$R = X.Y(\text{mod}M) = \bar{X}.\bar{Y}.R^{-1}(\text{mod}M).R \quad (۴)$$

برای بهبود این الگوریتم ، می‌توان از منطق استفاده شده در الگوریتم تقسیم پیمانه‌ای کمک گرفت. با چک کردن بیت های کم ارزش A ، در صورتیکه دوبیت کم‌ارزش آن برابر صفر (A بر ۴ بخش‌پذیر است) و یا اگر فقط کم ارزش‌ترین بیت آن صفر (عددزوج) بودA(بر ۲ بخش پذیر است) و درحالت سوم، کم ارزش‌ترین بیت آن برابر یک (Aعدد فرد)است. در اعداد زوج $A := A/2$ یا $A := A/4$ خواهد شد. در صورت فرد بودن عدد با قرار دادن معادل حقیقی*آن، به حالات زیر می رسد:

$$(a_1 a_0) = 2 \times a_1 + a_0 = \begin{cases} 1 \\ 3 \end{cases} \text{ or } = \begin{cases} -1 \\ -3 \end{cases} \quad (۵)$$

برای حالاتی‌که مقدار حقیقیA برابر۱و۳- است، درصورت جمع با عدد منفی یک و در غیر این‌صورت با جمع کردن عدد۱ مضربی‌از ۴ خواهند شد. عدد ۱- را در کم ارزش‌ترین بیت متغیر کمکیB قرار داده و A+B را محاسبه می کند.

۲.۳ الگوریتم رمز منحنی های بیضوی

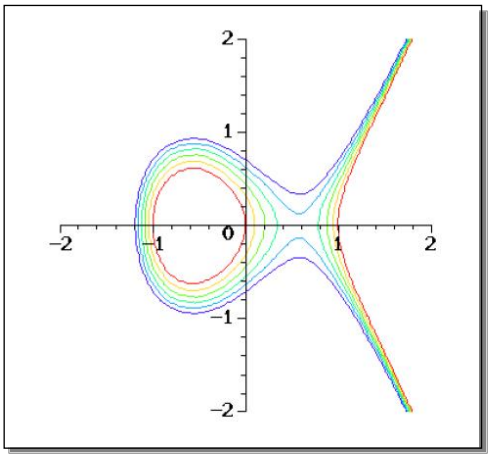
یکی از سیستم‌هایی که در سال‌های اخیر، برای انجام عملیات رمزنگاری مورد استفاده قرار می گیرد، سیستم رمزنگاری مبتنی بر منحنی های بیضوی است. این سیستم رمزنگاری برای نخستین بار در اواخر دهه ۸۰ توسط کوبلیتز† ارائه شد. به دلیل کوتاه بودن طول کلید و سطح بالای امنیتی آن، یکی از بهترین سیستم‌های رمزنگاری می باشد. از مزایای این سیستم نسبت به دیگر سیستم‌های رمزنگاری، می توان به مواردی چون دارا بودن بالاترین درجه محرمانگی به ازای هر بیت، نیاز به حافظه کمتر، توان مصرفی کمتر و کاهش پهنای باند مورد نیاز سیستم اشاره کرد(۸و۹)

* Integer
† Koblitz

از مهمترین و بارزترین ویژگی این رمز نگاری انجام عملیات روی میدان های متناهی می باشد. فرض کنید p یک عدد اول و میدان متناهی F_p شامل مجموعه ای از اعداد کوچکتر از p می باشد، منحنی های بیضوی E با مختصات دو بعدی* به صورت زیر تعریف می شود:

$$y^2=x^3+ax+b \qquad (۶)$$

فرض در این رابطه $a,b\in F_p$ و $4a^3+27b^2\neq 0 \pmod p$ می‌باشد، اگر مختصات(x,y) یک نقطه در رابطه (۶) صدق کند آن نقطه متعلق به منحنی های بیضوی است. باید توجه داشت که، $E(Fp)^{\square}$ مجموعه تمام نقاط بر روی منحنی های بیضوی (نقاط صحیح) و P نقطه ای بر روی E است.



شکل ۳: منحنی بیضوی

برای رمزنگاری در این سیستم، عدد تصادفی k در بازه [1,n-1] و متعلق به میدان F_p انتخاب کرده و آنرا به عنوان کید خصوصی در نظر می گیرد. سپس کلید عمومی Q را به صورت Q=kP محاسبه می کند، P نقطه ای روی منحنی های بیضوی می باشد.(۱۰و۱۱)

دشواری رمزنگاری منحنی های بیضوی به صورت مسأله لگاریتم گسسته است. به بیان دیگر، محاسبه k از روی نقاط P و Q دارای پیچیدگی نمایی است. این خصوصیت، مزیت اصلی سیستم رمزنگاری مبتنی بر منحنی های بیضوی می‌باشد.

