Devices for Modular Multiplication of Numbers with Analysis of Two Least Significant Bits of the Multiplier

Sakhybay Tynymbayev [1] [0000-0002-9326-9476], Rat Berdibayev [1] [000-0002-8341-9645], Turganbek Omar [1][0000-0002-8014-8069], Sergiy Gnatyuk [2,3] [0000-0003-4992-0564], Timur Namazbayev [4] [0000-0002-2389-2262], Sairan Adilbekkyzy [1] [0000-0002-3929-7070]

[1] Almaty University of Power Engineering and Telecommunication, Almaty, Kazakhstan
[2] National Aviation University, Kyiv, Ukraine
[3] Yessenov University, Aktau, Kazakhstan
[4] Al-Farabi Kazakh National University, Almaty, Kazakhstan
r.berdybaev@aues.kz, s.gnatyuk@nau.edu.ua

Abstract. Various approaches to the modular multiplication of multi-bit (large) numbers are analyzed. The advantages and disadvantages of these approaches are given. We consider a circuit solution that implements the algorithm for multiplying numbers modulo, where at each step of multiplication two bits of the multiplier are analyzed, which reduces the number of multiplication steps. The proposed multiplier modulo does not pre-calculations and all calculations do not go beyond the bit grid of the module.

Keywords: cryptography, public-key cryptosystem, hardware encryption, modulo number multiplier, partial remainder formers.

1 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.

2 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

 CybHyg-2019: International Workshop on Cyber Hygiene, Kyiv, Ukraine, November 30, 2019.

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 (nlogn logn) 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, 10, 11, 12, 13].

In the second modular multiplication method, using the Barrett or Montgomery algorithms [14, 15, 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 = 2k, 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 numbers modulo is performed in many steps, where at each step partial and intermediate remainders are calculated and the number of steps is determined by the bit number of the multiplier.

In such multipliers, depending on which bits of the multiplier the multiplication begins, a multiplier can be distinguished, where the multiplication begins with the analysis of the least and the highest significant bits. The multiplier of numbers modulo, where the multiplication begins with the analysis of the least significant bit of the multiplier was considered in [17].

In this paper, we consider a multiplier, where multiplication is carried out with the analysis of the two least significant bits of the multiplier, which leads to reduce in the number of multiplication steps [18-21].

In the process of multiplication, at each step of multiplication, the following actions are performed:

- the partial remainder $r_i = 4r_{i-1} mod P$ is calculated;
- bit вi+1 of the multiplier B is multiplied by $2r_i$, and вi is multiplied by r_i, з then they are summed and this sum is reduced modulo P: $r_i' = (2r_i * в_{i+1} + r_i * в_i) mod P$;
- the intermediate remainder $R_i = (r_i' + R_0) mod P$ is calculated.

The number of such steps is determined by the number N/2, where N is the number of digits B. Since A and B < P, it follows that r0=A, therefore R0 is determined by the formula $R_0 = (2r_0 * в_1 + r_0 * в_0) mod P$.

3 High-speed device for modular multiplication of numbers

The block diagram of the considered multiplier is shown in Fig. 1. The device consists of four blocks: 1 - a block of registers Rg3P, RgP and RgB, where during execution of operations the values of the tripled module - 3P, module P and multiplier B are stored respectively; 2 - block of formers of partial remainders FPR.N/2-1, FPR.N/2-2, ..., FPR.2, FPR.1; 3 – block of multipliers modulo MMod.N/2-1, MMod.N/2-2, ..., MMod.1, MMod.0; 4 - block of adders of remainders modulo ARM.N/2-1, ARM.N/2-2, ..., ARM..2, ARM..1. The device also includes a delay element 5.

