МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кваліфікаційна наукова
праця на правах рукопису

Охріменко Андрій Олександрович

УДК 004.056.55:003.26

ДИСЕРТАЦІЯ

МЕТОДИ АРИФМЕТИЧНИХ ПЕРЕТВОРЕНЬ В ПОЛЯХ І КІЛЬЦЯХ ДЛЯ КРИПТОГРАФІЧНИХ ЗАСТОСУВАНЬ

Спеціальність 05.13.21 – «Системи захисту інформації»

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело  А.О. Охріменко

Науковий керівник:
Ковтун Владислав Юрійович
кандидат технічних наук,
директор ТОВ «САЙФЕР ІТ»

Київ – 2020

АНОТАЦІЯ

Охріменко А.О. Методи арифметичних перетворень в полях і кільцях для криптографічних застосувань. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук (доктора філософії) за спеціальністю 05.13.21 – «Системи захисту інформації». – Національний авіаційний університет, Київ, 2020.

Дисертаційна робота присвячена розв'язанню актуальної науково-практичної задачі дослідження і розробки нових методів арифметичних перетворень над великими цілими числами з відкладеним переносом для підвищення швидкодії реалізації криптографічних перетворень, що мають місце в інформаційно-телекомунікаційних системах центрів сертифікації ключів національної інфраструктури відкритих ключів України. В роботі запропоновано метод представлення цілих чисел з відкладеним переносом, який за рахунок можливості відкласти операцію переносу зі старших розрядів в молодші та операцію займу з молодших розрядів у старші, дозволяє виключити взаємозалежність між машинними словами при виконанні арифметичних перетворень. Удосконалено методи арифметичних перетворень додавання, віднімання, зсуву вліво, зсуву вправо, множення, піднесення до квадрату, приведення за модулем, ділення та порівняння, які за рахунок використання цілих чисел в представленні з відкладеним переносом дозволяють підвищити швидкодію перетворень в полях та кільцях цілих чисел. Також в роботі запропоновано методи арифметичних перетворень множення, піднесення до квадрату та приведення за модулем великих цілих чисел з відкладеним переносом та розпаралелюванням в два та декілька потоків. Використання запропонованих методів дозволяє підвищити швидкодію перетворень в криптографічних системах електронного підпису, що використовуються в національній інфраструктурі відкритих ключів.

Ключові слова: електронний підпис, інфраструктура відкритих ключів, представлення цілих чисел, арифметичні операції, відкладений перенос підвищення швидкодії, розпаралелювання, просте поле, кільце цілих чисел, група точок еліптичної кривої, ECDSA, RSA, ДСТУ 41.

Okhrimenko A.O. Methods of arithmetic operations in rings of integers and prime fields for cryptographic applications. – Manuscript.

Thesis for a Candidate of Technical Science degree in specialty 05.13.21 – information security systems. – National Aviation University, Kyiv, 2020.

Thesis is devoted to solving the scientific and practical task of research and development new methods of arithmetic transformations over large integers with delayed carry for increasing implementation of cryptographic transformations that take place in information and telecommunication systems of certification authorities in national public key infrastructure of Ukraine.

The national PKI regulates the use of a qualified electronic signature according to the algorithms of DSTU 4145-2002, ECDSA, DSA and RSA. Operations of creating and verifying electronic signature are based on various mathematical methods: transformation in a ring of integers, field of integers and polynomials, in a group of points of an elliptic curve. All these transformations are impossible without arithmetic operations on integers.

In this work proposed the method of integer representation with delayed carry, which due to the possibility of postponing carry operation from higher to lower words and the loan operation from lower to higher words, eliminates the interdependence between machine words when performing arithmetic operations. Performing operations with integers in the DCF representation, the processor operates with machine words in which two blocks are allocated to store the carry bits and to store the information bits. To convert a binary number to DCF it is necessary to reserve in the machine word r-bits for carry, and the remaining v-bits are filled with bits from the integer in a binary form. To convert a number from a DCF representation to a binary form, it is necessary to adjust carry (iteratively apply the carry from the lower machine word to the higher).

To perform operations with integers in the DCF representation, it is necessary to modify the algorithms of arithmetic operations. Improved methods of arithmetic operations – addition, subtraction, left shift, right shift, multiplication, squaring,

modular reduction, division and comparison, which by using integers in the delayed carry representation can increase the speed of operations in fields and rings of integers.

The use of delayed carry allows to apply some approaches of parallelization for methods of arithmetic operations, for example, multiplication, squaring and modular reduction. In these operations, there is a multiplication operation, which consists of two multiplication cycles that can be parallelized. The first approach involves the parallel execution of the first and second multiplication cycles with subsequent adjustment of the results in two threads. The second approach involves the parallel execution of iterations of the first and the second multiplication cycles, followed by the merging of intermediate results, using multiple parallel threads. Proposed the methods of arithmetic operations of multiplication, squaring and modular reduction of large integers with delayed carry and parallelization in two or multiple threads.

This paper contains results of experimental studies of the proposed methods of arithmetic operations in fields and rings of integers, elliptic curves over the prime field, and cryptographic transformations of electronic signature schemes used in the national public key infrastructure of Ukraine. The use of the proposed methods allows to increase the speed of operations in cryptographic electronic signature systems in the national public key infrastructure.

Keywords: electronic signature, public key infrastructure, integer representation, arithmetic operations, delayed carry, speed enhancement, parallelization, prime field, ring of integers, elliptic curve points group, ECDSA, RSA, DSTU 4145.

Список публікацій здобувача:

1. А. Охрименко, В. Ковтун, «Умножения целых чисел с использованием отложенного переноса для криптосистем с открытым ключом», *Информационные технологи и системы в управлении, образовании, науке: Монография, под ред. проф. В.С. Пономаренко, Харьков: Цифрова друкарня* №1, 2013, С. 69-82.

2. А. Охрименко, В. Ковтун, «Метод повышения производительности операции приведения по простому модулю», *Информационные системы в*

управлении, образовании, промышленности: Монография, под ред. В.С. Пономаренко, Харьков: Вид-во ТОВ «Щедра садиба плюс», 2014, С. 204-219.

3. А. Охрименко, В. Ковтун, «Арифметические операции с отложенным переносом над целыми числами», *Информационные технологии и защита информации в информационно-коммуникационных системах: Монография, под ред. В.С. Пономаренко, Харьков: Вид-во ТОВ «Щедра садиба плюс», 2015, С. 193-207.*

4. R. Brumnik, V. Kovtun, A. Okhrimenko, S. Kavun, «Techniques for Performance Improvement of Integer Multiplication in Cryptographic Applications», *Mathematical Problems in Engineering, 2014, P. 1-7. (Scopus)*

5. V.Yu. Kovtun, M.G. Kovtun, A.O. Okhrimenko, «Commands integrity and authority in control radio link of UAV», *Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), 2015 IEEE International Conference, 2015, pp. 178-181. (Scopus)*

6. A.O. Okhrimenko, M.G. Kovtun, S.O. Gnatyuk, V.Yu. Kovtun, «Development of a search method of birationally equivalent binary edwards curves for binary weierstrass curves from DSTU 4145-2002», *in Proc. of 2nd Intern. Scientific-Practical Conf. on the Problems of Infocommunications Science and Technology (PIC S&T), Kharkiv, Ukraine, October 13-15, 2015, pp. 5-8. (Scopus)*

7. A. Okhrimenko, M. Kovtun, T. Gancarczyk, V. Karpinskyi, S. Gnatyuk. «Method of Algorithm Building for Modular Reducing by Irreducible Polynomial», in Proc. of the 16th International Conference on Control, Automation and Systems, Oct. 16-19, 2016, Gyeongju, Korea. pp.1476-1479. (Scopus)

8. A. Okhrimenko, V. Kovtun «Experimental research of the developed methods of arithmetic operations in cryptographic transformations according to the ECDSA scheme», *Proceedings 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Lviv, Ukraine, November 29, 2019. – CEUR Workshop Proceedings, p. 827-837. (Scopus)*

9. O.G. Korchenko, V.Yu. Kovtun, A.O. Okhrimenko, «Parallelization of Integer Squaring Algorithms with Delayed Carry», *Journal of Computer Networks, 2014, Vol. 2(2), pp 10-17.*

10. О.Г. Корченко, С.О. Гнатюк, Ю.Є. Хохлачова, А.О. Охріменко, «Основні критерії та вимоги до побудови сучасних криптосистем», *Вісник Інженерної академії України, №3-4, С. 77-83, 2011.*

11. В.Ю. Ковтун, А.А. Охрименко, В.В. Нечипорук, «Подходы к повышению производительности программной реализации операции умножения в поле целых чисел», *Захист інформації, Т. 14, № 1 (54), С. 68-75, 2012.*

12. С.О. Гнатюк, В.М. Кінзерявий, А.О. Охріменко, «Особливості криптографічного захисту державних інформаційних ресурсів», *Безпека інформації, Т. 17, № 1, С. 68-80, 2012.*

13. В.Ю. Ковтун, А.А. Охрименко, «Подходы к распараллеливанию программной реализации операции умножения в поле целых чисел», *Радиотехника. Всеукраинский межведомственный научно-технический сборник, № 171, С. 123-132, 2012.*

14. V. Kovtun, A. Okhrimenko, «Integer multiplication algorithm with delayed carry mechanism for public key cryptosystems», *Безпека інформації, Vol. 19, №1, pp. 45-50, 2013.*

15. А.А. Охрименко, «Эффективная программная реализация алгоритмов умножения целых чисел для современных платформ», *Вісник Інженерної академії України, № 2, С. 108-113, 2013.*

16. В.Ю. Ковтун, А.А. Охрименко, «Алгоритм возведения в квадрат целых чисел с использованием отложенного переноса», *Безпека інформації, Т. 19, №3, С. 188-192, 2013.*

17. А.А. Охрименко, «Обобщенные алгоритмы возведения в квадрат целых чисел с использованием отложенного переноса и технологий распараллеливания», *Вісник Інженерної академії України, № 1, С. 114-119, 2014.*

18. А.А. Охрименко, «Арифметика с отложенным переносом», *Захист інформації, Т. 16, № 2, С. 130-138, 2014.*

19. А.О. Охріменко, «Оптимізація програмної реалізації криптографічних алгоритмів», *Защита информации: сб. науч. труд., № 18, С. 44-51, 2011.*

20. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, С.Ю. Ковтун, С.П. Євсеєв, О.Г. Король, «Спосіб множення цілих чисел», Пат. 111632 Україна, МПК G06F 7/253 (2006.01). Заявка № u 2015 11473; заявл. 23.11.2015; опублік. 25.11.2016, Бюл. № 22.

21. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, С.П. Євсеєв, О.Г. Король, Р.В. Грищук, Г.П. Коц, «Спосіб піднесення до квадрата цілих чисел», Пат. 118065 Україна, МПК G06F 7/523 (Пат. 118066 Україна, МПК G06F 7/523 (2006.01)2006.01). Заявка № u 2016 13439; заявл. 27.12.2016; опублік. 25.07.2017, Бюл. № 14.

22. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, «Спосіб приведення за модулем цілих чисел», Пат. 118066 Україна, МПК G06F 7/523 (2006.01). Заявка № u 2016 13441; заявл. 27.12.2016; опублік. 25.07.2017, Бюл. № 14.

23. А.О. Охріменко, С.П. Євсеєв, Р.В. Грищук, О.Г. Король, Г.П. Коц, Р.В. Корольов, В.Ю. Ковтун, М.Г. Ковтун, «Спосіб криптографічного перетворення інформації з використанням подовжених кодів», Пат. 123375 Україна, МПК (2006) G09C 1/00. Заявка № u 2017 08985; заявл. 11.09.2017; опублік. 26.02.2018, Бюл. № 4.

24. А.О. Охріменко, С.П. Євсеєв, Р.В. Грищук, О.Г. Король, Г.П. Коц, Р.В. Корольов, В.Ю. Ковтун, М.Г. Ковтун, «Спосіб криптографічного перетворення інформації з використанням укорочених кодів», Пат. 123379 Україна, МПК (2006) G09C 1/00, H04L 9/06 (2006.01), G06F 21/72 (2013.01), G06F 21/60 (2013.01). Заявка № u 2017 08995; заявл. 11.09.2017; опублік. 26.02.2018, Бюл. № 4.

25. А.А. Охрименко, В.Ю. Ковтун, «Подходы к повышению быстродействия криптосистем с открытым ключом», *Проблеми і перспективи розвитку ІТ-індустрії: V міжнар. наук.-практ. конф., 25-26 квітня 2013 р., Харків, 2013, С. 202.*

26. А.А. Охрименко, «Применение механизмов отложеного переноса и распараллеливания для повышения производительности операции возведения в квадрат целых чисел», *Безопасность информации в информационно-телекоммуникационных системах: XVI Междун. науч.-практ. конф., 21–24 мая 2013 г., К., 2013, С. 28-29.*

27. А.А. Охрименко, В.Ю. Ковтун, «Алгоритм умножения целых чисел с использованием технологий распараллеливания», *Питання оптимізації обчислень (ПОО-XL): праці міжн. наук. конф., 30 вересня – 4 жовтня 2013 р., К., 2013, С.125-126.*

28. A. Okhrimenko, «Squaring algorithms for public-key cryptosystems», *Інфокомунікації – сучасність та майбутнє: матеріали III міжнар. наук.-пр. конф. мол. вчених 17-18 жовтня 2013 р., Одеса, 2013, С. 178-182.*

29. А.А. Охрименко, В.Ю. Ковтун, «Классификация методов повышения производительности операции приведения по большому простому модулю», *Проблеми і перспективи розвитку IT-індустрії: VI міжнар. наук.-практ. конф., 17-18 квітня 2014 р.: тези доп., Харків, 2014, С. 253.*

30. А.А. Охрименко, «Модифицированный алгоритм Баррета приведения целых чисел по модулю», *Інформаційні технології та комп'ютерна інженерія: IV міжнар. наук.-практ. конф., 28-30 травня 2014 р., Вінниця, 2014, С. 180-181.*

31. A. Okhrimenko, V.Yu. Kovtun, O.L. Stokipniy, «Integer representation with delayed carry», *AVIATION IN THE XXI-st CENTURY – Safety in Aviation and Space Technologies: VI World Congress, September 23-25, 2014., K., 2014, P. 1.11.10-1.11.14.*

32. A. Okhrimenko, «Arithmetic operations with delayed carry for public key transformations», *Актуальні питання забезпечення кібернетичної безпеки та захисту інформації Зб. наук. праць наук.-практ. конф., 25-28 лютого 2015 р., К., 2015, С. 83-84.*

33. А.А. Охрименко, В.Ю. Ковтун, «Операции арифметического сдвига над целыми числами с отложенным переносом», *Проблеми та перспективи*

розвитку IT-індустрії: VII міжнар. наук.-практ. конф., 17-18 квітня 2015 р., Харків, 2015, С. 30.

34. А.А. Охрименко, В.Ю. Ковтун, О.Л. Стокипный «Использование представления целых чисел с отложенным переносом в криптографических преобразованиях», *Безпека інформації в інформаційно-телекомунікаційних системах: матеріали XVI міжнар. наук.-практ. конф. 26-28 травня 2015 р.: К., 2015, С. 14-15.*

35. А.О Охріменко, М.Г. Ковтун, «Метод побудови алгоритму приведення по фіксованому модулю незвідного поліному», *Безпека інформації в інформаційно-телекомунікаційних системах: матеріали XVI міжнар. наук.-практ. конф. 25-26 травня 2016 р., Київ, 2016, С. 21.*

36. A. Okhrimenko, V. Kovtun «Experimental research of developed arithmetic transformations according to RSA», *XX International conference of higher education students and young scientists «POLIT. Challenges of science today: Modern information and communication technologies in aviation», Kyiv, May 1-3, 2020., p. 30-31.*

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АЦСК	–	Акредитований центр сертифікації ключів.
ДСТУ	–	Державний стандарт України.
ІВК	–	Інфраструктура відкритих ключів.
КЗІ	–	Криптографічний захист інформації.
КНЕДП	–	Кваліфікований надавач електронних довірчих послуг.
НБУ	–	Національний банк України.
НСЕЦП	–	Національна система електронного цифрового підпису.
ПТК	–	Програмно-технічний комплекс.
ЗЦ	–	Засвідчуючий центр.
ЦСК	–	Центр сертифікації ключів.
ЦЗО	–	Центральний засвідчуючий орган.
ЕК	–	Еліптична крива.
ЕП	–	Електронний підпис.
BF	–	Binary Form.
CMP	–	Certificate Management Protocol.
CPU	–	Central Processing Unit.
CRL	–	Certificate revocation list.
DCF	–	Delayed Carry Form.
DH	–	Diffie Hellman.
DSA	–	Digital Signature Algorithm.
ECDH	–	Elliptic Curve Diffie Hellman.
ECDSA	–	Elliptic Curve Digital Signature Algorithm.
ECGDSA	–	Elliptic Curve German Digital Signature Algorithm.
ECKCDSA	–	Elliptic Curve Korean Certificate-based Digital Signature Algorithm.
ECMQV	–	Elliptic Curve Menezes–Qu–Vanstone.
GF	–	Galois field.
IEEE	–	Institute of Electrical and Electronics Engineers.
KCDSA	–	Korean Certificate-based Digital Signature Algorithm.
MQV	–	Menezes–Qu–Vanstone.
NAF	–	Non-adjacent form.
OCSP	–	Online Certificate Status Protocol.
PKCS	–	Public Key Cryptography Standards.
PKI	–	Public Key Infrastructure.
RFC	–	Request for Comments.
RNS	–	Residue Number System.
RSA	–	Rivest, Shamir, Adleman.
SHA	–	Secure Hash Algorithm.
TSA	–	Time Stamping Authority.
TSP	–	Time Stamp Protocol.
X.509	–	Сімейство стандартів ITU-T для інфраструктури відкритого ключа.

ВСТУП

Актуальність. Розвиток сучасного суспільства визначається рядом базових процесів, серед яких виділяють глобалізацію та інформатизацію. Масове та всеосяжне використання інформаційних і телекомунікаційних технологій є реаліями сучасного суспільства, яке можна сміливо називати інформаційним. Глобалізаційні процеси втягують держави в безперервну гонку, в якій критичним є оперативна реакція на різні події, що відбуваються в світі. Оперативність досягається за допомогою високої інформатизації суспільства і активного розвитку інформаційних і телекомунікаційних технологій в багатьох галузях економіки і сферах життя людини. Держава виступає в ролі основної рушійної сили та регулятора цих процесів.

Автоматизація управлінських процесів в державі дозволяє значно збільшити масштаби та темпи розвитку всіх сфер економіки за рахунок більш якісної та ефективної обробки інформації, встановлення нових зв'язків між сферами, а також можливості відмови від паперового документообігу. З метою переходу до автоматизації управлінських процесів та розвитку інформаційного суспільства в Україні було прийнято ряд законів та постанов (зокрема, ЗУ «Про Національну програму інформатизації», «Про Концепцію Національної програми інформатизації», «Про телекомунікації», Постанову КМУ «Про схвалення Стратегії розвитку інформаційного суспільства в Україні» №386-р від 15 травня 2013 року, Розпорядження КМУ «Про схвалення Концепції розвитку цифрової економіки та суспільства України на 2018-2020 роки та затвердження плану заходів щодо її реалізації» №67-р від 17 січня 2018 року, спільний наказ Національного банку України, Національної комісії з цінних паперів та фондового ринку, Національної комісії, що здійснює державне регулювання у сфері ринків фінансових послуг, Міністерства фінансів України та Фонду гарантування вкладів фізичних осіб «Про схвалення Стратегії розвитку фінансового сектору України до 2025 року» від 16 січня 2020 року), створено державне агентство з питань електронного урядування України, яке в подальшому було перетворено в

Міністерство цифрової трансформації. Враховуючи необхідність переходу до електронного управління, виникла необхідність в регулюванні електронних управлінських процесів та забезпечення їх захисту від несанкціонованого доступу. З цією метою, зокрема, були розроблені ЗУ «Про електронні документи та електронний документообіг», а також «Про електронні довірчі послуги». Відповідно до ЗУ «Про електронні довірчі послуги», для широкого використання електронного підпису (ЕП) в країні функціонує та продовжує розвиватись Національна інфраструктура відкритих ключів (ІВК). Даним законом передбачається входження України в єдиний цифровий ринок ЄС за рахунок забезпечення транскордонного співробітництва, використання онлайн послуг, безпечної електронної ідентифікації та автентифікації.

Для створення електронного підпису, учасники електронного документообігу повинні мати особисті ключі та сертифікати відкритих ключів, які були отримані у центрі сертифікації ключів (ЦСК), а для створення юридично значимого електронного підпису – отримані в ЦСК, які є кваліфікованими надавачами електронних довірчих послуг (КНЕДП).

ЦСК відноситься до інформаційно-телекомунікаційних систем (ІТС) загального користування з територіально розподіленою інфраструктурою та забезпечують безперервне функціонування послуг електронного підпису в масштабі часу, наближеного до реального. Це стало можливим лише після появи високопродуктивних обчислювальних систем і високошвидкісного доступу до мережі Інтернет, що дозволяють працювати в умовах нерівномірного навантаження зі сторони користувачів довірчих послуг. Прикладом такого нерівномірного навантаження може бути подача звітності до державних органів в кінці звітних періодів.

Архітектура національної ІВК має ієрархічну структуру та висуває жорсткі вимоги як до обчислювальних потужностей вузлових елементів (КНЕДП, ЗЦ, ЦЗО), їх надійності та пропускної здатності телекомунікаційних мереж між ними. Досвід експлуатації ЦСК в державних органах (Національного Банку України, Офісу Генерального прокурора України,

Державної податкової служби) показує тенденцію до постійного зростання кількості сертифікатів відкритих ключів, що обслуговується, а також зростання кількості звернень до сервісів ЦСК (OCSP, TSP, CMP, HTTP). В результаті такого зростання, кількість звернень може перевищити розраховане навантаження на ІТС ЦСК, що приведе до погіршення якості обслуговування користувачів або навіть до повної відмови в обслуговуванні. Крім того, сервіси ЦСК (OCSP, TSP, CMP, HTTP) протягом дня можуть обробляти до кількох мільйонів запитів, що нерівномірно розподілені у часі. Слід зазначити, що при кожному формуванні відповіді на запит OCSP чи TSP, відповідний сервіс створює ЕП для цієї відповіді, відповідно отримувач зобов'язаний перевірити статус ЕП отриманої відповіді від сервісу. В поточній реалізації національної ІВК, при функціонуванні сервісів OCSP та TSP в ЦСК, основний час займає виконання криптографічних перетворення для створення та перевірки ЕП, приймання та обробка запитів, передача відповідей. Тобто, підвищення якості обслуговування користувачі довірчих послуг, а саме зменшення часу обробки запитів OCSP та TSP можливе за рахунок зменшення часу виконання криптографічних перетворень.

В національній ІВК при застосуванні кваліфікованого ЕП (КЕП) використовуються алгоритми ДСТУ 4145-2002, ECDSA, DSA та RSA. Операції зі створення та перевірки ЕП ґрунтуються на різних математичних апаратах: перетворення у кільці цілих чисел, полях цілих чисел та поліномів, в групі точок еліптичної кривої. Всі перераховані перетворення неможливі без арифметичних операцій над числами криптографічного розміру – великими цілими числами. При виконанні арифметичних операцій над цілими числами в двійковому представленні (масив машинних слів) може здійснюватися послідовний перенос з молодшого машинного слова в старше, або займ зі старшого в молодше. Послідовне виконання операцій над машинними словами на сучасних суперскалярних процесорах призводить до збільшення кількості тактів простою конвеєру та не дозволяє повноцінно використовувати потужності процесору. З іншого боку, необхідність послідового виконання

операцій над машинними словами не дозволяє здійснювати розпаралелювання арифметичних операцій між ядрами процесору. Інтерес представляє можливість відкласти врахування переносу чи займ між машинними словами.

Актуальність теми дисертаційного дослідження визначається необхідністю підвищення швидкодії ІТС ЦСК національної ІВК за рахунок підвищення швидкодії реалізації алгоритмів криптографічних перетворень на основі розробки методів та алгоритмів арифметичних перетворень над великими цілими числами з відкладеним переносом.

Зв'язок роботи з науковими програмами, планами, темами. Тематика дисертаційної роботи та одержані результати безпосередньо пов’язані з Постановою Президії Національної академії наук України №30 затвердженої 30.01.2019 р. «Про Основні наукові напрями та найважливіші проблеми фундаментальних досліджень у галузі природничих, технічних, суспільних і гуманітарних наук Національної академії наук України на 2019-2023 роки» та відповідають науковим напрямам в області «1.2. Інформатика», а саме:

- 1.2.1. Математичне моделювання та методи комп’ютерної математики:
  - 1.2.1.3. Розвиток теорії алгоритмів та обчислень, у тому числі паралельних.
  - 1.2.1.4. Дослідження математичних моделей, проблем комп’ютерної математики, оптимізації, оцінювання, ідентифікації.
  - 1.2.1.5. Дослідження обчислювальних алгоритмів, у тому числі квантових: розроблення теорії похибок, визначення складності, збіжності, стійкості, рекурентних співвідношень
- 1.2.8. Теорія та комп’ютерні технології інформаційної безпеки:
  - 1.2.8.2. Розроблення методів підвищення продуктивності систем асиметричної криптографії.

- 1.2.8.3. Розроблення ефективних криптографічних протоколів з використанням можливостей Національного стандарту електронного цифрового підпису на еліптичних кривих – ДСТУ 4145-2002.

Дисертаційне дослідження пов’язане зі Стратегією національної безпеки України від 26 травня 2015 року №287/2015 у контексті п.4.12 «Забезпечення кібербезпеки і безпеки інформаційних ресурсів, зокрема реформування системи технічного і криптографічного захисту інформації з урахуванням практики держав-членів НАТО та ЄС», зі Стратегією кібербезпеки України від 15 березня 2016 року №96/2016 і Рамковою програмою ЄС з досліджень та інновацій «Horizon Europe». Результати роботи відображені у звітах держбюджетних науково-дослідних робіт Національного авіаційного університету «Квантово-криптографічні методи захисту критичної інформаційної інфраструктури держави» (0117U006770) та «Система забезпечення конфіденційності критичної інформаційної інфраструктури держави на базі квантових детерміністичних протоколів» (д.р. № 0120U101400), у яких здобувач брав участь у якості виконавця.

Мета та задачі дослідження. Метою роботи є підвищення швидкодії інформаційно-телекомунікаційних систем центрів сертифікації ключів національної інфраструктури відкритих ключів за рахунок розробки методів арифметичних перетворень над великими цілими числами з відкладеним переносом.

Для досягнення поставленої мети **необхідно розв’язати такі основні задачі**:

– проаналізувати існуючі підходи до представлення цілих чисел, а також методи арифметичних перетворень над числами в цих представленнях;

– розробити метод представлення цілих чисел з відкладеним переносом;

– удосконалити методи основних арифметичних перетворень (додавання, віднімання, множення, піднесення до квадрату, приведення за модулем, ділення, порівняння та зсувів) з відкладеним переносом;

– удосконалити методи арифметичних перетворень множення, піднесення до квадрату та приведення за модулем з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках;

– удосконалити методи арифметичних перетворень множення, піднесення до квадрату та приведення за модулем з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків;

– програмно реалізувати розроблені методи арифметичних перетворень та експериментально дослідити їх для підтвердження ефективності в криптографічних перетвореннях з відкритим ключем для роботи з ЕП.

Об’єктом дослідження є процес криптографічних перетворень з відкритим ключем в інформаційно-телекомунікаційних системах ЦСК національної ІВК.

Предметом дослідження є методи арифметичних перетворень над великими цілими числами, що застосовуються у криптографічних перетвореннях з відкритим ключем.

Методи дослідження. Вибрані методи дослідження базуються на теорії ймовірності та комбінаторики (для аналізу складності алгоритмів); теорії полів, кілець та ідеалів (для удосконалення методів арифметичних перетворень над цілими числами); складності алгоритмів (для аналізу складності методів арифметичних перетворень); теорії криптографії (для аналізу криптографічних перетворень, що використовуються в різних схемах електронного підпису).

Наукова новизна отриманих результатів. У ході вирішення поставлених задач були отримані наступні результати.

– *вперше розроблено* метод представлення цілих чисел з відкладеним переносом, який за рахунок можливості відкласти операцію

переносу зі старших розрядів в молодші та операцію займу з молодших розрядів у старші, дозволяє виключити взаємозалежність між машинними операціями при виконанні арифметичних перетворень та в свою чергу підвищити швидкодію криптографічних перетворень з відкритим ключем.

– *удосконалено* методи арифметичних перетворень додавання, віднімання, зсуву вліво, зсуву вправо, множення, піднесення до квадрату, приведення за модулем, ділення та порівняння, які за рахунок використання цілих чисел в представленні з відкладеним переносом дозволяють підвищити швидкодію перетворень в полях та кільцях цілих чисел, що в свою чергу призводить до підвищення швидкодії криптографічних перетворень з відкритим ключем.

– *удосконалено* методи арифметичних перетворень множення, піднесення до квадрату та приведення за модулем великих цілих чисел з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках, що дозволяє підвищити швидкодію криптографічних перетворень з відкритим ключем.

– *удосконалено* методи арифметичних перетворень множення, піднесення до квадрату та приведення за модулем великих цілих чисел з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків, що дозволяє підвищити швидкодію криптографічних перетворень з відкритим ключем.

Практичне значення отриманих результатів полягає в наступному:

1. Удосконалено методи арифметичних перетворень множення (дозволив підвищити швидкодію реалізації в 1,05-1,7 разів для $w = 32$ біт, та 1,02-2,28 разів для $w = 64$ біт відносно прототипу), піднесення до квадрату (дозволив підвищити швидкодію реалізації в 1,25-3,19 разів для $w = 32$ біт, та 1,01-4,12 разів для $w = 64$ біт відносно прототипу), приведення за модулем (дозволив підвищити швидкодію реалізації в 1,02-1,8 разів для $w = 32$ біт, та 1,01-4,12 разів для $w = 64$ біт відносно прототипу) великих цілих чисел з відкладеним переносом.

2. Удосконалено методи арифметичних перетворень множення (дозволив підвищити швидкодію реалізації в 1,01-3,24 разів для $w=32$ біт, та 1,07-2,29 разів для $w=64$ біт відносно прототипу), піднесення до квадрату (дозволив підвищити швидкодію реалізації в 1,01-5,75 разів для $w=32$ біт, та 1,18-2,46 разів для $w=64$ біт відносно прототипу), приведення за модулем (дозволив підвищити швидкодію реалізації в 1,22-2,10 разів для $w=32$ біт, та 1,04-2,46 разів для $w=64$ біт відносно прототипу) великих цілих чисел з використанням відкладеного переносу та паралельним виконанням двох циклів множення в двох окремих потоках.

3. Удосконалено методи арифметичних перетворень множення (дозволив підвищити швидкодію реалізації в 1,34-8,29 разів для $w=32$ біт, та 1,05-5,1 разів для $w=64$ біт відносно прототипу), піднесення до квадрату (дозволив підвищити швидкодію реалізації в 1,31-17,3 разів для $w=32$ біт, та 1,09-4,6 разів для $w=64$ біт відносно прототипу), приведення за модулем (дозволив підвищити швидкодію реалізації в 1,22-5,14 разів для $w=32$ біт, та 1,31-4,6 разів для $w=64$ біт відносно прототипу) великих цілих чисел з використанням відкладеного переносу та паралельним виконанням ітерацій двох циклів множення в декілька потоків.

4. Удосконалено методи арифметичних перетворень, що дозволяють підвищити швидкодію перетворень у кільці цілих чисел (операція додавання за модулем – в 1,01-1,97 разів для $w=32$ біт, та 1,03-3,28 разів для $w=64$ біт; операція віднімання за модулем – в 1,01-1,34 разів для $w=32$ біт, та 1,03-2,41 разів для $w=64$ біт; операція множення за модулем – в 1,01-1,34 разів для $w=32$ біт, та 1,03-2,41 разів для $w=64$ біт; операція піднесення до квадрату за модулем – в 1,02-1,46 разів для $w=32$ біт, та 1,02-1,73 разів для $w=64$ біт; операція піднесення до степеню за модулем – в 1,38-1,75 разів для $w=32$ біт, та 1,01-1,76 разів для $w=64$ біт).

5. Удосконалено методи арифметичних перетворень, що дозволяють підвищити швидкодію перетворень у простому полі цілих чисел (операція додавання (модуль загального вигляду) – в 1,02-1,09 разів для $w=32$ біт, та

1,01-1,07 разів для $w=64$ біт; операція додавання (модуль спеціального вигляду) – в 1,01-1,08 разів для $w=32$ біт, та 1,01-1,04 разів для $w=64$ біт; операція віднімання (модуль загального вигляду) – в 1,01-1,08 разів для $w=32$ біт, та 1,02-1,05 разів для $w=64$ біт; операція віднімання (модуль спеціального вигляду) – в 1,01-1,07 разів для $w=32$ біт, та 1,01-1,08 разів для $w=64$ біт; операція множення (модуль загального вигляду) – в 1,02-1,09 разів для $w=32$ біт, та 1,01-1,07 разів для $w=64$ біт; операція множення (модуль спеціального вигляду) – в 1,02-1,07 разів для $w=32$ біт, та 1,01-1,08 разів для $w=64$ біт; операція піднесення до квадрату (модуль загального вигляду) – в 1,02-1,08 разів для $w=32$ біт, та 1,01-1,07 разів для $w=64$ біт; операція піднесення до квадрату (модуль спеціального вигляду) – в 1,02-1,07 разів для $w=32$ біт, та 1,01-1,08 разів для $w=64$ біт; операція піднесення до степеню (модуль загального вигляду) – в 1,03-1,12 разів для $w=32$ біт, та 1,02-1,14 разів для $w=64$ біт; операція піднесення до степеню (модуль спеціального вигляду) – в 1,03-1,09 разів для $w=32$ біт, та 1,02-1,13 разів для w=64 біт.).

6. Удосконалено методи арифметичних перетворень, що дозволяють підвищити швидкодію перетворень у групі точок ЕК над простим полем (додавання точок ЕК – в 1,02-1,08 разів для $w=32$ біт, та 1,01-1,09 разів для $w=64$ біт; подвоєння точок ЕК – в 1,01-1,06 разів для $w=32$ біт, та 1,02-1,08 разів для $w=64$ біт; додавання точок ЕК в змішаних координатах – в 1,03-1,06 разів для $w=32$ біт, та 1,02-1,07 разів для $w=64$ біт; подвоєння точок ЕК в проективних координатах – в 1,02-1,08 разів для $w=32$ біт, та 1,03-1,08 разів для $w=64$ біт; скалярне множення випадкової точки ЕК з використанням проективних координат – в 1,02-1,12 разів для $w=32$ біт, та 1,02-1,09 разів для $w=64$ біт; скалярне множення фіксованої точки ЕК з використанням проективних координат – в 1,02-1,12 разів для $w=32$ біт, та 1,02-1,10 разів для $w=64$ біт).

7. Удосконалено методи арифметичних перетворень, що дозволяють підвищити швидкодію криптографічних перетворень у криптосистемі ECDSA (генерування особистого ключа – в 1,02-1,18 разів для $w=32$ біт, та 1,02-1,09

разів для $w=64$ біт; генерування відкритого ключа – в 1,01-1,06 разів для $w=32$ біт, та 1,01-1,05 разів для $w=64$ біт; створення підпису – в 1,01-1,10 разів для $w=32$ біт, та 1,01-1,05 разів для $w=64$ біт; перевірка підпису – в 1,01-1,03 разів для $w=32$ біт, та 1,01-1,29 разів для $w=64$ біт).

8. Удосконалено методи арифметичних перетворень, що дозволяють підвищити швидкодію криптографічних перетворень у криптосистемі RSA (генерування особистого ключа – в 1,01-5,94 разів для $w=32$ біт, та 1,01-2,64 разів для $w=64$ біт; створення підпису – в 1,48-10,0 разів для $w=32$ біт, та 1,01-1,24 разів для $w=64$ біт; перевірка підпису – в 1,01-1,39 разів для $w=32$ біт, та 1,01-1,2 разів для $w=64$ біт).

9. Розроблено та отримано п’ять патентів України на корисну модель, а саме «Спосіб множення цілих чисел» (Патент 111632, опубліковано 26.11.2016, бюлетень № 22), «Спосіб піднесення до квадрату цілих чисел» (Патент 118065, опубліковано 25.07.2017, бюлетень № 14), «Спосіб приведення за модулем цілих чисел» (Патент 118065, опубліковано 25.07.2017, бюлетень № 14), «Спосіб криптографічного перетворення інформації з використанням подовжених кодів» (Патент 123375, опубліковано 26.02.2018, бюлетень №4), «Спосіб криптографічного перетворення інформації з використанням укорочених кодів» (Патент 123379, опубліковано 26.02.2018, бюлетень №4);

10. Методи арифметичних перетворень реалізовано у бібліотеках криптографічних примітивів «Шифр+ v.2.1» системи криптографічного захисту інформації «Шифр-X.509» ТОВ «Сайфер ЛТД», що має дійсний позитивний експертний висновок Держспецзв'язку України від 16.05.2017 № 04/03/02-1674 (Акт № 22/17 від 04.08.2017 р.). Результати дисертаційних досліджень впроваджено у діяльність Кваліфікованого надавача електронних довірчих послуг Офісу Генерального прокурора України (Акт №18\10\2-8654-19 від 08.10.2020 р.) та Національного авіаційного університету (Акт від 25.09.2020 р.).

Особистий внесок здобувача. Основні положення і результати дисертаційної роботи, що виносяться до захисту, отримані автором самостійно. У роботах, написаних у співавторстві, автору належать: [1-4,12,15,17,20-22,25,31,33,34] – постановка завдання та розробка арифметичних перетворень з відкладеним переносом; [9,14,27] – розробка методів арифметичних перетворень з розпаралелюванням та відкладеним переносом; [5-8, 36] – розробка методики та обробка результатів експериментального дослідження; [7,29,35] – аналіз алгоритмів приведення цілих чисел за фіксованим модулем; [11,13,14] – дослідження та обґрунтування вимог до побудови сучасних криптосистем для захисту інформаційних ресурсів держави; [23,24] – формування рекомендацій щодо реалізації криптографічних перетворень. З робіт, що опубліковані у співавторстві, у дисертаційній роботі використовуються виключно результати, отримані особисто здобувачем.

Апробація результатів дисертації. Основні положення дисертаційної роботи доповідалися та обговорювалися на таких наукових конференціях: НТК «Захист інформації з обмеженим доступом та автоматизація її обробки» (Київ, 2011 р., 2012 р.), МНПК молодих учених та студентів «Політ. Сучасні проблеми науки» (Київ, 2011 р., 2020 р.), МНПК «Інфокомунікації – сучасність та майбутнє» (Одеса, 2011 р., 2013 р.), НТК студентів та молодих учених «Наукоємні технології» (Київ, 2011 р.), НТК студентів та аспірантів «Захист інформації з обмеженим доступом та автоматизація її обробки» (Київ, 2012 р.), Всесвітній конгрес «Авіація у XXI столітті» – «Безпека в авіації та космічні технології» (Київ, 2012 р., 2014 р.), МНПК «Проблеми і перспективи розвитку IT-індустрії» (Харків, 2013 р., 2014 р., 2015 р.), МНПК «Інтегровані інтелектуальні робототехнічні комплекси (ІПРТК)» (Київ, 2013 р.), МНПК «Інформаційні технології та комп'ютерна інженерія» (Вінниця, 2014 р.), НПК «Актуальні питання забезпечення кібербезпеки та захисту інформації» (Київ, 2015 р.), МНПК «Безпека інформації в інформаційно-телекомунікаційних системах» (Київ, 2013 р., 2015 р., 2016 р.), Міжнар. конф. «Actual Problems of

Unmanned Aerial Vehicles Developments (APUAVD)» (Київ, 2015 р.), Міжнар. конф. «Control, Automation and Systems» (Кьонджу, 2016 р.) та ін.

Публікації. Основні положення дисертації опубліковано у 36 наукових працях, у тому числі – 3 колективних монографіях, 16 наукових статях (5 – у міжнародних рецензованих виданнях, що входять до бази даних SCOPUS, 9 – у вітчизняних фахових наукових журналах та 2 – у інших наукових виданнях), 5 патентів України на корисну модель, а також 12 матеріалів і тез доповідей на конференціях.

Структура роботи та її обсяг. Дисертація складається з анотації, змісту, переліку умовних позначень, вступу, чотирьох розділів, загальних висновків, додатків, списку використаних джерел (в кінці кожного розділу основної частини дисертації) і має 165 сторінок основного тексту, 65 рисунків, 13 таблиць, 127 сторінок додатків. Список використаних джерел містить 219 найменувань і займає 20 сторінок. Загальний обсяг дисертаційної роботи – 326 сторінок.

РОЗДІЛ 1
СУЧАСНІ МЕТОДИ КРИПТОГРАФІЧНОГО ЗАХИСТУ НАЦІОНАЛЬНОЇ ІНФРАСТУКТРУРИ ВІДКРИТИХ КЛЮЧІВ

1.1. Особливості функціонування національної інфраструктури відкритих ключів

Сучасні глобалізаційні процеси, що відбуваються в світі, сприяють розвитку інформатизації усіх галузей господарства. У свою чергу, в сучасних людських і бізнес-відносинах все більшу роль починають відігравати інформаційні технології, які дозволяють підвищити оперативність обміну інформацією, підвищити швидкість прийняття рішень, поліпшити якість надаваних товарів і послуг, розширити сфери ведення бізнесу, освоювати нові ринки, витримувати конкуренцію.

В результаті обміну інформацією виникає необхідність забезпечення безпеки цієї інформації. Для побудови систем захисту інформації розглядають стандартну модель безпеки інформації [1-3], що складається з наступних категорій: конфіденційність, цілісність і доступність. У свою чергу, цей перелік, звичайно, розширюється ще категоріями неспростовність, автентичність та справжність [3-6]. Перераховані категорії, які входять в модель безпеки інформації, реалізуються наступними послугами: шифрування, електронний підпис, вироблення спільного секрету. Кожна з перерахованих послуг, окремо, не здатна повністю вирішити всі завдання забезпечення інформаційної безпеки, однак, застосування їх в комплексі дозволяє вирішити більшість з них.

Для автоматизації різних видів діяльності, виникає необхідність побудови системи електронного документообігу, для якого найбільший інтерес представляють категорії цілісність, автентичність та неспростовність [3]. Зазначені категорії успішно реалізуються за допомогою електронного підпису (ЕП). Сфери застосування ЕП досить різноманітні – це і системи електронного документообігу різного призначення [7-17], системи подачі звітності до контролюючих органів [18-24], системи інтернет-банкінгу [25-31],

портали державних послуг [32-40], державні реєстри [41-43], системи електронних торгів та публічних закупівель [44-51], та інші. Це продиктовано головною властивістю ЕП – він може бути використаний в якості аналога власноручного підпису або печатки на паперовому документі [52-53].

Найбільший інтерес представляють системи захищеного електронного документообігу, оскільки електронний документообіг [53-55] є пріоритетним напрямком державної політики України. У свою чергу, ЕП є реквізитом електронного документа, який використовується для ідентифікації автора-підписувача іншими суб'єктами електронного документообігу [52, 56-57]. Взаємодія всіх суб'єктів і об'єктів інформаційного обміну в межах країни має базуватися на довірі, яка досягається за рахунок функціонування технологічно і нормативно підтриманої інфраструктури відкритих ключів (ІВК) [58, 59]. Під терміном національна ІВК слід розуміти організаційно-технічну структуру, призначену для надання послуг ЕП в рамках всієї країни, що об’єднує центри сертифікації ключів, контролюючий орган, підписантів і користувачів в єдину систему, що працює за єдиними принципами і правилами. Ці принципи і правила являють собою сукупність норм і вимог, що викладені в декількох десятках нормативно-правових актів, а також національних і міжнародних стандартах. У світовій практиці найбільшу популярність отримала архітектура ІВК на основі сімейства стандартів X.509 (XPKI) [3, 58, 59].

З прийняттям 22 травня 2003 року Закону України «Про електронний цифровий підпис» та Закону України «Про електронні документи та електронний документообіг» поняття електронного підпису, електронного цифрового підпису та електронного документу ввійшли в правове поле, а роботи по створенню національної інфраструктури відкритих ключів набули системний та організований характер. Споживачам електронних послуг стали доступні наступні сервіси: створення електронного документу, накладення ЕЦП та електронної печатки, позначка часу [60].

З ратифікацією 16 вересня 2014 року Угоди про асоціацію між Україною та Європейським Союзом наша країна зобов’язалась гармонізувати власне

законодавство з законодавством ЄС [61], зокрема привести Закон України «Про електронний цифровий підпис» у відповідність до регламенту ЄС № 910/2014 «Про електронну ідентифікацію та довірчі послуги для електронних транзакцій в межах внутрішнього ринку та про скасування Директиви 1999/93/ЄС» (eIDAS) від 23 липня 2014 року, який визначає основні вимоги до надавачів послуг електронного підпису та електронної ідентифікації [62]. Оскільки Закон України «Про електронний цифровий підпис» перестав задовольняти вимоги сучасних технологій та потребам споживачів електронних сервісів, необхідно було розробити новий закон, метою якого стало реформування законодавства у сфері електронного цифрового підпису з урахуванням досвіду Європейського Союзу, розбудови єдиного простору довіри на основі системи електронних довірчих послуг, визнання в Україні електронних довірчих послуг, які надаються іноземними постачальниками електронних довірчих послуг, що забезпечить активний розвиток транскордонного співробітництва та інтеграцію України у світовий електронний інформаційний простір. Таким законом став Закон України «Про електронні довірчі послуги», який був прийнятий 5 жовтня 2017 року та набув чинності 7 листопада 2018 року. Основні новації цього закону [52]:

- розширення спектру послуг із використанням технологій електронного цифрового підпису;
- нормативне врегулювання питань електронної ідентифікації;
- впровадження міжнародної моделі оцінки відповідності надавачів послуг;
- спрощення адміністративних процедур виходу надавачів послуг на ринок.

Важливість прийняття Закону України «Про електронні довірчі послуги» полягає в наступному:

- виконання зобов'язань держави, визначених Угодою про асоціацію з Європейським Союзом;

- створення нормативної та технологічної основи розвитку цифрової економіки;
- створення умов для розвитку та функціонування сфери електронних довірчих послуг;
- вільний обіг електронних довірчих послуг в Україні, а також можливості вільного доступу до електронних
- довірчих послуг постачальниками електронних довірчих послуг, що провадять діяльність в інших державах;
- підвищення рівня довіри громадян до електронних послуг, у тому числі транскордонних;
- рівні можливості для доступу до електронних довірчих послуг, у тому числі для осіб з обмеженими можливостями;
- свобода договору у сфері електронних довірчих послуг;
- захист прав і законних інтересів користувачів електронних довірчих послуг;
- відповідність вимог до надання електронних довірчих послуг європейським та міжнародним стандартам;
- інтероперабельність та технологічна нейтральність національних технічних рішень, а також недопущення їх дискримінації;
- захист персональних даних, що обробляються під час надання електронних довірчих послуг;
- відкритість для інновацій у сфері електронних довірчих послуг.

Закон України «Про електронні довірчі послуги» розширив перелік послуг, що, в свою чергу, відкриває більше можливостей для держави, бізнесу та громадян [52, 63]:

- автентифікація веб-сайту;
- електронна ідентифікація;
- створення електронного документу;
- накладення кваліфікованого електронного підпису та печатки;

- позначка часу;
- зареєстрована електронна доставка;
- зберігання електронних підписів та печаток.

У зв'язку з цим, кількість сервісів, що підтримують електронний підпис збільшилась з 10 (як було в 2006 році) до більш ніж 150 (в 2019 році) [64-65]. Наразі, це системи державних послуг, системи електронної ідентифікації, системи електронної звітності, системи електронного документообігу (СЕДО), системи електронних закупівель, медичні інформаційні системи, банківські системи, та інші.

Як наслідок, кількість активних власників сертифікатів ЕП за останні 6 років збільшилась більше ніж у 5 разів [64].

Рік	Кількість власників
2013	1714317
2014	3165950
2015	3670000
2016	4011358
2017	5500000
2018	8823000
2019	10400000

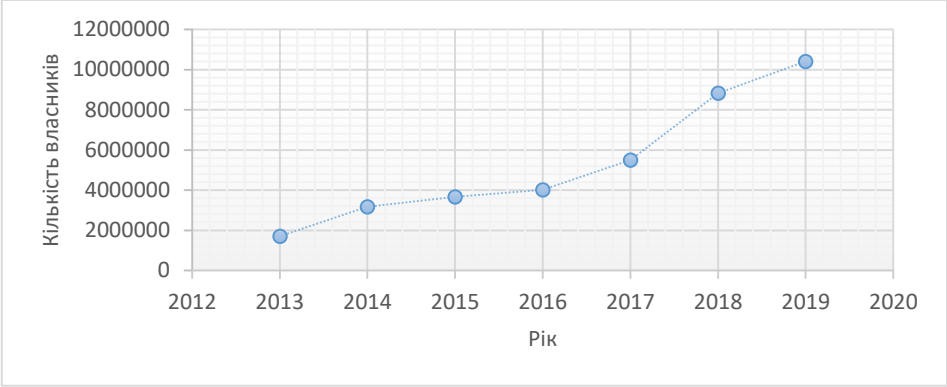


Рис. 1.1. Кількість активних власників сертифікатів ЕП за 2013-2019 рр.

В подальшому кількість користувачів ЕП буде тільки зростати. Все більше послуг буде надаватися в електронному вигляді з використанням ЕП як державою, так і комерційними організаціями. Цьому сприятиме, зокрема, поширення серед користувачів та подальший розвиток таких систем як MobileID [66-68], BankID [68], SmartID, інтегрованої системи електронної ідентифікації [70], онлайн-сервісу державних послуг «Дія» [71], та інших подібних сервісів.

Відповідно до Закону України «Про електронні довірчі послуги», до національної інфраструктури відкритих ключів входять [52]:

- користувачі електронних довірчих послуг (фізичні та юридичні особи);
- надавачі електронних довірчих послуг (юридичні особи);

- центральний засвідчувальний орган (ЦЗО);
- засвідчувальний центр (ЗЦ НБУ);
- контролюючий орган (Держспецзв'язку).

Контролюючий орган (Держспецзв'язку) виконує державний нагляд за дотриманням вимог законодавства у сфері електронних довірчих послуг.

ЦЗО відповідальний [52, 72], зокрема, за розробку нормативно-правових актів, норм, та стандартів у сфері електронних довірчих послуг, надання кваліфікованих електронних довірчих послуг надавачам електронних довірчих послуг з використанням самопідписаного сертифіката, ведення та підтримку в актуальному стані довірчого списку. Довірчий список містить інформацію з переліком надавачів електронних довірчих послуг та послуг, які вони надають, їх сертифікати та історія надавачів в часі. Довірчий список підписаний електронною печаткою ЦЗО (самопідписаний сертифікат), розміщений на веб-сайті ЦЗО та призначений для технологічного визнання електронних довірчих послуг та забезпечення транскордонної інтероперабельності. Довірчий список має бути завантажений до засобу перевірки електронного підпису, для аналізу даних, отримання сертифікатів надавача електронних довірчих послуг валідації підпису і видачі рішення про нього.

Засвідчувальний центр НБУ є регулятором в сфері електронного підпису та ідентифікації в банківській системі України [52, 56, 73-74], забезпечує кваліфікацію надавачів електронних довірчих послуг в ній та надає кваліфіковану електронну довірчу послугу формування, перевірки та підтвердження чинності кваліфікованого сертифіката електронного підпису чи печатки. Фактично забезпечує обслуговування окремої гілки сертифікатів центрів сертифікації банків. Засвідчувальний центр НБУ має власний самопідписаний сертифікат відкритого ключа, інформація про який внесена до довірчого списку.

Надавачами електронних довірчих послуг є державні та комерційні центри сертифікації ключів. Користувачі електронних довірчих послуг можуть мати ключі електронного підпису та шифрування в файловому

контейнері, на захищеному носії ключової інформації, смарт-карті, SIM-карті (MobileID), ID-карті чи на HSM.

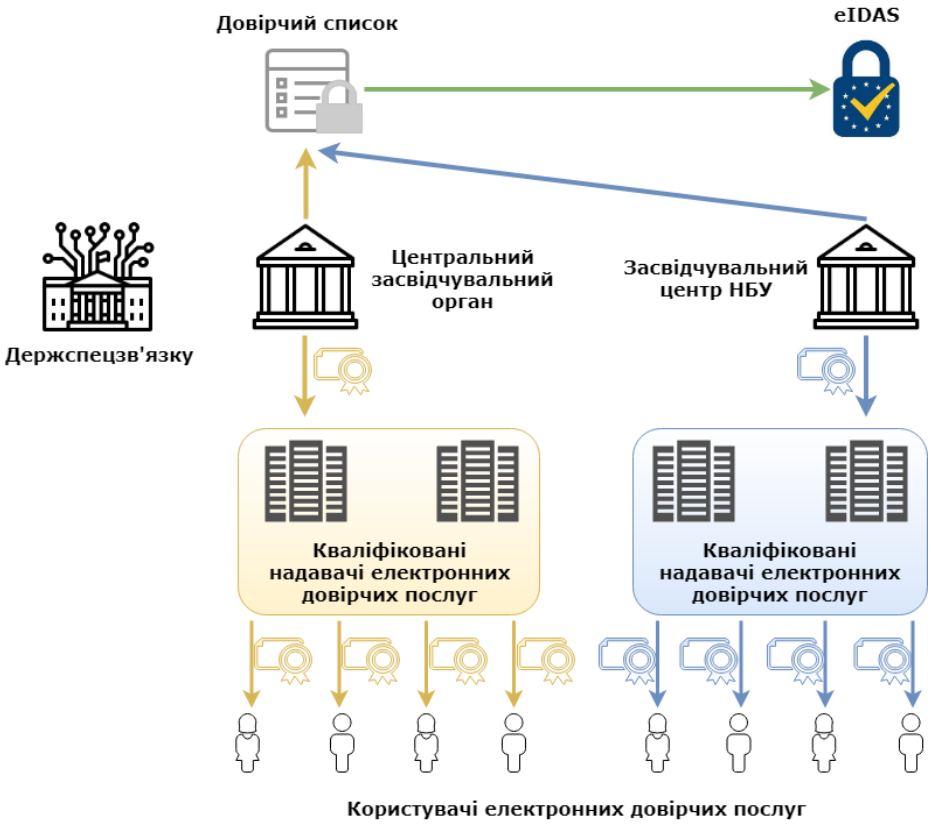


Рис. 1.1. Структура національної інфраструктури відкритих ключів

В національній інфраструктурі відкритих ключів при застосуванні кваліфікованого електронного підпису використовуються наступні алгоритми [75]:

- в межах країни з метою забезпечення електронного документообігу та електронної автентифікації осіб:
  - ДСТУ 4145-2002 з функцією гешування ГОСТ 34.311-95 (до 1 січня 2022);
  - ДСТУ 4145-2002 з функцією гешування ДСТУ 7564-2014 (обов’язкове використання з 1 січня 2022);
  - ECDSA (ДСТУ ISO/IEC 14888-3:2019) з функціями гешування sha256 або sha512 (FIPS PUB 180-4);
- для транскордонного співробітництва з будь-якою метою (додатково):
  - DSA (ДСТУ ETSI EN 119 312:2015);

- RSA (RFC 3447 PKCS#1: RSA Cryptography Specifications Version 2.1) з функцією гешування sha256 (FIPS PUB 180-4).

В банківській сфері, крім алгоритму ЕП на основі ДСТУ 4145-2002, широко використовується і ЕП на основі RSA [76-77].

Для взаємодії з ЦСК, що не входять до національної ІВК пропонується [78-79] використання шлюзу, побудованого на основі довірчих списків за допомогою кроссертифікації [80] сертифікатів ЦЗО України та цими ЦСК.

Для створення та перевірки ЕП на документах, користувачу необхідно отримати сертифікат відкритого ключа – сертифікувати ключі. Згідно XPKI, поняття ЕП нерозривно пов'язане з поняттям сертифікату відкритого ключа [3, 58, 81-82] – документ, що містить інформацію про відкритий ключ і його власника, який підписаний ключами видавця цього сертифіката (в даному випадку ЦСК) і призначений для підтвердження належності відкритого ключа його власнику.

На Рис. 1.3, показаний шлях сертифікації [82-83] для отримання сертифікату користувача, а також процес створення ЕП в національній ІВК [52].

Рис. 1.3. Шлях сертифікації сертифіката користувача

Відповідно до схеми на Рис. 1.3, повний шлях отримання сертифіката виглядає наступним чином:

1. ЦЗО за допомогою власного особистого ключа видає власний самопідписаний сертифікат, а також видає CRL і частковий CRL [72, 83].

2. ЦЗО формує сертифікат відкритих ключів надавачу довірчих послуг (ЦСК) [52].

3. ЦЗО вносить запис про виданий сертифікат, а також інформацію про надавача довірчих послуг до Довірчого списку та підписує його своєю електронною печаткою.

4. Далі ЦСК за допомогою свого особистого ключа видає CRL і частковий CRL, а також формує сертифікат користувачу.

5. Далі користувач, за допомогою власного особистого ключа, формує ЕП для документів.

Для перевірки ЕП використовуються сертифікати відкритих ключів, які доступні через сервіси ЦСК (web-сайт, LDAP-каталог, CMP-сервіс), або поширюються разом з ЕП (наприклад, в форматах ЕП CAdES, XAdES, PAdES, передбачена можливість зберігання сертифікатів або ланцюжків сертифікатів, OCSP відповідей, міток часу і навіть CRL). Також під час перевірки ЕП потрібне використання списку відкликаних сертифікатів (CRL) або сервісу OCSP [80, 83-84], для визначення статусу сертифікатів з ланцюжка сертифікатів підписанта.

В національній ІВК регламентовано використання наступних форматів електронного підпису [86-87]: базовий (CAdES-BES), з позначкою часу від ЕП (CAdES-T), з посиланнями на повні дані для перевірки (CAdES-C), з повними даними для перевірки (CAdES-X-Long). Найбільш поширеними є базовий підпис (CAdES-BES), через простоту та можливість створення без доступу до on-line сервісів надавача електронних довірчих послуг, та підпис з повними даними для перевірки (CAdES-X-Long), який забезпечує можливість встановлення дійсності ЕП у довгостроковому періоді (після закінчення строку чинності сертифіката підписувача). Базовий підпис включає набір обов’язкових атрибутів, цифровий підпис, обчислений за електронними даними та набором атрибутів, електронні дані, стосовно яких здійснюється

обчислення цифрового підпису. Підпис з повними даними для перевірки додатково включає позначку часу відносно ЕП, всі необхідні сертифікати та інформацію про статус кожного з них.

Створення ЕП в форматі CAdES-X-Long на документі, відповідно до схеми на Рис. 1.3, потребує формування двох запитів до сервера OCSP та одного запиту до сервера TSA, у відповідь на які формуються три відповіді, що містять ЕП. У випадку, якщо створюється ЕП на документ, що вже містить ЕП, перший підпис необхідно спочатку перевірити.

Перевірка ЕП в форматі CAdES-X-Long на документі, відповідно до схеми на Рис. 1.3, потребує обчислення 9 значень геш-функції та 9 ЕП:

1. перевірка ЕП (сертифікат підписанта);
2. перевірка ЕП OCSP-відповіді (сертифікат підписанта);
3. перевірка ЕП позначки часу (документ);
4. перевірка ЕП (сертифікат КНЕДП);
5. перевірка ЕП OCSP-відповіді (сертифікат КНЕДП);
6. перевірка ЕП (сертифікат ЦЗО);
7. перевірка ЕП (CRL ЦЗО);
8. перевірка ЕП (Довірчий список);
9. перевірка ЕП (сертифікат печатки ЦЗО).

При чому, якщо довірений ланцюжок сертифікатів вже сформований та один раз перевірений (сертифікат печатки ЦЗО, сертифікат ЦЗО, сертифікат КНЕДП та Довірчий список), то можна оптимізувати процес перевірки ЕП на документі – необхідно виконати обчислення лише 5 значень геш-функції та 5 ЕП.

В свою чергу, якщо необхідно перевірити ЕП, що створений ключами, які видані в іншій ІВК, то необхідно використовувати шлюз кроссертифікації [79]. Це в свою чергу подовжує ланцюжок сертифікатів і збільшує кількість перевірок.

З цього випливає, що при великій кількості ЕП на документах, які необхідно перевірити, час виконання створення та перевірки ЕП стає критичним, а навантаження на інфраструктуру ІТС ЦСК (зокрема на сервіси OCSP та TSA) досить значне. В зв’язку з цим, в деяких випадках ЦСК стає

об’єктом критичної інфраструктури – зупинка або некоректна його робота, затримка обробки запитів може вплинути на надання банківських, державних, медичних послуг, роботу з єдиними державними реєстрами, комерційну діяльність. Наприклад, згідно з офіційною статистикою [87], станом на 1 січня 2019 року в Україні зареєстровано приблизно 2,3 мільйона суб'єктів господарської діяльності, які щодня користуються послугами банків, обмінюються електронними документами і подають звітність до контролюючих органів. Системи подачі звітності в пікові періоди (подача річних, квартальних звітів, тощо) щодня обробляють десятки тисяч документів з 1-3 ЕП. У банківській сфері згідно зі статистикою СЕП НБУ [88] в 2019 році оброблялись 1,5 мільйони міжбанківських платіжних документів в день (при цьому внутрішньобанківських платіжних документів в рази більше). В самих банках, автоматизовані банківські системи при закритті банківського дня, оброблять десятки, а то і сотні тисяч документів з двома або трьома ЕП, що були сформовані співробітниками і клієнтами банку протягом дня. Всі ЕП повинні бути перевірені, а документи оброблені в стислі терміни.

Тому, розробка методів забезпечення оперативності обслуговування звернень до сервісів ЦСК (що працюють щодня цілодобово, в умовах нерівномірного навантаження та атак на відмову в обслуговуванні), за рахунок збільшення продуктивності криптографічних перетворень [89, 90, 92] є *актуальною науково-практичною задачею*.

1.2. Використання цілих чисел в криптографічних перетвореннях та методи їх представлення

1.2.1. Арифметичні перетворення в криптографічних застосуваннях

Криптографічні перетворення з відкритим ключем і арифметичні перетворення, які лежать в їх основі, варто розглядати у вигляді декількох рівнів. Зокрема, для криптографічних перетворень з відкритим ключем на еліптичних кривих [91] можна розглянути наступну ієрархію операцій [92] (Рис. 1.4):

- власне, криптографічні перетворення (шифрування і розшифрування, створення і перевірка електронного цифрового підпису, вироблення спільного секрету);
- арифметика в групі точок еліптичної кривої (додавання точок, подвоєння точки, скалярне множення точок, стиснення і відновлення точки, перевірка приналежності точки еліптичної кривої, генерація випадкової точки);
- арифметика в полі $GF(p)$ і $GF(2^m)$ (додавання, віднімання, порівняння, множення, піднесення до квадрату, приведення за модулем, ділення, інвертування, зсув вліво і вправо, піднесення до степеню, визначення квадратного кореня і ін.);
- команди (інструкції) CPU (mov, mul, shr, shl, xor, add, sub та ін.).

<table>
<tr><td>Криптографічні перетворення</td><td colspan="6">Шифрування / Розшифрування</td><td colspan="2">Електронний підпис</td><td colspan="2">Спільний секрет</td></tr>
<tr><td rowspan="2">Арифметика в групі точок еліптичної кривої</td><td colspan="2">Додавання</td><td colspan="2">Подвоєння</td><td colspan="2">Скалярне множенне</td><td colspan="4">Стиснення / Відновлення</td></tr>
<tr><td colspan="6">Перевірка належності точки кривій</td><td colspan="4">Генерація випадкової точки</td></tr>
<tr><td rowspan="2">Арифметика в полі GF(p) та GF(2^m)</td><td colspan="2">Додавання</td><td colspan="2">Віднімання</td><td colspan="2">Порівняння</td><td colspan="2">Множення</td><td colspan="2">Піднесення до квадрату</td><td colspan="2">Приведення за модулем</td></tr>
<tr><td>Ділення</td><td colspan="2">Інвертування</td><td colspan="3">Зсув (вліво, вправо)</td><td colspan="2">Піднесення до степеню</td><td colspan="2">Квадратний корінь</td><td>та інші</td></tr>
<tr><td>Команди CPU</td><td colspan="12">mov, mul, shr, shl, xor, add, sub ...</td></tr>
</table>

Рис. 1.4. Перетворення в криптосистемах на еліптичних кривих

Операції по створенню і перевірці ЕП ґрунтуються на різних математичних апаратах:

- *ДСТУ 4145-2002* – базується на перетвореннях в групі точок еліптичної кривої над двійковим полем GF(2^m), а також над елементами простого поля GF(p) [94, 98];
- *ECDSA* – базується на перетвореннях в групі точок еліптичної кривої над двійковим полем GF(2^m) або GF(p), а також операціях над елементами простого поля GF(p) [95];
- *DSA* – базується на перетвореннях в групі елементів двійкового поля GF(2^m) або простого поля GF(p), а також операціях над елементами простого поля GF(p) [96];
- *RSA* – базується на перетвореннях в кільці цілих чисел [97].

Таким чином, арифметичні перетворення над цілими числами присутні серед усіх вище перерахованих схем ЕП.

Аналіз криптографічних перетворень ДСТУ 4145-2002, ECDSA, DSA, RSA та інших, дозволяє систематизувати результати та зрозуміти, в яких перетвореннях використовуються ті чи інші арифметичні перетворення (Табл. 1.1).

Таблиця 1.1

Використання арифметичних операцій в криптографічних перетвореннях

№	*Операція*	*Використання*
1.	Додавання	Використовуються у багатьох криптографічних перетвореннях (наприклад, в IEEE P1363-2000 – DSA, ECDSA, DH1, DH2, MQV1, MQV2, ECDH1, ECDH2, ECMQV1, ECMQV2, використовується при генеруванні випадкової точки і визначенні належності точки кривої [99]), а також лежить в основі інших арифметичних операцій).
2.	Віднімання	
3.	Зсув вліво	Використовуються при генерації випадкової точки ЕК, стиснення і відновлення точки ЕК з стисненого стану [100]. Також присутні в операціях інвертування, обчисленні квадратного кореня і при вирішенні системи квадратних рівнянь.
4.	Зсув вправо	
5.	Множення	Стиснення і відновлення точки ЕК з стисненого стану, генерації випадкової точки ЕК, визначенні приналежності точки кривої [99-100].
6.	Піднесення до квадрату	Додавання і подвоєння точок ЕК, стиснення і відновлення точки ЕК з стисненого стану, генерації випадкової точки ЕК, визначенні приналежності точки кривої [101-102, 105], а також в перетворення піднесення до степеню.
7.	Приведення за модулем	У всіх операціях, де передбачається переповнення: зсув вліво, додавання і віднімання, множення, піднесення до квадрату, мультиплікативне інвертування, а також, всі перетворення, де використовуються перераховані вище перетворення [102-105]. Зокрема, приведення за модулем використовуються в алгоритмах RSA і Діффі-Хелмана. Також, це фундаментальна операція криптографічних алгоритмів на ЕК.
8.	Ділення	Обчислення квадратного кореня (для деяких простих модулів); генерація загальносистемних параметрів; обчислення випадкової точки ЕК; відновлення точки еліптичної кривої з стисненого стану [94, 103, 105-107].
9.	Порівняння	Накладання та перевірка цифрового підпису, стиснення і відновлення точки ЕК з стисненого стану [104-106].
10.	Інвертування	Додавання і подвоєння точок, множення точок (в проективних координатах), накладення і перевірка цифрового підпису, стиснення і відновлення точки ЕК з стисненого стану [102-103, 106].
11.	Піднесення до степеню	Використовується безпосередньо в RSA, в схемах на основі перетворень в групі елементів поля чисел $GF(p)$ по типу DSA, DH1, DH2, MQV1, MQV2 і т.п. [99, 101]. Крім того, використовується в перетвореннях на ЕК та ГЕК: визначення квадратного кореня, розв'язання квадратних рівнянь, обчислення випадкової точки, генерація загальносистемних параметрів [100, 102, 105].
12.	Визначення квадратного кореня	Генерація загальносистемних параметрів; обчислення випадкової точки ЕК; відновлення точки ЕК з стисненого стану [94, 105-107] (IEEE P1363-2000, ГОСТ 34.10-2018, ДСТУ 4145-2002).

Що стосується самих арифметичних операцій, то операції додавання та віднімання складають основу для інших арифметичних операцій, наприклад, множення і зсувів. Піднесення до квадрату є окремим випадком множення, коли обидва множника рівні. Методи додавання, порівняння, множення і зсуву лежать в основі методів приведення за модулем і ділення. Методи інвертування використовує перетворення додавання і віднімання, зсуву, приведення за модулем, а метод піднесення до степеню представляє собою послідовне множення і піднесення до квадрату. В основі методу визначення квадратного кореня лежить операція ділення, тому використовуються перетворення піднесення до квадрату і до степеню, множення, порівняння, зсуви, додавання і віднімання.

При програмній або апаратній реалізації операцій над цілими числами істотним є представлення чисел та можливості їх розпаралелювання [108].

1.2.2. Числа криптографічної довжини

У переважній більшості сучасних криптографічних систем виконуються перетворення з цілими числами. Великі цілі числа (не обов'язково прості) виступають в якості ключів, для виконання криптографічних перетворень.

Для досягнення бажаного рівня безпеки, в залежності від криптосистеми, використовуються цілі числа розміром від кількох сотень до кількох тисяч біт. З часом, для підтримки потрібного рівня безпеки, розмір повинен збільшуватися [109-111]. Так, наприклад, для криптосистеми RSA, рекомендований розмір ключів збільшився з 512 біт в 80-х і 90-х роках XX століття до 2048 на поточний момент часу (при цьому, зараз широко використовуються ключі розміром 3072 і 4096 біт) [110, 112-114].

Безліч науковців і організацій проводять дослідження, на підставі яких видають обґрунтування і рекомендації, щодо мінімальних довжин ключів, необхідних для забезпечення безпеки експлуатованих систем [110, 112-117]. Так, на даний момент, Національним інститутом стандартів і технологій США рекомендуються мінімальні значення довжин ключів [118]:

- для симетричних криптосистем – не нижче 112 біт;

- для криптосистем на основі факторизації цілих чисел – не нижче 2048 біт;
- для криптосистем на основі дискретного логарифмування – не нижче 2048 біт;
- для криптосистем на основі еліптичних кривих – не нижче 224 біт.

Проте, для довготривалого використання, значення довжин ключів мають бути більшими [118].

Для виконання перетворень над такими великими числами потрібен особливий підхід у виборі методів арифметичних операцій.

1.2.3. Представлення цілих чисел

Крім розміру цілих чисел, не менш важливим є їхнє представлення. Відомо безліч представлень цілих чисел, які можна умовно розділити на три групи [119, 120]:

- позиційні (значення кожного числового знаку (числа, слова) в записі числа залежить від його позиції (розряду));
- непозиційні (величина числа не залежить від положення цифр (чисел, слів) в записі);
- змішані.

У сучасній криптографії і обробці сигналів для виконання виключно специфічних задач використовуються деякі представлення цілих чисел [123-128].

Окремо варто зупинитися на наступних представленнях:

- двійкове представлення (binary form) [120, 124];
- двійкове представлення зі знаком (signed-digit binary) [127, 129-131];
- несуміжна форма (NAF) [123, 132];
- представлення в системі залишкових класів (Residue Number System) [123, 133-135];
- представлення в частотній області (frequency domain) [123, 136].

1. Двійкове представлення (binary form)

Представлення в позиційній системі числення з основою 2, яке широко використовується практично в усіх сучасних обчислювальних пристроях. Ціле позитивне число A в двійковому представлені має вигляд:

$$A = a_{l-1}2^{l-1} + a_{l-2}2^{l-2} + ... + a_1 2 + a_0 = \sum_{i=0}^{l-1} a_i 2^i ,$$

де, $a_i \in \{0,1\}$, а l – двійкова довжина числа.

У випадку, коли необхідно представити ціле число у вигляді машинних слів (масив машинних слів двійкової довжини w біт кожне):

$$A = a_{n-1}2^{(n-1)w} + a_{n-2}2^{(n-2)w} + \cdots + a_1 2^w + a_0 ,$$

де $n = \left\lceil \frac{l}{w} \right\rceil$ – число машинних слів необхідних для представлення числа двійкової довжини l; a_j – машинні слова довжини w біт.

2. Двійкове представлення зі знаком (signed-digit binary)

На відміну від двійкового представлення, представлення цілого числа зі знаком не унікальне. Число в двійковому представленні зі знаком може мати вигляд:

$$A = \sum_{i=0}^{l} a_i 2^i ,$$

де $a_i \in \{-1,0,1\}$, а l – двійкова довжина числа. Це представлення надлишкове. Наприклад, число 3 може бути представлено як $(011)_2$ або $(10\bar{1})_2$, де $\bar{1} = -1$. Таке представлення часто використовується в перетворенні експоненти для піднесення до степені за модулем (наприклад, в RSA).

3. Несуміжна форма (non-adjacent form, NAF)

NAF є різновидом двійкового представлення зі знаком, за умови, що в даному поданні числа не зустрічається два суміжних ненульових елемента. Кожне ціле число k має унікальне представлення в NAF, яке має мінімальну вагу будь-якого двійкового представлення числа k зі знаком.

Представлення цілого числа в двійковій формі $B = b_{l-1}2^{l-1} + b_{l-2}2^{l-2} + \cdots + b_1 2 + b_0$ в k-значній NAF-формі матиме вигляд:

$$B = a_{n-1}2^{(k-1)} + a_{n-2}2^{(k-2)} + \cdots + a_1 2 + a_0,$$

де k – число коефіцієнтів a_j необхідних для представлення двійкового числа довжиною l; b_i – двійкові коефіцієнти; $a_j \in \{-1, 0, 1\}$ – коефіцієнти.

Використання представлення чисел в NAF, дозволяє зменшити кількість множень, необхідних для виконання піднесення в ступінь. Для RSA використання NAF не ефективно, оскільки передбачається виконання перетворення ділення, яка має високу складність. Проте, в криптографії на еліптичних кривих це представлення широко використовується. Представлення NAF має безліч модифікацій. Більш докладно про них і про алгоритми перетворення чисел у NAF-форму, описано в роботі [128].

4. Представлення в системі залишкових класів (Residue Number System)

Представлення в непозиційній системі числення, яке засноване на понятті віднімання та китайської теореми про залишки, ефективне для дуже великих цілих чисел [123, 133-135].

Нехай $\langle x \rangle_a$ – RNS представлення цілого числа x:

$$\langle x \rangle_a = \left(x[a_1], x[a_2], ..., x[a_n] \right),$$

де $x[a_i] = x \bmod a_i$.

Множина $a = \{a_1, a_2, ..., a_n\}$ називається базисом, розміру n. Необхідно виконання умови $\gcd(a_i, a_j) = 1$ для $i \neq j$. З китайської теореми про залишки випливає, що число x, яке задовольняє вимогу $0 \le x < \prod_{i=1}^{n} a_i$ однозначно представлено в $\langle x \rangle_a$.

Відомою перевагою представлення в системі залишкових класів є те, що арифметичні перетворення (додавання, віднімання, множення, ділення) виконуються покомпонентно, якщо про результат відомо, що він є цілочисельним, а розмір компонентів набагато менше, ніж розмір модуля. Також представлення в системі залишкових класів дозволяє застосовувати технології розпаралелювання. Недоліками ж такого представлення є

відсутність ефективних методів порівняння чисел, повільні реалізації перетворення числа з двійкового представлення в представлення в системі залишкових класів і навпаки, складність виконання методу розподілу і труднощі у виявленні переповнення.

5. Представлення в частотній області (frequency domain)

Представлення в частотній області найчастіше використовується в цифровій обробці сигналів і орієнтоване переважно на апаратну реалізацію [123, 136-138]. Дане представлення також використовується в криптографії на еліптичних кривих [137, 139]. Із застосуванням дискретного перетворення Фур'є [136], це представлення використовується для підвищення продуктивності скалярного множення точок на еліптичній кривій [137], множення поліномів в полях Галуа [139] і множення за модулем чисел [138].

6. Порівняння способів представлення цілих чисел

Наведений вище список представлень цілих чисел не кінцевий, але дозволяє продемонструвати різноманітність підходів і може бути істотно розширений. Найбільш простим і універсальним з розглянутих способів є двійкове представлення цілих чисел, при цьому це представлення має як переваги, так і недоліки. Зокрема, двійкове представлення не завжди найефективніше і продуктивне. Решта представлень мають свою обмежену специфічну сферу застосування у певних криптосистемах.

Таблиця 1.2

Узагальнення порівняльного аналізу форм представлення цілих чисел

Назва	**Використання**	**Особливості**	**Переваги**	**Недоліки**
1	2	3	4	5
Двійкове представлення (binary form)	Універсальне представлення	Немає	Простота та широка область використання	Потрібно враховувати перенесення
Двійкове представлення зі знаком (signed-digit binary form)	Піднесення до степеня за модулем	Для запису експоненти	Дозволяє робити менше операцій множення, збільшує швидкість піднесення в ступінь по модулю	Не унікальне, ціле число може мати більше ніж одне представлення, надмірність

Продовження Табл. 1.2

1	2	3	4	5
Несуміжна форма (NAF)	У криптографії на еліптичних кривих	Розріджене двійкове представлення зі знаком	Мінімальне значення ваги Хеммінга для скаляра	Висока складність перетворення ділення, не застосовується в RSA
Представлення в системі залишкових класів (Residue Number System)	Використовується для RSA, для реалізацій на FPGA	Ґрунтується на китайській теоремі про залишки	Можливість не враховувати перенесення для операцій додавання, віднімання і множення	Великі накладні витрати при конвертації з двійкового представлення і назад, висока складність перетворення ділення
Представлення в частотній формі (frequency domain)	Криптографія на еліптичних кривих	Ґрунтується на дискретному перетворенні Фур'є	В деяких випадках при множенні за модулем	Висока складність

1.3. Аналіз арифметичних перетворень з цілими числами в двійковому представлені

В сучасній обчислювальній техніці цілі числа часто представляються в двійковій системі числення [120, 123]. Для роботи з цілими числами в такому представленні широко використовуються такі арифметичні перетворення як додавання, віднімання, зсув вліво і вправо, множення, піднесення в квадрат, приведення по модулю, ділення [121, 122]. Крім того, досить часто також використовується операція порівняння [140], яка є предикативною, логічною операцією.

Розглянемо існуючі методи арифметичних операцій над цілими числами в двійковому представленні для порівняння із запропонованими в цій роботі операціями над числами з відкладеним переносом. Далі більш докладно розглянуті ці методи і представлена оцінка їх обчислювальної складності. Для оцінки обчислювальної складності методів традиційно використовується підрахунок числа елементарних операцій, які необхідно виконати, при цьому не враховується можливість виконання самих операцій паралельно за рахунок

суперскалярності процесора, ні можливість виконання окремих функцій паралельно, не можливості кешу [141].

1.3.1. Додавання

Метод 1.1. *Метод додавання цілих чисел в двійковому представленні*

Розглянемо метод додавання двох цілих невід'ємних чисел [120, 124]. На вхід подаються два цілих числа $a, b \in Z$ в двійковому вигляді, які представлені як послідовності з n машинних слів, розміром w біт кожне. В циклі з інтервалом $[\overline{0, n-1}]$ виконується складання відповідних машинних слів двох чисел. На виході отримуємо результат – двійкове число $d \in Z$.

Вхід: $a, b \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = w$, $i = \overline{0, n-1}$, w – довжина машинного слова. *Вихід*: $d = a + b$, $d \in Z$, $d = \{d_n, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n}$.	
1. $c \leftarrow 0$. 2. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 2.1. $t^{(w)} \leftarrow a_i^{(w)} + b_i^{(w)}$. 2.2. $tc^{(w)} \leftarrow t^{(w)} + c$.	2.3. $c \leftarrow \left(t^{(w)} < a_i^{(w)}\right) \mid \left(tc^{(w)} < t^{(w)}\right)$. 2.4. $d_i^{(w)} \leftarrow tc^{(w)}$. 3. $d_n^{(w)} \leftarrow c$. 4. Return (d).

Рис. 1.5. Псевдокод методу додавання цілих чисел в двійковому представленні

Обчислювальна складність методу додавання цілих чисел в двійковій формі:

$$I_{add}\left(A_{1.1}\right) = 2I_{a\text{sgn}}^{w} + n\left(2I_{add}^{w} + 2I_{cmp}^{w} + I_{or}^{w} + 4I_{a\text{sgn}}^{w}\right).$$

1.3.2. Віднімання

Метод 1.2. *Метод віднімання цілих чисел в двійковому представленні*

Розглянемо метод віднімання двох цілих чисел [120, 124]. На вхід подаються два цілих числа $a, b \in Z$ в двійковому вигляді, які представлені як послідовності з n машинних слів, розміром w біт кожне. В циклі з інтервалом $[\overline{0, n-1}]$ виконується віднімання відповідних машинних слів двох чисел. На виході отримуємо результат – двійкове число $d \in Z$.

Вхід: $a, b \in Z$, $a, \le b$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = w$, $i = \overline{0, n-1}$, w – довжина машинного слова. *Вихід*: $d = a - b$, $d \in Z$, $d = \{d_n, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n}$.	
1. $c \leftarrow 0$. 2. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 2.1. $t^{(w)} \leftarrow a_i^{(w)} - b_i^{(w)}$. 2.2. $tc^{(w)} \leftarrow t^{(w)} - c$.	2.3. $c \leftarrow \left(t^{(w)} > a_i^{(w)}\right) \mid \left(tc^{(w)} > t^{(w)}\right)$. 2.4. $d_i^{(w)} \leftarrow tc^{(w)}$. 3. $d_n^{(w)} \leftarrow d_n^{(w)} - c$. 4. Return (d).

Рис. 1.6. Псевдокод методу віднімання цілих чисел в двійковому представленні

Обчислювальна складність методу віднімання двох чисел в двійковій формі:

$$I_{sub}\left(A_{1.2}\right) = 2I_{a\text{sgn}}^{w} + n\left(2I_{sub}^{w} + 2I_{cmp}^{w} + I_{or}^{w} + 4I_{a\text{sgn}}^{w}\right).$$

1.3.3. Зсув вліво

Метод 1.3. *Метод зсуву вліво цілих чисел в двійковому представленні на один біт*

Розглянемо метод зсуву цілого числа вліво на один біт [100]. На вхід подається ціле число $a \in Z$ в двійковому вигляді, яке представлено як послідовність з n машинних слів, розміром w біт кожне. В циклі з інтервалом $[\overline{n-1, 0})$ виконується зсув вліво кожного машинного слова числа на один біт. На виході отримуємо результат – двійкове число $d \in Z$.

Вхід: $a \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, $i = \overline{0, n-1}$, w – довжина машинного слова. *Вихід*: $d = (a << 1)$, $d \in Z$, $d = \{d_n, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n}$.	
1. $h^{(w)} \leftarrow \left(a_{n-1}^{(w)} << 1\right)$. 2. $d_n^{(w)} \leftarrow \left(a_{n-1}^{(w)} >> (w-1)\right)$. 3. For $i \leftarrow n-1$, $i <> 0$, $i{-}{-}$. 3.1. $l^{(w)} \leftarrow a_{i-1}^{(w)}$.	3.2. $d_i^{(w)} \leftarrow h^{(w)} \mid \left(l^{(w)} >> (w-1)\right)$. 3.3. $h^{(w)} \leftarrow \left(l^{(w)} << 1\right)$. 4. $d_0^{(w)} \leftarrow h^{(w)}$. 5. Return (d).

Рис. 1.7. Псевдокод методу зсуву вліво цілих чисел в двійковому представленні на один біт

Обчислювальна складність методу зсуву вліво:

$$I_{shl}\left(A_{1.3}\right) = 3I_{a\text{sgn}}^{w} + 2I_{sh}^{w} + n\left(2I_{sh}^{w} + I_{or}^{w} + 2I_{sub}^{w} + 3I_{a\text{sgn}}^{w}\right).$$

1.3.4. Зсув вправо

Метод 1.4. *Метод зсуву вправо цілого числа в двійковому представленні на один біт*

Розглянемо метод зсуву цілого числа вправо на один біт [100]. На вхід подається ціле число $a \in Z$ в двійковому вигляді, яке представлено як послідовність з n машинних слів, розміром w біт кожне. В циклі з інтервалом $[\overline{0, n-1})$ виконується зсув вправо кожного машинного слова числа a на один біт. На виході отримуємо результат – двійкове число $d \in Z$.

Вхід: $a \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, $i = \overline{0, n-1}$, $w-$ довжина машинного слова. *Вихід*: $d = (a >> 1)$, $d \in Z$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n} - 1$.	
1. $h^{(w)} \leftarrow a_0^{(w)}$. 2. $l^{(w)} \leftarrow \left(h^{(w)} >> 1\right)$. 3. For $i \leftarrow 0$, $i < n-1$, $i{+}{+}$. 3.1. $h^{(w)} \leftarrow a_{i+1}^{(w)}$.	3.2. $d_i^{(w)} \leftarrow l^{(w)} \mid \left(h^{(w)} << (w-1)\right)$. 3.3. $l^{(w)} \leftarrow \left(h^{(w)} >> 1\right)$. 4. $d_{n-1}^{(w)} \leftarrow l^{(w)}$. 5. Return (d).

Рис. 1.8. Псевдокод методу зсуву вправо цілих чисел в двійковому представленні на один біт

Обчислювальна складність методу зсуву вправо:

$$I_{shr}(A_{1.4}) = 3I_{a\,\mathrm{sgn}}^{w} + I_{sh}^{w} + n\left(2I_{sh}^{w} + I_{or}^{w} + 3I_{a\,\mathrm{sgn}}^{w} + I_{sub}^{w}\right).$$

1.3.5. Множення

Операція множення займає провідне місце серед операцій в кільцях і полях чисел, які складають основу для криптографічних перетворень з відкритим ключем [102, 124, 142-143]. При цьому, множення є досить трудомісткою операцією [118, 144]. Сьогодні відомі такі методи множення цілих чисел, які використовуються в криптографії:

- множення в стовпчик [102, 140, 145-147];
- Карацуби-Офмана [102, 145-146, 148];
- Тоома-Кука [145-146, 149];
- Шенхаге-Штрассена [145, 150];
- Comba [151-152];
- Фюрера (розвиток методу Шенхаге – Штрассена) [153].

Метод 1.5. *Метод множення двох цілих чисел в двійковому представленні в стовпчик*

Широко відомий і часто використовуваний метод множення двох цілих чисел, який базується на основі множення в стовпчик. Він може бути використаний для множення двох чисел довільного розміру. На вхід подаються два двійкові цілих числа $a=\{a_{n-1},\ldots,a_i,\ldots,a_1,a_0\}$ і $b=\{b_{m-1},\ldots,b_i,\ldots,b_1,b_0\}$, які представлені як послідовності машинних слів ($n=\log_{2^w} a$, $m=\log_{2^w} b$), розміром w біт кожне. Далі частина першого множника a_i послідовно, рядками множиться на всі частини другого множника b_j. Проміжний результат множення a_i і b_j формується з урахуванням всіх переносів. Кожен проміжний результат зміщується вліво на w біт, а потім всі проміжні результати послідовно підсумовуються. На виході отримуємо результат – двійкове число $d \in Z$, розміром $n+m$ машинних слів.

Вхід: $a,b \in Z$, $a=\{a_{n-1},\ldots,a_i,\ldots,a_1,a_0\}$, $b=\{b_{m-1},\ldots,b_i,\ldots,b_1,b_0\}$, $\log_2 a_i=\log_2 b_i=w$, w–довжина машинного слова. *Вихід*: $d=a\cdot b$, $d\in Z$, $d=\{d_{n+m-1},\ldots,d_i,\ldots,d_1,d_0\}$, $\log_2 d_i=w$.	
1. For $i\leftarrow 0$, $i\le n+m-1$, $i++$. 1.1. $d_i^{(w)}\leftarrow 0$; 2. For $i\leftarrow 0$, $i\le n-1$, $i++$. 2.1. $c^{(w)}\leftarrow 0$; 2.2. For $j\leftarrow 0$, $j\le m-1$, $j++$	2.2.1. $(uv)^{(2w)}\leftarrow d_{i+j}^{(w)}+a_i^{(w)}\cdot b_j^{(w)}+c^{(w)}$. 2.2.2. $d_{i+j}^{(w)}\leftarrow v^{(w)}$, $c^{(w)}\leftarrow u^{(w)}$. 2.3. $d_{i+n+1}^{(w)}\leftarrow u^{(w)}$. 3. Return (d).

Рис. 1.9. Псевдокод методу множення двох цілих чисел в двійковому представленні в стовпчик

Обчислювальна складність методу множення в стовпчик двох цілих чисел:

$$I_{mul}(A_{1.5})=3nI_{a\text{sgn}}^{(w)}+mI_{a\text{sgn}}^{(w)}+nm(I_{a\text{sgn}}^{2w}+2I_{a\text{sgn}}^{w}+2I_{add}^{w}+I_{mul}^{w}).$$

Однак, даний метод множення має велику обчислювальну складність, тому при множенні великих чисел він буде повільним. Через це для множення чисел криптографічної довжини, використовується множення методом Карацуби або методом Comba. Аналіз публікацій [144, 154], дозволив виділити

метод множення Comba [151], який показав кращі результати тестів швидкодії програмної реалізації на сучасних платформах [152, 155-158] для довжин чисел, що використовуються в криптографії.