در این سیستم رمزنگاری از عملیات ضرب عددی استفاده می شود. عملیات ضرب عددی به صورت زیر بیان می شود:

$$Q = k.P = P + P + \ldots + P \qquad (۷)$$

از رابطه بالا مشخص است که فقط از عملیات جمع، ضرب، توان برای محاسبه استفاده شده، زمان‌برترین آن‌ها ضرب است . با بهبود ضرب می توان کارایی کل محاسبات سیستم رمز منحنی های بیضوی را بالا برد. با توجه به اینکه عملیات در این سیستم مبتنی بر میدان است، استفاده از الگوریتم‌های در میدان متناهی ۲ به بهبود کارایی کمک زیادی می‌کند.

۳. الگوریتم بهبود یافته رمز منحنی های بیضوی

* Affin

† E یك خم بیضوي تعریف شده بر روي میدان F_p

همانطور که در بخش های قبل بیان شد، مهمترین، پیچیده ترین و اصلی ترین عملیات در الگوریتم رمز منحنی های بیضوی ضرب عددی یک عدد تصادفیK در مختصات نقطه ای یک نقطه مشخصP و محاسبه کلید است. با بررسی دقیق ویژگی های منحنی بیضوی و نحوه محاسبات در آن و در نظر گرفتن این منحنی بیانگر این است که برای بهبود کارایی عمل ضرب عددی سه راه وجود دارد، یکی از آنها انتخاب مناسب میدان منحنی بیضوی می باشد که مناسب ترین میدان از لحاظ پیاده سازی سخت افزاری انتخاب میدان دودویی است . سیستم نمایش اعداد می تواند به عنوان دومین راه، در بهبود عملیات ضرب عددی در منحنی بیضوی مؤثر باشد. برای همین منظور با مطالعه و تحقیق روش کدگذاری داده را به کار برده می شود و در آخر با استفاده از معماری جمع سریع می توان کارایی را به بهترین صورت بهبود داد.

با استفاده از کدگذاری مناسب می توان عمل جمع و به طبع آن عمل ضرب را در این سیستم بهینه تر نمود. برای این منظور ورودی های الگوریتم رمز را در صورت فرد بودن به صورت مضارب ۴ در می آوریم برای این کار کافی است فرمول زیر را در روش کد گذاری داده پیاده سازی نمود:

$$A = K \pm 1$$
$$A=4N \quad N=0,1,2,\ldots \qquad (8)$$

پیاده سازی الگوریتم رمز منحنی های بیضوی با استفاده از کدینگ بالا منجر به طراحی الگوریتمی برای جمع بیتی بدون انتشار رقم نقلی* خواهد شد که با کدگذاری عملوندهای جمع ، الگوریتمی مناسب از لحاظ فضا و زمان مصرفی ارائه می دهد.

با کمک الگوریتم جمع بیتی بدون انتشار رقم نقلی، و انجام عمل جمع به صورت بیت به بیت دو عدد A, B به فرم $A = \{a_n a_{n-1} \ldots a_0\}$ و $a_i \in \{-1,0,1\}$ انجام می‌شود. حاصل جمع S=X+Y با کمک متغیرهای کمکی $C,S,Y,\dot{S},\dot{Y},\dot{C}$ و قرار دادن مقدار اولیهC=0 ، Y=Y و S=X. منطبق با جدول۱ محاسبه می شود.

همانطورکه در بالا توضیح داده شد، در الگویتم۱، تصمیم‌گیری مبتنی بر کم‌ارزش‌ترین بیت A می‌باشد. در صورتیکه a_0 برابر صفر باشد(عددزوج است) و دو حالت وجود زیر دارد:

$$a_0 = 0 \rightarrow a_1 = \begin{cases} 0 \rightarrow A\ (mod 4) = 0\ , (A := A/4\ ) \\ 1/-1 \rightarrow A(mod\ 2) = 0\ , (A := A/2) \end{cases} \qquad (9)$$

