УДК 004.056.5

DOI: 10.18372/2073-4751.77.18659

**Марковський О.П.**, к.т.н.,
orcid.org/0000-0003-3483-4233,
e-mail: markovskyy@i.ua,
**Аль-Мріят Гассан Абдель Жаліль**,
orcid.org/0000-0002-4382-5309,
e-mail: grayatjo@gmail.com

# МЕТОД ПРИСКОРЕНОГО МОДУЛЯРНОГО ПІДНЕСЕННЯ ДО КВАДРАТУ ДОВГИХ ЧИСЕЛ ДЛЯ КРИПТОГРАФІЧНИХ ЗАСТОСУВАНЬ

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

## *Вступ*

Опубліковані в 1978 році принципи криптографії з відкритим ключем стали математичною базою більшості з сучасних механізмів захисту інформації. Вони лежать в основі стандартів цифрового підпису, обміну ключами, криптографічно строгої ідентифікації віддалених абонентів та інших [1]. Базова перевага криптографії з відкритим ключем – можливість створення ефективних та гнучких систем розподілення прав доступу до даних в сучасних умовах динамічного поглиблення інформаційної інтеграції набуває все більш вагомого значення. Разом з тим, в умовах сьогодення все більш чутливим стає недолік криптографії з відкритим ключем – потреба в значних за обсягом обчислювальних ресурсів для її реалізації. Це зумовлено тим, що в основі обчислювальної реалізації криптографії з відкритим ключем лежить операція модулярного експоненціювання $A^E \bmod M$, що виконується над числами, довжина $n$ яких значно перевищує розрядність $r$ процесорів. Зокрема, в сучасних умовах типовими є використання чисел розрядністю 2048 або 4096 [2].

Для механізмів захисту даних, що базуються на криптографії з відкритим ключем, рівень захисту напряму залежить від розрядності чисел. З огляду на те, що розширення використання хмарних технологій надає більші можливості для концентрації значних обчислювальних ресурсів для зламу механізмів захисту даних, очевидною є перспектива подальшого зростання довжини чисел, якими вони оперують. Реалізація модулярного експоненціювання потребує десятків і сотень мільйонів процесорних операцій, що призводить до суттєвих затримок функціонування засобів захисту даних. При збільшенні вдвоє розрядності чисел, над якими виконується операція модулярного експоненціювання, об'єм обчислень зростає в вісім разів [2]. Потреба в швидкій реалізації модулярного експоненціювання для задач захисту інформації стимулювала створення і широке використання спеціалізованих апаратних засобів – криптопроцесорів, якими нині оснащена переважна більшість комп'ютерних платформ середнього класу, включно до ноутбуків.

Разом з тим, існує достатньо широкий клас комп'ютерних платформ, для яких використання вартісних та складних спеціалізованих апаратних засобів для швидкої реалізації модулярного експоненціювання не може бути прийнятним. Мова йде про термінальні вбудовані мікроконтролери систем моніторингу стану та управління різноманітними об'єктами реального світу, обмін даними в яких здійснюється з використанням мережі Інтернет. Відповідно, для цього важливого класу комп'ютерних платформ мають реалізуватися протоколи інформаційної безпеки [3]. Для багатьох із них ці протоколи мають виконуватися в реальному часі. Це диктує нагальну необхідність пошуку шляхів прискорення виконання базової операції криптографії з відкритим ключем – модулярного експоненціювання. Аналіз цієї операції показує, що близько двох третин

обчислень займає виконання модулярного піднесення до квадрату. Відповідно, важливою задачею є пошук шляхів і можливостей прискорення обчислення модулярного квадрату.

Таким чином, наукова задача прискорення обчислення модулярного квадрату чисел великої довжини є актуальною і такою, що має практичне значення з огляду на особливості сучасного етапу розвитку комп'ютерних технологій.

### *Огляд та аналіз існуючих методів модулярного піднесення до квадрату*

Базовою обчислювальною операцією сучасної криптографії з відкритим ключем є модулярне експоненціювання, тобто обчислення $A^E \bmod M$, причому довжина $n$ чисел $A$,$E$ та $M$ значно перевищує розрядність $r$ процесора.

Класичний алгоритм обчислення модулярної експоненти передбачає виконання циклу, в якому здійснюється послідовний аналіз всіх розрядів коду експоненти $E$ [1]. В залежності від напрямку, я якому аналізуються розряди коду експоненти існує два різновиди класичного алгоритму. Якщо такий аналіз здійснюється від старших розрядів до молодшого, то в кожному з $n$ циклів виконується операція модулярного піднесення до квадрату поточного результату $R$: $R=R^2 \bmod M$ , початкове значення якого дорівнює одиниці, з наступним модулярним множенням $R$ на число $A$: $R=R\cdot A \bmod M$, якщо поточний розряд коду експоненти дорівнює одиниці. В різновиді класичного алгоритму модулярного експоненціювання з аналізом бітів експоненти від молодших розрядів до старших задіяні дві змінних $Q$ та $R$, початкові значення дорівнюють відповідно $A$ та одиниці: $Q=A$, $R=1$. І кожному із $n$ циклів поточне значення $Q$ підноситься до модулярного квадрату: $Q=Q^2 \bmod M$, після чого, за умови, що поточний розряд експоненти E дорівнює одиниці, значення $R$ множиться по модулю M на обчислений код $Q$: $R=R\cdot Q \bmod M$.

Аналіз класичного алгоритму модулярного експоненціювання показує, що операція модулярного піднесення до квадрату складає 2/3 від загального об'єму обчислень. Для різновиду модулярного експоненціювання з молодших розрядів коду експоненти, операція модулярного піднесення до квадрату лежить на критичному шляху алгоритму, тобто швидкість реалізації цієї операції практично визначає час модулярного експоненціювання.