Метод 1.6. *Метод Comba множення двох цілих чисел в двійковому представленні*

При використанні методу множення Comba [151, 154, 158], на вхід подаються два числа a і b, які представлені як набір машинних слів $a=(a_n,...,a_i,...,a_1,a_0)$ і $b=(b_n,...,b_j,...,b_1,b_0)$, де n – кількість машинних слів, необхідних для представлення числа ($n=\log_{2^w} a$), w – розмір машинного слова в бітах. На виході отримується результат $c=a\cdot b$ розміром $2n$ машинних слів. Даний спосіб реалізує підхід множення в стовпчик з невеликою різницею: множиться частина множника a_i ($i=\overline{0,n}$) на всі частини другого множника b_j ($j=\overline{1,n}$), в порядку виконання умови $(i+j==k)$ в стовпчиках, а не послідовно по рядках. Основу зазначеного способу множення складають два цикли формування результату множення. Перший цикл формує результат в інтервалі $k=[\overline{1,n})$ і містить вкладений цикл множення в інтервалах $i=[\overline{0,k}]$ і $j=(\overline{k,0}]$. Другий цикл формує результат в інтервалі $k=[\overline{n,2n-1})$ з використанням допоміжного інтервалу $l=(\overline{1,n-1}]$ і містить вкладений цикл множення в інтервалах $i=[\overline{l,n}]$ і $j=[\overline{k-l,n-l})$. У вкладених циклах обчислюється добуток $(uv)^{(2w)}\leftarrow a_i^{(w)}\cdot b_j^{(w)}$ результатом якого є $2w$-розрядне ціле число, яке потім розділяється на два w-розрядних $u^{(w)}$ і $v^{(w)}$. Накопичення суми відбувається в w-розрядних тимчасових змінних r_0, r_1 і r_2 (які попередньо проініціалізовані) на кожній ітерації:

$$r_0^{(w)}\leftarrow r_0^{(w)}+v^{(w)},$$

$$r_1^{(w)}\leftarrow r_1^{(w)}+u^{(w)}+carry,\ carry\leftarrow 0,$$

$$r_2^{(w)}\leftarrow r_2^{(w)}+carry,\ carry\leftarrow 0.$$

Присвоєння кінцевого результату, а також зміна акумуляторів суми r_0, r_1 і r_2 відбувається на кожній ітерації циклу формування результату:

$$c_k^{(w)} \leftarrow r_0^{(w)},$$

$$r_0^{(w)} \leftarrow r_1^{(w)},\ r_1^{(w)} \leftarrow r_2^{(w)},\ r_2^{(w)} \leftarrow 0.$$

Після завершення циклів формування результату відбувається обчислення $c_{2n-1}^{(w)} \leftarrow r_0^{(w)}$.

Результатом множення є ціле число $c = (c_{2n-1},...,c_k,...,c_1,c_0)$.

Вхід: $a,b \in Z$, $a = \{a_{n-1},\ldots,a_i,\ldots,a_1,a_0\}$, $b = \{b_{n-1},\ldots,b_i,\ldots,b_1,b_0\}$, $\log_2 a_i = \log_2 b_i = w$, w–довжина машинного слова, $n = \log_{2^w} a$, $nk = 2n-1$.

Вихід: $c = a \cdot b$, $c \in Z$, $c = \{c_{2n-1},\ldots,c_i,\ldots,c_1,c_0\}$, $\log_2 c_i = w$.

1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$, $r_2^{(w)} \leftarrow 0$.
2. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do
2.1. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do
2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$.
2.1.2. $r_0^{(w)} \leftarrow r_0^{(w)} + v^{(w)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(w)} + carry$, $carry \leftarrow 0$.
2.1.3. $r_2^{(w)} \leftarrow r_2^{(w)} + carry$, $carry \leftarrow 0$.
2.2. $c_k^{(w)} \leftarrow r_0^{(w)}$, $r_0^{(w)} \leftarrow r_1^{(w)}$, $r_1^{(w)} \leftarrow r_2^{(w)}$, $r_2^{(w)} \leftarrow 0$.
3. For $k \leftarrow n$, $l \leftarrow 1, k < nk$, $k{+}{+}$, $l{+}{+}$ do
3.1. For $i \leftarrow l$, $j \leftarrow k-l$, $i < n$, $i{+}{+}$, $j{-}{-}$ do
3.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$.
3.1.2. $r_0^{(w)} \leftarrow r_0^{(w)} + v^{(w)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(w)} + carry$, $carry \leftarrow 0$.
3.1.3. $r_2^{(w)} \leftarrow r_2^{(w)} + carry$, $carry \leftarrow 0$.
3.2. $c_k^{(w)} \leftarrow r_0^{(w)}$, $r_0^{(w)} \leftarrow r_1^{(w)}$, $r_1^{(w)} \leftarrow r_2^{w}$, $r_2^{(w)} \leftarrow 0$.
4. $c_{nk}^{(w)} \leftarrow r_0^{(w)}$.
5. Return (c).

Рис. 1.10. Псевдокод методу Comba множення двох цілих чисел в двійковому представленні

Обчислювальна складність методу множення двох цілих чисел методом Comba: $I_{mul}(A_{1.6}) = 2I_{a\,\mathrm{sgn}}^{(w)}\left(5n^2+4n+2\right) + 2n^2 I_{a\,\mathrm{sgn}}^{2w} + 2n^2 I_{mul}^{2w} + 8n^2 I_{add}^{w}$.

1.3.6. Піднесення до квадрату

Метод 1.7. *Метод піднесення до квадрату цілих чисел в двійковому представлені*

Операція піднесення до квадрату цілих чисел є окремим випадком множення, при якому обидва множники рівні. При піднесенні до квадрату для n-значного числа, необхідно провести тільки $\left(n^2+n\right)/2$ унікальних операцій множення, на відміну від n^2 операцій для множення в стовпчик [102, 142-143, 159].

На вхід подається двійкове ціле число $a=\{a_{n-1},\ldots,a_i,\ldots,a_1,a_0\}$, яке представлено як послідовність з n машинних слів, розміром w біт кожне. Далі обчислюються проміжні результати – значення квадратів в кожному рядку $a_i\cdot a_i$, а після, проміжні результати – значення подвійного результату множення $2\cdot a_i\cdot a_j$ з урахуванням всіх переносів. На виході отримуємо результат – двійкове число $d\in Z$, розміром $2n$ машинних слів.

Вхід: $a\in Z$, $a=\{a_{n-1},\ldots,a_1,a_0\}$, $\log_2 a_i=w$, w–довжина машинного слова. *Вихід*: $d=a\cdot a$, $d\in Z$, $d=\{d_{2n-1},\ldots,d_1,d_0\}$, $\log_2 d_i=w$.	
1. For $i\leftarrow 0$, $i\le 2n-1$, $i++$. 1.1. $d_i^{(w)}\leftarrow 0$; 2. For $i\leftarrow 0$, $i\le n-1$, $i++$. 2.1. $(uv)^{(2w)}\leftarrow d_{2i}^{(w)}+a_i^{(w)}\cdot a_i^{(w)}$. 2.2. $d_{2i}^{(w)}\leftarrow v^{(w)}$, $c^{(w)}\leftarrow u^{(w)}$.	2.3. For $j\leftarrow i+1$, $j\le n-1$, $j++$ 2.3.1. $(uv)^{(2w)}\leftarrow d_{i+j}^{(w)}+2\cdot a_i^{(w)}\cdot a_j^{(w)}+c^{(w)}$ 2.3.2. $d_{i+j}^{(w)}\leftarrow v^{(w)}$, $c^{(w)}\leftarrow u^{(w)}$. 2.4. $d_{i+n}^{(w)}\leftarrow u^{(w)}$. 3. Return (d).

Рис. 1.11. Псевдокод методу піднесення в квадрат цілих чисел в двійковому представлені

Обчислювальна складність методу піднесення в квадрат цілих чисел:

$$I_{sqr}\left(A_{1.7}\right)=n^2I_{mul}^{w}+n^2I_{add}^{w}+(n^2+4n)I_{a\mathrm{sgn}}^{w}+\frac{n^2+n}{2}I_{a\mathrm{sgn}}^{2w}.$$

Розглянуті особливості та закономірності перетворення піднесення в квадрат на прикладі методу множення в стовпчик, можливо їх застосування для оптимізації інших методів множення, зокрема для методу множення

Comba. Нижче представлено метод піднесення до квадрату цілих чисел на основі методу множення Comba.

Метод 1.8. *Метод піднесення до квадрату цілих чисел в двійковому представлені на основі методу множення Comba*

Вхід: $a \in Z$, $a = \{a_{n-1}, \ldots, a_i, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, w–довжина машинного слова, $n = \log_{2^w} a$, $nk = 2n - 1$.

Вихід: $c = a \cdot a$, $c \in Z$, $c = \{c_{2n-1}, \ldots, c_i, \ldots, c_1, c_0\}$, $\log_2 c_i = w$.

1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$, $r_2^{(w)} \leftarrow 0$.
2. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do
2.1. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do
2.1.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$.
2.1.2. if $(i < j)$ then $r_0^{(2w)} \leftarrow r_0^{(2w)} + (u^{(w)} << 1)$,
$r_1^{(2w)} \leftarrow r_1^{(2w)} + \left(u^{(w)} << 1\right) + carry$, $carry \leftarrow 0$;
else $r_0^{(w)} \leftarrow r_0^{(w)} + u^{(w)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(w)} + carry$, $carry \leftarrow 0$.
2.1.3. $r_2^{(w)} \leftarrow r_2^{(w)} + carry$, $carry \leftarrow 0$.
2.2. $c_k^{(w)} \leftarrow r_0^{(w)}$, $r_0^{(w)} \leftarrow r_1^{(w)}$, $r_1^{(w)} \leftarrow r_2^{(w)}$, $r_2^{(w)} \leftarrow 0$.
3. For $k \leftarrow n$, $l \leftarrow 1, k < nk$, $k{+}{+}$, $l{+}{+}$ do
3.1. For $i \leftarrow l$, $j \leftarrow k - l$, $i < n$, $i{+}{+}$, $j{-}{-}$ do
3.1.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$.
3.1.2. if $(i < j)$ then $r_0^{(2w)} \leftarrow r_0^{(2w)} + (u^{(w)} << 1)$,
$r_1^{(2w)} \leftarrow r_1^{(2w)} + \left(u^{(w)} << 1\right)$;
else $r_0^{(w)} \leftarrow r_0^{(w)} + u^{(w)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(w)} + carry$, $carry \leftarrow 0$.
3.1.3. $r_2^{(w)} \leftarrow r_2^{(w)} + carry$, $carry \leftarrow 0$.
3.2. $c_k^{(w)} \leftarrow r_0^{(w)}$, $r_0^{(w)} \leftarrow r_1^{(w)}$, $r_1^{(w)} \leftarrow r_2^w$, $r_2^{(w)} \leftarrow 0$.
4. $c_{nk}^{(w)} \leftarrow r_0^{(w)}$.
5. Return (c).

Рис. 1.12. Псевдокод методу піднесення в квадрат цілих чисел в двійковому представлені на основі методу множення Comba

У вкладених циклах на Рис. 1.12 (крок 2.1 та 3.1) добуток $a_i^{(w)} \cdot a_j^{(w)}$ обчислюється один раз при $i = j$, і двічі – при $i \neq j$. Добуток $a_i^{(w)} \cdot a_j^{(w)}$ і

$a_j^{(w)} \cdot a_i^{(w)}$ рівні, тому вони обчислюються один раз, а результат зсувається вліво на 1 біт, для подвоєння. Слід зазначити, що при піднесенні до квадрату значно скорочується кількість необхідних для виконання операцій множення.

Обчислювальна складність методу піднесення до квадрату цілих чисел на основі множення методом Comba:

$$I_{sqr}(A_{1.8}) = 4I_{a\text{sgn}}^{(w)} + 2n^2\left(I_{a\text{sgn}}^{2w} + I_{mul}^{2w} + 4I_{add}^{w} + 5I_{a\text{sgn}}^{w} + I_{cmp}^{w} + 2I_{shl}^{w}\right) + 8nI_{a\text{sgn}}^{w}.$$

1.3.7. Приведення за модулем та ділення

Існує велика кількість наукових публікацій та підходів до реалізації операцій ділення та приведення по модулю цілих чисел, аналіз яких дозволив запропонувати класифікацію зазначених операцій, що враховує різні аспекти методів [160-161] за:

1. представленням цілого числа (модуля, діленого і дільника) [120, 123-124];
2. модулем (дільником) – прості числа (загального вигляду, узагальнені мерсенові і псевдо-мерсенові) і довільні числа [161-164];
3. отриманими результатами (частка, остача);
4. направленням аналізу діленого (з молодших біт [165-166], зі старших біт [165], двосторонні і багатосторонні [165]);
5. модифікаціями методів. З огляду на різні підходи до ділення і приведення по модулю, виділяють кілька основних методів та їх модифікацій [127]: 1) класичний («ділення в стовпчик») [120, 145]; 2) метод Монтгомері [167] і його модифікації [168]; 3) метод Барретта [169] і його модифікації [146, 1169]; 4) універсальний і жорстко запрограмований спеціальний дільник (модуль), який, як правило, застосовується для простих модулів спеціального виду (наприклад, мерсенових, псевдо-мерсенових або узагальнених мерсенових чисел) [163-164]; 5) розширений алгоритм Евкліда [170]; 6) метод ітерацій Ньютона [171, 192]; 7) метод ділення Джебеліна [172] і його модифікації;
6. точністю результату (точне значення, наближене значення);
7. можливістю розпаралелювання;

8. наявністю передобчислень (одноразове перекриття [168], ітеративне (що чергується) перекриття [162], використання одного або декількох передобчислювальних значень [173], таблиця передобчислень для простого загального вигляду [173], таблиця передобчислень для простого спеціального виду DRF [173], таблиця передобчислень для модуля спеціального виду DRF [173]);
9. наявністю попереднього обчислення констант (з і без попереднього обчислення констант) [162];
10. методом множення, що використовується («в стовпчик» [102, 140, 145-147], Comba [151-152], Карацуби [102, 145-146, 148]).

Класифікація методів ділення і приведення за модулем дає можливість вибрати необхідний метод залежно від ряду умов, які мають місце в тій чи іншій ситуації.

1.3.7.1. Приведення за модулем

Метод 1.9. *Класичне приведення цілих чисел за простим модулем*

Розглянемо класичний метод приведення цілих чисел за модулем, який базується на методі ділення в стовпчик [120, 160]. Він може бути використаний для приведення за модулем цілих чисел довільного розміру. На вхід подаються число $a=(a_{m-1},...,a_i,...,a_1,a_0)$ і просте число $p=(p_{n-1},...,p_i,...,p_1,p_0)$, які представлені як набір машинних слів, де a_i і p_i – відповідні машинні слова чисел a і p, m – кількість машинних слів, які необхідні для подання числа a ($m=\lceil \log_b a \rceil$), n – кількість машинних слів, які необхідні для представлення числа p ($n=\lceil \log_b p \rceil$), b – число, що займає w біт ($b=2^w$), w – розмір машинного слова в бітах (наприклад, $w=32$). Спочатку виконується ініціалізація остачі $r \leftarrow a$. На наступному кроці, якщо виконується умова $r \geq pb^m$, виконується вирівнювання $r \leftarrow r - pb^m$. Основу методу складає цикл ділення в стовпчик в інтервалі $i=(\overline{n+m+1,n}]$, на кожній ітерації якого виконуються наступні дії:

1. Оскільки остача r повинна бути менше модуля p, то виконується перевірка $r_i == p_{n-1}$.
2. Якщо умова перевірки в пункті 1 задовольняється, то частку q необхідно зменшити: $q \leftarrow b-1$.
3. Якщо умова перевірки в пункті 1 не задовольняється, то виконується цілочисельне ділення двох сусідніх машинних слів числа r, на найстарше машинне слово модуля p: $q \leftarrow (r_i b + r_{i-1}) \operatorname{div} p_{n-1}$.
4. Присвоюється результат тимчасової змінної y: $y \leftarrow q(p_{n-1}b + p_{n-2})$.
5. В циклі корекції, поки виконується умова, що y більше, ніж відповідні три сусідні машинних слова остачі $(y > r_i b^2 + r_{i-1}b + r_{i-2})$, необхідно зменшити частку $q \leftarrow q-1$, а також змінити значення умови $y \leftarrow y - (p_{n-1}b + p_{n-2})$.
6. Отримати значення остачі: $r \leftarrow r - qpb^{i-n}$.
7. Оскільки значення r не може бути від'ємним, то необхідно виконати перевірку $r < 0$ і, за необхідності, виконати корекцію: $r \leftarrow r + pb^{i-n}$.

На виході отримуємо результат $r = a \bmod p$ $(r < p)$. Проте, програмна реалізація даного способу є досить повільною.

Вхід: $a, p \in Z$, $0 \le a < b^{n+m}$, $m = \lceil \log_b a \rceil$, $n = \lceil \log_b p \rceil$, $n \ge 2$, $m \ge 1$, $b = 2^w$, $p_{n-1} \ge b/2$, p – просте число. *Вихід*: $r = a \bmod p$.	
1. $r \leftarrow a$. 2. If $(r \ge p \cdot b^m)$ 2.1. $r \leftarrow r - p \cdot b^m$. 3. For $i \leftarrow n+m+1$, $i \ge n$, $i--$ 3.1. If $(r_i == p_{n-1})$ 3.1.1. $q \leftarrow b-1$. 3.1.2. Else 3.1.3. $q \leftarrow (r_i \cdot b + r_{i-1}) \operatorname{div} p_{n-1}$.	3.2. $y \leftarrow q \cdot (p_{n-1} \cdot b + p_{n-2})$. 3.3. While $(y > r_i \cdot b^2 + r_{i-1} \cdot b + r_{i-2})$ do 3.3.1. $q \leftarrow q-1$. 3.3.2. $y \leftarrow y - (p_{n-1} \cdot b + p_{n-2})$. 3.4. $r \leftarrow r - q \cdot p \cdot b^{i-n}$. 3.5 If $(r < 0)$ 3.5.1. $r \leftarrow r + p \cdot b^{i-n}$. 4. Return (r).

Рис. 1.13. Псевдокод класичного приведення цілих чисел за простим модулем

Обчислювальна складність класичного методу приведення за модулем:

$$I_{\mathrm{mod}}\left(A_{1.9}\right)=2I_{a\mathrm{sgn}}^{w}+I_{cmp}^{w}+I_{sub}^{w}+$$

$$+\left(m+1\right)\left(5I_{add}^{w}+I_{div}^{w}+2I_{sub}^{w}+2I_{mul}^{w}+3I_{cmp}^{w}+5I_{a\mathrm{sgn}}^{w}\right)+$$

$$+x\left(m+1\right)\left(I_{a\mathrm{sgn}}^{w}+I_{add}^{w}+2I_{sub}^{w}\right).$$

Програмна реалізація методу ділення в стовпчик, яка використовується в цьому методі, є досить повільною, що стало поштовхом до пошуку нових методів приведення за модулем [160, 167-169]. Так з'явились методи Барретта [169] та Монтгомері [167], які широко використовуються в криптографічних застосуваннях.

Метод 1.10. *Метод Монтгомері для приведення цілих чисел за простим модулем*

В основі методу приведення за модулем Монтгомері [167] лежить ділення на дільник спеціального виду, що є степенем b, замість звичайного ділення на дільник загального виду. Нехай $R>p$ є цілим, взаємно простим з модулем p, таким, що приведення за модулем R обчислюється легко. Класичним вибором є $R=b^k$. В цьому методі, числа $u<p$, представлені p-вирахуваннями, що відповідають R, тобто $uR \bmod p$. Таке представлення цілих чисел часто називають формою Монтгомері. Приведення Монтгомері числа u визначається як $uR^{-1} \bmod p$, де R^{-1} є мультиплікативною інверсією R за модулем p. Це приведення передбачає, що цілі числа представлені в формі Монтгомері. Нехай два цілих числа $d<p$ та $e<p$ представлені в формі Монтгомери $dR \bmod p$ та $eR \bmod p$, відповідно, тоді їх добуток має вид $deR^2 \bmod p$. Подальше застосування Методу 1.10, дозволить отримати $r=deR \bmod p$, яке також є в формі Монтгомері. Так в методі використовується передобчислена величина $\beta=-p^{-1} \bmod R$, яка після оптимізації [160] була приведена лише до $\beta=-p_0^{-1} \bmod b$. В основі оптимізації покладено спостереження, що в основі Методу 1.10 лежить накопичення суми добутків

p та r, до тих пір, поки результат стане кратним R. Такий ефект може бути досягнуто шляхом обчислення $r_i\beta \bmod b$, замість $r_i p_0 \bmod R$, де $\beta = -p_0^{-1} \bmod b$.

Для перевірки коректності методу, необхідно замінити оригінальне значення r на $\bar{r}$. Так, $\beta \cdot p_0^{-1} = -1 \bmod b$, кожен раз в п.2.1 вирішується рівняння $pt = pr_i\beta \equiv r_i \bmod b$. Слід зазначити, що приведення потребує не більше одного фінального віднімання p.

Вхід: ціле $a = d \cdot e$, $d, e \in \mathrm{GF}(p)$, $0 \le a < p \cdot b^n$, $n = \lceil \log_b p \rceil$, $n \ge 2$, $m \ge 1$, $b = 2^w$, $\beta = -p_0^{-1} \bmod b$, p – просте. *Вихід*: $r = a\left(b^n\right)^{-1} \bmod p$.	
1. $r \leftarrow a$. 2. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 2.1. $t \leftarrow (r_i \cdot \beta) \bmod b$. 2.2. $r \leftarrow r + (p \cdot t) \cdot b^i$.	3. $r \leftarrow r \operatorname{div} b^n$. 4. If $(r \ge p)$ 4.1. $r \leftarrow r - p$. 5. Return (r).

Рис. 1.14. Псевдокод методу Монтгомері

Обчислювальна складність методу Монтгомері для приведення за модулем:

$$I_{\bmod}(A_{1.10}) = 3I^w_{a\,\mathrm{sgn}} + n\left(2I^w_{a\,\mathrm{sgn}} + 3I^w_{mul} + I^w_{add} + I^w_{\bmod}\right) + I^w_{cmp} + I^w_{sub} + I^w_{div}.$$

Існує ряд модифікацій методу Монтгомері, розгляд яких виходить за рамки даної роботи.

Метод 1.11. *Метод Барретта для приведення цілих чисел за простим модулем*

В прикладній криптографії в якості методу приведення за модулем також широко використовується метод Барретта [160, 169]. Він базується на ідеї оцінки частки x/p, за допомогою операцій, які можуть бути передобчислені або менш затратні з обчислювальної точки зору, ніж ділення багатократної точності. Остача r від ділення x/p може бути обчислена як $r = x - p\lfloor x/p \rfloor$. Враховуючи цей, припустимо, що ділення на число b в деякій степені:

$$r = x - p\left\lfloor \frac{\frac{x}{b^{n-1}}\frac{b^{2n}}{p}}{b^{n+1}} \right\rfloor = x - p\left\lfloor \frac{\frac{x}{b^{n-1}}\mu}{b^{n+1}} \right\rfloor,$$

де $\mu = \left\lfloor \frac{b^{2n}}{p} \right\rfloor$ – заздалегідь обчислена константа.

В методі приведення, запропонованому Барреттом, використовується виключно множення багатократної точності та зсуви на розмір машинного слова (звичайне копіювання). Тоді оцінку остачі $\hat{r}$ від ділення x/p можна представити як:

$$\hat{r} = \left(x \bmod b^{n+1} - \left(p\hat{q} \bmod b^{n+1}\right)\right) \bmod b^{n+1}.$$

Оцінка $\hat{r}$ передбачає, що достатньо не більше ніж два віднімання модуля p для отримання точного значення остачі r.

Метод Барретта може бути використаний для приведення по модулю цілих чисел довільного розміру. На вхід подаються число $a = (a_{m-1},...,a_i,...,a_1,a_0)$ и просте число $p = (p_{n-1},...,p_i,...,p_1,p_0)$, які представлені як набір машинних слів, де a_i та p_i – відповідні машинні слова чисел a та p, m – кількість машинних слів, що необхідні для представлення числа a ($m = \lceil \log_b a \rceil$), n – кількість машинних слів, що необхідні для представлення числа p ($n = \lceil \log_b p \rceil$), b – число, яке займає w біт ($b = 2^w$), w – розмір машинного слова в бітах (наприклад, $w = 32$). Число a знаходиться в інтервалі $0 \le a < b^{2k}$, де $k = \lfloor \log_b p \rfloor + 1$. Для здійснення операцій приведення за модулем, необхідно виконати попереднє обчислення константи $\mu = \lfloor b^{2k}/p \rfloor$. Після цього необхідно виконати оцінку частки $\hat{q} \leftarrow \left\lfloor \left(\left(a \operatorname{div} b^{k-1}\right) \cdot \mu\right) \operatorname{div} b^{k+1} \right\rfloor$. Далі, використовуючи тимчасові змінні r_1 та r_2, обчислюються проміжні значення $r_1 \leftarrow \left(a \bmod b^{k+1}\right)$ та $r_2 \leftarrow \left(\hat{q} \cdot p \bmod b^{k+1}\right)$. Для обчислення остачі r, виконується віднімання: $r \leftarrow r_1 - r_2$. Для врахування переносу (якщо $r < 0$), виконується наступна операція: $r \leftarrow r + b^{k+1}$. У

випадку, якщо отримана остача $r \geq p$, необхідно виконати корекцію $r \leftarrow r - p$ поки r не стане менше p. На виході отримуємо результат $r = a \bmod p$ ($r < p$).

Вхід: $a, p \in Z$, $0 \leq a < b^{2k}$, $k = \lfloor \log_b p \rfloor + 1$, $b = 2^w$, p – просте, $\mu = \lfloor b^{2k} / p \rfloor$. *Вихід*: $r = a \bmod p$.	
1. $\hat{q} \leftarrow \left\lfloor \left(\left(a \operatorname{div} b^{k-1} \right) \cdot \mu \right) \operatorname{div} b^{k+1} \right\rfloor$. 2. $r_1 \leftarrow \left(a \bmod b^{k+1} \right)$. 3. $r_2 \leftarrow \left(\hat{q} \cdot p \bmod b^{k+1} \right)$. 4. If $(r_1 > r_2)$ 4.1. $r \leftarrow r_1 - r_2$.	4.3. Else 4.4. $r \leftarrow r_1 + b^{k+1} - r_2$. 5. While $(r \geq p)$ 5.1. $r \leftarrow r - p$. 6. Return (r).

Рис. 1.15. Псевдокод методу Баррета

Для використання методу Барретта необхідно виконати попередній розрахунок значення $\mu = \lfloor b^{2k} / p \rfloor$:

Вхід: $k = \lfloor \log_b p \rfloor + 1$, $b = 2^w$, p – просте. *Вихід*: $\mu = \lfloor b^{2k} / p \rfloor$.
1. $\mu = \lfloor b^{2k} \operatorname{div} p \rfloor$. 2. Return (μ).

Рис. 1.16. Розрахунок значення μ

При цьому розрахунок значення μ не є безпосередньо частиною методу Барретта і може виконуватись заздалегідь.

Обчислювальна складність методу Барретта для приведення за модулем (без врахування передобчислень):

$$I_{\bmod}\left(A_{1.11}\right) = 5I^w_{a\,\mathrm{sgn}} + 2I^w_{mul} + 2I^w_{div} + 2I^w_{\bmod} + I^w_{cmp} + 2I^w_{sub}.$$

Відомо, що в методі Барретта в п. 1 и п. 3 виконуються два часткових (не повних) множення великих цілих чисел: перше множення – числа a з передобчисленим значенням μ, а друге – модуля p з обчисленим значенням оцінки $\hat{q}$. В методі Барретта немає необхідності в повному множенні, так як використовується або старша, або молодша частина добутку. Виходячи з цього, можна зробити висновок, що швидкодія методу Барретта для

приведення за модулем, залежить від вибраного методу множення. В класичній реалізації методу Барретта використовується метод множення в стовпчик [102, 140] (Метод 1.5), але в криптографічних застосуваннях метод Барретта зазвичай використовується зі спеціальними версіями множення в стовпчик [174], або з іншими методами множення, наприклад, на основі методу множення Comba [151, 160] (Метод 1.6).

1.3.7.2. Ділення

Метод 1.12. *Класичний метод ділення в стовпчик*

Розглянемо класичний метод ділення двох цілих чисел. На вхід подаються два цілих невід'ємних числа $u = \{u_{m+n-1}, \ldots, u_1, u_0\}$ (довжиною $m+n$ машинних слів) та $v = \{v_{n-1}, \ldots, v_1, v_0\}$ (довжиною n машинних слів). Числа u та v представлені у вигляді набору машинних слів, де u_i та v_j – відповідні машині слова, довжиною w біт кожне. Для початку, необхідно ініціалізувати додаткове машинне слово $u_{m+n} \leftarrow 0$ та змінну $d \leftarrow 0$. Далі, необхідно виконати нормалізацію. В циклі з умовою $v_{n-1} < b/2$ необхідно виконати наступні перетворення:

$$v \leftarrow 2v,\ u \leftarrow 2u,\ d \leftarrow 2d.$$

На наступному етапі реалізується безпосередньо цикл ділення в інтервалі $i = [\overline{m,0}]$, в якому виконуються наступні дії:

1. Присвоюється значення тимчасової змінної $\hat{q}$: $\hat{q} \leftarrow \min\left(\left\lfloor (u_{i+n}b + u_{i+n-1})/v_{n-1} \right\rfloor, b-1\right)$.
2. Якщо задовольняється умова $\hat{q}(v_{n-1}b + v_{n-2}) > \left(u_{i+n}b^2 + u_{i+n-1}b + u_{i+n-2}\right)$ в додатковому циклі, то необхідно зменшити значення $\hat{q}$ на 1: $\hat{q} \leftarrow \hat{q} - 1$.
3. Провести заміну $\{u_{i+n}, \ldots, u_i\}$ на $\{u_{i+n}, \ldots, u_i\} - \hat{q}\{v_{n-1}, \ldots, v_0\}$.
4. Якщо отриманий в п.5 результат від'ємний ($\{u_{i+n}, \ldots, u_i\} < 0$), необхідно зменшити значення $\hat{q}$ на 1 та виконати компенсацію додаванням: $\{u_{i+n}, \ldots, u_i\} \leftarrow \{u_{i+n}, \ldots, u_i\} + \{0, v_{n-1}, \ldots, v_0\}$.
5. Присвоїти $q_i \leftarrow \hat{q}$.

Після завершення роботи циклу, обчислюється остача:

$$r \leftarrow u/d.$$

На виході отримуємо результат ділення $q = \lfloor u/v \rfloor$ та остачу від ділення $r = u \bmod v$.

Вхід: $u, v \in Z$, $u = \{u_{m+n-1}, \ldots, u_1, u_0\}$, $v = \{v_{n-1}, \ldots, v_1, v_0\}$, $v_{n-1} > 0$, $n > 1$, w – довжина машинного слова, $b = 2^w$ – число, яке займає w біт. *Вихід*: $q = \lfloor u/v \rfloor$, $r = u \bmod v$, $q, r \in Z$, $q = \{q_m, \ldots, q_1, q_0\}$, $r = \{r_{n-1}, \ldots, r_1, r_0\}$.
1. $u_{m+n} \leftarrow 0$, $d \leftarrow 0$ 2. While $v_{n-1} < b/2$ do 2.1. $v \leftarrow 2v$, $u \leftarrow 2u$, $d \leftarrow 2d$. 3. For $i \leftarrow m$, $i \le 0$, $i--$. 3.1. $\hat{q} \leftarrow \min\left(\lfloor (u_{i+n} b + u_{i+n-1})/v_{n-1} \rfloor, b-1 \right)$. 3.2. While $\hat{q}(v_{n-1} b + v_{n-2}) > \left(u_{i+n} b^2 + u_{i+n-1} b + u_{i+n-2} \right)$ do 3.2.1. $\hat{q} \leftarrow \hat{q} - 1$. 3.3. $\{u_{i+n}, \ldots, u_i\} \leftarrow \{u_{i+n}, \ldots, u_i\} - \hat{q}\{v_{n-1}, \ldots, v_0\}$. 3.4. if $\{u_{i+n}, \ldots, u_i\} < 0$ then 3.4.1. $\hat{q} \leftarrow \hat{q} - 1$, 3.4.2. $\{u_{i+n}, \ldots, u_i\} \leftarrow \{u_{i+n}, \ldots, u_i\} + \{0, v_{n-1}, \ldots, v_0\}$. 3.5. $q_i \leftarrow \hat{q}$. 4. $r \leftarrow u/d$. 5. Return (q, r).

Рис. 1.17. Псевдокод методу класичного ділення в стовпчик

Загалом, метод представляє собою серію успішних оновлень значення діленого u, шляхом віднімання дільника v, і подальшого множення на поточне машинне слово частки $\hat{q}$. Результат кожного такого оновлення зсувається вправо відносно попереднього.

Обчислювальна складність класичного методу ділення двох чисел в двійковій формі:

$$I_{div}(A_{1.12}) = 3I^w_{a\text{sgn}} + m\left(5I^w_{add} + I^w_{div} + 4I^w_{sub} + 2I^w_{mul} + 3I^w_{cmp} + 6I^w_{a\text{sgn}}\right) + I^w_{div}.$$

Метод 1.13. *Метод Барретта для ділення цілих чисел*

Метод приведення за модулем Барретта (Метод 1.11) може бути використаний для ділення цілих чисел довільного розміру. На вхід подається число $a = (a_{m-1}, ..., a_i, ..., a_1, a_0)$ – ділене, и число $p = (p_{n-1}, ..., p_i, ..., p_1, p_0)$ – дільник, які представлені як набір машинних слів, де a_i та p_i – відповідні

машинні слова чисел a та p, m – кількість машинних слів, необхідних для представлення чисел a ($m=\lceil \log_b a \rceil$), n – кількість машинних слів, необхідних для представлення числа p ($n=\lceil \log_b p \rceil$), b – число, яке займає w біт ($b=2^w$), w – розмір машинного слова в бітах (наприклад, $w=32$). Число a знаходиться в інтервалі $0 \le a < b^{2k}$, де $k=\lfloor \log_b p \rfloor +1$. Для здійснення перетворення приведення за модулем, необхідно виконати попереднє обчислення константи $\mu=\left\lfloor b^{2k}/p \right\rfloor$ $\mu=\left\lfloor \frac{b^{2k}}{p} \right\rfloor$. Після цього, необхідно виконати оцінку частки $q \leftarrow \left\lfloor \left(\left(a \operatorname{div} b^{k-1} \right) \cdot \mu \right) \operatorname{div} b^{k+1} \right\rfloor$. Далі, обчислюється проміжне значення остачі $r \leftarrow a \bmod b^{k+1} - q \cdot p \bmod b^{k+1}$. Для врахування переносу (поки виконуються умови $r<0$ та $r \ge p$), виконуються наступні перетворення:

$$r \leftarrow r+p \text{ та } q \leftarrow q-1 \text{ (при } r<0),$$

$$r \leftarrow r-p \text{ та } q \leftarrow q+1 \text{ (при } r \ge p).$$

На виході отримуємо частку $q=a/p$ та остачу $r=a \bmod p$ ($r<p$).

Вхід: $a, p \in Z$, $0 \le a < b^{2k}$, $k=\lfloor \log_b p \rfloor +1$, $b=2^w$, p – просте, $\mu=\left\lfloor b^{2k}/p \right\rfloor$. *Вихід*: $r=a \bmod p$, $q=a/p$.	
1. $q \leftarrow \left\lfloor \left(\left(a \operatorname{div} b^{k-1} \right) \cdot \mu \right) \operatorname{div} b^{k+1} \right\rfloor$. 2. $r \leftarrow a \bmod b^{k+1} - q \cdot p \bmod b^{k+1}$. 3. While $\left((r<0) or (r \ge p) \right)$ 3.1. If $(r<0)$ 3.1.1. $r \leftarrow r+p$.	3.1.2. $q \leftarrow q-1$. 3.2. Else 3.2.1. $r \leftarrow r-p$. 3.2.2. $q \leftarrow q+1$. 4. Return (q,r).

Рис. 1.18. Псевдокод методу Баррета для ділення цілих чисел

Як і для операцій приведення за модулем, при використанні методу Барретта для ділення цілих чисел, необхідно виконати попереднє обчислення значення $\mu=\left\lfloor b^{2k}/p \right\rfloor$, відповідно до Рис. 1.16.

Обчислювальна складність методу ділення Барретта (без врахування передобчислень):

$$I_{\text{div}}\left(A_{1.13}\right)=4I^{w}_{a\text{sgn}}+2I^{w}_{mul}+2I^{w}_{div}+2I^{w}_{\text{mod}}+I^{w}_{add}+2I^{w}_{sub}+2I^{w}_{cmp}.$$

1.3.8. Порівняння

Метод 1.14. *Метод порівняння цілих чисел в двійковому представленні*

Розглянемо метод порівняння двох цілих чисел [140]. На вхід подаються два цілих числа $a=\{a_m,\ldots,a_1,a_0\}$ та $b=\{b_n,\ldots,b_1,b_0\}$ в двійковому виді, які представлені як послідовності з машинних слів, розміром w біт кожне. Результати порівняння зручно представити у виді функції:

$$sign\,(a-b)=\begin{cases}+1,\ if\ \ a>b,\\ \ 0,\ if\ \ a=b,\\ -1,\ if\ \ a<b.\end{cases}$$

Спочатку необхідно порівняти кількість машинних слів, які необхідні для представлення чисел a та b, і, у випадку неоднакової кількості, визначити більше число. Далі, якщо $m==n$, необхідно визначити значення індексу перших відмінних машинних слів чисел a та b і визначити більше з них. На виході отримуємо результат $r=sign\,(a-b)$.

Вхід: $a,b\in Z$, $a=\{a_m,\ldots,a_1,a_0\}$, $b=\{b_n,\ldots,b_1,b_0\}$, $\log_2 a_i=\log_2 b_i=w$, $a_m\neq 0$, $b_n\neq 0$, w – довжина машинного слова. *Вихід*: $r=sign\,(a-b)$.	
1. If $m\neq n$ then 1.1. $r\leftarrow sign\,(m-n)$. 2. Else 2.1. $i\leftarrow m$. 2.2. While $\left(a_i^{(w)}==b_i^{(w)}\right)$ do 2.2.1. $i\leftarrow i-1$.	2.3. If $i==-1$ then 2.3.1. $r\leftarrow 0$. 2.4. Else 2.4.1. $r\leftarrow sign\,(a_i^{(w)}-b_i^{(w)})$. 3. Return (r).

Рис. 1.19. Псевдокод методу порівняння цілих чисел в двійковому представленні

Обчислювальна складність методу порівняння цілих чисел в двійковій формі:

$$I_{cmp}\left(A_{1.15}\right)=4I^{w}_{a\text{sgn}}+I_{sub}+2I_{cmp}+m\left(I_{sub}+I_{a\text{sgn}}\right)+I^{w}_{sub}+I^{w}_{cmp}.$$

1.4. Висновки до першого розділу

У першому розділі дисертаційної роботи було проведено аналіз наукової літератури за темою дисертації, зокрема, проаналізовано особливості функціонування національної ІВК. В результаті дослідження було отримано наступні результати:

1. Встановлено, що для складових частин національної ІВК, а саме ІТС ЦСК, характерною особливістю є обробка великої кількості криптографічних операцій по створенню та перевірці ЕП, що направлені до сервісів ЦСК (зокрема до сервісів OCSP та TSP). У зв'язку з гармонізацією законодавства, поступовим збільшенням кількості електронних послуг, в яких застосовується ЕП, та кількості користувачів цих послуг, призводить до збільшення навантаження на сервіси ЦСК.

2. Арифметичні перетворення над цілими числами присутні в усіх алгоритмах ЕП, що використовуються в національній ІВК при створенні та перевірці ЕП, не залежно від математичного апарату, на якому вони ґрунтуються. Оскільки арифметичні перетворення з цілими числами є основою криптографічних перетворень з відкритим ключем, то підвищити швидкодію останніх можна за рахунок підвищення швидкодії арифметичних операцій.

3. Аналіз представлень цілих чисел показав, найбільш простим та універсальним є двійкове представлення цілого числа, проте це представлення не завжди найефективніше з точки зору швидкодії. Решта представлень цілих чисел мають свою обмежену специфічну сферу застосування у певних криптосистемах.

4. Аналіз арифметичних операцій над цілими числами в двійковому представленні показав, що в основних операціях наявний перенос з молодших машинних слів в старші, чи займ з старших машинних слів в молодші. Наявність переносу не дозволяє ефективно використовувати суперскалярну архітектуру сучасних процесорів і не дозволяє використовувати розпаралелювання операцій.

5. Підвищити швидкодію криптографічних перетворень з відкритим ключем можливо за рахунок відкладення переносу чи займу при виконанні

арифметичних операцій з цілими числами. Відкладення переносу при виконанні арифметичних операцій дозволить використовувати розпаралелювання для базових операцій.

Список використаних джерел у першому розділі

1. J.H. Saltzer, M.D. Schroeder, "The Protection of In formation in Computer Systems", Proc. IEEE, vol. 63, no. 9, pp. 1278-1308, 1975.
2. NIST SP 800-27 Rev A, Engineering Principles for Information Technology Security (A Baseline for Achieving Security), Revision A
3. William Stallings. 2016. Network Security Essentials: Applications and Standards (6th ed.). Pearson, 464 pages
4. OECD (2015), Digital Security Risk Management for Economic and Social Prosperity: OECD Recommendation and Companion Document, OECD Publishing, Paris. DOI: http://dx.doi.org/10.1787/9789264245471-en
5. Parker, D., “Toward a New Framework for Information Security,” in [The Computer Security Handbook], Wiley; 6 edition (2015). Chapter 3 of book. DOI: 10.1002/9781118851678.ch3
6. С.О. Гнатюк, В.М. Кінзерявий, А.О. Охріменко, «Особливості криптографічного захисту державних інформаційних ресурсів», Безпека інформації, Т. 17, № 1, С. 68-80, 2012.
7. Document.Online. Сучасний електронний документообіг. URL: https://document.online/
8. АСКОД.Онлайн. URL: https://askod.online/
9. Платформа ПТАХ. URL: https://edi.com.ua/uk
10. Сервіс FlyDoc(Флайдок). URL: https://flydoc.ua/uk
11. Сервіс FREDO ДокМен. URL: https://fredo.com.ua/service/servis-fredo-dokmen/opis.html
12. iFin EDI. URL: https://edi.ifin.ua/
13. Edisoft EDI. URL: https://ediweb.com/uk-ua/solutions/edi
14. Електронний документообіг Star.Docs. URL: https://kyivstar.ua/uk/business/products/stardocs

15. Сервіс електронного документообігу Вчасно. URL: https://vchasno.ua/
16. Цифровий документообіг SmartSign. URL: https://sign.it.ua/
17. MASTER:Документообіг. https://masterbuh.com/product/10
18. Кабінет платника податків. URL: https://cabinet.tax.gov.ua/
19. СОТА. URL: https://sota-buh.com.ua/
20. M.E.Doc. URL: https://medoc.ua/uk
21. Соната. URL: https://sonata.biz.ua/
22. Онлайн-сервіс звітності Liga:REPORT. URL: https://report.ligazakon.net/
23. Арт-Звіт. URL: https://art-zvit.com.ua/
24. ТАКСЕР – онлайн звітність. URL: https://taxer.ua/uk/
25. CS iFOBS. URL: https://www.csltd.com.ua/ifobs
26. iBank 2 UA. ДБО СОФТ. URL: http://dbosoft.com.ua/
27. FINIK.PRO. URL: http://finik.pro/ua.html
28. Cipher ELPay. URL: https://cipher.com.ua/uk/products/elpay
29. SR:iBank. URL: http://www.soft-review.com.ua/about/
30. НОКК. Клиент-банк. URL: http://www.nokk.kiev.ua/program_internet.htm
31. Приватбанк Приват24: https://www.privat24.ua/
32. Інтегрована система електронної ідентифікації. URL: https://id.gov.ua/
33. Портал державних послуг. URL: https://igov.gov.ua/
34. Єдиний державний портал адміністративних послуг. URL: https://my.gov.ua/
35. Електронний кабінет водія. URL: https://e-driver.hsc.gov.ua/accounts/login/
36. Державні послуги онлайн. URL: https://diia.gov.ua/
37. Кабінет електронних сервісів Міністерства юстиції. URL: https://kap.minjust.gov.ua/
38. Он-лайн будинок юстиції. URL: https://online.minjust.gov.ua/
39. Веб-ресурс електронних послуг Держгеокадастру. URL: https://e.land.gov.ua/
40. Електронний кабінет платника податків. URL: https://cabinet.tax.gov.ua/
41. Єдиний державний реєстр декларацій. URL: https://portal.nazk.gov.ua/

42. Державний реєстр речових прав на нерухоме майно. URL: https://kap.minjust.gov.ua/services?product_id=1
43. Єдиний реєстр досудових розслідувань. URL: https://erdr.gp.gov.ua/
44. Електронний майданчик «Держзакупівлі.Онлайн». URL: https://www.dzo.com.ua/
45. Торговельний майданчик «E-Tender. URL: https://e-tender.ua/
46. Тендерний майданчик «Zakupki.Prom.ua». URL: https://zakupki.prom.ua/
47. Електронний майданчик «SmartTender». URL: https://smarttender.biz/
48. Електронний майданчик «Open-Tender». URL: https://open-tender.com.ua/
49. Товарна біржа «ПУБЛІЧНІ АУКЦІОНИ». URL: https://gov.auction/
50. Електронний майданчик «25h8.auction». URL: https://25h8.auction/
51. Майданчик «ТЕНДЕР-online». URL: https://tender-online.com.ua/
52. Закон України від 05.10.2017 р. № 2155-VIII «Про електронні довірчі послуги»
53. Закон України Про електронні документи та електронний документообіг від 22.05.2003 № 851-IV
54. Розпорядження Кабінету Міністрів України від 24 червня 2016 р. № 474-р «Деякі питання реформування державного управління України»
55. Закон України «Про Національну програму інформатизації» від 04.02.1998 № 74/98-ВР (Редакція станом на 01.08.2016)
56. Постанова Правління Національного Банку України № 78 від 14.08.2017 «Про затвердження Положення про застосування електронного підпису в банківській системі України»
57. Постанова Кабінет Міністрів України № 749 від 19 вересня 2018 р. «Про затвердження Порядку використання електронних довірчих послуг в органах державної влади, органах місцевого самоврядування, підприємствах, установах та організаціях державної форми власності»
58. Горбенко Ю. І. Інфраструктури відкритих ключів. Електронний цифровий підпис. Теорія та практика : монографія / Ю. І. Горбенко, І. Д. Горбенко ; МОН України, Харк. нац. ун-т радіоелектроніки, ЗАТ "Ін-т інформац. технологій". – Харків : Форт, 2010. – 608 с. : іл. – ISBN 978-966-8599-76-7.

59. Горбенко І.Д., Горбенко Ю.І. Прикладна криптологія: Теорія. Практика. Застосування: Монографія. Вид. 2-ге, перероб. і доп. - Харків: Видавництво Форт, 2012. - 880 с.
60. Закон України «Про електронний цифровий підпис» від 22.05.2003 № 852-IV
61. Закон України «Про ратифікацію Угоди про асоціацію між Україною, з однієї сторони, та Європейським Союзом, Європейським співтовариством з атомної енергії і їхніми державами-членами, з іншої сторони» від 16.09.2014 № 1678-VII
62. Регламент (ЄС) № 910/2014 Європейського Парламенту та Ради від 23 липня 2014 року про електронну ідентифікацію та довірчі послуги для електронних транзакцій в межах внутрішнього ринку та про скасування Директиви 1999/93/ЄС
63. Костенко О.В, Проблеми правового регулювання визнання іноземних електронних довірчих послуг в Україні. - Науковий вісник Ужгородського національного університету. Серія ПРАВО. Випуск 51. Том.1. - 2018. с.192-197
64. Центральний засвідчувальний орган. Оцінка стану розвитку. URL: czo.gov.ua/development. Доступ станом на 19.07.2019.
65. Кваліфікований надавач електронних довірчих послуг «АЦСК органів юстиції України». URL: https://ca.informjust.ua/partnership. Доступ станом на 19.07.2019.
66. Послуга ЕЦП (MobileID) для абонентів Київстар. URL: https://kyivstar.ua/uk/business/products/mobile-id
67. Послуга MobileID для бізнесу Vodafone Україна. URL: https://business.vodafone.ua/uk/business/products-and-solutions/mobile-id
68. MobileID Lifecell. URL: https://www.lifecell.ua/uk/mobileid/
69. BankID НБУ. URL: https://id.bank.gov.ua/
70. Інтегрована система електронної ідентифікації. URL: https://id.gov.ua/
71. Державні послуги онлайн – Дія. URL: https://diia.gov.ua/
72. Регламент роботи центрального засвідчувального органу. Затверджено наказом Міністерства цифрової трансформації України № 27 від 19 грудня 2019 р.

73. Правила організації захисту електронних банківських документів з використанням засобів захисту інформації Національного банку України. Затверджено Постановою Правління Національного банку України 26.11.2015 № 829

74. Постанова Правління Національного банку України № 103 від 25.09.2018 «Про затвердження Інструкції про ведення касових операцій банками в Україні»

75. Про встановлення вимог до технічних засобів, процесів їх створення, використання та функціонування у складі інформаційно-телекомунікаційних систем під час надання електронних довірчих послуг. Наказ адміністрації Державної служби спеціального зв'язку та захисту інформації України № 3563/5/610 від 18.11.2019.

76. Положення про захист електронних банківських документів з використанням засобів захисту інформації Національного банку України. (Затверджено Постановою Правління Національного банку України № 829 від 26.11.2015).

77. Левшаков С. Ф. Проблеми становлення інфраструктури електронних цифрових підписів в банківській системі України / С. Ф. Левшаков // Вісник Української академії банківської справи. – 2010. – № 2. – С. 80–85.

78. Мелащенко А. О. Проблемы интероперабельности Национальной системы электронных цифровых подписей / А. О. Мелащенко, О. Л. Перевозчикова // Кибернетика и системный анализ. – 2009. – № 3. – С. 55–63.

79. Мелащенко А. О. Кроссертификация Украины / А. О. Мелащенко, О. Л. Перевозчикова // Проблемы программирования. – 2010. – № 2–3. – С. 299–308.

80. ISO/IEC 9594-8:2017 Information technology - Open Systems Interconnection - The Directory - Part 8: Public-key and attribute certificate frameworks

81. NIST SP 800-32 Introduction to Public Key Technology and the Federal PKI Infrastructure

82. Y. Elley, A. Anderson, S. Hanna, S. Mullan, R. Perlman, and S. Proctor. Building certification paths: Forward vs. reverse. In The 10th Annual Network and Distributed System Security Symposium, 2001.

83. RFC 5280. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
84. RFC 6960. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP
85. Вимоги до формату підписаних даних. Затверджено наказом Міністерства юстиції України, Адміністрації Державної служби спеціального зв'язку та захисту інформації України № 1236/5/453 від 20.08.2012
86. ДСТУ ETSI TS 101 733:2017 Електронні підписи та інфраструктури (ESI). Розширені електронні CMS-підписи (CAdES) (ETSI TS 101 733:2013, IDT)
87. Національний Банк України. Інформація про кількість клієнтів банків та кількість відкритих клієнтами рахунків. URL: https://old.bank.gov.ua/control/uk/publish/article?art_id=66320&cat_id=66315. Станом на 01.02.2020.
88. Національний банк України. У 2019 році СЕП щоденно обробляла близько 1,5 млн платежів на суму 130 млрд грн. URL: https://bank.gov.ua/news/all/u-2019-rotsi-sep-schodenno-obroblyala-blizko-15-mln-platejiv-na-sumu-130-mlrd-grn
89. А.О. Охріменко, «Оптимізація програмної реалізації криптографічних алгоритмів», Защита информации: сб. науч. труд., № 18, С. 44-51, 2011.
90. А.О. Охріменко, С.О. Гнатюк, «Використання технологій паралельної обробки даних в криптографічних перетвореннях», Захист інформації з обмеженим доступом та автоматизація її обробки: ІІІ наук.-техн. конф., 8-9 лютого 2011 р. : тези доп., К., 2011, С. 11-12.
91. Бессалов А.В., Телиженко А.Б. Криптосистемы на эллиптических кривых: Учеб. пособие. - К.: ІВЦ "Видавництво "Політехніка"", 2004. - 224 с.:іл.
92. А.О. Охріменко, «Аналіз методів оптимізації програмних реалізацій криптографічних алгоритмів», ПОЛІТ–2011. Сучасні проблеми науки: IX міжнар. наук.-практ. конф. молодих учених і студентів, 6-7 квітня 2011 р., К., 2011, С. 163-164.

93. V.Yu. Kovtun, M.G. Kovtun, A.O. Okhrimenko, «Commands integrity and authority in control radio link of UAV», Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), 2015 IEEE International Conference, 2015, pp. 178-181.
94. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих.
95. ДСТУ ISO/IEC 14888-3:2019 Інформаційні технології. Методи захисту. Цифрові підписи з доповненням. Частина 3. Механізми на основі дискретного логарифмування (ISO/IEC 14888-3:2018, IDT)
96. ДСТУ ETSI TS 119 312:2015 Електронні підписи й інфраструктури (ESI). Криптографічні комплекти (ETSI TS 119 312:2014, IDT)
97. RFC 3447. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography. Specifications Version 2.1
98. A.O. Okhrimenko, M.G. Kovtun, S.O. Gnatyuk, V.Yu. Kovtun, «Development of a search method of birationally equivalent binary Edwards curves for binary Weierstrass curves from DSTU 4145-2002», in Proc. of 2nd Intern. Scientific-Practical Conf. on the Problems of Infocommunications Science and Technology (PIC S&T), Kharkiv, Ukraine, October 13-15, 2015, pp. 5-8.
99. NIST SP 800-56A Rev. 3. Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography
100. RFC 6979. Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)
101. Rolf Oppliger. Contemporary Cryptography, Second Edition (Artech House Computer Security Series) 2nd Edition, 2012, 571 pages
102. Hankerson, D.; Vanstone, S.; Menezes, A. Guide to Elliptic Curve Cryptography. Springer Professional Computing. New York: Springer. - 2004. – 311 p. doi:10.1007/b97644
103. Ian F. Blake, Gadiel Seroussi, and Nigel Smart, editors, Advances in Elliptic Curve Cryptography, London Mathematical Society Lecture Note Series 317, Cambridge University Press, 2005

104. López, J. and Dahab, R. An Overview of Elliptic Curve Cryptography, Technical Report IC-00-10, State University of Campinas, 2000.
105. NIST FIPS 186-4. Digital Signature Standard (DSS)
106. IEEE Standard Specifications for Public-Key Cryptography," in IEEE Std 1363-2000 , vol., no., pp.1-228, 29 Aug. 2000
107. ГОСТ 34.10-2018. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи»
108. А.О. Охріменко, «Ефективна програмна реалізація арифметичних операцій в полі цілих чисел», Наукоємні технології: Наук.-техн. конф. студ. та мол. учених, 14-18 листопада 2011 р., К., 2012, С. 24.
109. Kaliski B. (2011) Moore's Law. In: van Tilborg H.C.A., Jajodia S. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA
110. Selecting Cryptographic Key Sizes, Arjen K. Lenstra and Eric R. Verheul, Journal Of Cryptology, vol. 14, p. 255-293, 2001
111. О.Г. Корченко, С.О. Гнатюк, Ю.Є. Хохлачова, А.О. Охріменко, «Основні критерії та вимоги до побудови сучасних криптосистем», Вісник Інженерної академії України, №3-4, С. 77-83, 2011.
112. NIST SP 800-57 Part 1. Revision 4. Recommendation for Key Management.
113. Yearly Report on Algorithms and Keysizes (2012), D.SPA.20 Rev. 1.0, ICT-2007-216676 ECRYPT II, 09/2012.
114. Algorithms, Key Size and Protocols Report (2018), H2020-ICT-2014 – Project 645421, D5.4, ECRYPT-CSA, 02/2018.
115. Mécanismes cryptographiques - Règles et recommandations, Rev. 2.03, ANSSI, 02/2014.
116. NSA. Commercial National Security Algorithm (CNSA) Suite Factsheet. January 2016. URL: https://apps.nsa.gov/iaarchive/library/ia-guidance/ia-solutions-for-classified/algorithm-guidance/commercial-national-security-algorithm-suite-factsheet.cfm

117. BSI TR-02102 Kryptographische Verfahren: Empfehlungen und Schlüssellängen. Version 2020-01.
118. NIST SP 800-131A Rev 2. Transitioning the Use of Cryptographic Algorithms and Key Lengths.
119. Ifrah G. The Universal History of Numbers: From Prehistory to the Invention of the Computer. 1st ed. Wiley, 2000. 656 pp.
120. Кнут Д.Э. Искусство программирования, том 2. Получисленные алгоритмы. 3-е изд. Москва: Вильямс, 2007. 832 с.
121. Задірака В.К., Олексюк О.С. Компютерна арифметика багаторозрядних чисел: Наукове видання. - Київ, 2003. - 264 с.
122. Евдокимов В.Ф., Стасюк А.И. Параллельные вычислительные структуры на основе разрядных методов вычислений. - К.: Наукова думка, 1987. - 310 с.
123. Bhattacharyya SS, Deprettere EF, Leupers R, Takala J, editors. Handbook of Signal Processing Systems. 2nd ed. Springer, 2013. 1399 pp.
124. Brent R.P., Zimmermann P. Modern Computer Arithmetic. New York: Cambridge University Press, 2011. 236 pp.
125. Guillermin N. A High Speed Coprocessor for elliptic curve scalar multiplication over Fp // Cryptographic Hardware and Embedded Systems - CHES 2010, Lecture Notes in Computer Science. Springer. 2010. Vol. 6225. pp. 48–64.
126. Jahani S., Samsudin A., and Govindarajan Subramanian K. Efficient Big Integer Multiplication and Squaring Algorithms for Cryptographic Applications // Journal of Applied Mathematics. 2014. Vol. 2014. P. 9.
127. Menezes A.J., Vanstone S.A., and Oorschot P.C.V. Handbook of Applied Cryptography. 1st ed. Boca Raton: CRC Press, 1996. 780 pp.
128. Yanik T., Savas E., and Koc C.K. Incomplete reduction in modular arithmetic // Computers and Digital Techniques, IEEE Proceedings. March 2002. Vol. 149. No. 2. pp. 46-52.
129. Ebeid N., Hasan M.A. On binary signed digit representations of integers // Designs, Codes and Cryptography. January 2007. Vol. 42. No. 1. pp. 43-65.

130. Gollmann D., Han Y., and Mitchell C. Redundant integer representation and fast exponentiation // Designs, Codes and Cryptography. 1998. No. 7. pp. 135–151.
131. Okeya K., Schmidt-Samoa K., Spahn C., and Takagi T. Signed Binary Representations Revisited // CRYPTO 2004, Lecture Notes in Computer Science. Santa Barbara. 2004. Vol. 3152. pp. 123-139.
132. Muir J.A. Efficient Integer Representations for Cryptographic Operations: Thesis requirement for the degree of Doctor of Philosophy in Combinatorics and Optimization. University of Waterloo, 2004.
133. Bajard J.C., Meloni N., and Plantard T. Efficient RNS Bases for Cryptography // IMACS'05 : World Congress: Scientific Computation, Applied Mathematics and Simulation. Paris. 2005.
134. Bajard J.C., Plantard T. RNS bases and conversions // Proceedings of the SPIE. 2004. Vol. 5559. pp. 60-69.
135. Soderstrand MA, Jenkins WK, Jullien GA, Taylor FJ, editors. Residue number system arithmetic: modern applications in digital signal processing. Piscataway: IEEE Press, 1986. 418 pp.
136. Oppenheim A., Schafer R., and Buck J. Discrete-Time Signal Processing. 3rd ed. Prentice Hall, 1999. 1120 pp.
137. Utku G., Selçuk B. Elliptic Curve Cryptography on Constrained Microcontrollers Using Frequency Domain Arithmetic // Computational Science and Its Applications (ICCSA 2014). Springer. 2014. Vol. 8584. pp. 493-506.
138. Николайчук Я.М., Касянчук М.М., Якименко І.З., Івасьєв С.В. Ефективний метод модулярного множення в теоретико-числовому базисі Радемахера–Крестенсона // Вісник Національного університету "Львівська політехніка". Комп’ютерні системи та мережі. 2014. No. 806. pp. 195-199.
139. Selçuk B., Berk S. Finite field polynomial multiplication in the frequency domain with application to elliptic curve cryptography // Proceedings of the 21th International Symposium on Computer and Information Sciences (ISCIS 2006). Springer. 2006. Vol. 4263. pp. 991-1001.

140. Francis J. Wright, James E. F. Skea. Mathematics and Algorithms for Computer Algebra. Course of lectures. 2003.

141. Качко, Е. Г. Распараллеливание алгоритмов умножения чисел многократной точности. Вестник Уфимского государственного авиационного технического университета, 15 (5 (45)), (2011). с.142-147.

142. Cohen Henri. Handbook of Elliptic and Hyperelliptic Curve Cryptography. / Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, Frederik Vercauteren – Chapman & Hall/CRC. – 2006. – 808 p.

143. Denis T., Rose G. BigNum Math: Implementing Cryptographic Multiple Precision Arithmetic. – Elsevier / Syngress. – 2006. – 336 p. ISBN 978-1-59749-112-9

144. Ковтун В.Ю. Подходы к повышению производительности программной реализации операции умножения в поле целых чисел / Ковтун В.Ю., Охрименко А.А., Нечипорук В.В. // – Защита информации. − №1 (54). – 2012. − С. 68-75.

145. Shahram Jahani. ZOT-MK: A NEW ALGORITHM FOR BIG INTEGER MULTIPLICATION. Thesis submitted in fulfillment of the requirements for the degree of Master of Science. June 2009

146. Hasselström, K.: Fast Division of Large Integers. Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Stockholm, Sweden (2003)

147. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии. — М.: МЦНМО, 2003. — 328 с. ISBN 5-94057-103-4.

148. А. Карацуба, Ю. Офман, “Умножение многозначных чисел на автоматах”, Докл. АН СССР, 145:2 (1962), 293–294

149. R. Crandall, C. Pomerance. Prime Numbers – A Computational Perspective. Second Edition, Springer, 2005.

150. Schönhage, A., Strassen, V.: Multiplikation großer Zahlen. Computing 7, 281–292 (1971)

151. Comba P.G. Exponentiation cryptosystems on the IBM PC // IBM Systems Journal. –Vol. 29(4), 1990. – P. 526–538.

152. Großschadl J. Energy-Efficient Software Implementation of Long Integer Modular Arithmetic / J.Großschadl, R.Avanzi, E.Sava, S.Tillich // Advances in Cryptology Prociding in CHES'2005. – Springer-Verlag. 2005. LNCS 3659. P.75-90.
153. Fürer, M.: Faster Integer Multiplication. Book Faster Integer Multiplication, Series Faster Integer Multiplication (2007)
154. А. Охрименко, В. Ковтун, «Умножения целых чисел с использованием отложенного переноса для криптосистем с открытым ключом», Информационные технологи и системы в управлении, образовании, науке: Монография, под ред. проф. В.С. Пономаренко, Харьков: Цифрова друкарня №1, 2013, С. 69-82.
155. Pascal Giorgi, Thomas Izard, Arnaud Tisserand. Comparison of Modular Arithmetic Algorithms on GPUs. ParCo'09: International Conference on Parallel Computing, France.
156. Hong S-M. New Modular Multiplication algorithms for fast modular exponeniation / S-M.Hong, S-Y.Oh, H.Yoon // Advances in Cryptology Proceedings of Eurocrypt '96. – Springer-Verlag, 1996. – P.166-177.
157. Paar, C. 1999. Implementation options for finite field arithmetic for elliptic curve cryptosystems. In Proceedings of the 3rd Workshop on Elliptic Curve Cryptography (ECC'99).
158. J Großschädl, E Savaş. Instruction Set Extensions for Fast Arithmetic in Finite Fields GF(p) and GF(2m). International Workshop on Cryptographic Hardware and Embedded Systems CHES2004, pp.133-147
159. А.А. Охрименко, «Обобщенные алгоритмы возведения в квадрат целых чисел с использованием отложенного переноса и технологий распараллеливания», Вісник Інженерної академії України, № 1, С. 114-119, 2014.
160. А. Охрименко,В. Ковтун, «Метод повышения производительности операции приведения по простому модулю», Информационные системы в управлении, образовании, промышленности: Монография, под ред. В.С. Пономаренко, Харьков: Вид-во ТОВ «Щедра садиба плюс», 2014, С. 204-219.

161. Гнатюк С. Подходы к повышению производительности расширенного алгоритма Евклида для деления больших чисел двойной точности на большие числа одинарной точности / С. Гнатюк, В. Ковтун, О. Бердник, М. Ковтун // Безпека інформації. - 2015. - Т. 21, № 1. - С. 40-51.

162. L. Hars. Long Modular Multiplication for Cryptographic Applications.//Cryptographic Hardware and Embedded Systems - CHES 2004.//LNCS Volume 3156, 2004, pp 45-61

163. Solinas, J.A.: Generalized Mersenne numbers. Technical Report CORR 99-39, Centre for Applied Cryptographic Research, University of Waterloo (1999)

164. Huapeng Wu. On Computation of Polynomial Modular Reduction. Technical report, Centre of Applied Cryptographic Research, University of Waterloo, June 2000.

165. P. Giorgi, L. Imbert, T. Izard. Multipartite Modular Multiplication. RR-11024, 2011, pp.25

166. S. Gueron. Enhanced Montgomery Multiplication. Cryptographic Hardware and Embedded Systems — CHES 2002. Lecture Notes in Computer Science Volume 2523, 2003, pp 46-56.

167. Montgomery, P.L.: Modular Multiplication Without Trial Division. Mathematics of Computation 144(170), 519–521 (1985)

168. W. Hasenplaugh, G. Gaubatz, V. Gopal. Fast Modular Reduction. Proceedings of the 18th IEEE Symposium on Computer Arithmetic, pp 225-229 . IEEE Computer Society Washington, DC, USA 2007

169. P. Barrett. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. Proceedings CRYPTO'86, pp. 311-323.

170. Stehlé D., Zimmermann P. A Binary Recursive Gcd Algorithm. In: Buell D. (eds) Algorithmic Number Theory. ANTS 2004. Lecture Notes in Computer Science, vol 3076. Springer, Berlin, Heidelberg

171. D. Takahashi. A Parallel Algorithm for Multiple-Precision Division by a Single-Precision Integer. Large-Scale Scientific Computing. LNCS Volume 4818, 2008, pp 729-736.

172. Jebelean, Tudor. “An Algorithm for Exact Division.” J. Symb. Comput. 15 (1993): 169-180.
173. C.H. Lim, H.S. Hwang and P.J. Lee, Fast modular reduction with precomputation, Proc. of 1997 Korea-Japan Joint Workshop on Information Security and Cryptology (JW-ISC’97), Oct. 26–28, 1997, pp.65–79.
174. Burnikel, C., Ziegler, J.: Fast recursive division. Research Report MPI-I-98-1-022, MPI Saarbrücken, 1998.

РОЗДІЛ 2
РОЗРОБКА МЕТОДУ ПРЕДСТАВЛЕННЯ ЦІЛИХ ЧИСЕЛ З ВІДКЛАДЕНИМ ПЕРЕНОСОМ ТА МЕТОДІВ АРИФМЕТИЧНИХ ПЕРЕТВОРЕНЬ НАД НИМИ

2.1. Представлення цілих чисел з відкладеним переносом

Не дивлячись на широке застосування описаних в попередньому розділі представлень цілих чисел, вони мають ряд недоліків. Зокрема, вузька сфера застосування, складність реалізації деяких операцій, низька ефективність при реалізації на сучасних мікропроцесорах [1-4]. У всіх арифметичних операціях в приведених представленнях присутня операція переносу чи займу, яка негативно позначається на їх швидкодії на сучасних процесорах (не виконується умова спарювання команд). Добитись суттєвого підвищення швидкодії в операціях з цілими числами, можна шляхом відкладення виконання операції переносу з старших розрядів в молодші, а для займу навпаки – з молодших розрядів в старші [5, 6].

Пропонується нове представлення цілих чисел з відкладеним переносом Delayed Carry Form (DCF) [7-11], Рис. 2.1.

Двійкове число $a = \{a_{n-1}, \ldots, a_1, a_0\}$ в DCF представленні являє собою послідовність з машинних слів $d_{\text{DCF}} = \{d_{m-1}, \ldots, d_1, d_0\}_{\text{DCF}}$ розміром w –біт, таке, що кожне машинне слово $d_i^{(w)} = a_i^{(r)} \,||\, a_i^{(v)}$, де $a_i^{(r)}$ блок довжиною r біт, виділених під перенос, а $a_i^{(v)}$ блок довжиною v біт, заповнюється бітами з числа a (де $w = r + v$). В DCF-представленні, звичайне число a двійкової довжини l, в вигляді m блоків довжиною v біт, буде містити $m \cdot r$ біт надлишкової інформації, відведеної для зберігання переносів.

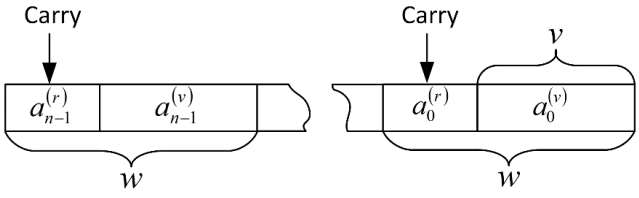


Рис. 2.1. Представлення цілих чисел в DCF

Слід зазначити, що число m може бути розраховано виходячи з міркувань, що кількість інформаційних біт в двійковому представленні має відповідати числу інформаційних біт в DCF представлені: $m \cdot v = n \cdot w$.

Таки чином: $m = \dfrac{n \cdot w}{v} = \dfrac{n \cdot w}{w - r}$.

В DCF представленні, процесор оперує машинними словами, в яких виділені біти для накопичення переносу і для зберігання безпосередньо самого числа, тобто для процесора операції над машинними словами, що містять переноси виглядають прозоро.

2.1.1. Перетворення двійкового числа в DCF представлення

Метод 2.1. *Метод перетворення цілих чисел в DCF представлення*

Для перетворення двійкового числа $a_i^{(w)}$ в DCF представлення необхідно зарезервувати (обнулити) в машинному слові $d_i^{(w)}$ довжиною w–біт, r – біт під перенос $a_i^{(r)}$, а v – біт, що залишилися, заповнюються бітами з числа $a_i^{(v)}$ в неперервній двійковій формі (Рис. 2.2).

Вхід: $a \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = v$, $i = \overline{0, n-1}$, $w = v + r$, w–довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, $m = \left\lceil \dfrac{n \cdot w}{w - r} \right\rceil$ – кількість машинних слів, що необхідна для представлення числа в DCF. *Вихід*: $d = [a]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{m-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, m-1}$.
1. For $i \leftarrow 0$, $i < m$, $i{+}{+}$. 1.1. $d_i^{(w)} \leftarrow a_0^{(v)}$.
1.2. $a \leftarrow (a >> v)$. 2. Return (d).

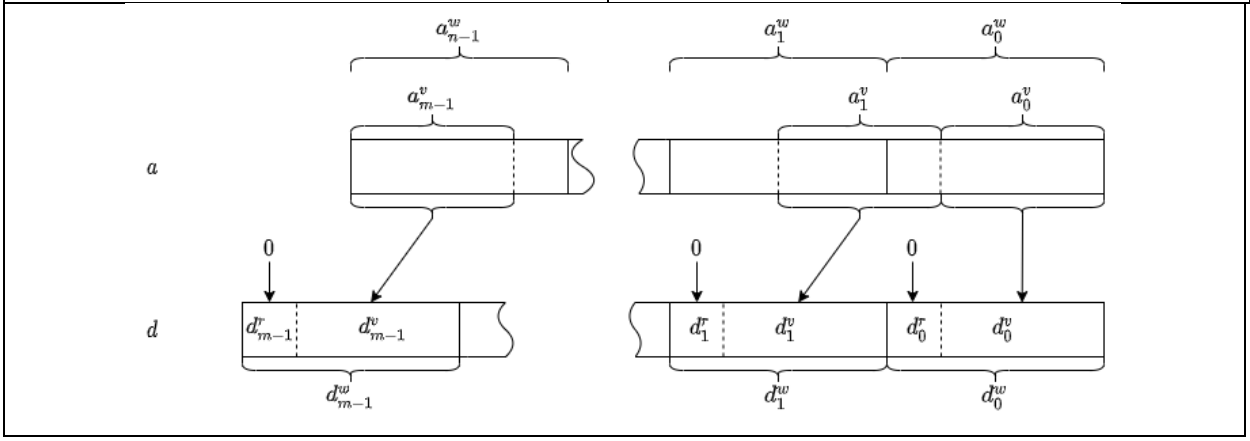


Рис. 2.2. Псевдокод методу перетворення двійкового числа в DCF представлення

Обчислювальна складність запропонованого методу перетворення двійкового числа в DCF представлення: $I_{DCF}\left(A_{2.1}\right)=m\left(2I^{w}_{a\,\mathrm{sgn}}+I_{shr}\right)$.

2.1.2. Перетворення DCF числа в двійкове представлення (коригування переносів)

Метод 2.2. *Метод коригування переносів DCF-числа*

Для перетворення числа з DCF представлення в двійкову неперервну форму, необхідно виконати коригування переносів – ітеративно врахувати перенос з молодшого машинного слова в старше. Найстаршим переносом можна знехтувати, так як він виходить за межі розрядної сітки числа.

Вхід: $a\in Z_{DCF}$, $a=\{a_{n-1},\ldots,a_1,a_0\}$, $\log_2 a_i=w$, $i=\overline{0,n-1}$, $w=v+r$, $x=v/w$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, може бути від'ємним.

Вихід: $d=Z$, $d=\{d_{l-1},\ldots,d_1,d_0\}$, $\log_2 d_i=w$, $i=\overline{0,l-1}$, $l=\left\lceil\frac{nv}{w}\right\rceil$.

1. $t^{(r)}\leftarrow a_0^{(r)}$.
2. $d_0^{(w)}\leftarrow a_0^{(v)}$.
3. For $i\leftarrow 1$, $i<n$, $i{+}{+}$.
3.1. $t^{(r)}\,\|\,t^{(v)}\leftarrow a_i^{(w)}+t^{(r)}$
3.2. $j\leftarrow\lfloor i\cdot x\rfloor$.
3.3. $s\leftarrow(r\cdot i)\bmod w$.
3.4. $d_j^{(w)}\leftarrow d_j^{(w)}\,|\left(t^{(v)}<<(w-s)\right)$.
3.5. if $(s>0)$ then $d_{j+1}^{(w)}\leftarrow d_{j+1}^{(w)}\,|\left(t^{(v)}>>s\right)$.
4. Return (d).

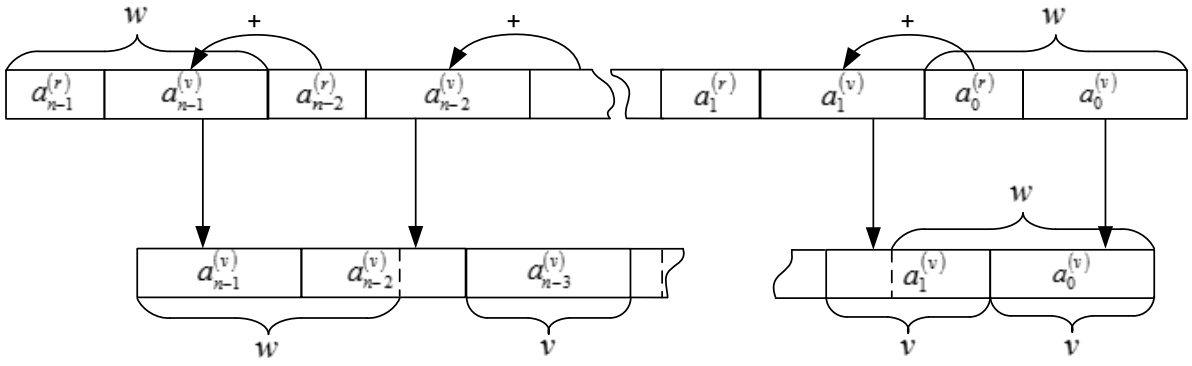


Рис. 2.3. Метод коригування переносів

На Рис. 2.3 демонструється робота методу коригування переносу $a_i^{(r)}\,\|\,a_i^{(v)}\leftarrow a_i^{(w)}+a_{i-1}^{(r)}$. З методу слідує, що операція коригування переносів виконується послідовно, від молодших машинних слів до старших.

Обчислювальна складність методу коригування представлена нижче:

$$I_{corr}\left(A_{2.2}\right)=I_{a\text{sgn}}^{w}+n\left(5I_{a\text{sgn}}^{w}+4I_{add}^{w}+I_{sub}^{w}+2I_{mul}^{w}+I_{\text{mod}}^{w}+I_{cmp}^{w}+2I_{sh}^{w}\right).$$

2.1.3. Особливості використання чисел в DCF представленні

При використанні цілих чисел в DCF представленні присутня певна надлишковість інформації, котра визначається кількістю біт r, виділених під блок переносу в кожному машинному слові.

Кількість біт r, виділених під зберігання переносу слід визначати з міркувань продуктивності. Як правило виконується вирівнювання по межі одного байту, двох байт, чотирьох байт або ж із міркувань зберігання максимально можливого розміру накопиченого переносу. Так, для зберігання переносів в діапазоні $\{-127,128\}$, які можуть бути накопичені в результаті 128 операцій додавання та 127 операцій віднімання, або 255 операцій додавання/віднімання, необхідно виділити 1 байт під перенос. Також рекомендується виконувати проміжне коригування переносів (врахування переносів), при досягненні можливої межі.

Розмір числа в DCF представленні можна обчислити за допомогою виразу:

$$m=\left\lceil \frac{n\cdot w}{w-r} \right\rceil, \qquad (1)$$

де n та m – кількість машинних слів розміром w біт, необхідних для представлення цілого числа в двійковій неперервній формі та DCF відповідно, r – кількість біт, виділених під перенос в машинному слові, причому $r<w$.

Аналітична оцінка надлишковості (в машинних словах) представлення цілих чисел в DCF може бути обчислена:

$$R\left(d_{DCF}\right)=m-n=\left\lceil n\cdot\left(\tfrac{w}{w-r}-1\right)\right\rceil. \qquad (2)$$

Відповідно, аналітична оцінка надлишковості (в бітах), представлення цілих чисел в DCF може бути обчислена:

$$R'\left(d_{DCF}\right)=w\cdot\left(m-n\right)=\left\lceil n\cdot w\cdot\left(\tfrac{w}{w-r}-1\right)\right\rceil.$$

В Табл. 2.1 приведена аналітична оцінка довжини числа в DCF представленні та оцінка його надлишковості, відповідно до (1) та (2).

Знаками «+» помічені довжини машинних слів, котрі підтримують блоки переносів заданого розміру в бітах.

Таблиця 2.1

Аналітичні оцінки довжини DCF представлення числа та його надлишковості в залежності від довжини блоку відкладеного переносу

Довжина блоку для відкладеного переносу, біт	Довжина машинного слова		Надлишковість, кількість слів		Довжина числа в DCF, кількість слів	
	32 біт	64 біт	32 біт	64 біт	32 біт	64 біт
8	+	+	$n/3$	$n/7$	$4n/3$	$8n/7$
16	+	+	$n/2$	$n/3$	$3n/2$	$4n/3$
24	+	+	$3n$	$3n/5$	$4n$	$8n/5$
32	–	+	–	n	–	$2n$

Не дивлячись на приведені в Табл. 2.1 рекомендації, для зберігання переносів може бути відведено і більша кількість біт, проте при цьому слід враховувати, що відповідним чином росте надлишковість при зберіганні чисел в DCF представленні. З іншого боку, збільшення числа розрядів для зберігання відкладеного переносу, дозволяє на більше число арифметичних операцій відкласти операцію коригування переносів.

Особливістю запропонованого DCF представлення цілих чисел є відсутність необхідності врахування переносів та займів, що дозволяє позбавитися від зайвих операцій присвоєння та перевірок при реалізації на мовах програмування високого рівня, а також від аналізу регістра позначок на можливий перенос. В свою чергу, це приводить до підвищення ефективності програмної реалізації на процесорах з суперскалярною архітектурою та можливостями сучасних компіляторів по передбаченню переходів, паралельному виконанню команд, розгортанню циклів і т.д. Виключенням є операція коригування переносів, яка виконується послідовно від молодших машинних слів до старших.

Враховуючи специфіку DCF, коли порядок виконання операцій над машинними словами не є суттєвим, стає очевидною можливість паралельного виконання операцій над відповідними словами. В програмній реалізації можливо

застосовувати схему розпаралелювання з використанням множини потоків, наприклад, з використанням технології OpenMP, або аж до виконання кожної операції в окремому потоці з використанням технології NVIDIA CUDA (Рис. 2.4).

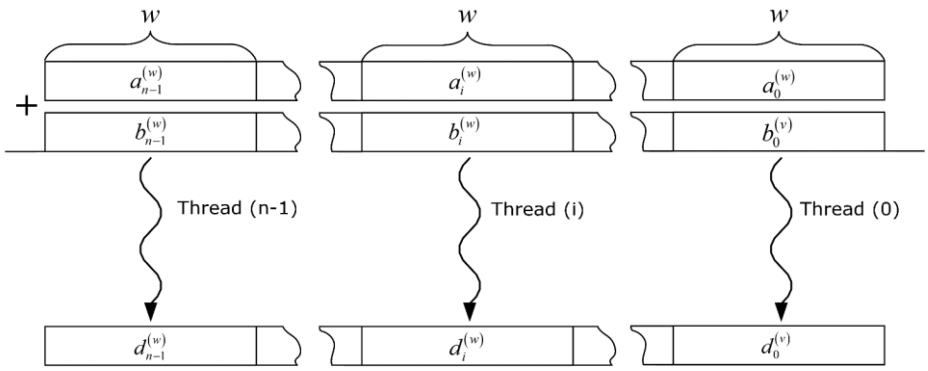


Рис. 2.4. Модель розпаралелювання на основі технології NVIDIA CUDA

2.2. Операції з числами в DCF представленні

Для виконання операцій з числами в DCF представленні, необхідно модифікувати звичні методи арифметичних перетворень з цілими числами [7, 11]. Нижче запропоновані методи основних арифметичних операцій, в яких використовуються числа в DCF представленні.

2.2.1. Додавання

Метод 2.3. *Метод додавання цілих чисел з відкладеним переносом*

Розглянемо метод додавання цілих чисел, в якому доданки та сума представляються в DCF. На вхід подаються два цілих числа в DCF представленні $a, b \in Z_{DCF}$, які представлені як послідовність з n машинних слів, розміром w біт кожне. Виконується операція додавання в циклі в інтервалі $[\overline{0, n-1}]$ відповідних машинних слів двох чисел. На виході отримуємо результат – ціле число в DCF представленні $d \in Z_{DCF}$.

Вхід: $a, b \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = w$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу. *Вихід*: $d = [a+b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $d_i^{(w)} \leftarrow a_i^{(w)} + b_i^{(w)}$.	2. Return (d).

Рис. 2.5. Псевдокод методу додавання цілих чисел з відкладеним переносом

Виходячи з особливостей представлення доданків $a_i^{(w)} \leftarrow a_i^{(r)} \| a_i^{(v)}$, $b_i^{(w)} \leftarrow b_i^{(r)} \| b_i^{(v)}$ і суми $d_i^{(w)} \leftarrow d_i^{(r)} \| d_i^{(v)}$, класичний запис додавання чисел в DCF (Рис. 2.6), може бути представлений в спрощеній формі (Рис. 2.6).

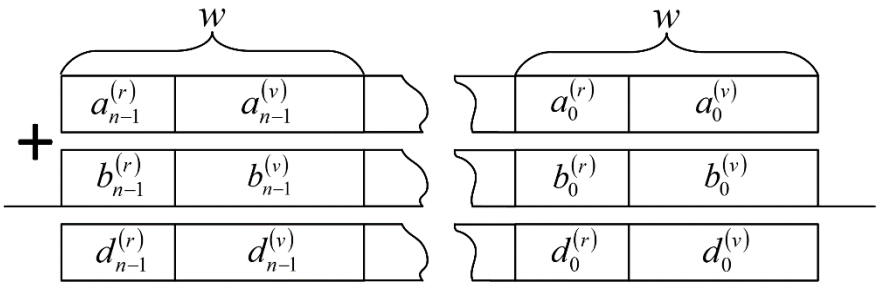


Рис. 2.6. Додавання чисел в DCF представленні

Даний метод є найбільш простим серед запропонованих методів додавання, так як не оперує бітами, в тому числі і переносами:

$$d_i^{(r)} \| d_i^{(v)} \leftarrow a_i^{(r)} \| a_i^{(v)} + b_i^{(r)} \| b_i^{(v)}.$$

У спрощеній формі (Рис. 2.7), додаються відразу машинні слова $d_i^{(w)} \leftarrow a_i^{(w)} + b_i^{(w)}$, без виділення областей зберігання самого числа і переносу, що дозволяє досягнути максимальної продуктивності при його реалізації на сучасних процесорах. Процесори оперують виключно машинними словами, при цьому перенос з молодшого машинного слова в старше не виконується.

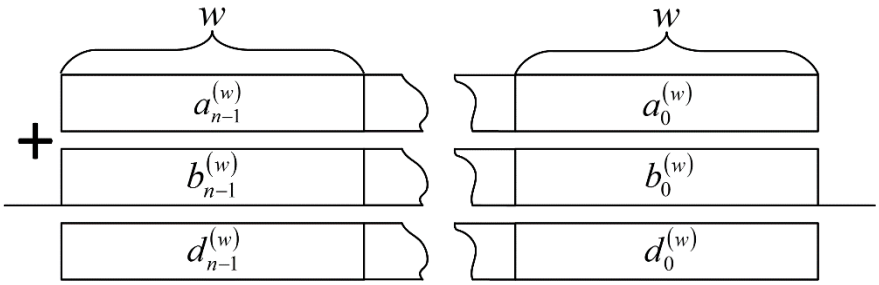


Рис. 2.7. Спрощена форма додавання чисел в DCF

Обчислювальна складність запропонованого методу додавання двох чисел в DCF представленні:

$$I_{add}\left(A_{2.3}\right) = n\left(I_{add}^{w} + I_{a\,\mathrm{sgn}}^{w}\right).$$

2.2.1.1. Змішане додавання

Існує два варіанти змішаного додавання цілих чисел:

- доданки представляються в звичній двійковій (неперервній) формі, а сума – в DCF;
- один з доданків представляється в DCF, другий доданок і сума – в неперервній двійковій формі.

Розглянемо метод додавання, в якому доданки представляються в звичній двійковій (неперервній) формі, а сума – в DCF. Відмінністю методу змішаного додавання від звичного методу додавання полягає в тому, що в процесі додавання, доданки перетворюються в DCF у відповідності до Методу 2.1, і лише потім виконується їх додавання.

Виходячи з особливостей представлення доданків $a_i^{(w)} \leftarrow a_i^{(r)} \| a_i^{(v)}$, $b_i^{(w)} \leftarrow b_i^{(r)} \| b_i^{(v)}$, де області переносів ініціалізуються наступним чином $a_i^{(r)} \leftarrow 0$, $b_i^{(r)} \leftarrow 0$ і суми $d_i^{(w)} \leftarrow d_i^{(r)} \| d_i^{(v)}$, класичний запис змішаного додавання чисел в DCF представленні показаний на Рис. 2.8.

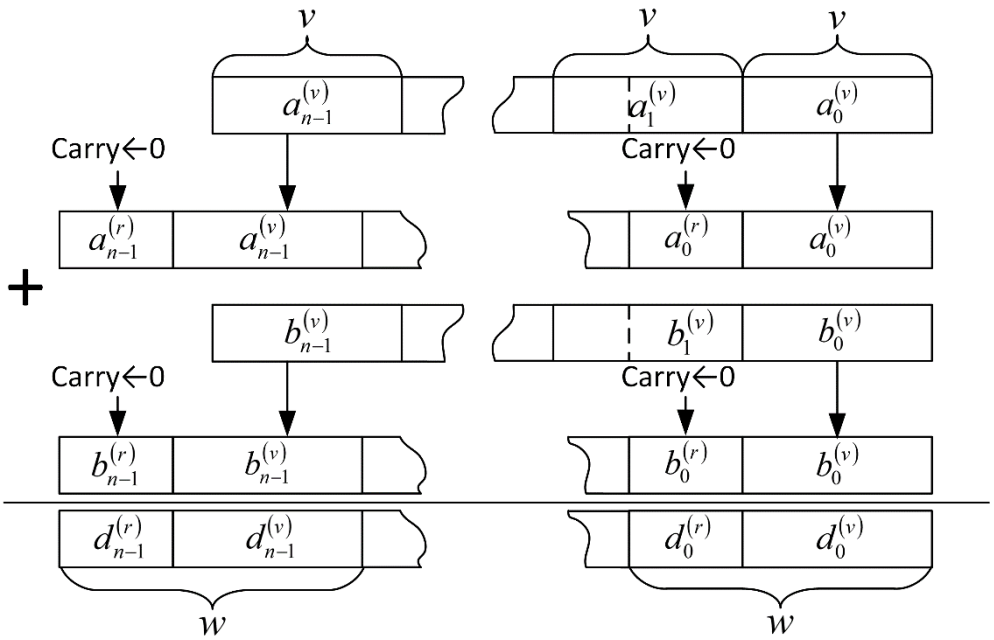


Рис. 2.8. Змішане додавання чисел

Метод 2.4 відрізняється від Методу 2.3 тим, що виконується виділення v біт з кожного доданку в звичному двійковому представленні і заповнення нулями r біт, що залишилися.

Дане перетворення є досить трудомістким при програмній реалізації, що потребує вибирати область для зберігання переносів довжиною r біт, кратну байту.

Метод 2.4. *Метод змішаного додавання цілих чисел з відкладеним переносом*

Вхід: $a, b \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = v$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $a_i^{(r)} \leftarrow 0$, $b_i^{(r)} \leftarrow 0$.	1.2. $d_i^{(w)} \leftarrow a_i^{(r)} \,\|\, a_i^{(v)} + b_i^{(r)} \,\|\, b_i^{(v)}$. 2. Return (d).