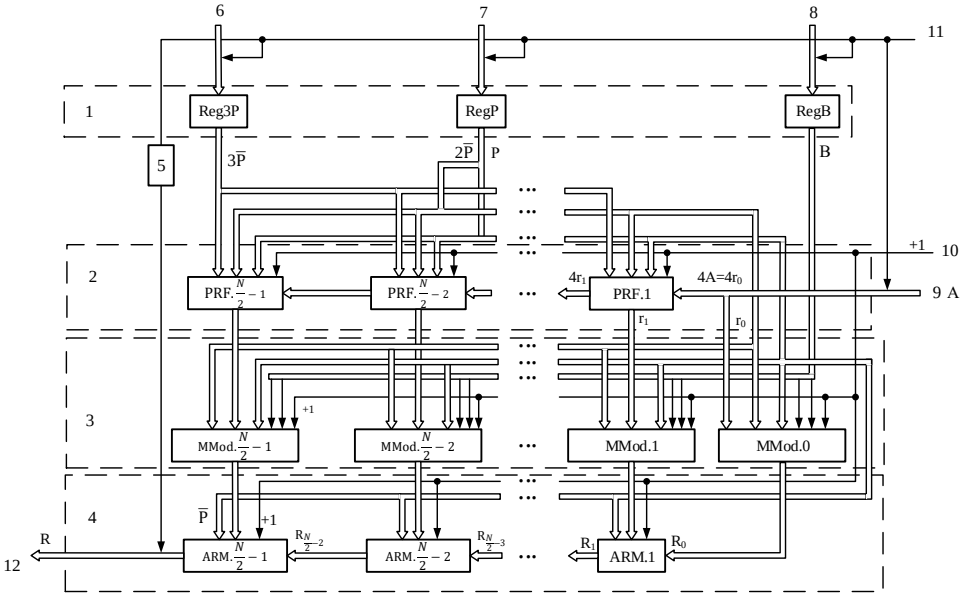

Fig. 1. Devices for modular multiplication of numbers with analysis of two least significant bits of the multiplier

To the inputs of the FPR are supplied from the outputs of the registers of the modules $3\overline{P}, 2\overline{P}$ and $\overline{P}$. The value of $2\overline{P}$ is formed by shifting $\overline{P}$ by one bit towards a higher

order of bit positions. From the outputs of the register RgP, the value $2\overline{P}$ and $\overline{P}$ are also fed to the inputs of the multipliers modulo MMod. Besides, from the RgB register outputs, a pair of bits starting from the low order bits (b1, b0), (b3, b2), ..., (bN-1, bN-2) are fed to the inputs of the multipliers MMod.0, MMod.1, ..., MMod.N/2-1. Signal “+1” is fed to the inputs FPR, MMod and ARM. The inputs for receiving the bits of the multiplicand A are connected to the inputs of Mod.0, FPR.1. At the inputs of MMod.0, bits A are fed without modifications, and to the inputs of FPR.1, they are fed with a shift by two bits towards a higher order. Further, the outputs of the FPR.i are fed to the inputs of the FPR.i+1 with a shift by two bits towards a higher order and without changes to the inputs of the MMod.i. The inputs ARM.i are connected with the outputs MMod.i and MMod.i-1, and the outputs ARM.i are connected to the inputs of the next ARM.i+1. The output of the device is the ARM.N/2-1 outputs.

A device for multiplying modulo numbers with the analysis of two digits of a multiplier per step works as follows. The values of 3P from input 6, P from input 7 and multiplier B from input 8 are received in registers Rg3P, RgP and RgB, and the multiplicand A from input 9 is fed to the inputs of FPR.1 with by a shift of two bits to the left and without a shift to the input of MMod.0 with the signal "Start", which is fed to the input 11. This signal is also fed to the input of the delay element 5, where it is delayed for the time of the formation of the final result R=(A*B)modP. After submitting the multiplier A=r0 by the MMod.0 circuit, the operation R0 = ((2*A*b1) + (A*b0))modP is performed, which is fed to the ARM.1 inputs. At the same time, the scheme of FPR.1 performs an operation to form a partial remainder r1=(4*A)modP. The partial remainder r1 from the outputs of the FPR.1 is fed to the inputs of the FPR.2 with a shift of two bits towards the higher order, i.e. 4r1. From the output of the former r1 without shift is also fed to the inputs of MMod.1, where the operation r1’ = ((2*r1*b3) + (r1*b2))modP is performed, which is fed to the inputs of ARM.1, where the intermediate remainder R1 = (r1’+R0)modP is formed and the value of R1 is fed to theARM.2 inputs.

Further, r1 from the outputs of the FPR.1 with a shift of two bits to the left is fed to the input of the FPR.2, where a partial remainder r2 = (4*r1)modP is formed. This remainder with a shift of two bits in the direction of the higher ones is fed to the inputs of the FPR.3 and without shift to the inputs of MMod.2. At the output of MMod.2, a partial remainder r2’ = ((2*r2*b5) + (r2*b4))modP is formed, which is transmitted to the ARM.2 inputs, where the intermediate remainder R2 = (r2’+R1)modP is calculated.

