BULLETIN OF NATIONAL ACADEMY OF SCIENCES
OF THE REPUBLIC OF KAZAKHSTAN
ISSN 1991-3494
Volume 4, Number 380 (2019), 38 – 45 **https://doi.org/10.32014/2019.2518-1467.90**

UDC 004.056
IRSTI 81.93.29

S. Tynymbayev[1], R. Sh. Berdibayev[1], T. Omar[1],
S. A. Gnatyuk[2], T. A. Namazbayev[3], S. Adilbekkyzy[1]

[1]Almaty university of Power Engineering and Telecommunication, Almaty, Kazakhstan,
[2]National aviation university, Kyiv, Ukraine,
[3]al-Farabi Kazakh national university, Almaty, Kazakhstan.
E-mail: s.tynym@mail.ru, r.berdybaev@aues.kz, oturkal17@gmail.com,
s.qnatyuk@nau.edu.ua, tirnagog@mail.ru, sairaccn.02.95@mail.ru

DEVICES FOR MULTIPLYING MODULO NUMBERS WITH ANALYSIS OF THE LOWER BITS OF THE MULTIPLIER

Abstract. Various approaches of modulo multiplying multi-bit (large) numbers in modulus are considered. An algorithm for multiplying numbers is given, where the modular multiplication process is divided into steps, and in each step, by combining the multiplication operations of the previous partial remainder by two with the operation of reducing the multiplication results modulo, partial remainders is formed. The circuit diagrams of multipliers of numbers modulo with the analysis of the lower bits of the multiplier with the sequential and matrix formation of remainders are considered. The proposed modulo multipliers do not require pre-calculations and all calculations do not go beyond the bit grid of the module.

Keywords: public-key cryptosystem, hardware encryption, modular multiplication, remainder former.

Introduction. In asymmetric cryptosystems, data encryption and decryption procedures are performed by modular exponentiation of the number a to the power x modulo P (a^x *modP*), which can be implemented in hardware and/or software [1, 2]. Hardware encryption has several significant advantages over software encryption, one of which is higher speed [3]. Hardware implementation ensures its integrity. At the same time, the generation and storage of keys, as well as encryption, are carried out in the encoder board itself, and not in the computer's RAM. Thus, the security of the implementation of the algorithm itself is ensured, which is also an important advantage. Therefore, the development of high-speed operating units of hardware cryptoprocessors for asymmetric encryption, despite their high cost, is an urgent task.

Approaches to the multiplication modulo. Modular multiplication of numbers can be done in three ways. In the first method, the operation is divided into two stages. At the first stage, n-bit numbers A and B are multiplied and a 2n-bit number C is formed. At the second stage, the product C = A*B is reduced by the module P.

Nowadays, a great deal of experience has been gained in the development of high-speed integer multipliers and devices for squaring. These include Brown, Wallace multipliers, Dadda multipliers, systolic and vedic multipliers and quadrants, where the computational complexity is O (n^2) bit operations. But these multipliers are very effective in calculating "low-bit" numbers, which are widely used in the construction of operating units of computers of various classes [4].

In cryptography for multiplication of multi-bit numbers, which allow to calculate the required product faster than O (n^2) steps (bit operations), the Karatsuba method [5], whose complexity is $O(n^{\log_2 3})$, the Toom -Cook algorithm [6] with complexity of order $O(n2^{\sqrt{2\log_2 n}})$ bit operations. And the Shengghe-Strassen algorithm [7] allows to multiply two n-bit numbers for O(nlog$n$ log$n$) bit operations.

The modular reduction operation, which is performed in the second stage, is the receipt of the remainder of dividing the product C = A*B by the module P. In [8], various ways of modular reduction of the numbers were analyzed. It is shown that the most effective construction tool is a modular device based on a dividing device. Part of such a dividing device includes a partial remainder former. Based on partial remainder formers, high-performance matrix and pipeline devices of modular reduction are easily implemented [9-13].

In the second modular multiplication method, using the Barrett or Montgomery algorithms [14-16], the process of multiplying large numbers by the module is accelerated. However, these algorithms require preliminary calculations associated with the need to use the algorithm for dividing large numbers, therefore representing the greatest complexity:

– Barrett algorithm requires constant predictions

$$\mu = \left\lfloor \frac{d^{2m}}{N} \right\rfloor$$

where $d = 2^k$, k-size of a word in bits, m-number of words in module. The effectiveness of the Barrett algorithm depends entirely on how effectively the preliminary calculations will be performed, which are performed by dividing large numbers.

– for the Montgomery algorithm, prediction of the constant "$r^2(modN)$", is required, using division with remainder.