Рис. 2.9. Псевдокод методу змішаного додавання цілих чисел з відкладеним переносом

Обчислювальна складність запропонованого методу змішаного додавання двох чисел:

$$I_{add}\left(A_{2.4}\right) = n\left(I_{add}^{w} + 3I_{a\,\mathrm{sgn}}^{w} + 2I_{or}^{w}\right).$$

Також, слід розглянути метод, в якому в якості доданків використовується число в DCF і число в неперервній двійковій формі, а сума – в DCF представленні.

Виходячи з особливостей представлення доданків $a_i^{(w)} \leftarrow a_i^{(r)} \,\|\, a_i^{(v)}$ і $b_i^{(w)} \leftarrow b_i^{(r)} \,\|\, b_i^{(v)}$, де область переносу одного з доданків в двійковій неперервній формі ініціалізується наступним чином $b_i^{(r)} \leftarrow 0$ і суми $d_i^{(w)} \leftarrow d_i^{(r)} \,\|\, d_i^{(v)}$, класичний запис додавання чисел в DCF представленні (Рис. 2.6) можна представити в спрощеній формі (Рис. 2.10).

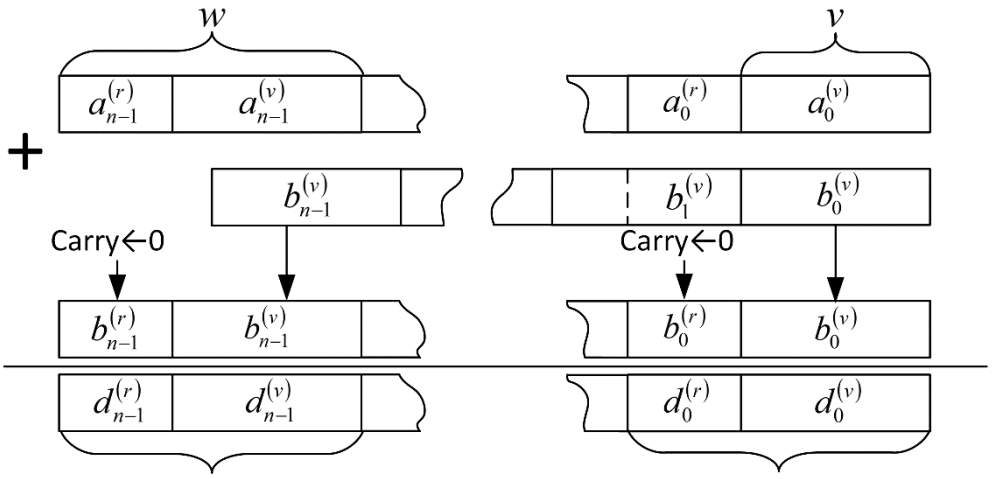


Рис. 2.10. Спрощена форма змішаного додавання

Метод 2.5. *Метод змішаного додавання цілого числа та числа в DCF*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b \in Z$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = w$, $\log_2 b_i = v$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $b_i^{(r)} \leftarrow 0$. 1.2. $d_i^{(w)} \leftarrow a_i^{(r)} \,\|\, a_i^{(v)} + b_i^{(r)} \,\|\, b_i^{(v)}$. 2. Return (d).

Рис. 2.11. Псевдокод методу змішаного додавання цілого числа та числа в DCF

Обчислювальна складність запропонованого методу змішаного додавання двох чисел:

$$I_{add}(A_{2.5}) = n\left(I_{add}^{w} + 2I_{a\,\mathrm{sgn}}^{w} + 2I_{or}^{w}\right).$$

2.2.2. Віднімання

Другою за значимістю операцією з цілими числами, після додавання, є віднімання. При відніманні, перенос (займ) виконується не з молодших розрядів в старші, а навпаки – з старших розрядів в молодші. При відніманні, значення переносу може ставати від'ємним.

Розглянемо метод віднімання цілих чисел, в якому зменшуване, від'ємник і різниця представлені в DCF. На вхід подаються два цілих числа в DCF представленні $a, b \in Z_{DCF}$, які представлені як послідовність з n

машинних слів, розміром w біт кожне. Операція віднімання виконується в циклі в інтервалі $[\overline{0,n-1}]$ відповідних машинних слів двох чисел. На виході отримуємо результат – ціле число в DCF представленні $d \in Z_{DCF}$.

Виходячи з особливостей представлення зменшуваного $a_i^{(w)} \leftarrow a_i^{(r)} \| a_i^{(v)}$, від'ємника $b_i^{(w)} \leftarrow b_i^{(r)} \| b_i^{(v)}$ и різниці $d_i^{(w)} \leftarrow d_i^{(r)} \| d_i^{(v)}$, класичний запис віднімання чисел в DCF представленні (Рис. 2.12), може бути представлена в спрощеній формі (Рис. 2.13).

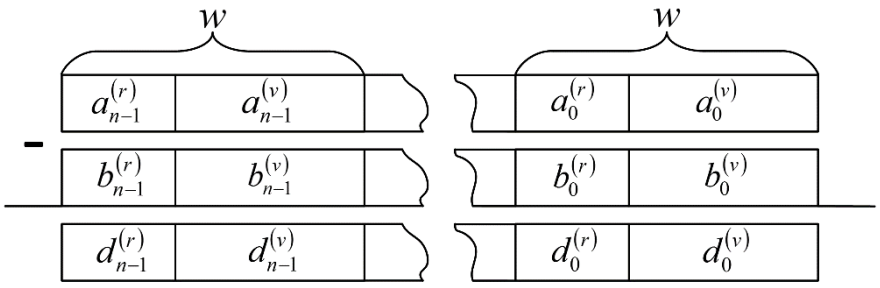


Рис. 2.12. Віднімання чисел в DCF представленні

Даний метод є найбільш простим, серед запропонованих методів віднімання, так як не оперує бітами (в тому числі і займами): $d_i^{(r)} \| d_i^{(v)} \leftarrow a_i^{(r)} \| a_i^{(v)} - b_i^{(r)} \| b_i^{(v)}$, В спрощеній формі (Рис. 2.13), віднімаються відразу машинні слова $d_i^{(w)} \leftarrow a_i^{(w)} - b_i^{(w)}$, без виділення областей зберігання самого числа і займа, що дозволяє досягнути максимальної продуктивності при його реалізації на сучасних процесорах. Процесори оперують виключно машинними словами, при цьому займ з старшого машинного слова в молодше не виконується.

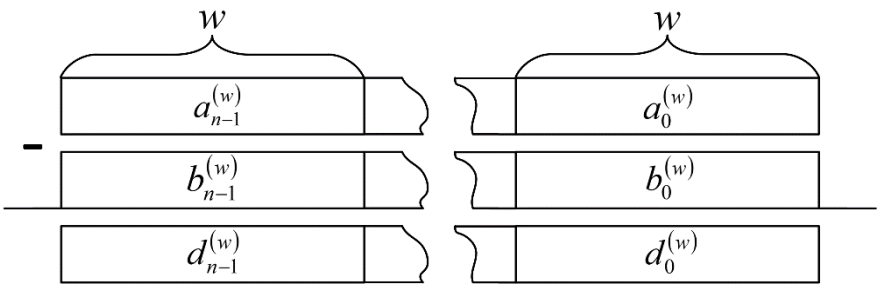


Рис. 2.13. Спрощена форма віднімання чисел в DCF представленні

Метод 2.6. *Метод віднімання цілих чисел з відкладеним займом*

Вхід: $a, b \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = w$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, може бути від'ємним. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $d_i^{(w)} \leftarrow a_i^{(w)} - b_i^{(w)}$.	2. Return (d).

Рис. 2.14. Псевдокод методу *віднімання цілих чисел з відкладеним займом*

Обчислювальна складність запропонованого методу віднімання двох чисел: $I_{sub}(A_{2.6}) = n\left(I_{sub}^{w} + I_{a\,\mathrm{sgn}}^{w}\right)$.

2.2.2.1.Змішане віднімання

Існує три варіанти змішаного віднімання чисел:

- зменшуване та від'ємник представляються в звичайній двійковій формі, а різниця – в DCF представленні;
- зменшуване представлено в DCF, від'ємник представляється в двійковій формі, а різниця – в DCF представленні;
- зменшуване представлено в двійковій формі, а від'ємник і різниця представляються в DCF.

Розглянемо метод віднімання, в якому зменшуване та від'ємник представляються в двійковій формі, а різниця – в DCF представленні (Рис. 2.15, Метод 2.7). Відмінністю методу змішаного віднімання від звичайного методу віднімання полягає в тому, що в процесі віднімання зменшуване та від'ємник представляються в DCF, у відповідності до Методу 2.1, і лише потім виконується їх віднімання.

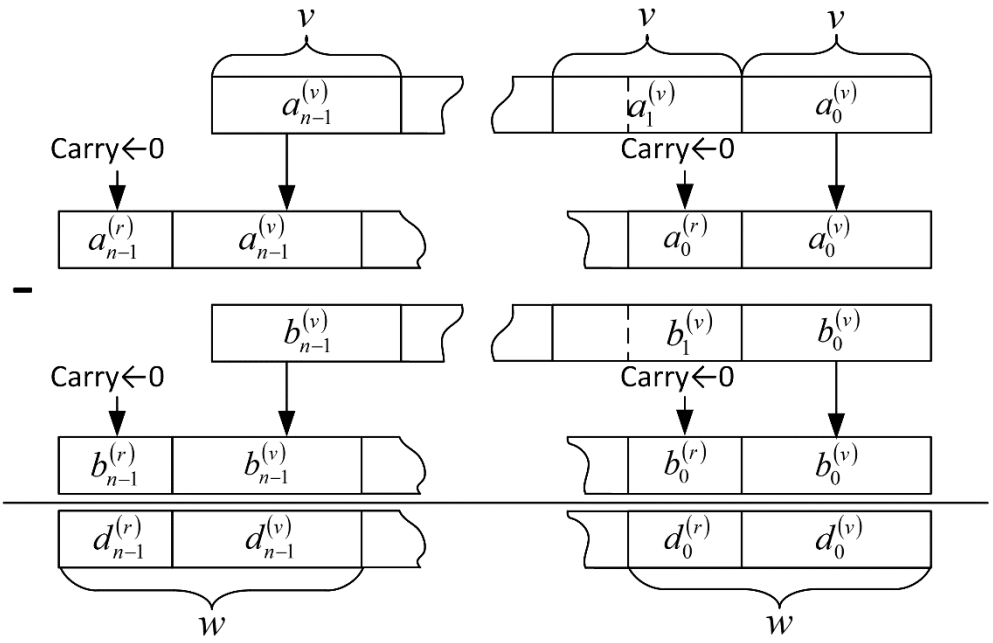


Рис. 2.15. Змішане віднімання двійкових чисел в DCF представленні

Метод 2.7. *Метод змішаного віднімання цілих чисел з відкладеним переносом*

Вхід: $a, b \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = \log_2 b_i = v$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, може бути від'ємним. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $a_i^{(r)} \leftarrow 0$, $b_i^{(r)} \leftarrow 0$.	1.2. $d_i^{(w)} \leftarrow a_i^{(r)} \,\|\, a_i^{(v)} - b_i^{(r)} \,\|\, b_i^{(v)}$. 2. Return (c).

Рис. 2.16. Псевдокод методу змішаного віднімання цілих чисел з відкладеним переносом

Обчислювальна складність запропонованого методу змішаного віднімання двох чисел: $I_{sub}(A_{2.7}) = n\left(I_{sub}^{w} + 3I_{a\text{sgn}}^{w} + 2I_{or}^{w}\right)$.

Розглянемо метод віднімання, в якому зменшуване представлене в DCF, а від'ємник представляється в двійковій формі (Рис. 2.17, Метод 2.8). Різниця представляється в DCF. Відмінністю від попереднього методу змішаного віднімання полягає в тому, що в процесі віднімання, від'ємник перетворюється в DCF у відповідності до Методу 2.1, і лише потім виконується віднімання.

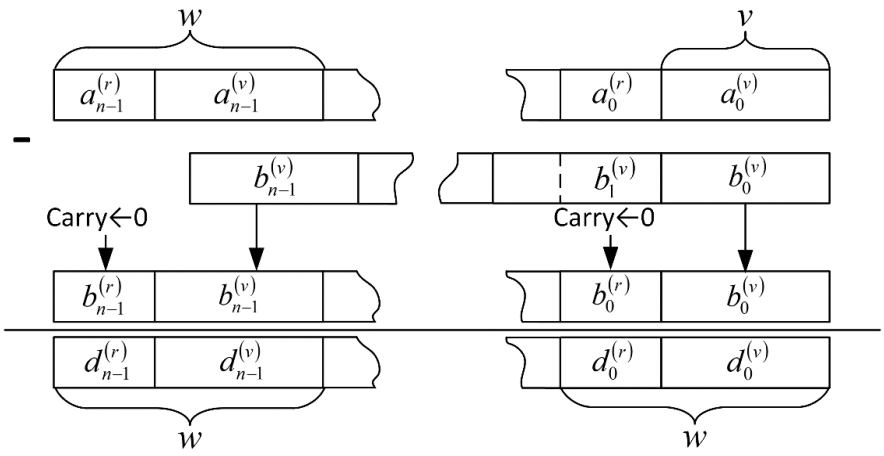


Рис. 2.17. Змішане віднімання числа в DCF та числа в двійковому представленні

Метод 2.8. *Метод змішаного віднімання числа в DCF представленні та числа в двійковому представленні з відкладеним переносом*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b \in Z$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = w$, $\log_2 b_i = v$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, що може бути від'ємним. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \| d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $b_i^{(r)} \leftarrow 0$.	1.2. $d_i^{(w)} \leftarrow a_i^{(r)} \| a_i^{(v)} - b_i^{(r)} \| b_i^{(v)}$. 2. Return (d).

Рис. 2.18. Псевдокод методу 2.8

Обчислювальна складність запропонованого методу змішаного віднімання числа в DCF та цілого числа в двійковому представленні:

$$I_{sub}(A_{2.8}) = n\left(I_{sub}^{w} + 2I_{a\,\mathrm{sgn}}^{w} + 2I_{or}^{w}\right).$$

Розглянемо метод віднімання, в якому зменшуване представлене в двійковій формі, а від'ємник представляється в DCF (Рис. 2.19, 2.9). Різниця представляється в DCF. В процесі віднімання, зменшуване перетворюється в DCF у відповідності до Методу 2.1, і лише потім виконується віднімання.

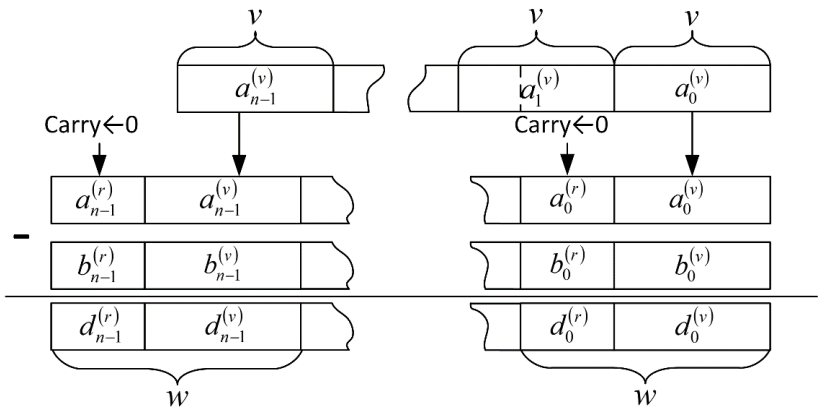


Рис. 2.19. Змішане віднімання двійкового числа та числа в DCF

Метод 2.9. *Метод змішаного віднімання двійкового числа та числа в DCF з відкладеним переносом*

Вхід: $a \in Z$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b \in Z_{DCF}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $\log_2 a_i = v$, $\log_2 b_i = w$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, що може бути від'ємним. *Вихід*: $d = [a + b]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow 0$, $i < n$, $i{+}{+}$. 1.1. $a_i^{(r)} \leftarrow 0$.	1.2. $d_i^{(w)} \leftarrow a_i^{(r)} \,\|\, a_i^{(v)} - b_i^{(r)} \,\|\, b_i^{(v)}$. 2. Return (d) .

Рис. 2.20. Псевдокод методу 2.9

Обчислювальна складність запропонованого методу змішаного віднімання двійкового числа та числа в DCF:

$$I_{sub}(A_{2.9}) = n\left(I_{sub}^{w} + 2I_{a\,\mathrm{sgn}}^{w} + 2I_{or}^{w}\right).$$

2.2.3. Зсув вліво на c біт

Існує два варіанти зсуву вліво [12]:

- звичайний зсув вліво (число, яке зсувається представлене в DCF);
- змішаний зсув вліво (число, яке зсувається представлене в двійковій неперервній формі, необхідне попереднє перетворення в DCF).

Розглянемо метод зсуву цілого числа вліво на c біт, в якому число, яке зсувається і результат зсуву представляються в DCF. На вхід подається ціле число в DCF представленні $a \in Z_{DCF}$, яке представлене як послідовність з n

машинних слів, розміром w біт кожне. Виконується операція зсуву на c біт в циклі в інтервалі $[\overline{n-1},0)$ відповідних машинних слів. На виході отримуємо результат – ціле число в DCF представленні $d \in Z_{DCF}$.

При виконанні зсуву слід звернути увагу на можливе переповнення розрядів відведених під відкладений перенос та вихід за межі розрядної сітки.

При зсуві числа в DCF на c біт (Рис. 2.21) слід розглянути два випадки. В першому випадку, число в DCF нормалізоване, тобто області, що відведені під перенос пусті (заповнені нулями). В другому випадку, DCF-число не нормалізовані, тобто області, що відведені під перенос заповнені двійковими даними. Перед виконанням зсуву, не нормалізоване число в DCF необхідно або нормалізувати (врахувати переноси та очистити області переносу від даних) або збільшити області переносів, оскільки можливі їх переповнення.

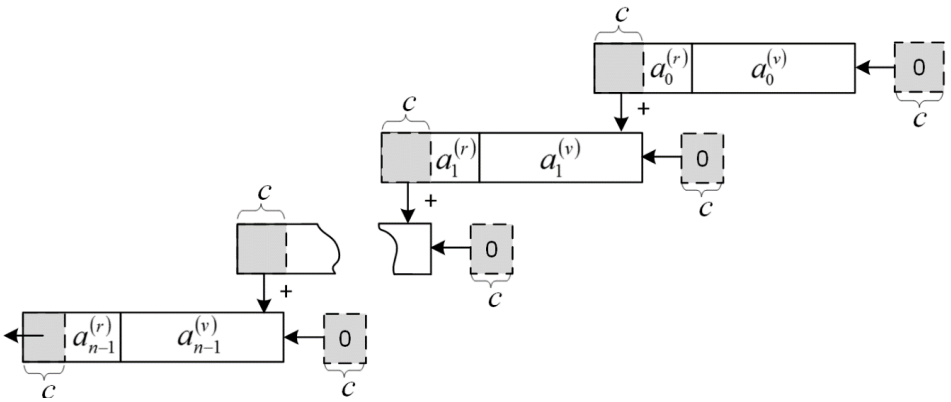


Рис. 2.21. Зсув числа в DCF вліво

Метод 2.10. *Метод зсуву числа в DCF вліво*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, $i = \overline{0, n-1}$, $w = v + r$, $c \le r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, c – кількість біт зсуву. *Вихід*: $d = [a << c]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.
1. For $i \leftarrow n-1$, $i <> 0$, $i--$. 1.1. $d_i^{(w)} \leftarrow (a_i^{(w)} << c) + (a_{i-1}^{(w)} >> (w-c))$. 2. Return (d).

Рис. 2.22. Псевдокод методу зсуву числа DCF вліво

Обчислювальна складність запропонованого методу зсуву вліво:

$$I_{shl}(A_{2.10}) = n\left(2I_{sh}^{w} + I_{a\,\mathrm{sgn}}^{w} + I_{add}^{w} + I_{sub}^{w}\right).$$

2.2.3.1. Змішаний зсув вліво на c біт

Розглянемо метод зсуву цілого числа вліво на c біт, в якому число, яке зсувається представлене в двійковому вигляді, а результат представлений в DCF. Відмінністю методу змішаного зсуву вліво, від звичайного методу зсуву вліво є те, що в процесі зсуву число, що зсувається, перетворюється в DCF у відповідності до Методу 2.1, і лише потім виконується сам зсув (Метод 2.11, Рис. 2.24). В процесі перетворення числа, що зсувається в DCF, отримуємо нормалізоване число в DCF.

Метод 2.11. *Метод змішаного зсуву чисел в двійковій неперервній формі вліво та результатом в DCF*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, $i = \overline{0, n-1}$, $w = v + r$, $c \le r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, c – кількість біт зсуву. *Вихід*: $d = [a << c]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.	
1. For $i \leftarrow n-1$, $i <> 0$, $i--$. 1.1. $a_i^{(r)} \leftarrow 0$	1.2. $d_i^{(w)} \leftarrow (a_i^{(w)} << c) + (a_{i-1}^{(w)} >> (w-c))$. 2. Return (d).

Рис. 2.23. Псевдокод методу 2.11

Обчислювальна складність методу змішаного зсуву вліво:

$$I_{shl}(A_{2.11}) = n\left(2I_{sh}^{w} + 2I_{a\,\mathrm{sgn}}^{w} + I_{add}^{w} + I_{sub}^{w}\right).$$

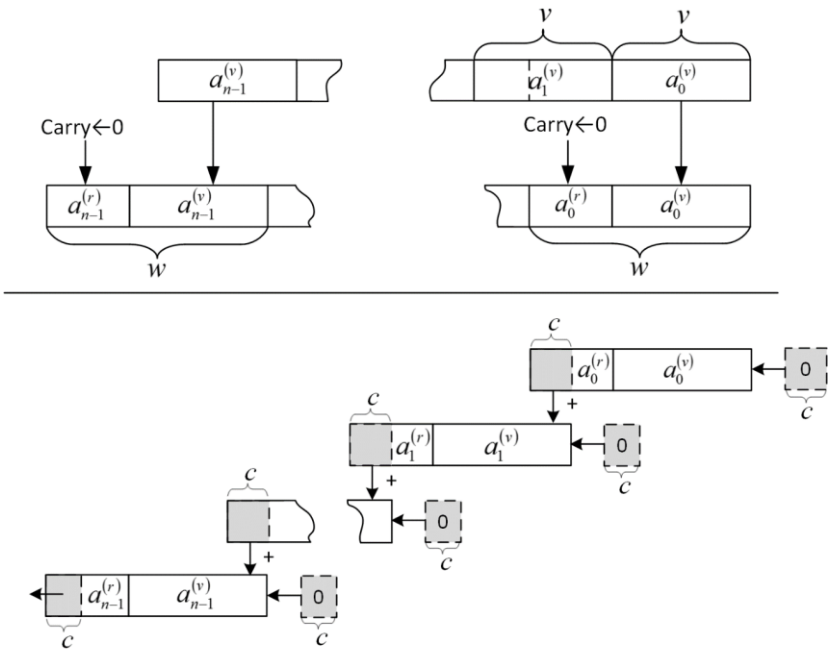


Рис 2.24. Змішаний зсув вліво числа в DCF

2.2.4. Зсув вправо на c біт

Існує два варіанти зсуву вправо [12]:

- звичайний зсув вправо (число, яке зсувається представлене в DCF);
- змішаний зсув вправо (число, яке зсувається представлене в двійковій неперервній формі, необхідне попереднє перетворення в DCF).

Розглянемо метод зсуву цілого числа вправо на c біт, в якому число, що зсувається та результат представляються в DCF. На вхід подається ціле число в DCF представленні $a \in Z_{DCF}$, яке представлене як послідовність з n машинних слів, розміром w біт кожне. Виконується операція зсуву на c біт в циклі в інтервалі $[\overline{0, n-1}]$ відповідних машинних слів. На виході отримуємо результат – ціле число в DCF представленні $d \in Z_{DCF}$.

При зсуві числа в DCF на c біт (Рис. 2.25) слід врахувати два випадки. В першому випадку, число в DCF нормалізоване, тобто області, що відведені під перенос пусті (заповнені нулями). В другому випадку, число в DCF не нормалізоване, тобто області, що відведені під перенос заповнені двійковими даними. Перед виконанням зсуву не нормалізоване число в DCF необхідно або нормалізувати (врахувати переноси та очистити області переносу від даних) або збільшити області переносів, оскільки можливі їх переповнення.

Метод 2.12. *Метод зсуву числа в DCF вправо*

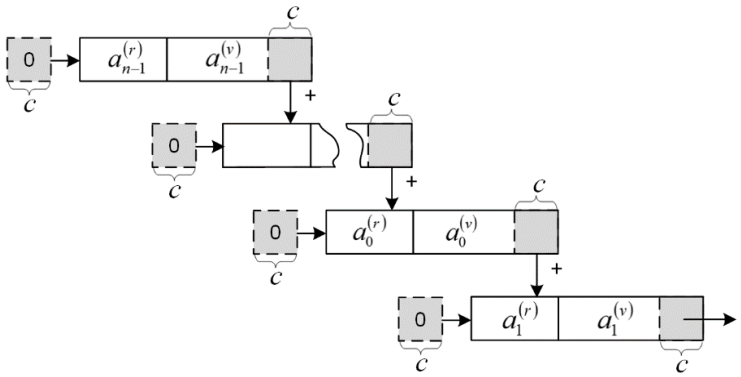
Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $\log_2 a_i = w$, $i = \overline{0, n-1}$, $w = v + r$, $c \le r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, c – кількість біт зсуву. *Вихід*: $d = [a >> c]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{n-1}, \ldots, d_1, d_0\}$, $\log_2 d_i = w$, $i = \overline{0, n-1}$.
1. For $i \leftarrow 0$, $i < n$, $i{-}{-}$. 1.1. $d_i^{(w)} \leftarrow ((a_{i+1}^{(w)} << (w-c)) >> (r-c)) + (a_i^{(w)} >> c)$. 2. Return (d).


Рис. 2.25. Метод зсуву числа в DCF вправо

Обчислювальна складність методу зсуву вправо:

$$I_{shr}\left(A_{2.12}\right)=n\left(3I_{sh}^{w}+I_{a\,\mathrm{sgn}}^{w}+2I_{sub}^{w}+I_{add}^{w}\right).$$

2.2.4.1. Змішаний зсув вправо на c біт

Розглянемо метод зсуву цілого числа вправо на c біт, в якому число, що зсувається представлене в двійковому вигляді, а результат представлений в DCF. Відмінністю методу змішаного зсуву вправо від звичайного методу зсуву вправо є те, що в процесі зсуву число, що зсувається перетворюється в DCF у відповідності до Методу 2.1, і лише потім виконується сам зсув (Метод 2.7, Рис. 2.26). В процесі перетворення числа, що зсувається в DCF, отримуємо нормалізоване число в DCF.

Метод 2.13. *Метод змішаного зсуву числа в двійковій неперервній формі вправо і результатом в DCF*

Вхід: $a\in Z_{DCF}$, $a=\{a_{n-1},\ldots,a_1,a_0\}$, $\log_2 a_i=w$, $i=\overline{0,n-1}$, $w=v+r$, $c\le r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, c – кількість біт зсуву. *Вихід*: $d=\left[a>>c\right]_{DCF}$, $d\in Z_{DCF}$, $d=\{d_{n-1},\ldots,d_1,d_0\}$, $\log_2 d_i=w$, $i=\overline{0,n-1}$.	
1. For $i\leftarrow 0$, $i<n$, $i{-}{-}$. 1.1. $a_i^{(r)}\leftarrow 0$.	1.2. $d_i^{(w)}\leftarrow((a_{i+1}^{(w)}<<(w-c))>>(r-c))+(a_i^{(w)}>>c)$. 2. Return (d).

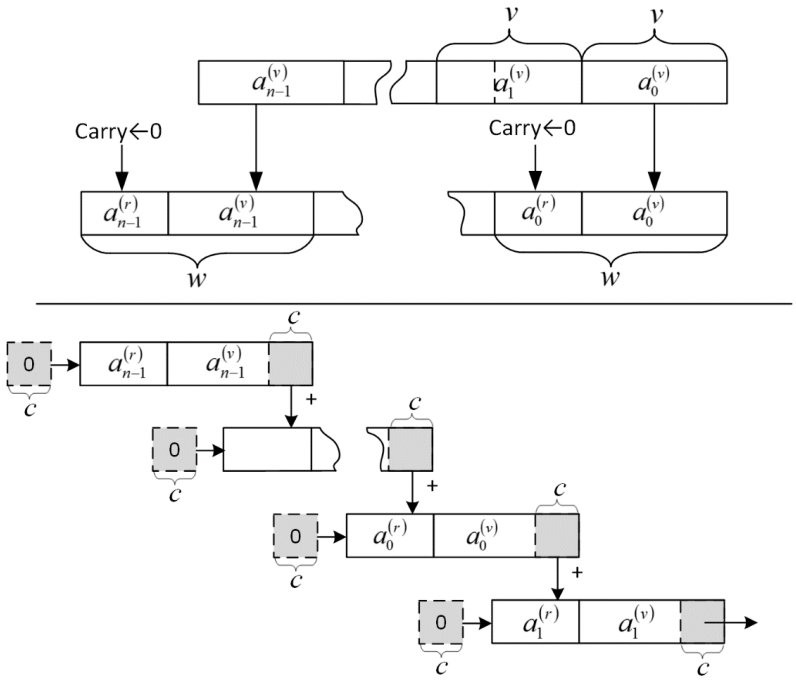


Рис. 2.26. Змішаний зсув числа в DCF вправо

Обчислювальна складність методу змішаного зсуву вправо:

$$I_{shr}\left(A_{2.13}\right) = n\left(3I_{sh}^{w} + 2I_{a\,\mathrm{sgn}}^{w} + 2I_{sub}^{w} + I_{add}^{w}\right).$$

2.2.5. Множення

Розглянуті вище арифметичні операції над числами представленими в DCF, використовуються в більш складних та часто використовуваних операціях – множенні та піднесенні до квадрату.

Множення виконується над двома множниками – цілими числами $a, b \in Z_{DCF}$, що представлені в DCF, а результатом множення є ціле число $d \in Z_{DCF}$ в DCF представленні. Вхідні значення – множники представлені у вигляді послідовності з n машинних слів, розміром w біт кожне. Це обумовлено тим, що сучасні процесори при множенні чисел, оперують машинними словами в двійковій неперервній формі розміром w біт кожне. На виході отримуємо результат – число з $2n$ машинних слів, розміром w біт кожне. Основу методу складають два цикли формування результатів множення цілих чисел в інтервалах $[\overline{0, n-1}]$ (Рис. 2.28) та $[\overline{n, 2n-1})$ (Рис. 2.29).

Метод 2.14. *Метод множення чисел в DCF*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $b \in Z_{DCF}$, $b = \{b_{n-1}, \ldots, b_1, b_0\}$, $a_i^{(w)} = a_i^{(r)} \,\|\, a_i^{(v)}$, причому $a_i^{(r)} = 0$, $b_i^{(w)} = b_i^{(r)} \,\|\, b_i^{(v)}$, причому $b_i^{(r)} = 0$, $i = \overline{0, n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, що може бути від'ємним. *Вихід*: $d = \left[a \cdot b\right]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{2n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,\|\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0, 2n-1}$.
1. $\alpha \leftarrow 2n-1$. 2. $l \leftarrow n-1$. 3. For $k \leftarrow 0$, $k < \alpha$, $k{+}{+}$ do 3.1. $d_k^{(w)} \leftarrow 0$. 4. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do 4.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$. 4.2. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do 4.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot b_j^{(v)}$. 4.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$.

4.3. $d_k^{(w)} \leftarrow d_k^{(w)} + r_0^{(w)},\ d_{k+1}^{(w)} \leftarrow d_{k+1}^{(w)} + r_1^{(w)}$.
5. For $k \leftarrow n,\ k < \alpha,\ k++$ do
5.1. $r_0^{(w)} \leftarrow 0,\ r_1^{(w)} \leftarrow 0$.
5.2. For $i \leftarrow k-l,\ j \leftarrow n-1,\ i < n,\ i++,\ j--$ do
5.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot b_j^{(v)}$.
5.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)},\ r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$.
5.3. $d_k^{(w)} \leftarrow d_k^{(w)} + r_0^{(w)},\ d_{k+1}^{(w)} \leftarrow d_{k+1}^{(w)} + r_1^{(w)}$.
6. Return (d).

Рис. 2.27. Псевдокод методу 2.14

Для наочності запропонованого методу множення з відкладеним переносом чисел в DCF та результатом в DCF представленні, результати процесів формування представлені графічно.

Як видно на Рис. 2.28, виділяються множники $a_i^{(v)}$ та $b_j^{(v)}$, які розширюються до машинного слова розміром w-біт, з якими процесор і виконує операцію множення. Але так як старші r-біт кожного множника є пустими, то і результат множення не перевищить $2v$-біт і буде розміщуватись в двох машинних словах w-біт (п. 4.2.1 та п. 5.2.1). Після чого результат $(\beta\delta)^{(2v)}$ розділяється на старшу $\beta^{(v)} \leftarrow \mathrm{Hi}(\beta\delta)^{(2v)}$ та молодшу частини $\delta^{(v)} \leftarrow \mathrm{Low}(\beta\delta)^{(2v)}$. В подальшому, старші та молодші частини, накопичуються в п. 4.2.2 та п.5.2.2, з врахуванням можливого переносу в машинних словах $r_0^{(w)}$ та $r_1^{(w)}$. Після виходу з циклу п.4.2 та п.5.2 відбувається накопичення результату в двох машинних словах $d_k^{(w)}$ та $d_{k+1}^{(w)}$, розміром w-біт кожне (п.4.3 та п.5.3). Слід звернути увагу, що п.4.3 та п.5.3 є потенційно небезпечними з точки зору неврахованого переповнення $d_k^{(w)}$ і $d_{k+1}^{(w)}$, що накладає додаткові обмеження на попередню оцінку розміру блоку накопичення переносу для $d_k^{(r)}$ и $d_{k+1}^{(r)}$.

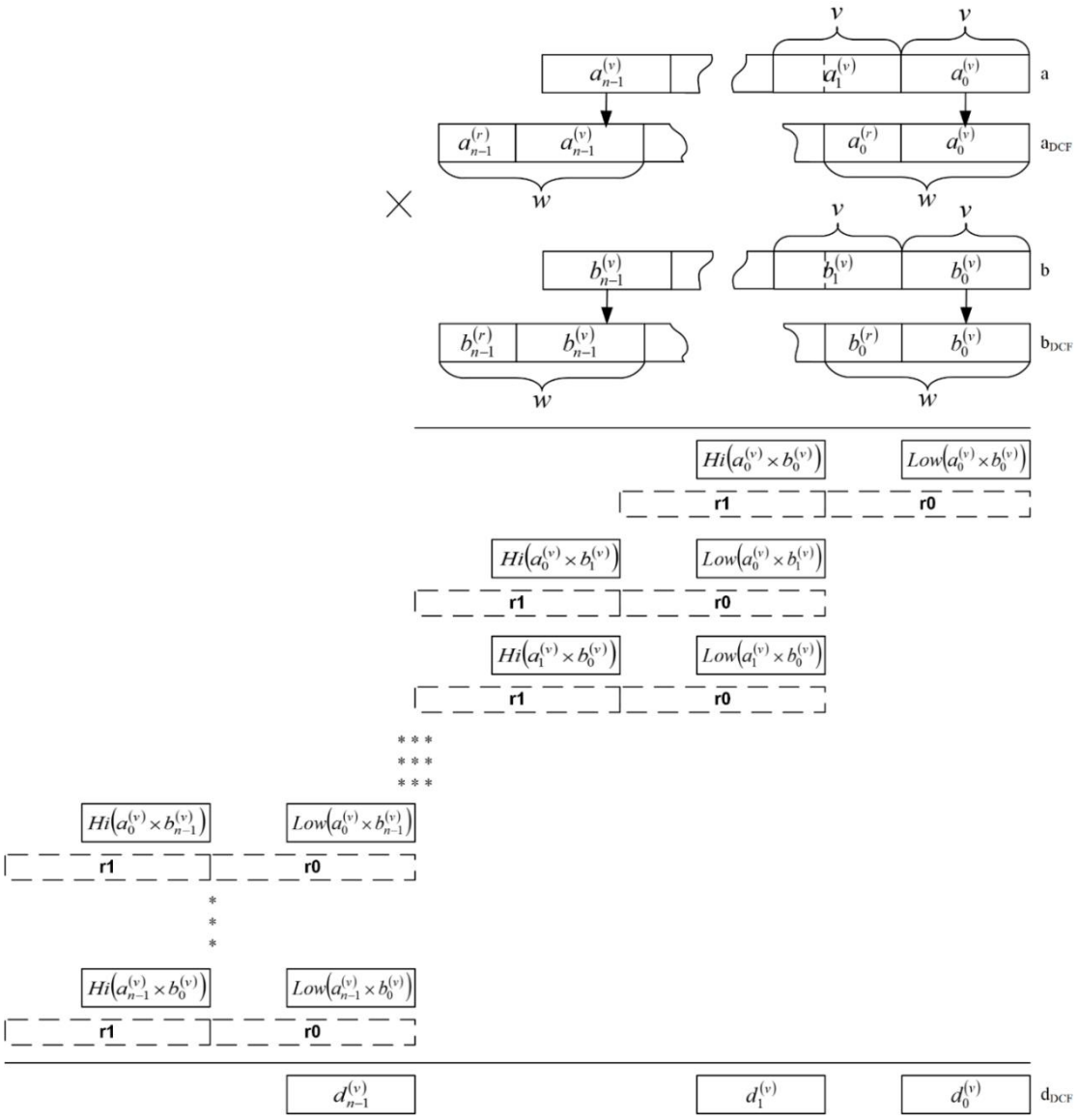


Рис. 2.28. Графічна інтерпретація 1-го циклу формування результату запропонованого методу множення

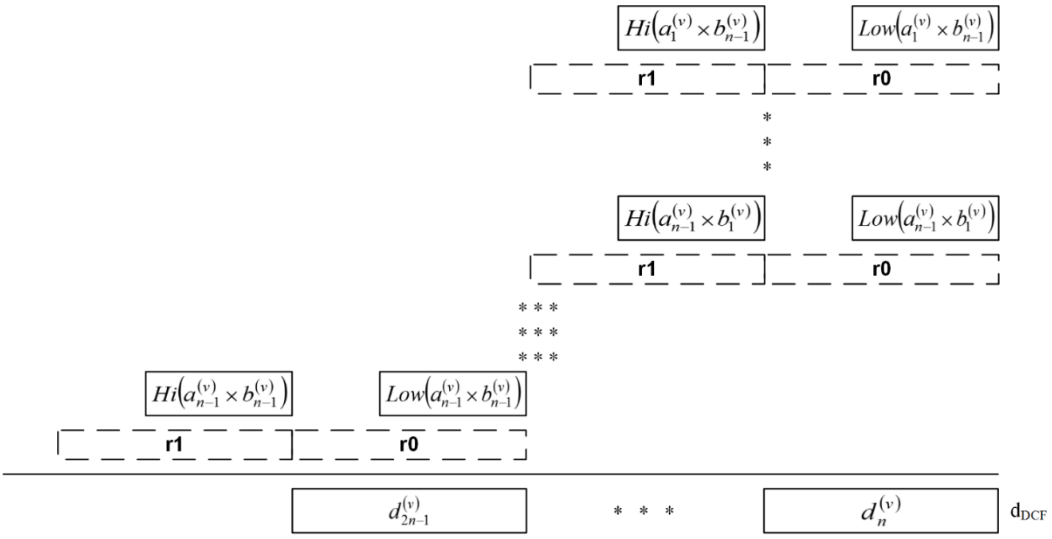


Рис 2.29. Графічна інтерпретація 2-го циклу формування результату запропонованого методу множення

Для порівняння методів використовується оцінка їх обчислювальної та просторової складності. Інтерес представляє саме обчислювальна складність. Слід зазначити, що не дивлячись на те, що в методі присутні елементи різної двійкової довжини w-біт та v-біт, процесор може оперувати лише машинними словами w-біт. Так, складність циклу п.3 складає $2n \cdot I^w_{a\,\mathrm{sgn}}$, а кроків складного циклу п.4: для п.4.1. складає $2n \cdot I^w_{a\,\mathrm{sgn}}$, для ітерацій циклу п.4.2 в п.4.2.1 та п.4.2.2 слід використовувати формулу обчислення суми арифметичної прогресії, для підрахунку числа операцій: п.4.2.1 $n\frac{n+1}{2} I^w_{mul}$ та п.4.2.2 $n \cdot (n+1) \cdot I^w_{add}$. Наступна ітерація циклу п.4 це п.4.3, число операцій якого складає $2n \cdot I^w_{add}$.

Далі, складний цикл п.5: для п.5.1. складає $2n \cdot I^w_{a\,\mathrm{sgn}}$, для ітерацій циклу п.5.2 в п.5.2.1 та п.5.2.2 слід використовувати формулу обчислення суми арифметичної прогресії, для підрахунку числа операцій: п.5.2.1 $\frac{n \cdot (n-1)}{2} \cdot I^w_{mul}$ та п.5.2.2 $n \cdot (n-1) \cdot I^w_{add}$. Наступна ітерація циклу п.5 це п.5.3, число операцій якого складає $(n-2) \cdot I^w_{add}$.

Обчислювальна складність запропонованого методу множення:

$$I_{mul}(A_{2.14}) = 6n \cdot I^w_{a\,\mathrm{sgn}} + n^2 \cdot I^w_{mul} + (2n^2 + 3 \cdot n - 2) \cdot I^w_{add}.$$

2.2.6. Піднесення до квадрату

Піднесення до квадрату є окремим випадком множення, при якому обидва множника рівні. Піднесення до квадрату виконується цілим числом $a \in Z_{DCF}$, представленим в DCF, а результатом піднесення до квадрату є ціле число $d \in Z_{DCF}$ в DCF. Вхідне значення – число представлене у вигляді послідовності з n машинних слів, розміром w-біт кожне. Це обумовлене тим, що сучасні процесори, при множенні чисел (відсутня інструкція по піднесенню до квадрату цілих чисел), оперують машинними словами в двійковій неперервній формі w-біт кожне. На виході отримуємо результат – число з $2n$ машинних слів, розміром w-біт кожне. Основу методу складають

два цикли формування результатів піднесення до квадрату цілих чисел в інтервалах $[\overline{0,n-1}]$ та $[\overline{n,2n-1})$.

Метод 2.15. *Метод піднесення до квадрату чисел в DCF та результатом в DCF, з відкладеним переносом*

Вхід: $a \in Z_{DCF}$, $a = \{a_{n-1}, \ldots, a_1, a_0\}$, $a_i^{(w)} = a_i^{(r)} \,||\, a_i^{(v)}$, причому $a_i^{(r)} = 0$, $i = \overline{0,n-1}$, $w = v + r$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, що може бути від'ємним.

Вихід: $d = [a^2]_{DCF}$, $d \in Z_{DCF}$, $d = \{d_{2n-1}, \ldots, d_1, d_0\}$, $d_i^{(w)} = d_i^{(r)} \,||\, d_i^{(v)}$, $\log_2 d_i = w$, $i = \overline{0,2n-1}$.

1. $\alpha \leftarrow 2n-1$.
2. $l \leftarrow n-1$.
3. For $k \leftarrow 0$, $k < \alpha$, $k{+}{+}$ do
3.1. $d_k^{(w)} \leftarrow 0$.
4. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do
4.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$.
4.2. For $i \leftarrow 0$, $j \leftarrow k$, $i \le j$, $i{+}{+}$, $j{-}{-}$ do
4.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot a_i^{(v)}$.
4.2.2. If $(i < j)$ then $r_0^{(w)} \leftarrow r_0^{(w)} + \left(v^{(v)} << 1\right)$,

$$r_1^{(w)} \leftarrow r_1^{(w)} + \left(u^{(v)} << 1\right) | \left(u^{(v)} >> (w-1)\right).$$

4.2.3. Else $r_0^{(w)} \leftarrow r_0^{(w)} + v^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(v)}$.
4.3. $d_k^{(w)} \leftarrow d_k^{(w)} + r_0^{(w)}$, $d_{k+1}^{(w)} \leftarrow d_{k+1}^{(w)} + r_1^{(w)}$.
5. For $k \leftarrow n$, $k < \alpha$, $k{+}{+}$ do
5.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$.
5.2. For $i \leftarrow k-l$, $j \leftarrow n-1$, $i \le j$, $i{+}{+}$, $j{-}{-}$ do
5.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot a_i^{(v)}$.
5.2.2. If $(i < j)$ then $r_0^{(w)} \leftarrow r_0^{(w)} + \left(v^{(v)} << 1\right)$,

$$r_1^{(w)} \leftarrow r_1^{(w)} + \left(u^{(v)} << 1\right) | \left(u^{(v)} >> (w-1)\right).$$

5.2.3. Else $r_0^{(w)} \leftarrow r_0^{(w)} + v^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + u^{(v)}$.
5.3. $d_k^{(w)} \leftarrow d_k^{(w)} + r_0^{(w)}$, $d_{k+1}^{(w)} \leftarrow d_{k+1}^{(w)} + r_1^{(w)}$.
6. Return (d).

Рис. 2.30. Псевдокод методу 2.15

Множники $a_i^{(v)}$ та $a_i^{(v)}$ розширюються до машинного слова w-біт, з якими процесор і виконує операцію множення. Але так як старші r-біт кожного множника є пустими, то і результат множення не перевищить $2v$-біт та буде розміщатись в 2-х машинних словах w-біт (п. 4.2.1 та п. 5.2.1). Після чого результат $(\beta\delta)^{(2v)}$ розділяється на старшу $\beta^{(v)} \leftarrow \mathrm{Hi}(\beta\delta)^{(2v)}$ та молодшу $\delta^{(v)} \leftarrow \mathrm{Low}(\beta\delta)^{(2v)}$ частини. В подальшому, старші та молодші частини, накопичуються в п. 4.2.2 та п.5.2.2, з врахуванням можливого переносу в машинних словах $r_0^{(w)}$ та $r_1^{(w)}$. Відмінною особливістю методу піднесення до квадрату від множення є подвоєння результату множення п.4.2.1 та п.5.2.1 в п.4.2.2 та п.5.2.2 відповідно. По виходу з циклу п.4.2 та п.5.2 відбувається накопичення результату в 2-х машинних словах w-біт $d_k^{(w)}$ та $d_{k+1}^{(w)}$ (п.4.3 та п.5.3). Слід звернути увагу, що п.4.3 та п.5.3 є потенційно небезпечними з точки зору неврахованого переповнення $d_k^{(w)}$ та $d_{k+1}^{(w)}$, що накладає додаткові обмеження на попередню оцінку розміру блоку накопичення переносу для $d_k^{(r)}$ та $d_{k+1}^{(r)}$.

Для порівняння методів використовується оцінка їх обчислювальної та просторової складності. Інтерес представляє саме обчислювальна складність. Слід зазначити, що не дивлячись на те, що в методі присутні елементи різної двійкової довжини w-біт та v-біт, процесор може оперувати лише машинними словами w-біт. Так, складність циклу п.3 складає $2n \cdot I_{a\,\mathrm{sgn}}^{w}$, а для кроків складного циклу п.4.: для п.4.1. складає $2n \cdot I_{a\,\mathrm{sgn}}^{w}$, для ітерацій циклу п.4.2 в п.4.2.1, п.4.2.2 та п.4.2.3 слід використовувати формулу обчислення суми арифметичної прогресії, для підрахунку числа операцій: п.4.2.1 $n\frac{n+1}{2} I_{mul}^{w}$. Слід зазначити, що імовірність виконання умови $(i < j)$ складає $\frac{1}{3}$, відповідно до формули $(a+b)^2 = a^2 + 2ab + b^2$. В зв'язку з цим, складність

п.4.2.2 $\frac{1}{3}\left(\frac{n+1}{2}n-n\right)\left(2\cdot I_{add}^{w}+3\cdot I_{sh}^{w}+I_{con}^{w}+I_{cmp}^{w}\right)$ та складність противного випадку п.4.2.3 $\frac{2}{3}(n-1)\cdot n\cdot I_{add}^{w}$. Наступною ітерацією циклу п.4 є п.4.3, число операцій якого складає $2n\cdot I_{add}^{w}$.

Далі, складний цикл п.5.: для п.5.1. складає $2n\cdot I_{a\,\mathrm{sgn}}^{w}$, для ітерацій циклу п.5.2 в п.5.2.1 та п.5.2.2 слід використовувати формулу обчислення суми арифметичної прогресії, для підрахунку числа операцій: п.5.2.1 $\frac{n\cdot(n-1)}{2}\cdot I_{mul}^{w}$ та п.5.2.2 $n\cdot(n-1)\cdot I_{add}^{w}$. Наступна ітерація циклу п.5 є п.5.3, число операцій якого складає $(n-2)\cdot I_{add}^{w}$.

Обчислювальна складність запропонованого методу піднесення до квадрату:

$$I_{sqr}\left(A_{2.15}\right)=(6n-1)\cdot I_{a\,\mathrm{sgn}}^{w}+n^{2}\cdot I_{mul}^{w}+n^{2}\cdot I_{cmp}^{w}+\frac{1}{3}\left(2n^{2}+20n-2\right)\cdot I_{add}^{w}+$$

$$+\left(n^{2}+1\right)\cdot I_{sh}^{w}+\frac{1}{3}\left(n^{2}+1\right)\cdot I_{con}^{w}.$$

2.2.7. Приведення за модулем

Для операцій в полі цілих чисел обов'язковим є приведення по простому модулю після кожної операції. Відомо декілька методів приведення за модулем, але для операцій в полях, поширення набув [2] саме метод Барретта. На основі методу Барретта для приведення цілих чисел по модулю пропонується метод з використанням DCF представлення чисел. Особливістю методу Барретта є наявність двох часткових (неповних) множень, які можна виконати за допомогою запропонованого раніше методу множення.

На вхід подається велике ціле число $a\in Z_{DCF}$ та просте число p (модуль) в двійковій неперервній формі. Для виконання часткових множень, необхідно враховувати той факт, що на вхід обох множень подаються числа в DCF, а результат множення також в DCF. Після першого часткового множення виконується лише врахування переносу, але після другого часткового

множення результат приводиться до двійкової неперервної форми. На виході результат $c = a \bmod p$ – залишок від ділення в двійковій неперервній формі.

Метод 2.16. *Приведення цілих чисел за простим модулем з частковим множенням (результати часткового множення в DCF представленні)*

Вхід: ціле $a = d \cdot b$, $d, b \in \mathrm{GF}(p)$, $2n = \log_{2^w} a$, $n = \log_{2^w} d = \log_{2^w} b$, $a = \{a_{2n-1}, \ldots, a_1, a_0\}$, $a_i^{(w)} = a_i^{(r)} \,\|\, a_i^{(v)}$, причому $a_i^{(r)} = 0$, $i = \overline{0, 2n-1}$ $k = \left\lfloor \log_{2^w} p \right\rfloor + 1$, $\mu = \left\lfloor \left(2^w\right)^{2k} \Big/ p \right\rfloor$, p – модуль, просте число. $w = v + r$, $x = v / w$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу. *Вихід*: $c = a \bmod p$, $c \in \mathrm{GF}(p)$
1. For $x \leftarrow 0$, $x < 3n$, $k{+}{+}$ do 1.1. $q_x^{(w)} \leftarrow 0$. 2. For $x \leftarrow 0$, $x < 2n$, $x{+}{+}$ do 2.1. $\alpha \leftarrow \min(k + x; 2n)$. 2.2. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$. 2.3. For $i \leftarrow k-1$, $j \leftarrow x$, $i < \alpha$, $i{+}{+}$, $j{-}{-}$ do 2.3.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot \mu_j^{(v)}$. 2.3.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$. 2.4. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)}$, $q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}$. 3. For $x \leftarrow 2n$, $e \leftarrow k$, $x < 3n$, $x{+}{+}$, $e{+}{+}$ do 3.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$. 3.2. For $i \leftarrow e$, $j \leftarrow 2n-1$, $i < 2n$, $i{+}{+}$, $j{-}{-}$ do 3.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot \mu_j^{(v)}$. 3.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$. 3.3. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)}$, $q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}$. 4. $t^{(r)} \leftarrow q_0^{(r)}$, $g_0^{(w)} \leftarrow q_0^{(v)}$ 5. For $i \leftarrow 1$, $i < 3n$, $i{+}{+}$. 5.1. $t^{(r)} \,\|\, t^{(v)} \leftarrow q_i^{(w)} + t^{(r)}$ 5.2. $g_i^{(w)} \leftarrow q_i^{(v)}$. 6. For $x \leftarrow 0$, $x < 3n$, $k{+}{+}$ do 6.1. $q_x^{(w)} \leftarrow 0$. 7. For $x \leftarrow 0$, $x < n$, $x{+}{+}$ do 7.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$. 7.2. For $i \leftarrow 0$, $j \leftarrow x$, $i \le x$, $i{+}{+}$, $j{-}{-}$ do

7.2.1. $(\beta\delta)^{(2v)} \leftarrow g_{k+1+i}^{(v)} \cdot p_j^{(v)}$.

7.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$.

7.3. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)}$, $q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}$.

8. For $x \leftarrow n$, $e \leftarrow 1, x < k+1$, $x{+}{+}$, $e{+}{+}$ do

8.1. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$.

8.2. For $i \leftarrow e$, $j \leftarrow n-1$, $i < n$, $i{+}{+}$, $j{-}{-}$ do

8.2.1. $(\beta\delta)^{(2v)} \leftarrow g_{k+1+i}^{(v)} \cdot p_j^{(v)}$.

8.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$.

8.3. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)}$, $q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}$.

9. $t^{(r)} \leftarrow q_0^{(r)}$, $g_0^{(w)} \leftarrow q_0^{(v)}$.

10. For $i \leftarrow 1$, $i < n_3$, $i{+}{+}$ do

10.1. $t^{(r)} \,||\, t^{(v)} \leftarrow q_i^{(w)} + t^{(r)}$

10.2. $j \leftarrow \lfloor i \cdot x / w \rfloor$.

10.3. $s \leftarrow (r \cdot i) \bmod w$.

10.4. $g_j^{(w)} \leftarrow g_j^{(w)} | \left(t^{(v)} << (w-s)\right)$.

10.5. if $(s > 0)$ then $g_{j+1}^{(w)} \leftarrow g_{j+1}^{(w)} | \left(t^{(v)} >> s\right)$.

11. $j \leftarrow k$.

12. For , $(j \geq 0)$ and $\left(g_j^{(w)} == a_j^{(w)}\right)$, $j{-}{-}$ do

13. If $\left(g_j^{(w)} > a_j^{(w)}\right)$ then

13. 1. $c \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$.

14. Else

14.1. $c \leftarrow a \bmod \left(2^w\right)^{k+1}$

15. $c \leftarrow c - g$

16. While $(c \geq p)$ do

16.1. $c \leftarrow c - p$

17. Return (c).

Рис. 2.31. Псевдокод методу приведення цілих чисел за простим модулем з частковим множенням

Розглянемо обчислювальну складність запропонованого методу. Слід зазначити, що не дивлячись на те, що в методі оперуються елементи різної двійкової довжини w-біт та v-біт, процесор може оперувати лише машинними словами w-біт. Обчислювальна складність основних частин методу:

- перше часткове множення (п.2 – п.3) –

$$10nI_{xor}^{w}+8nI_{add}^{w}+6nI_{a\,\mathrm{sgn}}^{w}+\frac{1}{2}\left(I_{mul}^{w}+I_{add}^{w}\right)\left(27n^{2}-11kn-17n+3k-k^{2}\right)$$

- врахування переносів (п.4 – п.5) – $2I_{a\,\mathrm{sgn}}^{w}+(3n-1)\left(I_{add}^{w}+I_{con}^{w}+I_{sh}^{w}\right)$

- друге часткове множення (п.7 – п.8) –

$$2(\mathrm{k}+1)I_{a\,\mathrm{sgn}}^{w}+I_{mul}^{w}\left(\left\lfloor\frac{1+n}{2}\right\rfloor+\frac{(n-1)}{2}\right)+I_{add}^{w}\left(2n\left\lfloor\frac{1+n}{2}\right\rfloor+(n-1)^{2}+2k+1\right)$$

- подальше приведення результату до двійкової неперервної форми (п.9 – п.10) –

$$2I_{add}^{w}+(3n-1)\left(I_{add}^{w}+I_{con}^{w}+I_{sh}^{w}+2I_{mul}^{w}+I_{\mathrm{mod}}^{w}\right)+\left\lfloor\frac{v}{w}\right\rfloor\left(I_{cmp}^{w}+I_{sh}^{w}+I_{con}^{w}\right)$$

- подальша корекція (п.11 – п.16) –

$$6I_{a\,\mathrm{sgn}}^{w}+\left(\frac{(k+1)}{2}\right)I_{\mathrm{cmp}}^{w}+\frac{1}{2}k\left(4I_{cmp}^{w}+I_{dec}^{w}+3I_{sub}^{w}+I_{mul}^{w}+2I_{add}^{w}\right).$$

Обчислювальна складність запропонованого методу приведення за модулем:

$$I_{\mathrm{mod}}\left(A_{2.16}\right)=(12n+2k+9)I_{a\,\mathrm{sgn}}^{w}+10I_{xor}^{w}+$$
$$+\left(n\left(\frac{41}{2}+2\left\lfloor\frac{1+n}{2}\right\rfloor\right)+\frac{29}{2}n^{2}+\frac{9}{2}k-\frac{1}{2}k^{2}-\frac{11}{2}kn+2\right)I_{add}^{w}+$$
$$+\left(2k+\frac{31}{2}n^{2}-\frac{11}{2}kn-\frac{1}{2}k^{2}-2+n\left(14+\left\lfloor\frac{1+n}{2}\right\rfloor\right)\right)I_{mul}^{w}+$$
$$+\left(\frac{5}{2}k+\frac{1}{2}+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{cmp}^{w}+\left(6n-2+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{sh}^{w}+\left(6n-2+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{con}^{w}+$$
$$+\left(\frac{3}{2}k\right)I_{sub}^{w}+(3n-1)I_{\mathrm{mod}}^{w}+\left(\frac{1}{2}k\right)I_{dec}^{w}.$$

2.2.8. Ділення

Для різних обчислювальних задач, в тому числі і в криптографії, виникає необхідність в цілочисельному діленні з остачею. Розглянемо метод ділення

цілих чисел з використанням DCF. За основу методу ділення пропонується взяти розглянутий вище метод приведення за модулем, що базується на основі методу Барретта з використанням DCF.

На вхід подається велике ціле число $a \in Z_{DCF}$ (ділене) та p (дільник) в двійковій неперервній формі. Для виконання часткових множень, необхідно враховувати той факт, що на вхід обох множень подаються числа в DCF, а результат множення також в DCF. Після першого часткового множення виконується лише врахування переносу, проте після другого часткового множення, результат приводиться до двійкової неперервної форми. На виході отримуємо частку q та остачу від ділення c також в двійковій неперервній формі. Відмінність полягає в тому, що після знаходження остачі від ділення, обчислюється значення частки шляхом уточнення значення оцінки частки, отриманого після виконання першого часткового множення і переводу його в двійкову неперервну форму.

Метод 2.17. *Ділення цілих чисел з частковим множенням (результати часткового множення в DCF представленні)*

Вхід: ділене $a = q \cdot p + c$, дільник p, $d, b \in \mathrm{GF}(p)$, $a, p \in Z$, $n = \log_{2^w} p$, $2n = \log_{2^w} a$, $a = \{a_{2n-1}, \ldots, a_1, a_0\}$, $a_i^{(w)} = a_i^{(r)} \,\|\, a_i^{(v)}$, причому $a_i^{(r)} = 0$, $i = \overline{0, n_2 - 1}$, $3n = n + 2n$, $k = \lfloor \log_{2^w} p \rfloor + 1$, $\mu = \left\lfloor \left(2^w\right)^{2k} / p \right\rfloor$, $w = v + r$, $x = v / w$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу. *Вихід*: частка $q = \lfloor a / p \rfloor$, остача $c = a \bmod p$, $q, c \in Z$.
1. For $x \leftarrow 0$, $x < n_3$, $k{+}{+}$ do 1.1. $q_x^{(w)} \leftarrow 0$. 2. For $x \leftarrow 0$, $x < 2n$, $x{+}{+}$ do 2.1. $\alpha \leftarrow \min(k + x; 2n)$. 2.2. $r_0^{(w)} \leftarrow 0$, $r_1^{(w)} \leftarrow 0$. 2.3. For $i \leftarrow k - 1$, $j \leftarrow x$, $i < \alpha$, $i{+}{+}$, $j{-}{-}$ do 2.3.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot \mu_j^{(v)}$. 2.3.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)}$, $r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}$. 2.4. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)}$, $q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}$.

3. For $x \leftarrow 2n,\ e \leftarrow k, x < 3n,\ x{+}{+},\ e{+}{+}$ do

3.1. $r_0^{(w)} \leftarrow 0,\ r_1^{(w)} \leftarrow 0.$

3.2. For $i \leftarrow e,\ j \leftarrow 2n-1,\ i < 2n,\ i{+}{+},\ j{-}{-}$ do

3.2.1. $(\beta\delta)^{(2v)} \leftarrow a_i^{(v)} \cdot \mu_j^{(v)}.$

3.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)},\ r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}.$

3.3. $q_x^{(w)} \leftarrow q_x^{(w)} + r_0^{(w)},\ q_{x+1}^{(w)} \leftarrow q_{x+1}^{(w)} + r_1^{(w)}.$

4. $t^{(r)} \leftarrow q_0^{(r)},\ g_0^{(w)} \leftarrow q_0^{(v)}$

5. For $i \leftarrow 1,\ i < 3n,\ i{+}{+}.$

5.1. $t^{(r)} \,||\, t^{(v)} \leftarrow q_i^{(w)} + t^{(r)}$

5.2. $g_i^{(w)} \leftarrow q_i^{(v)}.$

6. For $x \leftarrow 0,\ x < 3n,\ k{+}{+}$ do

6.1. $q_x^{(w)} \leftarrow 0.$

7. For $x \leftarrow 0,\ x < n,\ x{+}{+}$ do

7.1. $r_0^{(w)} \leftarrow 0,\ r_1^{(w)} \leftarrow 0.$

7.2. For $i \leftarrow 0,\ j \leftarrow x,\ i \le x,\ i{+}{+},\ j{-}{-}$ do

7.2.1. $(\beta\delta)^{(2v)} \leftarrow g_{k+1+i}^{(v)} \cdot p_j^{(v)}.$

7.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)},\ r_1^{(w)} \leftarrow r_1^{(w)} + \delta^{(v)}.$

7.3. $f_x^{(w)} \leftarrow f_x^{(w)} + r_0^{(w)},\ f_{x+1}^{(w)} \leftarrow f_{x+1}^{(w)} + r_1^{(w)}.$

8. For $x \leftarrow n,\ e \leftarrow 1, x < k+1,\ x{+}{+},\ e{+}{+}$ do

8.1. $r_0^{(w)} \leftarrow 0,\ r_1^{(w)} \leftarrow 0.$

8.2. For $i \leftarrow e,\ j \leftarrow n-1,\ i < n,\ i{+}{+},\ j{-}{-}$ do

8.2.1. $(\beta\delta)^{(2v)} \leftarrow g_{k+1+i}^{(v)} \cdot p_j^{(v)}.$

8.2.2. $r_0^{(w)} \leftarrow r_0^{(w)} + \delta^{(v)},\ r_1^{(w)} \leftarrow r_1^{(w)} + \beta^{(v)}.$

8.3. $f_x^{(w)} \leftarrow f_x^{(w)} + r_0^{(w)},\ f_{x+1}^{(w)} \leftarrow f_{x+1}^{(w)} + r_1^{(w)}.$

9. $t^{(r)} \leftarrow f_0^{(r)},\ g_0^{(w)} \leftarrow f_0^{(v)}.$

10. For $i \leftarrow 1,\ i < n_3,\ i{+}{+}.$

10.1. $t^{(r)} \,||\, t^{(v)} \leftarrow f_i^{(w)} + t^{(r)}.$

10.2. $j \leftarrow \lfloor i \cdot x / w \rfloor.$

10.3. $s \leftarrow (r \cdot i) \bmod w.$

10.4. $f_j^{(w)} \leftarrow f_j^{(w)} \,|\, \left(t^{(v)} << (w-s)\right).$

10.5. if $(s > 0)$ then $g_{j+1}^{(w)} \leftarrow g_{j+1}^{(w)} \,|\, \left(t^{(v)} >> s\right).$

11. $j \leftarrow k.$

12. For , $(j \ge 0)$ and $\left(g_j^{(w)} == a_j^{(w)}\right),\ j{-}{-}$ do.

13. If $\left(g_j^{(w)} > a_j^{(w)}\right)$ then
13. 1. $c \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$.
14. Else
14.1. $c \leftarrow a \bmod \left(2^w\right)^{k+1}$
15. $c \leftarrow c - g$
16. While $(c \geq p)$ do
16.1. $c \leftarrow c - p$
16.2. $q \leftarrow q + 1$.
17. Return (q, c).

Рис. 2.32. Псевдокод методу ділення цілих чисел з частковим множенням

Розглянемо обчислювальну складність запропонованого методу. Слід зазначити, що не дивлячись на те, що в методі оперуються елементи різної двійкової довжини w -біт та v -біт, процесор може оперувати лише машинними словами w -біт. Обчислювальна складність основних частин методу:

- перше часткове множення (п.2 – п.3) – $10nI_{xor}^w + 8nI_{add}^w + 6nI_{a\,\mathrm{sgn}}^w + \frac{1}{2}\left(I_{mul}^w + I_{add}^w\right)\left(27n^2 - 11kn - 17n + 3k - k^2\right)$
- врахування переносів (п.4 – п.5) – $2I_{a\,\mathrm{sgn}}^w + (3n-1)\left(I_{add}^w + I_{con}^w + I_{sh}^w\right)$
- друге часткове множення (п.7 – п.8) – $2(\mathrm{k}+1)I_{a\,\mathrm{sgn}}^w + I_{mul}^w\left(\left\lfloor \frac{1+n}{2} \right\rfloor + \frac{(n-1)}{2}\right) + I_{add}^w\left(2n\left\lfloor \frac{1+n}{2} \right\rfloor + (n-1)^2 + 2k + 1\right)$
- подальше приведення результату до двійкової неперервної форми (п.9 – п.10) – $2I_{add}^w + (3n-1)\left(I_{add}^w + I_{con}^w + I_{sh}^w + 2I_{mul}^w + I_{\mathrm{mod}}^w\right) + \left\lfloor \frac{v}{w} \right\rfloor\left(I_{cmp}^w + I_{sh}^w + I_{con}^w\right)$
- подальша корекція (п.11 – п.16) – $7I_{a\,\mathrm{sgn}}^w + \left(\frac{(k+1)}{2}\right)I_{\mathrm{cmp}}^w + \frac{1}{2}k\left(4I_{cmp}^w + I_{dec}^w + 3I_{sub}^w + I_{mul}^w + 4I_{add}^w\right)$.

Обчислювальна складність запропонованого методу ділення:

$$I_{div}\left(A_{2.17}\right) = (12n + 2k + 10)I_{a\,\mathrm{sgn}}^w + 10I_{xor}^w +$$

$$+\left(n\left(\frac{41}{2}+2\left\lfloor\frac{1+n}{2}\right\rfloor\right)+\frac{29}{2}n^2+\frac{11}{2}k-\frac{1}{2}k^2-\frac{11}{2}kn+2\right)I_{add}^{w}+$$

$$+\left(2k+\frac{31}{2}n^2-\frac{11}{2}kn-\frac{1}{2}k^2-2+n\left(14+\left\lfloor\frac{1+n}{2}\right\rfloor\right)\right)I_{mul}^{w}+$$

$$+\left(\frac{5}{2}k+\frac{1}{2}+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{cmp}^{w}+\left(6n-2+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{sh}^{w}+\left(6n-2+\left\lfloor\frac{v}{w}\right\rfloor\right)I_{con}^{w}+$$

$$+\left(\frac{3}{2}k\right)I_{sub}^{w}+(3n-1)I_{\mathrm{mod}}^{w}+\left(\frac{1}{2}k\right)I_{dec}^{w}.$$

2.2.9. Порівняння

Розглядаючи Метод 1.14 можна помітити, що порівняння двох великих цілих чисел відбувається шляхом порівняння їх відповідних машинних слів від старших до молодших. При чому в загальному випадку спочатку визначається пара перших (починаючи з найстарших) не рівних між собою машинних слів чисел a та b, а потім відбувається їх порівняння. У випадку з числами в DCF представленні, застосувати такий метод не можна оскільки машинне слово числа в DCF представленні містить перенос в старше машинне слово і порівнювати машинні слова таких чисел буде некоректно. Тому, для виконання порівняння чисел в DCF представленні, необхідно спершу виконати коригування переносів (перетворити в двійкову форму), а потім виконати порівняння відповідно до Методу 1.14.

Одним із варіантів використання елементів DCF в операції порівняння є підхід, при якому процес перетворення двох чисел, що порівнюються, в двійкове представлення суміщається з операцією порівняння машинних слів в одному циклі (Метод 2.18).

Метод 2.18. *Метод порівняння цілих чисел в DCF представленні*

Вхід: $a,b\in Z_{DCF}$, $a=\{a_{n-1},\ldots,a_1,a_0\}$, $b=\{b_{n-1},\ldots,b_1,b_0\}$, $\log_2 a_i=\log_2 b_i=w$, $i=\overline{0,n-1}$, $w=v+r$, $x=v/w$, w – довжина машинного слова, v – двійкова довжина числа, r – двійкова довжина переносу, що може бути від'ємним. *Вихід*: $res=sign(a-b)$.
1. $j\leftarrow 0$. 2. $t_1^{(r)}\leftarrow a_0^{(r)}$, $d_0^{(w)}\leftarrow a_0^{(v)}$, $t_2^{(r)}\leftarrow b_0^{(r)}$, $e_0^{(w)}\leftarrow b_0^{(v)}$. 3. For $i\leftarrow 1$, $i<n$, $i++$.

3.1. $t_1^{(r)} \| t_1^{(v)} \leftarrow a_i^{(w)} + t_1^{(r)},\ t_2^{(r)} \| t_2^{(v)} \leftarrow b_i^{(w)} + t_2^{(r)}$

3.2. $j \leftarrow \lfloor i \cdot x \rfloor$.

3.3. $s \leftarrow (r \cdot i) \bmod w$.

3.4. $d_j^{(w)} \leftarrow d_j^{(w)} \,|\, \left(t_1^{(v)} << (w-s)\right),\ e_j^{(w)} \leftarrow e_j^{(w)} \,|\, \left(t_2^{(v)} << (w-s)\right)$.

3.5. if $(s > 0)$ then $d_{j+1}^{(w)} \leftarrow d_{j+1}^{(w)} \,|\, \left(t_1^{(v)} >> s\right),\ e_{j+1}^{(w)} \leftarrow e_{j+1}^{(w)} \,|\, \left(t_2^{(v)} >> s\right)$.

3.6. $r_j \leftarrow d_j^{(w)} - e_j^{(w)}$.

4. While $r_j == 0$ do

4.1. $j \leftarrow j - 1$.

5. If $j == -1$ then $res \leftarrow 0$.

6. Else $res \leftarrow sign\,(r_j^{(w)})$.

7. Return (res).

Рис. 2.33. Псевдокод методу порівняння цілих чисел в DCF представленні

На Рис. 2.34 показана графічна інтерпретація запропонованого методу порівняння.

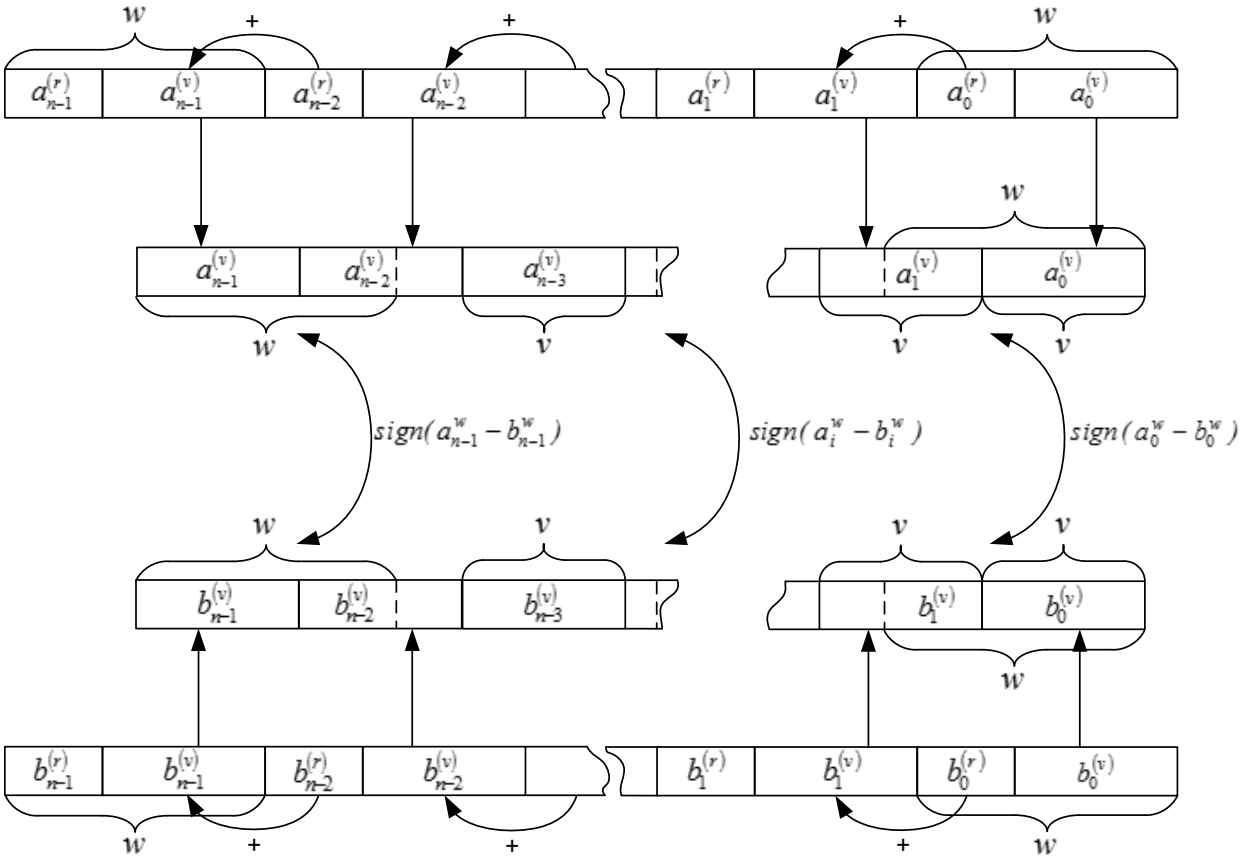


Рис. 2.34. Порівняння двох чисел в DCF представленні

Враховуючи особливості запропонованого методу, обчислювальна складність такого рішення буде рівна складності Методу 1.14 та подвійної

складності Методу 2.2. Обчислювальна складність методу порівняння цілих чисел в DCF представленні:

$$I_{cmp}\left(A_{2.18}\right)=6I^{w}_{a\,\mathrm{sgn}}+2\left(n-1\right)I^{w}_{add}+2\left(n-1\right)I^{w}_{mul}+$$
$$+\left(\mathrm{n}-1+2\left\lfloor\frac{v}{w}\right\rfloor\right)I^{w}_{con}+\left(\mathrm{n}-1+2\left\lfloor\frac{v}{w}\right\rfloor\right)I^{w}_{sh}+$$
$$+\left(\mathrm{n}+\frac{1}{2}\left\lfloor\frac{v}{w}\cdot n\right\rfloor\right)I^{w}_{cmp}+\left(\mathrm{n}-1+\frac{1}{2}\left\lfloor\frac{v}{w}\cdot n\right\rfloor\right)I^{w}_{sub}.$$

2.2.10. Інші методи

Як вже було розглянуто раніше, операції інвертування, піднесення до степені та обчислення квадратного кореню є досить трудомісткими та складаються з ряду більш простих операцій. Враховуючи той факт, що серед таких операцій можуть бути множення, піднесення до квадрату, ділення, приведення за модулем та порівняння, то при реалізації інвертування, піднесення до степені та обчислення квадратного кореню з врахуванням особливостей DCF, буде потрібне неодноразове перетворення даних в двійкове представлення і назад в DCF представлення. Це в свою чергу ускладнює методи та їх програмну реалізацію, а також негативно позначається на продуктивності.

2.3. Аналіз операцій з числами в DCF представленні

Для формалізації та спрощення аналізу методів арифметичних операцій над числами в DCF представленні, слід звернутись до табл. 2.2, в якій представлені умови застосування запропонованих методів. В табл. 2.2 показані всі запропоновані методи, їх позначення, форма представлення цілих чисел, які подаються на вхід та результату на виході. Позначення методів в більшості випадків формуються як скорочена назва методу із зазначенням форми представлення чисел на вході. Скорочення BF (Binary Form) в таблиці позначає, що використовується число в двійковій неперервній формі, а скорочення DCF (Delayed Carry Form) – що використовується число з відкладеним переносом (в DCF представленні). У випадку з операцією порівняння, позначення SIGN означає, що на виході не число – а результат порівняння.

Таблиця 2.2

Умови використання запропонованих методів

№	Назва методу	Позначення	Вхід 1	Вхід 2	Вихід
1	2	3	4	5	6
1	Перетворення цілих чисел з двійкової неперервної форми в DCF	B2D	BF	–	DCF
2	Перетворення цілих чисел з DCF в двійкову неперервну форму	D2B	DCF	–	BF
3	Додавання чисел в DCF	AddDD	DCF	DCF	DCF
4	Змішане додавання цілих чисел в двійковій неперервній формі та DCF, результат в DCF	AddBD	DCF	BF	DCF
		AddBB	BF	BF	DCF
5	Віднімання чисел в DCF	SubDD	DCF	DCF	DCF
6	Змішане віднімання чисел в DCF	SubDB	DCF	BF	DCF
		SubBD	BF	DCF	DCF
		SubBB	BF	BF	DCF
7	Зсув вліво	ShlD	DCF	–	DCF
8	Змішаний зсув вліво	ShlB	BF	–	DCF
9	Зсув вправо	ShrD	DCF	–	DCF
10	Змішаний зсув вправо	ShrB	BF	–	DCF
11	Множення чисел	MulBB	BF	BF	DCF
12	Піднесення до квадрату	SqrB	BF	–	DCF
13	Приведення за модулем	ModBB	BF	BF	BF
14	Ділення	DivBB	BF	BF	BF
15	Порівняння	CmpDD	DCF	DCF	SIGN

В табл. 2.3 приводиться порівняння обчислювальної складності існуючих методів для чисел в двійковому представленні (BF) та запропонованих методів для роботи з числами в DCF представленні. Можна помітити, що кількість операцій та складність більшості запропонованих методів (за виключенням найбільш складних, таких як порівняння, ділення та приведення за модулем) нижче аналогічних існуючих методів. Крім того, досить трудомістка операція перетворення числа з DCF в двійкову неперервну форму, може виконуватись тільки один раз, по завершенню всіх необхідних операцій над числами в DCF. Тому інтерес представляють послідовності операцій MulBB->AddDD->…->D2B, Add()->…->D2B або Sub()->…->D2B, котрі дозволяють виключити операцію коригування переносу на кожному кроці. З іншого боку, велика кількість таких операцій над числами в DCF представленні може привести до переповнення розрядів, відведених для

накопичення відкладеного переносу. В зв'язку з цим слід враховувати таку можливість і своєчасно коригувати переноси.

Таблиця 2.3

Порівняння обчислювальної складності існуючих та запропонованих методів

Додавання	BF	$2I^w_{a\text{sgn}} + n\left(2I^w_{add} + 2I^w_{cmp} + I^w_{or} + 4I^w_{a\text{sgn}}\right)$
	DCF	$n\left(I^w_{add} + I^w_{a\text{sgn}}\right)$
Віднімання	BF	$2I^w_{a\text{sgn}} + n\left(2I^w_{sub} + 2I^w_{cmp} + I^w_{or} + 4I^w_{a\text{sgn}}\right)$
	DCF	$n\left(I^w_{sub} + I^w_{a\text{sgn}}\right)$
Зсув вліво	BF	$3I^w_{a\text{sgn}} + 2I^w_{sh} + n\left(2I^w_{sh} + I^w_{or} + 2I^w_{sub} + 3I^w_{a\text{sgn}}\right)$
	DCF	$n\left(2I^w_{sh} + I^w_{a\text{sgn}} + I^w_{add} + I^w_{sub}\right)$
Зсув вправо	BF	$3I^w_{a\text{sgn}} + I^w_{sh} + n\left(2I^w_{sh} + I^w_{or} + 3I^w_{a\text{sgn}} + I^w_{sub}\right)$
	DCF	$n\left(3I^w_{sh} + I^w_{a\text{sgn}} + 2I^w_{sub} + I^w_{add}\right)$
Множення (методом Comba)	BF	$2I^{(w)}_{a\text{sgn}}\left(5n^2 + 4n + 2\right) + 2n^2 I^{2w}_{a\text{sgn}} + 2n^2 I^{2w}_{mul} + 8n^2 I^w_{add}$
	DCF	$6n \cdot I^w_{a\text{sgn}} + n^2 \cdot I^w_{mul} + \left(2n^2 + 3 \cdot n - 2\right) \cdot I^w_{add}$
Піднесення до квадрату (методом Comba)	BF	$n^2 I^w_{mul} + n^2 I^w_{add} + (n^2 + 4n) I^w_{a\text{sgn}} + \frac{n^2 + n}{2} I^{2w}_{a\text{sgn}}$
	DCF	$\left(6n-1\right) \cdot I^w_{a\text{sgn}} + n^2 \cdot I^w_{mul} + n^2 \cdot I^w_{cmp} + \frac{1}{3}\left(2n^2 + 20n - 2\right) \cdot I^w_{add} +$ $+\left(n^2 + 1\right) \cdot I^w_{sh} + \frac{1}{3}\left(n^2 + 1\right) \cdot I^w_{con}$
Приведення за модулем (метод Барретта)	BF	$5I^w_{a\text{sgn}} + 2I^w_{mul} + 2I^w_{div} + 2I^w_{\text{mod}} + I^w_{cmp} + 2I^w_{sub}$
	DCF	$\left(12n + 2k + 9\right) I^w_{a\text{sgn}} + 10 I^w_{xor} + \left(\frac{3}{2}k\right) I^w_{sub} + \left(3n-1\right) I^w_{\text{mod}} + \left(\frac{1}{2}k\right) I^w_{dec} +$ $+\left(n\left(\frac{41}{2} + 2\left\lfloor\frac{1+n}{2}\right\rfloor\right) + \frac{29}{2}n^2 + \frac{9}{2}k - \frac{1}{2}k^2 - \frac{11}{2}kn + 2\right) I^w_{add} +$ $+\left(2k + \frac{31}{2}n^2 - \frac{11}{2}kn - \frac{1}{2}k^2 - 2 + n\left(14 + \left\lfloor\frac{1+n}{2}\right\rfloor\right)\right) I^w_{mul} +$ $+\left(\frac{5}{2}k + \frac{1}{2} + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{cmp} + \left(6n - 2 + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{sh} + \left(6n - 2 + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{con}$
Ділення	BF	$4I^w_{a\text{sgn}} + 2I^w_{mul} + 2I^w_{div} + 2I^w_{\text{mod}} + I^w_{add} + 2I^w_{sub} + 2I^w_{cmp}$
	DCF	$\left(12n + 2k + 10\right) I^w_{a\text{sgn}} + 10 I^w_{xor} + \left(\frac{3}{2}k\right) I^w_{sub} + \left(3n-1\right) I^w_{\text{mod}} + \left(\frac{1}{2}k\right) I^w_{dec} +$ $+\left(n\left(\frac{41}{2} + 2\left\lfloor\frac{1+n}{2}\right\rfloor\right) + \frac{29}{2}n^2 + \frac{11}{2}k - \frac{1}{2}k^2 - \frac{11}{2}kn + 2\right) I^w_{add} +$ $+\left(2k + \frac{31}{2}n^2 - \frac{11}{2}kn - \frac{1}{2}k^2 - 2 + n\left(14 + \left\lfloor\frac{1+n}{2}\right\rfloor\right)\right) I^w_{mul} +$ $+\left(\frac{5}{2}k + \frac{1}{2} + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{cmp} + \left(6n - 2 + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{sh} + \left(6n - 2 + \left\lfloor\frac{v}{w}\right\rfloor\right) I^w_{con}$

Продовження Табл. 2.3

Порівняння	BF	$4I^w_{a\text{sgn}} + I_{sub} + 2I_{cmp} + m\left(I_{sub} + I_{a\text{sgn}}\right) + I^w_{sub} + I^w_{cmp}$
	DCF	$6I^w_{a\text{sgn}} + 2(n-1)I^w_{add} + 2(n-1)I^w_{mul} +$ $+\left(\text{n}-1+2\left\lfloor\frac{v}{w}\right\rfloor\right)I^w_{con} + \left(\text{n}-1+2\left\lfloor\frac{v}{w}\right\rfloor\right)I^w_{sh} +$ $+\left(\text{n}+\frac{1}{2}\left\lfloor\frac{v}{w}\cdot n\right\rfloor\right)I^w_{cmp} + \left(\text{n}-1+\frac{1}{2}\left\lfloor\frac{v}{w}\cdot n\right\rfloor\right)I^w_{sub}$

2.4. Висновки до другого розділу

Таким чином, у другому розділі в результаті проведених досліджень отримані такі наукові та практичні результати:

1. Для підвищення швидкодії криптографічних перетворень з відкритим ключем запропоноване представлення цілих чисел з відкладеним переносом (DCF). Використання такого представлення дозволяє досягти підвищення швидкодії в операціях з цілими числами, за рахунок відкладення виконання операції переносу з старших розрядів в молодші, а для займу з молодших розрядів в старші.
2. Проведена аналітична оцінка надлишковості при використанні цілих чисел в DCF представленні, а також розрахунок очікуваної довжини числа в DCF представленні, в залежності від розміру області для зберігання відкладених переносів (займів).
3. Розроблені методи арифметичних операцій над цілими числами в DCF представленні – додавання, віднімання, множення, піднесення до квадрату, приведення за модулем, ділення, порівняння та зсуви. Кількість операцій та складність більшості запропонованих методів (за виключенням найбільш складних, таких як порівняння, ділення та приведення за модулем) нижче аналогічних існуючих методів. Крім того, досить трудомістка операція перетворення числа з DCF в двійкову неперервну форму, може виконуватись тільки один раз, по завершенню всіх необхідних операцій над числами в DCF.

4. Враховуючи специфіку DCF представлення, коли порядок виконання операцій над машинними словами не є суттєвим, стає очевидною можливість паралельного виконання операцій над відповідними словами.

Список використаних джерел у другому розділі

1. Hankerson, D.; Vanstone, S.; Menezes, A. Guide to Elliptic Curve Cryptography. Springer Professional Computing. New York: Springer. - 2004. – 311 p. doi:10.1007/b97644
2. Bhattacharyya SS, Deprettere EF, Leupers R, Takala J, editors. Handbook of Signal Processing Systems. 2nd ed. Springer, 2013. 1399 pp.
3. Brent R.P., Zimmermann P. Modern Computer Arithmetic. New York: Cambridge University Press, 2011. 236 pp.
4. Menezes A.J., Vanstone S.A., and Oorschot P.C.V. Handbook of Applied Cryptography. 1st ed. Boca Raton: CRC Press, 1996. 780 pp.
5. А. Охрименко, В. Ковтун, «Умножения целых чисел с использованием отложенного переноса для криптосистем с открытым ключом», Информационные технологи и системы в управлении, образовании, науке: Монография, под ред. проф. В.С. Пономаренко, Харьков: Цифрова друкарня №1, 2013, С. 69-82.
6. R. Brumnik, V. Kovtun, A. Okhrimenko, S. Kavun, «Techniques for Performance Improvement of Integer Multiplication in Cryptographic Applications», Mathematical Problems in Engineering, 2014, P. 1-7.
7. А.А. Охрименко, «Арифметика с отложенным переносом», Захист інформації, Т. 16, № 2, С. 130-138, 2014.
8. A. Okhrimenko, V.Yu. Kovtun, O.L. Stokipniy, «Integer representation with delayed carry», AVIATION IN THE XXI-st CENTURY – Safety in Aviation and Space Technologies: VI World Congress, September 23-25, 2014., К., 2014, P. 1.11.10-1.11.14.
9. A. Okhrimenko, «Arithmetic operations with delayed carry for public key transformations», Актуальні питання забезпечення кібернетичної безпеки та захисту інформації Зб. наук. праць наук.-практ. конф., 25-28 лютого 2015 р., К., 2015, С. 83-84.
10. А.А. Охрименко, В.Ю. Ковтун, А.Л. Стокипный «Использование представления целых чисел с отложенным переносом в криптографических преобразованиях», Безпека інформації в

інформаційно-телекомунікаційних системах: матеріали XVI міжнар. наук.-практ. конф. 26-28 травня 2015 р.: К., 2015, С. 14-15.

11. А. Охрименко, В. Ковтун, «Арифметические операции с отложенным переносом над целыми числами», Информационные технологии и защита информации в информационно-коммуникационных системах: Монография, под ред. В.С. Пономаренко, Харьков: Вид-во ТОВ «Щедра садиба плюс», 2015, С. 193-207.
12. А.А. Охрименко, В.Ю. Ковтун, «Операции арифметического сдвига над целыми числами с отложенным переносом», Проблеми та перспективи розвитку ІТ-індустрії: VII міжнар. наук.-практ. конф., 17-18 квітня 2015 р., Харків, 2015, С. 30.