In the same way, R3, R4, …, RN/2-1 are formed. After the end of the calculation of RN/2-1 delayed signal "Start" on the delay element 5, the result is given to the output of device 12.

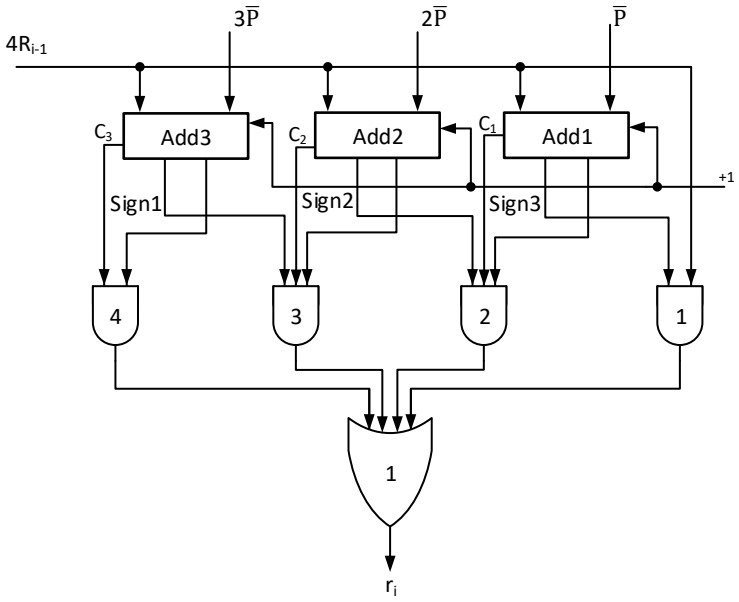

Fig. 2. Functional diagram PRF

Fig. 2 shows a functional diagram of the formers of partial remainders FPR. The FPR consists of three Add1, Add2, Add3 adders and four blocks of logical circuits AND.1 ÷ AND.4, a block of OR circuits. The partial remainder ri-1 with a shift by two bits to the higher order is fed to the left inputs of the adders and to the right inputs the values of the modules $3\overline{P}, 2\overline{P}$ and $\overline{P}$ are fed. At the inputs of the least significant bits of the adders, a signal "+1". The addition operation in the two's complements in all adders is performed simultaneously and at the same time carry signals from the bits of the sign C3, C2, C1 and the signs Sign3, Sign2, Sign1 on the corresponding adders are generated.

At the outputs of two or three adder's results can be formed with positive signs. Of these, the smallest remainder is chosen, while other positive and negative calculation values are blocked.

Table 1 shows the conditions for the formation of the smallest value of the remainder ri depending on the values of the carries C3, C2, C1 and signs Sign3, Sign2, Sign1.

Table 1. Conditions for the formation of the smallest positive reminder r_i

№	Carries			Signs			Adders outputs			$4r_{i-1}$
	C3	C2	C1	Sign3	Sign2	Sign1	Add3	Add2	Add1	
1	1	1	1	0	0	0	r_i	–	–	–
2	0	1	1	1	0	0	–	r_i	–	–
3	0	0	1	1	1	0	–	–	r_i	–
4	0	0	0	1	1	1	–	–	–	$r_i=4r_{i-1}$

From this table, as well as from Fig. 2, it can be seen that when C3 = C2 = C1 and Sign3 = Sign2 = Sign1 = 0, the smallest positive result is determined by the difference 4ri-1 – 3P, this difference is transmitted to the output of the circuit AND.4 by signal C3 = 1. At the same time, signals Sign3 = Sign2 = 0 block transmissions of codes from outputs Add2 and Add1 and the value 4ri-1 by circuits AND.1, AND.2 and AND.3. When the values of C3 = 0 and C2 = C1 = 1, positive differences are formed at the outputs of the adders Add2 and Add1. In this case, the outputs of the Add3 adder are blocked by the signal C3 = 0, and the signals Sign2 = Sign1 = 0 block the output of the adder Add1 and the submission of the value 4ri-1 to the outputs of the OR circuit and positive value of the ri is transmitted to the output of the FPR. When C3 = C2 = 0 and C1 = 1, the result ri is formed at the output of Add1. When C3 = C2 = C1 = 0, these signals block the outputs Add 3, Add2 and Add1 and by the signal Sign1 = 1, the value 4ri-1 is transmitted by the circuit AND.1 to the output of the FPR.