Операція модулярного піднесення до квадрату включає дві складові: власне піднесення числа A до квадрату та віднаходження залишку від ділення квадрату на модуль ( модулярна редукція квадрату). Теоретично, виконання першої складової потребує приблизно вдвічі менше обчислювальних ресурсів ніж другої [4]. Ці операції реалізуються або послідовно, або їх виконання суміщається в часі. На практиці модулярне піднесення до квадрату виконується або побітово [5], або посекційно [6], з розділенням чисел на $s=n/r$ секцій, довжина яких дорівнює розрядності $r$ процесора. Важлими аспектом ефективної реалізації піднесення до квадрату довгих чисел є організація виключення операційної надмірності. Зокрема, при попарному множенні секцій, ця надмірність виявляється в тому, що добутки секцій з симетричними індексами рівні між собою. Виключення операційної надмірності дозволяє практично вдвічі скорити кількість процесорних множень.

Для реалізації другої складової модулярного піднесення до квадрату – модулярної редукції на практиці застосовуються дві технології, які дозволяють замінити складну в обчислювальному плані операцію ділення довгих чисел на більш ефективні операції. Перша з цих технологій запропонована Барретом [7] реалізує заміну операції ділення довгих чисел двома операціями множення. Відповідно, операція модулярного піднесення до квадрату реалізується трьома операціями множення. До числа недоліків розглянутої технології можна віднести те, що в результаті всіх трьох множень формуються числа розрядністю $2\cdot n$, подальша обробка яких потребує більших ресурсів.

Друга технологія модулярної редукції запропонована Монтгомері [8]. Ця технологія здійснює заміну операцію ділення довгих чисел на значно більш просту операцію логічного зсуву та додавання. Технологія Монтгомері передбачає чередування формування квадрату та редукції. Як результат – довжина чисел, з якими оперує ця технологія, не перевищує $n$+1. Технологіє орієнтована на формування часткових добутків з послідовним аналізом одного розряду множника. Відповідно, часткова модулярна редукція також здійснюється зі зсувом результату на один розряд. В якості недоліку технології Монтгомері можна розглядати те, що отриманий з її застосуванням результат відрізняється від справжнього: тобто потрібно використовувати спеціальні операції корекції.

При реалізації модулярного піднесення до квадрату на термінальних мікроконтролерах більша продуктивність досягається при застосуванні посекційної обробки. Проте таке рішення не дозволяє застосувати технологію Монтгомері. Тобто в такому варіанті не вдається досягти одночасності проведення множень та редукції [5].

Ще один ефективний шлях комп'ютерної реалізації модулярної редукції полягає в використанні передобчислень [9], які залежать тільки від модуля. Оскільки модуль в реальних застосуваннях є частиною відкритого ключа, то від змінюється рідко. Основний недолік використання технології передобчислень для швидкої реалізації модулярної редукції полягає в потребі в великих об'ємах пам'яті для зберігання таблиць передобчислень. Цей недолік є критичним для термінальних мікроконтролерів.

Перспективним шляхом прискорення виконання операції піднесення до квадрату довгих чисел виглядає застосування схем скороченого множення, запропонованих Карацубою А. [10]. Інша схема скорочено множення на основі швидкого перетворення Фурє запропонована Шенхаге-Штрасера [11], що має обчислювальну складність $O(n \cdot log\ n \cdot log\ log\ n)$. Пізніше цей підхід було вдосконалено в рамках широко відомого методу Фюрера [12]. Вказані підходи відносяться до множення двох різних чисел і не досліджувались стосовно задачі прискорення піднесення до квадрату. Самі методи мають більше теоретичне ніж практичне значення, оскільки вказана вище оцінка складності досягається асимптотично, тобто при достатньою великих значеннях $n$ і рекурсивному розкладенні множників.

Виконаний оглядовий аналіз відомих методів показав, що в повній мірі за показниками швидкодії вони не задовольняють сучасним вимогам при операції реалізації модулярного піднесення до квадрату в реальному часі на термінальних мікроконтролера систем моніторингу стану та управління об'єктами реального світу.

### *Мета досліджень*

Мета досліджень полягає в прискоренні комп'ютерної реалізації алгоритмів захисту даних з відкритим ключем на термінальних обчислювальних платформах *IoT* за рахунок підвищення продуктивності виконання їх базової операції – модулярного піднесення до квадрату чисел, розрядність яких значно перевищує розрядність процесора.

### *Організація мультипроцесорної реалізації модулярного множення*

Для досягнення поставленої мети пропонується реалізувати теоретичні можливості зменшення обчислювальної складності операції піднесення до квадрату в поєднанні з організацією ступінчатої модулярної редукції часткових добутків однакової ваги. При цьому для прискорення самої операції модулярної редукції пропонується організувати одночасну обробку $\eta$ двійкових розрядів часткового добутку.

При обчислення модулярного квадрату $A^2\ mod\ M$ на $r$-розрядному термінальному мікроконтролері, число $A$ ділиться на $s=n/r$ секцій, довжиною $r$ : $A=a_{s-1}\cdot 2^{(s-1)\cdot r} + a_{s-2}\cdot 2^{(s-2)\cdot r}+\ldots+a_1\cdot 2^r + a_0$, $\forall j\in\{0,1,\ldots,s\text{-}1\}$: $a_j\in\{0,1,\ldots,2^r\text{-}1\}$.

Обчислення квадрату $A^2$, з урахуванням того, що $\forall j,i\in\{0,1,\ldots,s\text{-}1\}$: $a_i\cdot a_j = a_j\cdot a_i$,

реалізується додаванням часткових добутків – результатів попарних множень секцій з урахуванням їх ваг за наступною формулою:

$$A^2 = \sum_{j=0}^{s-1} (a_j{}^2 \cdot 2^{2\cdot j\cdot r} + \sum_{i=j+1}^{s-1} 2\cdot a_j \cdot a_i \cdot 2^{(i+j)\cdot r}). \quad (1)$$

Згідно формули (1) для обчислення $A^2$ потрібно виконати $(s+1)\cdot s/2$ операцій процесорного множення та, в середньому, $1.25\cdot s\cdot(s+1)$ операцій процесорного додавання.