РОЗДІЛ 3

РОЗРОБКА МЕТОДІВ АРИФМЕТИЧНИХ ПЕРЕТВОРЕНЬ З ВИКОРИСТАННЯМ ВІДКЛАДЕНОГО ПЕРЕНОСУ ТА РОЗПАРАЛЕЛЮВАННЯ

3.1. Підходи до розпаралелювання деяких методів арифметичних операцій з відкладеним переносом

Використання відкладеного переносу дозволяє застосовувати деякі підходи до розпаралелювання методів арифметичних операцій. Пропонується розглянути підходи до розпаралелювання деяких методів операцій, зокрема множення, піднесення до квадрату та приведення за модулем. Для зручності, розглядається окремий випадок, де довжина блоку r (для зберігання переносу), дорівнює довжині блоку v (який заповнюється бітами з числа в двійковому представленні) і відповідає довжині машинного слова w (тобто, $r = v = w$). Слід зазначити, що у вказаних методах присутня операція множення. В якості прототипу вибрано множення методом Comba [1], в якому присутні два цикли множення, які можна розпаралелювати:

- паралельне виконання першого та другого циклу множення, з подальшим коригуванням результатів, використовуючи два паралельні потоки.
- паралельне виконання ітерацій першого та другого циклу множення, з подальшим злиттям проміжних результатів, з використанням множини паралельних потоків.

В якості технології розпаралелювання обрана технологія OpenMP.

3.1.1. Методи множення

Нижче представлені метод множення, в якому для накопичення переносу відведено блок розміром як розмір машинного слова w ($r = v = w$) і два методи, в яких застосовуються два різних підходи до розпаралелювання [2-12].

Метод 3.1. *Метод множення Modified Comba*

На вхід подається два великих числа a і b, які представляються у вигляді машинних слів $a=(a_n,...,a_i,...,a_1,a_0)$ и $b=(b_n,...,b_j,...,b_1,b_0)$ розміром w біт, n - кількість машинних слів необхідних для представлення чисел a і b. На виході отримуємо результат $c=a\cdot b$, розміром $2n$ машинних слів.

Основу вказаного методу множення складають два цикли формування результату множення [8]. Перший цикл формує результат в інтервалі $k=[\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i=[\overline{0,k}]$ та $j=[\overline{k,0}]$. Другий цикл формує результат в інтервалі $k=[\overline{n,2n-1})$ з використанням допоміжного інтервалу $l=[\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах $i=[\overline{l,n})$ та $j=[\overline{k-l,n-l})$. У вкладених циклах виконується множення $(uv)^{(2w)}\leftarrow a_i^{(w)}\cdot b_j^{(w)}$, результатом якого є ціле число розміром $2w$, яке потім розділяється на два w - розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$ розрядних тимчасових змінних r_0 та r_1 (які заздалегідь проініціалізовані) на кожній ітерації:

$$r_0^{(2w)}\leftarrow r_0^{(2w)}+v^{(w)},\ r_1^{(2w)}\leftarrow r_1^{(2w)}+u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу) з використанням $2w$ -розрядних тимчасових змінних r_1 та r_2 (r_2 заздалегідь проініціалізована):

$$r_1^{(2w)}\leftarrow r_1^{(2w)}+\mathrm{hi}_{(w)}\left(r_0^{(2w)}\right),\ r_2^{(2w)}\leftarrow r_2^{(2w)}+\mathrm{hi}_{(w)}\left(r_1^{(2w)}\right).$$

та відбувається присвоєння кінцевого результату і зміна тимчасових змінних r_0, r_1 та r_2

$$c_k^{(w)}\leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_0^{(2w)}\leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right),\ r_1^{(2w)}\leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right),\ r_2^{(2w)}\leftarrow 0.$$

Наприкінці відбувається формування результату $c_{2n-1}^{(w)}\leftarrow low_{(w)}(r_0^{(2w)})$.

Результатом множення є ціле число $c=(c_{2n-1},...,c_k,...,c_1,c_0)$.

Вхід: $a, b \in \mathrm{GF}(p)$, $n = \log_{2^w} a$, $nk = 2n-1$, w –довжина машинного слова.

Вихід: $c = a \cdot b$.

1. $r_0^{(2w)} \leftarrow 0$, $r_1^{(2w)} \leftarrow 0$, $r_2^{(2w)} \leftarrow 0$.
2. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do
2.1. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do
2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$.
2.1.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$
2.2. $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$, $r_2^{(2w)} \leftarrow r_2^{(2w)} + \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right)$
2.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right)$, $r_2^{(2w)} \leftarrow 0$.
3. For $k \leftarrow n$, $l \leftarrow 1, k < nk$, $k{+}{+}$, $l{+}{+}$ do
3.1. For $i \leftarrow l$, $j \leftarrow k-l$, $i < n$, $i{+}{+}$, $j{-}{-}$ do
3.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$.
3.1.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$
3.2. $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$, $r_2^{(2w)} \leftarrow r_2^{(2w)} + \mathrm{hi}_{(2)}\left(r_1^{(2w)}\right)$
3.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right)$, $r_2^{(2w)} \leftarrow 0$.
4. $c_{nk}^{(w)} \leftarrow \mathrm{low}_{(2)}\left(r_0^{(2w)}\right)$.
5. Return (c).

Рис. 3.1. Псевдокод методу множення Modified Comba

На Рис. 3.2 графічна інтерпретація циклу формування результату в інтервалі $k = [\overline{0,n})$, а на Рис. 3.3 показана графічна інтерпретація циклу формування результату в інтервалі $k = [\overline{n,2n-1})$ для запропонованого методу множення при $n = 3$, де чітко простежуються складання результатів відповідних добутків за стовпчиками.

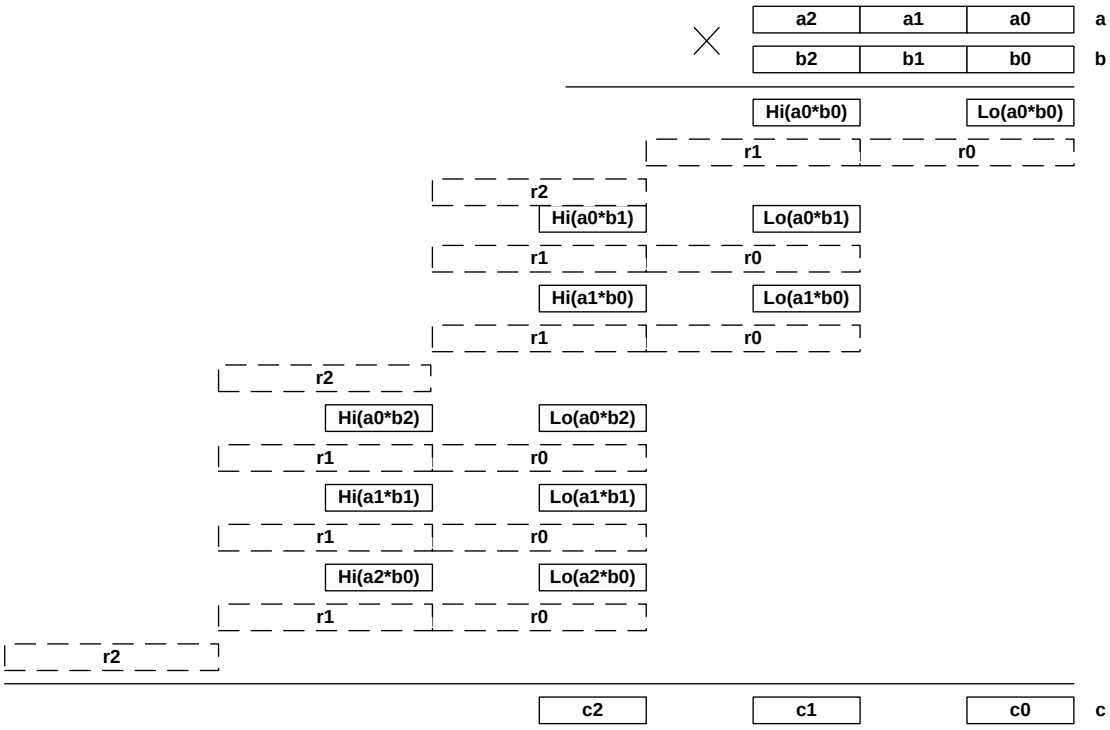


Рис. 3.2. Графічна інтерпретація циклу 2 запропонованого методу

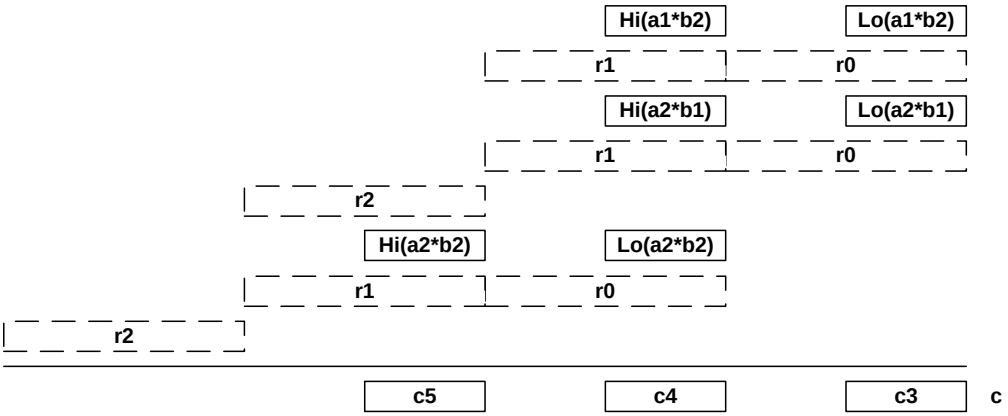


Рис. 3.3. Графічна інтерпретація циклу 3 запропонованого методу

Обчислювальна складність запропонованого методу:

$$I_{mul}(A_{3.1}) = 3I_{assign}^{2w} + n\left(\left\lceil \frac{n}{2} \right\rceil \left(I_{mul}^{w} + 2I_{add}^{2w} + I_{assign}^{2w}\right) + 5I_{assign}^{2w} + 2I_{add}^{2w} + I_{assign}^{w}\right) +$$

$$+n \cdot \left(\left\lfloor \frac{n}{2} \right\rfloor \left(I_{mul}^{w} + 2I_{add}^{2w} + I_{assign}^{2w}\right) + 5I_{assign}^{2w} + 2I_{add}^{2w} + I_{assign}^{w}\right) + I_{assign}^{w}$$

Метод 3.2. *Метод множення Modified Comba з розпаралелюванням в два потоки*

На вхід подається два великих числа a та b, які представляються у вигляді машинних слів $a = (a_n, ..., a_i, ..., a_1, a_0)$ та $b = (b_n, ..., b_j, ..., b_1, b_0)$ розміром w біт, де n – кількість машинних слів необхідних для

представлення чисел a та b. На виході отримуємо результат $c = a \cdot b$, розміром $2n$ машинних слів.

Основу вказаного методу множення складають два цикли формування результату множення, які виконуються паралельно в два потоки. Перший цикл формує результат в інтервалі $k = [\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i = [\overline{0,k}]$ та $j = [\overline{k,0}]$. Другий цикл формує результат в інтервалі $k = [\overline{n,2n-1})$ з використанням допоміжного інтервалу $l = [\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах $i = [\overline{l,n})$ та $j = [\overline{k-l,n-l})$. Кожний потік використовує власні тимчасові змінні rl_0, rl_1 та rl_2, які заздалегідь проініціалізовані. У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних тимчасових змінних rl_0 та rl_1 на кожній ітерації.

$$rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)},\ rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу) з використанням $2w$ - розрядних тимчасових змінних rl_1 та rl_2:

$$rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \mathrm{hi}_{(w)}\left(rl_0^{(2w)}\right),\ rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \mathrm{hi}_{(w)}\left(rl_1^{(2w)}\right).$$

і відбувається присвоєння кінцевого результату і зміна тимчасових змінних rl_0, rl_1 та rl_2

$$c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(rl_0^{(2w)}\right),\ rl_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_1^{(2w)}\right),\ rl_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_2^{(2w)}\right),\ rl_2^{(2w)} \leftarrow 0.$$

Після завершення першого циклу формування результату множення для збереження значення можливого переносу використовується глобальна змінна r_0 : $r_0^{(2w)} \leftarrow rl_0^{(2w)}$.

Після завершення роботи двох паралельних потоків, необхідно виконати коригування результатів роботи другого циклу формування результатів

множення за рахунок переносу, отриманого в результаті роботи першого циклу. Для цього використовується цикл коригування результатів в інтервалі $k = [\overline{n, 2n-1})$:

$r_0^{(2w)} \leftarrow r_0^{(2w)} + c_k^{(w)},\ c_k^{(w)} \leftarrow \text{low}_{(w)}\left(r_0^{(2w)}\right),\ \text{low}_{(w)}\left(r_0^{(2w)}\right) \leftarrow hi_{(w)}\left(r_0^{(2w)}\right),\ hi_{(w)}\left(r_0^{(2w)}\right) \leftarrow 0.$

Після завершення циклу коригування відбувається формування результату $c_{2n-1}^{(w)}$: $c_{2n-1}^{(w)} \leftarrow c_{2n-1}^{(w)} + low_{(w)}(r_0^{(2w)})$.

Результатом множення є ціле число $c = (c_{2n-1}, ..., c_k, ..., c_1, c_0)$.

Вхід: $a, b \in \text{GF}(p),\ n = \log_{2^w} a,\ nk = 2n-1,\ w$ –довжина машинного слова. *Вихід*: $c = a \cdot b$.
1. #pragma omp parallel sections private ($r_0^{(2w)}, r_1^{(2w)}$) begin 1.1. #pragma omp section begin 1.1.1. $rl_0^{(2w)} \leftarrow 0,\ rl_1^{(2w)} \leftarrow 0,\ rl_2^{(2w)} \leftarrow 0$. 1.1.2. For $k \leftarrow 0,\ k < n,\ k{+}{+}$ do 1.1.2.1. For $i \leftarrow 0,\ j \leftarrow k,\ i \le k,\ i{+}{+},\ j{-}{-}$ do 1.1.2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$. 1.1.2.1.2. $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)},\ rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$ 1.1.2.2. $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \text{hi}_{(w)}\left(rl_0^{(2w)}\right),\ rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \text{hi}_{(w)}\left(rl_1^{(2w)}\right)$ 1.1.2.3. $c_k^{(w)} \leftarrow \text{low}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_0^{(2w)} \leftarrow \text{low}_{(w)}\left(rl_1^{(2w)}\right)$, $rl_1^{(2w)} \leftarrow \text{low}_{(w)}\left(rl_2^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow 0$. 1.1.3. $r_0^{(2w)} \leftarrow rl_0^{(2w)}$. #pragma omp section end 1.2. #pragma omp section begin 1.2.1. $rl_0^{(2w)} \leftarrow 0,\ rl_1^{(2w)} \leftarrow 0,\ rl_2^{(2w)} \leftarrow 0$. 1.2.2. For $k \leftarrow n,\ l \leftarrow 1, k < nk,\ k{+}{+},\ l{+}{+}$ do 1.2.2.1. For $i \leftarrow l,\ j \leftarrow n-1,\ i < n,\ i{+}{+},\ j{-}{-}$ do 1.2.2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$. 1.2.2.1.2. $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)},\ rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$ 1.2.2.2. $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \text{hi}_{(w)}\left(rl_0^{(2w)}\right),\ rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \text{hi}_{(w)}\left(rl_1^{(2w)}\right)$ 1.2.2.3. $c_k^{(w)} \leftarrow \text{low}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_0^{(2w)} \leftarrow \text{low}_{(w)}\left(rl_1^{(2w)}\right)$, $rl_1^{(2w)} \leftarrow \text{low}_{(w)}\left(rl_2^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow 0$. #pragma omp section end #pragma omp parallel sections end

2. For $k \leftarrow n,\ k < nk,\ k{+}{+}$ do

2.1. $r_0^{(2w)} \leftarrow r_0^{(2w)} + c_k^{(w)}$.

2.2. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$.

2.3. $\mathrm{low}_{(w)}\left(r_0^{(2w)}\right) \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.

2.4. $\mathrm{hi}_{(w)}\left(r_0^{(2w)}\right) \leftarrow 0$.

3. $c_{nk}^{(w)} \leftarrow c_{nk}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$.

4. Return (c).

Рис. 3.4. Псевдокод методу множення Modified Comba з розпаралелюванням в 2 потоки

Обчислювальна складність запропонованого методу:

$$I_{mul}\left(A_{3.2}\right) = 4I_{assign}^{2w} + \max\Big[n\Big(\lceil \tfrac{n}{2} \rceil\Big(I_{mul}^{w} + 2I_{add}^{2w+w}\Big) + 2I_{add}^{2w+w} + 4I_{assign}^{2w}\Big),$$

$$n\Big(\lfloor \tfrac{n}{2} \rfloor\Big(I_{mul}^{w} + 2I_{add}^{2w+w}\Big) + 2I_{add}^{2w+w} + 4I_{assign}^{2w}\Big)\Big] + \tfrac{n}{2}\Big[I_{add}^{2w+w} + 3I_{assign}^{w}\Big] + I_{add}^{w}.$$

На Рис. 3.5 показана графічна інтерпретація роботи запропонованого методу множення з двома потоками при $n = 3$.

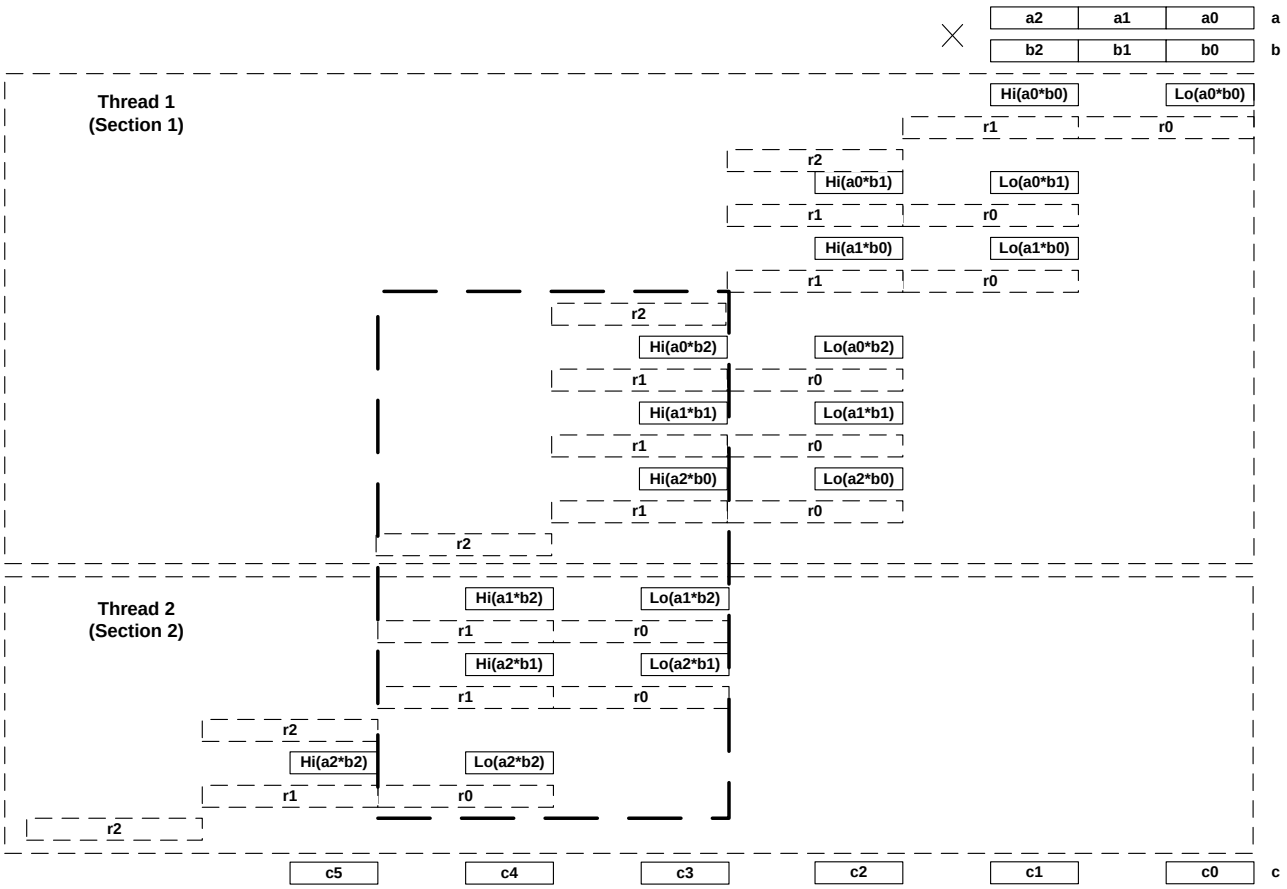


Рис. 3.5. Графічна інтерпретація циклу 2 запропонованого методу з розпаралелюванням в два потоки

Метод 3.3. *Метод множення Modified Comba з розпаралелюванням в декілька потоків*

На вхід подається два великих числа a і b, які представляються у вигляді машинних слів $a=(a_n,...,a_i,...,a_1,a_0)$ та $b=(b_n,...,b_j,...,b_1,b_0)$ розміром w біт, де n – кількість машинних слів необхідних для представлення чисел a і b. На виході отримуємо результат $c=a\cdot b$, розміром $2n$ машинних слів.

Основу вказаного методу множення складають два цикли формування результату множення, ітерації яких не залежать одна від одної і можуть виконуватись в окремих потоках. Перший цикл формує результат в інтервалі $k=[\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i=[\overline{0,k}]$ та $j=[\overline{k,0}]$. Другий цикл формує результат в інтервалі $k=[\overline{n,2n-1})$ з використанням допоміжного інтервалу $l=[\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах $i=[\overline{l,n})$ и $j=[\overline{k-l,n-l})$. Кожний потік в рамках ітерації використовує заздалегідь проініціалізовані масиви $r0_i^{(2w)}$ та $r1_i^{(2w)}$), де $i=[\overline{0,2n-1})$. У вкладених циклах використовуються власні тимчасові змінні rl_0 и rl_1, які ініціалізуються на кожній ітерації циклів формування результату. У вкладених циклах виконується множення $(uv)^{(2w)}\leftarrow a_i^{(w)}\cdot b_j^{(w)}$, результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ і $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$- розрядних змінних rl_0 та rl_1 на кожній ітерації:

$$rl_0^{(2w)}\leftarrow rl_0^{(2w)}+v^{(w)},\ rl_1^{(2w)}\leftarrow rl_1^{(2w)}+u^{(w)}.$$

На кожній ітерації циклу формування результату виконується збереження переносів:

$$r0_k^{(2w)}\leftarrow rl_0^{(2w)},\ r1_k^{(2w)}\leftarrow rl_1^{(2w)}.$$

Після завершення циклів формування результатів, необхідно виконати цикл врахування збережених переносів в інтервалі $k=[\overline{0,2n-1})$, з

використанням заздалегідь проініціалізованих тимчасових змінних $r^{(2w)}$, $rll_0^{(2w)}, rll_1^{(2w)}$:

$$rll_0^{(2w)} \leftarrow r0_k^{(2w)},\ rll_1^{(2w)} \leftarrow r1_k^{(2w)}\ \ rll_0^{(2w)} \leftarrow rll_0^{(2w)} + low_{(w)}(r^{(2w)}),$$

$$c_k^{(w)} \leftarrow low_{(w)}(rll_0^{(2w)}),\ rll_1^{(2w)} \leftarrow rll_1^{(2w)} + low_{(w)}(rll_0^{(2w)}),\ r^{(2w)} \leftarrow rll_1^{(2w)}.$$

Після завершення циклу врахування збережених переносів відбувається формування результату $c_{2n-1}^{(w)}$:

$$c_{2n-1}^{(w)} \leftarrow low_{(w)}(r^{(2w)}).$$

Результатом множення є ціле число $c = (c_{2n-1},...,c_k,...,c_1,c_0)$.

Вхід: $a, b \in \mathrm{GF}(p)$, $n = \log_{2^w} a$, $nk = 2n-1$, w – довжина машинного слова. *Вихід*: $c = a \cdot b$.
1. $l \leftarrow 1$. 2. Arrays $r0_i^{(w)}$ and $r1_i^{(w)}$, $i = \overline{0, nk}$. 3. #pragma omp parallel private ($r0_i^{(w)}, r1_i^{(w)}$) reduction (+ : l) begin 4. #pragma omp for nowait begin 4.1. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do 4.1.1. $rl_0^{(w)} \leftarrow 0$, $rl_1^{(w)} \leftarrow 0$. 4.1.2. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do 4.1.2.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$. 4.1.2.2. $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$ 4.1.3. $r0_k^{(2w)} \leftarrow rl_0^{(2w)}$, $r1_k^{(2w)} \leftarrow rl_1^{(2w)}$ #pragma omp for end 5. #pragma omp for nowait begin 5.1. For $k \leftarrow n$, $k < nk$, $k{+}{+}$ do 5.1.1. $rl_0^{(2w)} \leftarrow 0$, $rl_1^{(2w)} \leftarrow 0$. 5.1.2. For $i \leftarrow l$, $j \leftarrow n-1$, $i < n$, $i{+}{+}$, $j{-}{-}$ do 5.1.2.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$. 5.1.2.2. $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$ 5.1.3. $r0_k^{(2w)} \leftarrow rl_0^{(2w)}$, $r1_k^{(2w)} \leftarrow rl_1^{(2w)}$ 5.1.4. $l{+}{+}$. #pragma omp for end #pragma omp parallel end 6. $r_0^{(2w)} \leftarrow 0$.

7. For $k \leftarrow 0,\ k < nk,\ k++$ do

7.1. $rll_0^{(2w)} \leftarrow r0_k^{(2w)},\ rll_1^{(2w)} \leftarrow r1_k^{(2w)}$.

7.2. $rll_0^{(2w)} \leftarrow rll_0^{(2w)} + \mathrm{low}_{(w)}\left(r^{(2w)}\right)$.

7.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(rll_0^{(2w)}\right)$.

7.4. $rll_1^{(2w)} \leftarrow rll_1^{(2w)} + \mathrm{low}_{(w)}\left(rll_0^{(2w)}\right)$.

7.5. $r^{(2w)} \leftarrow rll_1^{(2w)}$.

8. $c_{nk}^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r^{(2w)}\right)$.

9. Return (c).

Рис. 3.6. Псевдокод методу 3.3

Обчислювальна складність запропонованого методу:

$$I_{mul}\left(A_{3.3}\right) = \tfrac{n}{Z}\left[4I_{assign}^{w} + \left\lceil \tfrac{n}{2} \right\rceil\left(I_{mul}^{w} + 2I_{add}^{2w+w}\right) + \tfrac{n}{Z}\left[4I_{assign}^{w} + \left\lfloor \tfrac{n}{2} \right\rfloor\left(I_{mul}^{w} + 2I_{add}^{2w+w}\right)\right] + \right.$$
$$\left. + I_{assign}^{2w} + I_{assign}^{w} + \left(2n-1\right)\left[3I_{assign}^{2w} + 3I_{add}^{2w+w}\right],\right.$$

де Z – число потоків.

3.1.2. Методи піднесення до квадрату

Нижче представлені метод піднесення до квадрату, в якому для накопичення переносу відведено блок розміром з машинне слово w $(r = v = w)$ і два методи, в яких застосовуються два різних підходи до розпаралелювання [13-18].

Метод 3.4. *Метод піднесення до квадрату на основі методу множення Modified Comba*

На вхід подається велике ціле число a, яке представляється у вигляді машинних слів $a = (a_n, ..., a_i, ..., a_1, a_0)$ розміром w біт, n – кількість машинних слів необхідних для представлення числа a. На виході отримуємо результат $c = a \cdot a$ розміром $2n$ машинних слів.

Основу вказаного методу піднесення до квадрату складають два цикли формування результату множення, які виконуються паралельно в два потоки [16]. Перший цикл формує результат в інтервалі $k = [\overline{0, n})$ та містить вкладений цикл множення в інтервалах $i = \left\lfloor \overline{0, k/2} \right\rfloor$ та $j = \left\lceil \overline{k, k/2} \right\rceil$. Другий цикл формує

результат в інтервалі $k=[\overline{n,2n-1})$ з використанням допоміжного інтервалу $l=[\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах i=$\left\lfloor \overline{l,k/2-l} \right\rfloor$ та j=$\left\lceil \overline{k-l,k/2-l} \right\rceil$. У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$ - розрядних заздалегідь проініціалізованих тимчасових змінних r_0 та r_1 на кожній ітерації циклу. При цьому, при i<j, для виключення повторень при накопиченні суми:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + \left(v^{(w)} << 1\right),\ r_1^{(2w)} \leftarrow r_1^{(2w)} + \left(\left(u^{(w)} << 1\right) OR \left(u^{(w)} >> (w-1)\right)\right),$$

а при i = j:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)},\ r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується корегування (врахування переносу) з використанням $2w$ - розрядних тимчасових змінних r_1 та r_2 (r_2 заздалегідь проініціалізована):

$$r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right),\ r_2^{(2w)} \leftarrow r_2^{(2w)} + \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right).$$

та відбувається присвоєння кінцевого результату і зміна тимчасових змінних r_0, r_1 та r_2

$$c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right),\ r_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right),\ r_2^{(2w)} \leftarrow 0.$$

Після завершення циклів формування результату множення відбувається формування результату $c_{2n-1}^{(w)} \leftarrow low_{(w)}(r_0^{(2w)})$.

Результатом піднесення до квадрату є ціле число $c=(c_{2n-1},...,c_k,...,c_1,c_0)$.

Вхід: $a \in \mathrm{GF}(p)$, $n = \log_{2^w} a$, $nk = 2n-1$, w – довжина машинного слова.
Вихід: $c = a \cdot a$.

1. $r_0^{(2w)} \leftarrow 0$, $r_1^{(2w)} \leftarrow 0$, $r_2^{(2w)} \leftarrow 0$.
2. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do
2.1. For $i \leftarrow 0$, $j \leftarrow k$, $i \le j$, $i{+}{+}$, $j{-}{-}$ do
2.1.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$.
2.1.2. if $(i < j)$ then $r_0^{(2w)} \leftarrow r_0^{(2w)} + \left(u^{(w)} << 1\right)$,
$r_1^{(2w)} \leftarrow r_1^{(2w)} + \left(\left(u^{(w)} << 1\right) \ or \ \left(u^{(w)} >> (w-1)\right)\right)$;
else $r_0^{(2w)} \leftarrow r_0^{(2w)} + u^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$.
2.2. $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$, $r_2^{(2w)} \leftarrow r_2^{(2w)} + \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right)$
2.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$,
$r_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right)$, $r_2^{(2w)} \leftarrow 0$.
3. For $k \leftarrow n$, $l \leftarrow 1, k < nk$, $k{+}{+}$, $l{+}{+}$ do
3.1. For $i \leftarrow l$, $j \leftarrow k - l$, $i \le j$, $i{+}{+}$, $j{-}{-}$ do
3.1.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$.
3.1.2. if $(i < j)$ then $r_0^{(2w)} \leftarrow r_0^{(2w)} + \left(u^{(w)} << 1\right)$,
$r_1^{(2w)} \leftarrow r_1^{(2w)} + \left(\left(u^{(w)} << 1\right) \ or \ \left(u^{(w)} >> (w-1)\right)\right)$;
else $r_0^{(2w)} \leftarrow r_0^{(2w)} + u^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$
3.2. $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$, $r_2^{(2w)} \leftarrow r_2^{(2w)} + \mathrm{hi}_{(2)}\left(r_1^{(2w)}\right)$
3.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_2^{(2w)}\right)$, $r_2^{(2w)} \leftarrow 0$.
4. $c_{nk}^{(w)} \leftarrow \mathrm{low}_{(2)}\left(r_0^{(2w)}\right)$.
5. Return (c).

Рис. 3.7. Псевдокод методу піднесення до квадрату на основі методу множення Modified Comba

Обчислювальна складність запропонованого методу представлена нижче:

$$I_{sqr}(A_{3.4}) = n^2\left(I_{mul}^w + \left(\frac{3n+1}{n}\right)I_{add}^{2w+w} + 3\left(\frac{n-1}{2n}\right)I_{shift}^w\right).$$

Метод 3.5. *Метод піднесення до квадрату на основі методу множення Modified Comba з розпаралелюванням в два потоки*

На вхід подається велике ціле число a, яке представляється у вигляді машинних слів $a=(a_n,...,a_i,...,a_1,a_0)$ розміром w біт, n – кількість машинних слів необхідних для представлення числа a. На виході отримуємо результат $c=a\cdot a$ розміром $2n$ машинних слів.

Основу вказаного методу піднесення до квадрату складають два цикли формування результату множення, які виконуються паралельно в два потоки. Перший цикл формує результат в інтервалі $k=[\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i=\left\lfloor \overline{0,k/2} \right\rfloor$ и $j=\left\lceil \overline{k,k/2} \right\rceil$. Другий цикл формує результат в інтервалі $k=[\overline{n,2n-1})$ з використанням допоміжного інтервалу $l=[\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах i=$\left\lfloor \overline{l,k/2-l} \right\rfloor$ та j=$\left\lceil \overline{k-l,k/2-l} \right\rceil$. Кожний потік використовує власні тимчасові змінні $rl_0{}^{(2w)}$, $rl_1{}^{(2w)}$ та $rl_2{}^{(2w)}$, які заздалегідь проініціалізовані. У вкладених циклах виконується множення $(uv)^{(2w)}\leftarrow a_i^{(w)}\cdot a_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних тимчасових змінних rl_0 та rl_1 на кожній ітерації циклу. При цьому, при i<j, для виключення повторень при накопиченні суми:

$$rl_0^{(2w)}\leftarrow rl_0^{(2w)}+\left(v^{(w)}<<1\right),\ rl_1^{(2w)}\leftarrow rl_1^{(2w)}+\left(\left(u^{(w)}<<1\right)OR\left(u^{(w)}>>(w-1)\right)\right),$$

а при i = j:

$$rl_0^{(2w)}\leftarrow rl_0^{(2w)}+v^{(w)},\ rl_1^{(2w)}\leftarrow rl_1^{(2w)}+u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу) з використанням $2w$ -розрядних тимчасових змінних rl_1 та rl_2:

$$rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \mathrm{hi}_{(w)}\left(rl_0^{(2w)}\right),\ rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \mathrm{hi}_{(w)}\left(rl_1^{(2w)}\right).$$

та відбувається присвоєння кінцевого результату і зміна тимчасових змінних rl_0 , rl_1 та rl_2

$$c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(rl_0^{(2w)}\right),\ rl_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_1^{(2w)}\right),\ rl_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_2^{(2w)}\right),\ rl_2^{(2w)} \leftarrow 0.$$

Після завершення першого циклу формування результату піднесення до квадрату для збереження значення можливого переносу використовується глобальна змінна r_0 : $r_0^{(2w)} \leftarrow rl_0^{(2w)}$.

Після завершення роботи двох паралельних потоків, необхідно виконати корекцію результатів роботи другого циклу формування результатів піднесення до квадрату за рахунок переносу, отриманого в результаті роботи першого циклу. Для цього використовується цикл корекції результатів в інтервалі $k = [\overline{n, 2n-1})$:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + c_k^{(w)},\ c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ \mathrm{low}_{(w)}\left(r_0^{(2w)}\right) \leftarrow hi_{(w)}\left(r_0^{(2w)}\right), hi_{(w)}\left(r_0^{(2w)}\right) \leftarrow 0.$$

Після завершення циклу корекції відбувається формування результату $c_{2n-1}^{(w)}$: $c_{2n-1}^{(w)} \leftarrow c_{2n-1}^{(w)} + low_{(w)}(r_0^{(2w)})$.

Результатом піднесення до квадрату є ціле число $c = (c_{2n-1}, ..., c_k, ..., c_1, c_0)$.

Вхід: $a \in \mathrm{GF}(p)$, $n = \log_{2^w} a$, $nk = 2n-1$, w – довжина машинного слова. *Вихід*: $c = a \cdot a$.
1. #pragma omp parallel sections private($r_0^{(2w)}, r_1^{(2w)}$) begin 1.1. #pragma omp section begin 1.1.1. $rl_0^{(2w)} \leftarrow 0$, $rl_1^{(2w)} \leftarrow 0$, $rl_2^{(2w)} \leftarrow 0$. 1.1.2. For $k \leftarrow 0$, $k < n$, $k{+}{+}$ do 1.1.2.1. For $i \leftarrow 0$, $j \leftarrow k$, $i \le k$, $i{+}{+}$, $j{-}{-}$ do 1.1.2.1.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$. 1.1.2.1.2. if $(i < j)$ then $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + \left(v^{(w)} << 1\right)$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \left(\left(u^{(w)} << 1\right) OR \left(u^{(w)} >> (w-1)\right)\right)$, else $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$.

1.1.2.2. $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \mathrm{hi}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \mathrm{hi}_{(w)}\left(rl_1^{(2w)}\right)$

1.1.2.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_1^{(2w)}\right)$, $rl_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_2^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow 0$.

1.1.3. $r_0^{(2w)} \leftarrow rl_0^{(2w)}$.

#pragma omp section end

1.2. #pragma omp section begin

1.2.1. $rl_0^{(2w)} \leftarrow 0$, $rl_1^{(2w)} \leftarrow 0$, $rl_2^{(2w)} \leftarrow 0$.

1.2.2. For $k \leftarrow n$, $l \leftarrow 1, k < nk$, $k++$, $l++$ do

1.2.2.1. For $i \leftarrow l$, $j \leftarrow n-1$, $i < n$, $i++$, $j--$ do

1.2.2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot b_j^{(w)}$.

1.2.2.1.2. if $(i < j)$ then $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + \left(v^{(w)} << 1\right)$,

$rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \left(\left(u^{(w)} << 1\right) OR \left(u^{(w)} >> (w-1)\right)\right)$,

else $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$.

1.2.2.2. $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \mathrm{hi}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow rl_2^{(2w)} + \mathrm{hi}_{(w)}\left(rl_1^{(2w)}\right)$

1.2.2.3. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(rl_0^{(2w)}\right)$, $rl_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_1^{(2w)}\right)$, $rl_1^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(rl_2^{(2w)}\right)$, $rl_2^{(2w)} \leftarrow 0$.

#pragma omp section end

#pragma omp parallel sections end

2. For $k \leftarrow n$, $k < nk$, $k++$ do

2.1. $r_0^{(2w)} \leftarrow r_0^{(2w)} + c_k^{(w)}$.

2.2. $c_k^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$.

2.3. $\mathrm{low}_{(w)}\left(r_0^{(2w)}\right) \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.

2.4. $\mathrm{hi}_{(w)}\left(r_0^{(2w)}\right) \leftarrow 0$.

3. $c_{nk}^{(w)} \leftarrow c_{nk}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$.

4. Return (c).

Рис. 3.8. Псевдокод методу 3.5

Обчислювальна складність запропонованого методу представлена нижче:

$$I_{sqr}(A_{3.5}) = \max\left[n\left(\left\lceil \tfrac{n}{2} \right\rceil \left(I_{mul}^{w} + 2\left(\frac{n+1}{2n} \right) I_{add}^{2w+w} + 3\left(\frac{n-1}{2n} \right) I_{shift}^{w} \right) + 2I_{add}^{2w+w} \right), \right.$$

$$\left. n\left(\left\lfloor \tfrac{n}{2} \right\rfloor \left(I_{mul}^{w} + 2\left(\frac{n+1}{2n} \right) I_{add}^{2w+w} + 3\left(\frac{n-1}{2n} \right) I_{shift}^{w} \right) + 2I_{add}^{2w+w} \right) \right] + \tfrac{n}{2} I_{add}^{2w+w} + I_{add}^{w}.$$

Метод 3.6. *Метод піднесення до квадрату на основі методу множення Modified Comba з розпаралелюванням в декілька потоків*

На вхід подається велике ціле число a, яке представляється у вигляді машинних слів $a = (a_n, ..., a_i, ..., a_1, a_0)$ розміром w біт, n – кількість машинних слів необхідних для представлення числа a. На виході отримуємо результат $c = a \cdot a$ розміром $2n$ машинних слів.

Основу вказаного методу піднесення до квадрату складають два цикли формування результату множення, ітерації яких не залежать одна від одної і можуть виконуватись в окремих потоках. Перший цикл формує результат в інтервалі $k = [\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i = \left\lfloor \overline{0,k/2} \right\rfloor$ та $j = \left\lceil \overline{k,k/2} \right\rceil$. Другий цикл формує результат в інтервалі $k = [\overline{n,2n-1})$ з використанням допоміжного інтервалу $l = [\overline{1,n-1})$ та містить вкладений цикл множення в інтервалах i=$\left\lfloor \overline{l,k/2-l} \right\rfloor$ та j=$\left\lceil \overline{k-l,k/2-l} \right\rceil$. Кожний потік в рамках ітерації використовує заздалегідь проініціалізовані масиви $r0_i^{(2w)}$ та $r1_i^{(2w)}$, де $i = [\overline{0,2n-1})$. У вкладених циклах використовуються власні тимчасові змінні rl_0 та rl_1, які ініціалізуються на кожній ітерації циклів формування результату. У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$, результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних змінних rl_0 та rl_1 на кожній ітерації циклу. При цьому, при i<j, для виключення повторень при накопиченні суми:

$$rl_0^{(2w)} \leftarrow rl_0^{(2w)} + \left(v^{(w)} << 1\right),\ rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \left(\left(u^{(w)} << 1\right) OR \left(u^{(w)} >> (w-1)\right)\right),$$

а при i = j:

$$rl_0^{(2w)} \leftarrow rl_0^{(2w)} + v^{(w)},\ rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується збереження переносів:

$$r0_k^{(2w)} \leftarrow rl_0^{(2w)},\ r1_k^{(2w)} \leftarrow rl_1^{(2w)}.$$

Після завершення циклів формування результатів, необхідно виконати цикл врахування збережених переносів в інтервалі $k = [\overline{0, 2n-1})$, з використанням заздалегідь проініціалізованих тимчасових змінних $r^{(2w)}$, $rll_0^{(2w)}$, $rll_1^{(2w)}$:

$$rll_0^{(2w)} \leftarrow r0_k^{(2w)},\ rll_1^{(2w)} \leftarrow r1_k^{(2w)}\ \ rll_0^{(2w)} \leftarrow rll_0^{(2w)} + low_{(w)}(r^{(2w)}),$$

$$c_k^{(w)} \leftarrow low_{(w)}(rll_0^{(2w)}),\ rll_1^{(2w)} \leftarrow rll_1^{(2w)} + low_{(w)}(rll_0^{(2w)}),\ r^{(2w)} \leftarrow rll_1^{(2w)}.$$

Після завершення циклу врахування збережених переносів відбувається формування результату $c_{2n-1}^{(w)}$:

$$c_{2n-1}^{(w)} \leftarrow low_{(w)}(r^{(2w)}).$$

Результатом піднесення до квадрату є ціле число $c = (c_{2n-1}, ..., c_k, ..., c_1, c_0)$.

Вхід: $a \in \mathrm{GF}(p)$, $n = \log_{2^w} a$, $nk = 2n-1$, w – довжина машинного слова. *Вихід*: $c = a \cdot a$.
1. $l \leftarrow 1$. 2. Arrays $r0_i^{(w)}$ and $r1_i^{(w)}$, $i = \overline{0, nk}$. 3. #pragma omp parallel private ($r0_i^{(w)}, r1_i^{(w)}$) reduction (+ : l) begin 4. #pragma omp for nowait begin 4.1. For $k \leftarrow 0$, $k < n$, $k++$ do 4.1.1. $rl_0^{(w)} \leftarrow 0$, $rl_1^{(w)} \leftarrow 0$. 4.1.2. For $i \leftarrow 0$, $j \leftarrow k$, $i \le j$, $i++$, $j--$ do 4.1.2.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$. 4.1.2.2. if $(i < j)$ then $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + \left(u^{(w)} << 1\right)$,

$rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \left(\left(u^{(w)} << 1 \right) \ or \ \left(u^{(w)} >> (w-1) \right) \right)$;

else $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + u^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$

4.1.3. $r0_k^{(2w)} \leftarrow rl_0^{(2w)}$, $r1_k^{(2w)} \leftarrow rl_1^{(2w)}$

#pragma omp for end

5. #pragma omp for nowait begin

5.1. For $k \leftarrow n$, $k < nk$, $k++$ do

5.1.1. $rl_0^{(2w)} \leftarrow 0$, $rl_1^{(2w)} \leftarrow 0$.

5.1.2. For $i \leftarrow l$, $j \leftarrow n-1$, $i \le j$, $i++$, $j--$ do

5.1.2.1. $\left(u^2\right)^{(2w)} \leftarrow a_i^{(w)} \cdot a_j^{(w)}$.

5.1.2.2. if $(i < j)$ then $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + \left(u^{(w)} << 1\right)$,

$rl_1^{(2w)} \leftarrow rl_1^{(2w)} + \left(\left(u^{(w)} << 1 \right) \ or \ \left(u^{(w)} >> (w-1) \right) \right)$;

else $rl_0^{(2w)} \leftarrow rl_0^{(2w)} + u^{(w)}$, $rl_1^{(2w)} \leftarrow rl_1^{(2w)} + u^{(w)}$

5.1.3. $r0_k^{(2w)} \leftarrow rl_0^{(2w)}$, $r1_k^{(2w)} \leftarrow rl_1^{(2w)}$

5.1.4. $l++$.

#pragma omp for end

#pragma omp parallel end

6. $r_0^{(2w)} \leftarrow 0$.

7. For $k \leftarrow 0$, $k < nk$, $k++$ do

7.1. $rll_0^{(2w)} \leftarrow r0_k^{(2w)}$, $rll_1^{(2w)} \leftarrow r1_k^{(2w)}$.

7.2. $rll_0^{(2w)} \leftarrow rll_0^{(2w)} + \text{low}_{(w)}\left(r^{(2w)}\right)$.

7.3. $c_k^{(w)} \leftarrow \text{low}_{(w)}\left(rll_0^{(2w)}\right)$.

7.4. $rll_1^{(2w)} \leftarrow rll_1^{(2w)} + \text{low}_{(w)}\left(rll_0^{(2w)}\right)$.

7.5. $r^{(2w)} \leftarrow rll_1^{(2w)}$.

8. $c_{nk}^{(w)} \leftarrow \text{low}_{(w)}\left(r^{(2w)}\right)$.

9. Return (c).

Рис. 3.9. Псевдокод методу 3.6

Обчислювальна складність запропонованого методу:

$$I_{mul}\left(A_{3.6}\right) = \frac{n}{Z}\left[\left\lceil \frac{n}{2} \right\rceil \left(I_{mul}^{w} + 2\left(\frac{n+1}{2n}\right) I_{add}^{2w+w} + 3\left(\frac{n-1}{2n}\right) I_{shift}^{w} \right)\right] +$$

$$+\frac{n}{Z}\left[\left\lfloor\frac{n}{2}\right\rfloor\left(I_{mul}^{w}+2\left(\frac{n+1}{2n}\right)I_{add}^{2w+w}+3\left(\frac{n-1}{2n}\right)I_{shift}^{w}\right)\right]+(2n-1)3I_{add}^{2w+w}$$

де Z – кількість потоків.

3.1.3. Методи приведення за модулем

Нижче представлені метод приведення за модулем, в якому для накопичення переносу відведено блок розміром з машинне слово w $(r=v=w)$ та два методи, в яких використовуються два різних підходи до розпаралелювання [19-23]. В якості прототипу вибрано приведення за модулем методом Барретта [24, 25].

Метод 3.7. *Метод приведення за модулем з використанням методу множення Modified Comba*

На вхід подається число $a=(a_{n2-1},...,a_i,...,a_1,a_0)$, яке представлено як добуток цілих чисел d і f, та просте число $p=(p_{g-1},...,p_i,...,p_1,p_0)$. Числа a та p подаються у вигляді наборів машинних слів, де a_i та p_i – відповідні машинні слова чисел a та p, $n2$ – кількість машинних слів, необхідних для представлення числа a ($n2=\lceil\log_b a\rceil$), g – кількість машинних слів, необхідних для представлення числа p ($g=\lceil\log_b p\rceil$), n – кількість машинних слів, необхідних для представлення чисел d та f ($n=\log_{2^w} d=\log_{2^w} f$), b – число, що займає w біт ($b=2^w$), w – розмір машинного слова в бітах (наприклад $w=32$). Для виконання операції приведення за модулем, необхідно виконати попереднє обчислення константи $\mu=\left\lfloor\frac{b^{2k}}{p}\right\rfloor$. Після цього, необхідно виконати послідовно два часткових множення методом Comba, з використанням відкладеного переносу [20]. Перед кожним частковим множенням відбувається ініціалізація $2w$- розрядних тимчасових змінних $r_0^{(2w)}$ та $r_1^{(2w)}$: $r_0^{(2w)}\leftarrow 0$, $r_1^{(2w)}\leftarrow 0$.

Основу першого часткового множення складають два цикли формування результату множення. Перший цикл формує результат в інтервалі $x=[\overline{0,n2})$ та містить вкладений цикл множення в інтервалах $i=[\overline{k-1,kn2})$ та $j=[\overline{x,kn2-k+1})$, де $kn2\leftarrow\min\left(k+x;n2\right)$. Другий цикл формує результат в інтервалі $x=[\overline{n2,n3})$ з використанням допоміжного інтервалу $e=[\overline{k,k+n})$, де $n3=n+n2$, та містить вкладений цикл множення в інтервалах $i=[\overline{e,n2})$ та $j=[\overline{n2-1,e-1})$. У вкладених циклах виконується множення $\left(uv\right)^{(2w)}\leftarrow a_i^{(w)}\cdot\mu_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w -розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$ - розрядних тимчасових змінних $r_0^{(2w)}$ та $r_1^{(2w)}$ на кожній ітерації:

$$r_0^{(2w)}\leftarrow r_0^{(2w)}+v^{(w)},\ r_1^{(2w)}\leftarrow r_1^{(2w)}+u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу):

$$q_k^{(w)}\leftarrow\text{low}_{(w)}\left(r_0^{(2w)}\right),\ r_1^{(2w)}\leftarrow r_1^{(2w)}+\text{hi}_{(w)}\left(r_0^{(2w)}\right),$$

та відбувається зміна тимчасових змінних $r_0^{(2w)}$ та $r_1^{(2w)}$:

$$r_0^{(2w)}\leftarrow\text{low}_{(w)}\left(r_1^{(2w)}\right),\ r_1^{(2w)}\leftarrow\text{hi}_{(w)}\left(r_1^{(2w)}\right).$$

Отримане в результаті першого часткового множення значення q використовується в другому частковому множенні. Основу другого часткового множення також складають два цикли формування результату множення. Перший цикл формує результат в інтервалі $x=[\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i=[\overline{0,x}]$ та $j=[\overline{x,0}]$. Другий цикл формує результат в інтервалі $x=[\overline{n,k+1})$ з використанням допоміжного інтервалу $e=[\overline{1,k-n})$, та містить вкладений цикл множення в інтервалах $i=[\overline{e,n})$ та $j=[\overline{n-1,e-1})$. У вкладених циклах виконується множення

$(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних тимчасових змінних $r_0^{(2w)}$ та $r_1^{(2w)}$ на кожній ітерації:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)},\ r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується корегування (врахування переносу):

$$q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right),$$

та відбувається зміна тимчасових змінних $r_0^{(2w)}$ та $r_1^{(2w)}$:

$$r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right),\ r_1^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right).$$

По завершенню двох часткових множень відбувається формування найстаршого значення $q_{k+1}^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$

Щоб врахувати можливий перенос, необхідно спочатку визначити значення індексу j (j знаходиться в інтервалі $[\overline{k,0}]$), для якого перестає виконуватися умова $q_j^{(w)} == a_j^{(w)}$. Далі, якщо $q_j^{(w)} > a_j^{(w)}$ то необхідно врахувати перенос у великому числі $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$, інакше перенос не враховується у великому числі $r \leftarrow a \bmod \left(2^w\right)^{k+1}$.

Для обчислення залишку c, виконується віднімання одного великого числа від іншого: $c \leftarrow r - q$. У випадку, якщо отриманий залишок $c \geq p$, необхідно виконати корекцію $c \leftarrow c - p$ поки r не стане менше p. На виході отримується результат $c = a \bmod p$.

Вхід: ціле $a = d \cdot b$, $d, b \in \mathrm{GF}(p)$, $n_2 = \log_{2^w} a$, $n = \log_{2^w} d = \log_{2^w} b$, $n_3 = n + n_2$. $k = \left\lfloor \log_{2^w} p \right\rfloor + 1$, $\mu = \left\lfloor \left(2^w\right)^{2k} \Big/ p \right\rfloor$, w – довжина машинного слова.

Вихід: $c = a \bmod p$, $c \in \mathrm{GF}(p)$.

1. $r_0^{(2w)} \leftarrow 0$, $r_1^{(2w)} \leftarrow 0$

2. For $x \leftarrow 0$, $x < n_2$, $x{+}{+}$ do

2.1. $kn_2 \leftarrow \min(k + x; n_2)$

2.2. For $i \leftarrow k-1$, $j \leftarrow x$, $i < kn_2$, $i{+}{+}$, $j{-}{-}$ do

2.2.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$.

2.2.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$.

2.3. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.

2.4. $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right)$.

3. For $x \leftarrow n_2$, $e \leftarrow k$, $x < n_3$, $x{+}{+}$, $e{+}{+}$ do

3.1. For $i \leftarrow e$, $j \leftarrow n_2 - 1$, $i < n_2$, $i{+}{+}$, $j{-}{-}$ do

3.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$.

3.1.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$.

3.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.

3.3. $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right)$.

4. $r_0^{(2w)} \leftarrow 0$, $r_1^{(2w)} \leftarrow 0$.

5. For $x \leftarrow 0$, $x < n$, $x{+}{+}$ do

5.1. For $i \leftarrow 0$, $j \leftarrow x$, $i \le x$, $i{+}{+}$, $j{-}{-}$ do

5.1.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$.

5.1.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$.

5.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.

5.3. $r_0^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1^{(2w)}\right)$.

6. For $x \leftarrow n$, $e \leftarrow 1$, $x < k+1$, $x{+}{+}$, $e{+}{+}$ do

6.1. For $i \leftarrow e$, $j \leftarrow n-1$, $i < n$, $i{+}{+}$, $j{-}{-}$ do

6.1.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$.

6.1.2. $r_0^{(2w)} \leftarrow r_0^{(2w)} + v^{(w)}$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + u^{(w)}$.

6.2. $q_x^{(w)} \leftarrow \text{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_1^{(2w)} \leftarrow r_1^{(2w)} + \text{hi}_{(w)}\left(r_0^{(2w)}\right)$.

6.3. $r_0^{(2w)} \leftarrow \text{low}_{(w)}\left(r_1^{(2w)}\right)$, $r_1^{(2w)} \leftarrow \text{hi}_{(w)}\left(r_1^{(2w)}\right)$.

7. $q_{k+1}^{(w)} \leftarrow \text{low}_{(w)}\left(r_0^{(2w)}\right)$.

8. $j \leftarrow k$

9. For , $\left(j \geq 0\right) and \left(q_j^{(w)} == a_j^{(w)}\right)$, $j--$ do.

10. If $\left(q_j^{(w)} > a_j^{(w)}\right)$

10. 1. $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$.

11. Else

11.1. $r \leftarrow a \bmod \left(2^w\right)^{k+1}$

12. $c \leftarrow r - q$

13. While $(c \geq p)$ do

13.1. $c \leftarrow c - p$

14. Return (c).

Рис. 3.10. Псевдокод методу 3.7

Розглянемо обчислювальну складність запропонованого методу. Обчислювальна складність основних частин методу:

- перше часткове множення (п.2 – п.4) –

$$2I_{a\text{sgn}}^{2w} + 10n \cdot I_{xor}^{w} + 8n \cdot I_{xor}^{2w} + 8n \cdot I_{sh}^{2w} + 2n \cdot I_{add}^{w} +$$

$$+ I_{mul}^{2w}\left(6n^2 - \frac{1}{2}k^2 + \frac{1}{2}k - \frac{5}{2}nk - \frac{7}{2}n\right) + I_{add}^{2w}\left(12n^2 - k^2 - 5nk - 5n + k + 1\right)$$

- друге часткове множення (п.7 – п.8) –

$$2I_{a\text{sgn}}^{2w} + 8n \cdot I_{xor}^{2w} + 8n \cdot I_{sh}^{2w} + I_{mul}^{2w}\left(n\left\lfloor\frac{1+n}{2}\right\rfloor + \frac{n}{2}(n-1)\right) +$$

$$+ I_{add}^{2w}\left(2n\left\lfloor\frac{1+n}{2}\right\rfloor + 3n + n^2\right)$$

- подальша корекція (п.11 – п.16) –

$$6I^{w}_{asqn} + I^{w}_{cmp}\left(\frac{5}{2}k + \frac{1}{2}\right) + \frac{1}{2}I^{w}_{dec} + I^{w}_{sub}\cdot\frac{3}{2}k + k\cdot I^{w}_{add} + \frac{1}{2}k\cdot I^{w}_{mul}.$$

Обчислювальна складність запропонованого методу приведення за модулем:

$$I_{\mathrm{mod}}\left(A_{3.7}\right) = 4n\cdot I^{2w}_{a\,\mathrm{sgn}} + \left(16n+1\right)\cdot I^{2w}_{xor} + 16n\cdot I^{2w}_{sh} +$$
$$+I^{2w}_{mul}\left(\frac{13}{2}n^2 - \frac{1}{2}k^2 + \frac{1}{2}k - \frac{5}{2}nk - 4n + n\left\lfloor\frac{1+n}{2}\right\rfloor\right) +$$
$$+I^{2w}_{add}\left(13n^2 - k^2 - 5nk - 2n + k + 1 + 2n\left\lfloor\frac{1+n}{2}\right\rfloor\right) +$$
$$+6I^{w}_{asqn} + 10n\cdot I^{w}_{xor} + \left(\frac{5}{2}k + \frac{1}{2}\right)I^{w}_{cmp} + \frac{1}{2}I^{w}_{dec} + \frac{3}{2}k\cdot I^{w}_{sub} + \left(2n+k\right)\cdot I^{w}_{add} + \frac{1}{2}I^{w}_{mul}$$

Метод 3.8. *Метод приведення за модулем з використанням методу множення Modified Comba з розпаралелюванням в два потоки*

На вхід подається число $a = (a_{n2-1},...,a_i,...,a_1,a_0)$, яке представлено як добуток цілих чисел d та f, и просте число $p = (p_{g-1},...,p_i,...,p_1,p_0)$. Числа a та p подаються у вигляді наборів машинних слів, де a_i та p_i – відповідні машинні слова чисел a та p, $n2$ – кількість машинних слів, необхідних для представлення числа a ($n2 = \lceil \log_b a \rceil$), g – кількість машинних слів, необхідних для представлення числа p ($g = \lceil \log_b p \rceil$), n – кількість машинних слів, необхідних для представлення чисел d та f ($n = \log_{2^w} d = \log_{2^w} f$), b – число, що займає w біт ($b = 2^w$), w – розмір машинного слова в бітах (наприклад, $w = 32$). Для виконання операції приведення за модулем, необхідно виконати попереднє обчислення константи $\mu = \left\lfloor \frac{b^{2k}}{p} \right\rfloor$ та проініціалізувати $2w$ - розрядну тимчасову змінну $r_0^{(2w)}$ ($r_0^{(2w)} \leftarrow 0$). Після цього, необхідно виконати послідовно два часткових множення методом Comba, з використанням відкладеного переносу.

Основу першого часткового множення складають два цикли формування результату множення, які виконуються паралельно, в два потоки.

Перший цикл формує результат в інтервалі $x=[\overline{0,n2})$ та містить вкладений цикл множення в інтервалах $i=[\overline{k-1,kn2})$ та $j=[\overline{x,kn2-k+1})$, де $kn2 \leftarrow \min(k+x;n2)$. Другий цикл формує результат в інтервалі $x=[\overline{n2,n3})$ з використанням допоміжного інтервалу $e=[\overline{k,k+n})$, де $n3=n+n2$, та містить вкладений цикл множення в інтервалах $i=[\overline{e,n2})$ та $j=[\overline{n2-1,e-1})$. Кожний потік використовує власні тимчасові змінні $r_0'^{(2w)}$ та $r_1'^{(2w)}$, які заздалегідь проініціалізовані ($r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$). У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ та $v^{(w)}$.

Накопичення відкладеного переносу відбувається в старших частинах $2w$- розрядних змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$ на кожній ітерації:

$$r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу) з використанням $2w$ - розрядних тимчасових змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$:

$$q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$$

та відбувається зміна тимчасових змінних $r_0'^{(2w)}$ і $r_1'^{(2w)}$:

$$r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right).$$

Після завершення першого циклу формування результату множення для збереження значення можливого переносу використовується глобальна змінна $r_0^{(2w)}$: $r_0^{(2w)} \leftarrow r_0'^{(2w)}$.

Після завершення роботи двох паралельних потоків, необхідно виконати корекцію результатів роботи другого циклу формування результатів множення за рахунок переносу, отриманого в результаті роботи першого

циклу. Для цього використовується цикл корекції результатів в інтервалі $x=[\overline{n2,n3})$:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + q_x^{(w)},\ q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_0^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right).$$

Після завершення циклу корекції відбувається формування значення $q_{n3}^{(w)}$:

$$q_{n3}^{(w)} \leftarrow q_{n3}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right).$$

Отримане в результаті першого часткового множення значення q використовується в другому частковому множенні. Основу другого часткового множення також складають два цикли формування результату множення. Перший цикл формує результат в інтервалі $x=[\overline{0,n})$ та містить вкладений цикл множення в інтервалах $i=[\overline{0,x}]$ та $j=[\overline{x,0}]$. Другий цикл формує результат в інтервалі $x=[\overline{n,k+1})$ з використанням допоміжного інтервалу $e=[\overline{1,k-n})$, та містить вкладений цикл множення в інтервалах $i=[\overline{e,n})$ и $j=[\overline{n-1,e-1})$. Кожний потік використовує власні тимчасові змінні $r_0'^{(2w)}$ и $r_1'^{(2w)}$, які заздалегідь проініціалізовані ($r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$).

У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w - розрядних $u^{(w)}$ та $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$- розрядних змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$ на кожній ітерації:

$$r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується коригування (врахування переносу) з використанням $2w$ - розрядних тимчасових змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$:

$$q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$$

та відбувається зміна тимчасових змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$:

$$r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right).$$

Після завершення першого циклу формування результату другого часткового множення для збереження значення можливого переносу використовується глобальна змінна $r_0^{(2w)}$: $r_0^{(2w)} \leftarrow r_0'^{(2w)}$.

Після завершення роботи двох паралельних потоків, необхідно виконати корекцію результатів роботи другого циклу формування результатів множення за рахунок переносу, отриманого в результаті роботи першого циклу. Для цього використовується цикл корекції результатів в інтервалі $x = [\overline{n, k+1})$:

$$r_0^{(2w)} \leftarrow r_0^{(2w)} + q_x^{(w)},\ q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_0^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right).$$

Після завершення циклу корекції відбувається формування значення $q_{n3}^{(w)}$:

$$q_{n3}^{(w)} \leftarrow q_{n3}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right).$$

Щоб врахувати можливий перенос, необхідно спочатку визначити значення індексу j (j знаходиться в інтервалі $[\overline{k, 0}]$), для якого перестає виконуватися умова $q_j^{(w)} == a_j^{(w)}$. Далі, якщо $q_j^{(w)} > a_j^{(w)}$ то необхідно врахувати перенос у великому числі $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$, інакше перенос не враховується у великому числі $r \leftarrow a \bmod \left(2^w\right)^{k+1}$.

Для обчислення залишку c, виконується віднімання одного великого числа від іншого: $c \leftarrow r - q$. У випадку, якщо отриманий залишок $c \geq p$, необхідно виконати корекцію $c \leftarrow c - p$ доки r не стане менше p. На виході отримується результат $c = a \bmod p$.

Вхід: ціле $a = d \cdot b$, $d, b \in \mathrm{GF}(p)$, $n_2 = \log_{2^w} a$, $n = \log_{2^w} d = \log_{2^w} b$, $n_3 = n + n_2$. $k = \lfloor \log_{2^w} p \rfloor + 1$, $\mu = \left\lfloor \left(2^w\right)^{2k} / p \right\rfloor$, w – довжина машинного слова.

Вихід: $c = a \bmod p$, $c \in \mathrm{GF}(p)$.

1. $r_0^{(2w)} \leftarrow 0$.
2. #pragma omp parallel sections
2.1. #pragma omp section begin
2.1.1. $r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$.
2.1.2. For $x \leftarrow 0$, $x < n_2$, $x{+}{+}$ do
2.1.2.1. $kn_2 \leftarrow \min(k + x; n_2)$
2.1.2.2 For $i \leftarrow k - 1$, $j \leftarrow x$, $i < kn_2$, $i{+}{+}$, $j{-}{-}$ do
2.1.2.2.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$.
2.1.2.2.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)}$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$.
2.1.2.3. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right)$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$.
2.1.2.4. $r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right)$, $r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right)$.
2.1.3. $r_0^{(2w)} \leftarrow r_0'^{(2w)}$.
2.2. #pragma omp section begin
2.2.1. $r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$.
2.2.2. For $x \leftarrow n_2$, $e \leftarrow k$, $x < n_3$, $x{+}{+}$, $e{+}{+}$ do
2.2.2.1. For $i \leftarrow e$, $j \leftarrow n_2 - 1$, $i < n_2$, $i{+}{+}$, $j{-}{-}$ do
2.2.2.1.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$.
2.2.2.1.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)}$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$.
2.2.2.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right)$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$.
2.2.2.3. $r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right)$, $r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right)$.
3. For $x \leftarrow n_2$, $x < n_3$, $x{+}{+}$ do
3.1. $r_0^{(2w)} \leftarrow r_0^{(2w)} + q_x^{(w)}$.
3.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$, $r_0^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right)$.
4. $q_{n_3}^{(w)} \leftarrow q_{n_3}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right)$.
5. #pragma omp parallel sections
5.1. #pragma omp section begin

5.1.1. $r_0'^{(2w)} \leftarrow 0,\ r_1'^{(2w)} \leftarrow 0.$

5.1.2. For $x \leftarrow 0,\ x < n,\ x{+}{+}$ do

5.1.2.1. For $i \leftarrow 0,\ j \leftarrow x,\ i \le x,\ i{+}{+},\ j{-}{-}$ do

5.1.2.1.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$

5.1.2.1.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}.$

5.1.2.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right).$

5.1.2.3. $r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right).$

5.1.3. $r_0^{(2w)} \leftarrow r_0'^{(2w)}.$

5.2. #pragma omp section begin

5.2.1. $r_0'^{(2w)} \leftarrow 0,\ r_1'^{(2w)} \leftarrow 0.$

5.2.2. For $x \leftarrow n,\ e \leftarrow 1, x < k+1,\ x{+}{+},\ e{+}{+}$ do

5.2.2.1. For $i \leftarrow e,\ j \leftarrow n-1,\ i < n,\ i{+}{+},\ j{-}{-}$ do

5.2.2.1.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$

5.2.2.1.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}.$

5.2.2.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right).$

5.2.2.3. $r_0'^{(2w)} \leftarrow \mathrm{low}_{(w)}\left(r_1'^{(2w)}\right),\ r_1'^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_1'^{(2w)}\right).$

5.2.3. $q_{k+1}^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right).$

6. For $x \leftarrow n,\ x < k+1,\ x{+}{+}$ do

6.1. $r_0^{(2w)} \leftarrow r_0^{(2w)} + q_x^{(w)}.$

6.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0^{(2w)}\right),\ r_0^{(2w)} \leftarrow \mathrm{hi}_{(w)}\left(r_0^{(2w)}\right).$

7. $q_{k+1}^{(w)} \leftarrow q_{k+1}^{(w)} + \mathrm{low}_{(w)}\left(r_0^{(2w)}\right).$

8. $j \leftarrow k$

9. For , $\left(j \ge 0\right) and \left(q_j^{(w)} == a_j^{(w)}\right),\ j{-}{-}$ do.

10. If $\left(q_j^{(w)} > a_j^{(w)}\right)$

10. 1. $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}.$

11. Else

11.1. $r \leftarrow a \bmod \left(2^w\right)^{k+1}$

12. $c \leftarrow r - q$ // without modular reduction
13. While $(c \geq p)$ do
13.1. $c \leftarrow c - p$ // without modular reduction
14. Return (c).

Рис. 3.11. Псевдокод методу 3.8

Розглянемо обчислювальну складність запропонованого методу. Обчислювальна складність основних частин методу:

- перше часткове множення, перший потік (п.2.1.1 – п.2.2.2.3) – $3I_{a\text{sgn}}^{2w} + 10n \cdot I_{xor}^{w} + 4n \cdot I_{xor}^{2w} + 4n \cdot I_{sh}^{2w} + 2n \cdot I_{add}^{w} +$
$+ I_{mul}^{2w}\left(\frac{9}{2}n^2 - \frac{1}{2}k^2 - 2nk - 2n\right) + I_{add}^{2w}\left(5n^2 - k^2 - 4nk - 2n\right)$
- перше часткове множення, другий потік (п.2.2.1 – п.2.1.3) – $2I_{a\text{sgn}}^{2w} + 4n \cdot I_{xor}^{2w} + 4n \cdot I_{sh}^{2w} + 2n \cdot I_{add}^{w} +$
$+ I_{mul}^{2w} \cdot \frac{1}{2}(3n - k)(n-1) + I_{add}^{2w}(3n - k)(n-1)$
- врахування переносів після першого часткового множення (п.3 – 4) – $n \cdot I_{add}^{2w} + (n+1) \cdot I_{xor}^{2w} + n \cdot I_{sh}^{2w} + I_{add}^{w}$
- друге часткове множення, перший потік (п.5.1.1 – п.5.1.3) – $3I_{a\text{sgn}}^{2w} + n\left\lfloor \frac{1+n}{2} \right\rfloor \cdot I_{mul}^{2w} + \left(2n\left\lfloor \frac{1+n}{2} \right\rfloor + 2n\right) I_{add}^{2w} + 4n \cdot I_{xor}^{2w} + 4n \cdot I_{sh}^{2w}$
- друге часткове множення, другий потік (п.5.2.1 – п.5.2.3) – $2I_{a\text{sgn}}^{2w} + \frac{n(n-1)}{2} \cdot I_{mul}^{2w} + (n^2 + n) I_{add}^{2w} + 5n \cdot I_{xor}^{2w} + 4n \cdot I_{sh}^{2w}$
- врахування переносів після другого часткового множення (п.6 – 7) – $(k - n + 1) I_{add}^{2w} + (k - n + 2) I_{xor}^{2w} + (k - n + 1) I_{sh}^{2w} + I_{add}^{w}$
- подальша корекція (п.8 – п.14) – $6I_{asqn}^{w} + I_{cmp}^{w}\left(\frac{5}{2}k + \frac{1}{2}\right) + \frac{1}{2}I_{dec}^{w} + I_{sub}^{w} \cdot \frac{3}{2}k + k \cdot I_{add}^{w} + \frac{1}{2}k \cdot I_{mul}^{w}$.

Обчислювальна складність запропонованого методу приведення за модулем:

$$I_{\mathrm{mod}}\left(A_{3.8}\right)=11I_{asqn}^{2w}+\left(17n+k+2\right)I_{xor}^{2w}+\left(16n+k+1\right)I_{sh}^{2w}+$$
$$+\left(9n^2-k^2-5nk-2n+2k+2n\left\lfloor\frac{1+n}{2}\right\rfloor+1\right)\cdot I_{add}^{2w}+$$
$$+\frac{1}{2}\left(13n^2-k^2-5nk-8n+k+2n\left\lfloor\frac{1+n}{2}\right\rfloor\right)\cdot I_{mul}^{2w}+$$
$$+10nI_{xor}^{w}+6I_{asqn}^{w}+I_{cmp}^{w}\left(\frac{5}{2}k+\frac{1}{2}\right)+\frac{1}{2}I_{dec}^{w}+I_{sub}^{w}\cdot\frac{3}{2}k+$$
$$+\left(2n+k+2\right)\cdot I_{add}^{w}+\frac{1}{2}k\cdot I_{mul}^{w}$$

Метод 3.9. *Метод приведення за модулем з використанням методу множення Modified Comba з розпаралелюванням в декілька потоків*

На вхід подається число $a=(a_{n2-1},...,a_i,...,a_1,a_0)$, яке представлене як добуток цілих чисел d і f, та просте число $p=(p_{g-1},...,p_i,...,p_1,p_0)$. Числа a та p подаються у вигляді наборів машинних слів, де a_i та p_i – відповідні машинні слова чисел a та p, $n2$ – кількість машинних слів, необхідних для представлення числа a ($n2=\lceil\log_b a\rceil$), g – кількість машинних слів, необхідних для представлення числа p ($g=\lceil\log_b p\rceil$), n – кількість машинних слів, необхідних для представлення чисел d та f ($n=\log_{2^w} d=\log_{2^w} f$), b – число, що займає w біт ($b=2^w$), w – розмір машинного слова в бітах (наприклад, $w=32$). Для виконання операції приведення за модулем, необхідно виконати попереднє обчислення константи $\mu=\left\lfloor\frac{b^{2k}}{p}\right\rfloor$. Після цього, необхідно виконати послідовно два часткових множення методом Comba, з використанням відкладеного переносу.

Основу першого часткового множення складають два цикли формування результату множення, ітерації яких не залежать одна від одної і можуть виконуватись в окремих потоках. Перший цикл формує результат в інтервалі

$x=[\overline{0,n2})$ та містить вкладений цикл множення в інтервалах $i=[\overline{k-1,kn2})$ та $j=[\overline{x,kn2-k+1})$, де $kn2\leftarrow\min\left(k+x;n2\right)$. Другий цикл формує результат в інтервалі $x=[\overline{n2,n3})$ де $n3=n+n2$, та містить вкладений цикл множення в інтервалах $i=[\overline{x-(n2-k),n2})$ та $j=[\overline{n2-1,x-1-(n2-k)})$. Кожний потік в рамках ітерації використовує заздалегідь проініціалізовані масиви $\overline{r}_x^{(2w)}$ та $\overline{\overline{r}}_x^{(2w)}$ для накопичення відкладенного переносу. У вкладених циклах використовуються власні тимчасові змінні $r_0'^{(2w)}$ та $r_1'^{(2w)}$, які ініціалізуються на кожній ітерації циклів формування результату ($r_0'^{(2w)}\leftarrow 0$, $r_1'^{(2w)}\leftarrow 0$). У вкладених циклах виконується множення $\left(uv\right)^{(2w)}\leftarrow a_i^{(w)}\cdot\mu_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ і $v^{(w)}$.

Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$ на кожній ітерації:

$$r_0'^{(2w)}\leftarrow r_0'^{(2w)}+v^{(w)},\ r_1'^{(2w)}\leftarrow r_1'^{(2w)}+u^{(w)}.$$

На кожній ітерації циклу формування результату виконується збереження переносів:

$$\overline{r}_x^{(2w)}\leftarrow r_0'^{(2w)},\ \overline{\overline{r}}_x^{(2w)}\leftarrow r_1'^{(2w)}.$$

Після завершення циклів формування результатів, необхідно виконати цикл врахування збережених переносів в інтервалі $x=[\overline{0,n3})$, з використанням заздалегідь проініціалізованої змінної $r^{(2w)}$ ($r^{(2w)}\leftarrow 0$):

$$r_0'^{(2w)}\leftarrow\overline{r}_x^{(2w)}+r^{(2w)}.\ q_x^{(w)}\leftarrow\text{low}_{(w)}\left(r_0'^{(2w)}\right),\ r^{(2w)}\leftarrow\overline{\overline{r}}_x^{(2w)}+\text{hi}_{(w)}\left(r_0'^{(2w)}\right).$$

Отримане в результаті першого часткового множення значення q використовується в другому частковому множенні. Основу другого часткового множення також складають два цикли формування результату множення. Перший цикл формує результат в інтервалі $x=[\overline{0,n})$ та містить

вкладений цикл множення в інтервалах $i=[\overline{0,x}]$ та $j=[\overline{x,0}]$. Другий цикл формує результат в інтервалі $x=[\overline{n,k+1})$, та містить вкладений цикл множення в інтервалах $i=[\overline{x-(n-1),n})$ та $j=[\overline{n-1,x-n})$. Кожний потік використовує власні тимчасові змінні $r_0'^{(2w)}$ та $r_1'^{(2w)}$, які заздалегідь проініціалізовані ($r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$).

У вкладених циклах виконується множення $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$ результатом якого є ціле число розміром $2w$, яке потім розділяється на два w-розрядних $u^{(w)}$ і $v^{(w)}$. Накопичення відкладеного переносу відбувається в старших частинах $2w$-розрядних змінних $r_0'^{(2w)}$ та $r_1'^{(2w)}$ на кожній ітерації:

$$r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}.$$

На кожній ітерації циклу формування результату виконується збереження переносів:

$$\overline{r}_x^{(2w)} \leftarrow r_0'^{(2w)},\ \overline{\overline{r}}_x^{(2w)} \leftarrow r_1'^{(2w)}.$$

Після завершення циклів формування результатів, необхідно виконати цикл врахування збережених переносів в інтервалі $x=[\overline{0,k+1})$, з використанням заздалегідь проініціалізованої змінної $r^{(2w)}$ ($r^{(2w)} \leftarrow 0$):

$$r_0'^{(2w)} \leftarrow \overline{r}_x^{(2w)} + r^{(2w)}.\ q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right),\ r^{(2w)} \leftarrow \overline{\overline{r}}_x^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right).$$

Після завершення циклу врахування збережених переносів відбувається формування найстаршого значення $q_{k+1}^{(w)}$:

$$q_{k+1}^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r^{(2w)}\right).$$

Щоб врахувати можливий перенос, необхідно спочатку визначити значення індексу j (j знаходиться в інтервалі $[\overline{k,0}]$) для якого перестає виконуватися умова $q_j^{(w)} == a_j^{(w)}$. Далі, якщо $q_j^{(w)} > a_j^{(w)}$ то необхідно врахувати

перенос у великому числі $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$, інакше перенос не враховується у великому числі $r \leftarrow a \bmod \left(2^w\right)^{k+1}$.

Для обчислення залишку c, виконується віднімання одного великого числа від іншого: $c \leftarrow r - q$. У випадку, якщо отриманий залишок $c \geq p$, необхідно виконати корекцію $c \leftarrow c - p$ доки r не стане менше p. На виході отримується результат $c = a \bmod p$.

Вхід: ціле $a = d \cdot b$, $d, b \in \mathrm{GF}(p)$, $n_2 = \log_{2^w} a$, $n = \log_{2^w} d = \log_{2^w} b$, $n_3 = n + n_2$. $k = \left\lfloor \log_{2^w} p \right\rfloor + 1$, $\mu = \left\lfloor \left(2^w\right)^{2k} \Big/ p \right\rfloor$, w – довжина машинного слова. *Вихід*: $c = a \bmod p$, $c \in \mathrm{GF}(p)$.
1. #pragma omp parallel begin 1.1. #pragma omp for nowait begin 1.1.1. For $x \leftarrow 0$, $x < n_2$, $x{+}{+}$ do 1.1.1.1. $r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$. 1.1.1.2. $kn_2 \leftarrow \min(k + x; n_2)$ 1.1.1.3 For $i \leftarrow k-1$, $j \leftarrow x$, $i < kn_2$, $i{+}{+}$, $j{-}{-}$ do 1.1.1.3.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$. 1.1.1.3.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)}$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$. 1.1.1.4. $\overline{r}_x^{(2w)} \leftarrow r_0'^{(2w)}$, $\overline{\overline{r}}_x^{(2w)} \leftarrow r_1'^{(2w)}$. 2.2. #pragma omp for nowait begin 2.2.1. For $x \leftarrow n_2$, $x < n_3$, $x{+}{+}$ do 2.2.1.1. $r_0'^{(2w)} \leftarrow 0$, $r_1'^{(2w)} \leftarrow 0$. 2.2.1.2. For $i \leftarrow x - (n_2 - k)$, $j \leftarrow n_2 - 1$, $i < n_2$, $i{+}{+}$, $j{-}{-}$ do 2.2.1.2.1. $(uv)^{(2w)} \leftarrow a_i^{(w)} \cdot \mu_j^{(w)}$. 2.2.1.2.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)}$, $r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$. 2.2.1.3. $\overline{r}_x^{(2w)} \leftarrow r_0'^{(2w)}$, $\overline{\overline{r}}_x^{(2w)} \leftarrow r_1'^{(2w)}$. 3. $r^{(2w)} \leftarrow 0$. 4. For $x \leftarrow 0$, $x < n_3$, $x{+}{+}$ do 4.1. $r_0'^{(2w)} \leftarrow \overline{r}_x^{(2w)} + r^{(2w)}$.

4.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right)$,

4.3. $r^{(2w)} \leftarrow \overline{\overline{r}}_x^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$.

5. #pragma omp parallel begin

5.1. #pragma omp for nowait begin

5.1.1. For $x \leftarrow 0,\ x < n,\ x{+}{+}$ do

5.1.1.1. $r_0'^{(2w)} \leftarrow 0,\ r_1'^{(2w)} \leftarrow 0$.

5.1.1.2 For $i \leftarrow 0,\ j \leftarrow x,\ i \le x,\ i{+}{+},\ j{-}{-}$ do

5.1.1.2.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$

5.1.1.2.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$.

5.1.1.3. $\overline{r}_x^{(2w)} \leftarrow r_0'^{(2w)},\ \overline{\overline{r}}_x^{(2w)} \leftarrow r_1'^{(2w)}$.

5.2. #pragma omp for nowait begin

5.2.1. For $x \leftarrow n,\ x < k+1,\ x{+}{+}$ do

5.2.1.1. $r_0'^{(2w)} \leftarrow 0,\ r_1'^{(2w)} \leftarrow 0$.

5.2.1.2 For $i \leftarrow x-(n-1),\ j \leftarrow (n-1),\ i \le n,\ i{+}{+},\ j{-}{-}$ do

5.2.1.2.1. $(uv)^{(2w)} \leftarrow q_{k+1+i}^{(w)} \cdot p_j^{(w)}$.

5.2.1.2.2. $r_0'^{(2w)} \leftarrow r_0'^{(2w)} + v^{(w)},\ r_1'^{(2w)} \leftarrow r_1'^{(2w)} + u^{(w)}$.

5.2.1.3. $\overline{r}_x^{(2w)} \leftarrow r_0'^{(2w)},\ \overline{\overline{r}}_x^{(2w)} \leftarrow r_1'^{(2w)}$.

6. $r^{(2w)} \leftarrow 0$.

7. For $x \leftarrow 0,\ x < k+1,\ x{+}{+}$ do

7.1. $r_0'^{(2w)} \leftarrow \overline{r}_x^{(2w)} + r^{(2w)}$.

7.2. $q_x^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r_0'^{(2w)}\right)$,

7.3. $r^{(2w)} \leftarrow \overline{\overline{r}}_x^{(2w)} + \mathrm{hi}_{(w)}\left(r_0'^{(2w)}\right)$.

8. $q_{k+1}^{(w)} \leftarrow \mathrm{low}_{(w)}\left(r^{(2w)}\right)$

9. $j \leftarrow k$

10. For , $(j \ge 0)\ and\ \left(q_j^{(w)} == a_j^{(w)}\right),\ j{-}{-}$ do.

11. If $\left(q_j^{(w)} > a_j^{(w)}\right)$

11. 1. $r \leftarrow \left(2^w\right)^{k+1} + a \bmod \left(2^w\right)^{k+1}$.

12. Else

12.1. $r \leftarrow a \bmod \left(2^w\right)^{k+1}$.
13. $c \leftarrow r - q$
14. While $(c \geq p)$ do
14.1. $c \leftarrow c - p$
15. Return (c).

Рис. 3.12. Псевдокод методу 3.9

Розглянемо обчислювальну складність запропонованого методу. Обчислювальна складність основних частин методу:

- перше часткове множення (п.1 – п.2) – $\left(4+4n\right)I_{a\,\mathrm{sgn}}^{2w} + 10n \cdot I_{xor}^{w} + 2n \cdot I_{add}^{w} +$
$+\left(6n^2 - \frac{1}{2}k^2 - \frac{5nk}{2} - \frac{7}{2}n + \frac{k}{2}\right) \cdot I_{mul}^{2w} + \left(12n^2 - k^2 - 5kn - 7n + k\right)I_{add}^{2w}$
- врахування переносів після першого часткового множення (п.3 – 4) – $I_{a\,\mathrm{sgn}}^{2w} + 6n \cdot I_{add}^{2w} + 3n \cdot I_{xor}^{2w} + 3n \cdot I_{sh}^{2w}$
- друге часткове множення (п.5) – $\left(4n - 2k + 2\right)I_{a\,\mathrm{sgn}}^{2w} + \left(n\left\lfloor \frac{1+n}{2} \right\rfloor + \frac{n^2}{2} - \frac{n}{2}\right) \cdot I_{mul}^{2w} + \left(2n\left\lfloor \frac{1+n}{2} \right\rfloor + n^2 - n\right)I_{add}^{2w}$
- врахування переносів після другого часткового множення (п.6 – 8) – $\left(2k+2\right)I_{add}^{2w} + \left(k+2\right)I_{xor}^{2w} + \left(k+1\right)I_{sh}^{2w} + I_{a\,\mathrm{sgn}}^{2w}$
- подальша корекція (п.8 – п.14) – $6I_{asqn}^{w} + I_{cmp}^{w}\left(\frac{5}{2}k + \frac{1}{2}\right) + \frac{1}{2}I_{dec}^{w} + I_{sub}^{w} \cdot \frac{3}{2}k + k \cdot I_{add}^{w} + \frac{1}{2}k \cdot I_{mul}^{w}$.

Обчислювальна складність запропонованого методу приведення за модулем:

$$I_{\mathrm{mod}}\left(A_{3.9}\right) = \left(8n - 2k + 8\right)I_{a\,\mathrm{sgn}}^{2w} + \left(3n + k + 2\right)I_{xor}^{2w} + \left(3n + k + 1\right)I_{sh}^{2w} +$$
$$+\left(n\left\lfloor \frac{1+n}{2} \right\rfloor + \frac{13n^2}{2} - \frac{k^2}{2} - 4n - \frac{5nk}{2} + \frac{k}{2}\right) \cdot I_{mul}^{2w} +$$
$$+\left(2n\left\lfloor \frac{1+n}{2} \right\rfloor + 13n^2 - k^2 - 5kn - 2n + 3k + 2\right)I_{add}^{2w} +$$

$$+6I_{asqn}^{w}+I_{cmp}^{w}\left(\frac{5}{2}k+\frac{1}{2}\right)+\frac{1}{2}I_{dec}^{w}+I_{sub}^{w}\cdot\frac{3}{2}k+\left(2n+k\right)\cdot I_{add}^{w}+\frac{1}{2}k\cdot I_{mul}^{w}+10n\cdot I_{xor}^{w}$$

3.2. Висновки до третього розділу

Таким чином, у третьому розділі в результаті проведених досліджень отримані такі наукові та практичні результати:

1. Розроблено методи арифметичних операцій множення, піднесення до квадрату та приведення за модулем з відкладеним переносом, в яких для накопичення переносу використовуються блоки такої ж довжини, що і блоки з бітами числа в двійковому представленні розміром 1 машинне слово кожен, що дозволяє ефективно продемонструвати підходи до розпаралелювання методів.
2. Розроблено методи арифметичних операцій множення, піднесення до квадрату та приведення за модулем з відкладеним переносом та розпаралелюванням в два потоки за рахунок паралельного виконання першого та другого циклу множення методом Comba, що використовується в цих операціях, з подальшим коригуванням результатів.
3. Розроблено методи арифметичних операцій множення, піднесення до квадрату та приведення за модулем з відкладеним переносом та розпаралелюванням в декілька потоків за рахунок паралельного виконання ітерацій першого та другого циклу множення методом Comba, що використовується в цих операціях, з подальшим коригуванням результатів.

Список використаних джерел у третьому розділі

1. Comba P.G. Exponentiation cryptosystems on the IBM PC // IBM Systems Journal. –Vol. 29(4), 1990. – P. 526–538.
2. А. Охрименко, В. Ковтун, «Умножения целых чисел с использованием отложенного переноса для криптосистем с открытым ключом», Информационные технологи и системы в управлении, образовании,

науке: Монография, под ред. проф. В.С. Пономаренко, Харьков: Цифрова друкарня №1, 2013, С. 69-82.

3. R. Brumnik, V. Kovtun, A. Okhrimenko, S. Kavun, «Techniques for Performance Improvement of Integer Multiplication in Cryptographic Applications», Mathematical Problems in Engineering, 2014, P. 1-7.
4. В.Ю. Ковтун, А.А. Охрименко, В.В. Нечипорук, «Подходы к повышению производительности программной реализации операции умножения в поле целых чисел», Захист інформації, Т. 14, № 1 (54), С. 68-75, 2012.
5. В.Ю. Ковтун, А.А. Охрименко, «Подходы к распараллеливанию программной реализации операции умножения в поле целых чисел», Радиотехника. Всеукраинский межведомственный научно-технический сборник, № 171, С. 123-132, 2012.
6. V. Kovtun, A. Okhrimenko, «Integer multiplication algorithm with delayed carry mechanism for public key cryptosystems», Безпека інформації, Vol. 19, №1, pp. 45-50, 2013.
7. А.А. Охрименко, «Эффективная программная реализация алгоритмов умножения целых чисел для современных платформ», Вісник Інженерної академії України, № 2, С. 108-113, 2013.
8. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, С.Ю. Ковтун, С.П. Євсеєв, О.Г. Король, «Спосіб множення цілих чисел», Пат. 111632 Україна, МПК G06F 7/253 (2006.01). Заявка № u 2015 11473; заявл. 23.11.2015; опублік. 25.11.2016, Бюл. № 22.
9. А.О. Охріменко, «Підвищення швидкодії арифметичних операцій в полі цілих чисел», Захист інформації з обмеженим доступом та автоматизація її обробки: IV наук.-техн. конф. студ. та асп., 9-10 лютого 2012 р., К., 2012, С. 22-23.
10. A. Okhrimenko, V. Kovtun, «Parallelization of software implementation of integer multiplication», AVIATION IN THE XXI-st CENTURY – Safety in Aviation and Space Technologies: V World Congress, September 25-27, 2012, K., 2012, P. 1.7.44–1.7.49.