ودر غیر اینصورت و برای حالتی که A عددی فرد است(با توجه به فرمول ۵ مقدار معادلA محاسبه شده است)، به مجموعه حالات زیر می رسد:

$$a_0 a_0 = 1 \rightarrow a_1 = \begin{cases} -1 \rightarrow A = -1 \\ 0 \rightarrow A = 1 \\ 1 \rightarrow A = 3 \end{cases}$$

$$A = \begin{cases} -3 \\ -1 \\ 1 \\ 3 \end{cases} \rightarrow$$

$$a_0 = -1 \rightarrow a_1 = \begin{cases} -1 \rightarrow A = -3 \\ 0 \rightarrow A = -1 \\ 1 \rightarrow A = 1 \end{cases} \qquad (10)$$

درحالتیکه A=-1,3 باشد، برای اینکه عدد مضربی از چهار باشد کافی است با عددی یک جمع شود. وقتی عدد A=1,-3 است عدد با منفی یک جمع خواهد شد. در الگویتم ضرب سیع، با توجه به اینکه در جمع‌های متوالی مقدار حاصل جمع هر مرحله

* Carry propagate

عدد اول در حاصل جمع مرحله بعد بوده، هر بیت آن می‌تواند متعلق به مجموعه حالت$\{-1,0,1\}$ باشد و عدد دوم در این فرمول عددی است که در این مرحله می‌خواهد به حاصل جمع مرحله قبل اضافه شود و عددی باینی $\{0,1\}$ است(جدول ۱). لذا برای حالاتی از فرمول ۱۱ که بیت های A مثبت بوده و قرار است از ۱ کم شود عدد اول را معادل 1- گرفته($b_0 = -1$ و مابقی بیت های B برابر صفر مقدار دهی می‌شود) بنابراین جمع (A-۱-) را محاسبه می‌کنیم و اگر بیت‌های A برابر 1- باشند و لازم به محاسبه(A-1) باشد امکان پذیر نبوده، برای حل این مشکل کافی است هر دو عدد A,B را منفی نمود و محاسبه ی نهایی را نیز منفی کرد.

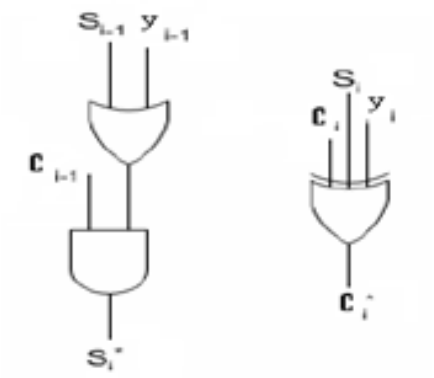
(۱۱)

$$\rightarrow(A-1) \xrightarrow{Exchange} -(-A+1)\,[a_0 a_1] = \begin{cases} [1-1] \\ [-1-1\,] \end{cases}$$

منطق الگوریت ضرب سریع به این صورت است که، اگر دوبیت ورودی جمع کننده با هم مساوی (طبق خاصیت CSA)رقم نقلی فقط در مرحله‌ی بعدی اثر گذاشته و در کل مدار منتشر نمی‌شود، در نتیجه تغییر خاصی ایجاد نکرده و جمع بیتی معمولی انجام می‌شود(فرمول۱۲).

$$for\ \forall i\ if\ a_i = b_i\ then\ s_i = x_i + y_i \qquad (12)$$

ولی اگر این دو بیت با هم برابر نباشند(یکی از دو بیت صفر و دیگری یک و یا منفی یک است) تنها فقط وقتی تغییر روش داریم که یکی از دوبیت صفر و دیگری یک باشد، که در این حالت هر دو بیت و بیت متناظر در $\dot{S}, \dot{Y}, \dot{C}$ ها برابر یک شده و حاصل جمع در این مرحله نیز برابر جمع دو بیت($S_i^{\cdot} + \dot{Y}_i$) تفریق بیتی با $\dot{c}_i$ خواهد بود. برای واضح تر شدن روش کار مدار این جمع کننده در شکل۲ آورده شده است:



شکل ۳جمع کننده سریع

می توان برای درک صحیح و کامل الگوریتم بهبود یافته ضرب مونتگمری با کمک جمع سریع مجموعه حالت و عملکرد متناسب هر یک از انها را در جدول زیر مشاهده نمود:

جدول ۱: جمع سریع با کد گذاری داده

A	$[a_1a_0]$	$[b_1b_0]$	$[\dot{a}_1\dot{a}_0]$	$[\dot{b}_1\dot{b}_0]$	$[\dot{c}_1\dot{c}_0]$	$[s_1s_0]$	$[\dot{c}_2]$
1	[01]	$[0\bar{1}]$	No change	No change	No change	[00]	**0**
-1	$[0\bar{1}]$	[01]	No change	No change	No change	[00]	**0**
3	[11]	[01]	No change	[11]	[01]	[00]	**1**
-1	$[\bar{1}1]$	[01]	No change	[11]	[01]	[00]	**0**
1	$[1\bar{1}] \rightarrow [\bar{1}1]$	$[0\bar{1}] \rightarrow$ [01]	No change	No change	No change	[00]	**0**
-3	$[11] \rightarrow [\bar{1}\,\bar{1}]$	$[0\bar{1}] \rightarrow$ [01]	No change	[11]	[01]	[00]	1

با توجه به جدول۱ می بینیم این روش پیاده سازی، منجر به تولید بیت صفر در کم ارزش ترین بیت‌های عدد ورودی می‌شود. در واقع عددA به ۲A یا ۴Aتبدیل خواهدشد. این تبدیل برای سیستم رمز ECC قابل قبول می‌باشد و مشکلی در الگوریتم رمز ایجاد نمی‌شود، زیرا ورودی‌های الگوریتم مونتگمری می‌توانند بیش از M و کمتر از ۲M(فرمول۹)باشند و طبق فرمول (۳) همواره از M کوچکتر است در صورت دوبرابر شدن نیز از ۲M بزرگتر نخواهد شد. در الگوریتم بهبود یافته چون طول ورودی‌ها (زوج می‌شوند) یک بیت افزایش می‌یابد، بنابراین تعداد سیکل‌های مورد نیاز الگوریتم یکی اضافه می‌شود.
طبق توضیحات بالا و باتوجه به جدول۱، شبه کد جمع سریع به صورت زیر می باشد:

Algorithm ۱(Fast ECC Addition)
Inputs: M: $2^{n-1} < M < 2^n$& gcd(M,2)=1
X,Y: $-M \leq X, Y < M$
Outputs: $Z = XY2^{-(n+1)} mod M$ $(-M < Z < M)$
Algorithm:
$A \coloneqq Y; B \coloneqq -1; U \coloneqq 0; V \coloneqq X; M \coloneqq M; \rho \coloneqq n;$
While $\rho \neq 0$ do
$if\ [a_1a_0] = 0\ then$ A $\coloneqq (A)/4; U \coloneqq (U)/4\ modM;$
else$if\ [a_0] = 0\ then$ A $\coloneqq (A)/2; U \coloneqq (U)/2\ modM;$
else $if\ ([a_1a_0] + [b_1b_0])mod4 = 0\ \ then\ q \coloneqq 1\ else\ q \coloneqq -1;$
A$\coloneqq (A + qB)/4; U \coloneqq (U + qV)/4\ modM;$
$\rho \coloneqq \rho - 1;$
$endwhile$
$if\ U \geq M\ then\ Z \coloneqq U - M\ else\ Z \coloneqq U;$

۴. بررسی نتایج پیاده سازی و مقایسه نتیجه پیاده سازی

برای پیاده سازی الگوریتم بهبودیافته‌ی مونتگمری، همانطور که در بخش های قبل بیان شد از الگوریتم جدید جمع سریع استفاده می شود. در این الگوریتم، جمع بیتی دو عدد X+Y با کمک متغیرهای کمکیS,C و به صورت $X + Y = (\dot{S} + \dot{Y})\emptyset\dot{C}$ امحاسبه می شود. معماری که برای جمع کننده به جای CSA استفاده می شود که همان جمع سریع با کدینگ جدید منطبق با جدول۱ در شکل۱ نشان داده شده است. تلفیق این معماری با الگوریتم ضرب مونتگمری در منطق تقسیم پیمانه ای منجر به بهبود کارایی از نظر فضای مصرفی و زمان اجرا شده است.
با پیاده سازی سخت افزاری این الگوریتم و استخراج نتایج آن بهبودی به صورت جداول زیر حاصل شد:

جدول ۲:حافظه مصرفی در روش الگوریتم رمز منحنی های بیضوی بهبود یافته

# of bit	Slice in new adder	Slice in booth adder	Memory Improve %
32	552	591	7
64	815	1023	19
128	1682	2050	18

با بررسی جدول بالا می توان مشاهده کرد که فضای مصرفی در الگوریتم بهبود یافته رمز منحنی های بیضوی، معادل ۱۸٪ در مصرف حافظه داشته است .