In the third method, the process of multiplying modulo numbers is performed in sets of steps, where its number is determined by the number of bits of the multiplier.

Depending on which bit of the multiplier multiplication begins, two types of the multiplier structure can be distinguished:

- modulo multiplier, where multiplication begins with the analysis of the lower bits;
- modulo multiplier, where multiplication begins with the analysis of the higher order bits of the multiplier.

The paper deals with the first type of multiplier. In such a multiplier, the following actions are performed at each multiplication step:

– the partial remainder former PRF_i calculates the partial remainder r_i. For what the previous partial remainder r_{i-1}, shifted by one bit towards the higher order bits, is reduced modulo P, i.e. $r_i = 2r_{i-1} modP$. When forming the first partial remainder r_1, the previous partial remainder is $r_0 = A$ (multiplicand), then the value of remainder r_i is determined by the formula $r_1 = 2r_0 modP$.

– the partial remainder r_i is logically multiplied by the i-bit of the multiplier B by the block And_i. The input of the block And_0 is $r_0 = A$ and the value of the bit b_0 of the multiplier.

– the partial remainder r_i from the outputs of the block And_i and the intermediate remainder R_{i-1} from the previous modulo adder $MAdd_{i-1}$ is fed to the inputs of the modulo adder $MAdd_i$, where the operation on the formation of the intermediate remainder $R_i = (r_i + R_{i-1})modP$ and the result of operations is fed to the inputs of $AddM_{i+1}$.

After performing N at the outputs of the modulo adder the result is generated $R = R_{N-1} = = r_{N-1} + R_{N-2} modP$.

In turn, a modulo multiplier with the analysis of the lower bits of the multiplier can be constructed in two ways.

In the first method, all partial and intermediate remainders are formed sequentially as the next lower bits of the multiplier are analyzed on the same partial remainder former and modulo adder.

In the second method, a separate driver is allocated for the formation of each partial residue, and each intermediate residue is formed on its modulo adder, where the drivers and adders in the multiplier are arranged in a matrix.

The modulo multiplier of numbers sequential action, where multiplication begins with the analysis of the lower bits of the multiplier. The functional diagram of the multiplier of numbers modulo a sequence of actions is shown in figure 1. The multiplier includes the shift register RegB, where before the start of operations the number B (multiplier) is stored, the register RegP where the module P is stored, cumulative partial remainder former (CPRF) and the cumulative modulo adder (CMAdd), flip-flop T, counter of clock pulses (CCP), delay lines DL.1, DL.2, DL.3, blocks of logic circuits $And_1 \div And_{10}$ and OR.