The functional diagram of the modular multiplier circuit (MMod) is shown in Fig. 3. The multiplier consists of the adder CM1 to the inputs of which are fed from the outputs of the blocks of circuits AND.1 and AND.2. The partial remainder ri and the bit bi+1 are fed to the inputs of the block of circuits AND.1, and the inputs of the block of the circuit AND.2 also supply ri and the bit bi from the register RgB. At the outputs of Add1, the sum ri(bi+1+bi) is formed, which is fed to the left inputs of the adder Add2 and Add3. The right-hand inputs of Add2 and Add3 are supplied with L(1) $\overline{P}$ (2$\overline{P}$) and $\overline{P}$, respectively, and the signal "+1" is applied to the lowest order bit position.

Table 2 shows the conditions for the formation of the smallest positive remainder ri', depending on the values of the carries C2 and C1 and the signs Sign2 and Sign1.

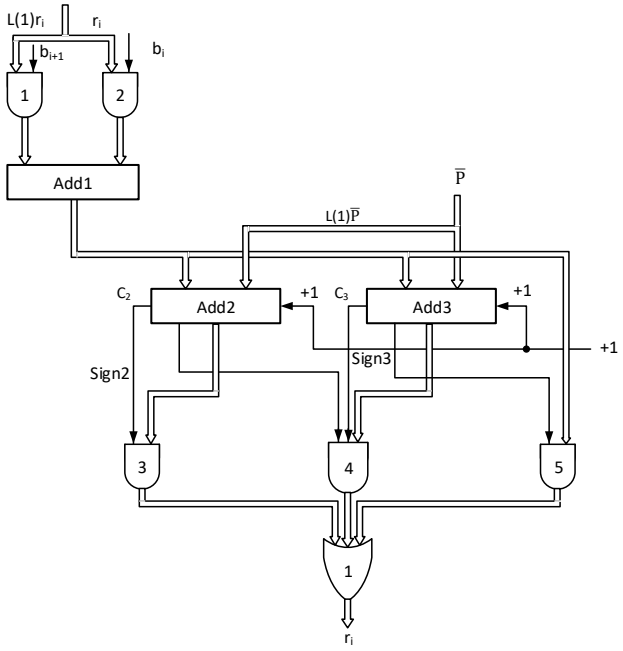

Fig. 3. Functional diagram MMod

Table 2. Conditions for the formation of the smallest positive remainder r_i'

№	C2	Sign2	C1	Sign1	Adders outputs		$3r_i$
					Add2	Add3	-
1	1	0	1	0	r_i'	-	
2	0	1	1	0	-	r_i'	
3	0	1	0	1	-	-	r_i'=3 r_i

Fig. 4 shows the functional diagram of the adder of remainders modulo P. On the adder Add1, partial remainder ri'is summed with the previous intermediate residue Ri-1 and the resulting sum ri'+Ri-1 is modulo P using the adder Add2 and the block diagram AND.1, AND.2 and block diagram OR.

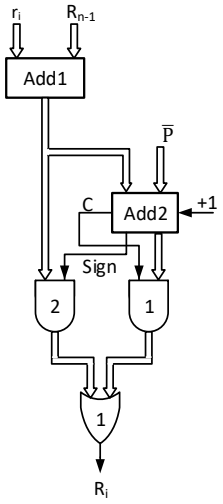

Fig. 4. shows the functional diagram of the adder of remainders modulo P.

Consider an example of multiplying numbers modulo. Let A=3710, B=3910=1001112, P=7710, 2P=15410 and 3P=23110.

For clarity, all calculations are performed in decimal notation.