جدول **3** *زمان اجرا در روش الگوریتم رمز منحنی های بیضوی بهبود یافته*

# of bit	Delay (ns) Sd adder	Delay (ns) booth adder	Time Improve %
32	10.086	9.38	11.7
64	12.572	10.288	12.2
128	16.124	11.577	13.9
160	20.2	13.353	15.1

همانطور که ملاحظه شد، در بهبود الگوریتم ضرب مونتگمری با منطق تقسیم پیمانه ای و اعمال کدینگ جدید(جمع سریع)، سرعت پردازش حدود ۱۵٪ بهبود یافته است.

۵. نتیجه گیری

دراین مقاله روش جدیدی برای الگوریتم رمز منحنی های بیضوی پیشنهاد شد . به همین منظور از روش ضرب پیمانه ای مونتگمری استفاده شد. سعی شد این الگوریتم را نیز با استفاده از بهبود جمع کننده بهبود داد. الگوریتم جدید ضرب، فقط به شیفت و جمع نیاز داشت و برای محاسبه سریع تر ضرب از الگوریتم جمع سریع استفاده شد. در اکثر روش های پیاده سازی سخت افزاری برای جمع از معماریCSA که سریع ترین معماری موجود است، (باعث کاهش اتلاف وقت برای دریافت رقم نقلی از واحدهای محاسباتی قبل می شود) استفاده شده است. ولی در این مقاله از روش جمعی با کدینگ جدید استفاده شد که علاوه بر کاهش فضای مصرفی به طور چشمگیر، افزایش سرعت پردازش را داشت. در الگوریتم های رمزمنحنی های بیضوی که سرعت محاسبات وابستگی زیادی به زمان پردازش ضرب دارد و از طرفی دیگر عمل ضرب یک سری جمع متوالی است با کاهش زمان محاسبات جمع، کارایی الگوریتم ضرب که طبق فرمول۹ محاسبه می‌شود، نیز بهبود می‌یابد. . برای این منظور، از روشی استفاده شد که علاوه بر داشتن ویژگی CSA که همان جلوگیری ازانتشار رقم نقلی و در نتیجه افزایش سرعت پردازش است، کاهش فضای مصرفی را نیز دارد، استفاده شده است. در این روش کارایی که همان ضرب فضای مصرفی در زمان پردازش است بهبودی حدود ۲۸٪ داشته است. از ضرب کننده جدید می توان در سیستم های رمز نگاری که نیاز به ضرب های پیمانه ای دارند، مانند ECC استفاده کرد.

۶. مراجع

[1] P.P.Santoso, E.Rilvani, A.B.Trisnawan, "Systematic literature review: comparison study of symmetric key and asymmetric key algorithm",2nd Nommensen International Conference on Technology and Engineering, IOP Conf 2018.

[2] Y. El.Housniand T. Belytschko, "Introduction to the Mathematical Foundations of Elliptic Curve Cryptography," EY Wavespace LAB – Paris, HAL Id: hal-01914807,2018.

[3] N.J.G.Saho, E.C.Ezin "Comparative Study on the Performance of Elliptic Curve Cryptography Algorithms with Cryptography through RSA Algorithm," Proceedings of CARI 2020,Ecole Polytechnique de Thiès, Sénégal October 2020,

[4] W.Bos, J.Alex Halderman, N.Heninger " Elliptic Curve Cryptography in Practice", ASIACRYPT, LNCS. Springer, 2013.

[5] E.Kurniasari,A.E.Putra,N.G.Augoestien,"Implementation of the Montgomery Modulae based RSA Algorithm on FPGA" 5th international conference onscience and technology (?ICST),2019.

[6] S.Sharifi, M.Esmaeildoust , M.R.Taheri, "Efficient Implementation of RNS Montgomery Multiplication Using Balanced RNS Bases", Journal of mathematics and computer science, August 2014.

[7] P.L.Montgomery, Modular Multipication without trial division, Math.Computation, VOL.44,1985.

[8] D. J. Guan, Montgomery Algorithm for Modular Multiplication , Math.Computation ,August 25, 2003.

[9] K.Manochehri , B.Sadeghian & S.Pourmozafari, Very Fast Multi-Operand Addition Method by Bitwise subtraction , EEE Information Tecnology, April 2008.

[10] Monika, T.Tomar, V. Kumar "Implementation of elliptic – curve cryptography", International Journal of Electrical Engineering and Technology (IJEET), Issue 2, March-April 2020

[11] S.Vollala,N.Ramasubramanian,U.Tiwari, "introduction to Montgomery multiplication," ,in book: Energy-Efficient Modular Exponential Techniques for public- ket cryptography (pp.121-133), July 2021.