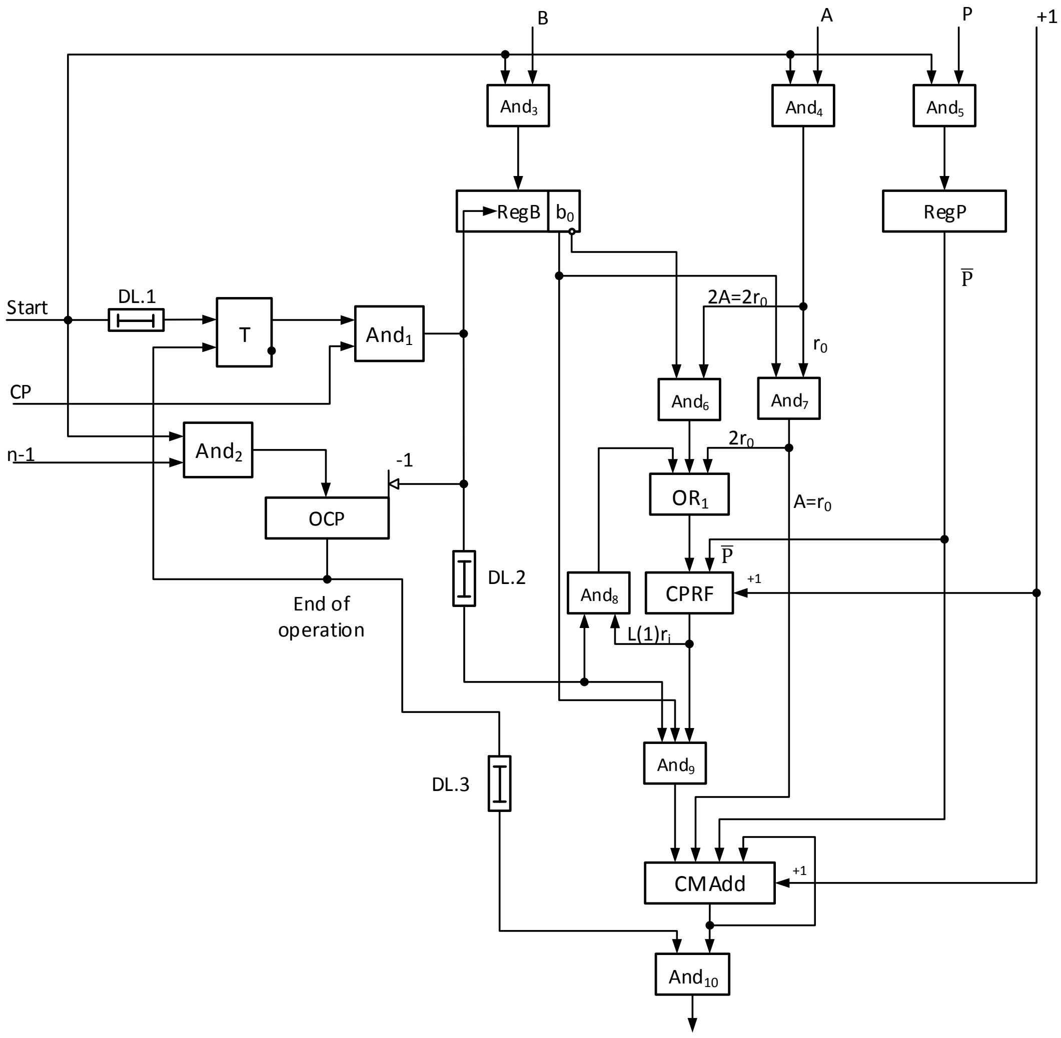

Figure 1 – Functional diagram of the multiplier of numbers modulo sequential action

Figure 2 shows the structure of the CPRF, which consists of the binary adder CM, the multiplexer MS, and the register of partial remainders RegPR.

The previous partial remainders (r_{i-1}) is fed to the left inputs of the adder with a shift by one bit in the direction of the higher order bits $(2 * r_{i-1})$. The second inputs of the adder Add are supplied with the bits of the return code of the module $\overline{P}$, and the low signal of the adder receives a single signal +1, which translates the return one complement code of the module into a two complement. In the process of adding $2 * r_{i-1}$ with P in the two complement, if $2 * r_{i-1} > P$, then the carry C = 1 occurs from the high-order bit of the adder, which controls the transfer to MS multiplexer output difference$r_i = 2r_{i-1} - P$. If $2r_{i-1} < P$, then we get the difference $2r_{i-1} - P$ with a negative sign (Sn = 1), which controls the transfer of the input code $2r_{i-1}$ and the result of the operation is stored in the register RegPR.

The structure of the cumulative modulo adder (CMAdd) is shown in figure 3. The CMAdd differs from the CPRF only by the adder Add, where the current partial remainder r_i is summed with the previous intermediate remainder (R_{i-1}). Then this sum is reduced modulo P, i.e. $R_i = (r_i + R_{i-1}) mod\ P$ and the value R_i is stored in the intermediate remainder register - RegR.

Consider the operation of the multiplier. On the “Start” signal, the operands B and P are received by the blocks of logic circuits And$_3$ and And$_5$, respectively, in the registers RegB and RegP. At the same time, the low-order bit of the multiplier B-b$_0$ is fixed in the low-order bit of the register RegB. The bits of the multiplicand A from the outputs of the block of circuit And$_4$ are fed to the inputs of the block of circuits And7 and shifted by one bit in the direction of the higher bits to the inputs of the block of circuits

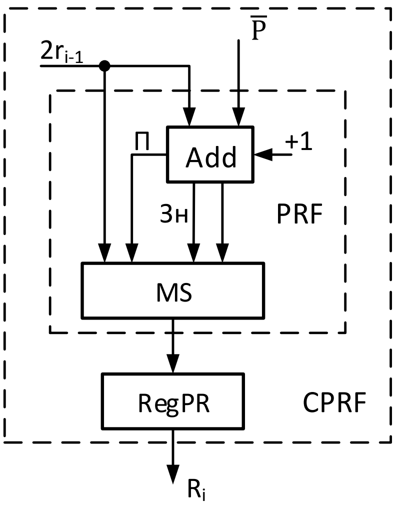

Figure 2 – CPRF structure

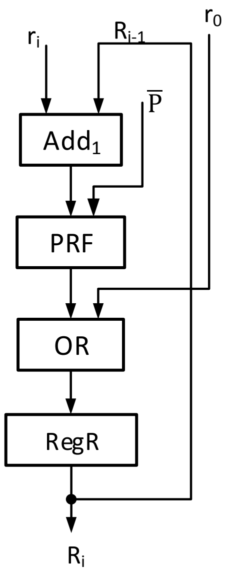

Figure 3 – CMAdd structure

And6. The input of the block of circuits And$_7$ also supplies the value of the bit b$_0$, and its inverse value $\overline{b_0}$ is fed to the input of the block of circuits And$_6$. The "Start" signal also records the number of shifts - N-1 (where N is the bits number of the multiplier) in the counter of clock pulses of the CCP.