Для подальших перетворень формули (2) обчислення квадрату числа А його доцільно представити у вигляді суми двох компонент С та *D*, причому перша з них – *C* – являє собою суму квадратів секцій, помножених на відповідні вагові коефіцієнти:

$$C = \sum_{j=1}^{s-1} a_j{}^2 \cdot 2^{2\cdot j\cdot r}. \quad (2)$$

Цілком очевидно, що перша адитивна компонента формули (1) – *C*, яка утворюється квадратами значень секцій числа А завжди має вагу $2^{\gamma\cdot r}$ з парними значеннями γ. Друга компонента *D* являє собою суму добутків секцій числа А з різними індексами, помножених на збільшений в два рази ваговий коефіцієнт ( за рахунок урахування таким чином рівності добутків секцій з симетричними індексами):

$$D = \sum_{j=1}^{s-2} \sum_{i=j+1}^{s-1} 2\cdot a_j \cdot a_i \cdot 2^{(i+j)\cdot 2}. \quad (3)$$

Для організації суміщення в часі піднесення до квадрату $A^2$ та модулярної редукції формулу (3) необхідно трансформувати таким чином, що об'єднати всі часткові добутки з однаковою вагою. Аналіз формули (3) показує, що вага часткових добутків являє собою ступінь числа два з показником $\gamma\cdot r$, причому значення γ змінюється від нуля до $2\cdot(s-1)$: $\gamma\in\{0,1,2,\ldots, 2\cdot(s-1)-1, 2\cdot(s-1)\}$. Кількість κ часткових добутків, що мають однакову вагу γ визначається із того, що сума індексів секцій, які входять в добуток $a_i\cdot a_j$ мають дорівнювати γ : $i+j=\gamma$ при тому, що $i,j<s$, визначається наступною формулою:

$$\begin{array}{l} \forall 0 < \gamma < s : \kappa = 1 + \left\lfloor \dfrac{\gamma - 1}{2} \right\rfloor, \\ \forall\ \gamma \geq s\ ; \kappa = s - j + 1 + \left\lfloor \dfrac{\gamma - 1}{2} \right\rfloor \end{array}. \quad (4)$$

З урахуванням формули (4) обчислення адитивної компоненти *D* може бути організовано таким чином, щоб в зовнішньому циклі здійснювався перебір всіх можливих вагових коефіцієнтів, а в внутрішньому – всіх мультиплікативних компонент з однакового вагою:

$$\begin{array}{l} D = \sum\limits_{\gamma=0}^{s-1} \sum\limits_{i=0}^{\left\lfloor \frac{\gamma}{2} \right\rfloor} a_i \cdot a_{\gamma-i} \cdot 2^{\gamma\cdot r} + \\ \quad + \sum\limits_{\lambda=s+1}^{2\cdot s-2} \sum\limits_{l=\gamma-s+1}^{\left\lfloor \frac{\gamma}{2} \right\rfloor} a_l \cdot a_{\gamma-l} \cdot 2^{\gamma\cdot r} \end{array}. \quad (5)$$

Для зменшення кількості операцій процесорного множення при реалізації формули (5) пропонується скористатися наступною рівністю:

$$\begin{array}{l} (a_i + a_{i+1}) \cdot (a_{\gamma-i} + a_{\gamma-i-1}) = \\ a_i \cdot a_{\gamma-i} + a_i \cdot a_{\gamma-i-1} + a_{i+1} \cdot a_{\gamma-i} + a_{i+1} \cdot a_{\gamma-i-1} \end{array}. \quad (6)$$

З рівності (6) випливає, що сума двох добутків $a_i\cdot a_{\gamma-i} + a_{i+1}\cdot a_{\gamma-i-1}$, які входять в формулу (5) з ваговим коефіцієнтом $2^{\gamma\cdot r}$ може бути обчислена в наступному вигляді:

$$\begin{array}{l} a_i \cdot a_{\gamma-i} + a_{i+1} \cdot a_{\lambda-i-1} = \\ (a_i + a_{i+1}) \cdot (a_{\gamma-i} + a_{\gamma-i-1}) - a_i \cdot a_{\gamma-i-1} - a_{i+1} \cdot a_{\gamma-i} \end{array}. \quad (7)$$

Аналіз формули (7) показує, що до її складу входить компонента $a_i\cdot a_{\gamma-i-1}$, яка в формулу (5) входить з ваговим коефіцієнтом $2^{1+(\gamma-1)\cdot r}$ , а також компонента $a_{i+1}\cdot a_{\gamma-i}$, що входить в формулу (7) з ваговим коефіцієнтом $2^{1+(\gamma+1)\cdot r}$. Це дозволяє організувати обчислювальну реалізацію формули (5) таким чином, що при парних значеннях γ послідовно обчислювались всі добутки $a_i\cdot a_{\gamma-i}$ в межах визначених γ. При непарних значеннях γ пропонується здійснювати обчислювання парами згідно формули (7) з використанням раніше отриманих значень

добутків $a_{i+1}\cdot a_{\gamma-i}$, $a_i\cdot a_{\gamma-i-1}$,. Це дозволяє зменшити практично вдвічі число процесорних множень при непарних значеннях γ.