11. А.А. Охрименко, В.Ю. Ковтун, «Подходы к повышению быстродействия криптосистем с открытым ключом», Проблеми і перспективи розвитку ІТ-індустрії: V міжнар. наук.-практ. конф., 25-26 квітня 2013 р., Харків, 2013, С. 202.
12. А.А. Охрименко, В.Ю. Ковтун, «Алгоритм умножения целых чисел с использова-нием технологий распараллеливания», Питання оптимізації обчислень (ПОО-XL): праці міжн. наук. конф., 30 вересня – 4 жовтня 2013 р., К., 2013, С.125-126.
13. O.G. Korchenko, V.Yu. Kovtun, A.O. Okhrimenko, «Parallelization of Integer Squaring Algorithms with Delayed Carry», Journal of Computer Networks, 2014, Vol. 2(2), pp 10-17
14. В.Ю. Ковтун, А.А. Охрименко, «Алгоритм возведения в квадрат целых чисел с использованием отложенного переноса», Безпека інформації, Т. 19, №3, С. 188-192, 2013.
15. А. Охрименко, «Обобщенные алгоритмы возведения в квадрат целых чисел с использованием отложенного переноса и технологий распараллеливания», Вісник Інженерної академії України, № 1, С. 114-119, 2014.
16. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, С.П. Євсеєв, О.Г. Король, Р.В. Грищук, Г.П. Коц, «Спосіб піднесення до квадрата цілих чисел», Пат. 118065 Україна, МПК G06F 7/523 (Пат. 118066 Україна, МПК G06F 7/523 (2006.01)2006.01). Заявка № u 2016 13439; заявл. 27.12.2016; опублік. 25.07.2017, Бюл. № 14.
17. А.А. Охрименко, «Применение механизмов отложеного переноса и распараллеливания для повышения производительности операции возведения в квадрат целых чисел», Безопасность информации в информационно-телекоммуникационных системах: XVI Междун. науч.-практ. конф., 21–24 мая 2013 г., К., 2013, С. 28-29.
18. A. Okhrimenko, «Squaring algorithms for public-key cryptosystems», Інфокомунікації – сучасність та майбутнє: матеріали III міжнар. наук.-пр. конф. мол. вчених 17-18 жовтня 2013 р., Одеса, 2013, С. 178-182.

19. А. Охрименко, В. Ковтун, «Метод повышения производительности операции приведения по простому модулю», Информационные системы в управлении, образовании, промышленности: Монография, под ред. В.С. Пономаренко, Харьков: Вид-во ТОВ «Щедра садиба плюс», 2014, С. 204-219.
20. А.О. Охріменко, В.Ю. Ковтун, М.Г. Ковтун, «Спосіб приведення за модулем цілих чисел», Пат. 118066 Україна, МПК G06F 7/523 (2006.01). Заявка № u 2016 13441; заявл. 27.12.2016; опублік. 25.07.2017, Бюл. № 14.
21. А.А. Охрименко, «Подходы к повышению производительности алгоритмов возведения в квадрат целых чисел», Інтегровані інтелектуальні робототехнічні комплекси: VI Міжнар. наук.-практ. конф., 27–29 травня 2013 р., К., 2013, С. 292-293.
22. А.А. Охрименко, В.Ю. Ковтун, «Классификация методов повышения производительности операции приведения по большому простому модулю», Проблеми і перспективи розвитку ІТ-індустрії: VI міжнар. наук.-практ. конф., 17-18 квітня 2014 р.: тези доп., Харків, 2014, С. 253.
23. А.А. Охрименко, «Модифицированный алгоритм Баррета приведения целых чисел по модулю», Інформаційні технології та комп’ютерна інженерія: IV міжнар. наук.-практ. конф., 28-30 травня 2014 р., Вінниця, 2014, С. 180-181.
24. P. Barrett. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. Proceedings CRYPTO'86, pp. 311-323.
25. Bhattacharyya SS, Deprettere EF, Leupers R, Takala J, editors. Handbook of Signal Processing Systems. 2nd ed. Springer, 2013. 1399 pp.

Розділ 4

ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБЛЕНИХ МЕТОДІВ

В даному розділі представлено результати експериментального дослідження розроблених методів представлення цілих чисел та арифметичних перетворень над ними в полях та кільцях цілих чисел, еліптичних кривих над полем цілих чисел, та в криптографічних перетвореннях схем електронного підпису, що застосовуються в національній інфраструктурі відкритих ключів України.

4.1. Методика проведення експерименту

1. Мета та задачі експерименту

Метою експерименту є дослідження ефективності запропонованих методів за допомогою розробленого програмного забезпечення.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- Дослідити запропонований метод представлення цілих чисел **за рахунок оцінки запропонованих методів арифметичних перетворень в реальних застосуваннях.**
- Дослідити запропоновані методи арифметичних перетворень над цілими числами.
- Дослідити запропоновані методи арифметичних перетворень в кільці цілих чисел.
- Дослідити запропоновані методи арифметичних перетворень в полі цілих чисел GF(p).
- Дослідити запропоновані методи арифметичних перетворень на еліптичних кривих над полем цілих чисел GF(p).
- Дослідити швидкодію криптографічних перетворень за схемою ECDSA з використанням запропонованих методів арифметичних перетворень над полем цілих чисел GF(p).

- Дослідити швидкодію криптографічних перетворень за схемою ДСТУ 4145-2002 з використанням запропонованих методів арифметичних перетворень над полем поліномів GF(2^m) та цілих чисел GF(p).
- Дослідити швидкодію криптографічних перетворень за схемою RSA з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел.

2. Вибір вхідних та вихідних параметрів

Вхідними параметрами для вирішення поставлених задач є згенеровані, відповідним чином, цілі числа. Вихідними параметрами є зібрана статистика швидкостей роботи запропонованих та класичних методів арифметичних перетворень.

3. Послідовність дій

Почергове виконання досліджень з заданими умовами та параметрами для різних платформ та типів процесорів запропонованих та класичних методів арифметичних перетворень.

Зібрана статистика використовується для аналізу ефективності запропонованих методів арифметичних перетворень.

4. Вибір кроку зміни чинників

Для експериментальних досліджень запропонованих методів арифметичних перетворень над цілими числами використовуються цілі числа розміром 128, 256, 512, 1024, 2048, 3072, 4096, 6144, 8192, 12288 та 16384 біт.

Для експериментальних досліджень запропонованих методів арифметичних перетворень в кільці цілих чисел використовуються кільця з модулями 512, 1024, 3072, 4096, 8192 та 16384 біт.

Для експериментальних досліджень запропонованих методів арифметичних перетворень в полі цілих чисел GF(p) використовуються псевдомерсенові та загальні прості модулі p192, p224, p256, p384 та p521.

Для експериментальних досліджень запропонованих методів арифметичних перетворень на еліптичних кривих над полем цілих чисел GF(p)

використовуються прості поля з NIST FIPS 186-3 – P-192, P-224, P-256, P-384 та P-521.

Для експериментальних досліджень запропонованих методів арифметичних перетворень в криптографічних перетвореннях за схемою ECDSA використовуються прості поля з NIST FIPS 186-3, а саме P-192, P-224, P-256, P-384 та P-521.

Для експериментальних досліджень запропонованих методів арифметичних перетворень в криптографічних перетвореннях за схемою ДСТУ 4145 використовуються двійкові поля з ДСТУ 4145-2002, а саме $GF(2^{163})$, $GF(2^{167})$, $GF(2^{173})$, $GF(2^{179})$, $GF(2^{191})$, $GF(2^{233})$, $GF(2^{257})$, $GF(2^{307})$, $GF(2^{367})$ та $GF(2^{431})$.

Для експериментальних досліджень запропонованих методів арифметичних перетворень в криптографічних перетвореннях за схемою RSA з використанням запропонованих методів арифметичних перетворень використовуються ключі розміром 512, 1024, 1536, 2048, 3072, 4096, 7168, 8192, 15360 та 16384 біт з публічними експонентами 3, 5, 17, 257, 65537 та 0 (число генерується в межах від 0 до 2^{32}).

5. Засоби проведення експерименту

Для оцінки ефективності запропонованих методів, була виконана програмна реалізація за допомогою Microsoft Visual Studio 2015 та GCC 9.2 [1] мовою програмування C++ та підтримкою OpenMP [2] без використання асемблерних вставок для наступних платформ:

- Server. Процесор Intel Xeon E2146G 3,50 GHz під управлінням ОС Microsoft Windows Server 2019, x64.
- Desktop. Процесор Intel Core-i7 4770 3,40 GHz під управлінням ОС Microsoft Windows 10, x64.
- Mobile. Процесор Intel Core-i7 6700HQ 2,60 GHz під управлінням ОС Microsoft Windows 10, x64.
- Embedded. Процесор Broadcom BC2711 1.50 GHz (ARMv8, Cortex-A72) під управлінням ОС Canonical Ubuntu 19.10, aarch64.

6. Аналіз результатів

Відповідно до обраної методики проведено експериментальні дослідження, опис та обробка яких міститься в п.4.2-4.8.

4.2. Експериментальні дослідження методів арифметичних перетворень над цілими числами

Для перевірки запропонованих методів арифметичних перетворень над цілими числами були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані класичні методи арифметичних операції. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації перетворень, що вимірювалися, в програмній реалізації:

- класичних методів арифметичних перетворень;
- запропонованих методів арифметичних перетворень без розпаралелювання;
- запропонованих методів арифметичних перетворень з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках;
- запропонованих методів арифметичних перетворень з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків (кількість потоків вибирається оптимальним чином бібліотекою OpenMP в залежності від кількості ядер процесора – для платформи Server було використано 12 потоків, для платформи Desktop та Mobile – 8 потоків, а для платформи Embedded – 4 потоки).

Експериментальні дослідження проводились саме для значущих арифметичних перетворень, з точки зору їх впливу:

- множення;
- піднесення до квадрату;
- приведення за модулем.

Значення швидкодії перетворень множення цілих чисел показані в мікросекундах. M – класичний метод множення, M1 – запропонований метод множення без розпаралелювання, M2 – запропонований метод множення з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках, M3 – запропонований метод множення з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків.

Значення швидкодії перетворень піднесення до квадрату цілих чисел показано в мікросекундах. S – класичний метод піднесення до квадрату, S1 – запропонований метод піднесення до квадрату без розпаралелювання, S2 – запропонований метод піднесення до квадрату з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках, S3 – запропонований метод піднесення до квадрату з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків.

Значення швидкодії перетворень приведення за модулем цілих чисел показані в мікросекундах. R – класичний метод приведення за модулем, R1 – запропонований метод приведення за модулем без розпаралелювання, R2 – запропонований метод приведення за модулем з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках, R3 – запропонований метод приведення за модулем з відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків.

Есперименталні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку Б, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії арифметичних перетворень (множення, піднесення до квадрату та приведення за модулем) над цілими числами можна зробити висновок, що запропоновані методи ефективніші, ніж

класичні. В Табл. 4.1 показані зведені дані нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень.

У Табл. 4.1 позначено: 1 – відношення результатів класичних методів до результатів запропонованих методів; 2 – відношення результатів класичних методів до результатів запропонованих методів з застосуванням розпаралелювання у 2 потоки; 3 – відношення результатів класичних методів до результатів запропонованих методів з застосуванням розпаралелювання у декілька потоків.

Таблиця 4.1

Зведена таблиця нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень

Операція		Платформа						
		Server		Desktop		Mobile		Embedded
		w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
M	1	1,05-1,54	1,04-1,30	1,09-1,63	1,16-2,28	1,07-1,70	1,08-2,22	1,02-2,00
	2	1,94-3,00	1,19-2,10	1,01-3,24	1,34-2,29	1,95-2,84	1,07-1,89	1,39-2,15
	3	2,01-8,29	1,20-5,10	2,74-6,08	1,05-4,09	1,34-5,16	1,40-3,19	1,12-3,79
S	1	1,25-3,19	1,02-1,34	1,33-2,75	1,01-2,79	1,33-3,16	1,08-2,00	1,03-4,12
	2	1,69-5,75	1,45-2,05	1,01-4,76	1,18-1,93	1,82-5,15	1,29-1,99	1,34-2,46
	3	1,52-17,3	1,09-2,60	2,04-8,72	1,11-3,31	1,31-9,85	1,13-3,02	1,53-4,60
R	1	1,02-1,18	1,01-1,57	1,06-1,47	1,12-2,54	1,06-1,80	1,04-1,43	1,03-4,12
	2	1,48-1,90	1,04-1,76	1,26-1,66	1,29-1,64	1,22-2,10	1,30-1,64	1,34-2,46
	3	1,22-5,14	1,31-3,87	1,31-3,62	1,74-2,96	1,61-3,38	1,88-2,67	1,53-4,60

Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної платформи, розрядності машинних слів та має локальний екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність. Загалом, відповідно до Табл. 4.1, запропоновані методи арифметичних перетворень ефективніші:

- множення цілих чисел – в 1,05-1,7 разів для *w*=32 біт, та 1,02-2,28 разів для *w*=64 біт;
- множення цілих чисел з розпаралелюванням в 2 потоки – в 1,01-3,24 разів для *w*=32 біт, та 1,07-2,29 разів для *w*=64 біт;

- множення цілих чисел з розпаралелюванням в декілька потоків – в 1,34-8,29 разів для w=32 біт, та 1,05-5,1 разів для w=64 біт;
- піднесення до квадрату цілих чисел – в 1,25-3,19 разів для w=32 біт, та 1,01-4,12 разів для w=64 біт;
- піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки – в 1,01-5,75 разів для w=32 біт, та 1,18-2,46 разів для w=64 біт;
- піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків – в 1,31-17,3 разів для w=32 біт, та 1,09-4,6 разів для w=64 біт;
- приведення за модулем цілих чисел – в 1,02-1,8 разів для w=32 біт, та 1,01-4,12 разів для w=64 біт;
- приведення за модулем цілих чисел з розпаралелюванням в 2 потоки – в 1,22-2,10 разів для w=32 біт, та 1,04-2,46 разів для w=64 біт;
- приведення за модулем цілих чисел з розпаралелюванням в декілька потоків – в 1,22-5,14 разів для w=32 біт, та 1,31-4,6 разів для w=64 біт;

Запропоновані методи арифметичних перетворень з розпаралелюванням дозволяють суттєво підвищити швидкодію для різних апаратних платформ, про що свідчать результати Табл. 4.2. Розпаралелювання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин, що перевищують 2048 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт. Ефективність розпаралелювання залежить від загальної архітектури процесорів та внутрішньої схеми між’ядерної взаємодії.

У Табл. 4.2 позначено: 1 – відношення результатів запропонованого методу до результатів запропонованого методу з розпаралелюванням у 2 потоки; 2 – відношення результатів запропонованого методу до результатів запропонованого методу з розпаралелюванням у декілька потоки.

Таблиця 4.2

Зведена таблиця нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень з використанням розпаралелювання

Операція		Платформа						
		Server		Desktop		Mobile		Embedded
		w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
M	1	1,3-2,35	1,53-2,03	1,06-2,71	1,32-1,96	1,17-2,33	1,02-1,74	1,33-1,96
	2	1,72-6,81	1,11-4,91	1,89-5,08	1,17-3,51	1,82-4,39	1,18-2,95	1,10-3,54
S	1	1,18-1,82	1,38-2,00	1,03-1,74	1,17-1,91	1,25-1,74	1,16-1,79	1,27-2,33
	2	1,09-5,43	1,04-2,43	1,58-3,18	1,10-3,28	1,20-3,18	1,01-2,72	1,49-4,36
R	1	1,02-1,62	1,05-1,63	1,10-1,50	1,12-1,43	1,11-1,64	1,11-1,39	1,15-1,50
	2	1,05-4,50	1,20-3,32	1,20-3,39	1,48-2,62	1,83-2,96	1,59-2,28	1,19-2,65

Загалом, відповідно до Табл. 4.2, запропоновані методи арифметичних перетворень з розпаралелюванням ефективніші запропонованих методів без розпаралелювання:

- множення цілих чисел з розпаралелюванням в 2 потоки – в 1,17-2,71 разів для *w*=32 біт, та 1,02-2,03 разів для *w*=64 біт;
- множення цілих чисел з розпаралелюванням в декілька потоків – в 1,72-6,81 разів для *w*=32 біт, та 1,10-4,91 разів для *w*=64 біт;
- піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки – в 1,03-1,82 разів для *w*=32 біт, та 1,16-2,33 разів для *w*=64 біт;
- піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків – в 1,09-5,43 разів для *w*=32 біт, та 1,01-4,36 разів для *w*=64 біт;
- приведення за модулем цілих чисел з розпаралелюванням в 2 потоки – в 1,02-1,64 разів для *w*=32 біт, та 1,05-1,63 разів для *w*=64 біт;
- приведення за модулем цілих чисел з розпаралелюванням в декілька потоків – в 1,05-4,5 разів для *w*=32 біт, та 1,19-3,32 разів для *w*=64 біт;

Крім того, слід зазначити, що сучасні процесори будуються за 64-х розрядною архітектурою, що показує значно більшу ефективність 64-х бітних реалізацій як класичних, так і запропонованих методів.

Далі буде розглядатись ефективність запропонованих методів арифметичних перетворень в кільці цілих чисел.

4.3. Експериментальні дослідження методів арифметичних перетворень в кільці цілих чисел

Для перевірки ефективності запропонованих методів арифметичних перетворень в кільці цілих чисел були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані класичні методи арифметичних перетворень. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації арифметичних перетворень в програмній реалізації:

- з використанням класичних методів арифметичних перетворень (original);
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання (with improvements);
- з використанням запропонованих методів арифметичних перетворень та розпаралелюванням в 2 потоки (with improvements and 2-thread implementation);
- з використанням запропонованих методів арифметичних перетворень та розпаралелюванням в декілька потоків (with improvements and multithread implementation). Кількість потоків вибирається оптимальним чином бібліотекою OpenMP в залежності від кількості ядер процесора – для платформи Server було використано 12 потоків, для платформи Desktop та Mobile – 8 потоків, а для платформи Embedded – 4 потоки).

Експериментальні дослідження проводились для деяких значущих методів арифметичних перетворень в кільці цілих чисел:

- додавання за модулем (Add);
- віднімання за модулем (Sub);
- множення за модулем (Mul);
- піднесення до квадрату за модулем (Sqr);
- піднесення до степені за модулем (Pow).

Значення швидкодії вказаних методів в кільці цілих чисел зазначені в мікросекундах.

Есперементальні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку В, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії арифметичних перетворень в кільці цілих чисел можна зробити висновоак, що запропоновані методи ефективніші, ніж класичні. В Табл. 4.3 показані зведені дані нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень.

Таблиця 4.3

Зведена таблиця нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень у кільці цілих чисел

Операція	Платформа						
	Server		Desktop		Mobile		Embedded
	w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
Add	1,03-1,23	1,05-1,38	1,01-1,33	1,03-1,29	1,03-1,97	1,03-2,42	1,06-3,28
Sub	1,04-1,22	1,03-1,81	1,04-1,07	1,09-1,67	1,01-1,34	1,03-2,41	1,08-1,58
Mul	1,08-1,12	1,02-1,52	1,05-1,15	1,03-1,71	1,04-1,18	1,02-1,13	1,02-1,08
Sqr	1,02-1,46	1,02-1,52	1,09-1,37	1,02-1,73	1,05-1,45	1,03-1,17	1,02-1,05
Pow	1,57-1,75	1,02-1,56	1,42-174	1,03-1,76	1,38-1,70	1,01-1,17	1,02-1,14

Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної архітектури, розрядності машинних слів та має локальний екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність. Загалом, відповідно до Табл. 4.3, запропоновані методи арифметичних перетворень у кільці цілих чисел ефективніші:

- операція додавання за модулем – в 1,01-1,97 разів для *w*=32 біт, та 1,03-3,28 разів для *w*=64 біт;

- операція віднімання за модулем – в 1,01-1,34 разів для w=32 біт, та 1,03-2,41 разів для w=64 біт;
- операція множення за модулем – в 1,01-1,34 разів для w=32 біт, та 1,03-2,41 разів для w=64 біт;
- операція піднесення до квадрату за модулем – в 1,02-1,46 разів для w=32 біт, та 1,02-1,73 разів для w=64 біт;
- операція піднесення до степеню за модулем – в 1,38-1,75 разів для w=32 біт, та 1,01-1,76 разів для w=64 біт;

Запропоновані методи арифметичних перетворень з розпаралелюванням дозволяють підвищити швидкодію арифметичних перетворень для різних апаратних платформ за рахунок ефективного розпаралелювання, про що свідчать результати Табл. 4.4.

Таблиця 4.4

Нормалізовані результати вимірювання швидкодії арифметичних перетворень у кільці цілих чисел з використанням розпаралелювання

Операція		Платформа						
		Server		Desktop		Mobile		Embedded
		w=32 біт	w=64 біт	w=32 біт	w=64 біт	w=32 біт	w=64 біт	w=64 біт
Add	1	1,01-1,74	1,02-1,52	1,01-2,07	1,01-1,73	1,01-1,54	1,01-1,23	1,01-1,74
	2	1,02-1,97	1,01-1,48	1,01-1,05	1,03-1,73	1,08-1,74	1,02-1,10	1,02-1,97
Sub	1	1,03-1,23	1,01-1,02	1,01-1,07	1,01-1,20	1,02-1,15	1,01-1,13	1,03-1,23
	2	1,02-1,23	1,01-1,44	1,02-1,23	1,03-1,42	1,02-1,44	1,02-1,14	1,02-1,23
Mul	1	1,04-1,15	1,09-1,15	1,09-1,24	1,06-1,21	1,02-1,11	1,08-1,10	1,02-1,10
	2	1,02-1,07	1,02-1,04	1,02-1,11	1,05-1,07	1,15-1,20	1,11-1,14	1,02-1,07
Sqr	1	1,06-1,15	1,01-1,10	1,03-1,10	1,03-1,12	1,03-1,07	1,02-1,05	1,06-1,15
	2	1,01-1,03	1,01	1,01-1,05	1,01-1,05	1,08-1,12	1,06-1,09	1,01-1,03
Pow	1	1,16-1,92	1,42-2,01	1,24-2,50	1,31-1,84	1,52-1,89	1,37-1,78	1,16-1,92
	2	1,23-4,60	1,44-4,69	1,11-4,62	1,52-3,44	1,44-3,38	1,25-3,06	1,23-4,60

У Табл. 4.4 позначено: 1 – відношення результатів з застосуванням запропонованого методу до результатів з застосуванням запропонованого методу з розпаралелювання у 2 потоки; 2 – відношення результатів з застосуванням запропонованого методу до результатів з застосуванням запропонованого методу з розпаралелювання у декілька потоків.

Розпаралелювання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин що перевищують 2048 біт, а

розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт (виключення складає піднесення до степеню, коли має сенс вже при 2048 біт).

Загалом, відповідно до Табл. 4.4, запропоновані методи арифметичних перетворень в кільці цілих чисел з розпаралелюванням ефективніші запропонованих методів без розпаралелювання:

- операція додавання за модулем з розпаралелюванням в 2 потоки – в 1,01-2,07 разів для w=32 біт, та 1,01-1,74 разів для w=64 біт;
- операція додавання за модулем з розпаралелюванням в декілька потоків – в 1,01-1,97 разів для w=32 біт, та 1,01-1,97 разів для w=64 біт;
- операція віднімання за модулем з розпаралелюванням в 2 потоки – в 1,01-1,23 разів для w=32 біт, та 1,03-1,23 разів для w=64 біт;
- операція віднімання за модулем з розпаралелюванням в декілька потоків – в 1,02-1,44 разів для w=32 біт, та 1,01-1,44 разів для w=64 біт;
- операція множення за модулем з розпаралелюванням в 2 потоки – в 1,02-1,24 разів для w=32 біт, та 1,02-1,21 разів для w=64 біт;
- операція множення за модулем з розпаралелюванням в декілька потоків – в 1,02-1,2 разів для w=32 біт, та 1,02-1,14 разів для w=64 біт;
- операція піднесення до квадрату за модулем з розпаралелюванням в 2 потоки – в 1,03-1,15 разів для w=32 біт, та 1,01-1,15 разів для w=64 біт;
- операція піднесення до квадрату за модулем з розпаралелюванням в декілька потоків – в 1,01-1,12 разів для w=32 біт, та 1,01-1,09 разів для w=64 біт;
- операція піднесення до степеню за модулем з розпаралелюванням в 2 потоки – в 1,16-2,5 разів для w=32 біт, та 1,16-2,01 разів для w=64 біт;
- операція піднесення до степеню за модулем з розпаралелюванням в декілька потоків – в 1,11-4,62 разів для w=32 біт, та 1,23-4,69 разів для w=64 біт;

Крім того, слід зазначити, що сучасні процесори будуються за 64-х розрядною архітектурою, що показує значно більшу ефективність 64-х бітних реалізацій як класичних, так і запропонованих методів.

Далі будуть розглядатися арифметичні операції в полі простих чисел GF(p).

4.4. Експериментальні дослідження методів арифметичних перетворень в полі простих чисел GF(p)

Для перевірки запропонованих методів арифметичних перетворень в полі цілих чисел GF(p) були підготовлені дві програмні збірки – з використанням 32-бітних та 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані класичні методи арифметичних перетворень. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації арифметичних перетворень в програмній реалізації:

- з використанням класичних методів арифметичних перетворень (original);
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання (with improvements).

Використання розпаралелювання в 2 та декілька потоків не розглядається, оскільки ефект досягається при значно більших значеннях розміру полів, ніж ті, що використовуються в криптосистемах ECDSA, ДСТУ 4145-2002 та інших.

Відомо [3, 4], що операції у полі можуть бути оптимізовані за рахунок використання простих чисел спеціального вигляду, наприклад псевдомерсенові, які дозволяють суттєво підвищити швидкодію операції приведення за модулем. Далі будуть розглядатися результати для простих чисел загального вигляду та псевдомерсена.

Експериментальні дослідження проводились для деяких значущих арифметичних перетворень в кільці полі простих чисел GF(p):

- додавання за модулем (Add);
- віднімання за модулем (Sub);
- множення за модулем (Mul);
- піднесення до квадрату за модулем (Sqr);
- піднесення до степені за модулем (Pow).

Значення швидкодії вказаних перетворень в кільці цілих чисел зазначені в мікросекундах.

Есперимантальні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку Г, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії арифметичних перетворень у простому полі цілих чисел можна зробити висновок, що запропоновані методи ефективніші, ніж класичні. В Табл. 4.5 показані зведені дані нормалізованих результатів вимірювання швидкодії запропонованих методів арифметичних перетворень.

Таблиця 4.5

Нормалізовані результати вимірювання швидкодії арифметичних перетворень у простому полі цілих чисел

Операція		Платформа						
		Server		Desktop		Mobile		Embedded
		w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
Add	1	1,05-1,09	1,04-1,07	1,02-1,04	1,01-1,04	1,02-1,05	1,02-1,04	1,02-1,05
	2	1,04-1,08	1,02-1,04	1,01-1,03	1,02-1,03	1,02-1,04	1,02-1,03	1,01-1,04
Sub	1	1,04-1,08	1,04-1,05	1,01-1,04	1,02-1,05	1,01-1,04	1,02-1,04	1,02-1,05
	2	1,03-1,07	1,02-1,03	1,01-1,02	1,02-1,04	1,01-1,03	1,02-1,04	1,02-1,04
Mul	1	1,03-1,09	1,03-1,06	1,03-1,07	1,03-1,07	1,02-1,07	1,04-1,07	1,01-1,06
	2	1,02-1,04	1,04-1,08	1,03-1,06	1,03-1,06	1,02-1,07	1,01-1,06	1,01-1,04
Sqr	1	1,04-1,08	1,03-1,07	1,02-1,05	1,04-1,06	1,02-1,06	1,04-1,07	1,01-1,07
	2	1,02-1,07	1,03-1,08	1,03-1,05	1,03-1,05	1,03-1,06	1,02-1,06	1,01-1,05
Pow	1	1,03-1,06	1,03-1,05	1,03-1,12	1,03-1,14	1,03-1,11	1,03-1,13	1,02-1,10
	2	1,04-1,09	1,03-1,13	1,03-1,08	1,04-1,11	1,04-1,09	1,02-1,11	1,02-1,08

У Табл. 4.5 позначено: 1 – відношення результатів з застосуванням простих чисел загального вигляду; 2 – відношення результатів з застосуванням простих чисел спеціального (псевдомерсенового) вигляду.

Загалом, відповідно до Табл. 4.5, запропоновані методи арифметичних перетворень в простому полі цілих чисел ефективніші:

- операція додавання (модуль загального вигляду) – в 1,02-1,09 разів для *w*=32 біт, та 1,01-1,07 разів для *w*=64 біт;
- операція додавання (модуль спеціального вигляду) – в 1,01-1,08 разів для *w*=32 біт, та 1,01-1,04 разів для *w*=64 біт;

- операція віднімання (модуль загального вигляду) – в 1,01-1,08 разів для w=32 біт, та 1,02-1,05 разів для w=64 біт;
- операція віднімання (модуль спеціального вигляду) – в 1,01-1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт;
- операція множення (модуль загального вигляду) – в 1,02-1,09 разів для w=32 біт, та 1,01-1,07 разів для w=64 біт;
- операція множення (модуль спеціального вигляду) – в 1,02-1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт;
- операція піднесення до квадрату (модуль загального вигляду) – в 1,02-1,08 разів для w=32 біт, та 1,01-1,07 разів для w=64 біт;
- операція піднесення до квадрату (модуль спеціального вигляду) – в 1,02-1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт;
- операція піднесення до степеню (модуль загального вигляду) – в 1,03-1,12 разів для w=32 біт, та 1,02-1,14 разів для w=64 біт;
- операція піднесення до степеню (модуль спеціального вигляду) – в 1,03-1,09 разів для w=32 біт, та 1,02-1,13 разів для w=64 біт;

Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, на ефективність запропонованих методів практично не впливає їх двійкова довжина. Розпаралелювання перетворень у полі не виконувалося, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемах ECDSA, ДСТУ 4145-2002 та інших.

Крім того, слід зазначити, що сучасні процесори будуються за 64-х розрядною архітектурою, що показує значно більшу ефективність 64-х бітних реалізацій як класичних, так і запропонованих методів.

Далі будуть розглядатися арифметичні операції в групі точок ЕК над простим полем GF(p), які використовуються в криптографічних системах.

4.5. Експериментальні дослідження методів арифметичних перетворень в групі точок еліптичних кривих над простим полем GF(p)

Для перевірки запропонованих методів арифметичних перетворень в групі точок ЕК над простим полем GF(p) були підготовлені дві програмні збірки – з використанням 32-бітних та 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані класичні методи арифметичні операції. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації перетворень в програмній реалізації:

- з використанням класичних методів арифметичних перетворень (original);
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання (with improvements).

Використання розпаралелювання в 2 та декілька потоків не розглядається, оскільки ефект від їх використання досягається при більших значеннях розміру полів, ніж ті, що використовуються в криптосистемах ECDSA, ДСТУ 4145-2002 та інших.

Відомо [3, 4], що операції у полі можуть бути оптимізовані за рахунок використання простих чисел спеціального вигляду, наприклад псевдомерсенові, які дозволяють суттєво підвищити швидкодію операції приведення за модулем. Далі будуть розглядатися результати для простих чисел псевдомерсена, як рекомендовані у якості базових за NIST FIPS 186-3. Прості числа загального вигляду, які рекомендовані RFC 5639 у якості модулей базового поля – не розглядалися, бо не несуть принципово нову інформацію.

За результатами досліджень [5], координати Чудновського є найбільш ефективними для представлення точок ЕК, для перетворень над ними, тому будуть розглядатися операції над точками ЕК, як у афінних, проективних координатах Чудновського і змішаних.

Експериментальні дослідження проводились для деяких значущих перетворень в групі точок ЕК над простим полем GF(p):

- додавання точок ЕК над простим полем GF(p) в афінних координатах (add);
- подвоєння точок ЕК над простим полем GF(p) в афінних координатах (dbl);
- додавання точок ЕК над простим полем GF(p) в змішаних (використовуються проективні координати Чудновського) координатах (addmx Ch);
- подвоєння точок ЕК над простим полем GF(p) в проективних (проективні координати Чудновського) координатах (addmx Ch);
- скалярне множення випадкової точки ЕК над простим полем GF(p) з використанням проективних координат Чудновського для подвоєння та змішаного додавання точок на основі бінарного алгоритму зліва направо (mul (rand));
- скалярне множення фіксованої точки ЕК над простим полем GF(p) з використанням проективних координат Чудновського для подвоєння та змішаного додавання точок на основі алгоритму Лім-Лі з попередніми обчисленнями (mul (fixed)).

Значення швидкодії вказаних перетворень в группі точок ЕК над простим полем GF(p) зазначені в мікросекундах.

Есперементальні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку Ґ, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії перетворень у групі точок ЕК над простим полем (додавання, подвоєння та скалярне множення) можна зробити висновок, що запропоновані методи ефективніші, ніж класичні. В Табл. 4.6 показані зведені дані нормалізованих результатів вимірювання швидкодії перетворень з використанням запропонованих методів арифметичних перетворень.

Таблиця 4.6

Зведена таблиця нормалізованих результатів вимірювання швидкодії перетворень у групі точок ЕК над простим полем

Операція	Платформа						
	Server		Desktop		Mobile		Embedded
	w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
add	1,03-1,08	1,02-1,09	1,02-1,04	1,02-1,05	1,02-1,03	1,02-1,04	1,01-1,06
dbl	1,03-1,06	1,02-1,08	1,02-1,04	1,02-1,06	1,01-1,03	1,02-1,04	1,02-1,05
addmx Ch	1,03-1,06	1,03-1,07	1,03-1,04	1,02-1,04	1,03-1,04	1,02-1,04	-
dbl Ch	1,03-1,08	1,04-1,08	1,02-1,05	1,03-1,05	1,03-1,04	1,03-1,05	-
mul (rand)	1,06-1,12	1,05-1,09	1,02-1,07	1,02-1,06	1,02-1,05	1,03-1,07	1,02-1,06
mul (fixed)	1,06-1,12	1,06-1,10	1,03-1,07	1,03-1,07	1,02-1,07	1,03-1,07	1,02-1,08

Загалом, відповідно до Табл. 4.6, запропоновані методи арифметичних перетворень у групі точок ЕК над простим полем ефективніші:

- додавання точок ЕК – в 1,02-1,08 разів для *w*=32 біт, та 1,01-1,09 разів для *w*=64 біт;
- подвоєння точок ЕК – в 1,01-1,06 разів для *w*=32 біт, та 1,02-1,08 разів для *w*=64 біт;
- додавання точок ЕК в змішаних координатах – в 1,03-1,06 разів для *w*=32 біт, та 1,02-1,07 разів для *w*=64 біт;
- подвоєння точок ЕК в проективних координатах – в 1,02-1,08 разів для *w*=32 біт, та 1,03-1,08 разів для *w*=64 біт;
- скалярне множення випадкової точки ЕК з використанням проективних координат – в 1,02-1,12 разів для *w*=32 біт, та 1,02-1,09 разів для *w*=64 біт;
- скалярне множення фіксованої точки ЕК з використанням проективних координат – в 1,02-1,12 разів для *w*=32 біт, та 1,02-1,10 разів для *w*=64 біт.

Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, їх двійкова довжина практично не впливає на ефективність запропонованих методів. Розпаралелювання перетворень у базовому полі не застосовувалось, оскільки ефект від розпаралелювання досягається при значно

більших значеннях розміру полів, ніж ті, що використовуються у криптосистемах ECDSA, ДСТУ 4145-2002 та інших.

Крім того, слід зазначити, що сучасні процесори будуються за 64-х розрядною архітектурою, що показує значно більшу ефективність 64-х бітних реалізацій як класичних, так і запропонованих методів.

Далі будуть розглядатися результати криптографічної системи ECDSA, яка базується на операціях в групі точок ЕК над простим полем GF(p).

4.6. Експериментальні дослідження методів арифметичних перетворень в криптосистемі ECDSA над простим полем GF(p)

Міжнародний алгоритм електронного підпису ECDSA дозволений [6] до використання і активно використовується в національній інфраструктурі відкритих ключів України саме над простим полем GF(p).

Для перевірки запропонованих методів перетворень криптосистеми ECDSA над простим полем GF(p) були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані операції без застосування запропонованих методів. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації перетворень в програмній реалізації:

- з використанням класичних методів арифметичних перетворень (original);
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання (with improvements).

Використання розпаралелювання в 2 та декілька потоків не розглядається, оскільки ефект від їх використання досягається при більших значеннях розміру полів, ніж ті, що використовуються в криптосистемі ECDSA, ДСТУ 4145-2002 та інших.

Експериментальні дослідження проводились для основних перетворень криптосистеми ECDSA над простим полем GF(p):

- генерування особистого ключа (private key generation) – операції у простому полі цілих чисел, за модулем порядку групи – простого числа загального вигляду;
- генерування відкритого ключа (public key generation) – скалярне множення за фіксованою точкою ЕК (генератором групи);
- створення підпису (digest sign) – скалярне множення за фіксованою точкою ЕК (генератором групи);
- перевірка підпису (digest verify sign) – скалярне множення за довільною (відкритий ключ) і фіксованою точкою ЕК (генератором групи).

При генеруванні відкритого ключа та створення підпису відбувається скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі з використанням проективних координат Чудновського. При перевірці підпису відбувається множення фіксованої (базової) точки на основі алгоритму Лім-Лі і довільної точки (відкритого ключа) на основі двійкового алгоритму зліва направо з використанням проективних координат Чудновського.

Слід зазначити, що операції у базовому полі виконуються за модулем простого числа спеціального вигляду – псевдомерсена, а операції у полі порядку базової точки, відбувається за модулем простого числа загального вигляду.

Значення швидкодії вказаних перетворень криптосистеми ECDSA над простим полем GF(p) зазначені в мілісекундах.

Есперименталні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку Д, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії криптографічних перетворень у криптосистемі ECDSA, яка базується на перетвореннях групі точок ЕК над простим полем, можна зробити висновок, що запропоновані методи ефективніші, ніж класичні. В Табл. 4.7 показані зведені дані нормалізованих результатів вимірювання швидкодії перетворень з використанням запропонованих методів арифметичних перетворень.

Таблиця 4.7

Зведена таблиця нормалізованих результатів вимірювання швидкодії криптографічних перетворень криптосистеми ECDSA

Операція	Платформа						
	Server		Desktop		Mobile		Embedded
	w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
Private key generation	1,03-1,08	1,02-1,05	1,02-1,18	1,02-1,08	1,09-1,18	1,02-1,04	1,03-1,09
Public key generation	1,01-1,03	1,01-1,02	1,02-1,06	1,01-1,04	1,02-1,05	1,02-1,05	1,02-1,04
Digest sign	1,02-1,04	1,01-1,02	1,01-1,03	1,02-1,05	1,02-1,10	1,02-1,05	1,04-1,06
Digest verify sign	1,01-1,02	1,01-1,05	1,01-1,03	1,01-1,04	1,01-1,03	1,01-1,05	1,04-1,29

Загалом, відповідно до Табл. 4.7, криптографічні операції криптосистеми ECDSA з використанням запропонованих методів арифметичних перетворень у групі точок ЕК над простим полем ефективніші:

- генерування особистого ключа – в 1,02-1,18 разів для *w*=32 біт, та 1,02-1,09 разів для *w*=64 біт;
- генерування відкритого ключа – в 1,01-1,06 разів для *w*=32 біт, та 1,01-1,05 разів для *w*=64 біт;
- створення підпису – в 1,01-1,10 разів для *w*=32 біт, та 1,01-1,05 разів для *w*=64 біт;
- перевірка підпису – в 1,01-1,03 разів для *w*=32 біт, та 1,01-1,29 разів для *w*=64 біт.

Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, їх двійкова довжина практично не впливає на ефективність запропонованих методів. Розпаралелювання перетворень у базовому полі не застосовувалось, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемі ECDSA та інших.

Крім того, слід зазначити, що сучасні процесори будуються за 64-х розрядною архітектурою, що показує значно більшу ефективність 64-х бітних реалізацій як класичних, так і запропонованих методів.

Далі будуть розглядатися результати національної криптографічної системи ДСТУ 4145-2002, яка базується на арифметичних операціях в групі точок ЕК над простим полем GF(2^m).

4.7. Експериментальні дослідження методів арифметичних перетворень в криптосистемі ДСТУ 4145-2002 над двійковим полем GF(2^m)

ДСТУ 4145-2002 один з основних алгоритмів електронного підпису, що використовується в національній інфраструктурі відкритих ключів України. Проте, в ДСТУ 4145-2002 переважають операції з поліномами, а операції з цілими числами складають незначну частину і використовуються лише для перетворень у простому полі GF(p) порядку базової точки ЕК. В зв'язку з цим, запропоновані методи арифметичних перетворень не дають суттєвого ефекту для ДСТУ 4145-2002.

Для експериментального дослідження були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Експериментальні дослідження проводились для основних перетворень криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m):

- генерування особистого ключа (private key generation);
- генерування відкритого ключа (public key generation);
- обчислення передпідпису (presignature generation);
- створення підпису з передпідписом (digest sign with presignature);
- перевірка підпису (digest verify sign).

При генеруванні відкритого ключа, обчисленні передпідпису, створенню підпису з передпідписом відбувається скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі з використанням проективних координат Лопеса-Дахаба. При перевірці підпису застосовується скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі і довільної точки (відкритого ключа) на основі двійкового алгоритму зліва направо з використання проективних координат Лопеса-Дахаба.

Значення швидкодії вказаних перетворень криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) зазначені в мілісекундах.

В Додатку Е, для наочності, наведено результати вимірювання швидкодії основних перетворень криптосистеми ДСТУ 4145-2002 з використанням запропонованих методів арифметичних перетворень, що отримані для платформ Server, Desktop, Mobile та Embedded.

Результати швидкодії криптографічних перетворень національної криптосистеми ДСТУ 4145-2002 приводяться лише для порівняння з швидкодією криптосистеми ECDSA, оскільки криптосистема ДСТУ 4145-2002 базується на перетвореннях групі точок ЕК над двійковим полем та використовує незначну кількість перетворень у простому полі – поля порядку групи точок ЕК.

Розпаралелювання перетворень у полі порядку не виконувалося, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемі ДСТУ 4145-2002.

4.8. Експериментальні дослідження методів арифметичних перетворень в криптосистемі RSA

Міжнародний алгоритм електронного підпису RSA, як і алгоритм ECDSA, дозволений до використання [6] і активно використовується в національній інфраструктурі відкритих ключів України.

Для перевірки запропонованих методів арифметичних перетворень в криптосистемі RSA були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Для порівняння отриманих результатів в якості еталона були використані операції без застосування запропонованих методів. Порівняння проводилось шляхом співставлення середнього часу виконання 1 мільйону ітерації перетворень в програмній реалізації:

- з використанням класичних методів арифметичних перетворень (original).
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання (with improvements);
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання та розпаралелюванням в два потоки (with improvements and 2 threads). Розпаралелювання задіяне в операції генерування ключів – прості числа p та q генеруються в окремих потоках;
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання та розпаралелюванням в чотири потоки (with improvements and 4 threads). Розпаралелювання задіяне в операції генерування ключів – прості числа p та q генеруються в окремих потоках, а арифметичні операції множення та піднесення до квадрату виконуються в 2 потоки;
- з використанням запропонованих методів арифметичних перетворень без розпаралелювання та розпаралелюванням в декілька потоків (with improvements and multi threads). Розпаралелювання задіяне в операції генерування ключів – прості числа p та q генеруються в окремих потоках, а арифметичні операції множення та піднесення до квадрату виконуються в декілька потоків.

Слід зазначити, що операції піднесення до степеня виконуються у кільці цілих чисел у області визначення Монтгомері, що дозволяє суттєво зменшити складність операції приведення за модулем під час піднесення до степеня.

Експериментальні дослідження проводились для основних перетворень в криптосистемі RSA:

- генерування особистого ключа (private key generation);
- створення підпису (digest sign);

- перевірка підпису (digest verify sign).

Значення швидкодії вказаних перетворень в кільці цілих чисел зазначені в мілісекундах.

Есперимантальні результати, що отримані для платформ Server, Desktop, Mobile та Embedded представлені в Додатку Б, нижче представлено результати аналізу отриманих результатів.

За результатами аналізу швидкодії криптографічних перетворень у криптосистемі RSA можна зробити висновок, що запропоновані методи ефективніші, ніж класичні. В Табл. 4.8 показані зведені дані нормалізованих результатів вимірювання швидкодії перетворень з використанням запропонованих методів арифметичних перетворень.

Таблиця 4.8

Зведена таблиця нормалізованих результатів вимірювання швидкодії криптографічних перетворень у криптосистемі RSA

Операція		Платформа						
		Server		Desktop		Mobile		Embedded
		w=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=32 біт	*w*=64 біт	*w*=64 біт
PKG	1	1,01-3,72	1,02-2,19	1,09-4,,53	1,02-2,42	1,05-5,94	1,01-2,64	-
	2	1,08-6,33	1,01-3,25	1,04-4,16	1,08-3,75	1,01-3,51	1,01-4,57	-
	3	1,13-13,18	1,02-7,98	1,07-8,88	1,02-4,68	1,01-5,27	1,01-8,29	-
	4	1,11-12,74	1,09-6,74	1,02-11,86	1,09-6,74	1,05-7,94	1,11-8,02	-
S	1	1,58-10,0	1,01-1,10	1,48-1,86	1,01-1,15	1,62-2,33	1,04-1,24	1,01-1,13
	2	1,01-1,07	1,01-1,06	1,01-1,14	1,01-1,33	1,01-1,13	1,01-1,10	1,01-2,67
	3	1,33-1,93	1,17-2,04	1,31-2,42	1,18-1,93	1,05-1,86	1,20-1,84	1,19-1,89
	4	1,26-5,24	1,19-5,25	1,17-4,54	1,37-3,48	1,20-3,66	1,55-3,04	1,17-3,09
V	1	1,01-1,29	1,01-1,2	1,01-1,18	1,01-1,17	1,04-1,39	1,01-1,17	1,01-1,16
	2	1,01-1,09	1,01-1,06	1,01-1,25	1,01-1,13	1,01-1,09	1,01-1,10	2,01-2,54
	3	1,01-1,18	1,01-1,14	1,01-1,20	1,01-1,17	1,01-1,10	1,01-1,10	1,01-1,14
	4	1,01-1,12	-	1,01-1,20	-	-	-	1,01-1,14

У Табл. 4.8 позначено: 1 – відношення результатів з застосуванням класичних методів до результатів з застосуванням запропонованих методів; 2 – відношення результатів з застосуванням класичних методів до результатів з застосуванням запропонованих методів та розпаралелюванням у 2 потоки; 3 – відношення результатів з застосуванням класичних методів до результатів з застосуванням запропонованих методів та розпаралелюванням у 4 потоки;

4 – відношення результатів з застосуванням класичних методів до результатів з застосуванням запропонованих методів та розпаралелюванням у декілька потоків.

Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної платформи, розрядності машинних слів та має локальний екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність.

Загалом, відповідно до Табл. 4.8, криптографічні операції криптосистеми RSA з використанням запропонованих методів арифметичних перетворень ефективніші:

- генерування особистого ключа – в 1,01-5,94 разів для w=32 біт, та 1,01-2,64 разів для w=64 біт;
- створення підпису – в 1,48-10,0 разів для w=32 біт, та 1,01-1,24 разів для w=64 біт;
- перевірка підпису – в 1,01-1,39 разів для w=32 біт, та 1,01-1,2 разів для w=64 біт .

Загалом, відповідно до Табл. 4.8, криптографічні операції криптосистеми RSA з використанням запропонованих методів арифметичних перетворень з розпаралелюванням ефективніші, ніж з використанням запропонованих методів без розпаралелювання:

- генерування особистого ключа з застосуванням розпаралелювання у 2 потоки – в 1,01-6,33 разів для w=32 біт, та 1,01-4,57 разів для w=64 біт;
- створення підпису з застосуванням розпаралелювання у 2 потоки – в 1,01-1,13 разів для w=32 біт, та 1,01-2,67 разів для w=64 біт;
- перевірка підпису з застосуванням розпаралелювання у 2 потоки – в 1,01-1,25 разів для w=32 біт, та 1,01-2,54 разів для w=64 біт .

- генерування особистого ключа з застосуванням розпаралелювання у 4 потоки – в 1,01-13,18 разів для w=32 біт, та 1,01-8,29 разів для w=64 біт;
- створення підпису з застосуванням розпаралелювання у 4 потоки – в 1,05-2,42 разів для w=32 біт, та 1,19-2,04 разів для w=64 біт;
- перевірка підпису з застосуванням розпаралелювання у 4 потоки – в 1,01-1,20 разів для w=32 біт, та 1,01-1,17 разів для w=64 біт .
- генерування особистого ключа з застосуванням розпаралелювання у декілька потоків – в 1,02-12,74 разів для w=32 біт, та 1,09-8,02 разів для w=64 біт;
- створення підпису з застосуванням розпаралелювання у декілька потоків – в 1,17-5,24 разів для w=32 біт, та 1,17-5,25 разів для w=64 біт;
- перевірка підпису з застосуванням розпаралелювання у декілька потоків – в 1,01-1,20 разів для w=32 біт, та 1,01-1,14 разів для w=64 біт.

Розпаралелювання для операції генерації ключів показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин, що перевищують 1536 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт;

Розпаралелювання для операції підписання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин, що перевищують 1536 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт.

Розпаралелювання для операції перевірки підпису не є ефективним, через незначний розмір відкритого ключа, при якому, розпаралелювання у 2 і більше потоків призводить до суттєвих накладних витрат обчислювальних ресурсів процесору.

4.9. Висновки до розділу

За результатами проведених досліджень вимірювання швидкодії можливо зробити наступні висновки:

1. Оцінка швидкодії арифметичних перетворень (множення, піднесення до квадрату та приведення за модулем) над цілими числами виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів арифметичних перетворень; з використанням запропонованих методів арифметичних перетворень без розпаралелювання; з використанням запропонованих методів арифметичних перетворень та розпаралелюванням в 2 потоки; з використанням запропонованих методів арифметичних перетворень та розпаралелюванням в декілька потоків. Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної платформи, розрядності машинних слів та має локальний екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність. Загалом, запропоновані методи арифметичних перетворень ефективніші: множення цілих чисел – в 1,05-1,7 разів для w=32 біт, та 1,02-2,28 разів для w=64 біт; множення цілих чисел з розпаралелюванням в 2 потоки – в 1,01-3,24 разів для w=32 біт, та 1,07-2,29 разів для w=64 біт; множення цілих чисел з розпаралелюванням в декілька потоків – в 1,34-8,29 разів для w=32 біт, та 1,05-5,1 разів для w=64 біт; піднесення до квадрату цілих чисел – в 1,25-3,19 разів для w=32 біт, та 1,01-4,12 разів для w=64 біт; піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки – в 1,01-5,75 разів для w=32 біт, та 1,18-2,46 разів для w=64 біт; піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків – в 1,31-17,3 разів для w=32 біт, та 1,09-4,6 разів для w=64 біт; приведення за модулем цілих чисел – в 1,02-1,8 разів для w=32 біт, та 1,01-4,12 разів для w=64 біт; приведення за модулем цілих чисел з розпаралелюванням в 2 потоки – в 1,22-2,10 разів для w=32 біт, та 1,04-2,46 разів для w=64 біт; приведення за модулем цілих чисел з розпаралелюванням в декілька потоків – в 1,22-5,14 разів для w=32 біт, та 1,31-

4,6 разів для w=64 біт. Розпаралелювання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин, що перевищують 2048 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт. Ефективність розпаралелювання залежить від загальної архітектури процесорів та внутрішньої схеми між'ядерної взаємодії. Загалом, запропоновані методи арифметичних перетворень з розпаралелюванням ефективніші запропонованих методів без розпаралелювання: множення цілих чисел з розпаралелюванням в 2 потоки – в 1,17-2,71 разів для w=32 біт, та 1,02-2,03 разів для w=64 біт; множення цілих чисел з розпаралелюванням в декілька потоків – в 1,72-6,81 разів для w=32 біт, та 1,10-4,91 разів для w=64 біт; піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки – в 1,03-1,82 разів для w=32 біт, та 1,16-2,33 разів для w=64 біт; піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків – в 1,09-5,43 разів для w=32 біт, та 1,01-4,36 разів для w=64 біт; приведення за модулем цілих чисел з розпаралелюванням в 2 потоки – в 1,02-1,64 разів для w=32 біт, та 1,05-1,63 разів для w=64 біт; приведення за модулем цілих чисел з розпаралелюванням в декілька потоків – в 1,05-4,5 разів для w=32 біт, та 1,19-3,32 разів для w=64 біт.

2. Оцінка швидкодії арифметичних перетворень у кільці цілих чисел виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів арифметичних перетворень у кільці цілих чисел; з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел без розпаралелювання; з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел та розпаралелюванням в 2 потоки; з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел та розпаралелюванням в декілька потоків. Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної архітектури, розрядності машинних слів та має локальний

екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність. Загалом запропоновані методи арифметичних перетворень у кільці цілих чисел ефективніші: операція додавання за модулем – в 1,01-1,97 разів для w=32 біт, та 1,03-3,28 разів для w=64 біт; операція віднімання за модулем – в 1,01-1,34 разів для w=32 біт, та 1,03-2,41 разів для w=64 біт; операція множення за модулем – в 1,01-1,34 разів для w=32 біт, та 1,03-2,41 разів для w=64 біт; операція піднесення до квадрату за модулем – в 1,02-1,46 разів для w=32 біт, та 1,02-1,73 разів для w=64 біт; операція піднесення до степеню за модулем – в 1,38-1,75 разів для w=32 біт, та 1,01-1,76 разів для w=64 біт. Розпаралелювання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин що перевищують 2048 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт (виключення складає піднесення до степеню, коли має сенс вже при 2048 біт). Загалом, запропоновані методи арифметичних перетворень в кільці цілих чисел з розпаралелюванням ефективніші запропонованих методів без розпаралелювання: операція додавання за модулем з розпаралелюванням в 2 потоки – в 1,01-2,07 разів для w=32 біт, та 1,01-1,74 разів для w=64 біт; операція додавання за модулем з розпаралелюванням в декілька потоків – в 1,01-1,97 разів для w=32 біт, та 1,01-1,97 разів для w=64 біт; операція віднімання за модулем з розпаралелюванням в 2 потоки – в 1,01-1,23 разів для w=32 біт, та 1,03-1,23 разів для w=64 біт; операція віднімання за модулем з розпаралелюванням в декілька потоків – в 1,02-1,44 разів для w=32 біт, та 1,01-1,44 разів для w=64 біт; операція множення за модулем з розпаралелюванням в 2 потоки – в 1,02-1,24 разів для w=32 біт, та 1,02-1,21 разів для w=64 біт; операція множення за модулем з розпаралелюванням в декілька потоків – в 1,02-1,2 разів для w=32 біт, та 1,02-1,14 разів для w=64 біт; операція піднесення до квадрату за модулем з розпаралелюванням в 2 потоки – в 1,03-1,15 разів для w=32 біт, та 1,01-1,15 разів для w=64 біт; операція піднесення до квадрату за модулем з

розпаралелюванням в декілька потоків – в 1,01-1,12 разів для w=32 біт, та 1,01-1,09 разів для w=64 біт; операція піднесення до степеню за модулем з розпаралелюванням в 2 потоки – в 1,16-2,5 разів для w=32 біт, та 1,16-2,01 разів для w=64 біт; операція піднесення до степеню за модулем з розпаралелюванням в декілька потоків – в 1,11-4,62 разів для w=32 біт, та 1,23-4,69 разів для w=64 біт.

3. Оцінка швидкодії арифметичних перетворень у простому полі цілих чисел виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів і з використанням запропонованих методів арифметичних перетворень у простому полі цілих чисел. Розпаралелювання перетворень у полі не виконувалося, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемах ECDSA, ДСТУ 4145-2002 та інших. Розглядалися прості числа загального і спеціального вигляду (псевдомерсена) у якості модулей. Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, на ефективність практично не впливає їх двійкова довжина. Загалом запропоновані методи арифметичних перетворень в простому полі цілих чисел ефективніші: операція додавання (модуль загального вигляду) – в 1,02-1,09 разів для w=32 біт, та 1,01-1,07 разів для w=64 біт; операція додавання (модуль спеціального вигляду) – в 1,01-1,08 разів для w=32 біт, та 1,01-1,04 разів для w=64 біт; операція віднімання (модуль загального вигляду) – в 1,01-1,08 разів для w=32 біт, та 1,02-1,05 разів для w=64 біт; операція віднімання (модуль спеціального вигляду) – в 1,01-1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт; операція множення (модуль загального вигляду) – в 1,02-1,09 разів для w=32 біт, та 1,01-1,07 разів для w=64 біт; операція множення (модуль спеціального вигляду) – в 1,02-1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт; операція піднесення до квадрату (модуль загального вигляду) – в 1,02-1,08 разів для w=32 біт, та 1,01-1,07 разів для w=64 біт; операція піднесення до квадрату (модуль спеціального вигляду) – в 1,02-

1,07 разів для w=32 біт, та 1,01-1,08 разів для w=64 біт; операція піднесення до степеню (модуль загального вигляду) – в 1,03-1,12 разів для w=32 біт, та 1,02-1,14 разів для w=64 біт; операція піднесення до степеню (модуль спеціального вигляду) – в 1,03-1,09 разів для w=32 біт, та 1,02-1,13 разів для w=64 біт.

4. Оцінка швидкодії перетворень у групі точок ЕК над простим полем (додавання, подвоєння та скалярне множення) виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів і з використанням запропонованих методів арифметичних перетворень у групі точок ЕК над простим полем цілих чисел. Розпаралелювання перетворень у базовому полі не застосовувалось, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемах ECDSA, ДСТУ 4145-2002 та інших. Розглядалися прості числа загального і спеціального вигляду (псевдомерсена) у якості модулей, оскільки вони передбачені NIST FIPS 186-3. Оцінювалася швидкодія перетворень у групі точок ЕК над простим полем цілих чисел у афінних координатах, проективних координатах Чудновського та змішаних координатах. Оцінка швидкодії арифметичних перетворень у групі точок ЕК над простим полем проводилась для наступних перетворень: додавання і подвоєння точок у афінних координатах, додавання і подвоєння точок у проективних координатах Чудновського, скалярне множення точок ЕК з довільною точкою на основі бінарного алгоритму зліва направо, скалярне множення з фіксованою точкою за алгоритмом Лім-Лі. Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, на ефективність практично не впливає їх двійкова довжина. Загалом запропоновані методи арифметичних перетворень у групі точок ЕК над простим полем ефективніші: додавання точок ЕК – в 1,02-1,08 разів для w=32 біт, та 1,01-1,09 разів для w=64 біт; подвоєння точок ЕК – в 1,01-1,06 разів для w=32 біт, та 1,02-1,08 разів для w=64 біт; додавання точок ЕК в змішаних координатах – в 1,03-1,06 разів для w=32 біт, та 1,02-1,07 разів для w=64 біт; подвоєння точок ЕК в проективних координатах – в 1,02-1,08

разів для w=32 біт, та 1,03-1,08 разів для w=64 біт; скалярне множення випадкової точки ЕК з використанням проективних координат – в 1,02-1,12 разів для w=32 біт, та 1,02-1,09 разів для w=64 біт; скалярне множення фіксованої точки ЕК з використанням проективних координат – в 1,02-1,12 разів для w=32 біт, та 1,02-1,10 разів для w=64 біт.

5. Оцінка швидкодії криптографічних перетворень у криптосистемі ECDSA, яка базується на перетвореннях групі точок ЕК над простим полем (додавання, подвоєння та скалярне множення) виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів і з використанням запропонованих методів арифметичних перетворень у простому полі цілих чисел. Розпаралелювання перетворень у базовому полі не застосовувалось, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемі ECDSA. Розглядалися прості числа спеціального вигляду (псевдомерсена) у якості модулей для базового поля та прості числа загального вигляду у якості модулей для поля порядку групи точок ЕК, оскільки вони передбачені NIST FIPS 186-3. Оцінка швидкодії проводилась для криптографічних перетворень (генерування особистого ключа; генерування відкритого ключа, як скалярне множення за фіксованою точкою ЕК; створення підпису, як скалярне множення за фіксованою точкою ЕК; перевірка підпису, як скалярне множення за довільною і фіксованою точкою ЕК) криптосистеми ECDSA, яка базується на перетвореннях у групі точок ЕК над простим полем (додавання і подвоєння точок у афінних координатах, додавання і подвоєння точок у проективних координатах Чудновського, скалярне множення точок ЕК з довільною точкою на основі бінарного алгоритму зліва направо, скалярне множення з фіксованою точкою за алгоритмом Лім-Лі). Через незначну різницю у двійкових довжинах цілих чисел – елементів простого поля, на швидкодію практично не впливає їх двійкова довжина. Загалом, криптографічні операції криптосистеми ECDSA з використанням запропонованих методів

арифметичних перетворень у групі точок ЕК над простим полем ефективніші: генерування особистого ключа – в 1,02-1,18 разів для w=32 біт, та 1,02-1,09 разів для w=64 біт; генерування відкритого ключа – в 1,01-1,06 разів для w=32 біт, та 1,01-1,05 разів для w=64 біт; створення підпису – в 1,01-1,10 разів для w=32 біт, та 1,01-1,05 разів для w=64 біт; перевірка підпису – в 1,01-1,03 разів для w=32 біт, та 1,01-1,29 разів для w=64 біт.

6. Оцінка швидкодії криптографічних перетворень (генерація особистого ключа, створення і перевірка підпису) у криптосистемі ДСТУ 4145-2002, яка базується на перетвореннях групі точок ЕК над двійковим полем (додавання, подвоєння та скалярне множення) виконувалася для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів з використанням запропонованих методів арифметичних перетворень. Результати швидкодії криптографічних перетворень національної криптосистеми ДСТУ 4145-2002 приводяться лише для порівняння з швидкодією криптосистеми ECDSA, оскільки криптосистема ДСТУ 4145-2002 базується на перетвореннях групі точок ЕК над двійковим полем та використовує незначну кількість перетворень у простому полі – поля порядку групи точок ЕК. Розпаралелювання перетворень у полі порядку не виконувалося, оскільки ефект від розпаралелювання досягається при значно більших значеннях розміру полів, ніж ті, що використовуються у криптосистемі ДСТУ 4145-2002 та незначної кількості таких перетворень (у простому полі порядку групи точок ЕК). Розглядалися прості числа загального вигляду якості модулей для поля порядку групи точок ЕК, оскільки вони передбачені ДСТУ 4145-2002.

7. Оцінка швидкодії криптографічних перетворень (генерування ключів, створення підпису та перевірка підпису) у криптосистемі RSA виконується для 4 апаратних платформ з застосуванням 32-х та 64-х розрядних машинних слів: без використання запропонованих методів арифметичних перетворень у кільці цілих чисел; з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел без розпаралелювання; з використанням

запропонованих методів арифметичних перетворень у кільці цілих чисел та розпаралелюванням в 2 потоки (генерація простих чисел p і q відбувається в окремих потоках); з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел та розпаралелюванням в 4 потоки (генерація простих чисел p і q відбувається в окремих потоках, з розпаралелюванням арифметичних перетворень у 2 потоки); з використанням запропонованих методів арифметичних перетворень у кільці цілих чисел та розпаралелюванням в декілька потоків (генерація простих чисел p і q відбувається в окремих потоках, з розпаралелюванням арифметичних перетворень у декілька потоків). Ефективність запропонованих методів суттєво залежить від двійкової довжини цілого числа, апаратної платформи, розрядності машинних слів та має локальний екстремум, що дозволяє говорити про двійкові довжини цілих чисел, для яких запропоновані методи мають найбільшу ефективність. Загалом, криптографічні операції криптосистеми RSA з використанням запропонованих методів арифметичних перетворень ефективніші: генерування особистого ключа – в 1,01-5,94 разів для w=32 біт, та 1,01-2,64 разів для w=64 біт; створення підпису – в 1,48-10,0 разів для w=32 біт, та 1,01-1,24 разів для w=64 біт; перевірка підпису – в 1,01-1,39 разів для w=32 біт, та 1,01-1,2 разів для w=64 біт. Загалом, криптографічні операції криптосистеми RSA з використанням запропонованих методів арифметичних перетворень з розпаралелюванням ефективніші, ніж з використанням запропонованих методів без розпаралелювання: генерування особистого ключа з застосуванням розпаралелювання у 2 потоки – в 1,01-6,33 разів для w=32 біт, та 1,01-4,57 разів для w=64 біт; створення підпису з застосуванням розпаралелювання у 2 потоки – в 1,01-1,13 разів для w=32 біт, та 1,01-2,67 разів для w=64 біт; перевірка підпису з застосуванням розпаралелювання у 2 потоки – в 1,01-1,25 разів для w=32 біт, та 1,01-2,54 разів для w=64 біт; генерування особистого ключа з застосуванням розпаралелювання у 4 потоки – в 1,01-13,18 разів для w=32 біт, та 1,01-8,29 разів для w=64 біт; створення підпису з застосуванням розпаралелювання у 4

потоки – в 1,05-2,42 разів для w=32 біт, та 1,19-2,04 разів для w=64 біт; перевірка підпису з застосуванням розпаралелювання у 4 потоки – в 1,01-1,20 разів для w=32 біт, та 1,01-1,17 разів для w=64 біт; генерування особистого ключа з застосуванням розпаралелювання у декілька потоків – в 1,02-12,74 разів для w=32 біт, та 1,09-8,02 разів для w=64 біт; створення підпису з застосуванням розпаралелювання у декілька потоків – в 1,17-5,24 разів для w=32 біт, та 1,17-5,25 разів для w=64 біт; перевірка підпису з застосуванням розпаралелювання у декілька потоків – в 1,01-1,20 разів для w=32 біт, та 1,01-1,14 разів для w=64 біт. Розпаралелювання для операції генерації ключів показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин що перевищують 1536 біт, а розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт. Розпаралелювання для операції підписання показує свою ефективність при значно більшій довжині цілих чисел. Наприклад, для платформи Server, розпаралелювання у 2 потоки має сенс застосовувати для довжин що перевищують 1536 біт, та розпаралелювання у 4 і більше потоків має сенс при довжинах, що перевищують 3072 біт. Розпаралелювання для операції перевірки підпису не є ефективним, через незначний розмір відкритого ключа, при якому, розпаралелювання у 2 і більше потоків призводить до суттєвих накладних витрат обчислювальних ресурсів процесору.

Список використаних джерел у четвертому розділі

1. GCC, the GNU Compiler Collection. URL: https://gcc.gnu.org/
2. OpenMP. URL: https://www.openmp.org/
3. Taschwer M. (2001) Modular Multiplication Using Special Prime Moduli. In: Horster P. (eds) Kommunikationssicherheit im Zeichen des Internet. DuD-Fachbeiträge. Vieweg+Teubner Verlag. – 2001. URL: http://www.isys.uni-klu.ac.at/PDF/2001-0126-MT.pdf
4. Daniel V. Bailey. Computation in Optimal Extension Fields. A Thesis submitted to the Faculty of the Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science. May, 2000. - http://m.wpi.edu/Pubs/ETD/Available/etd-0428100-133037/unrestricted/bailey.pdf
5. Genus-1 curves over large-characteristic fields. URL: http://hyperelliptic.org/EFD/g1p/index.html
6. Про встановлення вимог до технічних засобів, процесів їх створення, використання та функціонування у складі інформаційно-телекомунікаційних систем під час надання електронних довірчих послуг. – Наказ Міністерства юстиції України та Адміністрації державної служби спеціального зв'язку та захисту інформації України № 3563/5/610 від 18.11.2019.
7. A. Okhrimenko, V. Kovtun «Experimental research of the developed methods of arithmetic operations in cryptographic transformations according to the ECDSA scheme» Proceedings 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Lviv, Ukraine, November 29, 2019. – CEUR Workshop Proceedings, p. 827-837.
8. A. Okhrimenko, V. Kovtun «Experimental research of developed arithmetic transformations according to RSA», XX International conference of higher education students and young scientists «POLIT. Challenges of science today: Modern information and communication technologies in aviation», Kyiv, May 1-3, 2020., p. 30-31.

ВИСНОВКИ

Результатом виконаної роботи є розв'язання актуальної науково-практичної задачі підвищення швидкодії ІТС ЦСК національної ІВК за рахунок підвищення швидкодії реалізації криптографічних алгоритмів на основі розробки методів та алгоритмів арифметичних перетворень над великими цілими числами з відкладеним переносом.

У процесі виконання дисертаційної роботи отримані такі основні результати:

1. Проведено аналіз сучасних методів представлення цілих чисел, визначено їх переваги і недоліки, сфери застосування. Встановлено, що найбільш універсальним є двійкове представлення цілих чисел, проте воно є не завжди оптимальним, а решта представлень мають вузьку сферу використання.

2. Розроблено метод представлення цілих чисел з відкладеним переносом, який дозволяє відкласти перенос при виконанні арифметичних перетворень, що в свою чергу дозволяє підвищити швидкодію криптографічних перетворень з відкритим ключем.

3. Удосконалено методи арифметичних перетворень, які за рахунок використання цілих чисел в представленні з відкладеним переносом, дозволяють підвищити швидкодію перетворень в полях (операція додавання (модуль загального вигляду) – в 1,02-1,09 разів для $w = 32$ біт, та 1,01-1,07 разів для $w = 64$ біт; операція додавання (модуль спеціального вигляду) – в 1,01-1,08 разів для $w = 32$ біт, та 1,01-1,04 разів для $w = 64$ біт; операція віднімання (модуль загального вигляду) – в 1,01-1,08 разів для $w = 32$ біт, та 1,02-1,05 разів для $w = 64$ біт; операція віднімання (модуль спеціального вигляду) – в 1,01-1,07 разів для $w = 32$ біт, та 1,01-1,08 разів для $w = 64$ біт; операція множення (модуль загального вигляду) – в 1,02-1,09 разів для $w = 32$ біт, та 1,01-1,07 разів для $w = 64$ біт; операція множення (модуль спеціального вигляду) – в 1,02-1,07 разів для $w = 32$ біт, та 1,01-1,08 разів для $w = 64$ біт; операція піднесення до квадрату (модуль загального вигляду) – в 1,02-1,08 разів для $w = 32$ біт, та 1,01-1,07 разів для $w = 64$ біт; операція піднесення до квадрату (модуль спеціального

вигляду) – в 1,02-1,07 разів для $w=32$ біт, та 1,01-1,08 разів для $w=64$ біт; операція піднесення до степеню (модуль загального вигляду) – в 1,03-1,12 разів для $w=32$ біт, та 1,02-1,14 разів для $w=64$ біт; операція піднесення до степеню (модуль спеціального вигляду) – в 1,03-1,09 разів для $w=32$ біт, та 1,02-1,13 разів для $w=64$ біт.) та кільцях цілих чисел (операція додавання за модулем – в 1,01-1,97 разів для $w=32$ біт, та 1,03-3,28 разів для $w=64$ біт; операція віднімання за модулем – в 1,01-1,34 разів для $w=32$ біт, та 1,03-2,41 разів для $w=64$ біт; операція множення за модулем – в 1,01-1,34 разів для $w=32$ біт, та 1,03-2,41 разів для $w=64$ біт; операція піднесення до квадрату за модулем – в 1,02-1,46 разів для $w=32$ біт, та 1,02-1,73 разів для $w=64$ біт; операція піднесення до степеню за модулем – в 1,38-1,75 разів для $w=32$ біт, та 1,01-1,76 разів для $w=64$ біт), що в свою чергу призводить до підвищення швидкодії криптографічних перетворень з відкритим ключем.

4. Удосконалено методи арифметичних перетворень множення (дозволив підвищити швидкодію реалізації в 1,01-3,24 разів для $w=32$ біт, та 1,07-2,29 разів для $w=64$ біт відносно прототипу), піднесення до квадрату (дозволив підвищити швидкодію реалізації в 1,01-5,75 разів для $w=32$ біт, та 1,18-2,46 разів для $w=64$ біт відносно прототипу) та приведення за модулем (дозволив підвищити швидкодію реалізації в 1,22-2,10 разів для $w=32$ біт, та 1,04-2,46 разів для $w=64$ біт відносно прототипу) великих цілих чисел з відкладеним переносом та паралельним виконанням двох циклів множення в двох окремих потоках, що дозволяє підвищити швидкодію криптографічних перетворень з відкритим ключем.

5. Розроблено методи арифметичних перетворень множення (дозволив підвищити швидкодію реалізації в 1,34-8,29 разів для $w=32$ біт, та 1,05-5,1 разів для $w=64$ біт відносно прототипу), піднесення до квадрату (дозволив підвищити швидкодію реалізації в 1,31-17,3 разів для $w=32$ біт, та 1,09-4,6 разів для $w=64$ біт відносно прототипу) та приведення за модулем (дозволив підвищити швидкодію реалізації в 1,22-5,14 разів для $w=32$ біт, та 1,31-4,6 разів для $w=64$ біт відносно прототипу) великих цілих чисел з

відкладеним переносом та паралельним виконанням ітерацій двох циклів множення в декілька потоків, що дозволяє підвищити швидкодію криптографічних перетворень з відкритим ключем.

6. В свою чергу розроблені та удосконалені методи арифметичних перетворень дозволяють підвищити швидкодію криптографічних операцій у криптосистемі ECDSA (генерування особистого ключа – в 1,02-1,18 разів для $w = 32$ біт, та 1,02-1,09 разів для $w = 64$ біт; генерування відкритого ключа – в 1,01-1,06 разів для $w = 32$ біт, та 1,01-1,05 разів для $w = 64$ біт; створення підпису – в 1,01-1,10 разів для $w = 32$ біт, та 1,01-1,05 разів для $w = 64$ біт; перевірка підпису – в 1,01-1,03 разів для $w = 32$ біт, та 1,01-1,29 разів для $w = 64$ біт), та у криптосистемі RSA (генерування особистого ключа – в 1,01-5,94 разів для $w = 32$ біт, та 1,01-2,64 разів для $w = 64$ біт; створення підпису – в 1,48-10,0 разів для $w = 32$ біт, та 1,01-1,24 разів для $w = 64$ біт; перевірка підпису – в 1,01-1,39 разів для $w = 32$ біт, та 1,01-1,2 разів для $w = 64$ біт)

7. Ефективність запропонованих методів арифметичних перетворень підтверджується результатами експериментального дослідження за допомогою розробленого програмного забезпечення.

8. Зазначені результати роботи впроваджено у діяльність ТОВ «Сайфер ЛТД» (Акт № 22/17 від 04.08.2017 р.), Національного авіаційного університету (Акт від 25.09.2020 р.) та Кваліфікованого надавача електронних довірчих послуг Офісу Генерального прокурора (Акт №18\10\2-8654-19 від 18.10.2020 р.), що підтверджено відповідними актами впровадження, які містяться у додатках до дисертаційної роботи.

ДОДАТОК А

ДОКУМЕНТИ, ЩО ПІДТВЕРДЖУЮТЬ ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОЇ РОБОТИ

САЙФЕР
Системи захисту інформації

Адреса: 04107, Київ, вул. Нагірна, 25
Тел./Факс: (044) 484-46-17, 484-46-12, 483-03-22
E-mail: sales@cipher.kiev.ua
http://www.elpay.com; http://www.cipher.kiev.ua

№ 22/17 від 04 серпня 2017 р.

АКТ
про впровадження результатів дисертаційної роботи
Охріменка Андрія Олександровича

Комісія у складі голови – директора товариства з обмеженою відповідальністю «Сайфер ЛТД», кандидата технічних наук Боровікова О.М., членів комісії – провідного розробника Бойко С.Т., провідного розробника Прокоповича Л.Ю., складено цей акт про те, що при розробці бібліотек криптографічних примітивів «Шифр+» v2.1, реалізовано такі результати наукових досліджень Охріменка Андрія Олександровича:

1. Для криптосистем за схемами ECDSA над полем GF(p), ECGDSA над полем GF(p), RSA1 Signature, RSA2 Signature, RSA Encryption, використано метод арифметичних перетворень над великими числами з відкладеним переносом, який дозволяє виключити взаємозалежність (послідовне виконання) арифметичних операцій з-за необхідності врахування переноса, що дозволило підвищити швидкодію (до 30%, в залежності від криптосистеми та довжини блоку перетворень) і стійкість до атак на реалізацію криптографічних примітивів.

Запропоновані, у результаті виконання наукових досліджень Охріменка Андрія Олександровича, алгоритми, дозволяють суттєво підвищити швидкодію криптографічних перетворень «Шифр+» v.2.1 у компонентах центру сертифікації ключів на основі системи криптографічного захисту інформації «Шифр-X.509».

Голова комісії
Директор ТОВ «Сайфер ЛТД», к.т.н. О.М. Боровіков

Члени комісії:

Провідний розробник С.Т. Бойко

Провідний розробник Л.Ю. Прокопович

Прокуратура України
ОФІС ГЕНЕРАЛЬНОГО ПРОКУРОРА
вул. Різницька, 13/15, м. Київ, 01011 факс: 280-26-03

08\10\20 № 18\10\2-8654-19

На № _____ від _____

АКТ
про впровадження результатів дисертаційної роботи
Охріменка Андрія Олександровича

м. Київ **08 жовтня 2020 року**

Комісією у складі голови – заступника начальника Департаменту інформаційних технологій Офісу Генерального прокурора, заступника керівника Акредитованого центру сертифікації ключів Генеральної прокуратури України (далі АЦСК ГПУ) – адміністратора сертифікації АЦСК ГПУ Купецького В.Б., членів комісії – адміністратора безпеки АЦСК ГПУ Косовських С.Г., системного адміністратора АЦСК ГПУ – Соловйова О.С., адміністратора реєстрації АЦСК ГПУ – Лисенка О.О., складено цей акт про те, що після оновлення системи криптографічного захисту інформації (далі СКЗІ) «Шифр-Х.509» до версії 2, яка побудована на основі бібліотек криптографічних примітивів «Шифр+» версії 2.1 (судячи з технічної документації), суттєво підвищилась ефективність роботи АЦСК ГПУ, а саме:

1. Зменшено час обробки запитів серверами OCSP, TSP, CMP та формування відповідей.

2. Зменшено час обробки запитів на видання сертифіката та на зміну його статусу (блокування, відновлення, скасування).

3. З'явилася можливість використовувати ключі для криптографічних алгоритмів ECDSA над полем GF(p), RSA1 Signature та RSA Encryption.

Збільшення швидкодії сервісів АЦСК ГПУ відбулося незважаючи на:

1. Збільшення кількості сертифікатів, що обслуговуються (збільшення на 17%), та запитів на зміну статусу сертифікатів.

2. Збільшення кількості сервісів, до яких підключено АЦСК ГПУ, що призвело до більш активного використання власних ключів підписувачами.

3. Залишення без зміни поточної обчислювальної інфраструктури.

За інформацією, отриманою від розробника програмного забезпечення СКЗІ АЦСК ГПУ – товариства з обмеженою відповідальністю «Сайфер БІС», встановлено, що у СКЗІ «Шифр-Х.509» версії 2 використовуються бібліотеки

криптографічних примітивів «Шифр+» версії 2.1, де реалізовано такі результати наукових досліджень Охріменка Андрія Олександровича:

для криптосистем за схемами ECDSA над полем GF(p), RSA1 Signature, RSA Encryption, використано метод арифметичних перетворень над великими цілими числами з відкладеним переносом, який дозволяє виключити взаємозалежність (послідовне виконання) арифметичних операцій через необхідність врахування переносу, що дозволило підвищити швидкодію (до 30%, в залежності від криптосистеми та довжини блоку перетворень) і стійкість до атак на реалізацію криптографічних примітивів.

Голова комісії:
Заступник начальника Департаменту інформаційних технологій Офісу Генерального прокурора, заступник керівника АЦСК ГПУ – адміністратор сертифікації АЦСК ГПУ **В.Б. Купецький**

Члени комісії:
Адміністратор безпеки АЦСК ГПУ **С.Г. Косовських**

Системний адміністратор АЦСК ГПУ **О.С. Соловйов**

Адміністратор реєстрації АЦСК ГПУ **О.О. Лисенко**

АКТ
впровадження у діяльність
НДЛ протидії кіберзагрозам в авіаційній галузі
результатів дисертаційної роботи
Охріменка Андрія Олександровича
«Методи арифметичних перетворень в полях і кільцях для криптографічних застосувань»,
подану на здобуття наукового ступеня кандидата технічних наук

Комісія у складі: голова – науковий керівник НДЛ протидії кіберзагрозам в авіаційній галузі д.т.н., доц. Гнатюк С.О., к.т.н. Кінзерявий В.М. та к.т.н. Сидоренко В.М. склали даний акт про те, що результати дисертаційної роботи Охріменка Андрія Олександровича впровадженні у науковий процес та використовувались в НДЛ протидії кіберзагрозам в авіаційній галузі під час виконання таких НДР молодих науковців:

1. «Квантово-криптографічні методи захисту критичної інформаційної інфраструктури держави» шифр 161-ДБ17 номер держреєстрації НДР 0117U006770.

Зокрема, у розробленому під час виконання роботи методі підвищення стійкості протоколів квантової криптографії було використано метод представлення цілих чисел з відкладеним переносом, який дозволяє виключити взаємозалежність між машинними операціями при виконанні арифметичних перетворень.

2. «Система забезпечення конфіденційності критичної інформаційної інфраструктури держави на базі квантових детерміністичних протоколів» шифр 305-ДБ20 номер держреєстрації НДР 0120U101400.

Зокрема, у розробленому швидкісному методі генерування псевдовипадкових послідовностей та відповідному генераторі псевдовипадкових послідовностей для криптографічних застосувань були використані удосконалені методи арифметичних перетворень з відкладеним переносом та розпаралелюванням, що покращують швидкість генерування псевдовипадкових послідовностей на 8-10%.

Якість та придатність отриманих результатів дисертаційного дослідження Охріменка А.О. перевірено та рекомендовано до використання в реальних умовах.

Голова комісії:

Науковий керівник НДЛ протидії кіберзагрозам в авіаційній галузі, заступник декана з наукової роботи ФККІП НАУ, д.т.н., доц — С. Гнатюк

Члени комісії:

Науковий співробітник НДЛ протидії кіберзагрозам в авіаційній галузі, к.т.н. — В. Кінзерявий

Науковий співробітник НДЛ протидії кіберзагрозам в авіаційній галузі, к.т.н. — В. Сидоренко

ДОДАТОК Б

Експериментальні дослідження методів арифметичних операцій над цілими числами

Експериментальні дослідження проводились саме для значущих арифметичних операцій:

- множення;
- піднесення до квадрату;
- приведення за модулем.

Значення швидкодії операцій множення цілих чисел показані в мікросекундах. M – класичний метод множення, M1 – запропонований метод множення без розпаралелювання, M2 – запропонований метод множення з використанням розпаралелювання в 2 потоки, M3 – запропонований метод множення з використанням розпаралелювання в декілька потоків.

Значення швидкодії операцій піднесення до квадрату цілих чисел показано в мікросекундах. S – класичний метод піднесення до квадрату, S1 – запропонований метод піднесення до квадрату без розпаралелювання, S2 – запропонований метод піднесення до квадрату з використанням розпаралелювання в 2 потоки, S3 – запропонований метод піднесення до квадрату з використанням розпаралелювання в декілька потоків.

Значення швидкодії операцій приведення за модулем цілих чисел показані в мікросекундах. R – класичний метод приведення за модулем, R1 – запропонований метод приведення за модулем без розпаралелювання, R2 – запропонований метод приведення за модулем з використанням розпаралелювання в 2 потоки, R3 – запропонований метод приведення за модулем з використанням розпаралелювання в декілька потоків.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблиці 1 наведені результати вимірювання швидкодії класичного та запропонованих методів множення великих цілих чисел.

Таблиця 1. Результати вимірювання швидкодії методів множення цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit				*w*=64 bit			
	M, mks	M1, mks	M2, mks	M3, mks	M, mks	M1, mks	M2, mks	M3, mks
128	0,0433	0,0412	0,5712	1,4621	0,0062	0,0050	0,4262	0,8587
256	0,2035	0,1940	0,5570	0,8595	0,0341	0,0317	0,3957	0,8314
512	0,7798	0,6397	0,8712	1,0135	0,0998	0,0964	0,5143	1,0031
1024	2,6863	2,2962	1,3817	1,3387	0,3564	0,3410	0,6251	0,9239
2048	10,3686	6,8130	3,4577	1,9506	1,1903	0,9178	1,0009	1,8508
3072	20,1915	13,1452	10,0810	5,6073	2,5475	2,3619	1,5440	2,1284
4096	33,3949	22,1509	11,5057	4,5503	4,3594	4,1176	2,4295	2,7558
6144	71,2978	50,0354	25,5951	9,4136	9,2764	8,3658	5,0936	4,8502
8192	125,7330	103,2270	44,3257	15,1626	15,9832	14,3656	8,1988	5,0742
12288	277,5920	231,0530	98,2396	54,9544	34,5814	33,3941	16,4874	10,0832
16384	492,6570	404,4440	171,9720	79,5962	60,4897	58,2752	28,8720	11,8643

Для оцінки ефективності запропонованих методів для платформи Server, наводиться відношення результатів вимірювання класичного методу множення до результатів вимірювання запропонованих методів множення. У таблиці 2 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод множення без розпаралелювання ефективніший в 1,051-1,536 раз для чисел на всіх довжинах для *w*=32 bit, та в 1,036-1,297 раз для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в 2 потоки ефективніший в 1,944-2,999 раз для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,189-2,097 раз для чисел на довжинах 2048-16384 біт для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в декілька потоків ефективніший в 2,007-8,292 раз для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,197-5,098 раз для чисел на довжинах 3072-16384 біт для *w*=64 bit.

Таблиця 2. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit			*w*=64 bit		
	M/M1	M/M2	M/M3	M/M1	M/M2	M/M3
128	1,051	0,076	0,030	1,238	0,015	0,007
256	1,049	0,365	0,237	1,077	0,086	0,041
512	1,219	0,895	0,769	1,036	0,194	0,099
1024	1,170	1,944	2,007	1,045	0,570	0,386
2048	1,522	2,999	5,316	1,297	1,189	0,643
3072	1,536	2,003	3,601	1,079	1,650	1,197
4096	1,508	2,902	7,339	1,059	1,794	1,582
6144	1,425	2,786	7,574	1,109	1,821	1,913
8192	1,218	2,837	8,292	1,113	1,949	3,150
12288	1,201	2,826	5,051	1,036	2,097	3,430
16384	1,218	2,865	6,189	1,038	2,095	5,098

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ алгоритму Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 3072 bit (*w*=32 bit) та 2048 bit (*w*=64 bit).

У таблиці 3 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при розмірі множників 1024 bit і вище для *w*=32 bit, та для чисел 3072 bit і вище для *w*=64 bit. Так, метод множення цілих чисел з розпаралелюванням у 2 потоки, ефективніший ніж без розпаралелювання в 1,3-2,35 рази для *w*=32 bit, та в 1,53-2,03 рази для *w*=64 bit. Метод множення цілих чисел з розпаралелюванням у декілька потоків ефективніший ніж без розпаралелювання в 1,72-6,81 рази для *w*=32 bit, та в 1,11-4,91 рази для *w*=64 bit.

Таблиця 3. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини з розпаралелюванням для платформи Server

Size, bit	*w*=32 bit		*w*=64 bit	
	M1/M2	M1/M3	M1/M2	M1/M3
128	0,07	0,03	0,01	0,01
256	0,35	0,23	0,08	0,04
512	0,73	0,63	0,19	0,10
1024	1,66	1,72	0,55	0,37
2048	1,97	3,49	0,92	0,50
3072	1,30	2,34	1,53	1,11
4096	1,93	4,87	1,69	1,49
6144	1,95	5,32	1,64	1,72
8192	2,33	6,81	1,75	2,83
12288	2,35	4,20	2,03	3,31
16384	2,35	5,08	2,02	4,91

Наступними розглядаються методи піднесення до квадрату.

У таблиці 4 наведені результати вимірювання швидкодії класичного та запропонованих методів піднесення до квадрату великих цілих чисел.

Таблиця 4. Результати вимірювання швидкодії методів піднесення до квадрату цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit				*w*=64 bit			
	S, mks	S1, mks	S2, mks	S3, mks	S, mks	S1, mks	S2, mks	S3, mks
128	0,0331	0,0264	0,5468	1,0644	0,0053	0,0043	0,4115	0,5636
256	0,1610	0,1286	0,5269	0,8088	0,0315	0,0281	0,3870	0,6453
512	0,5777	0,4112	0,7413	1,1552	0,0875	0,0784	0,4519	1,0746
1024	2,1249	1,5287	1,2582	1,3976	0,2891	0,2155	0,5805	1,2732
2048	6,1025	4,2260	2,6598	2,1281	0,7458	0,7088	0,8143	1,7358
3072	16,7274	9,0500	7,6893	5,0734	1,6666	1,5932	1,1532	2,1037
4096	37,5931	15,6106	9,0625	4,2143	3,0515	2,8889	2,0834	2,7887
6144	103,8620	36,2885	20,6796	7,6432	7,0036	6,6347	3,8397	2,7352
8192	190,8310	63,1693	34,7147	13,0181	12,2147	11,9123	5,9681	5,0262
12288	438,1680	137,8060	88,9007	36,2353	27,0522	26,3167	14,9337	11,9995
16384	772,5420	242,3670	134,4420	44,6671	48,0219	44,2396	25,0072	18,4340

Для оцінки ефективності запропонованих методів для платформи Server, наводиться відношення результатів вимірювання класичного методу піднесення до квадрату до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 5 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод піднесення до квадрату без використання розпаралелювання ефективніший в 1,252-3,187 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,025-1,341 рази для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в 2 потоки ефективніший в 1,689-5,746 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,445-2,047 рази для чисел на довжинах 3072-16384 біт для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в декілька потоків ефективніший в 1,52-17,296 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,094-2,605 рази для чисел на довжинах 4096-16384 біт для *w*=64 bit.