After receiving the multiplier B in the register RegB, if $b_0 = 1$, then the bits of the multiplicand A=$r_0$through the circuit of the block And$_7$ is fed to the input of the register of the intermediate remainder RegR of the CMAdd. In addition, the value A=$r_0$with a shift by one bit in the direction of the higher order through the circuits And$_7$ and OR1 is fed to the inputs of the CPRF where $r_1 = 2r_0 mod\ P = 2r_0 + \overline{P} + 1$. is formed. r_1 is stored in the register RegPR of the CPRF.

When values $b_0 = 0$ from the outputs of the block of the circuits And$_4$, the value of A is shifted to the high-order side through the circuits And6 and OR$_1$ is fed to the inputs of the CPRF, where $r_1 = 2r_0 mod\ P$ is formed, which is also stored in the registers of the RegPR of the CPRF. The low bit level $b_0 = 0$ prohibits the output $r_0 = A$ to the output of the block of circuits And$_7$. By the time of formation and storage of the first partial remainder r_1 from the output of the delay lines DL.1, the "Start" signal is fed to the flip-flop T input, which translates the flip-flop T to the single state, and allows the passage of the 1st clock pulse CP1 to the multiplier circuit. CP1 reduces the readings of the counter CCP by one and shifts the contents of the RegB register to the right by one bit. At the time of the shift of the register RegB, CP1 delaying on the delay lines DL.2 arrives at the control inputs of the block of circuits And$_8$ and And$_9$. If, after the shift in the low order PrB, $b_1 = 1$ is fixed, then the contents of RegPR of the CPRF are transmitted through the block of circuits And$_8$ to the input of the CMAdd, where the intermediate remainder $R_1 = (R_0 + r_1) mod\ P$ is formed, which is stored by RegR. At the same time, the pulse CP1from the output of the RegPR of the CPRF through the block of the And$_8$ and OR$_1$ circuit doubles the value $2r_1$ to the inputs of the CPRF, $r_2 = 2r_1 mod\ P$ and $r_2$are stored in the RegPR of the CPRF. By the end of the formation of r_2 in the register RegPR o and R_1, in the register RegR, the clock pulse CP2 arrives at the input of the multiplier, by which the contents of the register RegB are shifted by one bit to the left, reduces the readings of the counter CCP by one, forms a partial remainder r_3 in RegPR and the intermediate remainder r_3 in RegR, etc. After the filing of the n-1-th clock pulse, the counter CCP generates the "End of Operations" signal, which delays on DL.3 for the duration of the R_{n-1} result generation and is fed to the control input of the circuit block And$_{10}$ and the result of the operations is output. The "End of Operations" signal transfers the flip-flop T to the initial zero state and prevents the next clock signal from passing through the circuit And1 to the device. The parameters of the clock signals are determined by the delay signals on the CMAdd.

Consider the example of the multiplication of numbers by module.

Let A = 25; B = 2210 = 101102

P = 26. For convenience, all calculations are performed in decimal notation, which are shown in table 1.

Table 1 – The order of multiplication of A by B modulo R

Clock pulses	b_i	CPRF	CMAdd
Start	$b_0 = 0$ $b_1 = 1$	$r_1 = 2r_0 modP$= 50- 26=24	$R_0 = 0$ $R_1 = (R_0 + r_1)$mod26=24
CP1	$b_2 = 1$	$r_2 = 2r_1 modP$=48-26=22	$R_2 = (R_1 + r_2)$ mod P =24+22=46 mod 26 = 20
CP2	$b_3 = 0$	$r_3 = 2r_2 modP$=44-26-18	$R_3 = (R_2 + 0)$ mod P = 20
CP3	$b_4 = 1$	$r_4 = 2R_3 modP$=36-26=10	$R_4 = (R_3 + r_4)$ mod P = (20+10)mod 26 = 4
Checking: $R = (A \cdot B)modP = (25 \cdot 22)mod\ 26 = 550\ mod\ 26 = 4$.			