Іншими словами, пропонується при парних значеннях γ в формулі (5) обчислювати суму добутків $a_i\cdot a_{\gamma-i}$ зі збереженням їх значень в пам'яті. При непарних значеннях γ пропонується обчислювати суму добутків $(a_i + a_{i+1})\cdot(a_{\gamma-i}+a_{\gamma-i-1})$ від яких віднімаються збережені в пам'яті значення $a_i\cdot a_{\gamma-i-1}$ та $a_{i+1}\cdot a_{\gamma-i}$, обчислені для парних значень γ. Тобто, в технологічному плані, спочатку обчислюються компоненти суми $D$ для парних значень γ, а потім компоненти для непарних значень $\gamma>2$ $\gamma<2\cdot s-4$ з використанням збережених значень добутків при парних γ.

Таким чином, показано, що при застосуванні схем скороченого множення для прискорення комп'ютерної реалізації піднесення до квадрату на певному рівні обчислень потрібно використовувати результати двох суміжних рівнів: нижнього і верхнього. Виходячи з цього, в запропонованій схемі обчислення модулярного квадрату організується схема скороченого множення не на кожному рівні, а через рівень. Це означає, що на парних рівнях (відповідно, при парних значеннях γ) пропонується здійснювати обчислення всіх часткових добутків, які використовуються на тільки на поточному рівні, але й на наступному та попередньому. Крім того, на парних рівнях пропонується реалізувати обчислення компоненти С, тобто квадратів значень секцій довгого числа. Відповідно, при парних значеннях γ пропонується обчислювати подвоєну суму часткових добутків секцій, з різними номерами, сума яких дорівнює γ, додаючи до неї квадрат секції числа, номер якої дорівнює γ/2.

При обробці на непарних рівнях обчислень ( тобто при непарних значеннях γ) пропонується застосувати схему скороченого множення у відповідності з формулою (7) з використанням результатів обчислень на суміжних рівнях. При цьому в рамках обчислень на рівні для якого γ непарне, пропонується здійснювати також часткове обчислення часткових добутків верхнього рівня, які використовуються при скороченому множенні. При переході до наступного рівня, ці часткові добутки заново не вираховуються. Для збереження значень часткових добутків в запропонованому методі використовуються дві змінні: в $W$ зберігається сума часткових добутків попереднього рівня, яка використовується для реалізації схеми скороченого множення на поточному рівні, а в змінній $U$ формується і зберігається сума часткових добутків наступного рівня, яка використовується в схемі скороченого множення поточного рівня.

Для ефективної реалізації модулярного піднесення до квадрату пропонується здійснити суміщення процесів множення та модулярної редукції Монтгомері на рівні обробки часткових добутків одного рівня, тобто з однаковими вагомими коефіцієнтами. Таке рішення дозволяє працювати з форматом даних, який не перевищує $n+r$ двійкових розрядів або $s+1$ секцій. При використанні традиційної схеми суміщення множення та редукції Монтгомері, виконання останньої потребує $r$ циклів, в кожному із яких здійснюється додавання, при потребі, модуля і зсув часткового результату праворуч.

Для прискорення обчислювальної реалізації модулярної редукції Монтгомері пропонується організувати групову редукцію поточного результату одного рівня формування модулярного квадрату. Це означає, що модулярну редукцію Монтгомері пропонується реалізувати не на один, а відразу на $g$ розрядів. При такій редукції правий зсув поточного результату здійснюється не на один, а відразу на $g$ двійкових розрядів. Відповідно, для цього до поточного результату потрібно додавати корегуючий код, який являє собою добуток модуля на ціле число $l$ менше за $2^g$ , таке, щоб $g$ молодших розряди суми поточного результату та корегуючого коду прийняли нульове значення. Очевидно, що відповідні значення добутків модуля для кожного із $2^g$ можливих значень молодших $g$ двійкових розрядів поточного

результату мають бути обраховані попередньо і зберігатися в спеціальній таблиці передобчислень об'ємом $2^g$ ($n$+$r$)-бітових слів. Адресація такої таблиці здійснюється кодом $y$, який являє собою значення молодших $g$ двійкових розрядів поточного результату. Таким чином, таблиця передобчислень для здійснення групової редукції Монтгомері формується в залежності від значення $g$-розрядного коду $y$: Табличне значення $T[y]$ формується у вигляді:

$$\forall l \in \{1,2,...,2^g - 1\}: \\ y = 2^g - (l \cdot M) \,\&\, (2^g - 1),\ T[y] = l \cdot M \quad . \quad (8)$$

Порядок формування таблиці передобчислень для здійснення групової редукції Монтгомері може бути ілюстрований наступним прикладом. Якщо при $n$ =18 модуль $M$=227333, а $g$=3, то таблиця передобчислень для групової редукції Монтгомері, сформована у відповідності до формули (8) має вигляд, представлений в табл. 1.

*Таблиця 1*. Таблиця передобчислень для здійснення групової редукції Монтгомері при $g$=3 формується у відповідності з формулою (8)

| $y$ | $T[y]$ |
|---|---|
| 0 | 1 |
| 1 | $3 \cdot M$ = 681999 |
| 2 | $6 \cdot M$ = 1363998 |
| 3 | $1 \cdot M$ = 227333 |
| 4 | $4 \cdot M$ = 909332 |
| 5 | $7 \cdot M$ = 159133 1 |
| 6 | $2 \cdot M$ = 454666 |
| 7 | $5 \cdot M$ = 1136665 |

Використання групової редукції Монтгомері дозволяє практично в $g$ раз прискорити виконання цієї складної складової модулярного піднесення до квадрату. Практично єдиним обмежуючим чинником кратності $g$ прискорення є об'єм пам'яті таблиці передобчислень.

Таким чином, в запропонованому методі органічно капсулюється два чинника прискорення обчислення модулярного піднесення до квадрату: застосування схем прискореного множення секцій числа та групова редукція Монтгомері.