Таблиця 5. Нормалізовані результати вимірювання швидкодії методів піднесення до квадрату цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit			*w*=64 bit		
	S/S1	S/S2	S/S3	S/S1	S/S2	S/S3
128	1,254	0,061	0,031	1,215	0,013	0,009
256	1,252	0,306	0,199	1,122	0,081	0,049
512	1,405	0,779	0,500	1,117	0,194	0,081
1024	1,390	1,689	1,520	1,341	0,498	0,227
2048	1,444	2,294	2,868	1,052	0,916	0,430
3072	1,848	2,175	3,297	1,046	1,445	0,792
4096	2,408	4,148	8,920	1,056	1,465	1,094
6144	2,862	5,022	13,589	1,056	1,824	2,561
8192	3,021	5,497	14,659	1,025	2,047	2,430
12288	3,180	4,929	12,092	1,028	1,811	2,254
16384	3,187	5,746	17,296	1,085	1,920	2,605

Запропонований метод без розпаралелювання показує свою ефективність, із збільшенням розміру множників, що відповідає основним положенням теорії складності алгоритмів для піднесення до квадрату алгоритмом Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024 bit і вище (*w*=32 bit) та 3072 і вище bit (*w*=64 bit).

У таблиці 6 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при розмірі множників 1024 bit і вище для *w*=32 bit, для чисел 2048 bit у 2 потоки та 3072 bit і вище для *w*=64 bit. Так запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,18-1,82 рази для *w*=32 bit, та в 1,38-2,00 рази для *w*=64 bit. Метод піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,09-5,43 рази для *w*=32 bit, та в 1,04-2,43 рази для *w*=64 bit.

Таблиця 6. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини з розпаралелюванням платформи для Server

Розмір, біт	*w*=32 bit		*w*=64 bit	
	S1/S2	S1/S3	S1/S2	S1/S3
128	0,05	0,02	0,01	0,01
256	0,24	0,16	0,07	0,04
512	0,55	0,36	0,17	0,07
1024	1,21	1,09	0,37	0,17
2048	1,59	1,99	0,87	0,41
3072	1,18	1,78	1,38	0,76
4096	1,72	3,70	1,39	1,04
6144	1,75	4,75	1,73	2,43
8192	1,82	4,85	2,00	2,37
12288	1,55	3,80	1,76	2,19
16384	1,80	5,43	1,77	2,40

Наступними розглядаються методи приведення за модулем.

У таблиці 7 наведені результати вимірювання швидкодії класичного методу та запропонованих методів приведення за модулем.

Таблиця 7. Результати вимірювання швидкодії операцій приведення за модулем цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit				*w*=64 bit			
	R, mks	R1, mks	R2, mks	R3, mks	R, mks	R1, mks	R2, mks	R3, mks
128	0,1800	0,1600	2,1000	3,0200	0,0440	0,0410	0,8960	1,2920
256	0,5100	0,5000	2,0300	1,8500	0,1240	0,1120	0,8440	1,3900
512	2,7500	2,3600	2,3200	2,2500	0,5170	0,5120	1,0080	1,6130
1024	5,9700	5,2600	3,8200	2,8100	1,1350	0,7210	1,0900	1,9000
2048	27,8400	24,8500	15,9200	6,5300	4,5410	4,4630	4,2650	2,6490
3072	40,8600	34,5600	27,5400	14,2700	6,4600	5,9270	4,9690	4,9350
4096	109,1800	93,0500	57,3800	21,7900	17,1580	15,8420	9,7460	9,7900
6144	150,4900	134,1200	101,9100	33,5500	25,0710	21,4870	17,0900	6,4790
8192	265,5500	237,2900	179,2500	66,5300	41,7130	39,6170	29,7870	14,7100
12288	693,1600	606,7400	443,2000	134,8300	107,0480	94,8020	67,8440	29,8400
16384	1078,5600	912,9800	719,7300	210,9400	158,8910	138,1980	109,3890	43,1680

Для оцінки ефективності запропонованих методів для платформи Server, наводиться відношення результатів вимірювання класичного методу приведення за модулем до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 8 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод приведення за модулем без розпаралелювання ефективніший в 1,02-1,182 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,01-1,574 рази для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в 2 потоки ефективніший в 1,477-1,903 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,041-1,761 рази для чисел на довжинах 1024-16384 біт для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в декілька потоків ефективніший в 1,222-5,141 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,309-3,870 рази для чисел на довжинах 2048-16384 біт для *w*=64 bit.

Таблиця 8. Нормалізовані результати вимірювання швидкодії операцій приведення за модулем цілих чисел різної довжини для платформи Server

Розмір, біт	*w*=32 bit			*w*=64 bit		
	R/R1	R/R2	R/R3	R/R1	R/R2	R/R3
128	1,125	0,086	0,060	1,073	0,049	0,034
256	1,020	0,251	0,276	1,107	0,147	0,089
512	1,165	1,185	1,222	1,010	0,513	0,321
1024	1,135	1,563	2,125	1,574	1,041	0,597
2048	1,120	1,749	4,263	1,017	1,065	1,714
3072	1,182	1,484	2,863	1,090	1,300	1,309
4096	1,173	1,903	5,011	1,083	1,761	1,753
6144	1,122	1,477	4,486	1,167	1,467	3,870
8192	1,119	1,481	3,991	1,053	1,400	2,836
12288	1,142	1,564	5,141	1,129	1,578	3,587
16384	1,181	1,499	5,113	1,150	1,453	3,681

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ алгоритму Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 512 bit (*w*=32 bit) та 2048 bit (*w*=64 bit).

У таблиці 9 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при приведенні за модулем чисел 512 біт і вище для *w*=32 bit, та для чисел 2048 біт і вище для *w*=64 bit. Так метод приведення за модулем з розпаралелюванням в 2 потоки ефективніший методу приведення за модулем без розпаралелювання в 1,02-1,62 рази для *w*=32 bit, та в 1,05-1,63 рази для *w*=64 bit. Метод приведення за модулем цілих чисел з розпаралелюванням в декілька потоків ефективніший метод приведення за модулем без розпаралелювання в 1,05-4,5 рази для *w*=32 bit, та в 1,2-3,32 рази для *w*=64 bit.

Таблиця 9. Нормалізовані результати вимірювання швидкодії операцій приведення за модулем цілих чисел різної довжини з розпаралелюванням для платформи Server

Розмір, біт	*w*=32 bit		*w*=64 bit	
	R1/R2	R1/R3	R1/R2	R1/R3
128	0,08	0,05	0,05	0,03
256	0,25	0,27	0,13	0,08
512	1,02	1,05	0,51	0,32
1024	1,38	1,87	0,66	0,38
2048	1,56	3,81	1,05	1,68
3072	1,25	2,42	1,19	1,20
4096	1,62	4,27	1,63	1,62
6144	1,32	4,00	1,26	3,32
8192	1,32	3,57	1,33	2,69
12288	1,37	4,50	1,40	3,18
16384	1,27	4,33	1,26	3,20

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблиці 10 наведені результати вимірювання швидкодії класичного методу та запропонованих методів множення цілих чисел для платформи Desktop

Таблиця 10. Результати вимірювання швидкодії методів множення цілих чисел різної довжини для платформи Desktop

Розмір, біт	*w*=32 bit				*w*=64 bit			
	M, mks	M1, mks	M2, mks	M3, mks	M, mks	M1, mks	M2, mks	M3, mks
128	0,06	0,0538	0,6689	0,944	0,03421	0,01499	0,72536	0,70535
256	0,3047	0,2792	0,7814	1,0455	0,04042	0,03353	0,58535	0,81446
512	1,0946	0,9309	1,0843	1,5171	0,14075	0,11462	0,64302	1,13423
1024	4,2655	2,9475	1,7616	1,5589	0,51158	0,31657	0,65398	0,91057
2048	14,3421	9,2731	5,0842	2,9738	1,68574	1,14209	1,25904	1,60695
3072	27,7522	20,9022	10,0833	5,1727	3,34262	2,57715	1,94558	1,71916
4096	46,6316	28,5464	26,8204	8,3915	5,76244	4,66369	2,95498	4,00067
6144	98,2265	63,2237	40,6339	18,4219	12,2628	10,2838	5,73828	4,54305
8192	172,239	144,597	71,1195	28,4935	20,833	17,8901	9,49562	5,77928
12288	384,556	322	120,238	65,1565	45,8408	39,3488	20,5492	11,2127
16384	680,638	568,662	209,935	111,932	80,7857	69,104	35,2883	20,0633

Для оцінки ефективності запропонованих методів для платформи Desktop, наводиться відношення результатів вимірювання класичного методу множення до результатів вимірювання запропонованих методів множення.

У таблиці 11 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод множення ефективніший в 1,09-1,63 раз для чисел на всіх довжинах для *w*=32 bit, та в 1,16-2,28 раз для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в 2 потоки ефективніший в 1,01-3,24 раз для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,34-2,29 раз для чисел на довжинах 2048-16384 біт для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в декілька потоків ефективніший в 2,74-6,08 раз для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,05-4,09 раз для чисел на довжинах 2048-16384 біт для *w*=64 bit.

Таблиця 11. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини для платформи Desktop

Розмір, біт	*w*=32 bit			*w*=64 bit		
	M/M1	M/M2	M/M3	M/M1	M/M2	M/M3
128	1,12	0,09	0,06	2,28	0,05	0,05
256	1,09	0,39	0,29	1,21	0,07	0,05
512	1,18	1,01	0,72	1,23	0,22	0,12
1024	1,45	2,42	2,74	1,62	0,78	0,56
2048	1,55	2,82	4,82	1,48	1,34	1,05
3072	1,33	2,75	5,37	1,30	1,72	1,94
4096	1,63	1,74	5,56	1,24	1,95	1,44
6144	1,55	2,42	5,33	1,19	2,14	2,70
8192	1,19	2,42	6,04	1,16	2,19	3,60
12288	1,19	3,20	5,90	1,16	2,23	4,09
16384	1,20	3,24	6,08	1,17	2,29	4,03

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ алгоритму Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024-6144 bit (*w*=32 bit) та 512-4096 bit (*w*=64 bit).

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при множенні чисел 1024 біт і вище для *w*=32 bit, та для чисел 3072 біт і вище для *w*=64 bit. Так метод множення цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу множення без розпаралелювання в 1,06-2,71 рази для *w*=32 bit, та в 1,32-1,96 рази для *w*=64 bit. Метод множення цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу множення без розпаралелювання в 1,89-5,08 рази для *w*=32 bit, та в 1,17-3,51 рази для *w*=64 bit.

Таблиця 12. Нормалізовані результати вимірювання швидкодії методів множення цілих чисел різної довжини з розпаралелюванням для платформи Desktop

Розмір, біт	*w*=32 bit		*w*=64 bit	
	M1/M2	M1/M3	M1/M2	M1/M3
128	0,08	0,06	0,02	0,02
256	0,36	0,27	0,06	0,04
512	0,86	0,61	0,18	0,10
1024	1,67	1,89	0,48	0,35
2048	1,82	3,12	0,91	0,71
3072	2,07	4,04	1,32	1,50
4096	1,06	3,40	1,58	1,17
6144	1,56	3,43	1,79	2,26
8192	2,03	5,07	1,88	3,10
12288	2,68	4,94	1,91	3,51
16384	2,71	5,08	1,96	3,44

Наступними розглядаються методи піднесення до квадрату.

У таблиці 13 наведені результати вимірювання швидкодії класичного та запропонованих методів піднесення до квадрату.

Таблиця 13. Результати вимірювання швидкодії методів піднесення до квадрату цілих чисел різної довжини для платформи Desktop

Розмір, біт	*w*=32 bit				*w*=64 bit			
	S, mks	S1, mks	S2, mks	S3, mks	S, mks	S1, mks	S2, mks	S3, mks
128	0,046	0,0343	0,634	0,856	0,02506	0,00897	0,55723	0,67572
256	0,23	0,1725	0,72	0,9815	0,03351	0,02861	0,56952	0,73887
512	0,8821	0,5749	0,8731	1,1013	0,10179	0,09234	0,60131	0,88059
1024	3,0817	1,4443	1,4912	1,5113	0,34831	0,23158	0,58944	0,87916
2048	8,3602	5,4645	3,5418	3,464	0,90723	0,87272	1,00332	1,21454
3072	21,3011	12,3531	7,2833	5,243	1,94246	1,92824	1,64542	1,75009
4096	47,7614	22,6672	21,9526	8,3945	3,7774	3,70887	2,48883	3,37647
6144	122,481	49,0209	28,0932	16,5694	8,01811	7,91472	4,9419	3,06127
8192	233,179	84,9037	49,7668	28,5396	13,9989	13,8989	8,22193	4,93423
12288	518,904	190,62	109,355	62,454	31,1292	30,9128	16,3889	9,41439
16384	917,64	335,213	192,783	105,262	54,7463	54,235	28,3392	19,35

Для оцінки ефективності запропонованих методів для платформи Desktop, наводиться відношення результатів вимірювання класичного методу піднесення до квадрату до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод піднесення до квадрату без розпаралелювання ефективніший в 1,33-2,75 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,01-2,79 рази для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в 2 потоки ефективніший в 1,01-4,76 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,18-1,93 рази для чисел на довжинах 3072-16384 біт для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в декілька потоків ефективніший в 2,04-8,72 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,11-3,31 рази для чисел на довжинах 3072-16384 біт для *w*=64 bit.

Таблиця 14. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини для платформи Desktop

Розмір, біт	*w*=32 bit			*w*=64 bit		
	S/S1	S/S2	S/S3	S/S1	S/S2	S/S3
128	1,34	0,07	0,05	2,79	0,04	0,04
256	1,33	0,32	0,23	1,17	0,06	0,05
512	1,53	1,01	0,80	1,10	0,17	0,12
1024	2,13	2,07	2,04	1,50	0,59	0,40
2048	1,53	2,36	2,41	1,04	0,90	0,75
3072	1,72	2,92	4,06	1,01	1,18	1,11
4096	2,11	2,18	5,69	1,02	1,52	1,12
6144	2,50	4,36	7,39	1,01	1,62	2,62
8192	2,75	4,69	8,17	1,01	1,70	2,84
12288	2,72	4,75	8,31	1,01	1,90	3,31
16384	2,74	4,76	8,72	1,01	1,93	2,83

Запропонований метод без розпаралелювання показує свою ефективність, із збільшенням розміру множників, що відповідає основним положенням теорії складності алгоритмів для піднесення до квадрату методом Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024 bit (*w*=32 bit) та 3072 і вище bit (*w*=64 bit).

У таблиці 15 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при піднесенні до квадрату чисел 1024 біт і вище для *w*=32 bit, та для чисел 3072 біт і вище для *w*=64 bit. Так запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,03-1,74 рази для *w*=32 bit, та в 1,17-1,91 рази для *w*=64 bit. Запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,58-3,18 рази для *w*=32 bit, та в 1,10-3,28 рази для *w*=64 bit.

Таблиця 15. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини з розпаралелюванням для платформи Desktop

Розмір, біт	*w*=32 bit		*w*=64 bit	
	S1/S2	S1/S3	S1/S2	S1/S3
128	0,05	0,04	0,02	0,01
256	0,24	0,18	0,05	0,04
512	0,66	0,52	0,15	0,10
1024	0,97	0,96	0,39	0,26
2048	1,54	1,58	0,87	0,72
3072	1,70	2,36	1,17	1,10
4096	1,03	2,70	1,49	1,10
6144	1,74	2,96	1,60	2,59
8192	1,71	2,97	1,69	2,82
12288	1,74	3,05	1,89	3,28
16384	1,74	3,18	1,91	2,80

Наступними розглядаються методи приведення за модулем.

У таблиці 16 наведені результати вимірювання швидкодії класичного та запропонованих методів приведення за модулем.

Таблиця 16. Результати вимірювання швидкодії методів приведення за модулем цілих чисел різної довжини для платформи Desktop

Розмір біт	*w*=32 bit				*w*=64 bit			
	R, mks	R1, mks	R2, mks	R3, mks	R, mks	R1, mks	R2, mks	R3, mks
128	0,29	0,25	1,72	1,81	0,254	0,1	1,189	1,389
256	0,71	0,67	2,1	2,39	0,158	0,13	1,136	1,612
512	3,26	2,97	2,24	2,48	0,578	0,493	1,36	1,625
1024	8,42	5,73	5,22	3,77	1,458	0,827	1,478	1,993
2048	35,63	32,14	21,48	13,23	5,603	4,785	3,754	3,223
3072	51,07	45,84	36,53	16,88	7,7	6,681	5,982	4,229
4096	141,86	133,09	112,65	39,21	20,585	17,997	12,557	8,328
6144	198,3	179,1	139,7	56,01	31,144	25,817	20,756	11,302
8192	348,4	316,12	247,12	103,83	50,566	45,278	36,496	19,287
12288	911,85	815,55	612,16	251,77	128,41	112,941	87,607	46,466
16384	1375,8	1248,93	984,32	380,79	197,29	174,487	138,279	66,558

Для оцінки ефективності запропонованих методів для платформи Desktop, наводиться відношення результатів вимірювання класичного методу приведення за модулем до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 17 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод приведення за модулем ефективніший в 1,06-1,47 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,12-2,54 рази для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в 2 потоки ефективніший в 1,26-1,66 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,29-1,64 рази для чисел на довжинах 2048-16384 біт для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в декілька потоків ефективніший в 1,31-3,62 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,74-2,96 рази для чисел на довжинах 2048-16384 біт для *w*=64 bit.

Таблиця 17. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини для платформи Desktop

Розмір, біт	*w*=32 bit			*w*=64 bit		
	R/R1	R/R2	R/R3	R/R1	R/R2	R/R3
128	1,16	0,17	0,16	2,54	0,21	0,18
256	1,06	0,34	0,30	1,22	0,14	0,10
512	1,10	1,46	1,31	1,17	0,43	0,36
1024	1,47	1,61	2,23	1,76	0,99	0,73
2048	1,11	1,66	2,69	1,17	1,49	1,74
3072	1,11	1,40	3,03	1,15	1,29	1,82
4096	1,07	1,26	3,62	1,14	1,64	2,47
6144	1,11	1,42	3,54	1,21	1,50	2,76
8192	1,10	1,41	3,36	1,12	1,39	2,62
12288	1,12	1,49	3,62	1,14	1,47	2,76
16384	1,10	1,40	3,61	1,13	1,43	2,96

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ алгоритму Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024 bit (*w*=32 bit) та 6144 bit (*w*=64 bit).

У таблиці 18 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при приведенні за модулем чисел 512 біт та вище для *w*=32 bit, та для чисел 2048 біт та вище для *w*=64 bit. Так запропонований метод приведення за модулем з розпаралелюванням в 2 потоки ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,10-1,50 рази для *w*=32 bit, та в 1,12-1,43 рази для *w*=64 bit. Запропонований метод приведення за модулем з розпаралелюванням в декілька потоків ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,20-3,39 рази для *w*=32 bit, та в 1,48-2,62 рази для *w*=64 bit.

Таблиця 18. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини з розпаралелюванням для платформи Desktop

Розмір, біт	*w*=32 bit		*w*=64 bit	
	R1/R2	R1/R3	R1/R2	R1/R3
128	0,15	0,14	0,08	0,07
256	0,32	0,28	0,11	0,08
512	1,33	1,20	0,36	0,30
1024	1,10	1,52	0,56	0,41
2048	1,50	2,43	1,27	1,48
3072	1,25	2,72	1,12	1,58
4096	1,18	3,39	1,43	2,16
6144	1,28	3,20	1,24	2,28
8192	1,28	3,04	1,24	2,35
12288	1,33	3,24	1,29	2,43
16384	1,27	3,28	1,26	2,62

Далі будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

Значення швидкодії методів множення цілих чисел показані в мікросекундах.

У таблиці 19 наведені результати вимірювання швидкодії класичного та запропонованих методів множення.

Таблиця 19. Результати вимірювання швидкодії методів множення цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit				*w*=64 bit			
	M, mks	M1, mks	M2, mks	M3, mks	M, mks	M1, mks	M2, mks	M3, mks
128	0,0962	0,0566	1,2371	2,9214	0,0079	0,00679	1,21733	2,40316
256	0,3476	0,3235	1,242	2,7378	0,05072	0,03945	1,11551	2,54829
512	1,2463	1,0687	1,7871	2,5977	0,1741	0,13739	1,12263	2,5847
1024	4,7734	2,8703	2,4489	3,5643	0,65737	0,29575	1,18748	2,75091
2048	14,8952	9,0018	5,2481	4,9458	1,809	1,138	1,69663	2,81423
3072	27,1891	17,629	10,7139	7,5293	3,49665	2,62282	2,55904	3,75165
4096	43,3973	28,5667	15,769	10,6991	5,63828	4,74649	3,91142	4,03307
6144	93,3871	65,6517	35,8354	19,7165	11,7804	10,3728	6,51441	6,02522
8192	162,081	137,886	62,0733	31,4253	19,9636	18,0659	10,638	8,55332
12288	361,451	305,323	131,288	87,5561	43,2279	39,9294	23,5013	13,5349
16384	639,46	536,241	231,29	128,622	76,0651	70,4005	40,3448	25,4767

Для оцінки ефективності запропонованих методів для платформи Mobile, наводиться відношення результатів вимірювання класичного методу множення до результатів вимірювання запропонованих методів множення.

У таблиці 20 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод множення без розпаралелювання ефективніший в 1,07-1,70 раз для чисел на всіх довжинах для *w*=32 bit, та в 1,08-2,22 раз для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в 2 потоки ефективніший в 1,95-2,84 раз для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,07-1,89 раз для чисел на довжинах 2048-16384 біт для *w*=64 bit. Запропонований метод множення з використанням розпаралелювання в декілька потоків ефективніший в 1,34-5,16 раз для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,40-3,19 раз для чисел на довжинах 4096-16384 біт для *w*=64 bit.

Таблиця 20. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit			*w*=64 bit		
	M/M1	M/M2	M/M3	M/M1	M/M2	M/M3
128	1,70	0,08	0,03	1,16	0,01	0,00
256	1,07	0,28	0,13	1,29	0,05	0,02
512	1,17	0,70	0,48	1,27	0,16	0,07
1024	1,66	1,95	1,34	2,22	0,55	0,24
2048	1,65	2,84	3,01	1,59	1,07	0,64
3072	1,54	2,54	3,61	1,33	1,37	0,93
4096	1,52	2,75	4,06	1,19	1,44	1,40
6144	1,42	2,61	4,74	1,14	1,81	1,96
8192	1,18	2,61	5,16	1,11	1,88	2,33
12288	1,18	2,75	4,13	1,08	1,84	3,19
16384	1,19	2,76	4,97	1,08	1,89	2,99

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ методу Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024-6144 bit (*w*=32 bit) та 512-4096 bit (*w*=64 bit).

У таблиці 21 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при множенні чисел 1024 біт та вище для *w*=32 bit, та для чисел 3072 біт та вище для *w*=64 bit. Так запропонований метод множення цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу множення без розпаралелювання в 1,17-2,33 рази для *w*=32 bit, та в 1,02-1,74 рази для *w*=64 bit. Запропонований метод множення цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу множення без розпаралелювання в 1,82-4,39 рази для *w*=32 bit, та в 1,18-2,95 рази для *w*=64 bit.

Таблиця 21. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини з розпаралелюванням для платформи Mobile

Розмір, біт	*w*=32 bit		*w*=64 bit	
	M1/M2	M1/M3	M1/M2	M1/M3
128	0,05	0,02	0,01	0,00
256	0,26	0,12	0,04	0,02
512	0,60	0,41	0,12	0,05
1024	1,17	0,81	0,25	0,11
2048	1,72	1,82	0,67	0,40
3072	1,65	2,34	1,02	0,70
4096	1,81	2,67	1,21	1,18
6144	1,83	3,33	1,59	1,72
8192	2,22	4,39	1,70	2,11
12288	2,33	3,49	1,70	2,95
16384	2,32	4,17	1,74	2,76

Наступними розглядаються методи піднесення до квадрату.

У таблиці 22 наведені результати вимірювання швидкодії класичного та запропонованих методів піднесення до квадрату.

Таблиця 22. Результати вимірювання швидкодії методів піднесення до квадрату цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit				*w*=64 bit			
	S, mks	S1, mks	S2, mks	S3, mks	S, mks	S1, mks	S2, mks	S3, mks
128	0,0577	0,0344	1,3836	3,4539	0,00614	0,00517	1,08704	2,42378
256	0,279	0,2098	1,1857	2,7405	0,04574	0,03201	1,11348	2,47482
512	0,9911	0,6993	1,3618	3,215	0,12955	0,12005	1,0783	2,59055
1024	3,7487	1,4162	2,0634	2,8707	0,45295	0,22639	1,16631	2,76839
2048	8,1242	5,2824	4,2176	4,4081	0,99124	0,82177	1,42842	3,06647
3072	21,1965	12,2794	7,898	5,9454	2,01815	1,79367	2,14603	3,18871
4096	50,3543	21,0494	13,2013	9,8958	3,74277	3,35778	2,90158	3,30879
6144	136,601	46,7089	28,6512	16,2983	8,58174	7,66512	5,12558	5,11907
8192	247,165	83,4219	47,9396	26,2635	15,1168	13,5612	8,18625	8,03774
12288	569,018	181,366	120,378	57,7895	33,6636	30,3107	16,9326	11,1469
16384	1012,53	319,972	183,91	112,027	59,1451	52,9614	30,1182	28,5855

Для оцінки ефективності запропонованих методів для платформи Mobile, наводиться відношення результатів вимірювання класичного методу піднесення до квадрату до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 23 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод піднесення до квадрату без розпаралелювання ефективніший в 1,33-3,16 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,08-2,00 рази для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в 2 потоки ефективніший в 1,82-5,15 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,29-1,99 рази для чисел на довжинах 3072-16384 біт для *w*=64 bit. Запропонований метод піднесення до квадрату з використанням розпаралелювання в декілька потоків ефективніший в 1,31-9,85 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,13-3,02 рази для чисел на довжинах 4096-16384 біт для *w*=64 bit.

Таблиця 23. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрата цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit			*w*=64 bit		
	S/S1	S/S2	S/S3	S/S1	S/S2	S/S3
128	1,68	0,04	0,02	1,19	0,01	0,00
256	1,33	0,24	0,10	1,43	0,04	0,02
512	1,42	0,73	0,31	1,08	0,12	0,05
1024	2,65	1,82	1,31	2,00	0,39	0,16
2048	1,54	1,93	1,84	1,21	0,69	0,32
3072	1,73	2,68	3,57	1,13	0,94	0,63
4096	2,39	3,81	5,09	1,11	1,29	1,13
6144	2,92	4,77	8,38	1,12	1,67	1,68
8192	2,96	5,16	9,41	1,11	1,85	1,88
12288	3,14	4,73	9,85	1,11	1,99	3,02
16384	3,16	5,51	9,04	1,12	1,96	2,07

Запропонований метод без розпаралелювання показує свою ефективність, із збільшенням розміру множників, що відповідає основним положенням теорії складності алгоритмів для піднесення до квадрату алгоритмом Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024 bit і вище (*w*=32 bit) та 3072 і вище bit (*w*=64 bit).

У таблиці 24 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при піднесенні до квадрату чисел 2048 біт і вище для *w*=32 bit, та для чисел 4096 біт і вище для *w*=64 bit. Так запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,25-1,74 рази для *w*=32 bit, та в 1,16-1,79 рази для *w*=64 bit. Запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,20-3,18 рази для *w*=32 bit, та в 1,01-2,72 рази для *w*=64 bit.

Таблиця 24. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини з розпаралелюванням для платформи Mobile

Розмір, біт	*w*=32 bit		*w*=64 bit	
	S1/S2	S1/S3	S1/S2	S1/S3
128	0,02	0,01	0,00	0,00
256	0,18	0,08	0,03	0,01
512	0,51	0,22	0,11	0,05
1024	0,69	0,49	0,19	0,08
2048	1,25	1,20	0,58	0,27
3072	1,55	2,07	0,84	0,56
4096	1,59	2,13	1,16	1,01
6144	1,63	2,87	1,50	1,50
8192	1,74	3,18	1,66	1,69
12288	1,51	3,14	1,79	2,72
16384	1,74	2,86	1,76	1,85

Наступними розглядаються методи приведення за модулем.

У таблиці 25 наведені результати вимірювання швидкодії класичного та запропонованих методів приведення за модулем.

Таблиця 25. Результати вимірювання швидкодії методів приведення за модулем цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit				*w*=64 bit			
	R, mks	R1, mks	R2, mks	R3, mks	R, mks	R1, mks	R2, mks	R3, mks
128	0,29	0,24	2,78	7,55	0,058	0,053	2,241	4,877
256	0,92	0,87	3,14	5,17	0,21	0,147	2,231	4,918
512	4,31	3,93	3,53	5,71	0,668	0,641	2,28	4,777
1024	10,57	5,87	5,04	6,58	1,012	0,846	2,307	4,998
2048	34,05	29,97	20,25	16,42	5,936	4,759	4,165	6,02
3072	64,11	45,94	36,6	20,1	7,678	6,543	5,905	12,53
4096	136,32	127,89	78	47,38	21,362	18,038	13,001	11,35
6144	194,82	174,79	140,49	70,02	30,515	25,509	22,364	14,977
8192	366,33	320,9	240	110,93	57,895	44,622	38,333	23,109
12288	887,95	784,22	587,87	295,42	132,84	120,047	91,334	52,624
16384	1366,92	1197,05	952,67	403,9	204,021	172,346	146,282	76,527

Для оцінки ефективності запропонованих методів для платформи Mobile, наводиться відношення результатів вимірювання класичного методу приведення за модулем до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 26 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод приведення за модулем без розпаралелювання ефективніший в 1,06-1,80 рази для чисел на всіх довжинах для *w*=32 bit, та в 1,04-1,43 рази для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в 2 потоки ефективніший в 1,22-2,10 рази для чисел на довжинах 512-16384 біт для *w*=32 bit, та в 1,30-1,64 рази для чисел на довжинах 2048-16384 біт для *w*=64 bit. Запропонований метод приведення за модулем з використанням розпаралелювання в декілька потоків ефективніший в 1,61-3,38 рази для чисел на довжинах 1024-16384 біт для *w*=32 bit, та в 1,88-2,67 рази для чисел на довжинах 4096-16384 біт для *w*=64 bit.

Таблиця 26. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини для платформи Mobile

Розмір, біт	*w*=32 bit			*w*=64 bit		
	R/R1	R/R2	R/R3	R/R1	R/R2	R/R3
128	1,21	0,10	0,04	1,09	0,03	0,01
256	1,06	0,29	0,18	1,43	0,09	0,04
512	1,10	1,22	0,75	1,04	0,29	0,14
1024	1,80	2,10	1,61	1,20	0,44	0,20
2048	1,14	1,68	2,07	1,25	1,43	0,99
3072	1,40	1,75	3,19	1,17	1,30	0,61
4096	1,07	1,75	2,88	1,18	1,64	1,88
6144	1,11	1,39	2,78	1,20	1,36	2,04
8192	1,14	1,53	3,30	1,30	1,51	2,51
12288	1,13	1,51	3,01	1,11	1,45	2,52
16384	1,14	1,43	3,38	1,18	1,39	2,67

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ методу Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 1024-8192 bit (*w*=32 bit) та 1024-8192 bit (*w*=64 bit).

У таблиці 27 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при приведенні за модулем чисел 512-2048 біт та вище для *w*=32 bit, та для чисел 2048-4096 біт та вище для *w*=64 bit. Так запропонований метод приведення за модулем з розпаралелюванням в 2 потоки ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,11-1,64 рази для *w*=32 bit, та в 1,11-1,39 рази для *w*=64 bit. Запропонований метод приведення за модулем цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,83-2,96 рази для *w*=32 bit, та в 1,59-2,28 рази для *w*=64 bit.

Таблиця 27. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини з розпаралелюванням для платформи Mobile

Розмір, біт	*w*=32 bit		*w*=64 bit	
	R1/R2	R1/R3	R1/R2	R1/R3
128	0,09	0,03	0,02	0,01
256	0,28	0,17	0,07	0,03
512	1,11	0,69	0,28	0,13
1024	1,16	0,89	0,37	0,17
2048	1,48	1,83	1,14	0,79
3072	1,26	2,29	1,11	0,52
4096	1,64	2,70	1,39	1,59
6144	1,24	2,50	1,14	1,70
8192	1,34	2,89	1,16	1,93
12288	1,33	2,65	1,31	2,28
16384	1,26	2,96	1,18	2,25

Далі будуть розглядатися результати отримані для платформи Embedded (ARMv8 Cortex-A72)

Значення швидкодії методів множення цілих чисел показані в мікросекундах.

Враховуючи особливості платформи Embedded, для вимірювань використовувалась лише програмна збірка з 64-бітними машинними словами.

У таблиці 28 наведені результати вимірювання швидкодії класичного методу та запропонованих методів множення.

Таблиця 28. Результати вимірювання швидкодії методів множення цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit			
	M, mks	M1, mks	M2, mks	M3, mks
128	0,0093	0,0047	2,8864	2,8462
256	0,0747	0,0374	2,8383	3,5573
512	0,3136	0,2988	2,9592	8,555
1024	3,2177	3,0073	7,7378	18,894
2048	12,7006	12,4872	13,4355	11,364
3072	29,1384	27,859	21,0082	15,2157
4096	20,6884	19,7107	12,846	7,9166
6144	46,2236	44,4972	24,9631	17,4918
8192	82,8074	79,5585	42,11	22,4813
12288	189,168	178,766	96,024	69,2794
16384	346,298	316,031	161,063	91,4883

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання класичного методу множення до результатів вимірювання запропонованих методів множення.

У таблиці 29 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод множення без розпаралелювання ефективніший в 1,02-2,00 раз для чисел на всіх довжинах. Запропонований метод множення з використанням розпаралелювання в 2 потоки ефективніший в 1,39-2,15 раз для чисел на довжинах 3072-16384 біт. Запропонований метод множення з використанням розпаралелювання в декілька потоків ефективніший в 1,12-3,79 раз для чисел на довжинах 2048-16384 біт.

Таблиця 29. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit		
	M/M1	M/M2	M/M3
128	1,98	0,00	0,00
256	2,00	0,03	0,02
512	1,05	0,11	0,04
1024	1,07	0,42	0,17
2048	1,02	0,95	1,12
3072	1,05	1,39	1,92
4096	1,05	1,61	2,61
6144	1,04	1,85	2,64
8192	1,04	1,97	3,68
12288	1,06	1,97	2,73
16384	1,10	2,15	3,79

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ методу Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 256 біт (*w*=64 bit).

У таблиці 30 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при множенні чисел 2072 біт та вище. Так запропонований метод множення цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу множення без розпаралелювання в 1,33-1,96 рази. Запропонований метод множення цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу множення без розпаралелювання в 1,10-3,54 рази.

Таблиця 30. Нормалізовані результати вимірювання швидкодії запропонованих методів множення цілих чисел різної довжини з розпаралелюванням для платформи Embedded

Розмір, біт	*w*=64 bit	
	M1/M2	M1/M3
128	0,00	0,00
256	0,01	0,01
512	0,10	0,03
1024	0,39	0,16
2048	0,93	1,10
3072	1,33	1,83
4096	1,53	2,49
6144	1,78	2,54
8192	1,89	3,54
12288	1,86	2,58
16384	1,96	3,45

Наступними розглядаються методи піднесення до квадрату.

У таблиці 31 наведені результати вимірювання швидкодії класичного та запропонованих методів піднесення до квадрату.

Таблиця 31. Результати вимірювання швидкодії методів піднесення до квадрату цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit			
	S, mks	S1, mks	S2, mks	S3, mks
128	0,007	0,0017	2,8817	2,836
256	0,0794	0,0233	2,8517	3,5035
512	0,2715	0,2476	7,2066	8,2982
1024	2,3522	2,1209	7,2522	8,2727
2048	9,1351	8,6481	10,6132	9,9851
3072	19,9479	18,8498	14,9003	4,664
4096	14,5659	14,1161	8,2762	9,4987
6144	31,0532	29,4231	14,9854	9,0007
8192	53,4907	51,2321	23,9381	13,6842
12288	118,55	112,192	49,737	28,0145
16384	207,37	196,526	84,2362	45,0566

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання класичного методу піднесення до квадрату до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 32 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод піднесення до квадрату без розпаралелювання ефективніший в 1,03-4,12 рази для чисел на всіх довжинах. Запропонований метод піднесення до квадрату з використанням розпаралелювання в 2 потоки ефективніший в 1,34-2,46 рази для чисел на довжинах 3072-16384 біт. Запропонований метод піднесення до квадрату з використанням розпаралелюванням в декілька потоків ефективніший в 1,53-4,60 рази для чисел на довжинах 3072-16384 біт.

Таблиця 32. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit		
	S/S1	S/S2	S/S3
128	4,12	0,00	0,00
256	3,41	0,03	0,02
512	1,10	0,04	0,03
1024	1,11	0,32	0,28
2048	1,06	0,86	0,91
3072	1,06	1,34	4,28
4096	1,03	1,76	1,53
6144	1,06	2,07	3,45
8192	1,04	2,23	3,91
12288	1,06	2,38	4,23
16384	1,06	2,46	4,60

Запропонований метод без розпаралелювання показує свою ефективність, із збільшенням розміру множників, що відповідає основним положенням теорії складності алгоритмів для піднесення до квадрату методом Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 128 та 256 біт (*w*=64 bit).

У таблиці 33 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при піднесенні до квадрату чисел 3072 біт та вище. Так запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в 2 потоки ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,27-2,33 рази. Запропонований метод піднесення до квадрату цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу піднесення до квадрату без розпаралелювання в 1,49-4,36 рази.

Таблиця 33. Нормалізовані результати вимірювання швидкодії запропонованих методів піднесення до квадрату цілих чисел різної довжини з розпаралелюванням для платформи Embedded

Розмір, біт	*w*=64 bit	
	S1/S2	S1/S3
128	0,00	0,00
256	0,01	0,01
512	0,03	0,03
1024	0,29	0,26
2048	0,81	0,87
3072	1,27	4,04
4096	1,71	1,49
6144	1,96	3,27
8192	2,14	3,74
12288	2,26	4,00
16384	2,33	4,36

Наступними розглядаються методи приведення за модулем.

У таблиці 34 наведені результати вимірювання швидкодії класичного та запропонованих методів приведення за модулем.

Таблиця 34. Результати вимірювання швидкодії методів приведення за модулем цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit			
	R, mks	R1, mks	R2, mks	R3, mks
128	0,38	0,14	6,34	9,77
256	0,99	0,33	5,93	5,92
512	4,68	1,43	14,73	19,37
1024	29,27	9,11	17,22	18,1
2048	166,44	51,46	44,92	43,28
3072	248,99	71,57	47,59	45,73
4096	259,85	81,84	56,03	55,02
6144	375,63	113,02	93,98	46,32
8192	772,64	199,77	163,99	78,83
12288	1963,34	515,83	388,52	198,48
16384	2619,85	792,65	634,81	299,05

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання класичного методу приведення за модулем до результатів вимірювання запропонованих методів піднесення до квадрату.

У таблиці 35 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Так, запропонований метод приведення за модулем без розпаралелювання ефективніший в 2,71-3,81 рази для чисел на всіх довжинах. Запропонований метод приведення за модулем з використанням розпаралелюванням в 2 потоки ефективніший в 1,70-5,23 рази для чисел на довжинах 1024-16384 біт. Запропонований метод приведення за модулем з використанням розпаралелюванням в декілька потоків ефективніший в 1,62-9,89 рази для чисел на довжинах 1024-16384 біт.

Таблиця 35. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини для платформи Embedded

Розмір, біт	*w*=64 bit		
	R/R1	R/R2	R/R3
128	2,71	0,06	0,04
256	3,00	0,17	0,17
512	3,27	0,32	0,24
1024	3,21	1,70	1,62
2048	3,23	3,71	3,85
3072	3,48	5,23	5,44
4096	3,18	4,64	4,72
6144	3,32	4,00	8,11
8192	3,87	4,71	9,80
12288	3,81	5,05	9,89
16384	3,31	4,13	8,76

Запропонований метод без розпаралелювання показує свою ефективність, однак із збільшенням розміру множників, ефективність поступово зменшується, що відповідає основним положенням теорії складності алгоритмів, коли відбувається наближення до граничного/асимптотичного значення складності $O(n^2)$ алгоритму Comba. Найбільшу ефективність, запропонований метод, показує на двійковій довжині приблизно 3072 bit (w=64 bit).

У таблиці 36 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Використання розпаралелювання доцільне при приведенні за модулем чисел 2048 біт та вище. Так запропонований метод приведення за модулем з розпаралелюванням в 2 потоки ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,15-1,50 рази. Запрпонований метод приведення за модулем цілих чисел з розпаралелюванням в декілька потоків ефективніший запропонованого методу приведення за модулем без розпаралелювання в 1,19-2,65 рази.

Таблиця 36. Нормалізовані результати вимірювання швидкодії запропонованих методів приведення за модулем цілих чисел різної довжини з розпаралелюванням для платформи Embedded

Розмір, біт	*w*=64 bit	
	R1/R2	R1/R3
128	0,02	0,01
256	0,06	0,06
512	0,10	0,07
1024	0,53	0,50
2048	1,15	1,19
3072	1,50	1,57
4096	1,46	1,49
6144	1,20	2,44
8192	1,22	2,53
12288	1,33	2,60
16384	1,25	2,65

ДОДАТОК В

Експериментальні дослідження методів арифметичних операцій в кільці цілих чисел

Експериментальні дослідження проводились для деяких значущих методів арифметичних операцій в кільці цілих чисел:

- додавання за модулем (Add);
- віднімання за модулем (Sub);
- множення за модулем (Mul);
- піднесення до квадрату за модулем (Sqr);
- піднесення до степені за модулем (Pow);

Значення швидкодії вказаних методів в кільці цілих чисел зазначені в мікросекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблиці 1 наведені результати вимірювання швидкодії класичних та запропонованих методів у кільці цілих чисел для платформи Server.

Таблиця 1. Результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Server з використанням 32-бітних машинних слів.

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0320	0,0899	0,1492	0,1879	0,3310	1,5468
Sub, mks	0,0483	0,0856	0,1712	0,2348	0,4424	0,8692
Mul, mks	1,6870	6,2730	60,5430	90,9190	361,1300	1418,6900
Sqr, mks	1,5340	5,6630	55,7010	100,4990	422,3960	1707,1600
Pow, mks	1491	11489	169006	653678	5101700	40147200
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,026	0,081	0,131	0,172	0,316	1,504
Sub, mks	0,039	0,078	0,152	0,219	0,421	0,836
Mul, mks	1,547	5,614	55,150	84,092	330,787	1284,840
Sqr, mks	1,379	5,314	54,575	78,412	312,888	1169,170
Pow, mks	943	6966	107515	373769	3059900	23898700
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0513	0,0798	0,1295	0,1616	0,3040	0,8670
Sub, mks	0,0509	0,0636	0,1460	0,2069	0,4088	0,8117
Mul, mks	2,1240	5,3970	49,8220	73,8660	288,3840	1116,5200
Sqr, mks	1,8220	5,0170	47,5600	72,2930	282,9450	1100,5400
Pow, mks	1826	6013	63067	204891	1798090	12474100
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0255	0,0552	0,1283	0,1668	0,3070	0,7648
Sub, mks	0,0415	0,0635	0,1488	0,2095	0,4102	0,8146
Mul, mks	3,2140	6,7110	53,9290	78,2920	316,8600	1254,8300
Sqr, mks	2,4930	6,6280	53,4010	77,2850	304,1540	1141,9400
Pow, mks	2134	5652	32090	146704	745938	5196770

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 2 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,03-1,23 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,04-1,22 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,08-1,12 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,46 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,57-1,75 рази для чисел на всіх довжинах.

Таблиця 2. Нормалізовані результати вимірювання швидкодії запропонованих методів у кільці цілих чисел різної довжини для платформи Server

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,23	1,11	1,14	1,09	1,05	1,03
Sub	1,22	1,09	1,13	1,07	1,05	1,04
Mul	1,09	1,12	1,10	1,08	1,09	1,10
Sqr	1,11	1,07	1,02	1,28	1,35	1,46
Pow	1,58	1,65	1,57	1,75	1,67	1,68

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 3 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,74 рази для чисел довжиною 1024-16384 біт.
- Операція віднімання за модулем ефективніша в 1,03-1,23 рази для чисел довжиною 1024-16384 біт.
- Операція множення за модулем ефективніша в 1,04-1,15 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,06-1,15 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,16-1,92 рази для чисел довжиною 1024-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,97 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,02-1,23 рази для чисел довжиною 1024-16384 біт.
- Операція множення за модулем ефективніша в 1,02-1,07 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,03 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,23-4,6 рази для чисел довжиною 1024-16384 біт.

Таблиця 3. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Server з використанням розпаралелювання у два та більше потоків

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	0,51	1,02	1,01	1,06	1,04	1,74
Sub	0,78	1,23	1,04	1,06	1,03	1,03
Mul	0,73	1,04	1,11	1,14	1,15	1,15
Sqr	0,76	1,06	1,15	1,08	1,11	1,06
Pow	0,52	1,16	1,70	1,82	1,70	1,92
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,02	1,47	1,02	1,03	1,03	1,97
Sub	0,95	1,23	1,02	1,04	1,03	1,03
Mul	0,48	0,84	1,02	1,07	1,04	1,02
Sqr	0,55	0,80	1,02	1,01	1,03	1,02
Pow	0,44	1,23	3,35	2,55	4,10	4,60

У таблиці 4 наведені результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Server з використанням 64-бітних машинних слів.

Таблиця 4. Результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Server з використанням 64-бітних машинних слів

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0181	0,0240	0,0710	0,0912	0,1261	0,1744
Sub, mks	0,0141	0,0256	0,0646	0,0632	0,0945	0,1759
Mul, mks	0,3850	0,8520	8,6610	13,5660	52,8400	206,7020
Sqr, mks	0,3700	0,7960	7,9640	12,3730	48,3340	196,5370
Pow, mks	261	1103	19394	71375	535276	4085320
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0132	0,0181	0,0478	0,0872	0,1201	0,1665
Sub, mks	0,0113	0,0192	0,0356	0,0424	0,0907	0,1713
Mul, mks	0,2540	0,8190	8,5300	13,1180	51,4020	203,3230
Sqr, mks	0,2440	0,7650	7,7740	12,0380	47,4410	190,0670
Pow, mks	167	990	19059	66891	524867	4001820
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0087	0,0175	0,0335	0,0672	0,1178	0,1623
Sub, mks	0,0110	0,0189	0,0345	0,0413	0,0884	0,1695
Mul, mks	0,7960	1,2270	7,8390	11,6110	44,8390	176,8820
Sqr, mks	0,6240	1,1250	7,7280	11,2290	44,1450	173,2630
Pow, mks	939	2073	13441	39549	275729	1989520
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0089	0,0173	0,0362	0,0859	0,1143	0,1641
Sub, mks	0,0078	0,0181	0,0310	0,0418	0,0872	0,1703
Mul, mks	1,7310	1,8600	10,4140	12,5700	49,3420	198,5590
Sqr, mks	0,9570	1,7550	10,3580	11,8730	46,8900	188,3710
Pow, mks	1617	4135	13190	32098	175348	854033

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 5 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,05-1,38 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,03-1,81 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,52 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,52 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,02-1,56 рази для чисел на всіх довжинах.

Таблиця 5. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у кільці цілих чисел різної довжини для платформи Server з використанням 64-бітних машинних слів

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,38	1,33	1,48	1,05	1,05	1,05
Sub	1,25	1,33	1,81	1,49	1,04	1,03
Mul	1,52	1,04	1,02	1,03	1,03	1,02
Sqr	1,52	1,04	1,02	1,03	1,02	1,03
Pow	1,56	1,11	1,02	1,07	1,02	1,02

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 6 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,52 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,03 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,09-1,15 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,1 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,42-2,01 рази для чисел довжиною 3072-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,48 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,44 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,04 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01 раз для чисел довжиною 4096-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,44-4,69 рази для чисел довжиною 3072-16384 біт.

Таблиця 6. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини з розпаралелюванням у два та більше потоків для платформи Server з використанням 64-бітних машинних слів

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,52	1,03	1,43	1,30	1,02	1,03
Sub	1,03	1,02	1,03	1,03	1,03	1,01
Mul	0,32	0,67	1,09	1,13	1,15	1,15
Sqr	0,39	0,68	1,01	1,07	1,07	1,10
Pow	0,18	0,48	1,42	1,69	1,90	2,01
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,48	1,04	1,32	1,02	1,05	1,01
Sub	1,44	1,06	1,15	1,01	1,04	1,01
Mul	0,15	0,44	0,82	1,04	1,04	1,02
Sqr	0,25	0,44	0,75	1,01	1,01	1,01
Pow	0,10	0,24	1,44	2,08	2,99	4,69

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблиці 7 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій в кільці цілих чисел для платформи Desktop.

Таблиця 7. Результати вимірювання швидкодії арифметичних операцій в кільці цілих чисел різної довжини для платформи Desktop з використанням 32-бітних машинних слів.

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,045805	0,139754	0,245836	0,258219	0,483663	0,954718
Sub, mks	0,046097	0,113944	0,242049	0,306092	0,604532	1,20035
Mul, mks	2,315	8,21	81,062	125,346	490,381	1939,8
Sqr, mks	2,113	7,474	76,34	129,764	546,907	2172,63
Pow, mks	1943	14246	224119	860789	6740120	53443800
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,034491	0,135574	0,236381	0,248359	0,478033	0,93456
Sub, mks	0,042976	0,084245	0,232495	0,30521	0,603786	1,19995
Mul, mks	2,048	7,782	72,445	109,434	461,154	1819,75
Sqr, mks	1,826	6,865	68,595	103,213	403,559	1587,16
Pow, mks	1338	9287	137266	494019	4759170	37294200
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,03444	0,065515	0,206246	0,246293	0,467716	0,92745
Sub, mks	0,040191	0,084141	0,228946	0,305061	0,586065	1,18587
Mul, mks	2,312	7,099	66,611	95,799	388,245	1469,17
Sqr, mks	1,907	6,669	63,923	94,456	366,762	1464,37
Pow, mks	1379	7463	87678	257952	2399550	14946600
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,033474	0,134156	0,224704	0,243768	0,459189	0,90298
Sub, mks	0,041272	0,079312	0,224834	0,305056	0,584627	1,17104
Mul, mks	2,248	7,983	70,686	103,438	417,153	1711,95
Sqr, mks	1,9	7,211	67,844	101,119	383,237	1519,64
Pow, mks	1410	8338	46351	153638	1061040	8078990

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів

арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 8 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,33 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,04-1,07 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,05-1,15 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,09-1,37 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,42-1,74 рази для чисел на всіх довжинах.

Таблиця 8. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Desktop

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,33	1,03	1,04	1,04	1,01	1,02
Sub	1,07	1,35	1,04	1,00	1,00	1,00
Mul	1,13	1,05	1,12	1,15	1,06	1,07
Sqr	1,16	1,09	1,11	1,26	1,36	1,37
Pow	1,45	1,53	1,63	1,74	1,42	1,43

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 9 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-2,07 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,07 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,09-1,24 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,03-1,10 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,24-2,50 рази для чисел довжиною 1024-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,05 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,02-1,23 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,11 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,05 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,11-4,62 рази для чисел довжиною 1024-16384 біт.

Таблиця 9. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини з використанням розпаралелювання у два та більше потоків для платформи Desktop

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,00	2,07	1,15	1,01	1,02	1,01
Sub	1,07	1,00	1,02	1,00	1,03	1,01
Mul	0,89	1,10	1,09	1,14	1,19	1,24
Sqr	0,96	1,03	1,07	1,09	1,10	1,08
Pow	0,97	1,24	1,57	1,92	1,98	2,50
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,03	1,01	1,05	1,02	1,04	1,03
Sub	1,04	1,06	1,03	1,00	1,03	1,02
Mul	0,91	0,97	1,02	1,06	1,11	1,06
Sqr	0,96	0,95	1,01	1,02	1,05	1,04
Pow	0,95	1,11	2,96	3,22	4,49	4,62

У таблиці 10 наведені результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Desktop з використанням 64-бітних машинних слів.

Таблиця 10. Результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Desktop з використанням 64-бітних машинних слів

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01863	0,021323	0,098779	0,126032	0,135638	0,233242
Sub, mks	0,01721	0,018152	0,087528	0,129457	0,137476	0,233882
Mul, mks	0,522	1,256	11,061	19,287	85,505	253,829
Sqr, mks	0,496	1,051	10,306	17,147	63,404	241,819
Pow, mks	333	1382	23379	85899	642897	4966770
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01781	0,016492	0,095928	0,110783	0,105593	0,210042
Sub, mks	0,01028	0,016617	0,078649	0,103781	0,107683	0,208159
Mul, mks	0,306	1,177	10,716	17,86	62,429	242,775
Sqr, mks	0,287	0,993	10,058	15,275	58,643	227,99
Pow, mks	189	1265	22639	80567	618250	4757970
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01028	0,015645	0,05547	0,110153	0,096787	0,208585
Sub, mks	0,01019	0,016264	0,076649	0,086629	0,101754	0,204483
Mul, mks	0,98	1,595	10,094	14,797	54,107	207,777
Sqr, mks	0,767	1,591	9,755	14,302	52,364	203,937
Pow, mks	1133	3227	17301	55556	336728	2647110
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0103	0,01599	0,084485	0,083366	0,091492	0,190251
Sub, mks	0,01467	0,022226	0,055548	0,091138	0,090781	0,201743
Mul, mks	1,583	2,162	13,785	16,648	59,227	230,925
Sqr, mks	1,198	2,123	13,62	15,154	56,375	217,181
Pow, mks	1296	4273	14932	40393	214055	1382230

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 11 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,03-1,29 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,09-1,67 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,03-1,71 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,73 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,76 рази для чисел на всіх довжинах.

Таблиця 11. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Desktop з використанням 64-бітних машинних слів

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,05	1,29	1,03	1,14	1,28	1,11
Sub	1,67	1,09	1,11	1,25	1,28	1,12
Mul	1,71	1,07	1,03	1,08	1,37	1,05
Sqr	1,73	1,06	1,02	1,12	1,08	1,06
Pow	1,76	1,09	1,03	1,07	1,04	1,04

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,73 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,20 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,06-1,21 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,03-1,12 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,31-1,84 рази для чисел довжиною 3072-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,03-1,73 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,03-1,42 рази для чисел довжиною 3072-16384 біт.
- Операція множення за модулем ефективніша в 1,05-1,07 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,05 раз для чисел довжиною 4096-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,52-3,44 рази для чисел довжиною 3072-16384 біт.

Таблиця 12. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини з розпаралелюванням для платформи Server

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,73	1,05	1,73	1,01	1,09	1,01
Sub	1,01	1,02	1,03	1,20	1,06	1,02
Mul	0,31	0,74	1,06	1,21	1,15	1,17
Sqr	0,37	0,62	1,03	1,07	1,12	1,12
Pow	0,17	0,39	1,31	1,45	1,84	1,80
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,73	1,03	1,14	1,33	1,15	1,10
Sub	0,70	0,75	1,42	1,14	1,19	1,03
Mul	0,19	0,54	0,78	1,07	1,05	1,05
Sqr	0,24	0,47	0,74	1,01	1,04	1,05
Pow	0,15	0,30	1,52	1,99	2,89	3,44

Далі будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблиці 13 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операції у кільці цілих чисел для платформи Mobile.

Таблиця 13. Результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Mobile з використанням 32-бітних машинних слів.

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,05892	0,12369	0,401756	0,420527	0,520048	0,892203
Sub, mks	0,044989	0,10528	0,337983	0,421172	0,57775	1,1306
Mul, mks	2,477	8,062	77,356	138,929	472,511	1850,55
Sqr, mks	2,383	6,957	72,286	140,947	566,536	1982,72
Pow, mks	1810	14176	217409	835688	6619760	52394200
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,035875	0,062759	0,205463	0,398071	0,436144	0,865218
Sub, mks	0,042988	0,081753	0,251422	0,410985	0,570803	1,01984
Mul, mks	2,098	7,447	74,283	120,368	421,879	1657,1
Sqr, mks	2,174	6,347	69,055	114,367	390,494	1539,01
Pow, mks	1307	8869	139527	491615	3970970	31131000
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,032932	0,058785	0,202457	0,259292	0,419748	0,81483
Sub, mks	0,042137	0,080361	0,219343	0,271504	0,560096	0,92685
Mul, mks	3,182	7,838	68,225	108,476	414,099	1493,15
Sqr, mks	3,065	7,685	65,014	106,749	380,394	1483,26
Pow, mks	3539	10514	91861	293008	2508630	16484900
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,034631	0,152369	0,224196	0,228776	0,40235	0,79833
Sub, mks	0,041769	0,126535	0,337331	0,28584	0,559663	0,91158
Mul, mks	5,755	12,374	97,217	100,539	365,692	1445,8
Sqr, mks	5,348	12,123	99,925	105,561	353,137	1372,18
Pow, mks	6885	13808	97159	196213	1414250	9199030

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,03-1,97 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,34 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,04-1,18 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,05-1,45 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,38-1,70 рази для чисел на всіх довжинах.

Таблиця 14. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у кільці цілих чисел різної довжини для платформи Mobile

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,64	1,97	1,96	1,06	1,19	1,03
Sub	1,05	1,29	1,34	1,02	1,01	1,11
Mul	1,18	1,08	1,04	1,15	1,12	1,12
Sqr	1,10	1,10	1,05	1,23	1,45	1,29
Pow	1,38	1,60	1,56	1,70	1,67	1,68

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 15 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів з розпаралелюванням. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,54 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,02-1,15 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,11 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,03-1,07 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,52-1,89 рази для чисел довжиною 1024-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,08-1,74 рази для чисел довжиною 4096-16384 біт.
- Операція віднімання за модулем ефективніша в 1,02-1,44 рази для чисел довжиною 4096-16384 біт.
- Операція множення за модулем ефективніша в 1,15-1,20 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,08-1,12 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,44-3,38 рази для чисел довжиною 3072-16384 біт.

Таблиця 15. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Mobile з використанням розпаралелювання у два та більше потоків

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,09	1,07	1,01	1,54	1,04	1,06
Sub	1,02	1,02	1,15	1,51	1,02	1,10
Mul	0,66	0,95	1,09	1,11	1,02	1,11
Sqr	0,71	0,83	1,06	1,07	1,03	1,04
Pow	0,37	0,84	1,52	1,68	1,58	1,89
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,04	0,41	0,92	1,74	1,08	1,08
Sub	1,03	0,65	0,75	1,44	1,02	1,12
Mul	0,36	0,60	0,76	1,20	1,15	1,15
Sqr	0,41	0,52	0,69	1,08	1,11	1,12
Pow	0,19	0,64	1,44	2,51	2,81	3,38

У таблиці 16 наведені результати вимірювання швидкодії арифметичних операцій в кільці цілих чисел різної довжини для платформи Mobile з використанням 64-бітних машинних слів.

Таблиця 16. Результати вимірювання швидкодії арифметичних операцій в кільці цілих чисел різної довжини для платформи Mobile з використанням 64-бітних машинних слів.

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,03062	0,01814	0,04984	0,106026	0,142845	0,253467
Sub, mks	0,02649	0,018136	0,039858	0,0906	0,112982	0,227969
Mul, mks	0,356	1,134	10,835	20,659	64,06	249,12
Sqr, mks	0,331	0,996	10,192	19,214	60,624	232,342
Pow, mks	248	1386	24132	88663	680671	5215640
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01265	0,017179	0,048608	0,10321	0,12459	0,229158
Sub, mks	0,01101	0,017542	0,036871	0,083382	0,112882	0,226301
Mul, mks	0,321	1,035	10,662	18,295	62,568	241,72
Sqr, mks	0,299	0,964	9,799	16,392	58,845	226,492
Pow, mks	212	1260	23599	87672	617532	4754230
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01031	0,017064	0,041036	0,101713	0,115226	0,224577
Sub, mks	0,00979	0,017084	0,036526	0,081403	0,112094	0,222694
Mul, mks	1,52	1,993	11,133	16,686	57,775	219,687
Sqr, mks	1,331	1,894	10,711	16,074	55,783	217,207
Pow, mks	2474	5482	25713	64028	378862	2664750
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,01875	0,030305	0,047837	0,094632	0,113234	0,225501
Sub, mks	0,01504	0,025071	0,034166	0,072948	0,110602	0,221317
Mul, mks	3,054	4,85	19,252	15,979	55,506	217,665
Sqr, mks	3,44	4,915	18,592	15,512	54,282	208,283
Pow, mks	6491	13570	37632	69872	271583	1552200

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 17 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,03-2,42 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,03-2,41 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,13 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,03-1,17 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,01-1,17 рази для чисел на всіх довжинах.

Таблиця 17. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у кільці цілих чисел різної довжини для платформи Mobile з використанням 64-бітних машинних слів.

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	2,42	1,06	1,03	1,03	1,15	1,11
Sub	2,41	1,03	1,08	1,09	1,00	1,01
Mul	1,11	1,10	1,02	1,13	1,02	1,03
Sqr	1,11	1,03	1,04	1,17	1,03	1,03
Pow	1,17	1,10	1,02	1,01	1,10	1,10

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 18 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,23 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,01-1,13 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,08-1,10 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,05 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,37-1,78 рази для чисел довжиною 4096-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,10 рази для чисел довжиною 3072-16384 біт.
- Операція віднімання за модулем ефективніша в 1,02-1,14 рази для чисел довжиною 3072-16384 біт.
- Операція множення за модулем ефективніша в 1,11-1,14 рази для чисел довжиною 4096-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,06-1,09 раз для чисел довжиною 4096-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,25-3,06 рази для чисел довжиною 4096-16384 біт.

Таблиця 18. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини з використанням розпаралелювання у два та більше потоків для платформи Mobile з використанням 64-бітних машинних слів.

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,23	1,01	1,18	1,01	1,08	1,02
Sub	1,13	1,03	1,01	1,02	1,01	1,02
Mul	0,21	0,52	0,96	1,10	1,08	1,10
Sqr	0,22	0,51	0,91	1,02	1,05	1,04
Pow	0,09	0,23	0,92	1,37	1,63	1,78
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	0,67	0,57	1,02	1,09	1,10	1,02
Sub	0,73	0,70	1,08	1,14	1,02	1,02
Mul	0,11	0,21	0,55	1,14	1,13	1,11
Sqr	0,09	0,20	0,53	1,06	1,08	1,09
Pow	0,03	0,09	0,63	1,25	2,27	3,06

Наступними будуть розглядатися результати отримані для платформи Embedded (Cortex-A72)

У таблиці 19 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у кільці цілих чисел для платформи Embedded.

Таблиця 19. Результати вимірювання швидкодії методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Embedded з використанням 32-бітних машинних слів.

Original						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0338	0,1310	0,4475	0,4267	1,0935	2,1797
Sub, mks	0,0369	0,0682	0,3445	0,3345	0,7755	1,4699
Mul, mks	1,3272	5,2807	48,5417	72,7441	286,1910	1125,6400
Sqr, mks	1,2387	4,8602	45,2554	66,9267	261,9970	1053,9700
Pow, mks	870	6321	102011	366926	3184620	24636300
With improvements						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,032	0,115	0,159	0,233	0,333	0,673
Sub, mks	0,034	0,062	0,217	0,251	0,515	1,008
Mul, mks	1,304	5,056	47,715	71,617	265,298	1056,420
Sqr, mks	1,210	4,710	43,468	64,772	255,269	1000,010
Pow, mks	764	6197	93454	345175	3027480	23688300
With improvements and 2-thread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0302	0,1051	0,1537	0,2326	0,3101	0,6357
Sub, mks	0,0328	0,0582	0,1904	0,2435	0,5053	0,9795
Mul, mks	3,9599	6,1386	44,9185	64,7387	247,4520	978,5290
Sqr, mks	3,9385	6,7942	42,7037	60,7054	228,8500	899,5390
Pow, mks	7475	14327	74962	260312	1630980	10916900
With improvements and multithread implementation						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add, mks	0,0320	0,1100	0,1544	0,2260	0,3154	0,6179
Sub, mks	0,0355	0,0611	0,2025	0,2375	0,4943	0,9495
Mul, mks	4,0638	7,1524	42,8582	59,8533	240,0470	924,2560
Sqr, mks	3,9834	6,9672	41,4671	62,1576	225,7720	889,0180
Pow, mks	6569	14142	61272	171988	838059	5915970

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел.

У таблиці 20 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,06-3,28 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,08-1,58 рази для чисел на всіх довжинах.
- Операція множення за модулем ефективніша в 1,02-1,08 рази для чисел на всіх довжинах.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,05 рази для чисел на всіх довжинах.
- Операція піднесення до степеню за модулем ефективніша в 1,02-1,14 рази для чисел на всіх довжинах.

Таблиця 20. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у кільці цілих чисел різної довжини для платформи Embedded

without improvements / with improvement						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,06	1,14	2,82	1,83	3,28	3,24
Sub	1,08	1,10	1,58	1,34	1,50	1,46
Mul	1,02	1,04	1,02	1,02	1,08	1,07
Sqr	1,02	1,03	1,04	1,03	1,03	1,05
Pow	1,14	1,02	1,09	1,06	1,05	1,04

Для оцінки ефективності запропонованих методів з розпаралелюванням для платформи Embedded, наводиться відношення результатів вимірювання запропонованих методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в кільці цілих чисел з використанням розпаралелювання.

У таблиці 21 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,74 рази для чисел довжиною 1024-16384 біт.
- Операція віднімання за модулем ефективніша в 1,03-1,23 рази для чисел довжиною 1024-16384 біт.
- Операція множення за модулем ефективніша в 1,02-1,1 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,06-1,15 рази для чисел довжиною 1024-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,16-1,92 рази для чисел довжиною 1024-16384 біт.

Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,97 рази для чисел на всіх довжинах.
- Операція віднімання за модулем ефективніша в 1,02-1,23 рази для чисел довжиною 1024-16384 біт.
- Операція множення за модулем ефективніша в 1,02-1,07 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,03 рази для чисел довжиною 3072-16384 біт.
- Операція піднесення до степеню за модулем ефективніша в 1,23-4,6 рази для чисел довжиною 1024-16384 біт.

Таблиця 21. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в кільці цілих чисел різної довжини для платформи Embedded з використанням розпаралелювання у два та більше потоків

1 thread / 2 thread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	1,06	1,10	1,03	1,00	1,07	1,06
Sub	1,04	1,07	1,14	1,03	1,02	1,03
Mul	0,33	0,82	1,06	1,11	1,07	1,08
Sqr	0,31	0,69	1,02	1,07	1,12	1,11
Pow	0,10	0,43	1,25	1,33	1,86	2,17
1 thread / multithread						
	512 bit	1024 bit	3072 bit	4096 bit	8192 bit	16384 bit
Add	0,99	1,05	1,03	1,03	1,06	1,09
Sub	0,96	1,02	1,07	1,05	1,04	1,06
Mul	0,32	0,71	1,11	1,20	1,11	1,14
Sqr	0,30	0,68	1,05	1,04	1,13	1,12
Pow	0,12	0,44	1,53	2,01	3,61	4,00

Експериментальні дослідження методів арифметичних операцій в полі простих чисел GF(p)

Експериментальні дослідження проводились для деяких значущих арифметичних операцій в кільці полі простих чисел GF(p):

- додавання за модулем (Add);
- віднімання за модулем (Sub);
- множення за модулем (Mul);
- піднесення до квадрату за модулем (Sqr);
- піднесення до степені за модулем (Pow).

Значення швидкодії вказаних операцій в кільці цілих чисел зазначені в мікросекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблиці 1 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у полі цілих чисел GF(p) для платформи Server для 32-бітних машинних слів.

Таблиця 1. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Server з використанням 32-бітних машинних слів.

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,008051	0,008912	0,009063	0,011098	0,013576
Sub, mks	0,015154	0,010778	0,016934	0,013457	0,018453
Mul, mks	0,106	0,156	0,25	0,421	0,491
Sqr, mks	0,078	0,118	0,198	0,322	0,343
Pow, mks	25,56	45,78	91,65	212,8	299,61
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,008254	0,008937	0,008957	0,011325	0,013434
Sub, mks	0,009963	0,010726	0,011021	0,013811	0,018219
Mul, mks	0,293	0,364	0,487	0,921	1,800
Sqr, mks	0,263	0,331	0,437	0,852	1,653
Pow, mks	76,43	116,35	164,73	502,17	1367,86
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,007668	0,008312	0,008623	0,010389	0,012479
Sub, mks	0,014554	0,010166	0,016167	0,012763	0,017009
Mul, mks	0,1	0,146	0,229	0,399	0,475
Sqr, mks	0,074	0,109	0,185	0,303	0,329
Pow, mks	24,05	44,22	87,74	204,76	290,61
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,007739	0,008477	0,008632	0,010608	0,012496
Sub, mks	0,009660	0,010226	0,010448	0,013007	0,017026
Mul, mks	0,288	0,350	0,469	0,887	1,730
Sqr, mks	0,258	0,321	0,416	0,798	1,559
Pow, mks	73,22	110,35	158,59	471,14	1258,93

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 2 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,05-1,09 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,04-1,08 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,04-1,08 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,07 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,03-1,09 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,04-1,08 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,07 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,06 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,04-1,09 рази для всіх полів з загальним простим модулем.

Таблиця 2. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у полі цілих чисел GF(p) різної довжини для платформи Server з використанням 32-бітних машинних слів

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,05	1,07	1,05	1,07	1,09
Sub	1,04	1,06	1,05	1,05	1,08
Mul	1,06	1,07	1,09	1,06	1,03
Sqr	1,05	1,08	1,07	1,06	1,04
Pow	1,06	1,04	1,04	1,04	1,03
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,07	1,05	1,04	1,07	1,08
Sub	1,03	1,05	1,05	1,06	1,07
Mul	1,02	1,04	1,04	1,04	1,04
Sqr	1,02	1,03	1,05	1,07	1,06
Pow	1,04	1,05	1,04	1,07	1,09

У таблиці 3 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у полі цілих чисел GF(p) для платформи Server з використанням 64-бітних машинних слів.

Таблиця 3. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Server з використанням 64-бітних машинних слів

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,003376	0,00382	0,003742	0,004524	0,009976
Sub, mks	0,003536	0,003768	0,004924	0,005618	0,007125
Mul, mks	0,02263	0,0659	0,0663	0,11993	0,1006
Sqr, mks	0,01992	0,06335	0,06432	0,10453	0,08347
Pow, mks	6,2632	20,9352	27,5688	67,2473	67,9631
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,003184	0,003699	0,003746	0,004408	0,009939
Sub, mks	0,004116	0,004965	0,004051	0,006594	0,009447
Mul, mks	0,061	0,086	0,087	0,176	0,336
Sqr, mks	0,060	0,084	0,085	0,165	0,321
Pow, mks	18,54	29,29	35,54	101,68	253,55
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,003188	0,003585	0,003595	0,004278	0,009563

Sub, mks	0,00337	0,003626	0,004715	0,005392	0,006782
Mul, mks	0,02196	0,0619	0,0628	0,11408	0,09801
Sqr, mks	0,01929	0,05977	0,06023	0,10071	0,08129
Pow, mks	6,0433	19,9328	26,6237	65,4134	65,6529
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,003136	0,003598	0,003629	0,004293	0,009555
Sub, mks	0,004046	0,004817	0,003971	0,006418	0,009240
Mul, mks	0,059	0,083	0,083	0,164	0,320
Sqr, mks	0,058	0,082	0,080	0,154	0,299
Pow, mks	18,01	28,09	32,83	94,28	223,93

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 4 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,04-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,04-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,03 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,03-1,06 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,04-1,08 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,03-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,08 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,13 рази для всіх полів з загальним простим модулем.

Таблиця 4. Нормалізовані результати вимірювання запропонованих методів швидкодії арифметичних операцій в у поля цілих чисел GF(p) різної довжини для платформи Server для 64-бітних машинних слів

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,06	1,07	1,04	1,06	1,04
Sub	1,05	1,04	1,04	1,04	1,05
Mul	1,03	1,06	1,06	1,05	1,03
Sqr	1,03	1,06	1,07	1,04	1,03
Pow	1,04	1,05	1,04	1,03	1,04
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,03	1,03	1,03	1,04
Sub	1,02	1,03	1,02	1,03	1,02
Mul	1,04	1,04	1,05	1,08	1,05
Sqr	1,04	1,03	1,05	1,07	1,08
Pow	1,03	1,04	1,08	1,08	1,13

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770).

У таблиці 5 наведені результати вимірювання швидкодії класичний та запропонованих арифметичних операцій у полі цілих чисел GF(p) для платформи Desktop.

Таблиця 5. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Desktop з використанням 32-бітних машинних слів

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0104346	0,01054374	0,01131352	0,01398828	0,01681472
Sub, mks	0,0126593	0,01995528	0,02153628	0,02756898	0,02356328
Mul, mks	0,14317	0,23625	0,35298	0,5628	0,73188
Sqr, mks	0,10197	0,13974	0,2772	0,42224	0,4662
Pow, mks	33,99	59,9536	126,5428	279,0288	464,352
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,010102	0,010451	0,011103	0,013761	0,033970
Sub, mks	0,012787	0,019936	0,021578	0,027137	0,022986
Mul, mks	0,380070	0,496080	0,653100	1,271550	2,399840
Sqr, mks	0,346080	0,445120	0,561750	1,121670	2,100800
Pow, mks	106,9449	157,353100	223,829600	716,707400	1750,3344
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,01023	0,010337	0,010984	0,013714	0,016168
Sub, mks	0,012534	0,019564	0,021114	0,026766	0,022657
Mul, mks	0,139	0,225	0,333	0,536	0,684
Sqr, mks	0,099	0,137	0,264	0,406	0,444
Pow, mks	33	56,56	119,38	258,36	414,6
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,009904	0,010348	0,010885	0,013360	0,033304
Sub, mks	0,012536	0,019545	0,021155	0,026868	0,022535
Mul, mks	0,369000	0,477000	0,622000	1,211000	2,264000
Sqr, mks	0,336000	0,428000	0,535000	1,089000	2,020000
Pow, mks	103,83	152,77	211,16	669,82	1620,68

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 6 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,03 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,01-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,02 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,03-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,06 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,05 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,12 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,08 рази для всіх полів з загальним простим модулем.

Таблиця 6. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у полі цілих чисел GF(p) різної довжини для платформи Desktop

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,02	1,03	1,02	1,04
Sub	1,01	1,02	1,02	1,03	1,04
Mul	1,03	1,05	1,06	1,05	1,07
Sqr	1,03	1,02	1,05	1,04	1,05
Pow	1,03	1,06	1,06	1,08	1,12
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,01	1,02	1,03	1,02
Sub	1,02	1,02	1,02	1,01	1,02
Mul	1,03	1,04	1,05	1,05	1,06
Sqr	1,03	1,04	1,05	1,03	1,04
Pow	1,03	1,03	1,06	1,07	1,08

У таблиці 7 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у полі цілих чисел GF(p) для платформи Desktop для 64-бітних машинних слів.

Таблиця 7. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Desktop з використанням 64-бітних машинних слів

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0041664	0,0045237	0,0045226	0,0061561	0,0085004
Sub, mks	0,0057848	0,0068908	0,0067170	0,0069598	0,0088942
Mul, mks	0,02843	0,07572	0,09074	0,13961	0,12496
Sqr, mks	0,02733	0,07268	0,08539	0,12737	0,11471
Pow, mks	8,05089	25,38679	36,18904	91,49100	99,07660
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0046480	0,0045316	0,0045272	0,0057318	0,0082583
Sub, mks	0,0057657	0,0065818	0,0052945	0,0095999	0,0086891
Mul, mks	0,07351	0,10805	0,11654	0,22388	0,39239
Sqr, mks	0,06996	0,10384	0,11076	0,20782	0,36929
Pow, mks	22,81074	36,63358	44,18180	131,20126	306,11580
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0041251	0,00439194	0,00439092	0,00591929	0,00817346
Sub, mks	0,00567138	0,00675568	0,00645863	0,00662841	0,00855213
Mul, mks	0,02734	0,07351	0,0856	0,13048	0,11901
Sqr, mks	0,02628	0,06922	0,08132	0,12016	0,10822
Pow, mks	7,8164	23,9498	34,1406	83,9367	86,9093
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0045569	0,00439962	0,00439532	0,00556487	0,00801776
Sub, mks	0,00565264	0,00645273	0,0051403	0,00923067	0,00835486
Mul, mks	0,07137	0,1049	0,11099	0,21121	0,37018
Sqr, mks	0,06792	0,09985	0,10549	0,19792	0,35509
Pow, mks	21,9334	35,2246	42,0779	122,618	275,78

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 8 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,01-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,03 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,02-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,03-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,06 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,04-1,06 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,05 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,14 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,04-1,11 рази для всіх полів з загальним простим модулем.

Таблиця 8. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у полі цілих чисел GF(p) різної довжини для платформи Desktop з використанням 64-бітних машинних слів

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,01	1,03	1,03	1,04	1,04
Sub	1,02	1,02	1,04	1,05	1,04
Mul	1,04	1,03	1,06	1,07	1,05
Sqr	1,04	1,05	1,05	1,06	1,06
Pow	1,03	1,06	1,06	1,09	1,14
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,03	1,03	1,03	1,03
Sub	1,02	1,02	1,03	1,04	1,04
Mul	1,03	1,03	1,05	1,06	1,06
Sqr	1,03	1,04	1,05	1,05	1,04
Pow	1,04	1,04	1,05	1,07	1,11

Наступними будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблиці 9 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операції у полі цілих чисел GF(p) для платформи Mobile з використанням 32-бітних машинних слів.

Таблиця 9. Результати вимірювання швидкодії арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Mobile з використанням 32-бітних машинних слів.

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,01084	0,01059	0,01208	0,01418	0,01695
Sub, mks	0,01983	0,01308	0,02256	0,02786	0,02309
Mul, mks	0,5906	0,1943	0,3063	0,5425	0,6879
Sqr, mks	0,1428	0,1321	0,2478	0,4251	0,4558
Pow, mks	35,638	54,3119	117,6812	260,0392	445,3098
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,010778	0,011457	0,011826	0,014057	0,035127
Sub, mks	0,013192	0,012879	0,013942	0,017298	0,038937
Mul, mks	0,360060	0,496080	0,632320	1,199100	2,275890
Sqr, mks	0,325480	0,441870	0,569100	1,076250	2,027780
Pow, mks	101,0672	160,240200	212,837400	640,426500	1761,9959

Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,010628	0,010383	0,011725	0,013508	0,016302
Sub, mks	0,019631	0,012698	0,022116	0,026787	0,022198
Mul, mks	0,579	0,185	0,289	0,507	0,649
Sqr, mks	0,14	0,127	0,236	0,401	0,43
Pow, mks	34,6	52,73	111,02	245,32	401,18
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,010567	0,011016	0,011482	0,013516	0,034104
Sub, mks	0,013061	0,012504	0,013669	0,016794	0,037803
Mul, mks	0,353	0,477	0,608	1,142	2,127
Sqr, mks	0,316	0,429	0,542	1,025	1,913
Pow, mks	97,18	151,17	200,79	609,93	1616,51

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 10 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,01-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,03 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,02-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,07 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,02-1,06 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,03-1,06 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,11 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,04-1,09 рази для всіх полів з загальним простим модулем.

Таблиця 10. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у полі цілих чисел GF(p) різної довжини для платформи Mobile з використанням 32-бітних машинних слів

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,02	1,03	1,05	1,04
Sub	1,01	1,03	1,02	1,04	1,04
Mul	1,02	1,05	1,06	1,07	1,06
Sqr	1,02	1,04	1,05	1,06	1,06
Pow	1,03	1,03	1,06	1,06	1,11
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,04	1,03	1,04	1,03
Sub	1,01	1,03	1,02	1,03	1,03
Mul	1,02	1,04	1,04	1,05	1,07
Sqr	1,03	1,03	1,05	1,05	1,06
Pow	1,04	1,06	1,06	1,05	1,09

У таблиці 11 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у полі цілих чисел GF(p) для платформи Mobile з використанням 64-бітних машинних слів.

Таблиця 11. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Mobile з використанням 64-бітних машинних слів.

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,00442394	0,0048276	0,00482664	0,00578605	0,00817068
Sub, mks	0,0053468	0,00626769	0,00637139	0,00612203	0,01234563
Mul, mks	0,02791	0,08249	0,08919	0,14274	0,17430
Sqr, mks	0,02430	0,07728	0,07832	0,13477	0,10964
Pow, mks	8,27852	29,06128	37,37062	88,69327	94,35489
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,00417907	0,00472163	0,00486741	0,00570296	0,01254736
Sub, mks	0,00540331	0,00621404	0,0063248	0,00850026	0,01233523
Mul, mks	0,0738916	0,1080367	0,1102082	0,2093606	0,388416
Sqr, mks	0,0728382	0,1049464	0,1065406	0,194691	0,3604744
Pow, mks	23,149614	35,99232	42,368445	119,01435	297,44004
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,0043372	0,00468699	0,004641	0,00556351	0,00785642
Sub, mks	0,00524196	0,00608514	0,00612634	0,0058305	0,0118708
Mul, mks	0,02684	0,07932	0,08494	0,1334	0,16443
Sqr, mks	0,02337	0,0736	0,07459	0,12714	0,10247
Pow, mks	8,0374	27,4163	35,2553	82,1234	83,4999
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,00409713	0,00462905	0,00472564	0,00553685	0,0121819
Sub, mks	0,00529736	0,0060922	0,00614058	0,00817333	0,0118608
Mul, mks	0,07316	0,10489	0,10397	0,19751	0,36992
Sqr, mks	0,07141	0,10091	0,10051	0,18542	0,34661
Pow, mks	22,6957	34,944	40,3509	113,347	267,964

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,03 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,02-1,04 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,04-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,06 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,04-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,06 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,03-1,13 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,11 рази для всіх полів з загальним простим модулем.

Таблиця 12. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій в у поля цілих чисел GF(p) різної довжини для платформи Mobile з використанням 64-бітних машинних слів

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,03	1,04	1,04	1,04
Sub	1,02	1,03	1,04	1,05	1,04
Mul	1,04	1,04	1,05	1,07	1,06
Sqr	1,04	1,05	1,05	1,06	1,07
Pow	1,03	1,06	1,06	1,08	1,13
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,02	1,03	1,03	1,03
Sub	1,02	1,02	1,03	1,04	1,04
Mul	1,01	1,03	1,06	1,06	1,05
Sqr	1,02	1,04	1,06	1,05	1,04
Pow	1,02	1,03	1,05	1,05	1,11

Наступними будуть розглядатися результати отримані для платформи Embedded (Cortex-A72)

У таблиці 13 наведені результати вимірювання швидкодії класичних та запропонованих методів арифметичних операцій у простому полі GF(p) для платформи Embedded.

Таблиця 13. Результати вимірювання швидкодії методів арифметичних операцій в полі цілих чисел GF(p) різної довжини для платформи Embedded з використанням 64-бітних машинних слів

Pseudo-Mersenne prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,013424	0,030323	0,028699	0,046988	0,037283
Sub, mks	0,028762	0,025310	0,039715	0,054239	0,044573
Mul, mks	0,1114	0,2132	0,3322	0,5454	0,4817
Sqr, mks	0,0949	0,1793	0,3351	0,5354	0,3401
Pow, mks	25,9325	60,3127	161,4178	351,1166	363,4683
Common prime module (original)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,013272	0,030028	0,028544	0,047386	0,069440
Sub, mks	0,028745	0,025331	0,039163	0,053200	0,075862
Mul, mks	0,2504	0,3958	0,3861	0,7989	1,6590
Sqr, mks	0,2428	0,4038	0,3915	0,7562	1,5636
Pow, mks	81,2950	150,2800	179,8580	526,6328	1453,6021
Pseudo-Mersenne prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,013164	0,029440	0,027595	0,045574	0,035508
Sub, mks	0,028198	0,024814	0,038558	0,051656	0,042859
Mul, mks	0,1103	0,2070	0,3210	0,5145	0,4544
Sqr, mks	0,0940	0,1758	0,3219	0,5004	0,3239
Pow, mks	25,4240	57,9930	154,7630	325,1080	329,8260
Common prime module (with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add, mks	0,013141	0,029440	0,027579	0,045564	0,067417
Sub, mks	0,028181	0,024835	0,038395	0,051651	0,072945
Mul, mks	0,2479	0,3880	0,3730	0,7682	1,5952
Sqr, mks	0,2404	0,3943	0,3775	0,7202	1,4891
Pow, mks	79,701	144,500	172,443	487,623	1343,440

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання класичних методів арифметичних операцій в полі цілих чисел GF(p) до результатів вимірювання запропонованих методів арифметичних операцій в полі цілих чисел GF(p).

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання за модулем ефективніша в 1,02-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,04 рази для всіх полів з загальним простим модулем.
- Операція віднімання за модулем ефективніша в 1,02-1,05 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,04 рази для всіх полів з загальним простим модулем.
- Операція множення за модулем ефективніша в 1,01-1,06 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,04 рази для всіх полів з загальним простим модулем.
- Операція піднесення до квадрату за модулем ефективніша в 1,01-1,07 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,01-1,05 рази для всіх полів з загальним простим модулем.
- Операція піднесення до степеню за модулем ефективніша в 1,02-1,10 рази для всіх полів з псевдомерсеновим простим модулем, та в 1,02-1,08 рази для всіх полів з загальним простим модулем.

Таблиця 14. Нормалізовані результати вимірювання швидкодії запропонованих методів арифметичних операцій у полі цілих чисел GF(p) різної довжини для платформи Mobile

Pseudo-Mersenne prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,02	1,03	1,04	1,03	1,05
Sub	1,02	1,02	1,03	1,05	1,04
Mul	1,01	1,03	1,04	1,06	1,06
Sqr	1,01	1,02	1,04	1,07	1,05
Pow	1,02	1,04	1,04	1,08	1,10
Common prime module (original/with improvements)					
Операція/Поле	p192	p224	p256	p384	p521
Add	1,01	1,02	1,04	1,04	1,03
Sub	1,02	1,02	1,02	1,03	1,04
Mul	1,01	1,02	1,04	1,04	1,04
Sqr	1,01	1,02	1,04	1,05	1,05
Pow	1,02	1,04	1,04	1,08	1,08

Експериментальні дослідження методів арифметичних операцій в групі точок еліптичних кривих над простим полем GF(p)

Експериментальні дослідження проводились для деяких значущих операцій в групі точок ЕК над простим полем GF(p):

- додавання точок ЕК над простим полем GF(p) в афінних координатах (add);
- подвоєння точок ЕК над простим полем GF(p) в афінних координатах (dbl);
- додавання точок ЕК над простим полем GF(p) в змішаних (використовуються проективні координати Чудновського) координатах (addmx Ch);
- подвоєння точок ЕК над простим полем GF(p) в проективних (проективні координати Чудновського) координатах (addmx Ch);
- скалярне множення випадкової точки ЕК над простим полем GF(p) з використанням проективних координат Чудновського для подвоєння та змішаного додавання точок на основі бінарного алгоритму зліва направо (mul (rand));
- скалярне множення фіксованої точки ЕК над простим полем GF(p) з використанням проективних координат Чудновського для подвоєння та змішаного додавання точок на основі алгоритму Лім-Лі з попередніми обчисленнями (mul (fixed)).

Значення швидкодії вказаних операцій в группі точок ЕК над простим полем GF(p) зазначені в мікросекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблиці 1 наведені результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) з та без запропонованих методів для платформи Server для 32-бітних машинних слів.

Таблиця 1. Результати вимірювання швидкодії операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Server з використанням 32-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	5,73	8,221	10,12	19,92	38,465
dbl, mks	5,71	8,223	9,919	19,511	37,804
addmx Ch, mks	1,143	1,567	2,763	4,257	5,961
dbl Ch, mks	0,893	1,334	2,204	3,575	4,269
mul (rand), mks	330	540	1058	2500	3830
mul (fixed), mks	330	530	1060	2450	3780
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	5,56	7,942	9,567	18,781	35,702
dbl, mks	5,534	7,821	9,624	18,668	35,752
addmx Ch, mks	1,106	1,527	2,613	4,089	5,671
dbl Ch, mks	0,868	1,244	2,103	3,405	3,951
mul (rand), mks	310	510	970	2240	3480
mul (fixed), mks	310	500	970	2180	3500

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в групі точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів арифметичних операцій.

У таблиці 2 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів арифметичних операцій для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,03-1,08 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,03-1,06 рази для всіх полів.

- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,03-1,06 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,03-1,08 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,06-1,12 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,06-1,12 рази для всіх полів.

Таблиця 2. Нормалізовані результати вимірювання швидкодії операцій у групі точок ЕК над простим поле GF(p) різної довжини для платформи Server за результатами з використанням 32-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,03	1,04	1,06	1,06	1,08
dbl, mks	1,03	1,05	1,03	1,05	1,06
addmx Ch, mks	1,03	1,03	1,06	1,04	1,05
dbl Ch, mks	1,03	1,07	1,05	1,05	1,08
mul (rand), ms	1,06	1,06	1,09	1,12	1,10
mul (fixed), ms	1,06	1,06	1,09	1,12	1,08

У таблиці 3 наведені результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) для платформи Server з використанням 64-бітних машинних слів.

Таблиця 3. Результати вимірювання швидкодії арифметичних операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Server з використанням 64-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,9828	3,40242	3,8286	6,93764	13,9823
dbl, mks	2,02341	3,30226	3,71294	6,93253	14,0205
addmx Ch, mks	0,2731	0,74614	0,8298	1,23321	1,11314
dbl Ch, mks	0,2641	0,67663	0,75334	1,16352	0,98631
mul (rand), mks	112,5	265,3	413,7	791,2	857,2
mul (fixed), mks	114,1	269,1	421,2	794,2	851,8
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,91325	3,12871	3,52063	7,23764	13,7489
dbl, mks	1,93032	3,11261	3,42329	6,79931	13,7722
addmx Ch, mks	0,26614	0,71717	0,78094	1,18385	1,03626
dbl Ch, mks	0,25289	0,64028	0,70017	1,08125	0,92563
mul (rand), mks	106,8	249,3	387,7	744,5	788,2
mul (fixed), mks	106,6	249,6	392,4	745,8	776,2

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в групі точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів арифметичних операцій.