Matrix scheme of the device for modular multiplication with the analysis of the lower bits of the multiplier. Figure 4 shows the block diagram of the matrix multiplier of numbers, where multiplication begins with the lower order bits of the multiplier. The multiplier consists of the register of the multiplier RegB, the register of the module RegP, partial remainders former $PRF_1 \div PRF_{N-1}$, blocks of logic circuits $And_0 \div And_{N-1}$,, modulo adders $MAdd_1 \div MAdd_{N-1}$, delay line DL.3. The bits of the multiplier register in $в_{N-1}$, $в_{N-2}$, ..., $в_1$, are connected to the inputs of the block of circuits And_{N-1}, And_{N-2}, …, And_1,, respectively. The inverse value of the module P$\overline{P}$ of the register of RegP is connected with the inputs $PRF_1 \div PRF_{N-2}$ and $MAdd_1 \div MAdd_{N-2}$. The outputs of the PRF_i are connected with the inputs And_i and with the inputs of the next PRF_{i+1}. The outputs And_i are connected to the inputs of the $MAdd_i$. The $MAdd_i$ inputs are connected to the $MAdd_{i-1}$ outputs. The outputs $MAdd_i$ are connected to the inputs $MAdd_{i+1}$. Signal "+1" is fed to the inputs of $PRF_1 \div PRF_{N-2}$ and $MAdd_1 \div MAdd_{N-2}$.

Consider the operation of the matrix multiplier. The signal "Start", which is fed into the circuit through input 1, from input 2 is taken the bits of the module P in the register RegP, through input 3, the multiplicand A is taken to the input of the block And_0 and with a shift by one bit in the direction of the higher order bits is taken to input PRF.1, through input 4 the multiplier B is taken to the register RegB. In

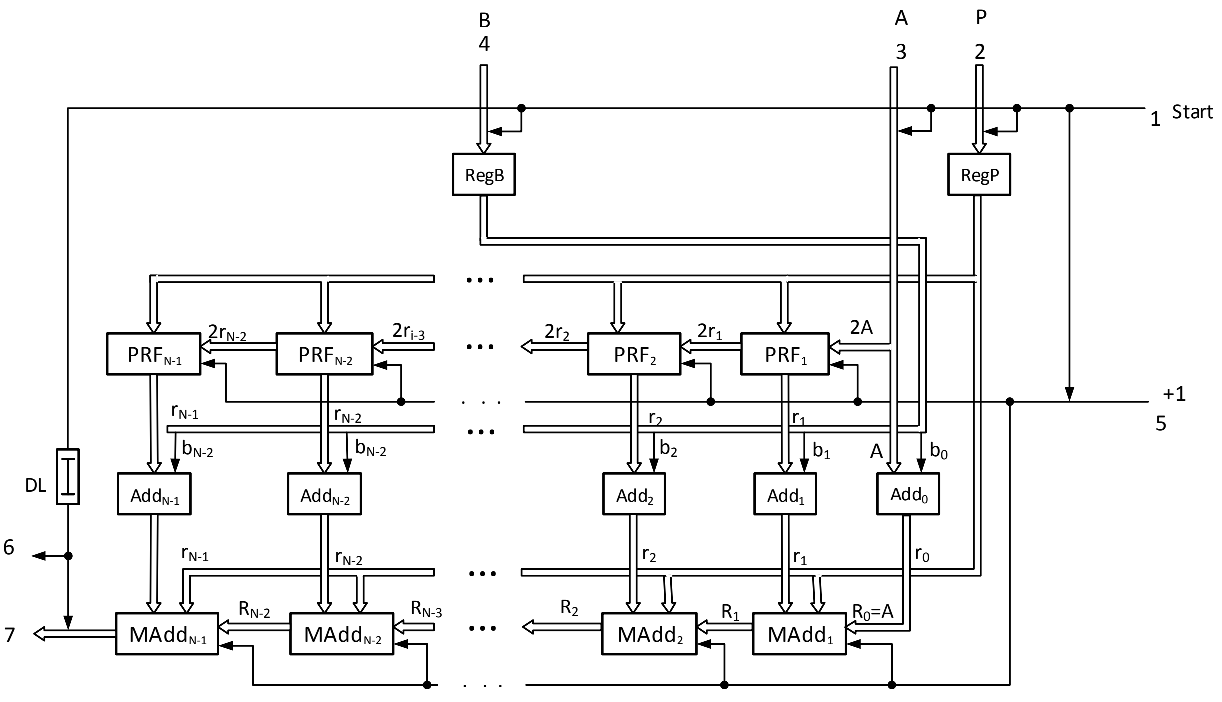

Figure 4 – Block diagram of the matrix multiplier of the numbers modulo (multiplication begins with the analysis of the lower order)