При рознесеному виконанні піднесення до квадрату та групової редукції Монтгомері обчислення проводяться над $2 \cdot n$ – розрядними числами. Відповідно, кожна операція над такими числами потребує $2 \cdot n/r$ процесорних операцій. Загальна кількість процесорних операцій додавання та зсувів для реалізації редукції дорівнює $1.5 \cdot n/g$ за умови застосування групової редукції Монтгомері. Якщо модулярна редукція виконується над $2 \cdot n$ – розрядними числами, то загальна кількість процесорних операцій становить $3 \cdot n^2/(r \cdot g)$. В запропонованому варіанті об'єднання операцій піднесення до квадрату та редукції операції здійснюються над практично вдвічі коротшими операндами, тобто загальна кількість потрібних процесорних операцій становить $1.5 \cdot n^2/(r \cdot g)$.

В формалізованому вигляді пропонований метод прискореного модулярного піднесення до квадрату довгих чисел зводиться до виконання наступної послідовності дій:

1. Початкове значення поточного результату $R$ встановлюється в нуль: $R$=0 так само, як ваговий коефіцієнт γ: γ=0.

2. Якщо значення γ парне, тобто γ *mod* 2 = 0, перехід на п. 7. Якщо γ=1: $R = R+2 \cdot a_0 \cdot a_1$, перехід на виконання п. 12.

3. Значення суми $U$ добутків наступного рівня встановлюється в нуль: $U$=0. Початкове значення індексу $i$ сканування добутків поточного рівня обчислюється за формулою: $i$= (γ-1)/2.

4. Обчислюється $R=R+2\cdot(a_i+a_{i-1})\cdot(a_{\gamma-i}+a_{\gamma-i+1})$, $U=U+2\cdot a_i\cdot a_{\gamma-i+1}$.

5. Зменшення індексу $i$ на два: $i=i-2$. Якщо $i > 0$ і $\gamma-i \le s-2$, то повернення на повторне виконання п. 5, інакше, якщо $i=0$ або $\gamma-i = s-1$ : $R=R+2\cdot a_i\cdot a_{\gamma-i}$.

6. Обчислюється $R=R-W-U$. $W=U$. Перехід на п. 12.

7. Обчислюється $R = R + a_{\gamma/2}\cdot a_{\gamma/2}$, якщо $\gamma=0$, перехід на п. 12 Якщо $\gamma = 2\cdot(s-1)$, перехід на п. 16 .

8. Якщо $\gamma=2$, $W=2\cdot a_0\cdot a_2$. $R=R+W$; перехід на п.12.

9. Обчислюється $R = R + W$. Початкове значення індексу $j$ встановлюється рівним одиниці: $j=1$.

10. Обчислюється $R = R + 2\cdot a_{\gamma/2-2\cdot j}\cdot a_{3+2\cdot j}$.

11. Здійснюється інкремент індексу $j$: $j=j+1$, якщо $\gamma/2-2\cdot j \ge 0$ та $3+2\cdot j<s$ реалізується повернення на повторне виконання п. 10.

12. Реалізується часткова модулярна редукція проміжного результату $R$. Лічильнику $u$ кількості груп присвоюється значення $r/g$: $u=r/g$.

13. Обчислюється номер $y$ рядка таблиці передобчислень : $y= R \bmod 2^g$. До поточного результату $R$ додається табличне значення $T[y]$ : $R = R + T[y]$. Двійковий код поточного результату $R$ зсувається на $g$ розрядів праворуч: $R = R>>g$.

14. Здійснюється декремент індексу $u$: $u=u-1$. Якщо $u>0$, виконується повернення на повторне виконання п. 13.

15. Збільшується на одиницю значення $\gamma$: $\gamma=\gamma+1$. Якщо $\gamma> 2\cdot(s-1)$, виконується повернення на повторне виконання п. 2.

16. Кінець виконання процедури модулярного піднесення до квадрату. В змінній $R$ зафіксовано результат: $R = A^2\cdot Q^{-1} \bmod M$, де $Q$ – модулярна інверсія $2^n \bmod M$. Тобто для отримання правильного результату потрібно домножити його на $2^n \bmod M$.

Функціонування розробленого методу може бути ілюстровано наступним прикладом піднесення до квадрату 18-розрядного числа $A$= 100 101 011 111 010 $110_2 = 153558_{10}$ по модулю $M$=227333. Відповідно, правильний результат цієї операції дорівнює $A^2 \bmod M = 153558^2 \bmod 227333 = 171272$. Якщо припустити, що розрядність $r$ процесора, на якому здійснюється операція модулярного піднесення до квадрату дорівнює трьом: $r=3$, то кількість $s$ секцій, на які розбивається число А дорівнює $s=n/r = 18/3 =6$. Відповідно, числові значення кодів секцій числа А дорівнюють: $a_0 = 110_2 = 6$, $a_1= 010_2 =2$, $a_2 = 111_2 =7$, $a_3 = 011 =3$, $a_4=101_2 = 5$, $a_5 = 100_2 = 4$. Оскільки для цього прикладу розрядність $r$ процесора дорівнює кількості $g$ розрядів секції $r=s=3$

Згідно з викладеним вище, в рамках п.1 значення поточного результату $R$ встановлюється в нуль: $R=0$. Коефіцієнт $\gamma$ початкової ваги встановлюється в нуль : $\gamma=0$. Оскільки $\gamma$ парне, то здійснюється перехід на п.7, в рамках якого значення $a_0 = 6$ підноситься до квадрату і додається до $R$: $R=R+a_0^2 = 6^2 =36 = 100100_2$. Оскільки три молодших двійкових розряди поточного результату $R$ дорівнюють 4, до нього додається табличне значення $T[4]$ : $R= R+T[4] = 36+909332 = 909368$ і отриманий результат зсувається праворуч на три розряди: $R>>3 = 113671$.