MMod.0	$r_0' = R_0 = ((2*b_1*A) + (b_0*A))$modP $r_0' = R_0 = ((2*1*37) + (1*37))$mod77 = 34	$b_1\ b_0$ 1 1
FPR.1	$r_1 = (4*A)$modP = (4*37)mod7 = 148 – 77 = 71	
MMod.1	$r_1' = ((2*b_1*r_1) + (b_2*r_1))$modP $r_1' = ((2*0*71) + (1*71))$mod77=71	$b_3\ b_2$ 0 1
ARM.1	$R_1 = (r_1' + R_0)$modP = (71+34)mod77 = 28	
FPR.2	$r_2 = (4*r_1)$modP = $(4*r_1)$ – 3P= 284 – 231 = 53	
MMod.2	$r_2' = ((2*b_5*r_2) + (b_4*r_2))$modP $r_2' = ((2*1*53) + (0*53))$mod77 = 106 – 77 = 29	$b_5\ b_4$ 1 0
ARM.2	$R_1 = (r_2' + R_1)$modP = (29+28)mod77 = 57	

Checking (A*B)modP = (37*39)mod77 = 57 = 1443mod77=57

Thus, when multiplying modulo numbers, analyzing at each step of multiplying two bits of the multiplier, starting with the least significant bits, we can speed up the multiplication process twice.

The practical part of the work was implemented on FPGAs (Field Programmable Gate Arrays). FPGA is a device for implementing high-performance logic circuits. In our case, the Nexys 4 Artix-7 FPGA Trainer Board, which is a platform for developing digital logic circuits based on the Artix-7 FPGA from Xilinx, was taken as the basis. FPGA Artix-7 at its base has 101,440 logic cells and 126,800 CLB Flip-Flops. The complexity of the implemented logic circuits depends directly on the number of FPGA resources. In our case, the resources of Artix-7 are enough to implement the device of modular multiplication of numbers with the analysis of two bits of the multiplier. About 1- 5% of the FPGA board's resources are enough to implement the circuit of a device with 8, 10, ... 24 bits. This suggests that based on Artix-7 FPGAs, it is possible to implement devices that multiply numbers modulo with much more bit depth.

Below are the results of the studies in the form of a timing diagrams (waveforms) intended for the device based on the Artix-7 FPGA.

Figures 4, 5 show the diagrams of the operation of the devices of multiplication of numbers modulo with the analysis of two bits of the multiplier. In the diagram, the designation r corresponds to the formers of partial remainders (FPR), rr - multipliers modulo (Mmod), and R - adders of remainders modulo (ARM). Figure 4 shows the operation of the device for 8 bit numbers. Since the device analyzes two bits of the multiplier, the operating time is also reduced by two.

In Fig. 4, on the front edge of the clock pulse CP1, the least two bits (B1B0) of register B are read and based on them the value of the register rr (Mmod) and R (ARM) is formed, which are equal and have the value rr0 = R0 = 141. And the value of the register r0 (FPR) will be equal to the value of the register A = 181. After the supply of CP2, the value of the register r is formed, which is calculated as $r_1 = (4 * A) \bmod P = 61$, the value of the module P will have one of three values. The values of the module P in our case are P1 = 221, P2 = 442, P3 = 663, which corresponds to the P and its doubled and tripled the values. In the case of register rr, the module P can have the same values as the FPR (register r). And at this moment in the register B the next two bits (B3B2) will be read, with the help of which the values of the registers rr1 and R1 will be formed. At this measure, the value of the rr1 and R1 registers will be different, and they will be calculated as $rr_1 = \big((2 * B_3 * r_1) + (B_2 * r_1)\big) mod\ P = 122$ and $R_1 = (rr_1 + R_0) mod\ P = 42$. The next step CP3 generates the value of the FPR in the register r2 = 23. After receiving the value of the FPR, the value of the registers rr2 and R2, which corresponds to the values calculated as $rr_2 = \big((2 * B_5 * r_2) + (B_4 * r_2)\big) mod\ P = 23$ and $R_2 = (rr_2 + R_1) mod\ P = 66$. In this measure, the values of the registers r_2 and rr_2 are equal, because $B_5B_4 = 01_2$. From this it follows that the value obtained in the calculation is equal to the value of r_2 itself, and it is currently less than the value of the module P. At the last fourth step (CP4), the register value$r_3 = 92$ is also first generated, which is used to calculate the Mmod value and is recorded in register $rr_3 = 184$. The desired remainder from operations $R = (A * B) mod\ P$ is generated in the adders of remainders modulo (ARM) and displayed in register R_3, which has a value of 28.