addition, the “Start” signal is fed to the input of the delay lines DL and receives from input 5 the signal “+ 1”. After receiving the multiplier B in the register RegB, module P in the register RegP and multiplicand A to the inputs PRF.1 and AND$_0$ and the signal “+ 1” is fed to the inputs PRF$_1$ ÷ PRF$_{N-1}$ and MAdd$_1$ ÷ MAdd$_{N-1}$. At the output of PRF.1, a partial remainder $r_1 = 2AmodP$ is formed, which is supplied with a shift by one bit towards the high order bits to the input of PRF$_2$ and without a shift of r_1 is transmitted to the information inputs of the block of circuits And$_1$, to the control input of which the value of bit $в_1$ from register RegB. When $b_1 = 1$, the value of r_1 from output And$_1$ is transmitted to the inputs of the modulo adder MAdd$_1$, and the second information inputs of which are fed the value $r_0 = R_0 = A$ at the output of MAdd$_1$, the intermediate remainder $R_1 = (r_1 + R_0)\ modP$ which is transmitted to the input of MAdd$_2$.

Similarly, partial remainder $r_3, \dots, r_{N-2}, r_{N-1}$ are formed at the outputs $PRF_3, \dots, PRF_{N-2}, PRF_{N-1}$ and intermediate remainder $R_3, \dots, R_{N-2}, R_{N-1}$.

At that time, at the PRF$_2$ outputs, a partial remainder $r_2 = 2r_1modP$ is formed, which, with a shift of one bit to the left towards the higher order bits, is transmitted to the input of the PRF$_3$. Partial remainder r_2 is simultaneously transmitted to the information inputs of the block of circuits And$_2$, and the control input of which is transferred to the value of bit $в_2$ from the register RegB. When $b_2 = 1$, the value of r_2 is transmitted to the input of the adder MAdd$_2$, to the second input of which the value R_1 is supplied from the output of MAdd$_1$. An intermediate remainder $R_1 = (r_1 + R_0)\ modP$ is formed at the output of MAdd$_2$.

Similarly, at the outputs $PRF_3, \dots, PRF_{N-2}, PRF_{N-1}$, partial remainders $r_3, \dots, r_{N-2}, r_{N-1}$ are sequentially formed. In parallel, partial remainders at the outputs of the $MAdd_3, \dots, MAdd_{N-2}, MAdd_{N-1}$ adders form intermediate remainders $R_3, \dots, R_{N-2}, R_{N-1}$.The remainder R_{N-1} is the result of multiplying the numbers A and B modulo P.

Figure 5 shows the structure of the adder modulo MAdd, which consists of a binary adder, where the partial remainder r_i is summed with the intermediate remainder R_{i-1} and this sum is reduced modulo using the PRF.

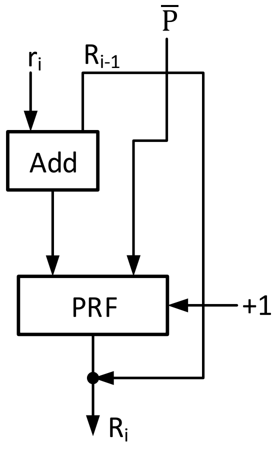

Figure 5 – Structure of MADD

Table 2 shows the order of execution of multiplication operations modulo the matrix multiplier, where A=27$_{10}$; B=23$_{10}$=10111$_2$; P=35$_{10}$. For convenience, all arithmetic operations are performed in the decimal number system.

Table 2 – Calculation order of the R = (27*23)*mod*35

PRF$_4$	PRF$_3$	PRF$_2$	PRF$_1$	PRF$_0$
$r_4 = 2r_3mod35=12$	$r_3 = 2r_2mod35=6$	$r_2 = 2r_1mod35=38-35=3$	$r_1 = 2Amod35=54-35=19$	–
And_4	And_3	And_2	And_1	And_0
$b_4 = 1$ $r_4 = 12$	$b_3 = 0$ $r_3 = 0$	$b_2 = 1$ $r_2 = 3$	$b_1 = 1$ $r_1 = 19$	$b_0 = 1$ $R_0 = A = 27$
MAdd$_4$	MAdd$_3$	MAdd$_2$	MAdd$_1$	
$R = R_4 = (R_2 + r_4)$ modP = (12+14)=26	$R_2 = (R_2 + r_3)$ modP = 14	$R_2 = (R_1 + r_2)$ modP = 14	$R_1 = (R_0 + r_1)$ modP = 11	–
Checking R = (27*23) = 621*mod*35 = 26.				