При $\gamma=1$ до поточного значення $R$ має додаватися лише один частковий доданок, тобто схема скороченого множення не застосовується: у відповідності до п.2 $R = R+2\cdot a_0\cdot a_1 = 113671+2\cdot6\cdot2 = 113695$. Молодші три двійкові розряди цього числа утворюють код $y$=7: відповідно групова редукція Монтгомері здійснюється шляхом додавання табличного значення $T[7]$ : $R= R+T[7] = 113695+1136665 = 1250360$ і отриманий результат зсувається праворуч на три розряди: $R>>3 = 156295$.

При $\gamma=2$ в п.7 до поточного значення $R$ додається квадрат коду першої секції $a_1$ числа $A$: $R=R+a_1\cdot a_1 = 156295 + 4 = 156299$. В рамках наступного п.8 обчислюється число $W=2\cdot a_0\cdot a_2 = 2\cdot6\cdot7=84$, яке зберігається в пам'яті, після чого обчислене значення додається до поточного результату : $R=R+W= 156299+84 = 156383$, після чого

здійснюється перехід на п.12, який реалізує групову редукцію Монтгомері, тобто додавання до $R$ коду $T$[7] і зсуву отриманого результату на 3 розряду праворуч. В результаті отримується значення поточного результату $R$=161631.

При наступному значенні γ=3 у відповідності з п.4 здійснюється скорочене множення при значенні $i$= (γ-1)/2 = 1 шляхом обчислення $R=R+2\cdot(a_1+a_0)\cdot(a_2+a_3)$ = $R$+ 2·( 2+6)·(7+3) = 161631 +2·8·10 = 161791. Після цього обчислюється значення $U=U+2\cdot a_i\cdot a_{\gamma-i+1}$ = 0+2·$a_1$·$a_3$ = 2·2·3=12. Оскільки $i$-2<0 цикл проходу добутків поточного рівня закінчується і здійснюється перехід на п.6 в якому обчислюється остаточний результат формування добутку доданків шляхом виконання операції $R=R-W-U$ = 161791-84-12 = 161695. Значення $U$ копіюється в змінну $W$. Після цього виконується в рамках п. 12 групова редукція поточного результату $R$.

В подальшому робота здійснюється на описаною схемою. Проміжні результати роботи по циклам роботи для розглянутого прикладу наведені в табл. 2. На останньому циклі роботи при γ=10, тобто γ=2·($s$-1) після додавання до поточного результату $R$ квадрату старшої секції $a_5$ числа $A$: $R= R + a_5\cdot a_5$ = 7749 + 4·4 = 7765, згідно п.7 здійснюється перехід на кінцевий п.16. Тобто редукція на останньому циклі не виконується. В силу використання технології Монтгомері отриманий результат $R$= 7765 не співпадає з істинним $A^2$ *mod* $M$ = $153558^2$ *mod* 227333 = 171272. Для отримання останнього потрібно помножити $R$ на постійне число $2^{30}$ *mod* $M$ = $2^{30}$ *mod* 227333 = 48065: $7765\cdot 2^{30}$ *mod* 227333 = 7765·48065 *mod* 227333 = 171272. На практиці, при використанні операції модулярного піднесення до квадрату, як складової модулярного експоненціювання, корекція результату виконується у вигляді однієї операції після обчислення модулярної експоненти, тобто час корекції практично не впливає на швидкодію.

В рамках розглянутого прикладу будо виконано 15 операцій процесорного множення. Якщо не застосувати реалізовану в методу схему скороченого множення, то кількість процесорних множень становить 21. Якщо для реалізації модулярного піднесення до квадрату використовувати звичайне модулярне множення, то для наведеного прикладу реалізується 36 операцій процесорного множення.

*Таблиця* 2. Діаграма значень змінних запропонованого методу по циклах його роботи при реалізації модулярного піднесення до квадрату $153558^2$ *mod* 227333

| γ | $R$ після додавання часткових добутків | $W$ | $U$ | $R$ після групової редукції |
|---|---|---|---|---|
| 0 | 36 | 0 | 0 | 113671 |
| 1 | 113695 | 0 | 0 | 156295 |
| 2 | 156383 | 84 | 0 | 161631 |
| 3 | 161695 | 84 | 12 | 162295 |
| 4 | 162416 | 12 | 0 | 20302 |
| 5 | 20412 | 12 | 42 | 116218 |
| 6 | 116313 | 42 | 0 | 99789 |
| 7 | 99875 | 42 | 24 | 40901 |
| 8 | 40950 | 24 | 0 | 61952 |
| 9 | 61993 | 0 | 0 | 7749 |
| 10 | 7765 | | | |

Зрозуміло, що на противагу наведено*ve* прикладу, для якого $n$=18, при реальних значеннях $n$=2048 або $n$=4096, ефект скорочення кількісті процесорних множень від застовування скороченого множення значно більший.

### *Оцінка ефективності*

В якості основного показника ефективності може слугувати оцінка прискорення обчислення модулярного квадрату, яке забезпечує застосування запропонованого методу в порівнянні з відомими технологіями виконання цієї важливої для криптографічних застосувань операції. Такою базовою технологією може бути метод посекційного модулярного піднесення до квадрату $A^2$ *mod M* з розділенням фаз отримання квадрату $A^2$ та модулярної редукції з використанням технології Монтгомері.

Цілком очевидним, що в базовому варіанті потрібно здійснити $s\cdot(s\text{-}1)/2$ операцій секційного множення $r$-розрядних чисел. При цьому виключаються операції множення секцій з симетричними індексами.