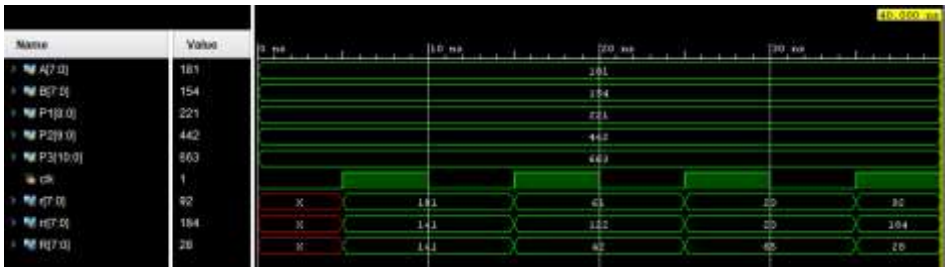

Fig. 5. Diagram of the algorithm for an 8-bit number

Figure 5 shows a diagram of the operation of the device of modular multiplication of numbers with the analysis of two bits of the multiplier, but for working with 10-bit numbers. This time, the full cycle of calculations takes 5 cycles, which is two times less than the number itself. At the first clock cycle (CP1) the value of the registers r_0, rr_0, R_0 will have a value of 441, 656, 656, respectively. At CP2, the value of the FPR will be $\llbracket \mathrm{r}_1 = (4 * \mathrm{A}) \bmod \mathrm{P} = 430$, where register A = 441. The Mmod and ARM values will have the value $rr_1 = \big((2 * B_3 * r_2) + (B_2 * r_1)\big) mod\ P = 193$ and $R_1 = (rr_1 + R_0) mod\ P = 182$. The next clock pulse (CP3) generates the following values in the registers$r_2 = 386$, $rr_2 = 105$ and $R_2 = 287$. During CP4, the FPR has a value of $r_3 = 210$, and the registers for storing Mmod and ARM data are $rr_3 = 420$ и $R_3 = 40$. The last CP5 forms the remainder, which is obtained during the operation $R = (A * B) mod\ P = 213$, where the value A = 441, B = 421 and P = 667.

Fig. 6. Diagramm of the algorithm for an 10-bit number

Figure 6 shows a diagram of the operation of the device. The remainders for number A were also calculated with a bit capacity of 12, 14, 16 and 24. Figure 6 shows the dependence of the spent FPGA resource Artix-7 on the bit capacity of the reduced number A. In this figure, LUT (Look-Up Table - conversion table), FF - Flip-Flops, BUFG – a global clock buffer that is independent of architecture, IO – inputs and outputs.