У таблиці 4 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,09 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,08 рази для всіх полів.
- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,03-1,07 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,04-1,08 рази для всіх полів.

- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,05-1,09 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,06-1,10 рази для всіх полів.

Таблиця 4. Нормалізовані результати вимірювання швидкодії операцій у групі точок ЕК над простим поле GF(p) різної довжини для платформи Server для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,04	1,09	1,09	1,05	1,02
dbl, mks	1,05	1,06	1,08	1,02	1,02
addmx Ch, mks	1,03	1,04	1,06	1,04	1,07
dbl Ch, mks	1,04	1,06	1,08	1,08	1,07
mul (rand), ms	1,05	1,06	1,07	1,06	1,09
mul (fixed), ms	1,07	1,08	1,07	1,06	1,10

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблиці 5 наведені результати вимірювання швидкодії операцій у групі тоок ЕК над простим полем GF(p) з та без запропонованих методів для платформи Desktop.

Таблиця 5. Результати вимірювання швидкодії операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Desktop з використанням 32-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	7,101	10,327	13,031	26,050	45,394
dbl, mks	7,008	10,287	12,868	25,937	45,688
addmx Ch, mks	1,517	2,167	3,707	6,005	7,323
dbl Ch, mks	1,242	1,768	2,956	4,824	5,443
mul (rand), mks	420	630	1350	3240	5020
mul (fixed), mks	410	620	1340	3130	5040
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	6,941	10,164	12,530	25,048	44,072
dbl, mks	6,871	10,125	12,493	24,939	43,931
addmx Ch, mks	1,473	2,098	3,564	5,774	7,110
dbl Ch, mks	1,194	1,684	2,898	4,729	5,234
mul (rand), mks	410	610	1300	3030	4780
mul (fixed), mks	400	600	1280	3010	4710

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання арифметичних операцій в групах точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 6 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,03-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,02-1,05 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,02-1,07 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,03-1,07 рази для всіх полів.

Таблиця 6. Нормалізовані результати вимірювання швидкодії операцій у полі цілих чисел GF(p) різної довжини для платформи Desktop за результатами з використанням запропонованих методів для 32-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,02	1,02	1,04	1,04	1,03
dbl, mks	1,02	1,02	1,03	1,04	1,04
addmx Ch, mks	1,03	1,03	1,04	1,04	1,03
dbl Ch, mks	1,04	1,05	1,02	1,02	1,04
mul (rand), ms	1,02	1,04	1,04	1,07	1,05
mul (fixed), ms	1,03	1,04	1,05	1,04	1,07

У таблиці 7 наведені результати вимірювання швидкодії операції у полі цілих чисел GF(p) з та без запропонованих методів для платформи Desktop для 64-бітних машинних слів.

Таблиця 7. Результати вимірювання швидкодії операцій в групах точок еліптичних кривих над простим полем GF(p) різної довжини для платформи Desktop з використанням 64-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	2,5953	4,0617	4,6065	9,5544	19,9392
dbl, mks	3,0594	4,0583	4,6180	9,4333	19,6689
addmx Ch, mks	0,4028	0,9599	1,1119	1,6781	1,4614
dbl Ch, mks	0,3556	0,8882	1,0743	1,5305	1,2155
mul (rand), mks	135,7	340,1	518,7	1027,3	1072,8
mul (fixed), mks	136,3	343,2	516,9	1022,1	1049,2
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	2,5544	3,9357	4,4636	9,1605	18,9897
dbl, mks	2,8829	3,9787	4,5275	9,0880	18,8580
addmx Ch, mks	0,3949	0,9283	1,0795	1,6135	1,4052
dbl Ch, mks	0,3453	0,8459	1,0330	1,4716	1,1576
mul (rand), mks	133,4	333,4	508,2	988,2	1011,7
mul (fixed), mks	132,3	330,0	497,0	963,9	980,4

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання арифметичних операцій в групах точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 8 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,05 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,06 рази для всіх полів.
- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,03-1,05 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,02-1,06 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,03-1,07 рази для всіх полів.

Таблиця 8. Нормалізовані результати вимірювання швидкодії операцій у полі цілих чисел GF(p) різної довжини для платформи Desktop за результатами з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,02	1,03	1,03	1,04	1,05
dbl, mks	1,06	1,02	1,02	1,04	1,04
addmx Ch, mks	1,02	1,03	1,03	1,04	1,04
dbl Ch, mks	1,03	1,05	1,04	1,04	1,05
mul (rand), ms	1,02	1,02	1,02	1,04	1,06
mul (fixed), ms	1,03	1,04	1,04	1,06	1,07

Наступними будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблиці 9 наведені результати вимірювання швидкодії операцій у групі точок ЕК над просим полем GF(p) з та без запропонованих методів для платформи Mobile для 32-бітних машинних слів.

Таблиця 9. Результати вимірювання швидкодії операцій в групах точок ЕК над простим полем GF(p) різної довжини для платформи Mobile з використанням 32-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	7,200	10,985	13,550	24,980	48,113
dbl, mks	7,396	10,409	13,295	24,677	47,784
addmx Ch, mks	1,440	2,495	3,668	5,491	7,731
dbl Ch, mks	1,136	1,748	2,939	4,506	5,177
mul (rand), mks	397,80	686,40	1404,00	3055,50	4557,00
mul (fixed), mks	408,00	676,00	1328,70	3109,60	4654,50
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	7,011	10,791	13,155	24,252	46,712
dbl, mks	7,251	10,245	13,163	24,193	46,392
addmx Ch, mks	1,398	2,415	3,527	5,331	7,434
dbl Ch, mks	1,103	1,681	2,853	4,333	5,026
mul (rand), mks	390,00	660,00	1350,00	2910,00	4340,00
mul (fixed), mks	400,00	650,00	1290,00	2990,00	4350,00

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в групі точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 10 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,03 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,01-1,03 рази для всіх полів.
- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,03-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,03-1,04 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,02-1,05 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,02-1,07 рази для всіх полів.

Таблиця 10. Нормалізовані результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) різної довжини для платформи Mobile за результатами з використанням запропонованих методів для 32-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,03	1,02	1,03	1,03	1,03
dbl, mks	1,02	1,02	1,01	1,02	1,03
addmx Ch, mks	1,03	1,03	1,04	1,03	1,04
dbl Ch, mks	1,03	1,04	1,03	1,04	1,03
mul (rand), ms	1,02	1,04	1,04	1,05	1,05
mul (fixed), ms	1,02	1,04	1,03	1,04	1,07

У таблиці 11 наведені результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) з та без запропонованих методів для платформи Mobile для 64-бітних машинних слів.

Таблиця 11. Результати вимірювання швидкодії арифметичних операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Mobile з використанням 64-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	2,3698	3,8833	4,5627	9,3623	18,6795
dbl, mks	2,4449	3,8148	4,5489	9,0403	18,8449
addmx Ch, mks	0,3384	1,0014	1,0254	1,6138	1,3846
dbl Ch, mks	0,3300	0,8521	0,9537	1,4458	1,1481
mul (rand), mks	133,49	346,92	507,05	1046,03	1097,71
mul (fixed), mks	132,97	344,24	519,17	1019,13	1099,85
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	2,3325	3,7629	4,4213	8,9763	18,1354
dbl, mks	2,3737	3,7400	4,4598	8,7431	18,0680
addmx Ch, mks	0,3318	0,9685	0,9860	1,5517	1,3313
dbl Ch, mks	0,3204	0,8115	0,9170	1,3902	1,1147
mul (rand), mks	129,60	330,40	482,90	977,60	1025,90
mul (fixed), mks	129,10	331,00	499,20	970,60	1027,90

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в групі точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція додавання точок ЕК над простим полем GF(p) в змішаних координатах ефективніша в 1,02-1,04 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в проективних координатах ефективніша в 1,03-1,05 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,03-1,07 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,03-1,07 рази для всіх полів.

Таблиця 12. Нормалізовані результати вимірювання швидкодії операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Mobile з використанням запропонованих методів з використанням 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,02	1,03	1,03	1,04	1,03
dbl, mks	1,03	1,02	1,02	1,03	1,04
addmx Ch, mks	1,02	1,03	1,04	1,04	1,04
dbl Ch, mks	1,03	1,05	1,04	1,04	1,03
mul (rand), ms	1,03	1,05	1,05	1,07	1,07
mul (fixed), ms	1,03	1,04	1,04	1,05	1,07

Наступними будуть розглядатися результати отримані для платформи Embedded (Cortex-A72)

У таблиці 13 наведені результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) з та без запропонованих методів для платформи Embedded.

Таблиця 13. Результати вимірювання швидкодії операцій в групі точок ЕК над простим полем GF(p) різної довжини для платформи Embedded з використанням 64-бітних машинних слів

NIST FIPS 186-3 EC (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	7,6376	10,5438	12,5274	28,4171	60,3388
dbl, mks	6,7548	10,5964	12,2750	27,8572	58,8201
mul (rand), mks	441,7237	775,1653	1763,4960	3466,6380	3945,7016
mul (fixed), mks	437,9390	793,4233	1776,2844	3565,0260	3985,0117
NIST FIPS 186-3 EC x86 (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	7,4878	10,4394	12,0456	26,8086	56,9234
dbl, mks	6,6224	10,3886	11,9175	26,5307	56,5578
mul (rand), mks	428,8580	759,9660	1679,5200	3301,5600	3722,3600
mul (fixed), mks	429,3520	762,9070	1675,7400	3300,9500	3724,3100

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій в групі точок ЕК над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція додавання точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,01-1,06 рази для всіх полів.
- Операція подвоєння точок ЕК над простим полем GF(p) в афінних координатах ефективніша в 1,02-1,05 рази для всіх полів.
- Операція скалярного множення випадкової точки ЕК над простим полем GF(p) ефективніша в 1,02-1,06 рази для всіх полів.
- Операція скалярного множення фіксованої точки ЕК над простим полем GF(p) ефективніша в 1,02-1,08 рази для всіх полів.

Таблиця 14. Нормалізовані результати вимірювання швидкодії операцій у групі точок ЕК над простим полем GF(p) різної довжини для платформи Embedded за результатами без та з використанням запропонованих методів

Операція/Базове поле	p192	p224	p256	p384	p521
add, mks	1,02	1,01	1,04	1,06	1,06
dbl, mks	1,02	1,02	1,03	1,05	1,04
mul (rand), ms	1,03	1,02	1,05	1,05	1,06
mul (fixed), ms	1,02	1,04	1,06	1,08	1,07

Експериментальні дослідження методів арифметичних операцій в криптосистемі ECDSA над простим полем GF(p)

Експериментальні дослідження проводились для основних операцій криптосистеми ECDSA над простим полем GF(p):

- генерування особистого ключа (private key generation) – операції у простому полі цілих чисел, за модулем порядку групи – простого числа загального вигляду;
- генерування відкритого ключа (public key generation) – скалярне множення за фіксованою точкою ЕК (генератором групи);
- створення підпису (digest sign) – скалярне множення за фіксованою точкою ЕК (генератором групи);
- перевірка підпису (digest verify sign) – скалярне множення за довільною (відкритий ключ) і фіксованою точкою ЕК (генератором групи).

При генеруванні відкритого ключа та створення підпису відбувається скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі з використанням проективних координат Чудновського. При перевірці підпису відбувається множення фіксованої (базової) точки на основі алгоритму Лім-Лі і довільної точки (відкритого ключа) на основі двійкового алгоритму зліва направо з використанням проективних координат Чудновського.

Слід зазначити, що операції у базовому полі виконуються за модулем простого числа спеціального вигляду – псевдомерсена, а операції у полі порядку базової точки, відбувається за модулем простого числа загального вигляду.

Значення швидкодії вказаних операцій криптосистеми ECDSA над простим полем GF(p) зазначені в мілісекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблиці 1 наведені результати вимірювання швидкодії операції криптосистеми ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Server з використанням 32-бітних машинних слів.

Таблиця 1. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Server з використанням 32-бітних машинних слів.

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,013	0,013	0,016	0,020	0,040
Public key generation, ms	0,301	0,511	1,012	2,323	3,479
Digest sign, ms	0,327	0,540	1,090	2,303	3,610
Digest verify sign, ms	0,336	0,560	1,122	2,320	3,640
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,012	0,012	0,015	0,019	0,039
Public key generation, ms	0,297	0,505	1,002	2,291	3,390
Digest sign, ms	0,321	0,530	1,050	2,218	3,487
Digest verify sign, ms	0,332	0,550	1,091	2,273	3,595

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 2 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,03-1,08 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,01-1,03 рази для всіх полів, що розглядаються.

- Операція створення підпису ефективніша в 1,02-1,04 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,02 рази для всіх полів, що розглядаються.

Таблиця 2. Нормалізовані результати вимірювання швидкодії операцій криптосистеми ECDSA над простим поле GF(p) різної довжини для платформи Server за результатами без та з використанням запропонованих методів для 32-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,08	1,08	1,07	1,05	1,03
Public key generation	1,01	1,01	1,01	1,01	1,03
Digest sign	1,02	1,02	1,04	1,04	1,04
Digest verify sign	1,01	1,02	1,03	1,02	1,01

У таблиці 3 наведені результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Server для 64-бітних машинних слів.

Таблиця 3. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Server з використанням 64-бітних машинних слів

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,004	0,006	0,006	0,011	0,021
Public key generation, ms	0,103	0,257	0,394	0,773	0,847
Digest sign, ms	0,117	0,269	0,403	0,782	0,863
Digest verify sign, ms	0,112	0,258	0,423	0,823	0,884
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,004	0,006	0,007	0,012	0,021
Public key generation, ms	0,106	0,260	0,399	0,782	0,851
Digest sign, ms	0,118	0,273	0,412	0,789	0,877
Digest verify sign, ms	0,115	0,271	0,428	0,833	0,903

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 4 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-1,05 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,01-1,02 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,01-1,02 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,05 рази для всіх полів, що розглядаються.

Таблиця 4. Нормалізовані результати вимірювання швидкодії операцій криптосистеми ECDSA над простим поле GF(p) різної довжини для платформи Server за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,02	1,02	1,03	1,05	1,02
Public key generation	1,02	1,01	1,01	1,01	1,01
Digest sign	1,01	1,01	1,02	1,01	1,02
Digest verify sign	1,02	1,05	1,01	1,01	1,02

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблиці 5 наведені результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів.

Таблиця 5. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Desktop з використанням 32-бітних машинних слів.

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,013	0,013	0,015	0,020	0,051
Public key generation, ms	0,410	0,681	1,360	3,030	4,775
Digest sign, ms	0,416	0,682	1,384	2,996	4,806
Digest verify sign, ms	0,420	0,694	1,400	3,100	4,937
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,011	0,011	0,013	0,018	0,050
Public key generation, ms	0,403	0,642	1,330	2,863	4,604
Digest sign, ms	0,405	0,664	1,350	2,954	4,723
Digest verify sign, ms	0,408	0,680	1,370	3,072	4,856

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 6 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-1,18 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,02-1,06 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,01-1,03 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,03 рази для всіх полів, що розглядаються.

Таблиця 6. Нормалізовані результати вимірювання швидкодії операцій криптосистеми ECDSA над простим поле GF(p) різної довжини для платформи Desktop за результатами без та з використанням запропонованих методів з використанням 32-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,18	1,18	1,15	1,11	1,02
Public key generation	1,02	1,06	1,02	1,06	1,04
Digest sign	1,03	1,03	1,03	1,01	1,02
Digest verify sign	1,03	1,02	1,02	1,01	1,02

У таблиці 7 наведені результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Desktop для 64-бітних машинних слів.

Таблиця 7. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Desktop з використанням 64-бітних машинних слів

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,0048	0,0068	0,0078	0,0137	0,0265
Public key generation, ms	0,1307	0,3079	0,4841	0,9165	1,0273
Digest sign, ms	0,1372	0,3345	0,4950	0,9830	1,0524
Digest verify sign, ms	0,1338	0,3558	0,5162	0,9905	1,0941
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,0047	0,0064	0,0072	0,0132	0,0257
Public key generation, ms	0,1252	0,2991	0,4785	0,9096	1,0174
Digest sign, ms	0,1320	0,3262	0,4840	0,9333	1,0326
Digest verify sign, ms	0,1322	0,3442	0,5119	0,9561	1,0589

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 8 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-1,08 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,01-1,04 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,02-1,05 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,04 рази для всіх полів, що розглядаються.

Таблиця 8. Нормалізовані результати вимірювання швидкодії операцій криптосистемі ECDSA над простим поле GF(p) різної довжини для платформи Desktop за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,02	1,06	1,08	1,04	1,03
Public key generation	1,04	1,03	1,01	1,01	1,01
Digest sign	1,04	1,03	1,02	1,05	1,02
Digest verify sign	1,01	1,03	1,01	1,04	1,03

Наступними будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблиці 9 наведені результати вимірювання швидкодії операцій криптосистемі ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів.

Таблиця 9. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Mobile з використанням 32-бітних машинних слів

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,012	0,013	0,013	0,035	0,056
Public key generation, ms	0,410	0,681	1,360	2,918	4,466
Digest sign, ms	0,43	0,67	1,31	2,88	4,54
Digest verify sign, ms	0,44	0,74	1,45	3,14	4,69
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,011	0,011	0,011	0,030	0,050
Public key generation, ms	0,389	0,663	1,337	2,872	4,372
Digest sign, ms	0,42	0,63	1,19	2,64	4,31
Digest verify sign, ms	0,43	0,72	1,41	3,09	4,66

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 10 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,09-1,18 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,02-1,05 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,02-1,10 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,03 рази для всіх полів, що розглядаються.

Таблиця 10. Нормалізовані результати вимірювання швидкодії операцій криптосистеми ECDSA над простим поле GF(p) різної довжини для платформи Mobile за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,09	1,18	1,18	1,17	1,12
Public key generation	1,05	1,03	1,02	1,02	1,02
Digest sign	1,02	1,06	1,10	1,09	1,05
Digest verify sign	1,02	1,03	1,03	1,02	1,01

У таблиці 11 наведені результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Mobile для 64-бітних машинних слів.

Таблиця 11. Результати вимірювання швидкодії операцій криптосистеми ECDSA над простим полем GF(p) різної довжини для платформи Mobile з використанням 64-бітних машинних слів

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,0056	0,0073	0,0086	0,0151	0,0278
Public key generation, ms	0,1325	0,3324	0,5248	0,9793	1,0387
Digest sign, ms	0,1435	0,3515	0,5233	0,9843	1,0733
Digest verify sign, ms	0,1431	0,3636	0,5682	1,0745	1,1541
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,0054	0,0071	0,0083	0,0148	0,0272
Public key generation, ms	0,1275	0,3237	0,5156	0,9284	1,0031
Digest sign, ms	0,1403	0,3356	0,5078	0,9563	1,0493
Digest verify sign, ms	0,1395	0,3450	0,5482	1,0599	1,1473

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-1,04 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,02-1,05 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,02-1,05 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,05 рази для всіх полів, що розглядаються.

Таблиця 12. Нормалізовані результати вимірювання швидкодії операцій криптосистемі ECDSA над простим поле GF(p) різної довжини для платформи Mobile за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,04	1,03	1,04	1,02	1,02
Public key generation	1,04	1,03	1,02	1,05	1,04
Digest sign	1,02	1,05	1,03	1,03	1,02
Digest verify sign	1,03	1,05	1,04	1,01	1,01

Наступними будуть розглядатися результати отримані для платформи Embedded (Cortex-A72)

У таблиці 13 наведені результати вимірювання швидкодії операцій криптосистемі ECDSA над простим полем GF(p) з та без запропонованих методів для платформи Embedded з використанням 32-бітних машинних слів.

Таблиця 13. Результати вимірювання швидкодії операцій в криптосистемі ECDSA над простим полем GF(p) різної довжини для платформи Embedded для 32-бітних машинних слів

ECDSA (original)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,013	0,017	0,019	0,038	0,070
Public key generation, ms	0,453	0,841	1,723	3,396	3,973
Digest sign, ms	0,474	0,875	1,785	3,580	4,235
Digest verify sign, ms	0,483	0,884	1,830	4,832	4,591
ECDSA (with improvements)					
Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation, ms	0,012	0,016	0,018	0,035	0,068
Public key generation, ms	0,436	0,819	1,674	3,278	3,908
Digest sign, ms	0,456	0,828	1,683	3,378	4,049
Digest verify sign, ms	0,466	0,830	1,686	3,760	4,014

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій криптосистеми ECDSA над простим полем GF(p) без використання запропонованих методів до результатів з використанням запропонованих методів.

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,03-1,09 рази для всіх полів, що розглядаються.
- Операція генерування відкритого ключа ефективніша в 1,02-1,04 рази для всіх полів, що розглядаються.
- Операція створення підпису ефективніша в 1,04-1,06 рази для всіх полів, що розглядаються.
- Операція перевірки підпису ефективніша в 1,04-1,29 рази для всіх полів, що розглядаються.

Таблиця 14. Нормалізовані результати вимірювання швидкодії операцій криптосистеми ECDSA над простим поле GF(p) різної довжини для платформи Embedded за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Операція/Базове поле	p192	p224	p256	p384	p521
Private key generation	1,08	1,06	1,06	1,09	1,03
Public key generation	1,04	1,03	1,03	1,04	1,02
Digest sign	1,04	1,06	1,06	1,06	1,05
Digest verify sign	1,04	1,07	1,09	1,29	1,14

ДОДАТОК Е

Експериментальні дослідження методів арифметичних операцій в криптосистемі ДСТУ 4145-2002 над двійковим полем GF(2m)

Для експериментального дослідження були підготовлені дві програмні збірки – з використанням 32-бітних машинних слів, та з використанням 64-бітних машинних слів. Експериментальні дослідження проводились для основних операцій криптосистеми ДСТУ 4145-2002 над двійковим полем $GF(2^m)$:

- генерування особистого ключа (private key generation);
- генерування відкритого ключа (public key generation);
- обчислення передпідпису (presignature generation);
- створення підпису з передпідписом (digest sign with presignature);
- перевірка підпису (digest verify sign).

При генеруванні відкритого ключа, обчисленні передпідпису, створенню підпису з передпідписом відбувається скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі з використанням проективних координат Лопеса-Дахаба. При перевірці підпису застосовується скалярне множення фіксованої (базової) точки на основі алгоритму Лім-Лі і довільної точки (відкритого ключа) на основі двійкового алгоритму зліва направо з використання проективних координат Лопеса-Дахаба.

Значення швидкодії вказаних операцій криптосистеми ДСТУ 4145-2002 над двійковим полем $GF(2^m)$ зазначені в мілісекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G).

У таблицях 1 та 2 наведені результати вимірювання швидкодії операції криптосистеми ДСТУ 4145-2002 над двійковим полем $GF(2^m)$ для платформи Server з використанням 32-бітних і 64-бітних машинних слів.

Таблиця 1. Результати вимірювання швидкодії операцій криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2m) різної довжини для платформи Server з використанням 32-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
$GF(2^{163})$	0,007	0,314	0,314	0,323	0,326
$GF(2^{167})$	0,007	0,309	0,316	0,320	0,328
$GF(2^{173})$	0,007	0,338	0,336	0,331	0,341
$GF(2^{179})$	0,007	0,340	0,346	0,360	0,373
$GF(2^{191})$	0,008	0,425	0,386	0,373	0,380
$GF(2^{233})$	0,012	0,786	0,780	0,789	0,815
$GF(2^{257})$	0,014	1,317	1,337	1,324	1,372
$GF(2^{307})$	0,017	1,639	1,624	1,628	1,690
$GF(2^{367})$	0,022	2,746	2,750	2,751	2,830
$GF(2^{431})$	0,029	4,406	4,399	4,400	4,510

Таблиця 2. Результати вимірювання швидкодії операцій криптосистеми ДСТУ 4145-2002 над двійковим полем $GF(2^m)$ різної довжини для платформи Server з використанням 64-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presig, ms	Digest verify sign, ms
$GF(2^{163})$	0,00369	0,19396	0,19289	0,19329	0,20134
$GF(2^{167})$	0,00366	0,19277	0,19174	0,19193	0,19793
$GF(2^{173})$	0,00387	0,20234	0,20406	0,20421	0,21120
$GF(2^{179})$	0,00393	0,20927	0,21238	0,21260	0,22090
$GF(2^{191})$	0,00415	0,22193	0,22059	0,22269	0,23320
$GF(2^{233})$	0,00576	0,45503	0,45657	0,45729	0,47280
$GF(2^{257})$	0,00720	0,77492	0,77357	0,77403	0,79947
$GF(2^{307})$	0,00839	0,95386	0,93940	0,94076	0,96310
$GF(2^{367})$	0,01096	1,58477	1,57612	1,57805	1,62040
$GF(2^{431})$	0,01458	2,44735	2,45634	2,45810	2,51416

В таблицях відображені результати середнього часу виконання 1 мільйону ітерації операцій в програмній реалізації.

Наступними будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблицях 3 та 4 наведені результати вимірювання швидкодії операції криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) для платформи Desktop для 32-бітних і 64-бітних машинних слів.

Таблиця 3. Результати вимірювання швидкодії операцій криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) різної довжини для платформи Desktop з використанням 32-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
GF(2^{163})	0,008	0,409	0,408	0,408	0,412
GF(2^{167})	0,008	0,41	0,413	0,412	0,423
GF(2^{173})	0,008	0,429	0,431	0,434	0,441
GF(2^{179})	0,009	0,454	0,452	0,455	0,459
GF(2^{191})	0,01	0,479	0,48	0,483	0,489
GF(2^{233})	0,014	0,999	1,003	1,003	1,025
GF(2^{257})	0,017	1,68	1,679	1,683	1,719
GF(2^{307})	0,021	2,04	2,053	2,056	2,218
GF(2^{367})	0,028	3,452	3,45	3,455	3,713
GF(2^{431})	0,036	5,499	6,081	6,119	5,554

Таблиця 4. Результати вимірювання швидкодії операцій в криптосистемі ДСТУ 4145-2002 над двійковим полем GF(2^m) різної довжини для платформи Desktop з використанням 64-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
GF(2^{163})	0,00405	0,22031	0,22087	0,2216	0,23228
GF(2^{167})	0,00414	0,21695	0,2197	0,22003	0,22953
GF(2^{173})	0,00423	0,22923	0,23343	0,23379	0,2415
GF(2^{179})	0,00441	0,24274	0,24705	0,25117	0,25836
GF(2^{191})	0,00474	0,25266	0,26682	0,26589	0,28079
GF(2^{233})	0,00704	0,531	0,52708	0,51429	0,52793
GF(2^{257})	0,00849	0,85054	0,84687	0,84814	0,85831
GF(2^{307})	0,01032	1,03092	1,02783	1,0284	1,04751
GF(2^{367})	0,01264	1,71864	1,71374	1,71501	1,74504
GF(2^{431})	0,01693	2,67203	2,66542	2,66703	2,71298

В таблицях відображені результати середнього часу виконання 1 мільйону ітерації операцій в програмній реалізації.

Наступними будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблицях 5 та 6 наведені результати вимірювання швидкодії операції криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) для платформи Mobile для 32-бітних і 64-бітних машинних слів.

Таблиця 5. Результати вимірювання швидкодії операцій криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) різної довжини для платформи Mobile з використанням 32-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
GF(2^{163})	0,009	0,401	0,425	0,405	0,435
GF(2^{167})	0,009	0,404	0,411	0,412	0,438
GF(2^{173})	0,009	0,443	0,445	0,437	0,461
GF(2^{179})	0,01	0,455	0,454	0,451	0,488
GF(2^{191})	0,01	0,481	0,487	0,483	0,565
GF(2^{233})	0,016	1,006	1,017	1,018	1,078
GF(2^{257})	0,018	1,791	1,727	1,72	1,856
GF(2^{307})	0,023	2,131	2,124	2,126	2,265
GF(2^{367})	0,03	3,576	3,581	3,6	3,785
GF(2^{431})	0,038	5,721	5,708	5,719	6,097

Таблиця 6. Результати вимірювання швидкодії операцій в криптосистемі ДСТУ 4145-2002 над двійковим полем GF(2^m) різної довжини для платформи Mobile з використанням 64-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
GF(2^{163})	0,00466	0,24476	0,24592	0,24525	0,26233
GF(2^{167})	0,00477	0,25786	0,26064	0,2613	0,28008
GF(2^{173})	0,00501	0,26132	0,27348	0,27354	0,29176
GF(2^{179})	0,00535	0,28275	0,28532	0,28492	0,30533
GF(2^{191})	0,00734	0,58353	0,5961	0,58085	0,62321
GF(2^{233})	0,00902	0,97467	0,97026	0,96929	1,03063
GF(2^{257})	0,01055	1,18324	1,18479	1,18557	1,25892
GF(2^{307})	0,0138	1,9873	1,98376	1,98365	2,11257
GF(2^{367})	0,01814	3,07849	3,07844	3,08223	3,27165
GF(2^{431})	0,00466	0,24476	0,24592	0,24525	0,26233

В таблицях відображені результати середнього часу виконання 1 мільйону ітерації операцій в програмній реалізації.

Embedded (Cortex-A72)

У таблицях 7 наведені результати вимірювання швидкодії операції криптосистеми ДСТУ 4145-2002 над двійковим полем GF(2^m) для платформи Embedded для 64-бітних машинних слів.

Таблиця 7. Результати вимірювання швидкодії операцій в криптосистемі ДСТУ 4145-2002 над двійковим полем GF(2^m) різної довжини для платформи Embedded з використанням 64-бітних машинних слів

Операція / Базове поле	Private key generation, ms	Public key generation, ms	Presignature generation, ms	Digest sign with presign, ms	Digest verify sign, ms
GF(2^{163})	0,01	0,4272	0,6361	0,4426	0,8609
GF(2^{167})	0,01	0,6253	0,7913	0,4658	0,8377
GF(2^{173})	0,026	0,9339	0,6632	0,4684	0,4767
GF(2^{179})	0,0108	0,4836	0,4864	0,4876	0,4968
GF(2^{191})	0,0115	0,5237	0,5312	0,5325	0,5376
GF(2^{233})	0,0166	1,1073	1,1346	1,1363	1,1508
GF(2^{257})	0,0208	1,9075	1,9191	1,9218	1,936
GF(2^{307})	0,0244	2,3375	2,3499	2,3528	2,5512
GF(2^{367})	0,0339	4,1457	3,9485	3,9543	4,0333
GF(2^{431})	0,0452	6,6417	6,4386	6,4442	6,2556

В таблиці відображені результати середнього часу виконання 1 мільйону ітерації операцій в програмній реалізації.

Експериментальні дослідження методів арифметичних операцій в криптосистемі RSA

Експериментальні дослідження проводились для основних операцій в криптосистемі RSA:

- генерування особистого ключа (private key generation);
- створення підпису (digest sign);
- перевірка підпису (digest verify sign).

Значення швидкодії вказаних операцій в кільці цілих чисел зазначені в мілісекундах.

Першими будуть розглядатися результати отримані для платформи Server (Intel Xeon E2146G)

У таблицях 1 та 2 наведені результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа без та з запропонованими методами для платформи Server для 32-бітних машинних слів.

Таблиця 1. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів без застосування запропонованих методів (original)

Original (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	29,3333	1,00	0,18	4096	3	24473,00	529,00	11,65
	5	29,3333	1,00	0,18		5	30117,30	529,00	11,96
	17	28,3333	1,00	0,19		17	29492,30	522,00	11,25
	257	26,0000	1,00	0,20		257	26818,30	531,00	11,41
	65537	35,0000	1,00	0,21		65537	30267,00	534,00	12,48
	0	26,3333	1,00	0,25		0	50568,70	529,00	15,60
1024	3	321,667	9,00	0,71	7168	3	291343,00	2750,00	35,42
	5	271,667	9,00	0,71		5	379897,00	2745,00	35,18
	17	249,333	9,00	0,71		17	213870,00	2737,00	35,24
	257	236,667	9,00	0,75		257	263709,00	2746,00	36,65
	65537	196,667	9,00	0,77		65537	169688,00	2769,00	38,60
	0	257,000	9,00	1,06		0	161047,00	2742,00	49,70
1536	3	1122,00	30,00	1,64	8192	3	433724,00	4076,00	42,69
	5	989,00	30,00	1,68		5	560082,00	4081,00	45,77
	17	898,67	30,00	1,73		17	384302,00	4080,00	44,90
	257	1040,33	30,00	1,76		257	491371,00	4075,00	48,13
	65537	844,33	29,00	1,81		65537	362166,00	4078,00	52,22
	0	801,00	29,00	2,29		0	244084,00	4070,00	63,68
2048	3	2189,00	69,00	2,78	15360	3	5447980,00	26385,00	157,22
	5	2486,33	69,00	2,85		5	3160560,00	26389,00	159,54
	17	2215,67	78,00	2,98		17	7005980,00	27107,00	163,92
	257	2355,00	72,00	3,17		257	2792050,00	27167,00	168,79
	65537	2664,00	68,00	3,09		65537	2266640,00	27080,00	187,93
	0	3412,00	69,00	3,96		0	4175370,00	27148,00	225,31
3072	3	12431,70	226,00	6,43	16384	3	5718480,00	32789,00	182,14
	5	6179,67	231,00	6,57		5	4011790,00	32834,00	177,14
	17	16254,00	232,00	6,81		17	3364670,00	32843,00	181,71
	257	6924,67	229,00	6,77		257	5104150,00	32840,00	185,16
	65537	7550,67	234,00	7,45		65537	5617990,00	32854,00	205,11
	0	14665,00	231,00	8,94		0	3903290,00	32796,00	243,82

Таблиця 2. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа з застосуванням запропонованих методів для платформи Server з використанням 32-бітних машинних слів (with improvements)

With improvements (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	21,3333	0,10	0,17	4096	3	15551,00	310,00	11,23
	5	27,00	0,10	0,16		5	23026,00	312,00	11,73
	17	24,3333	0,10	0,18		17	22744,30	307,00	11,17
	257	19,3333	0,10	0,17		257	14809,30	305,00	11,14
	65537	19,3333	0,10	0,20		65537	13435,30	307,00	12,20
	0	21,3333	0,10	0,23		0	18738,00	308,00	13,61
1024	3	139,3330	5,00	0,66	7168	3	287182,00	1732,00	34,92
	5	193,333	5,00	0,7		5	178111,00	1735,00	34,14
	17	170,667	5,00	0,68		17	128634,00	1732,00	34,49
	257	157,667	5,00	0,7		257	134729,00	1722,00	35,97
	65537	193,667	5,00	0,76		65537	139689,00	1735,00	34,94
	0	179,667	5,00	0,82		0	72904,70	1720,00	43,75
1536	3	528,333	16,00	1,62	8192	3	426827,00	2242,00	42,18
	5	729,333	16,00	1,67		5	275564,00	2232,00	45,26
	17	526,667	16,00	1,71		17	332989,00	2233,00	44,37
	257	749,000	16,00	1,67		257	199602,00	2229,00	48,03
	65537	637,333	16,00	1,66		65537	178406,00	2281,00	47,12
	0	496,333	16,00	1,97		0	195496,00	2237,00	52,46
2048	3	1396,33	39,00	2,76	15360	3	1811140,00	14773,00	147,86
	5	1685,00	39,00	2,81		5	1193380,00	14729,00	158,56
	17	1334,33	38,00	2,81		17	2378480,00	14736,00	162,09
	257	1351,33	39,00	2,78		257	2056480,00	14723,00	157,49
	65537	2171,33	38,00	3,02		65537	1388520,00	14734,00	165,62
	0	1746,00	39,00	3,62		0	2352320,00	14726,00	190,77
3072	3	5776,67	126,00	6,38	16384	3	3321920,00	17813,00	172,84
	5	5791,67	127,00	6,44		5	3438670,00	17742,00	166,36
	17	4374,00	130,00	6,71		17	3168230,00	17761,00	171,27
	257	6840,33	127,00	6,42		257	2760750,00	17763,00	173,84
	65537	5388,00	128,00	6,92		65537	2476430,00	17818,00	187,09
	0	4904,33	128,00	7,89		0	3053580,00	17769,00	225,69

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 3 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-3,72 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція створення підпису ефективніша в 1,58-10,00 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,29 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.

Таблиця 3. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server за результатами без та з використанням запропонованих методів для 32-бітних машинних слів

Original / With improvements ($w=32$ bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,38	10,00	1,06	4096	3	1,57	1,71	1,04
	5	1,09	10,00	1,13		5	1,31	1,70	1,02
	17	1,16	10,00	1,06		17	1,30	1,70	1,01
	257	1,34	10,00	1,18		257	1,81	1,74	1,02
	65537	1,81	10,00	1,05		65537	2,25	1,74	1,02
	0	1,23	10,00	1,09		0	2,70	1,72	1,15
1024	3	2,31	1,80	1,08	7168	3	1,01	1,59	1,01
	5	1,41	1,80	1,01		5	2,13	1,58	1,03
	17	1,46	1,80	1,04		17	1,66	1,58	1,02
	257	1,50	1,80	1,07		257	1,96	1,59	1,02
	65537	1,02	1,80	1,01		65537	1,21	1,60	1,10
	0	1,43	1,80	1,29		0	2,21	1,59	1,14
1536	3	2,12	1,88	1,01	8192	3	1,02	1,82	1,01
	5	1,36	1,88	1,01		5	2,03	1,83	1,01
	17	1,71	1,88	1,01		17	1,15	1,83	1,01
	257	1,39	1,88	1,05		257	2,46	1,83	1,00
	65537	1,32	1,81	1,09		65537	2,03	1,79	1,11
	0	1,61	1,81	1,16		0	1,25	1,82	1,21
2048	3	1,57	1,77	1,01	15360	3	3,01	1,79	1,06
	5	1,48	1,77	1,01		5	2,65	1,79	1,01
	17	1,66	2,05	1,06		17	2,95	1,84	1,01
	257	1,74	1,85	1,14		257	1,36	1,85	1,07
	65537	1,23	1,79	1,02		65537	1,63	1,84	1,13
	0	1,95	1,77	1,09		0	1,78	1,84	1,18
3072	3	2,15	1,79	1,01	16384	3	1,72	1,84	1,05
	5	1,07	1,82	1,02		5	1,17	1,85	1,06
	17	3,72	1,78	1,01		17	1,06	1,85	1,06
	257	1,01	1,80	1,05		257	1,85	1,85	1,07
	65537	1,40	1,83	1,08		65537	2,27	1,84	1,10
	0	2,99	1,80	1,13		0	1,28	1,85	1,08

Далі, у таблиці 4, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 4. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads)

With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	14,6667	0,10	0,18	4096	3	11760,30	304,00	10,72
	5	14,6667	0,10	0,17		5	7455,00	298,00	10,75
	17	15,6667	0,10	0,17		17	10264,30	298,00	10,64
	257	10,3333	0,10	0,18		257	7769,67	299,00	11,04
	65537	9,3333	0,10	0,18		65537	9540,33	304,00	11,67
	0	11,6667	0,10	0,22		0	14250,70	302,00	12,90
1024	3	125,0	5,00	0,72	7168	3	80215,70	1671,00	33,73
	5	139,0	5,00	0,69		5	93932,70	1697,00	32,76
	17	100,0	5,00	0,74		17	138959,00	1691,00	33,78
	257	93,3333	5,00	0,71		257	76575,00	1695,00	34,55
	65537	101,6670	5,00	0,76		65537	44920,30	1681,00	34,59
	0	121,6670	5,00	0,90		0	56314,00	1682,00	42,25
1536	3	474,000	16,00	1,59	8192	3	84961,70	2136,00	41,84
	5	636,667	16,00	1,65		5	74511,00	2126,00	43,51
	17	388,667	16,00	1,67		17	83394,30	2145,00	43,65
	257	388,333	16,00	1,57		257	107548,00	2144,00	46,37
	65537	340,333	16,00	1,62		65537	244007,00	2135,00	46,56
	0	312,000	16,00	1,90		0	148673,00	2149,00	51,68
2048	3	1240,67	38,00	2,70	15360	3	704614,00	14631,00	144,16
	5	916,33	38,00	2,75		5	766829,00	14633,00	153,46
	17	879,67	37,00	2,76		17	720114,00	14650,00	156,76
	257	1035,00	38,00	2,76		257	1764310,00	14644,00	152,40
	65537	1362,00	37,00	2,94		65537	2283870,00	14641,00	164,23
	0	1452,33	38,00	3,51		0	1407600,00	14629,00	188,05
3072	3	3889,00	125,00	6,03	16384	3	986033,00	17479,00	166,70
	5	5354,33	125,00	6,39		5	2213590,00	17453,00	160,82
	17	3619,67	128,00	6,59		17	1300130,00	17460,00	162,60
	257	2468,67	126,00	6,41		257	1771600,00	17463,00	161,35
	65537	2631,33	126,00	6,72		65537	1060220,00	17486,00	178,04
	0	2113,00	125,00	7,52		0	482164,00	17479,00	213,22

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій криптосистеми RSA з використанням запропонованих методів (with improvements) до результатів вимірювання операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads).

У таблиці 5 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,08-6,33 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,07 рази в залежності від розміру відкритої експоненти для ключів довжиною 2048 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,09 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.

Таблиця 5. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads)

With improvements / With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,45	1,00	0,94	4096	3	1,32	1,02	1,05
	5	1,84	1,00	0,94		5	3,09	1,05	1,09
	17	1,55	1,00	1,06		17	2,22	1,03	1,05
	257	1,87	1,00	0,94		257	1,91	1,02	1,01
	65537	2,07	1,00	1,11		65537	1,41	1,01	1,05
	0	1,83	1,00	1,05		0	1,31	1,02	1,06
1024	3	1,11	1,00	0,92	7168	3	3,58	1,04	1,04
	5	1,39	1,00	1,01		5	1,90	1,02	1,04
	17	1,71	1,00	0,92		17	0,93	1,02	1,02
	257	1,69	1,00	0,99		257	1,76	1,02	1,04
	65537	1,90	1,00	1,00		65537	3,11	1,03	1,01
	0	1,48	1,00	0,91		0	1,29	1,02	1,04
1536	3	1,11	1,00	1,02	8192	3	5,02	1,05	1,01
	5	1,15	1,00	1,01		5	3,70	1,05	1,04
	17	1,36	1,00	1,02		17	3,99	1,04	1,02
	257	1,93	1,00	1,06		257	1,86	1,04	1,04
	65537	1,87	1,00	1,02		65537	0,73	1,07	1,01
	0	1,59	1,00	1,04		0	1,31	1,04	1,02
2048	3	1,13	1,03	1,02	15360	3	2,57	1,01	1,03
	5	1,84	1,03	1,02		5	1,56	1,01	1,03
	17	1,52	1,03	1,02		17	3,30	1,01	1,03
	257	1,31	1,03	1,01		257	1,17	1,01	1,03
	65537	1,59	1,03	1,03		65537	0,61	1,01	1,01
	0	1,20	1,03	1,03		0	1,67	1,01	1,01
3072	3	1,49	1,01	1,06	16384	3	3,37	1,02	1,04
	5	1,08	1,02	1,01		5	1,55	1,02	1,03
	17	1,21	1,02	1,02		17	2,44	1,02	1,05
	257	2,77	1,01	1,00		257	1,56	1,02	1,08
	65537	2,05	1,02	1,03		65537	2,34	1,02	1,05
	0	2,32	1,02	1,05		0	6,33	1,02	1,06

У таблиці 6 наведені результати вимірювання криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в 4 потоки з використанням 32-бітних машинних слів.

Таблиця 6. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	51,3333	1,00	0,18	4096	3	8084,00	165,00	10,39
	5	42,0000	1,00	0,19		5	2957,67	166,00	10,86
	17	48,6667	1,00	0,19		17	6077,33	166,00	10,42
	257	43,0000	1,00	0,20		257	4139,33	165,00	10,64
	65537	39,6667	1,00	0,22		65537	11932,30	166,00	11,04
	0	41,0000	1,00	0,29		0	7656,00	166,00	12,89
1024	3	230,6670	5,00	0,70	7168	3	57357,70	911,00	33,46
	5	173,000	5,00	0,72		5	56691,30	908,00	32,74
	17	207,000	5,00	0,76		17	29145,30	899,00	33,13
	257	211,000	5,00	0,89		257	40309,00	908,00	33,02
	65537	300,667	5,00	0,84		65537	32634,00	900,00	34,04
	0	179,333	5,00	0,87		0	55693,30	908,00	37,99
1536	3	376,000	11,00	1,55	8192	3	115616,00	1273,00	41,49
	5	435,333	12,00	1,64		5	115777,00	1269,00	43,36
	17	604,333	11,00	1,60		17	25255,70	1268,00	43,91
	257	472,000	12,00	1,64		257	128775,00	1271,00	46,49
	65537	546,333	11,00	1,62		65537	199737,00	1271,00	44,85
	0	639,333	11,00	1,95		0	82867,30	1368,00	52,00
2048	3	1129,67	25,00	2,73	15360	3	926071,00	7800,00	147,01
	5	782,67	25,00	2,75		5	924132,00	7750,00	148,33
	17	918,67	25,00	2,78		17	827839,00	7714,00	156,59
	257	888,33	24,00	2,73		257	721221,00	7894,00	156,60
	65537	734,00	24,00	2,96		65537	921537,00	7714,00	158,79
	0	1128,00	25,00	3,31		0	762524,00	7726,00	174,94
3072	3	2467,33	77,00	6,12	16384	3	1896640,00	9393,00	164,41
	5	3364,00	76,00	6,01		5	1295930,00	9398,00	164,39
	17	2511,33	78,00	6,50		17	713671,00	9462,00	168,07
	257	2789,33	78,00	6,25		257	1388400,00	9400,00	170,85
	65537	3857,00	76,00	6,45		65537	1261930,00	9430,00	182,12
	0	4961,33	78,00	7,00		0	1377200,00	9440,00	190,90

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням у 4 потоки (with improvements and 4 threads).

У таблиці 7 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням у 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,13-13,18 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,33-1,93 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,18 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.

Таблиця 7. Нормалізовані результати вимірювання криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,42	0,10	0,94	4096	3	1,92	1,88	1,08
	5	0,64	0,10	0,84		5	7,79	1,88	1,08
	17	0,50	0,10	0,95		17	3,74	1,85	1,07
	257	0,45	0,10	0,85		257	3,58	1,85	1,05
	65537	0,49	0,10	0,91		65537	1,13	1,85	1,11
	0	0,52	0,10	0,79		0	2,45	1,86	1,06
1024	3	0,60	1,00	0,94	7168	3	5,01	1,90	1,04
	5	1,12	1,00	0,97		5	3,14	1,91	1,04
	17	0,82	1,00	0,89		17	4,41	1,93	1,04
	257	0,75	1,00	0,79		257	3,34	1,90	1,09
	65537	0,64	1,00	0,90		65537	4,28	1,93	1,03
	0	1,00	1,00	0,94		0	1,31	1,89	1,15
1536	3	1,41	1,45	1,05	8192	3	3,69	1,76	1,02
	5	1,68	1,33	1,02		5	2,38	1,76	1,04
	17	0,87	1,45	1,07		17	13,18	1,76	1,01
	257	1,59	1,33	1,02		257	1,55	1,75	1,03
	65537	1,17	1,45	1,02		65537	0,89	1,79	1,05
	0	0,78	1,45	1,01		0	2,36	1,64	1,01
2048 bit	3	1,24	1,56	1,01	15360 bit	3	1,96	1,89	1,01
	5	2,15	1,56	1,02		5	1,29	1,90	1,07
	17	1,45	1,52	1,01		17	2,87	1,91	1,04
	257	1,52	1,63	1,02		257	2,85	1,87	1,01
	65537	2,96	1,58	1,02		65537	1,51	1,91	1,04
	0	1,55	1,56	1,09		0	3,08	1,91	1,09
3072	3	2,34	1,64	1,04	16384	3	1,75	1,90	1,05
	5	1,72	1,67	1,07		5	2,65	1,89	1,01
	17	1,74	1,67	1,03		17	4,44	1,88	1,02
	257	2,45	1,63	1,03		257	1,99	1,89	1,02
	65537	1,40	1,68	1,07		65537	1,96	1,89	1,03
	0	0,99	1,64	1,13		0	2,22	1,88	1,18

У таблиці 8 наведені результати вимірювання швидкодії операцій криптосистеми RSA з запропонованими методами та розпаралелюванням у декілька потоків для 32-бітних машинних слів.

Таблиця 8. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа з запропонованими методами та розпаралелюванням в декілька потоків (with improvements and multi threads) для платформи Server з використанням 32-бітних машинних слів

With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	264,667	4,00	0,24	4096	3	9227,00	124,00	12,91
	5	297,667	4,00	0,25		5	11929,00	75,00	13,14
	17	302,667	3,00	0,26		17	19172,00	124,00	13,45
	257	358,333	4,00	0,28		257	16409,30	124,00	13,14
	65537	299,333	2,00	0,28		65537	13791,70	123,00	13,09
	0	380,667	4,00	0,54		0	15684,70	82,00	13,47
1024	3	605,667	9,00	0,96	7168	3	36301,00	496,00	37,33
	5	693,667	8,00	0,84		5	44539,00	495,00	39,99
	17	1039,670	9,00	0,89		17	113874,00	505,00	37,73
	257	607,333	9,00	0,91		257	65105,00	498,00	37,41
	65537	686,667	9,00	0,97		65537	70778,00	491,00	38,28
	0	592,333	8,00	1,00		0	65577,30	431,00	40,78
1536	3	1672,33	17,00	1,87	8192	3	33499,00	720,00	48,60
	5	1859,67	17,00	2,00		5	63273,00	733,00	47,56
	17	1736,67	18,00	1,91		17	43638,00	730,00	48,41
	257	2226,0	17,00	2,04		257	29904,30	425,00	49,45
	65537	2249,33	17,00	2,24		65537	30239,70	715,00	50,39
	0	1859,33	17,00	2,32		0	71181,70	728,00	52,35
2048	3	1993,33	29,00	3,16	15360	3	775832,00	3818,00	175,27
	5	2856,00	27,00	3,55		5	451960,00	3838,00	174,64
	17	2681,00	27,00	3,28		17	320201,00	3724,00	173,95
	257	2507,67	17,00	3,31		257	579163,00	3672,00	169,31
	65537	2482,33	30,00	3,51		65537	302176,00	4019,00	177,00
	0	2981,33	28,00	3,83		0	287975,00	4152,00	177,71
3072	3	8173,00	61,00	7,39	16384	3	1366290,00	4708,00	185,96
	5	3276,67	63,00	7,50		5	1160030,00	4641,00	177,69
	17	5206,00	35,00	7,15		17	1166820,00	4659,00	180,92
	257	4379,33	62,00	7,47		257	1293290,00	4660,00	181,10
	65537	2784,33	60,00	7,67		65537	1562040,00	4660,00	187,71
	0	3549,67	59,00	7,86		0	1372750,00	4706,00	201,62

Для оцінки ефективності запропонованих методів для платформи Server з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 9 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,11-12,74 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,26-5,24 рази в залежності від розміру відкритої експоненти для ключів довжиною 2048 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,12 рази лише для відкритої експоненти 0 для ключів довжиною 4096 біт та більше.

Таблиця 9. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у декілька потоки (with improvements and multi threads)

With improvements / With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,08	0,03	0,71	4096	3	1,69	2,50	0,87
	5	0,09	0,03	0,64		5	1,93	4,16	0,89
	17	0,08	0,03	0,69		17	1,19	2,48	0,83
	257	0,05	0,03	0,61		257	0,90	2,46	0,85
	65537	0,06	0,05	0,71		65537	0,97	2,50	0,93
	0	0,06	0,03	0,43		0	1,19	3,76	1,01
1024	3	0,23	0,56	0,69	7168	3	7,91	3,49	0,94
	5	0,28	0,63	0,83		5	4,00	3,51	0,85
	17	0,16	0,56	0,76		17	1,13	3,43	0,91
	257	0,26	0,56	0,77		257	2,07	3,46	0,96
	65537	0,28	0,56	0,78		65537	1,97	3,53	0,91
	0	0,30	0,63	0,82		0	1,11	3,99	1,07
1536	3	0,32	0,94	0,87	8192	3	12,74	3,11	0,87
	5	0,39	0,94	0,84		5	4,36	3,05	0,95
	17	0,30	0,89	0,90		17	7,63	3,06	0,92
	257	0,34	0,94	0,82		257	6,67	5,24	0,97
	65537	0,28	0,94	0,74		65537	5,90	3,19	0,94
	0	0,27	0,94	0,85		0	2,75	3,07	1,00
2048	3	0,70	1,34	0,87	15360	3	2,33	3,87	0,84
	5	0,59	1,44	0,79		5	2,64	3,84	0,91
	17	0,50	1,41	0,86		17	7,43	3,96	0,93
	257	0,54	2,29	0,84		257	3,55	4,01	0,93
	65537	0,87	1,27	0,86		65537	4,60	3,67	0,94
	0	0,59	1,39	0,95		0	8,17	3,55	1,07
3072	3	0,71	2,07	0,86	16384	3	2,43	3,78	0,93
	5	1,77	2,02	0,86		5	2,96	3,82	0,94
	17	0,84	3,71	0,94		17	2,72	3,81	0,95
	257	1,56	2,05	0,86		257	2,13	3,81	0,96
	65537	1,94	2,13	0,90		65537	1,59	3,82	1,00
	0	1,38	2,17	1,00		0	2,22	3,78	1,12

Далі розглядаються результати вимірювання для 64-бітних машинних слів.

У таблиці 10 і 11 наведені результати вимірювання швидкодії операцій криптосистеми RSA без запропонованих методів для 64-бітних машинних слів

Таблиця 10. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа без застосування запропонованих методів (original) для платформи Server з використанням 64-бітних машинних слів

Original (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	6,00	0,01	0,06	4096	3	2050,00	53,00	2,82
	5	5,00	0,01	0,06		5	2694,33	53,00	2,87
	17	5,00	0,01	0,06		17	2716,00	53,00	2,89
	257	6,33	0,01	0,06		257	3159,00	54,00	2,92
	65537	5,00	0,01	0,06		65537	2101,33	54,00	3,04
	0	6,67	0,01	0,07		0	3201,33	53,00	4,01
1024	3	38,6667	0,01	0,21	7168	3	24339,00	272,00	8,74
	5	43,6667	0,01	0,21		5	21011,70	274,00	8,75
	17	26,3333	0,01	0,21		17	25215,30	272,00	8,91
	257	27,00	0,01	0,21		257	18585,70	277,00	9,04
	65537	38,6667	0,01	0,21		65537	38558,30	271,00	9,36
	0	25,3333	0,01	0,27		0	21187,70	269,00	11,98
1536	3	143,333	2,00	0,42	8192	3	26256,00	409,00	11,54
	5	98,333	2,00	0,43		5	49101,30	403,00	11,60
	17	114,333	3,00	0,45		17	55254,00	417,00	11,51
	257	136,667	3,00	0,45		257	40965,00	401,00	11,70
	65537	87,6667	3,00	0,46		65537	28063,70	403,00	12,08
	0	98,6667	2,00	0,58		0	58209,30	405,00	15,73
2048	3	293,333	7,00	0,73	15360	3	241217,00	2529,00	39,46
	5	279,000	6,00	0,72		5	354104,00	2548,00	39,37
	17	225,333	7,00	0,73		17	382408,00	2541,00	40,08
	257	269,667	6,00	0,76		257	519327,00	2532,00	40,54
	65537	177,333	6,00	0,77		65537	276820,00	2543,00	41,12
	0	307,333	6,00	0,99		0	287454,00	2544,00	53,68
3072	3	1513,00	23,00	1,67	16384	3	399786,00	3075,00	45,05
	5	878,667	23,00	1,68		5	375260,00	3075,00	45,42
	17	1151,00	23,00	1,67		17	184295,00	3078,00	45,88
	257	906,00	23,00	1,68		257	958198,00	3079,00	46,26
	65537	1138,67	22,00	1,73		65537	989078,00	3071,00	47,62
	0	1157,67	23,00	2,28		0	613890,00	3086,00	61,54

Таблиця 11. Результати вимірювання швидкодії арифметичних операцій криптосистеми RSA для різної довжини ключа з застосуванням запропонованих методів арифметичних операцій (with improvements) для платформи Server з використанням 64-бітних машинних слів

With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	5,67	0,01	0,05	4096	3	2446,33	52,00	2,74
	5	4,33	0,01	0,05		5	2062,33	52,00	2,82
	17	4,33	0,01	0,05		17	3248	51,00	2,86
	257	4,67	0,01	0,05		257	1561,67	52,00	2,81
	65537	4,70	0,01	0,05		65537	2252	52,00	3
	0	5,33	0,01	0,06		0	2051,33	52,00	3,91
1024	3	38,00	0,01	0,20	7168	3	15687	264,00	8,36
	5	38,00	0,01	0,20		5	16152,7	269,00	8,41
	17	36,00	0,01	0,21		17	31717,7	264,00	8,8
	257	40,67	0,01	0,21		257	15686,7	271,00	8,84
	65537	29,33	0,01	0,21		65537	17619,3	270,00	9,05
	0	42,00	0,01	0,27		0	31254,3	268,00	11,76
1536	3	139	2,00	0,43	8192	3	28644,7	390,00	11,41
	5	105,333	2,00	0,43		5	36335,7	396,00	11,35
	17	149,333	3,00	0,44		17	42892,3	396,00	11,31
	257	184,667	3,00	0,46		257	25971,7	392,00	11,28
	65537	103,333	2,00	0,46		65537	28372	397,00	12
	0	114,333	2,00	0,57		0	52313,3	395,00	15,48
2048	3	456	6,00	0,74	15360	3	351931	2481,00	38,15
	5	315,333	7,00	0,74		5	387939	2480,00	38,66
	17	242	6,00	0,73		17	367667	2482,00	39,06
	257	210	6,00	0,73		257	618933	2485,00	39,71
	65537	304,667	6,00	0,77		65537	541226	2491,00	40,38
	0	233,667	7,00	0,99		0	328153	2480,00	52,25
3072	3	773	22,00	1,6	16384	3	309212	2995,00	44,8
	5	730,333	22,00	1,59		5	885618	3000,00	44,6
	17	1056	22,00	1,65		17	512454	3000,00	44,98
	257	913	22,00	1,59		257	811495	3006,00	45,5
	65537	924,667	21,00	1,7		65537	526275	2991,00	47,33
	0	944	21,00	2,15		0	499913	3069,00	61

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів вимірювання арифметичних операцій в криптосистемі RSA з використанням запропонованих методів (with improvements).

У таблиці 12 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-2,19 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,10 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,17-1,2 рази в залежності від розміру відкритої експоненти для ключів довжиною 512 біт, та в 1,01-1,06 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 12. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Original / With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	1,06	1,00	1,20	4096	3	0,84	1,02	1,03
	5	1,15	1,00	1,20		5	1,31	1,02	1,02
	17	1,15	1,00	1,20		17	0,84	1,04	1,01
	257	1,36	1,00	1,20		257	2,02	1,04	1,04
	65537	1,06	1,00	1,20		65537	0,93	1,04	1,01
	0	1,25	1,00	1,17		0	1,56	1,02	1,03
1024	3	1,02	1,00	1,05	7168	3	1,55	1,03	1,05
	5	1,15	1,00	1,05		5	1,30	1,02	1,04
	17	0,73	1,00	1,00		17	0,79	1,03	1,01
	257	0,66	1,00	1,00		257	1,18	1,02	1,02
	65537	1,32	1,00	1,00		65537	2,19	1,00	1,03
	0	0,60	1,00	1,00		0	0,68	1,00	1,02
1536	3	1,03	1,00	0,98	8192	3	0,92	1,05	1,01
	5	0,93	1,00	1,00		5	1,35	1,02	1,02
	17	0,77	1,00	1,02		17	1,29	1,05	1,02
	257	0,74	1,00	0,98		257	1,58	1,02	1,04
	65537	0,85	1,50	1,00		65537	0,99	1,02	1,01
	0	0,86	1,00	1,02		0	1,11	1,03	1,02
2048	3	0,64	1,17	0,99	15360	3	0,69	1,02	1,03
	5	0,88	0,86	0,97		5	0,91	1,03	1,02
	17	0,93	1,17	1,00		17	1,04	1,02	1,03
	257	1,28	1,00	1,04		257	0,84	1,02	1,02
	65537	0,58	1,00	1,00		65537	0,51	1,02	1,02
	0	1,32	0,86	1,00		0	0,88	1,03	1,03
3072	3	1,96	1,05	1,04	16384	3	1,29	1,03	1,01
	5	1,20	1,05	1,06		5	0,42	1,03	1,02
	17	1,09	1,05	1,01		17	0,36	1,03	1,02
	257	0,99	1,05	1,06		257	1,18	1,02	1,02
	65537	1,23	1,05	1,02		65537	1,88	1,03	1,01
	0	1,23	1,10	1,06		0	1,23	1,01	1,01

Далі, у таблиці 13, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 13. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Server з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads).

With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	3,66667	0,01	0,05	4096	3	2049,33	50,00	2,67
	5	2,33333	0,01	0,05		5	3322,00	51,00	2,71
	17	4,00000	0,01	0,05		17	1692,00	50,00	2,70
	257	3,00000	0,01	0,06		257	3017,33	50,00	2,74
	65537	3,66667	0,01	0,05		65537	2475,33	50,00	2,95
	0	4,33333	0,01	0,08		0	1422,00	51,00	3,80
1024	3	20,3333	1,00	0,21	7168	3	28185,00	255,00	8,16
	5	23,000	1,00	0,21		5	18961,30	255,00	8,18
	17	22,3333	1,00	0,21		17	9744,33	256,00	8,66
	257	17,000	1,00	0,22		257	17414,00	266,00	8,77
	65537	20,3333	1,00	0,22		65537	20251,70	265,00	8,81
	0	16,000	1,00	0,29		0	19783,00	265,00	11,10
1536	3	73,6667	4,00	0,66	8192	3	17644,70	387,00	11,05
	5	100,333	3,00	0,45		5	17406,00	390,00	11,17
	17	47,333	2,00	0,44		17	33221,30	389,00	11,13
	257	60,667	2,00	0,45		257	27044,00	390,00	11,11
	65537	46,000	3,00	0,47		65537	57442,00	390,00	11,60
	0	72,000	3,00	0,60		0	35225,30	392,00	14,76
2048	3	162,00	6,00	0,74	15360	3	588489,00	2458,00	37,57
	5	140,667	6,00	0,77		5	413543,00	2438,00	38,16
	17	173,00	6,00	0,74		17	364142,00	2448,00	38,69
	257	310,667	7,00	0,76		257	480111,00	2447,00	39,00
	65537	161,667	7,00	0,78		65537	280333,00	2450,00	39,49
	0	230,00	7,00	1,02		0	286881,00	2448,00	51,61
3072	3	646,667	22,00	1,68	16384	3	345925,00	2900,00	44,34
	5	570,00	23,00	1,67		5	390326,00	2914,00	44,24
	17	731,667	22,00	1,67		17	589125,00	2900,00	44,47
	257	1593,33	23,00	1,74		257	454421,00	2900,00	44,99
	65537	519,00	24,00	1,81		65537	281966,00	2905,00	45,18
	0	571,33	23,00	2,28		0	452144,00	2904,00	58,95

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в два потоки (with improvements and 2 threads).

У таблиці 14 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-3,25 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,06 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,06 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.

Таблиця 14. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,55	1,00	1,00	4096	3	1,19	1,04	1,03
	5	1,86	1,00	1,00		5	0,62	1,02	1,04
	17	1,08	1,00	1,00		17	1,92	1,02	1,06
	257	1,56	1,00	0,83		257	0,52	1,04	1,03
	65537	1,28	1,00	1,00		65537	0,91	1,04	1,02
	0	1,23	1,00	0,75		0	1,44	1,02	1,03
1024	3	1,87	0,01	0,95	7168	3	0,56	1,04	1,02
	5	1,65	0,01	0,95		5	0,85	1,05	1,03
	17	1,61	0,01	1,00		17	3,25	1,03	1,02
	257	2,39	0,01	0,95		257	0,90	1,02	1,01
	65537	1,44	0,01	0,95		65537	0,87	1,02	1,03
	0	2,63	0,01	0,93		0	1,58	1,01	1,06
1536	3	1,89	0,50	0,65	8192	3	1,62	1,01	1,03
	5	1,05	0,67	0,96		5	2,09	1,02	1,02
	17	3,15	1,50	1,00		17	1,29	1,02	1,02
	257	3,04	1,50	1,02		257	0,96	1,01	1,02
	65537	2,25	0,67	0,98		65537	0,49	1,02	1,03
	0	1,59	0,67	0,95		0	1,49	1,01	1,05
2048	3	2,81	1,00	1,00	15360	3	0,60	1,01	1,02
	5	2,24	1,17	0,96		5	0,94	1,02	1,01
	17	1,40	1,00	0,99		17	1,01	1,01	1,01
	257	0,68	0,86	0,96		257	1,29	1,02	1,02
	65537	1,88	0,86	0,99		65537	1,93	1,02	1,02
	0	1,02	1,00	0,97		0	1,14	1,01	1,01
3072	3	1,20	1,00	0,95	16384	3	0,89	1,03	1,01
	5	1,28	0,96	0,95		5	2,27	1,03	1,01
	17	1,44	1,00	0,99		17	0,87	1,03	1,01
	257	0,57	0,96	0,91		257	1,79	1,04	1,01
	65537	1,78	0,88	0,94		65537	1,87	1,03	1,05
	0	1,65	0,91	0,94		0	1,11	1,06	1,03

У таблиці 15 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням у декілька потоків для 64-бітних машинних слів.

Таблиця 15. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Server з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в чотири потоки (with improvements and 4 threads).

With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	40,0000	0,01	0,06	4096	3	1932,67	31,00	2,85
	5	33,00	0,01	0,06		5	1847,00	32,00	2,94
	17	40,0000	0,01	0,07		17	2117,00	32,00	2,93
	257	42,3333	0,01	0,07		257	2576,00	32,00	2,94
	65537	33,3333	0,01	0,08		65537	1691,67	32,00	3,03
	0	38,6667	0,01	0,18		0	1947,67	31,00	3,52
1024	3	137,3330	2,00	0,21	7168	3	11264,00	147,00	8,91
	5	104,667	2,00	0,22		5	8828,33	146,00	8,86
	17	95,333	2,00	0,22		17	9539,67	145,00	8,77
	257	86,000	2,00	0,23		257	7684,67	147,00	8,86
	65537	116,333	2,00	0,24		65537	9158,00	143,00	9,17
	0	93,667	2,00	0,38		0	18402,30	148,00	10,74
1536	3	294,000	4,00	0,45	8192	3	22135,30	213,00	11,52
	5	198,333	4,00	0,46		5	23888,70	208,00	11,70
	17	269,667	4,00	0,45		17	13868,30	212,00	11,59
	257	312,667	3,00	0,47		257	18580,30	212,00	11,51
	65537	218,667	3,00	0,49		65537	27696,00	211,00	12,06
	0	199,667	4,00	0,64		0	20327,30	208,00	13,62
2048	3	358,333	7,00	0,74	15360	3	91317,00	1240,00	40,32
	5	351,333	7,00	0,75		5	217655,00	1227,00	40,01
	17	386,333	6,00	0,75		17	258068,00	1231,00	39,65
	257	384,00	7,00	0,76		257	115518,00	1229,00	40,29
	65537	323,667	6,00	0,78		65537	342500,00	1224,00	41,40
	0	527,333	6,00	1,01		0	278730,00	1237,00	48,45
3072	3	977,67	17,00	1,66	16384	3	275744,00	1538,00	46,02
	5	1413,00	17,00	1,70		5	320889,00	1557,00	45,89
	17	889,33	17,00	1,66		17	121490,00	1594,00	45,66
	257	603,67	16,00	1,69		257	101642,00	1541,00	46,23
	65537	1059,67	17,00	1,74		65537	180189,00	1545,00	47,03
	0	622,00	16,00	2,10		0	157186,00	1567,00	53,63

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання швидкодії операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в чотири потоки (with improvements and 4 threads).

У таблиці 16 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-7,98 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,17-2,04 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,14 рази для деяких відкритих експонент для ключів довжиною 3072 біт та більше.

Таблиця 16. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,14	1,00	0,83	4096	3	1,27	1,68	0,96
	5	0,13	1,00	0,83		5	1,12	1,63	0,96
	17	0,11	1,00	0,71		17	1,53	1,59	0,98
	257	0,11	1,00	0,71		257	0,61	1,63	0,96
	65537	0,14	1,00	0,63		65537	1,33	1,63	0,99
	0	0,14	1,00	0,33		0	1,05	1,68	1,11
1024	3	0,28	0,01	0,95	7168	3	1,39	1,80	0,94
	5	0,36	0,01	0,91		5	1,83	1,84	0,95
	17	0,38	0,01	0,95		17	3,32	1,82	1,00
	257	0,47	0,01	0,91		257	2,04	1,84	1,00
	65537	0,25	0,01	0,88		65537	1,92	1,89	0,99
	0	0,45	0,01	0,71		0	1,70	1,81	1,09
1536	3	0,47	0,50	0,96	8192	3	1,29	1,83	0,99
	5	0,53	0,50	0,93		5	1,52	1,90	0,97
	17	0,55	0,75	0,98		17	3,09	1,87	0,98
	257	0,59	1,00	0,98		257	1,40	1,85	0,98
	65537	0,47	0,67	0,94		65537	1,02	1,88	1,00
	0	0,57	0,50	0,89		0	2,57	1,90	1,14
2048	3	1,27	0,86	1,00	15360	3	3,85	2,00	0,95
	5	0,90	1,00	0,99		5	1,78	2,02	0,97
	17	0,63	1,00	0,97		17	1,42	2,02	0,99
	257	0,55	0,86	0,96		257	5,36	2,02	0,99
	65537	0,94	1,00	0,99		65537	1,58	2,04	0,98
	0	0,44	1,17	0,98		0	1,18	2,00	1,08
3072	3	0,79	1,29	0,96	16384	3	1,12	1,95	0,97
	5	0,52	1,29	0,94		5	2,76	1,93	0,97
	17	1,19	1,29	0,99		17	4,22	1,88	0,99
	257	1,51	1,38	0,94		257	7,98	1,95	0,98
	65537	0,87	1,24	0,98		65537	2,92	1,94	1,01
	0	1,52	1,31	1,02		0	3,18	1,96	1,14

У таблиці 17 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів.

Таблиця 17. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Server з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads).

With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	284,3330	3,00	0,10	4096	3	5860,00	33,00	4,43
	5	303,33	1,00	0,09		5	5325,67	43,00	4,96
	17	316,0000	2,00	0,11		17	9958,00	43,00	4,73
	257	319,3330	1,00	0,11		257	11790,70	42,00	4,29
	65537	293,0000	1,00	0,13		65537	9711,33	31,00	4,40
	0	280,6670	1,00	0,31		0	7948,00	27,00	4,99
1024	3	625,00	3,00	0,33	7168	3	20100,70	120,00	13,21
	5	778,333	6,00	0,33		5	20808,70	73,00	13,05
	17	880,667	3,00	0,34		17	43979,00	117,00	13,24
	257	611,00	3,00	0,35		257	22994,30	71,00	13,28
	65537	607,333	3,00	0,37		65537	23898,00	72,00	14,52
	0	682,333	3,00	0,60		0	26560,30	114,00	14,23
1536	3	1262,330	6,00	0,72	8192	3	13457,30	102,00	17,37
	5	1249,330	6,00	0,72		5	33247,70	169,00	17,32
	17	1513,000	6,00	0,72		17	8517,00	102,00	17,54
	257	1576,00	10,00	0,69		257	23320,70	104,00	17,43
	65537	1511,00	6,00	0,74		65537	10855,00	108,00	17,28
	0	1299,67	6,00	1,00		0	29979,70	108,00	19,36
2048	3	2515,00	9,00	1,14	15360	3	92843,30	720,00	59,09
	5	2499,67	9,00	1,18		5	120065,00	548,00	60,53
	17	3082,00	15,00	1,15		17	101964,00	477,00	57,80
	257	2809,33	10,00	1,15		257	91764,00	576,00	58,77
	65537	3264,67	15,00	1,28		65537	94705,00	799,00	63,76
	0	3106,00	15,00	1,75		0	97008,30	635,00	61,38
3072	3	5516,33	24,00	2,90	16384	3	218756,00	571,00	72,98
	5	2956,00	25,00	2,47		5	257561,00	702,00	72,03
	17	2157,67	15,00	2,47		17	353571,00	571,00	66,88
	257	2613,33	16,00	2,51		257	292403,00	831,00	68,44
	65537	2615,00	24,00	2,64		65537	261503,00	937,00	67,94
	0	2924,67	23,00	3,40		0	251773,00	836,00	72,76

Для оцінки ефективності запропонованих методів для платформи Server з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання швидкодії операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 18 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,09-6,74 рази в залежності від розміру відкритої експоненти для ключів довжиною 7168 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,19-5,25 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 18. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Server за результатами з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів

With improvements / With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,02	0,00	0,50	4096	3	0,42	1,58	0,62
	5	0,01	0,01	0,56		5	0,39	1,21	0,57
	17	0,01	0,01	0,45		17	0,33	1,19	0,60
	257	0,01	0,01	0,45		257	0,13	1,24	0,66
	65537	0,02	0,01	0,38		65537	0,23	1,68	0,68
	0	0,02	0,01	0,19		0	0,26	1,93	0,78
1024	3	0,06	0,00	0,61	7168	3	0,78	2,20	0,63
	5	0,05	0,00	0,61		5	0,78	3,68	0,64
	17	0,04	0,00	0,62		17	0,72	2,26	0,66
	257	0,07	0,00	0,60		257	0,68	3,82	0,67
	65537	0,05	0,00	0,57		65537	0,74	3,75	0,62
	0	0,06	0,00	0,45		0	1,18	2,35	0,83
1536	3	0,11	0,33	0,60	8192	3	2,13	3,82	0,66
	5	0,08	0,33	0,60		5	1,09	2,34	0,66
	17	0,10	0,50	0,61		17	5,04	3,88	0,64
	257	0,12	0,30	0,67		257	1,11	3,77	0,65
	65537	0,07	0,33	0,62		65537	2,61	3,68	0,69
	0	0,09	0,33	0,57		0	1,74	3,66	0,80
2048	3	0,18	0,67	0,65	15360	3	3,79	3,45	0,65
	5	0,13	0,78	0,63		5	3,23	4,53	0,64
	17	0,08	0,40	0,63		17	3,61	5,20	0,68
	257	0,07	0,60	0,63		257	6,74	4,31	0,68
	65537	0,09	0,40	0,60		65537	5,71	3,12	0,63
	0	0,08	0,47	0,57		0	3,38	3,91	0,85
3072	3	0,14	0,92	0,55	16384	3	1,41	5,25	0,61
	5	0,25	0,88	0,64		5	3,44	4,27	0,62
	17	0,49	1,47	0,67		17	1,45	5,25	0,67
	257	0,35	1,38	0,63		257	2,78	3,62	0,66
	65537	0,35	0,88	0,64		65537	2,01	3,19	0,70
	0	0,32	0,91	0,63		0	1,99	3,67	0,84

Далі будуть розглядатися результати отримані для платформи Desktop (Intel Core-i7 4770)

У таблицях 19 та 20 наведені результати вимірювання швидкодії операції криптосистеми для різної довжини ключа без та з запропонованими методами для платформи Desktop для 32-бітних машинних слів.

Таблиця 19. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів без застосування запропонованих методів (original)

Original (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	42,6667	1,00	0,23	4096	3	33151,30	681,00	14,86
	5	43,6667	1,00	0,24		5	62648,30	687,00	15,12
	17	31,0000	1,00	0,24		17	45215,70	684,00	15,20
	257	35,0000	1,00	0,24		257	38085,00	683,00	15,45
	65537	34,6667	1,00	0,26		65537	34403,00	682,00	16,30
	0	47,3333	1,00	0,34		0	33779,00	682,00	20,84
1024	3	295,667	12,00	0,93	7168	3	260557,00	3585,00	44,79
	5	272,000	12,00	0,96		5	317110,00	3594,00	46,18
	17	258,333	12,00	1,00		17	371711,00	3589,00	46,84
	257	422,000	11,00	1,04		257	154296,00	3593,00	45,11
	65537	282,333	12,00	1,09		65537	285870,00	3590,00	52,27
	0	307,000	12,00	1,41		0	260090,00	3595,00	63,47
1536	3	928,333	39,00	2,12	8192	3	668125,00	5289,00	58,11
	5	988,330	38,00	2,13		5	508197,00	5280,00	59,85
	17	997,000	38,00	2,16		17	747220,00	5290,00	57,74
	257	976,667	38,00	2,28		257	632363,00	5288,00	63,14
	65537	989,333	38,00	2,44		65537	485208,00	5285,00	66,49
	0	1102,330	38,00	2,97		0	420594,00	5285,00	80,03
2048	3	4293,00	90,00	3,55	15360	3	5299680,00	34398,00	205,75
	5	3305,33	91,00	3,63		5	6665460,00	34391,00	206,93
	17	2831,00	89,00	3,76		17	6532360,00	34424,00	208,67
	257	3227,00	89,00	4,06		257	7061080,00	34428,00	212,40
	65537	2788,67	90,00	4,42		65537	2906540,00	34565,00	221,81
	0	3603,67	90,00	5,18		0	3403650,00	34510,00	292,95
3072	3	13439,00	298,00	8,37	16384	3	5688360,00	41666,00	224,65
	5	9950,67	297,00	8,00		5	5654230,00	41747,00	228,50
	17	9299,67	301,00	8,87		17	9103770,00	41651,00	244,08
	257	11292,30	297,00	8,98		257	7368040,00	41639,00	238,27
	65537	11738,30	301,00	9,58		65537	7050100,00	41649,00	270,50
	0	10410,30	297,00	11,49		0	5947870,00	41747,00	319,10

Таблиця 20. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів (with improvements)

With improvements (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	28,3333	1,00	0,22	4096	3	16059,30	459,00	13,71
	5	26,3333	1,00	0,23		5	23316,70	459,00	14,86
	17	25,0000	1,00	0,23		17	15479,30	459,00	13,98
	257	29,3333	1,00	0,23		257	23840,00	460,00	15,41
	65537	31,3333	1,00	0,24		65537	24687,70	460,00	16,00
	0	31,3333	1,00	0,29		0	16382,00	458,00	18,20
1024	3	230,6670	7,00	0,92	7168	3	197298,00	2384,00	42,99
	5	172,333	7,00	0,9		5	80987,00	2387,00	44,22
	17	185,667	7,00	0,99		17	147328,00	2386,00	44,88
	257	163,000	7,00	0,94		257	117184,00	2379,00	44,32
	65537	200,667	7,00	1,05		65537	150305,00	2384,00	46,62
	0	196,000	7,00	1,21		0	224938,00	2385,00	56,79
1536	3	695,000	21,00	2,04	8192	3	391616,00	2883,00	53,83
	5	576,333	21,00	2,06		5	347564,00	2884,00	54,03
	17	700,667	21,00	2,09		17	269033,00	2882,00	57,42
	257	870,667	21,00	2,20		257	502720,00	2886,00	56,94
	65537	910,000	22,00	2,10		65537	280869,00	2887,00	62,44
	0	657,667	24,00	2,56		0	341487,00	2887,00	69,84
2048	3	1807,33	49,00	3,43	15360	3	2652520,00	22745,00	200,14
	5	1459,33	49,00	3,59		5	1471060,00	22727,00	200,30
	17	2451,67	49,00	3,75		17	3422280,00	22758,00	204,79
	257	2662,33	49,00	3,95		257	4268640,00	22806,00	208,90
	65537	1380,67	49,00	4,03		65537	1123170,00	22736,00	209,16
	0	2261,33	49,00	4,71		0	3014950,00	22722,00	247,38
3072	3	8291,67	165,00	8,16	16384	3	2750170,00	27510,00	216,97
	5	7793,67	165,00	7,75		5	4366720,00	27511,00	222,57
	17	5334,00	165,00	8,40		17	4010910,00	27536,00	224,23
	257	8797,67	166,00	8,69		257	4765820,00	27553,00	231,87
	65537	8368,67	165,00	8,81		65537	5526730,00	27606,00	256,62
	0	8122,00	165,00	10,09		0	2767710,00	27560,00	301,37

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 21 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,09-4,53 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція створення підпису ефективніша в 1,48-1,86 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція перевірки підпису ефективніша в 1,01-1,18 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.

Таблиця 21. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop за результатами без та з використанням запропонованих методів для 32-бітних машинних слів

Original / With improvements (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,51	1,00	1,05	4096	3	2,06	1,48	1,08
	5	1,66	1,00	1,04		5	2,69	1,50	1,02
	17	1,24	1,00	1,04		17	2,92	1,49	1,09
	257	1,19	1,00	1,04		257	1,60	1,48	1,00
	65537	1,11	1,00	1,08		65537	1,39	1,48	1,02
	0	1,51	1,00	1,17		0	2,06	1,49	1,15
1024	3	1,28	1,71	1,01	7168	3	1,32	1,50	1,04
	5	1,58	1,71	1,03		5	3,92	1,51	1,04
	17	1,39	1,71	1,01		17	2,52	1,50	1,04
	257	2,59	1,57	1,11		257	1,32	1,51	1,02
	65537	1,41	1,71	1,04		65537	1,90	1,51	1,12
	0	1,57	1,71	1,17		0	1,16	1,51	1,12
1536	3	1,34	1,86	1,04	8192	3	1,71	1,83	1,08
	5	1,71	1,81	1,03		5	1,46	1,83	1,11
	17	1,42	1,81	1,03		17	2,78	1,84	1,01
	257	1,12	1,81	1,04		257	1,26	1,83	1,11
	65537	1,09	1,73	1,16		65537	1,73	1,83	1,06
	0	1,68	1,58	1,16		0	1,23	1,83	1,15
2048	3	2,38	1,84	1,03	15360	3	2,00	1,51	1,03
	5	2,26	1,86	1,01		5	4,53	1,51	1,03
	17	1,15	1,82	1,00		17	1,91	1,51	1,02
	257	1,21	1,82	1,03		257	1,65	1,51	1,02
	65537	2,02	1,84	1,10		65537	2,59	1,52	1,06
	0	1,59	1,84	1,10		0	1,13	1,52	1,18
3072	3	1,62	1,81	1,03	16384	3	2,07	1,51	1,04
	5	1,28	1,80	1,03		5	1,29	1,52	1,03
	17	3,72	1,78	1,01		17	2,27	1,51	1,09
	257	1,01	1,80	1,05		257	1,55	1,51	1,03
	65537	1,40	1,83	1,08		65537	1,28	1,51	1,05
	0	2,99	1,80	1,13		0	2,15	1,51	1,06

Далі, у таблиці 22, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 22. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads).

With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	18,6667	1,00	0,23	4096	3	20144,30	425,00	13,51
	5	15,3333	1,00	0,22		5	24276,00	429,00	13,43
	17	19,0000	1,00	0,23		17	21196,30	429,00	13,59
	257	14,0000	1,00	0,24		257	21483,30	428,00	14,32
	65537	24,6667	1,00	0,24		65537	14137,30	429,00	15,33
	0	16,0000	1,00	0,30		0	17403,70	431,00	16,99
1024	3	181,667	9,00	1,19	7168	3	99199,70	2186,00	42,15
	5	205,333	7,00	0,93		5	114805,00	2182,00	42,73
	17	138,333	7,00	0,91		17	82167,30	2184,00	43,03
	257	118,333	7,00	0,98		257	109333,00	2189,00	43,88
	65537	159,333	8,00	1,04		65537	69813,70	2190,00	45,32
	0	152,333	7,00	1,16		0	54115,70	2194,00	52,23
1536	3	418,667	21,00	1,89	8192	3	510818,00	2694,00	52,63
	5	698,000	21,00	1,90		5	408350,00	2690,00	53,28
	17	474,333	21,00	1,89		17	256253,00	2706,00	56,26
	257	553,667	21,00	1,96		257	225097,00	2690,00	56,18
	65537	410,667	21,00	2,00		65537	345928,00	2689,00	61,45
	0	360,333	21,00	2,37		0	201637,00	2690,00	66,26
2048	3	961,667	47,00	3,38	15360	3	2497620,00	22273,00	169,41
	5	1296,000	47,00	3,51		5	2524290,00	22271,00	169,16
	17	1582,330	47,00	3,62		17	3304140,00	22262,00	164,04
	257	1857,330	47,00	3,80		257	2815630,00	22255,00	181,80
	65537	1030,330	47,00	3,89		65537	2208230,00	22267,00	188,72
	0	1172,670	48,00	4,34		0	1224830,00	22280,00	221,05
3072	3	3531,67	156,00	7,10	16384	3	2386200,00	27044,00	203,46
	5	5991,00	156,00	7,39		5	1882080,00	27159,00	206,14
	17	4926,67	160,00	8,15		17	2062200,00	27049,00	211,70
	257	10412,70	163,00	8,21		257	4002210,00	27021,00	220,48
	65537	4669,67	162,00	8,44		65537	2340680,00	27056,00	244,14
	0	3695,67	164,00	9,35		0	2137100,00	27001,00	289,03

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в два потоки (with improvements and 2 threads).

У таблиці 23 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,04-4,16 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,14 рази в залежності від розміру відкритої експоненти для ключів довжиною 2048 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,25 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.

Таблиця 23. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads)

With improvements / With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,52	1,00	0,96	4096	3	0,80	1,08	1,01
	5	1,72	1,00	1,05		5	0,96	1,07	1,11
	17	1,32	1,00	1,00		17	0,73	1,07	1,03
	257	2,10	1,00	0,96		257	1,11	1,07	1,08
	65537	1,27	1,00	1,00		65537	1,75	1,07	1,04
	0	1,96	1,00	0,97		0	0,94	1,06	1,07
1024	3	1,27	0,78	0,77	7168	3	1,99	1,09	1,02
	5	0,84	1,00	1,00		5	0,71	1,09	1,03
	17	1,34	1,00	1,09		17	1,79	1,09	1,04
	257	1,38	1,00	0,96		257	1,07	1,09	1,01
	65537	1,26	0,88	1,01		65537	2,15	1,09	1,03
	0	1,29	1,00	1,04		0	4,16	1,09	1,09
1536	3	1,66	1,00	1,08	8192	3	0,77	1,07	1,02
	5	0,83	1,00	1,08		5	0,85	1,07	1,01
	17	1,48	1,00	1,11		17	1,05	1,07	1,02
	257	1,57	1,00	1,12		257	2,23	1,07	1,01
	65537	2,22	1,05	1,05		65537	0,81	1,07	1,02
	0	1,83	1,14	1,08		0	1,69	1,07	1,05
2048	3	1,88	1,04	1,01	15360	3	1,06	1,02	1,18
	5	1,13	1,04	1,02		5	0,58	1,02	1,18
	17	1,55	1,04	1,04		17	1,04	1,02	1,25
	257	1,43	1,04	1,04		257	1,52	1,02	1,15
	65537	1,34	1,04	1,04		65537	0,51	1,02	1,11
	0	1,93	1,02	1,09		0	2,46	1,02	1,12
3072	3	2,35	1,06	1,15	16384	3	1,15	1,02	1,07
	5	1,30	1,06	1,05		5	2,32	1,01	1,08
	17	1,08	1,03	1,03		17	1,94	1,02	1,06
	257	0,84	1,02	1,06		257	1,19	1,02	1,05
	65537	1,79	1,02	1,04		65537	2,36	1,02	1,05
	0	2,20	1,01	1,08		0	1,30	1,02	1,04

У таблиці 24 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в 4 потоки 32-бітних машинних слів.

Таблиця 24. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в чотири потоки (with improvements and 4 threads).

With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	69,6667	2,00	0,23	4096	3	6486,67	206,00	13,32
	5	59,6667	2,00	0,25		5	13514,00	207,00	13,71
	17	72,3333	2,00	0,25		17	12568,00	202,00	13,15
	257	53,3333	1,00	0,26		257	17703,70	207,00	14,16
	65537	89,0000	2,00	0,33		65537	8788,67	203,00	15,05
	0	95,3333	2,00	0,50		0	15316,70	205,00	15,62
1024	3	298,6670	7,00	0,96	7168	3	65888,30	1232,00	42,57
	5	251,333	7,00	0,99		5	65857,00	1230,00	42,33
	17	210,667	7,00	1,11		17	134147,00	1229,00	42,93
	257	276,333	7,00	1,00		257	56962,00	1231,00	42,54
	65537	219,667	7,00	1,02		65537	95361,70	1225,00	46,17
	0	218,667	7,00	1,21		0	67076,30	1232,00	47,42
1536	3	714,000	16,00	1,92	8192	3	66581,30	1815,00	53,14
	5	477,333	16,00	1,92		5	82309,30	1816,00	53,94
	17	948,000	16,00	1,94		17	122464,00	1818,00	56,40
	257	645,333	16,00	2,16		257	153149,00	1817,00	56,33
	65537	706,000	16,00	2,07		65537	58354,30	1818,00	56,27
	0	577,333	16,00	2,47		0	71901,00	1821,00	66,77
2048	3	1205,33	34,00	3,30	15360	3	1366930,00	9446,00	195,18
	5	1804,67	35,00	3,35		5	961813,00	9387,00	191,70
	17	1225,00	35,00	3,67		17	1091990,00	9398,00	193,78
	257	1201,67	34,00	3,88		257	480506,00	9441,00	203,83
	65537	1175,67	34,00	3,75		65537	935691,00	9415,00	198,38
	0	1068,00	34,00	4,25		0	812729,00	9458,00	224,68
3072	3	3449,00	106,00	7,60	16384	3	551697,00	14029,00	216,95
	5	3792,00	108,00	7,64		5	1017160,00	14047,00	218,07
	17	4125,33	107,00	7,74		17	919074,00	14034,00	218,97
	257	7888,00	105,00	8,45		257	1561910,00	14071,00	224,60
	65537	2557,33	106,00	8,51		65537	1732150,00	14065,00	242,25
	0	3544,00	106,00	9,70		0	741618,00	14015,00	262,43

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням у чотири потоки (with improvements and 4 threads).