The magnitude of the delay on the DL determine the longest chain necessary for the formation of the result $R = R_{N-1}$: PRF$_1$ – And$_1$ – MAdd$_1$ ÷ MAdd$_{N-1}$. Then

$$\tau_{DL} = \tau_{PRF} + \tau_{And} + N - 1(\tau_{MAdd})$$

where τ_{PRF} is the delay time on the PRF; $\tau_{И}$ is the delay time of the AND circuit; τ_{MAdd} is the delay time on MADD.

Conclusion. In the proposed modulo multipliers, no precalculations is required; at each stage of the formation of the intermediate remainder, the multiplication and reduction operations are combined; All calculations do not go beyond the bit grid of the module.

Acknowledgement. The work was carried out by the authors within the framework of program-targeted financing of the Science Committee of the Ministry of Education and Science of the Republic of Kazakhstan (BR053236757 "Development of software and hardware and software for cryptographic protection of information during its transmission and storage in infocommunication systems and general purpose networks").

С. Тынымбаев[1], Р. Ш. Бердибаев[1], Т. Омар[1], С. А. Гнатюк[2], Т. А. Намазбаев[3], С. Әділбекқызы[1]

[1]Алматы энергетика және байланыс университеті, Алматы, Қазақстан,
[2]Ұлттық авиациялық университеті, Киев, Украина,
[3]әл-Фараби атындағы Қазақ ұлттық университеті, Алматы, Қазақстан

КӨБЕЙТКІШТІҢ ТӨМЕНГІ РАЗРЯДТАРЫН ТАЛДАУ АРҚЫЛЫ САНДАРДЫ МОДУЛЬ БОЙЫНША КӨБЕЙТУ ҚҰРЫЛҒЫСЫ

Аннотация. Көп таңбалы (үлкен) сандарды модуль бойынша көбейту жолдары қарастырылады. Сандарды көбейту алгоритмі берілген, онда модуль бойынша көбейту үрдісі қадамдарға бөлінеді және әр кезеңде, бұрынғы ішінара қалдықтың көбейту әрекеттерін модуль бойынша көбейту нәтижелерін моульге келтіру операциясы көмегімен біріктіру арқылы ішінара қалдықтар пайда болады. Көбейткіштің төменгі разрядтарын талдаумен сандардың көбейту құрылғысы мен қалдықтардың матрицалық қалыптасуы бар схемалық шешімдер қарастырылады. Ұсынылған модуль бойынша көбейту құрылғылары алдын-ала есептеулерді талап етпейді және барлық есептеулер модульдің разряд торынан тыс жүрмейді.

Түйін сөздер: ашық кілттік криптожүйе, аппараттық шифрлау, модульдік көбейту, қалдық құрастырушы.

С. Тынымбаев[1], Р. Ш. Бердибаев[1], Т. Омар[1], С. А. Гнатюк[2], Т. А. Намазбаев[3], С. Әділбекқызы[1]

[1]Алматинский университет энергетики и связи, Алматы, Казахстан,
[2]Национальный авиационный университет, Киев, Украина,
[3]Казахский национальный университет им. аль-Фараби, Алматы, Казахстан

УСТРОЙСТВА УМНОЖЕНИЯ ЧИСЕЛ ПО МОДУЛЮ, НАЧИНАЯ С АНАЛИЗА МЛАДШИХ РАЗРЯДОВ МНОЖИТЕЛЯ

Аннотация. Рассматриваются различные способы умножения многоразрядных (больших) чисел по модулю. Приводится алгоритм умножения чисел, где процесс умножения по модулю разбиваются на шаги и в каждом шаге путем совмещения операций умножения предыдущего частичного остатка на два с операцией приведения результатов умножения по модулю формируются частичные остатки. Рассмотрены схемные решения умножителей чисел по модулю с анализом младших разрядов множителя с последовательным и матричным формированием остатков. В предложенных умножителях по модулю не требуются выполнять предвычисления и все вычисления не выходят за разрядной сетки модуля.

Ключевые слова: криптосистема с открытым ключом, аппаратное шифрование, умножение чисел по модулю, формирователь остатков.

Information about authors:

Tynymbayev Sakhybay, Professor of the Department of Information security systems, Candidate of technical sciences, Almaty university of Power Engineering and Telecommunication, Almaty, Kazakhstan; s.tynym@mail.ru; https://orcid.org/0000-0002-9326-9476

Berdibayev Rat, Head of the Department of Information security systems, Candidate of political sciences, Almaty university of Power Engineering and Telecommunication, Almaty, Kazakhstan; r.berdybaev@aues.kz; https://orcid.org/0000-0002-8341-9645

Gnatyuk Sergiy, Associated professor, Doctor of science, National aviation university, Kyiv, Ukraine; s.qnatyuk@nau.edu.ua; https://orcid.org/0000-0003-4992-0564