В запропонованому методі кількість часткових множень для парних значень γ тотожна базовому варіанту, оскільки для зазначених значень γ виконуються окремо всі операції попарного множення секцій. Для непарних значень γ кількість попарних множень, завдяки застосуванню схеми скороченого множення практично вдвічі менша. Таким чином, вважаючи, що для великих значеннях $s$ кількість операцій множення для парних і непарних значень γ приблизно однакова, кількість операцій секційного множення в запропонованому методу становить 0.5+0.5·0.5= 0.75 від базового варіанту. Іншими словами, за рахунок застосування технологій скороченого множення запропонована схема дозволяє, в середньому, зменшити в 1.25 раз кількість процесорних множень при модулярному піднесенні до квадрату.

Крім операції піднесення до квадрату, в базовому варіанті здійснюється редукція Монтгомері $2n$-розрядного коду результату цієї операції. Редукція організована у вигляді $n$ циклів, в кожному з яких в залежності від значення молодшого біту поточного результату здійснюється додавання модулю M до нього і завжди реалізується його зсув праворуч. Відповідно, враховуючи, що кожна з цих операцій виконуються послідовно над $2\cdot n/r = 2\cdot s$ секціями, в середньому, редукція Монтгомері потребує $3\cdot n\cdot s$ процесорних операцій додавання і зсуву. Якщо позначити через $t_m$ час виконання процесором операції множення, а через $t_o$ – час реалізації операцій типу додавання чи зсуву, то загальний час $T_B$ модулярного піднесення до квадраті в базовому варіанті обчислюється формулою:

$$T_B = \frac{s\cdot(s+1)}{2}\cdot t_m + 3\cdot n\cdot s\cdot t_0. \quad (9)$$

В запропонованому методі групова редукція Монтгомері виконується $2\cdot(s\text{-}1)$ раз: в усіх циклах зміни γ, крім останнього. Редукція на кожному циклі здійснюється над $n$-розрядними числами і виконується $r/g$ раз. Редукція в запропонованому методі складається з додавання до $n$-розрядного поточного результату коду з таблиці передобчислень та зсуву. Тобто, загальна кількість операцій типу додавання чи зсуву становить $2\cdot(s\text{-}1)\cdot r/g$. Враховуючи, що ці операції виконуються послідовно над $n/r$ секціями , загальна кількість процесорних операцій вказаного типу для реалізації групової редукції складає $2\cdot(s\text{-}1)\cdot n/g$. Таким чином, загальний час T модулярного піднесення до квадрату в запропонованому методу обчислюється за формулою:

$$T = \frac{0.75\cdot s\cdot(s+1)}{2}\cdot t_m + \frac{2\cdot n\cdot(s-1)}{g}\cdot t_0. \quad (10)$$

Коефіцієнт прискорення β модулярного піднесення до квадрату, яке забезпечує запропонований метод, визначається відношенням часу виконання цієї операції в базовому варіанті та при використанні розробленого методу:

$$\beta = \frac{T_0}{T} = \frac{s\cdot(s+1)\cdot t_m + 6\cdot n\cdot s\cdot t_0}{0.75\cdot s\cdot(s+1)\cdot t_m + \frac{4\cdot n\cdot(s-1)\cdot t_0}{g}}. \quad (11)$$

Якщо позначити через $\mu=t_m/t_0$ співвідношення часу виконання процесором операцій множення і зсуву, а також вважати несуттєвою різницю між $s$ та $s$-1, формула (11) може бути спрощена до виду:

$$\beta \approx \frac{\mu+6\cdot r}{0.75\cdot\mu+\frac{4\cdot r}{g}}. \qquad (12)$$

Для термінальних мікроконтролерів час виконання операції множення можна, в першому наближенні, вважати в $r$ раз більшим за час операції додавання. Тобто, при типовій для сучасних мікроконтролерів розрядності $r$=32, значення $\mu\approx 32$. Для таких обчислювальних пристроїв формула (12) трансформується до більш простого виду:

$$\beta = \frac{7\cdot g}{0.75\cdot g+4}. \qquad (13)$$

З отриманої формули (13) видно, що для таких обчислювальних платформ, часова ефективність запропонованого методу повністю визначається значенням $g$ – тобто кількістю розрядів, для яких одночасно виконується редукція Монтгомері. Для типового для практики значення $g$=8 обчислене за формулою (13) значення $\beta$ дорівнює 5.6. Проведені експериментальні дослідження показали близькі до наведеної оцінки результати. З формули (13) також випливає, що теоретично граничне значення прискорення обчислення модулярного квадрату для цього класу обчислювальних платформ становить 7/0.25 = 9.33.

Якщо в процесорі, на якому реалізується запропонований метод, застосовуються спеціальні апаратні засоби прискореного множення, то різниці в часі виконання ним команд несуттєва. Відповідно, для таких обчислювальних платформ значення $\mu$ близьке до одиниці, тобто, що всі операції виконуються процесором за однаковий час. Формула (12) при $\mu$=1, $g$=8 та $r$=32 дає значення $\beta$= 11.52.

### *Висновки*

В результаті проведених досліджень, направленої на прискорення процесорної реалізації домінуючої операції механізмів криптографічного захисту з відкритим ключем – модулярного піднесення до квадрату довгих чисел отримані наступні результати.

Теоретично обґрунтовано та запропоновано метод посекційного обчислення модулярного квадрату, який відрізняється циклічною організацією процесу по рівням обчислення сум часткових модулярних добутків з однаковими ваговими коефіцієнтами з виконанням часткової групової редукції Монтгомері, причому на половині рівнів формування добутків здійснюється за схемою скороченого множення, за рахунок чого скорочується кількість процесорних операцій множення і прискорюється отримання залишку від ділення на модуль.

Таким чином, основними чинниками досягнення поставленої мети – прискорення модулярного множення в запропонованому методі виступають організація скороченого множення при виключенні подвійного обчислення добутків секцій з симетричними індексами, чередування циклів підрахунку сум часткових добутків з однаковими вагомими коефіцієнтами та групової редукції Монтгомері, яка, з використанням передобчислень, що залежать лише від модуля, реалізує обробку відразу групи розрядів.