У таблиці 25 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням у 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,07-8,88 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,31-2,42 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,20 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.

Таблиця 25. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,41	0,50	0,96	4096	3	2,48	2,23	1,03
	5	0,44	0,50	0,92		5	1,73	2,22	1,08
	17	0,35	0,50	0,92		17	1,23	2,27	1,06
	257	0,55	1,00	0,88		257	1,35	2,22	1,09
	65537	0,35	0,50	0,73		65537	2,81	2,27	1,06
	0	0,33	0,50	0,58		0	1,07	2,23	1,17
1024	3	0,77	1,00	0,96	7168	3	2,99	1,94	1,01
	5	0,69	1,00	0,94		5	1,23	1,94	1,04
	17	0,88	1,00	0,89		17	1,10	1,94	1,05
	257	0,59	1,00	0,94		257	2,06	1,93	1,04
	65537	0,91	1,00	1,03		65537	1,58	1,95	1,01
	0	0,90	1,00	1,00		0	3,35	1,94	1,20
1536	3	0,97	1,31	1,06	8192	3	5,88	1,59	1,01
	5	1,21	1,31	1,07		5	4,22	1,59	1,00
	17	0,74	1,31	1,08		17	2,20	1,59	1,02
	257	1,35	1,31	1,02		257	3,28	1,59	1,01
	65537	1,29	1,38	1,01		65537	4,81	1,59	1,11
	0	1,14	1,50	1,04		0	4,75	1,59	1,05
2048	3	1,50	1,44	1,04	15360	3	1,94	2,41	1,03
	5	0,81	1,40	1,07		5	1,53	2,42	1,04
	17	2,00	1,40	1,02		17	3,13	2,42	1,06
	257	2,22	1,44	1,02		257	8,88	2,42	1,02
	65537	1,17	1,44	1,07		65537	1,20	2,41	1,05
	0	2,12	1,44	1,11		0	3,71	2,40	1,10
3072	3	2,40	1,56	1,07	16384	3	4,98	1,96	1,00
	5	2,06	1,53	1,01		5	4,29	1,96	1,02
	17	1,29	1,54	1,09		17	4,36	1,96	1,02
	257	1,12	1,58	1,03		257	3,05	1,96	1,03
	65537	3,27	1,56	1,04		65537	3,19	1,96	1,06
	0	2,29	1,56	1,04		0	3,73	1,97	1,15

У таблиці 26 наведені результати вимірювання швидкодії операцій криптосистеми RSA з запропонованими методами та розпаралелюванням у декілька потоків для 32-бітних машинних слів.

Таблиця 26. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads)

With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	260,333	4,00	0,33	4096	3	15817,70	116,00	16,94
	5	390,667	5,00	0,36		5	14763,00	117,00	16,09
	17	316,667	5,00	0,35		17	21621,00	117,00	16,69
	257	308,333	5,00	0,39		257	16086,00	116,00	16,76
	65537	265,333	2,00	0,37		65537	22782,00	115,00	16,93
	0	305,000	2,00	0,47		0	9432,33	115,00	17,48
1024	3	625,333	6,00	1,17	7168	3	46076,30	555,00	47,76
	5	594,667	6,00	1,20		5	62493,30	557,00	48,98
	17	876,667	8,00	1,19		17	129710,00	573,00	50,59
	257	646,667	13,00	1,24		257	84973,30	564,00	49,89
	65537	612,667	7,00	1,24		65537	78848,30	556,00	52,97
	0	605,667	6,00	1,42		0	96544,00	566,00	52,84
1536	3	1539,67	13,00	2,50	8192	3	48234,00	817,00	67,22
	5	1814,67	13,00	2,45		5	78265,30	812,00	63,19
	17	1570,67	13,00	2,55		17	53793,30	894,00	65,92
	257	1866,7	13,00	2,59		257	42381,00	825,00	68,55
	65537	1827,33	12,00	2,59		65537	58721,00	837,00	69,17
	0	1894,67	13,00	2,81		0	31194,70	848,00	69,44
2048	3	2346,67	23,00	4,31	15360	3	305052,00	5141,00	218,54
	5	2503,67	23,00	4,23		5	613720,00	5100,00	208,14
	17	2725,00	23,00	4,27		17	846329,00	5103,00	222,26
	257	2415,67	23,00	4,26		257	762540,00	5141,00	218,89
	65537	2873,67	23,00	4,60		65537	314130,00	5136,00	218,71
	0	2184,00	23,00	4,79		0	670676,00	5113,00	227,86
3072	3	6776,00	57,00	9,68	16384	3	1364900,00	6310,00	217,82
	5	3134,00	56,00	9,36		5	2868220,00	6065,00	247,14
	17	3333,33	56,00	9,87		17	1698130,00	6101,00	240,56
	257	5022,67	58,00	9,61		257	2227690,00	6360,00	246,90
	65537	3925,33	63,00	9,34		65537	1544440,00	6119,00	237,60
	0	3069,67	57,00	10,27		0	1147980,00	6171,00	250,27

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 27 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-11,86 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,17-4,54 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,20 рази лише для відкритої експоненти 0 для ключів довжиною 4096 біт та більше.

Таблиця 27. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у декілька потоки (with improvements and multi threads)

With improvements / With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,11	0,25	0,67	4096	3	1,02	3,96	0,81
	5	0,07	0,20	0,64		5	1,58	3,92	0,92
	17	0,08	0,20	0,66		17	0,72	3,92	0,84
	257	0,10	0,20	0,59		257	1,48	3,97	0,92
	65537	0,12	0,50	0,65		65537	1,08	4,00	0,95
	0	0,10	0,50	0,62		0	1,74	3,98	1,04
1024	3	0,37	1,17	0,79	7168	3	4,28	4,30	0,90
	5	0,29	1,17	0,78		5	1,30	4,29	0,90
	17	0,21	0,88	0,83		17	1,14	4,16	0,89
	257	0,25	0,54	0,76		257	1,38	4,22	0,89
	65537	0,33	1,00	0,85		65537	1,91	4,29	0,88
	0	0,32	1,17	0,85		0	2,33	4,21	1,07
1536	3	0,45	1,62	0,82	8192	3	8,12	3,53	0,80
	5	0,32	1,62	0,84		5	4,44	3,55	0,86
	17	0,45	1,62	0,82		17	5,00	3,22	0,87
	257	0,47	1,62	0,85		257	11,86	3,50	0,83
	65537	0,50	1,83	0,81		65537	4,78	3,45	0,90
	0	0,35	1,85	0,91		0	10,95	3,40	1,01
2048	3	0,77	2,13	0,80	15360	3	8,70	4,42	0,92
	5	0,58	2,13	0,85		5	2,40	4,46	0,96
	17	0,90	2,13	0,88		17	4,04	4,46	0,92
	257	1,10	2,13	0,93		257	5,60	4,44	0,95
	65537	0,48	2,13	0,88		65537	3,58	4,43	0,96
	0	1,04	2,13	0,98		0	4,50	4,44	1,09
3072	3	1,22	2,89	0,84	16384	3	2,01	4,36	1,00
	5	2,49	2,95	0,83		5	1,52	4,54	0,90
	17	1,60	2,95	0,85		17	2,36	4,51	0,93
	257	1,75	2,86	0,90		257	2,14	4,33	0,94
	65537	2,13	2,62	0,94		65537	3,58	4,51	1,08
	0	2,65	2,89	0,98		0	2,41	4,47	1,20

Далі розглядаються результати вимірювання для 64-бітних машинних слів.

У таблицях 28 і 29 наведені результати вимірювання швидкодії операцій криптосистеми RSA без запропонованих методів для 64-бітних машинних слів.

Таблиця 28. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів без застосування запропонованих методів (original)

Original (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	7,33333	0,1	0,07	4096	3	4141,00	64,00	3,63
	5	7,3333	0,1	0,06		5	2564,67	62,00	3,67
	17	6,3333	0,1	0,06		17	4960,33	65,00	3,69
	257	7,3333	0,1	0,06		257	2127,33	69,00	3,69
	65537	7,3333	0,1	0,06		65537	3319,33	64,00	3,81
	0	5,3333	0,1	0,09		0	3061,67	65,00	4,89
1024	3	44,3333	1,00	0,23	7168	3	51550,00	329,00	10,37
	5	33,0000	1,00	0,24		5	32163,30	320,00	10,41
	17	33,3333	1,00	0,24		17	19337,00	324,00	10,41
	257	43,333	1,00	0,24		257	13370,00	323,00	10,78
	65537	45,0000	1,00	0,26		65537	19167,00	325,00	11,00
	0	35,3333	1,00	0,31		0	35571,70	320,00	14,03
1536	3	168,333	3,00	0,53	8192	3	51605,30	490,00	13,56
	5	124,000	3,00	0,53		5	53802,30	491,00	14,35
	17	109,333	3,00	0,54		17	46588,00	496,00	14,72
	257	157,000	3,00	0,54		257	38750,00	496,00	14,89
	65537	123,6670	3,00	0,55		65537	43692,30	497,00	15,00
	0	178,0000	3,00	0,70		0	88196,30	498,00	18,30
2048	3	472,333	7,00	0,89	15360	3	338977,00	2971,00	45,48
	5	247,333	7,00	0,89		5	485386,00	2972,00	46,21
	17	405,000	9,00	0,92		17	360670,00	2966,00	46,88
	257	250,333	9,00	0,90		257	804069,00	2961,00	47,62
	65537	289,667	9,00	0,94		65537	600379,00	2961,00	49,61
	0	354,000	7,00	1,22		0	817751,00	3078,00	63,14
3072	3	1504,00	27,00	2,12	16384	3	369497,00	3642,00	53,05
	5	912,000	27,00	2,19		5	464214,00	3691,00	52,83
	17	1170,33	27,00	2,30		17	540933,00	3689,00	53,27
	257	881,33	29,00	2,34		257	687678,00	3687,00	54,00
	65537	1397,67	29,00	2,40		65537	805719,00	3687,00	56,23
	0	1169,67	31,00	2,91		0	665368,00	3683,00	73,81

Таблиця 29. Результати вимірювання швидкодії арифметичних операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих методів (with improvements)

With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	11	0,1	0,06	4096	3	5111,33	60	3,41
	5	6,33333	0,1	0,06		5	3386,67	60	3,39
	17	6,00000	0,1	0,06		17	3366,67	60	3,48
	257	7,66667	0,1	0,06		257	4601,33	60	3,49
	65537	7,33333	0,1	0,06		65537	2220,67	61	3,6
	0	6	0,1	0,08		0	3204	60	4,58
1024	3	42	1	0,23	7168	3	25327	313	10,17
	5	37,3333	1	0,24		5	36579	317	10,32
	17	44,3333	1	0,23		17	36683	318	10,34
	257	36,6667	1	0,24		257	23171	315	10,61
	65537	50	1	0,25		65537	26015,3	313	10,79
	0	49,3333	1	0,33		0	30102,3	318	13,7
1536	3	206	3	0,53	8192	3	65655,7	471	13,26
	5	107,333	3	0,54		5	25752,3	474	13,36
	17	144,000	4	0,54		17	61251,7	487	14,17
	257	125,667	3	0,55		257	26867	470	13,74
	65537	102,000	3	0,57		65537	43291,7	469	14,39
	0	248	4	0,7		0	81946	465	17,77
2048	3	214,333	7	0,89	15360	3	151595	2941	44,74
	5	287,667	7	0,9		5	449696	2943	45,32
	17	244,667	8	0,88		17	166652	2944	45,69
	257	385,667	9	0,94		257	730998	2940	46,47
	65537	268,333	9	0,93		65537	385288	2944	47,77
	0	333	9	1,19		0	768236	2942	60,86
3072	3	1546,67	26	1,99	16384	3	307095	3576	52,31
	5	816,000	26	1,96		5	307460	3570	52,5
	17	1079,67	26	1,99		17	380434	3573	52,96
	257	765,667	28	2,03		257	682933	3574	53,68
	65537	769,333	26	2,11		65537	333466	3572	55,15
	0	1606,67	27	2,63		0	600438	3577	72,41

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 30 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-2,42 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,15 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,17 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 30. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop за результатами без та з використанням запропонованих методів для 64-бітних машинних слів.

Original / With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,67	1,00	1,17	4096	3	0,81	1,07	1,06
	5	1,16	1,00	1,00		5	0,76	1,03	1,08
	17	1,06	1,00	1,00		17	1,47	1,08	1,06
	257	0,96	1,00	1,00		257	0,46	1,15	1,06
	65537	1,00	1,00	1,00		65537	1,49	1,05	1,06
	0	0,89	1,00	1,13		0	0,96	1,08	1,07
1024	3	1,06	1,00	1,00	7168	3	2,04	1,05	1,02
	5	0,88	1,00	1,00		5	0,88	1,01	1,01
	17	0,75	1,00	1,04		17	0,53	1,02	1,01
	257	1,18	1,00	1,00		257	0,58	1,03	1,02
	65537	0,90	1,00	1,04		65537	0,74	1,04	1,02
	0	0,72	1,00	0,94		0	1,18	1,01	1,02
1536	3	0,82	1,00	1,00	8192	3	0,79	1,04	1,02
	5	1,16	1,00	0,98		5	2,09	1,04	1,07
	17	0,76	0,75	1,00		17	0,76	1,02	1,04
	257	1,25	1,00	0,98		257	1,44	1,06	1,08
	65537	1,21	1,00	0,96		65537	1,01	1,06	1,04
	0	0,72	0,75	1,00		0	1,08	1,07	1,03
2048	3	2,20	1,00	1,00	15360	3	2,24	1,01	1,02
	5	0,86	1,00	0,99		5	1,08	1,01	1,02
	17	1,66	1,13	1,05		17	2,16	1,01	1,03
	257	0,65	1,00	0,96		257	1,10	1,01	1,02
	65537	1,08	1,00	1,01		65537	1,56	1,01	1,04
	0	1,06	0,78	1,03		0	1,06	1,05	1,04
3072	3	0,97	1,04	1,07	16384	3	1,20	1,02	1,01
	5	1,12	1,04	1,12		5	1,51	1,03	1,01
	17	1,08	1,04	1,16		17	1,42	1,03	1,01
	257	1,15	1,04	1,15		257	1,01	1,03	1,01
	65537	1,82	1,12	1,14		65537	2,42	1,03	1,02
	0	0,73	1,15	1,11		0	1,11	1,03	1,02

Далі, у таблиці 31, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 31. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads).

With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	13,33330	0,1	0,06	4096	3	3133,33	57,00	3,01
	5	3,33333	0,1	0,07		5	2165,33	56,00	3,04
	17	4,00000	0,1	0,06		17	1930,67	57,00	3,07
	257	3,00000	0,1	0,06		257	2149,33	57,00	3,20
	65537	3,66667	0,1	0,07		65537	1632,33	57,00	3,23
	0	2,66667	0,1	0,08		0	2166,67	58,00	4,26
1024	3	27,6667	1,00	0,25	7168	3	10087,00	303,00	9,58
	5	21,000	1,00	0,24		5	13740,30	303,00	9,80
	17	20,3333	1,00	0,23		17	11923,70	303,00	9,83
	257	26,667	1,00	0,24		257	27647,30	303,00	10,45
	65537	19,0000	1,00	0,24		65537	36485,00	302,00	10,72
	0	24,333	1,00	0,30		0	13752,70	315,00	12,99
1536	3	55,0000	3,00	0,54	8192	3	74938,70	436,00	12,24
	5	75,000	3,00	0,53		5	39024,00	447,00	12,32
	17	76,333	3,00	0,54		17	23095,70	446,00	13,40
	257	86,333	3,00	0,54		257	15775,00	446,00	13,72
	65537	78,667	3,00	0,56		65537	34421,00	447,00	14,06
	0	104,667	3,00	0,69		0	49162,30	449,00	16,86
2048	3	309,667	7,00	0,88	15360	3	314013,00	2844,00	43,36
	5	177,333	7,00	0,87		5	347484,00	2840,00	43,29
	17	147,667	8,00	0,88		17	175240,00	2841,00	43,75
	257	281,000	9,00	0,94		257	378266,00	2840,00	44,35
	65537	275,667	9,00	0,98		65537	483722,00	2844,00	45,74
	0	210,667	8,00	1,19		0	232647,00	2841,00	59,99
3072	3	777,667	24,00	1,77	16384	3	525532,00	3476,00	50,87
	5	886,00	24,00	1,81		5	530193,00	3482,00	50,58
	17	1000,000	24,00	1,81		17	255386,00	3478,00	52,43
	257	985,00	25,00	2,00		257	577941,00	3472,00	52,48
	65537	770,00	25,00	2,04		65537	347381,00	3474,00	52,34
	0	706,33	26,00	2,38		0	416735,00	3574,00	66,48

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в два потоки (with improvements and 2 threads).

У таблиці 32 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,08-3,75 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,33 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,13 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 32. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	0,83	1,00	1,00	4096	3	1,63	1,05	1,13
	5	1,90	1,00	0,86		5	1,56	1,07	1,12
	17	1,50	1,00	1,00		17	1,74	1,05	1,13
	257	2,56	1,00	1,00		257	2,14	1,05	1,09
	65537	2,00	1,00	0,86		65537	1,36	1,07	1,11
	0	2,25	1,00	1,00		0	1,48	1,03	1,08
1024	3	1,52	1,00	0,92	7168	3	2,51	1,03	1,06
	5	1,78	1,00	1,00		5	2,66	1,05	1,05
	17	2,18	1,00	1,00		17	3,08	1,05	1,05
	257	1,37	1,00	1,00		257	0,84	1,04	1,02
	65537	2,63	1,00	1,04		65537	0,71	1,04	1,01
	0	2,03	1,00	1,10		0	2,19	1,01	1,05
1536	3	3,75	1,00	0,98	8192	3	0,88	1,08	1,08
	5	1,43	1,00	1,02		5	0,66	1,06	1,08
	17	1,89	1,33	1,00		17	2,65	1,09	1,06
	257	1,46	1,00	1,02		257	1,70	1,05	1,00
	65537	1,30	1,00	1,02		65537	1,26	1,05	1,02
	0	2,37	1,33	1,01		0	1,67	1,04	1,05
2048	3	0,69	1,00	1,01	15360	3	0,48	1,03	1,03
	5	1,62	1,00	1,03		5	1,29	1,04	1,05
	17	1,66	1,00	1,00		17	0,95	1,04	1,04
	257	1,37	1,00	1,00		257	1,93	1,04	1,05
	65537	0,97	1,00	0,95		65537	0,80	1,04	1,04
	0	1,58	1,13	1,00		0	3,30	1,04	1,01
3072	3	1,99	1,08	1,12	16384	3	0,58	1,03	1,03
	5	0,92	1,08	1,08		5	0,58	1,03	1,04
	17	1,08	1,08	1,10		17	1,49	1,03	1,01
	257	0,78	1,12	1,02		257	1,18	1,03	1,02
	65537	1,00	1,04	1,03		65537	0,96	1,03	1,05
	0	2,27	1,04	1,11		0	1,44	1,00	1,09

У таблиці 33 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням у 4 потоки для 64-бітних машинних слів.

Таблиця 33. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в чотири потоки (with improvements and 4 threads)

With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	52,3333	1,00	0,08	4096	3	2609,00	42,00	3,11
	5	49,00	1,00	0,08		5	2297,67	41,00	3,16
	17	56,0000	1,00	0,09		17	3709,33	41,00	3,34
	257	41,6667	1,00	0,09		257	2391,00	40,00	3,43
	65537	42,3333	1,00	0,10		65537	1428,33	41,00	3,35
	0	42,6667	1,00	0,23		0	1491,67	41,00	4,13
1024	3	153,0000	2,00	0,24	7168	3	23397,70	176,00	9,23
	5	120,333	2,00	0,26		5	8370,00	178,00	9,43
	17	132,667	2,00	0,26		17	10948,70	183,00	9,65
	257	142,000	2,00	0,29		257	11222,00	179,00	10,33
	65537	106,333	2,00	0,32		65537	23849,30	177,00	10,59
	0	138,333	2,00	0,56		0	14915,00	178,00	12,48
1536	3	206,667	5,00	0,55	8192	3	22779,00	258,00	12,53
	5	324,333	5,00	0,57		5	7243,00	254,00	12,42
	17	267,333	5,00	0,55		17	24581,30	257,00	13,37
	257	310,333	5,00	0,56		257	20043,70	255,00	13,39
	65537	359,000	5,00	0,58		65537	26847,70	256,00	13,68
	0	206,333	6,00	0,80		0	19373,00	255,00	15,95
2048	3	473,667	9,00	0,97	15360	3	206344,00	1544,00	44,09
	5	345,000	8,00	0,92		5	207222,00	1551,00	44,84
	17	370,000	10,00	0,92		17	163383,00	1541,00	45,66
	257	484,00	8,00	0,95		257	205867,00	1546,00	45,51
	65537	337,333	8,00	0,95		65537	93605,00	1543,00	46,60
	0	395,667	9,00	1,31		0	187772,00	1534,00	54,63
3072	3	1790,00	21,00	2,01	16384	3	267972,00	1858,00	51,15
	5	1407,00	21,00	2,02		5	199110,00	1851,00	51,97
	17	1235,67	22,00	2,02		17	97196,00	1860,00	52,27
	257	1159,00	20,00	2,06		257	145883,00	1855,00	52,85
	65537	1416,67	21,00	2,08		65537	612143,00	2339,00	54,09
	0	1183,67	21,00	2,50		0	170107,00	1856,00	61,86

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в 4 потоки (with improvements and 4 threads).

У таблиці 34 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,02-4,68 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.
- Операція створення підпису ефективніша в 1,18-1,93 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,17 рази для деяких відкритих експонент для ключів довжиною 3072 біт та більше.

Таблиця 34. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,21	0,10	0,75	4096	3	1,96	1,43	1,10
	5	0,13	0,10	0,75		5	1,47	1,46	1,07
	17	0,11	0,10	0,67		17	0,91	1,46	1,04
	257	0,18	0,10	0,67		257	1,92	1,50	1,02
	65537	0,17	0,10	0,60		65537	1,55	1,49	1,07
	0	0,14	0,10	0,35		0	2,15	1,46	1,11
1024	3	0,27	0,50	0,96	7168	3	1,08	1,78	1,10
	5	0,31	0,50	0,92		5	4,37	1,78	1,09
	17	0,33	0,50	0,88		17	3,35	1,74	1,07
	257	0,26	0,50	0,83		257	2,06	1,76	1,03
	65537	0,47	0,50	0,78		65537	1,09	1,77	1,02
	0	0,36	0,50	0,59		0	2,02	1,79	1,10
1536	3	1,00	0,60	0,96	8192	3	2,88	1,83	1,06
	5	0,33	0,60	0,95		5	3,56	1,87	1,08
	17	0,54	0,80	0,98		17	2,49	1,89	1,06
	257	0,40	0,60	0,98		257	1,34	1,84	1,03
	65537	0,28	0,60	0,98		65537	1,61	1,83	1,05
	0	1,20	0,67	0,88		0	4,23	1,82	1,11
2048	3	0,45	0,78	0,92	15360	3	0,73	1,90	1,01
	5	0,83	0,88	0,98		5	2,17	1,90	1,01
	17	0,66	0,80	0,96		17	1,02	1,91	1,00
	257	0,80	1,13	0,99		257	3,55	1,90	1,02
	65537	0,80	1,13	0,98		65537	4,12	1,91	1,03
	0	0,84	1,00	0,91		0	4,09	1,92	1,11
3072	3	0,86	1,24	0,99	16384	3	1,15	1,92	1,02
	5	0,58	1,24	0,97		5	1,54	1,93	1,01
	17	0,87	1,18	0,99		17	3,91	1,92	1,01
	257	0,66	1,40	0,99		257	4,68	1,93	1,02
	65537	0,54	1,24	1,01		65537	0,54	1,53	1,02
	0	1,36	1,29	1,05		0	3,53	1,93	1,17

У таблиці 35 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів.

Таблиця 35. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Desktop з використанням 64-бітних машинних слів з застосуванням запропонованих швидкодії та розпаралелюванням в декілька потоків (with improvements and multi threads).

With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	235,6670	1,00	0,10	4096	3	6482,00	31,00	5,33
	5	235,00	1,00	0,11		5	7026,00	31,00	5,11
	17	236,0000	1,00	0,11		17	9848,67	31,00	5,12
	257	213,3330	1,00	0,13		257	7136,33	31,00	5,21
	65537	235,6670	1,00	0,15		65537	9571,33	31,00	5,25
	0	262,6670	1,00	0,35		0	8695,33	31,00	5,63
1024	3	459,33	3,00	0,38	7168	3	21600,00	115,00	14,94
	5	494,667	3,00	0,40		5	17722,30	115,00	15,22
	17	714,000	3,00	0,41		17	47757,30	115,00	15,17
	257	550,00	3,00	0,40		257	29806,00	128,00	15,04
	65537	540,333	3,00	0,41		65537	28577,30	128,00	15,14
	0	509,333	3,00	0,77		0	27340,00	114,00	16,29
1536	3	1359,000	6,00	0,83	8192	3	17159,30	156,00	19,86
	5	1255,330	6,00	0,83		5	28235,30	157,00	19,88
	17	1293,670	6,00	0,84		17	16714,00	171,00	19,55
	257	1336,33	6,00	0,86		257	23074,00	156,00	19,74
	65537	1208,33	6,00	0,88		65537	17449,30	157,00	20,01
	0	1143,33	6,00	1,22		0	13904,70	157,00	21,26
2048	3	1489,33	9,00	1,42	15360	3	227411,00	858,00	65,81
	5	1476,00	14,00	1,38		5	233082,00	859,00	66,23
	17	2288,33	9,00	1,48		17	123538,00	848,00	65,61
	257	1925,67	9,00	1,38		257	117760,00	857,00	66,16
	65537	2187,00	9,00	1,43		65537	146572,00	858,00	67,11
	0	1364,00	9,00	1,74		0	131968,00	860,00	71,67
3072	3	4069,67	19,00	2,97	16384	3	290518,00	1031,00	77,14
	5	2080,33	19,00	3,05		5	283182,00	1032,00	77,79
	17	3229,67	18,00	3,04		17	385702,00	1028,00	78,29
	257	2478,00	18,00	3,03		257	429529,00	1114,00	78,14
	65537	2457,00	18,00	3,06		65537	411471,00	1030,00	77,96
	0	2121,00	18,00	3,52		0	361657,00	1030,00	83,92

Для оцінки ефективності запропонованих методів для платформи Desktop з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 36 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,09-6,74 рази в залежності від розміру відкритої експоненти для ключів довжиною 8192 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,37-3,48 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 36. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop за результатами з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів

With improvements / With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,05	0,10	0,60	4096	3	0,42	1,58	0,62
	5	0,03	0,10	0,55		5	0,39	1,21	0,57
	17	0,03	0,10	0,55		17	0,33	1,19	0,60
	257	0,04	0,10	0,46		257	0,13	1,24	0,66
	65537	0,03	0,10	0,40		65537	0,23	1,68	0,68
	0	0,02	0,10	0,23		0	0,26	1,93	0,78
1024	3	0,09	0,33	0,61	7168	3	0,78	2,20	0,63
	5	0,08	0,33	0,60		5	0,78	3,68	0,64
	17	0,06	0,33	0,56		17	0,72	2,26	0,66
	257	0,07	0,33	0,60		257	0,68	3,82	0,67
	65537	0,09	0,33	0,61		65537	0,74	3,75	0,62
	0	0,10	0,33	0,43		0	1,18	2,35	0,83
1536	3	0,15	0,50	0,64	8192	3	2,13	3,82	0,66
	5	0,09	0,50	0,65		5	1,09	2,34	0,66
	17	0,11	0,67	0,64		17	5,04	3,88	0,64
	257	0,09	0,50	0,64		257	1,11	3,77	0,65
	65537	0,08	0,50	0,65		65537	2,61	3,68	0,69
	0	0,22	0,67	0,57		0	1,74	3,66	0,80
2048	3	0,14	0,78	0,63	15360	3	3,79	3,45	0,65
	5	0,19	0,50	0,65		5	3,23	4,53	0,64
	17	0,11	0,89	0,59		17	3,61	5,20	0,68
	257	0,20	1,00	0,68		257	6,74	4,31	0,68
	65537	0,12	1,00	0,65		65537	5,71	3,12	0,63
	0	0,24	1,00	0,68		0	3,38	3,91	0,85
3072	3	0,38	1,37	0,67	16384	3	1,41	5,25	0,61
	5	0,39	1,37	0,64		5	3,44	4,27	0,62
	17	0,33	1,44	0,65		17	1,45	5,25	0,67
	257	0,31	1,56	0,67		257	2,78	3,62	0,66
	65537	0,31	1,44	0,69		65537	2,01	3,19	0,70
	0	0,76	1,50	0,75		0	1,99	3,67	0,84

Далі будуть розглядатися результати отримані для платформи Mobile (Intel Core-i7 6700HQ)

У таблицях 37 та 38 наведені результати вимірювання швидкодії операції криптосистеми для різної довжини ключа без та з запропонованими методами для платформи Mobile для 32-бітних машинних слів.

Таблиця 37. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів без застосування запропонованих методів (original)

Original (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	36,3333	1,00	0,25	4096	3	34078,00	728,00	15,23
	5	40,6667	1,00	0,32		5	58055,30	760,00	16,58
	17	40,0000	1,00	0,25		17	40676,30	732,00	16,73
	257	39,3333	1,00	0,26		257	34441,00	747,00	16,87
	65537	45,3333	1,00	0,29		65537	55805,30	753,00	17,21
	0	45,3333	1,00	0,36		0	34677,00	735,00	22,65
1024	3	436,000	13,00	1,06	7168	3	202949,00	3888,00	47,89
	5	337,000	14,00	1,00		5	472530,00	3619,00	49,47
	17	310,667	13,00	1,03		17	270540,00	3599,00	47,18
	257	344,667	13,00	1,08		257	292930,00	3874,00	49,06
	65537	435,667	13,00	1,17		65537	315170,00	3863,00	57,03
	0	403,667	13,00	1,46		0	250692,00	3878,00	65,91
1536	3	1156,670	39,00	2,25	8192	3	394126,00	5509,00	59,99
	5	1008,000	43,00	2,27		5	438581,00	5813,00	59,94
	17	1118,330	42,00	2,36		17	758765,00	5559,00	62,75
	257	1125,000	42,00	2,49		257	761814,00	5799,00	67,94
	65537	1026,000	42,00	2,58		65537	650499,00	5570,00	70,88
	0	1105,670	42,00	3,21		0	434924,00	5794,00	86,89
2048	3	3788,67	106,00	4,04	15360	3	6517900,00	37214,00	202,25
	5	2803,67	98,00	4,04		5	2756150,00	36758,00	215,92
	17	2764,00	98,00	4,33		17	6701760,00	37534,00	217,93
	257	2674,33	93,00	3,80		257	6638030,00	35922,00	216,46
	65537	3059,67	90,00	4,36		65537	10883400,00	37574,00	251,97
	0	4373,00	98,00	5,65		0	6397760,00	35637,00	296,27
3072	3	11226,00	328,00	9,31	16384	3	6533670,00	44647,00	253,26
	5	10465,00	329,00	9,32		5	6907020,00	43273,00	259,47
	17	13427,00	323,00	8,69		17	14130500,00	44387,00	250,99
	257	12995,30	322,00	9,95		257	11188100,00	41791,00	275,70
	65537	14554,00	335,00	10,51		65537	7134120,00	42367,00	261,67
	0	14839,70	326,00	12,79		0	14339500,00	41914,00	361,60

Таблиця 38. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів (with improvements)

With improvements (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	31,0000	1,00	0,22	4096	3	16874,30	389,00	13,79
	5	26,3333	1,00	0,23		5	26602,30	386,00	14,69
	17	24,6667	1,00	0,23		17	18618,70	386,00	14,05
	257	26,0000	1,00	0,23		257	20786,30	385,00	15,04
	65537	31,6667	1,00	0,26		65537	19814,30	387,00	15,56
	0	25,6667	1,00	0,29		0	21129,30	388,00	17,91
1024	3	238,3330	7,00	0,88	7168	3	109280,00	2226,00	43,02
	5	270,6670	6,00	0,91		5	192859,00	2226,00	43,64
	17	253,6670	6,00	0,92		17	211067,00	2222,00	42,78
	257	205,0000	6,00	1,02		257	218187,00	2223,00	46,66
	65537	158,3330	6,00	0,97		65537	88599,70	2224,00	45,58
	0	219,6670	6,00	1,14		0	239369,00	2222,00	54,58
1536	3	587,6670	20,00	1,90	8192	3	369346,00	2882,00	55,24
	5	545,6670	20,00	2,06		5	170492,00	2905,00	56,31
	17	622,0000	20,00	2,11		17	320005,00	2880,00	59,04
	257	707,0000	20,00	2,01		257	256500,00	2879,00	54,51
	65537	546,3330	20,00	2,12		65537	365245,00	2882,00	60,80
	0	725,3330	20,00	2,41		0	186506,00	2878,00	69,46
2048	3	2030,33	49,00	3,67	15360	3	3248040,00	19285,00	194,38
	5	1521,67	48,00	3,55		5	2270570,00	19311,00	192,18
	17	2445,33	49,00	3,72		17	2207600,00	19286,00	207,82
	257	2235,67	48,00	3,67		257	4144910,00	19268,00	205,34
	65537	1605,67	53,00	3,70		65537	1831160,00	19272,00	204,34
	0	2228,00	48,00	4,45		0	3311130,00	19271,00	243,54
3072	3	4984,67	161,00	8,43	16384	3	6149550,00	23160,00	214,68
	5	7407,67	162,00	8,20		5	3869460,00	22811,00	230,94
	17	7311,33	161,00	7,82		17	2929780,00	22872,00	216,77
	257	5094,33	173,00	8,11		257	4572740,00	22842,00	225,04
	65537	7158,67	161,00	8,63		65537	1883670,00	22813,00	246,96
	0	5443,33	162,00	10,07		0	3435620,00	22874,00	283,04

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 39 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,05-5,94 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція створення підпису ефективніша в 1,62-2,33 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.
- Операція перевірки підпису ефективніша в 1,04-1,39 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються.

Таблиця 39. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile за результатами без та з використанням запропонованих методів для 32-бітних машинних слів

Original / With improvements (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,17	1,00	1,14	4096	3	2,02	1,87	1,10
	5	1,54	1,00	1,39		5	2,18	1,97	1,13
	17	1,62	1,00	1,09		17	2,18	1,90	1,19
	257	1,51	1,00	1,13		257	1,66	1,94	1,12
	65537	1,43	1,00	1,12		65537	2,82	1,95	1,11
	0	1,77	1,00	1,24		0	1,64	1,89	1,26
1024	3	1,83	1,86	1,20	7168	3	1,86	1,75	1,11
	5	1,25	2,33	1,10		5	2,45	1,63	1,13
	17	1,22	2,17	1,12		17	1,28	1,62	1,10
	257	1,68	2,17	1,06		257	1,34	1,74	1,05
	65537	2,75	2,17	1,21		65537	3,56	1,74	1,25
	0	1,84	2,17	1,28		0	1,05	1,75	1,21
1536	3	1,97	1,95	1,18	8192	3	1,07	1,91	1,09
	5	1,85	2,15	1,10		5	2,57	2,00	1,06
	17	1,80	2,10	1,12		17	2,37	1,93	1,06
	257	1,59	2,10	1,24		257	2,97	2,01	1,25
	65537	1,88	2,10	1,22		65537	1,78	1,93	1,17
	0	1,52	2,10	1,33		0	2,33	2,01	1,25
2048	3	1,87	2,16	1,10	15360	3	2,01	1,93	1,04
	5	1,84	2,04	1,14		5	1,21	1,90	1,12
	17	1,13	2,00	1,16		17	3,04	1,95	1,05
	257	1,20	1,94	1,04		257	1,60	1,86	1,05
	65537	1,91	1,70	1,18		65537	5,94	1,95	1,23
	0	1,96	2,04	1,27		0	1,93	1,85	1,22
3072	3	2,25	2,04	1,10	16384	3	1,06	1,93	1,18
	5	1,41	2,03	1,14		5	1,79	1,90	1,12
	17	1,84	2,01	1,11		17	4,82	1,94	1,16
	257	2,55	1,86	1,23		257	2,45	1,83	1,23
	65537	2,03	2,08	1,22		65537	3,79	1,86	1,06
	0	2,73	2,01	1,27		0	4,17	1,83	1,28

Далі, у таблиці 40, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 40. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads)

With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	18,3333	1,00	0,23	4096	3	9880,67	361,00	13,61
	5	14,0000	1,00	0,25		5	14429,70	370,00	14,13
	17	19,3333	1,00	0,26		17	10387,70	361,00	14,04
	257	15,6667	1,00	0,26		257	24429,30	363,00	14,89
	65537	17,0000	1,00	0,28		65537	19834,70	380,00	15,47
	0	17,0000	1,00	0,34		0	9393,33	366,00	16,89
1024	3	182,667	8,00	1,02	7168	3	50878,70	2189,00	42,60
	5	205,000	8,00	0,99		5	148323,00	2205,00	42,61
	17	108,667	7,00	0,97		17	144568,00	2214,00	42,28
	257	145,000	7,00	1,00		257	118394,00	2181,00	44,85
	65537	117,333	7,00	1,00		65537	144518,00	2184,00	45,28
	0	172,667	7,00	1,25		0	68256,70	2212,00	51,64
1536	3	410,000	24,00	2,17	8192	3	308709,00	2833,00	53,97
	5	656,333	22,00	2,24		5	156462,00	2811,00	54,98
	17	302,000	22,00	2,19		17	171162,00	2851,00	55,74
	257	454,667	22,00	2,20		257	146330,00	2834,00	54,42
	65537	333,000	22,00	2,34		65537	126615,00	2843,00	59,70
	0	566,333	22,00	2,70		0	101501,00	2871,00	68,34
2048	3	1238,670	56,00	3,74	15360	3	3224820,00	17035,00	178,03
	5	1508,670	56,00	3,84		5	3246750,00	17871,00	190,71
	17	912,000	60,00	4,04		17	2596670,00	17945,00	202,13
	257	1203,000	56,00	4,01		257	3442240,00	17901,00	199,85
	65537	1217,000	52,00	4,28		65537	1270540,00	17052,00	204,17
	0	1055,000	52,00	4,83		0	3585860,00	17832,00	228,89
3072	3	4277,00	174,00	7,88	16384	3	2906630,00	20946,00	210,75
	5	6069,00	180,00	9,26		5	2033300,00	21742,00	212,23
	17	3924,00	170,00	8,37		17	3683690,00	21917,00	213,31
	257	5784,33	176,00	9,26		257	3188680,00	21911,00	219,98
	65537	8881,00	177,00	9,12		65537	2051310,00	21703,00	238,56
	0	3880,00	171,00	9,91		0	2323110,00	21818,00	271,67

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в два потоки (with improvements and 2 threads).

У таблиці 41 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-3,51 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,13 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,09 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.

Таблиця 41. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads)

With improvements / With improvements and 2 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	1,69	1,00	0,96	4096	3	1,71	1,08	1,01
	5	1,88	1,00	0,92		5	1,84	1,04	1,04
	17	1,28	1,00	0,88		17	1,79	1,07	1,00
	257	1,66	1,00	0,88		257	0,85	1,06	1,01
	65537	1,86	1,00	0,93		65537	1,00	1,02	1,01
	0	1,51	1,00	0,85		0	2,25	1,06	1,06
1024	3	1,30	0,88	0,86	7168	3	2,15	1,02	1,01
	5	1,32	0,75	0,92		5	1,30	1,01	1,02
	17	2,33	0,86	0,95		17	1,46	1,00	1,01
	257	1,41	0,86	1,02		257	1,84	1,02	1,04
	65537	1,35	0,86	0,97		65537	0,61	1,02	1,01
	0	1,27	0,86	0,91		0	3,51	1,00	1,06
1536	3	1,43	0,83	0,88	8192	3	1,20	1,02	1,02
	5	0,83	0,91	0,92		5	1,09	1,03	1,02
	17	2,06	0,91	0,96		17	1,87	1,01	1,06
	257	1,55	0,91	0,91		257	1,75	1,02	1,00
	65537	1,64	0,91	0,91		65537	2,88	1,01	1,02
	0	1,28	0,91	0,89		0	1,84	1,00	1,02
2048	3	1,64	0,88	0,98	15360	3	1,01	1,13	1,09
	5	1,01	0,86	0,92		5	0,70	1,08	1,01
	17	2,68	0,82	0,92		17	0,85	1,07	1,03
	257	1,86	0,86	0,92		257	1,20	1,08	1,03
	65537	1,32	1,02	0,86		65537	1,44	1,13	1,00
	0	2,11	0,92	0,92		0	0,92	1,08	1,06
3072	3	1,17	0,93	1,07	16384	3	2,12	1,11	1,02
	5	1,22	0,90	0,89		5	1,90	1,05	1,09
	17	1,86	0,95	0,93		17	0,80	1,04	1,02
	257	0,88	0,98	0,88		257	1,43	1,04	1,02
	65537	0,81	0,91	0,95		65537	0,92	1,05	1,04
	0	1,40	0,95	1,02		0	1,48	1,05	1,04

У таблиці 42 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в 4 потоки 32-бітних машинних слів.

Таблиця 42. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в чотири потоки (with improvements and 4 threads)

With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	124,0000	3,00	0,26	4096	3	16819,70	224,00	13,71
	5	107,6670	3,00	0,27		5	12381,00	243,00	13,78
	17	132,3330	3,00	0,29		17	8603,00	225,00	13,95
	257	139,6670	3,00	0,28		257	8749,00	242,00	14,92
	65537	117,6670	3,00	0,34		65537	10663,00	223,00	14,94
	0	98,0000	2,00	0,47		0	11972,30	228,00	16,74
1024	3	462,0000	10,00	1,07	7168	3	77297,30	1231,00	42,75
	5	445,667	11,00	1,05		5	73981,70	1225,00	42,87
	17	289,000	10,00	1,00		17	60464,30	1228,00	42,41
	257	367,333	9,00	1,11		257	90162,30	1335,00	45,20
	65537	358,333	9,00	1,05		65537	129646,00	1310,00	45,49
	0	392,667	10,00	1,46		0	129816,00	1327,00	51,41
1536	3	1081,330	20,00	2,31	8192	3	123530,00	1846,00	54,37
	5	884,667	20,00	2,24		5	136469,00	1944,00	55,39
	17	1129,000	20,00	2,20		17	77646,70	1852,00	55,17
	257	830,000	19,00	2,29		257	141211,00	1960,00	54,20
	65537	746,333	18,00	2,17		65537	136419,00	1847,00	59,97
	0	790,667	20,00	2,83		0	61009,00	1933,00	65,57
2048	3	1412,33	54,00	4,25	15360	3	642922,00	10359,00	190,14
	5	1930,33	37,00	3,87		5	1591560,00	10383,00	191,87
	17	1775,67	39,00	4,15		17	1174310,00	10355,00	200,56
	257	1609,33	40,00	4,05		257	2158490,00	10410,00	201,88
	65537	2627,00	40,00	4,12		65537	1341110,00	10379,00	201,43
	0	2468,67	40,00	4,35		0	941820,00	10364,00	224,29
3072	3	2778,33	107,00	7,67	16384	3	1166500,00	12981,00	211,55
	5	2997,33	114,00	7,63		5	2770440,00	13020,00	212,36
	17	2391,33	107,00	7,73		17	1895600,00	12671,00	212,88
	257	4050,00	108,00	7,97		257	2025870,00	13357,00	221,67
	65537	5370,00	113,00	8,38		65537	1820930,00	12823,00	240,29
	0	4366,33	114,00	9,81		0	1360350,00	12949,00	259,19

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в чотири потоки (with improvements and 4 threads).

У таблиці 43 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-5,27 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,05-1,86 рази в залежності від розміру відкритої експоненти для ключів довжиною 2048 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,10 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 43. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,25	0,33	0,85	4096	3	1,00	1,74	1,01
	5	0,24	0,33	0,85		5	2,15	1,59	1,07
	17	0,19	0,33	0,79		17	2,16	1,72	1,01
	257	0,19	0,33	0,82		257	2,38	1,59	1,01
	65537	0,27	0,33	0,76		65537	1,86	1,74	1,04
	0	0,26	0,50	0,62		0	1,76	1,70	1,07
1024	3	0,52	0,70	0,82	7168	3	1,41	1,81	1,01
	5	0,61	0,55	0,87		5	2,61	1,82	1,02
	17	0,88	0,60	0,92		17	3,49	1,81	1,01
	257	0,56	0,67	0,92		257	2,42	1,67	1,03
	65537	0,44	0,67	0,92		65537	0,68	1,70	1,00
	0	0,56	0,60	0,78		0	1,84	1,67	1,06
1536	3	0,54	1,00	0,82	8192	3	2,99	1,56	1,02
	5	0,62	1,00	0,92		5	1,25	1,49	1,02
	17	0,55	1,00	0,96		17	4,12	1,56	1,07
	257	0,85	1,05	0,88		257	1,82	1,47	1,01
	65537	0,73	1,11	0,98		65537	2,68	1,56	1,01
	0	0,92	1,00	0,85		0	3,06	1,49	1,06
2048	3	1,44	0,91	0,86	15360	3	5,05	1,86	1,02
	5	0,79	1,30	0,92		5	1,43	1,86	1,00
	17	1,38	1,26	0,90		17	1,88	1,86	1,04
	257	1,39	1,20	0,91		257	1,92	1,85	1,02
	65537	0,61	1,33	0,90		65537	1,37	1,86	1,01
	0	0,90	1,20	1,02		0	3,52	1,86	1,09
3072	3	1,79	1,50	1,10	16384	3	5,27	1,78	1,01
	5	2,47	1,42	1,07		5	1,40	1,75	1,09
	17	3,06	1,50	1,01		17	1,55	1,81	1,02
	257	1,26	1,60	1,02		257	2,26	1,71	1,02
	65537	1,33	1,42	1,03		65537	1,03	1,78	1,03
	0	1,25	1,42	1,03		0	2,53	1,77	1,09

У таблиці 44 наведені результати вимірювання швидкодії операцій криптосистеми RSA з запропонованими методами та розпаралелюванням у декілька потоків для 32-бітних машинних слів.

Таблиця 44. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads)

With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	402,667	8,00	0,40	4096	3	15340,00	184,00	20,66
	5	445,333	5,00	0,40		5	21081,30	189,00	19,99
	17	556,667	4,00	0,42		17	31031,30	150,00	20,40
	257	498,333	8,00	0,52		257	24348,00	146,00	21,01
	65537	419,667	7,00	0,54		65537	21754,30	147,00	21,34
	0	438,667	5,00	0,75		0	31321,70	144,00	22,31
1024	3	1032,670	11,00	1,37	7168	3	78027,70	638,00	61,77
	5	1107,670	12,00	1,40		5	119983,00	785,00	61,10
	17	1307,670	11,00	1,41		17	201490,00	609,00	62,44
	257	978,000	13,00	1,52		257	111605,00	798,00	62,00
	65537	883,667	19,00	1,53		65537	112330,00	645,00	62,39
	0	865,667	12,00	1,95		0	54061,00	607,00	64,64
1536	3	2144,33	22,00	3,15	8192	3	46520,70	884,00	79,28
	5	2384,67	23,00	3,07		5	128821,00	925,00	80,42
	17	3084,67	26,00	3,03		17	61623,70	959,00	77,62
	257	3056,0	23,00	3,22		257	59574,30	1014,00	82,49
	65537	2628,00	34,00	3,30		65537	104869,00	996,00	81,31
	0	2278,00	22,00	3,83		0	79249,70	1016,00	86,04
2048	3	4052,00	35,00	5,10	15360	3	603221,00	5973,00	235,19
	5	3453,33	39,00	5,51		5	1013340,00	5466,00	243,01
	17	5000,33	37,00	5,46		17	607904,00	5819,00	231,08
	257	4967,00	40,00	5,49		257	629372,00	5765,00	238,09
	65537	4034,67	44,00	5,57		65537	516249,00	5769,00	244,34
	0	4883,67	56,00	6,23		0	500741,00	5688,00	240,28
3072	3	11354,70	82,00	11,74	16384	3	1188280,00	6418,00	269,40
	5	6612,67	90,00	11,77		5	1938460,00	6438,00	251,42
	17	6017,33	127,00	11,50		17	1814270,00	6566,00	271,45
	257	8093,67	103,00	11,79		257	2368290,00	6448,00	252,45
	65537	5164,33	80,00	12,08		65537	2024760,00	6350,00	279,35
	0	8070,33	81,00	12,47		0	1703600,00	6413,00	269,23

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 32-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 45 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,05-7,94 рази в залежності від розміру відкритої експоненти для ключів довжиною 7168 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,20-3,66 рази в залежності від розміру відкритої експоненти для ключів довжиною 2048 біт та більше.

Таблиця 45. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Desktop з використанням 32-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у декілька потоки (with improvements and multi threads)

With improvements / With improvements and multi threads (*w=32* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,08	0,13	0,55	4096	3	1,10	2,11	0,67
	5	0,06	0,20	0,58		5	1,26	2,04	0,73
	17	0,04	0,25	0,55		17	0,60	2,57	0,69
	257	0,05	0,13	0,44		257	0,85	2,64	0,72
	65537	0,08	0,14	0,48		65537	0,91	2,63	0,73
	0	0,06	0,20	0,39		0	0,67	2,69	0,80
1024	3	0,23	0,64	0,64	7168	3	1,40	3,49	0,70
	5	0,24	0,50	0,65		5	1,61	2,84	0,71
	17	0,19	0,55	0,65		17	1,05	3,65	0,69
	257	0,21	0,46	0,67		257	1,95	2,79	0,75
	65537	0,18	0,32	0,63		65537	0,79	3,45	0,73
	0	0,25	0,50	0,58		0	4,43	3,66	0,84
1536	3	0,27	0,91	0,60	8192	3	7,94	3,26	0,70
	5	0,23	0,87	0,67		5	1,32	3,14	0,70
	17	0,20	0,77	0,70		17	5,19	3,00	0,76
	257	0,23	0,87	0,62		257	4,31	2,84	0,66
	65537	0,21	0,59	0,64		65537	3,48	2,89	0,75
	0	0,32	0,91	0,63		0	2,35	2,83	0,81
2048	3	0,50	1,40	0,72	15360	3	5,38	3,23	0,83
	5	0,44	1,23	0,64		5	2,24	3,53	0,79
	17	0,49	1,32	0,68		17	3,63	3,31	0,90
	257	0,45	1,20	0,67		257	6,59	3,34	0,86
	65537	0,40	1,20	0,66		65537	3,55	3,34	0,84
	0	0,46	0,86	0,71		0	6,61	3,39	1,01
3072	3	0,44	1,96	0,72	16384	3	5,18	3,61	0,80
	5	1,12	1,80	0,70		5	2,00	3,54	0,92
	17	1,22	1,27	0,68		17	1,61	3,48	0,80
	257	0,63	1,68	0,69		257	1,93	3,54	0,89
	65537	1,39	2,01	0,71		65537	0,93	3,59	0,88
	0	0,67	2,00	0,81		0	2,02	3,57	1,05

Далі розглядаються результати вимірювання для 64-бітних машинних слів.

У таблицях 46 і 47 наведені результати вимірювання швидкодії операцій криптосистеми RSA без запропонованих методів для 64-бітних машинних слів.

Таблиця 46. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів без застосування запропонованих методів (original)

Original (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	7,66667	0,10	0,07	4096	3	4970,00	69,00	3,68
	5	7,6667	0,10	0,07		5	7013,00	69,00	3,70
	17	5,6667	0,10	0,07		17	2621,33	68,00	3,76
	257	12,3333	0,10	0,07		257	3495,00	68,00	3,82
	65537	6,6667	0,10	0,07		65537	3774,33	69,00	3,94
	0	8,3333	0,10	0,09		0	5991,67	69,00	5,13
1024	3	44,6667	1,00	0,24	7168	3	31428,70	350,00	11,19
	5	40,6667	1,00	0,24		5	43367,70	353,00	11,34
	17	42,3333	1,00	0,24		17	32808,30	349,00	11,23
	257	50,667	1,00	0,25		257	22748,30	368,00	11,53
	65537	37,3333	1,00	0,28		65537	28512,30	351,00	11,86
	0	44,3333	1,00	0,34		0	32328,70	364,00	15,38
1536	3	176,333	3,00	0,54	8192	3	47080,30	517,00	14,65
	5	169,667	3,00	0,55		5	49655,00	526,00	15,03
	17	146,667	3,00	0,53		17	100363,00	522,00	14,78
	257	199,667	3,00	0,56		257	88095,70	521,00	14,93
	65537	145,6670	3,00	0,58		65537	34198,70	525,00	15,59
	0	134,6670	3,00	0,72		0	79398,70	525,00	19,87
2048	3	530,667	8,00	1,06	15360	3	397558,00	3289,00	51,30
	5	537,333	8,00	0,91		5	345107,00	3298,00	52,17
	17	555,333	8,00	0,92		17	457027,00	3331,00	50,92
	257	357,333	8,00	0,94		257	512134,00	3322,00	52,09
	65537	382,000	8,00	0,98		65537	707097,00	3342,00	54,07
	0	299,333	8,00	1,25		0	561511,00	3313,00	68,85
3072	3	1238,67	31,00	2,19	16384	3	575400,00	4017,00	58,43
	5	2062,000	29,00	2,08		5	846877,00	4017,00	59,48
	17	1730,00	30,00	2,08		17	884545,00	4048,00	58,70
	257	1186,00	30,00	2,13		257	283945,00	4032,00	60,51
	65537	1827,33	29,00	2,17		65537	1396880,00	3988,00	61,95
	0	1143,00	29,00	2,80		0	800941,00	3980,00	79,30

Таблиця 47. Результати вимірювання швидкодії арифметичних операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів (with improvements)

With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	8,6667	0,10	0,08	4096	3	4443,67	61,00	3,64
	5	7,0000	0,10	0,06		5	3529,33	65,00	3,64
	17	5,6667	0,10	0,06		17	3631,00	64,00	3,71
	257	6,3333	0,10	0,06		257	3555,00	63,00	3,71
	65537	6,0000	0,10	0,07		65537	4804,33	64,00	3,84
	0	8,0000	0,10	0,08		0	3428,00	65,00	4,90
1024	3	41,0000	1,00	0,24	7168	3	13216,00	314,00	10,91
	5	47,667	1,00	0,2		5	33783,00	313,00	11,09
	17	38,667	1,00	0,24		17	24022,30	314,00	11,15
	257	31,667	1,00	0,25		257	28803,30	318,00	11,21
	65537	34,333	1,00	0,25		65537	24730,70	316,00	11,80
	0	55,333	1,00	0,32		0	24085,30	312,00	14,59
1536	3	191,333	3,00	0,55	8192	3	19564,70	473,00	14,53
	5	141,667	3,00	0,53		5	66696,30	506,00	14,69
	17	82,667	3,00	0,53		17	77502,70	467,00	14,66
	257	145,333	3,00	0,55		257	36677,70	479,00	14,45
	65537	141,667	3,00	0,55		65537	47924,30	464,00	15,22
	0	86,000	3,00	0,70		0	51641,30	465,00	18,80
2048	3	378,67	8,00	0,92	15360	3	567433,00	2950,00	50,73
	5	344,33	8,00	0,89		5	666713,00	2985,00	50,66
	17	352,33	8,00	0,91		17	268177,00	2951,00	50,34
	257	346,00	8,00	0,92		257	481871,00	3047,00	51,81
	65537	366,33	7,00	0,94		65537	353852,00	3000,00	53,48
	0	390,00	8,00	1,19		0	1035380,00	3004,00	68,04
3072	3	1111,00	25,00	2,12	16384	3	517478,00	3635,00	58,04
	5	2038,67	26,00	2,05		5	498183,00	3629,00	58,14
	17	1257,33	25,00	2,05		17	1363210,00	3614,00	58,06
	257	1613,67	26,00	2,10		257	498837,00	3623,00	60,32
	65537	773,00	26,00	2,06		65537	528230,00	3687,00	61,83
	0	1506,00	28,00	2,78		0	883050,00	3625,00	78,30

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 48 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-2,64 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,04-1,24 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,17 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.

Таблиця 48. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Original / With improvements (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,88	1,00	0,88	4096	3	1,12	1,13	1,01
	5	1,10	1,00	1,17		5	1,99	1,06	1,02
	17	1,00	1,00	1,17		17	0,72	1,06	1,01
	257	1,95	1,00	1,17		257	0,98	1,08	1,03
	65537	1,11	1,00	1,00		65537	0,79	1,08	1,03
	0	1,04	1,00	1,13		0	1,75	1,06	1,05
1024	3	1,09	1,00	1,00	7168	3	2,38	1,11	1,03
	5	0,85	1,00	1,00		5	1,28	1,13	1,02
	17	1,09	1,00	1,00		17	1,37	1,11	1,01
	257	1,60	1,00	1,00		257	0,79	1,16	1,03
	65537	1,09	1,00	1,12		65537	1,15	1,11	1,01
	0	0,80	1,00	1,06		0	1,34	1,17	1,05
1536	3	0,92	1,00	0,98	8192	3	2,41	1,09	1,01
	5	1,20	1,00	1,04		5	0,74	1,04	1,02
	17	1,77	1,00	1,00		17	1,29	1,12	1,01
	257	1,37	1,00	1,02		257	2,40	1,09	1,03
	65537	1,03	1,00	1,05		65537	0,71	1,13	1,02
	0	1,57	1,00	1,03		0	1,54	1,13	1,06
2048	3	1,40	1,00	1,15	15360	3	0,70	1,11	1,01
	5	1,56	1,00	1,02		5	0,52	1,10	1,03
	17	1,58	1,00	1,01		17	1,70	1,13	1,01
	257	1,03	1,00	1,02		257	1,06	1,09	1,01
	65537	1,04	1,14	1,04		65537	2,00	1,11	1,01
	0	0,77	1,00	1,05		0	0,54	1,10	1,01
3072	3	1,11	1,24	1,03	16384	3	1,11	1,11	1,01
	5	1,01	1,12	1,01		5	1,70	1,11	1,02
	17	1,38	1,20	1,01		17	0,65	1,12	1,01
	257	0,73	1,15	1,01		257	0,57	1,11	1,00
	65537	2,36	1,12	1,05		65537	2,64	1,08	1,00
	0	0,76	1,04	1,01		0	0,91	1,10	1,01

Далі, у таблиці 49, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 49. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads)

With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	3,66667	0,10	0,07	4096	3	3753,00	60,00	3,49
	5	4,33333	0,10	0,07		5	2648,00	62,00	3,57
	17	4,00000	0,10	0,07		17	1836,00	61,00	3,54
	257	6,00000	0,10	0,07		257	1799,00	62,00	3,67
	65537	4,66667	0,10	0,07		65537	1369,00	63,00	3,66
	0	4,33333	0,10	0,09		0	4469,67	64,00	4,65
1024	3	15,0000	1,00	0,27	7168	3	20212,30	304,00	10,19
	5	27,667	1,00	0,28		5	13437,30	309,00	10,20
	17	21,3333	1,00	0,26		17	23263,70	305,00	10,34
	257	24,333	1,00	0,27		257	16850,00	304,00	10,40
	65537	40,6667	1,00	0,27		65537	14729,00	311,00	10,76
	0	32,667	1,00	0,34		0	17821,30	304,00	13,73
1536	3	74,3333	3,00	0,58	8192	3	19371,70	457,00	14,29
	5	80,667	3,00	0,63		5	14600,70	458,00	14,56
	17	56,667	3,00	0,57		17	21483,00	458,00	14,45
	257	80,667	3,00	0,65		257	17571,00	463,00	14,32
	65537	132,000	3,00	0,59		65537	49646,00	462,00	14,71
	0	119,667	4,00	0,72		0	29007,00	457,00	17,82
2048	3	266,000	8,00	0,95	15360	3	568601,00	2929,00	50,42
	5	177,000	8,00	0,94		5	418644,00	2953,00	50,07
	17	315,333	8,00	0,98		17	341713,00	2902,00	50,33
	257	255,333	8,00	1,00		257	308804,00	2909,00	50,86
	65537	199,333	8,00	1,00		65537	200275,00	2957,00	51,72
	0	166,667	8,00	1,31		0	426036,00	2997,00	65,60
3072	3	771,333	27,00	2,18	16384	3	508512,00	3541,00	57,23
	5	564,67	28,00	2,04		5	328161,00	3572,00	57,21
	17	684,000	28,00	2,08		17	411984,00	3591,00	57,33
	257	1124,33	27,00	2,16		257	326491,00	3589,00	59,37
	65537	714,67	27,00	2,24		65537	367522,00	3637,00	60,91
	0	886,33	27,00	2,88		0	464311,00	3578,00	77,14

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в 2 потоки (with improvements and 2 threads).

У таблиці 50 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-4,57 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,01-1,10 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,10 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.

Таблиця 50. Результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 2 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	2,36	1,00	1,14	4096	3	1,18	1,02	1,04
	5	1,62	1,00	0,86		5	1,33	1,05	1,02
	17	1,42	1,00	0,86		17	1,98	1,05	1,05
	257	1,06	1,00	0,86		257	1,98	1,02	1,01
	65537	1,29	1,00	1,00		65537	3,51	1,02	1,05
	0	1,85	1,00	0,89		0	0,77	1,02	1,05
1024	3	2,73	1,00	0,89	7168	3	0,65	1,03	1,07
	5	1,72	1,00	0,86		5	2,51	1,01	1,09
	17	1,81	1,00	0,92		17	1,03	1,03	1,08
	257	1,30	1,00	0,93		257	1,71	1,05	1,08
	65537	0,84	1,00	0,93		65537	1,68	1,02	1,10
	0	1,69	1,00	0,94		0	1,35	1,03	1,06
1536	3	2,57	1,00	0,95	8192	3	1,01	1,04	1,02
	5	1,76	1,00	0,84		5	4,57	1,10	1,01
	17	1,46	1,00	0,93		17	3,61	1,02	1,01
	257	1,80	1,00	0,85		257	2,09	1,03	1,01
	65537	1,07	1,00	0,93		65537	0,97	1,00	1,03
	0	0,72	0,75	0,97		0	1,78	1,02	1,05
2048	3	1,42	1,00	0,97	15360	3	1,00	1,01	1,01
	5	1,95	1,00	0,95		5	1,59	1,01	1,01
	17	1,12	1,00	0,93		17	0,78	1,02	1,00
	257	1,36	1,00	0,92		257	1,56	1,05	1,02
	65537	1,84	0,88	0,94		65537	1,77	1,01	1,03
	0	2,34	1,00	0,91		0	2,43	1,00	1,04
3072	3	1,44	0,93	0,97	16384	3	1,02	1,03	1,01
	5	3,61	0,93	1,00		5	1,52	1,02	1,02
	17	1,84	0,89	0,99		17	3,31	1,01	1,01
	257	1,44	0,96	0,97		257	1,53	1,01	1,02
	65537	1,08	0,96	0,92		65537	1,44	1,01	1,02
	0	1,70	1,04	0,97		0	1,90	1,01	1,02

У таблиці 51 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованими методами та розпаралелюванням у 4 потоків для 64-бітних машинних слів.

Таблиця 51. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в чотири потоки (with improvements and 4 threads)

With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Private key generation, ms	Digest sign, ms	Digest verify sign, ms
512	3	99,0000	2,00	0,08	4096	3	1654,33	51,00	3,84
	5	88,33	2,00	0,09		5	4265,33	52,00	3,83
	17	110,3330	2,00	0,09		17	6235,33	51,00	3,86
	257	98,3333	2,00	0,10		257	4530,67	51,00	3,89
	65537	91,6667	2,00	0,14		65537	3656,33	52,00	3,98
	0	76,6667	2,00	0,41		0	3413,00	52,00	4,81
1024	3	323,0000	4,00	0,28	7168	3	27682,30	201,00	11,71
	5	269,667	4,00	0,27		5	24862,30	200,00	11,81
	17	251,000	4,00	0,29		17	12885,70	201,00	11,85
	257	215,333	6,00	0,31		257	19943,30	201,00	11,95
	65537	266,667	4,00	0,34		65537	16318,00	212,00	11,97
	0	357,667	4,00	0,68		0	29021,70	204,00	14,17
1536	3	495,667	7,00	0,58	8192	3	20962,70	290,00	15,74
	5	452,000	7,00	0,59		5	21639,00	287,00	15,14
	17	522,667	7,00	0,59		17	20260,00	289,00	15,39
	257	555,667	8,00	0,60		257	23104,00	290,00	15,42
	65537	438,333	7,00	0,62		65537	22004,00	288,00	15,70
	0	507,000	8,00	0,94		0	20813,00	288,00	18,26
2048	3	1189,330	11,00	1,09	15360	3	120927,00	1657,00	51,87
	5	998,333	11,00	1,04		5	225259,00	1679,00	52,59
	17	585,000	11,00	0,99		17	105015,00	1677,00	53,71
	257	855,00	11,00	1,01		257	630471,00	1657,00	53,70
	65537	788,000	11,00	1,03		65537	292069,00	1668,00	54,24
	0	762,667	11,00	1,41		0	124960,00	1661,00	61,90
3072	3	2327,67	28,00	2,18	16384	3	185136,00	2091,00	60,71
	5	1524,33	27,00	2,17		5	296375,00	2026,00	61,19
	17	1833,67	28,00	2,23		17	322339,00	2058,00	61,58
	257	1970,33	28,00	2,25		257	169553,00	2012,00	61,96
	65537	1419,67	28,00	2,28		65537	237980,00	2014,00	61,61
	0	1646,00	28,00	2,97		0	179030,00	2026,00	71,98

Для оцінки ефективності запропонованих методів для платформи Mobile з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в 4 потоки (with improvements and 4 threads).

У таблиці 52 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 4 потоки для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,01-8,29 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,20-1,84 рази в залежності від розміру відкритої експоненти для ключів довжиною 4096 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,10 рази для деяких відкритих експонент для ключів довжиною 4096 біт та більше.

Таблиця 52. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням у 4 потоки (with improvements and multi threads)

With improvements / With improvements and 4 threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,09	0,05	1,00	4096	3	2,69	1,20	0,95
	5	0,08	0,05	0,67		5	0,83	1,25	0,95
	17	0,05	0,05	0,67		17	0,58	1,25	0,96
	257	0,06	0,05	0,60		257	0,78	1,24	0,95
	65537	0,07	0,05	0,50		65537	1,31	1,23	0,96
	0	0,10	0,05	0,20		0	1,00	1,25	1,02
1024	3	0,13	0,25	0,86	7168	3	0,48	1,56	0,93
	5	0,18	0,25	0,89		5	1,36	1,57	0,94
	17	0,15	0,25	0,83		17	1,86	1,56	0,94
	257	0,15	0,17	0,81		257	1,44	1,58	0,94
	65537	0,13	0,25	0,74		65537	1,52	1,49	0,99
	0	0,15	0,25	0,47		0	0,83	1,53	1,03
1536	3	0,39	0,43	0,95	8192	3	0,93	1,63	0,92
	5	0,31	0,43	0,90		5	3,08	1,76	0,97
	17	0,16	0,43	0,90		17	3,83	1,62	0,95
	257	0,26	0,38	0,92		257	1,59	1,65	0,94
	65537	0,32	0,43	0,89		65537	2,18	1,61	0,97
	0	0,17	0,38	0,74		0	2,48	1,61	1,03
2048	3	0,32	0,73	0,84	15360	3	4,69	1,78	0,98
	5	0,34	0,73	0,86		5	2,96	1,78	0,96
	17	0,60	0,73	0,92		17	2,55	1,76	0,94
	257	0,40	0,73	0,91		257	0,76	1,84	0,96
	65537	0,46	0,64	0,91		65537	1,21	1,80	0,99
	0	0,51	0,73	0,84		0	8,29	1,81	1,10
3072	3	0,48	0,89	0,97	16384	3	2,80	1,74	0,96
	5	1,34	0,96	0,94		5	1,68	1,79	0,95
	17	0,69	0,89	0,92		17	4,23	1,76	0,94
	257	0,82	0,93	0,93		257	2,94	1,80	0,97
	65537	0,54	0,93	0,90		65537	2,22	1,83	1,00
	0	0,91	1,00	0,94		0	4,93	1,79	1,09

У таблиці 53 наведені результати вимірювання швидкодії операцій криптосистеми RSA з використанням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів.

Таблиця 53. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Mobile з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads)

With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	394,6670	4,00	0,15	4096	3	8466,00	61,00	6,58
	5	404,33	6,00	0,15		5	15692,00	70,00	6,50
	17	406,3330	4,00	0,17		17	16501,70	54,00	6,64
	257	435,6670	7,00	0,23		257	11369,00	90,00	6,65
	65537	413,6670	7,00	0,34		65537	13914,00	75,00	6,61
	0	400,6670	6,00	1,13		0	15091,30	56,00	7,91
1024	3	811,33	11,00	0,50	7168	3	37303,00	172,00	19,70
	5	989,000	9,00	0,47		5	38953,00	162,00	19,66
	17	1152,330	9,00	0,52		17	73857,30	174,00	19,49
	257	893,67	9,00	0,50		257	41401,30	164,00	19,81
	65537	965,667	9,00	0,61		65537	39837,70	204,00	19,15
	0	955,333	14,00	1,34		0	46789,70	157,00	21,47
1536	3	2085,330	14,00	1,03	8192	3	19744,00	226,00	26,07
	5	2544,000	22,00	1,04		5	54682,70	225,00	25,98
	17	2984,330	13,00	1,03		17	45253,00	227,00	26,25
	257	2851,00	13,00	1,14		257	32755,00	296,00	26,07
	65537	2862,00	14,00	1,17		65537	25900,00	278,00	26,47
	0	2204,33	18,00	1,72		0	28005,30	220,00	27,59
2048	3	2639,67	25,00	1,65	15360	3	212752,00	1144,00	88,24
	5	3307,33	31,00	1,75		5	283757,00	1159,00	87,61
	17	2782,67	22,00	1,72		17	119987,00	1030,00	85,10
	257	3136,33	24,00	1,81		257	151133,00	1108,00	85,57
	65537	3951,00	31,00	1,79		65537	225590,00	1102,00	86,20
	0	4161,33	19,00	2,39		0	129048,00	1058,00	93,78
3072	3	11426,70	54,00	3,72	16384	3	430561,00	1385,00	100,56
	5	3889,00	32,00	3,64		5	449615,00	1329,00	102,30
	17	3394,00	46,00	3,85		17	459412,00	1342,00	99,22
	257	4129,33	37,00	3,89		257	609881,00	1192,00	102,89
	65537	4010,67	55,00	4,02		65537	608062,00	1238,00	101,86
	0	4451,67	57,00	4,77		0	585103,00	1200,00	109,18

Для оцінки ефективності запропонованих методів для платформи Mobile (Intel Core-i7 6700HQ) з використанням 64-бітних машинних слів, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 54 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція генерування особистого ключа ефективніша в 1,11-8,02 рази в залежності від розміру відкритої експоненти для ключів довжиною 8192 біт та більше, за деякими виключеннями.
- Операція створення підпису ефективніша в 1,55-3,04 рази в залежності від розміру відкритої експоненти для ключів довжиною 7168 біт та більше.

Таблиця 54. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Mobile за результатами з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в декілька потоків (with improvements and multi threads) для 64-бітних машинних слів

With improvements / With improvements and multi threads (*w=64* bit)									
Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Private key generation	Digest sign	Digest verify sign
512	3	0,02	0,03	0,53	4096	3	0,52	1,00	0,55
	5	0,02	0,02	0,40		5	0,22	0,93	0,56
	17	0,01	0,03	0,35		17	0,22	1,19	0,56
	257	0,01	0,01	0,26		257	0,31	0,70	0,56
	65537	0,01	0,01	0,21		65537	0,35	0,85	0,58
	0	0,02	0,02	0,07		0	0,23	1,16	0,62
1024	3	0,05	0,09	0,48	7168	3	0,35	1,83	0,55
	5	0,05	0,11	0,51		5	0,87	1,93	0,56
	17	0,03	0,11	0,46		17	0,33	1,80	0,57
	257	0,04	0,11	0,50		257	0,70	1,94	0,57
	65537	0,04	0,11	0,41		65537	0,62	1,55	0,62
	0	0,06	0,07	0,24		0	0,51	1,99	0,68
1536	3	0,09	0,21	0,53	8192	3	0,99	2,09	0,56
	5	0,06	0,14	0,51		5	1,22	2,25	0,57
	17	0,03	0,23	0,51		17	1,71	2,06	0,56
	257	0,05	0,23	0,48		257	1,12	1,62	0,55
	65537	0,05	0,21	0,47		65537	1,85	1,67	0,57
	0	0,04	0,17	0,41		0	1,84	2,11	0,68
2048	3	0,14	0,32	0,56	15360	3	2,67	2,58	0,57
	5	0,10	0,26	0,51		5	2,35	2,58	0,58
	17	0,13	0,36	0,53		17	2,24	2,87	0,59
	257	0,11	0,33	0,51		257	3,19	2,75	0,61
	65537	0,09	0,23	0,53		65537	1,57	2,72	0,62
	0	0,09	0,42	0,50		0	8,02	2,84	0,73
3072	3	0,10	0,46	0,57	16384	3	1,20	2,62	0,58
	5	0,52	0,81	0,56		5	1,11	2,73	0,57
	17	0,37	0,54	0,53		17	2,97	2,69	0,59
	257	0,39	0,70	0,54		257	0,82	3,04	0,59
	65537	0,19	0,47	0,51		65537	0,87	2,98	0,61
	0	0,34	0,49	0,58		0	1,51	3,02	0,72

Далі будуть розглядатися результати отримані для платформи Embedded (Cortex-A72)

У таблицях 55 та 56 наведені результати вимірювання швидкодії операції криптосистеми для різної довжини ключа без та з запропонованими методами для платформи Mobile для 64-бітних машинних слів. Крум того, для платформи Embedded генерація ключів не виконувалась, оскільки є досить ресурсоємною операцією, особливо при великих значеннях довжини ключа. Для створення та перевірки підпису використовувались попередньо згенеровані ключі.

Таблиця 55. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Embedded без застосування запропонованих методів (original)

Original							
Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms
512	3	1,90	0,21	3072 bit	3	75,90	6,25
	5	0,60	0,19		5	75,70	6,21
	17	0,60	0,19		17	75,60	6,23
	257	0,60	0,19		257	75,90	6,32
	65537	0,60	0,21		65537	75,50	6,68
	0	0,60	0,24		0	75,60	7,41
1024	3	5,50	0,75	4096	3	260,70	11,43
	5	4,40	0,73		5	259,90	11,56
	17	4,40	0,78		17	259,70	11,64
	257	4,30	0,79		257	259,50	11,63
	65537	4,30	0,81		65537	260,40	12,33
	0	4,40	0,93		0	259,40	13,54
1536	3	14,40	1,74	7168	3	1538,60	35,70
	5	14,30	1,75		5	1343,20	37,91
	17	14,40	1,75		17	1341,90	34,74
	257	14,30	1,77		257	1343,30	36,88
	65537	14,30	2,11		65537	1343,00	39,79
	0	14,70	2,41		0	1429,80	43,65
2048	3	33,20	2,95	8192	3	1983,20	45,93
	5	32,80	2,97		5	1991,10	46,43
	17	32,90	2,99		17	1985,30	46,91
	257	32,90	3,11		257	1987,60	48,85
	65537	32,80	3,14		65537	1984,00	49,34
	0	32,80	3,56		0	1986,10	57,22

Таблиця 56. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Embedded з застосуванням запропонованих методів (with improvements)

With improvements							
Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms
512	3	2,20	0,50	3072	3	71,90	6,20
	5	1,50	0,47		5	71,70	6,20
	17	1,60	0,49		17	71,90	6,09
	257	1,50	0,48		257	72,00	6,16
	65537	1,50	0,40		65537	71,70	6,64
	0	0,60	0,24		0	72,00	7,40
1024	3	5,40	0,74	4096	3	249,10	11,28
	5	4,20	0,72		5	248,30	11,26
	17	4,30	0,76		17	247,90	11,57
	257	4,20	0,77		257	250,30	11,57
	65537	4,20	0,79		65537	248,70	12,11
	0	4,30	0,89		0	248,50	13,34
1536	3	13,80	1,71	7168	3	1360,30	35,68
	5	13,80	1,72		5	1303,00	33,79
	17	13,90	1,73		17	1312,90	34,22
	257	13,80	1,76		257	1303,50	35,79
	65537	13,80	1,83		65537	1304,40	38,99
	0	14,30	2,08		0	1304,70	41,33
2048	3	31,70	2,92	8192	3	1935,30	45,55
	5	31,60	2,93		5	1929,90	46,10
	17	31,70	2,96		17	1938,90	45,42
	257	31,70	3,01		257	1935,70	47,02
	65537	31,50	3,11		65537	1938,50	47,89
	0	31,60	3,54		0	1934,70	55,25

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій в криптосистемі RSA без використання запропонованих методів (original) до результатів з використанням запропонованих методів (with improvements).

У таблиці 57 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів для зазначених операцій:

- Операція створення підпису ефективніша в 1,01-1,13 рази в залежності від розміру відкритої експоненти для ключів довжиною 1024 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,16 рази в залежності від розміру відкритої експоненти для ключів довжиною 1024 біт та більше.

Таблиця 57. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Embedded за результатами без та з використанням запропонованих методів для 64-бітних машинних слів

Original / With improvements							
Ключ, біт	Public exponent	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Digest sign	Digest verify sign
512	3	0,86	0,43	3072	3	1,06	1,01
	5	0,4	0,4		5	1,06	1,00
	17	0,38	0,4		17	1,05	1,02
	257	0,4	0,4		257	1,05	1,02
	65537	0,4	0,51		65537	1,05	1,01
	0	1	1,01		0	1,05	1,00
1024	3	1,02	1,01	4096	3	1,05	1,01
	5	1,05	1,01		5	1,05	1,03
	17	1,02	1,03		17	1,05	1,01
	257	1,02	1,03		257	1,04	1,01
	65537	1,02	1,03		65537	1,05	1,02
	0	1,02	1,04		0	1,04	1,02
1536	3	1,04	1,02	7168	3	1,13	1,00
	5	1,04	1,02		5	1,03	1,12
	17	1,04	1,01		17	1,02	1,02
	257	1,04	1,01		257	1,03	1,03
	65537	1,04	1,15		65537	1,03	1,02
	0	1,03	1,16		0	1,10	1,06
2048	3	1,05	1,01	8192	3	1,02	1,01
	5	1,04	1,01		5	1,03	1,01
	17	1,04	1,01		17	1,02	1,03
	257	1,04	1,03		257	1,03	1,04
	65537	1,04	1,01		65537	1,02	1,03
	0	1,04	1,01		0	1,03	1,04

Далі, у таблиці 58, будуть розглядатися результати вимірювання швидкодії операцій в криптосистемі RSA з застосуванням запропонованих методів та розпаралелювання генерації ключів у 2 потоки.

Таблиця 58. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Embedded з застосуванням запропонованих методів та розпаралелюванням в два потоки (with improvements and 2 threads)

With improvements and 2 threads							
Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms
512	3	2,20	0,20	3072	3	71,20	5,85
	5	0,60	0,19		5	71,30	5,95
	17	0,60	0,19		17	71,20	5,94
	257	0,60	0,19		257	71,40	6,11
	65537	0,60	0,21		65537	71,20	6,39
	0	0,60	0,24		0	71,30	7,26
1024	3	5,10	0,71	4096	3	238,30	10,83
	5	4,10	0,72		5	237,50	10,76
	17	4,30	0,76		17	238,40	11,06
	257	4,20	0,76		257	242,00	11,12
	65537	4,20	0,79		65537	238,40	11,86
	0	4,30	0,89		0	237,70	12,67
1536	3	13,80	1,71	7168	3	1317,40	33,57
	5	13,80	1,72		5	1283,40	33,54
	17	13,80	1,73		17	1302,40	33,90
	257	13,70	1,75		257	1292,00	35,42
	65537	13,80	1,77		65537	1290,30	38,27
	0	13,70	2,07		0	1301,30	40,40
2048	3	31,70	2,92	8192	3	1915,50	45,13
	5	31,60	2,92		5	1915,00	45,00
	17	31,60	2,95		17	1918,20	45,06
	257	31,70	3,01		257	1915,50	46,34
	65537	31,50	3,11		65537	1917,60	47,44
	0	31,50	3,54		0	1914,20	54,79

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з використанням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads).

У таблиці 59 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 2 потоки для зазначених операцій:

- Операція створення підпису ефективніша в 1,01-2,67 рази в залежності від розміру відкритої експоненти для всіх довжин ключа, що розглядаються, за деякими виключеннями.
- Операція перевірки підпису ефективніша в 1,01-2,54 рази в залежності від розміру відкритої експоненти для ключів довжиною 1536 біт та більше.

Таблиця 59. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Embedded з використанням 64-бітних машинних слів з застосуванням запропонованих методів та розпаралелюванням в 2 потоки (with improvements and 2 threads)

With improvements / With improvements and 2 threads							
Ключ, біт	Public exponent	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Digest sign	Digest verify sign
512	3	1,00	2,53	3072	3	1,01	1,06
	5	2,50	2,52		5	1,01	1,04
	17	2,67	2,54		17	1,01	1,03
	257	2,50	2,53		257	1,01	1,01
	65537	2,50	1,96		65537	1,01	1,04
	0	1,00	1,00		0	1,01	1,02
1024	3	1,06	1,04	4096	3	1,05	1,04
	5	1,02	1,00		5	1,05	1,05
	17	1,00	1,00		17	1,04	1,05
	257	1,00	1,02		257	1,03	1,04
	65537	1,00	1,00		65537	1,04	1,02
	0	1,00	1,00		0	1,05	1,05
1536	3	1,00	1,00	7168	3	1,03	1,06
	5	1,00	1,00		5	1,02	1,01
	17	1,01	1,00		17	1,01	1,01
	257	1,01	1,01		257	1,01	1,01
	65537	1,00	1,04		65537	1,01	1,02
	0	1,04	1,01		0	1,00	1,02
2048	3	1,00	1,00	8192	3	1,01	1,01
	5	1,00	1,00		5	1,01	1,02
	17	1,00	1,00		17	1,01	1,01
	257	1,00	1,00		257	1,01	1,01
	65537	1,00	1,00		65537	1,01	1,01
	0	1,00	1,00		0	1,01	1,01

У таблиці 60 наведені результати вимірювання швидкодії операцій криптосистеми RSA з застосуванням запропонованих методів та розпаралелюванням в 4 потоки з використанням 64-бітних машинних слів.

Таблиця 60. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Embedded з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements and 4 threads							
Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms
512	3	7,00	0,25	3072	3	58,30	5,47
	5	5,50	0,23		5	59,20	5,52
	17	5,60	0,25		17	58,60	5,66
	257	5,30	0,28		257	58,50	5,62
	65537	5,30	0,36		65537	60,30	5,90
	0	5,40	0,65		0	58,50	6,51
1024	3	11,60	0,79	4096	3	174,40	11,20
	5	11,30	0,77		5	171,00	11,15
	17	11,40	0,81		17	169,70	11,43
	257	11,20	0,87		257	169,20	11,43
	65537	11,20	0,92		65537	174,00	11,78
	0	11,30	1,19		0	172,90	12,62
1536	3	23,20	1,75	7168	3	720,50	35,43
	5	22,50	1,75		5	917,60	33,58
	17	22,50	1,78		17	733,30	33,70
	257	22,40	1,83		257	918,20	34,05
	65537	22,40	1,92		65537	735,80	34,81
	0	22,40	2,33		0	733,10	37,78
2048	3	36,80	2,94	8192	3	1062,80	45,19
	5	36,50	2,94		5	1056,70	45,20
	17	36,70	2,97		17	1062,30	45,28
	257	37,00	4,70		257	1060,00	46,36
	65537	63,10	3,15		65537	1055,80	47,56
	0	37,50	3,65		0	1051,60	49,54

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів з розпаралелюванням в 4 потоки (with improvements and 4 threads).

У таблиці 61 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в 4 потоки для зазначених операцій:

- Операція створення підпису ефективніша в 1,19-1,89 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,14 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.

Таблиця 61. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Embedded з застосуванням запропонованих методів та розпаралелюванням в 4 потоки (with improvements and 4 threads)

With improvements / With improvements and 4 threads							
Ключ, біт	Public exponent	Digest sign	Digest verify sign	Ключ, біт	Public exponent	Digest sign	Digest verify sign
512	3	0,31	2,00	3072	3	1,23	1,13
	5	0,27	2,07		5	1,21	1,12
	17	0,29	1,95		17	1,23	1,08
	257	0,28	1,73		257	1,23	1,10
	65537	0,28	1,12		65537	1,19	1,12
	0	0,11	0,37		0	1,23	1,14
1024	3	0,47	0,94	4096	3	1,43	1,01
	5	0,37	0,94		5	1,45	1,01
	17	0,38	0,93		17	1,46	1,01
	257	0,38	0,89		257	1,48	1,01
	65537	0,38	0,86		65537	1,43	1,03
	0	0,38	0,75		0	1,44	1,06
1536	3	0,59	0,98	7168	3	1,89	1,01
	5	0,61	0,98		5	1,42	1,01
	17	0,62	0,98		17	1,79	1,02
	257	0,62	0,97		257	1,42	1,05
	65537	0,62	0,95		65537	1,77	1,12
	0	0,64	0,89		0	1,78	1,09
2048	3	0,86	0,99	8192	3	1,82	1,01
	5	0,87	1,00		5	1,83	1,02
	17	0,86	0,99		17	1,83	1,00
	257	0,86	0,64		257	1,83	1,01
	65537	0,50	0,99		65537	1,84	1,01
	0	0,84	0,97		0	1,84	1,12

У таблиці 62 наведені результати вимірювання швидкодії операцій криптосистеми RSA з запропонованими методами та розпаралелюванням у декілька потоків.

Таблиця 62. Результати вимірювання швидкодії операцій в криптосистемі RSA для різної довжини ключа для платформи Embedded з застосуванням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads)

With improvements and multi threads							
Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms	Ключ, біт	Public exponent	Digest sign, ms	Digest verify sign, ms
512	3	7,30	0,25	3072	3	43,50	5,61
	5	5,70	0,30		5	43,30	5,68
	17	5,80	0,26		17	43,50	5,81
	257	5,90	0,29		257	43,50	5,75
	65537	5,70	0,37		65537	43,60	6,00
	0	5,80	0,75		0	43,40	6,48
1024	3	12,40	0,79	4096	3	118,20	11,06
	5	12,40	0,77		5	118,20	10,94
	17	12,30	2,06		17	117,80	11,21
	257	14,20	1,43		257	213,50	11,26
	65537	11,70	0,92		65537	117,80	11,56
	0	11,50	1,25		0	116,40	12,10
1536	3	20,20	1,80	7168	3	654,60	35,47
	5	25,80	1,80		5	441,80	33,62
	17	19,60	1,83		17	642,30	33,83
	257	19,30	1,87		257	441,20	34,12
	65537	25,20	1,95		65537	443,10	38,61
	0	19,60	2,28		0	440,40	38,78
2048	3	35,50	3,03	8192	3	625,80	44,80
	5	28,80	3,07		5	625,50	44,62
	17	29,00	3,07		17	628,50	44,38
	257	29,00	3,12		257	626,60	46,96
	65537	28,40	3,27		65537	628,60	47,38
	0	35,20	3,60		0	631,70	48,50

Для оцінки ефективності запропонованих методів для платформи Embedded, наводиться відношення результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів (with improvements) до результатів вимірювання операцій в криптосистемі RSA з використанням запропонованих методів та розпаралелюванням в декілька потоків (with improvements and multi threads).

У таблиці 63 наведені нормалізовані результати, для наглядної демонстрації ефективності запропонованих методів. Ефективність запропонованих методів з розпаралелюванням в декілька потоків для зазначених операцій:

- Операція створення підпису ефективніша в 1,17-3,09 рази в залежності від розміру відкритої експоненти для ключів довжиною 3072 біт та більше.
- Операція перевірки підпису ефективніша в 1,01-1,14 рази лише для відкритої експоненти 0 для ключів довжиною 3072 біт та більше.

Таблиця 63. Нормалізовані результати вимірювання швидкодії операцій криптосистеми RSA для різної довжини ключа для платформи Embedde з застосуванням запропонованих методів та розпаралелюванням у декілька потоки (with improvements and multi threads)

With improvements / With improvements and multi threads							
Ключі	Public exponent	Digest sign	Digest verify sign	Ключі	Public exponent	Digest sign	Digest verify sign
512	3	0,30	1,96	3072	3	1,65	1,11
	5	0,26	1,57		5	1,66	1,09
	17	0,28	1,89		17	1,65	1,05
	257	0,25	1,66		257	1,66	1,07
	65537	0,26	1,08		65537	1,64	1,11
	0	0,10	0,32		0	1,66	1,14
1024	3	0,44	0,94	4096	3	2,11	1,02
	5	0,34	0,94		5	2,10	1,03
	17	0,35	0,37		17	2,10	1,03
	257	0,30	0,54		257	1,17	1,03
	65537	0,36	0,86		65537	2,11	1,05
	0	0,37	0,71		0	2,13	1,10
1536	3	0,68	0,95	7168	3	2,08	1,01
	5	0,53	0,95		5	2,95	1,01
	17	0,71	0,95		17	2,04	1,01
	257	0,72	0,94		257	2,95	1,05
	65537	0,55	0,94		65537	2,94	1,01
	0	0,73	0,92		0	2,96	1,07
2048	3	0,89	0,96	8192	3	3,09	1,02
	5	1,10	0,96		5	3,09	1,03
	17	1,09	0,96		17	3,08	1,02
	257	1,09	0,96		257	3,09	1,00
	65537	1,11	0,95		65537	3,08	1,01
	0	0,90	0,99		0	3,06	1,14