Omar Turganbek, Associated Professor of the Department of Information security systems, Candidate of political sciences, Almaty university of Power Engineering and Telecommunication, Almaty, Kazakhstan; oturkal17@gmail.com; https://orcid.org/0000-0002-8014-8069

Namazbayev Timur, Senior lecturer of the Department of Solid state physics and Nonlinear Physics, Master of Engineering Science, al-Farabi Kazakh national university, Almaty, Kazakhstan; tirnagog@mail.ru; https://orcid.org/0000-0002-2389-2262

Adilbekkyzy Sairan, Engineer of the Department of Information security systems, Candidate of political sciences, Almaty university of Power Engineering and Telecommunication, Almaty, Kazakhstan; sairan.02.95@mail.ru; https://orcid.org/0000-0002-3929-7070

REFERENCES

[1] Ryabko B.Y., Fionov A.I. (2014). Fundamentals of modern cryptography for information technology professionals. M.: Scientific world, 2014. 173 p. (in Rus.).

[2] Akhmetov B.S., Korchenko A.G., Sidenko V.V., Drens Y.A., Seilova N.A. (2015). Applied cryptology: encryption methods. Almaty: KazNRTU after K. I. Satpayev, 2015. 496 p. (in Rus.).

[3] Aitkhozhayeva E.Zh., Tynymbayev S.T. (2014) Aspects of hardware reduction modulo in asymmetric cryptography // Bulletin of National academy of sciences of the Republic of Kazakhstan. 2014. Vol. 5. P. 88-93. ISSN 1991-349421 (in Rus.).

[4] Orlov S.A., Tsilker B.J. (2014). Organization of computers and systems / 3rd ed. SPb.: Peter, 2014. ISBN 978-5-496-01145-7 (in Rus.).

[5] Karatsuba A.A., Ofman Y.P. (1962). Multiplications of multi-digit numbers on automata // DANSSR. 1962. Vol. 145. P. 293-314 (in Rus.).

[6] Cook S.A., Aanderaa S.O. (1969). On the minimum computation time of functions. Trans. AMS, 142 (1969). P. 291-314.

[7] Schonhage A., Strassen V. (1973). Fast multiplication of large numbers: Cybernetic collection. 1973. Issue 2. P. 87-98 (in Rus.).

[8] Kovtun M., Kovtun V. (2017) Review and classification of algorithms for dividing and modulating large integers for cryptographic applications [Kompaniya Sayfer] [http://docplayer.ru/30671408-Obzor-i-klassifikaciya-algoritmov- deleniya-i-privedeniya-po-modulyu-bolshih-celyh-chisel-dlya-kriptograficheskih-prilozheniy. html] (in Rus.)

[9] Petrenko V.I., Chipiga A.F. (1995). Modulus multiplexer. Combination recurrent former of remainders. Patent of the Russian Federation. No. 2029435 (in Rus.).

[10] Petrenko V.I., Sidorchuk A.V., Kuz'minov J.V. (2007) Modulus multiplexer. Patent of the Russian Federation. No. 2299461 (in Rus.).

[11] Kopytov V.V., Petrenko V.I., Sidorchuk A.V. (2009). Device for generating remainder from arbitrary modulus of number. Patent of the Russian Federation. No. 2368942 (in Rus.).

[12] Tynymbayev S.T., Aitkhozhayeva Y.Zh., Adilbekkyzy S. (2018). High speed device for modular reduction // Bulletin of National academy of sciences of the Republic of Kazakhstan. 2018. Vol. 6, N 376. P. 147-152. ISSN 2518-1467 (Online). ISSN 1991-3494 (Print). https://doi.org/10.32014/2018.2518-1467.38

[13] Aitkhozhayeva E.Zh., Tynymbayev S.T. (2016) The remainder generator by an arbitrary modulus of the number. Patent of the RK. No. 30983 (in Rus.) .

[14] Tynymbayev S., Berdibaev R.S., Omar T., Shaikulova A.A., Magauin B. (2018). High-speed devices of reduction // Materials of the XIV International Asian School – Seminar "Problems of optimization of complex systems". July 20-31, 2018. Part 2. Almaty, 2018.

[15] Berrett P. (1987) Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor. Springer, Berlin, Heidelberg. DOI: 10.1007/3-540-47721-7_24 (in Rus.).

[16] Montgomery P.L. (1985). Modular Multiplication without Trial Division // Math. Compulation. Vol. 44, N 170 (Apr., 1985). P. 519-521. DOI: 10.20307/2007970.

[17] Pisek E., Henige T.M. (2013) Method and apparatus for efficient modulo multiplication. Patent US No. 8417756 B2.