Теоретично доведено і експериментально перевірено, що використання запропонованого методу обчислення модулярного квадрату дозволяє прискорити реалізацію цієї операції в 5-6 раз. Оскільки ця операція лежить на критичному шляху базової операції криптографії з відкритим ключем – модулярного експоненціювання, то, використання запропонованого методу забезпечує практично п'ятикратне прискорення реалізації широкого класу алгоритмів захисту інформації.

### *Література*

1. Schneier B. Applied Cryptography. Protocols. Algorithms and Source codes in C. New York : John Wiley & Sons, Inc., 1996. 758 p.

2. Giorgi P., Imbert L., Izard T. Parallel modular multiplication on multi-core processors. *2013 IEEE 21st Symposium on*

*Computer Arithmetic* : proceedings, Austin, TX, USA, 7–10 April 2013 / IEEE. Piscataway, 2013. P. 135–142. DOI: 10.1109/ARITH.2013.20

3. Jurcut A. D., Ranaweera P. Xu L. Introduction to IoT Security. *IoT Security: Advances in Authentication* / ed. by M. Liyanage et al. Hoboken, 2020. P. 27–64.

4. Zuras D. More on squaring and multiplying larges integers. *IEEE Transactions on Computers*. 1994. Vol. 43, no. 8. P. 899–908.

5. Markovskyi O. et al. An Accelerate Approach for Public Key Cryptography Implementation on IoT Terminal Platforms. *2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT)* : proceedings, Greece, Athens, 13–15 October / IEEE. Danvers, 2023. 4 p. DOI: 10.1109/DESSERT61349.2023.10416516

6. Menezes A. J., van Oorschot P. C., Vanstone S. A, Handbook of Applied Cryptography. 1st ed. Boca Raton : CRC Press, 1996. 780 p.

7. Barrett P. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. *Lecture Notes in Computer Science. Vol. 263. Advances in Cryptology - CRYPTO '86. Proceedings* / ed. by A. M. Odlyzko. Berlin, 1987. P. 311–323.

8. Montgomery P. Modular multiplication without trial. *Mathematics of Computation*. 1985. Vol. 44, no. 170. P. 519–521.

9. Dhem J.-F., Quisquater J.-J. Recent results on modular multiplications for smart cards. *Lecture Notes in Computer Science. Vol. 1820. Smart Card. Research and Applications. Third International Conference, CARDIS'98, Louvain-la-Neuve, Belgium, September 14-16, 1998. Proceedings* / ed. by J.-J. Quisquater, B. Schneier. Berlin, 2000. P. 336–352.

10. Karatsuba A., Ofman Y. Multiplication of many-digital numbers by automatic computers. *Proc. USSR Acad. Sci*. 1962. Vol. 145, no. 2. P. 293–294.

11. Schŏnhage A., Strassen V. Schnelle Multiplikation grosser Zahlen. *Computing*. 1971. Vol. 7. 1971. P. 281–292.

12. Fὺrer M. Faste Integer Multiplication. *SIAM Journal on Computing*. 2009. Vol. 39, no. 3. P. 979–1005.

**Марковський О.П., Аль-Мріят Гассан Абдель Жаліль**

## МЕТОД ПРИСКОРЕНОГО МОДУЛЯРНОГО ПІДНЕСЕННЯ ДО КВАДРАТУ ДОВГИХ ЧИСЕЛ ДЛЯ КРИПТОГРАФІЧНИХ ЗАСТОСУВАНЬ

*В статті теоретично обґрунтовано та розроблено метод прискореного модулярного піднесення до квадрату чисел великої розрядності.*

*Прискорення обчислення модулярного квадрату досягається за рахунок комбінованого застосування технологій скороченого множення, перемежування циклів мультиплікативних операцій та групової редукції, а також організації одночасної обробки групи розрядів з використанням передобчислень при реалізації модулярної редукції.*

*Детально викладено методику побудови таблиці передобчислень для групової редукції Монтгомері а також запропоновану процедуру швидкого обчислення модулярного квадрату з застосуванням технологій скороченого множення та одночасної редукції Монтгомері групи розрядів. Виклад проілюстровано числовим прикладом.*

*Показано, що запропонований метод дозволяє прискорити в 5-6 раз обчислювальну реалізацію важливої для криптографічних застосувань операції модулярного множення довгих чисел в порівнянні з відомими технологіми.*

***Ключові слова**: модулярне експоненціювання; модулярне піднесення до квадрату; модулярна редукція Монтгомері; технології скороченого множення; криптографія з відкритим ключем.*

**Markovskyi O.P., Al-Mrayat Ghassan Abdel Jalil Halil**

**METHOD FOR ACCELERATED MODULAR SQUARING OF LONG LENGTH NUMBERS FOR CRYPTOGRAPHIC APPLICATION**

*The article theoretically substantiates and proposes a method for accelerated modular squaring of long length numbers.*

*Acceleration modular square calculation is achieved through the combined using of abridged multiplication technology, alternating cycles of multiplicative operations and partial modular reduction, as well as organizing simultaneous processing of a group of bits using pre-calculations under reduction.*

*The methodology for constructing of precomputations table for group Montgomery reduction is described in detail, as well as the proposed procedure for quickly calculating a modular square using abridged multiplication technologies and simultaneous Montgomery reduction of several digits. The presentation is illustrated with a numerical example.*

*It is shown that the proposed method makes it possible to speed up the computational implementation of the modular squaring operation, which is important for cryptographic applications, by 5-6 times compared to known technologies.*

***Keywords**: modular exponentiation; modular squaring; Montgomary modular reductions; abridged multiplication technologies; open key cryptography.*