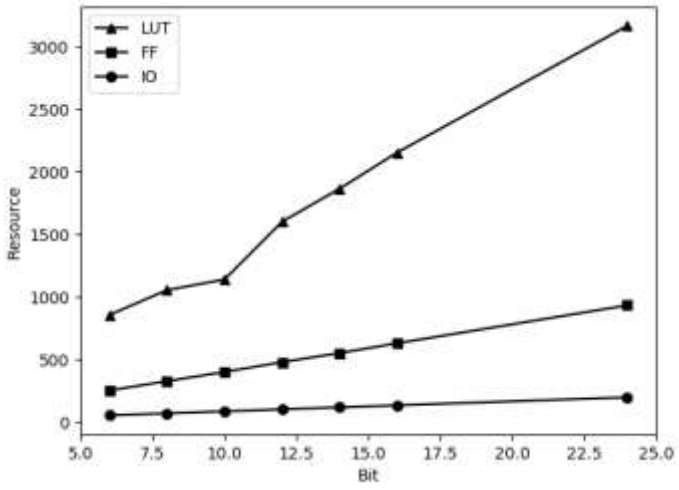

Fig. 7. The amount of resources spent

References

1. Ryabko B.Ya., Fionov A.I. Fundamentals of Modern Cryptography for Information Technology Professionals. - M.: Scientific world, 2014.- 173 p.
2. Akhmetov B.S., Korchenko A.G., Sidenko V.V., Drens YU.A., Seylova N.A. Prikladnaya kriptologiya: metody shifrovaniya. – Almaty: KazNITU im.K.I.Satpayeva, 2015. -496 s.:il.
3. Aytkhozhayeva Ye.ZH., Tynymbayev S.T. Aspekty apparatnogo privedeniya po modulyu v asimmetrichnoy kriptografii. Zhurnal Vestnik NAN RK.-№5(2014). Almaty, 2014. – s.88-93.
4. Orlov S.A., Tsil'ker B.YA. Organizatsiya EVM sistem: Uchebnik dlya vuzov, 3-izd.- .SPb.:Piter, 2015. -688 s.
5. Karatsuba A.A., Ofman YU.P. Umnozheniya mnogorazryadnykh chisel na avtomatakh. DANSSR. 1962. T.145.s.293-314.
6. Cook S.A., Aanderaa S.O. On the minimum computation time of functions. Trans. AMS, 142(1969), p.291-314.
7. Shenkhage A., Shtrassen V. Bystroye umnozheniye bol'shikh chisel. Kiberneticheskiy sbornik. 1973. Vyp 2. s.87-98.
8. Kovtun M., Kovtun V. Obzor i klassifikatsiya algoritmov deleniya i privedeniya po modulyu bol'shikh tselykh chisel dlya kriptograficheskikh prilozheniy [Elektronnyy resurs.] – http://docplayer.ru/30671408-Obzor-i-klassaifikaciya-algoritmov-privedeniys-po-modulyu-bolshih-chisel-dlya-kriptograficheskih-prilozheniy.html
9. Kombinatsionnyy rekurrentnyy formirovatel' ostatkov: pat. 2029435 oRos RF: MPK N03M7/18/ Petrenko V.I., Chipiga A.F.; zayavitel' i patentoobladatel' Petrenko V.I., Chipiga A.F. – №5032302/24; zaayavl.16.03.1992; opubl.20.02.1995, – 3s.
10. Ustroystvo dlya formirovaniya ostatka po proizvol'nomu modulyu ot chisla: pat. 236942 oRos RF: MPK N03M7/18, G06F 7/72 / Petrenko V.I., Sidorchuk A.V., Kuz'minov YU.V.; zayavitel' i patentoobladatel' GOU VPO Stavrolopol'skiy vo-yennyy institut svyazi RV.– №20101066858/08; zaayavl.10.01.2009; opubl.27.09.2009, Byul.№27– 9 s.

11. S.Tynymbayev, Y.Zh.Aitkhozhayeva, S.Adilbekkyzy. High speed device for modular reduction. Bulletin of National Academy of Sciences of the Republic of Kazakhstan. Volume 6, number 376 (2018)
12. Tynymbayev S.T., Aytkhozhayeva Ye.Zh. formirovatel' ostatka po proizvol'nomu modulyu. Patent RK №30983 ot 19.02.2016, opublikovan byulleten' №3 ot 16.03.2016, MPK G06F.
13. Tanymbayev S.T., Berdibayev R.Sh., Omar T, Shaykulova A.A., Magauin B. Bystrodeystvuyushchiye ustroystva privedeniya chisla po modulyu. Materialy XIV Mezhdunarodnoy Aziatskoy shkoly – seminara «Problemy optimizatsii slozhnym sistem». 20-31 iyulya 2018, chast' 2, Almaty 2018.
14. 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.
15. Montgomery P.L. Modular Multiplication without Trial Division // Math.Compulation. Vol. 44, No. 170. (Apr., 1985), pp.519-521. DOI: 10.20307/2007970.
16. Pisek E, Henige TM (2013) Method and apparatus for efficient modulo multiplication. Patent US No.8417756 B2.
17. S.Tynymbayev, R.Sh.Berdibayev, T.Omar, S.A.Gnatyuk, T.A. Namazbayev, S. Adilbekkyzy // Device for multiplying modulo numbers with analysis of the lower bits of the multiplier // Bulletin of National Academy of Sciences of the Republic of Kazakhstan. Volume 4, Number 380 (2019), 38-45.\
18. S. Gnatyuk, A. Okhrimenko, M. Kovtun, T. Gancarczyk, V. Karpinskyi, Method of Algorithm Building for Modular Reducing by Irreducible Polynomial, Proceedings of the 16th International Conference on Control, Automation and Systems, Oct. 16-19, Gyeongju, Korea, 2016, pp. 1476-1479.
19. Hu Z., Gnatyuk S., Kovtun M., Seilova N. Method of searching birationally equivalent Edwards curves over binary fields, Advances in Intelligent Systems and Computing, Vol. 754, pp. 309-319, 2019.
20. Fedushko, S., Ustyianovych, T., Gregus, M. (2020) Real-time high-load infrastructure transaction status output prediction using operational intelligence and big data technologies. Electronics (Switzerland), Volume 9, Issue 4, Article number 668. DOI: 10.3390/electronics9040668
21. S. Gnatyuk, V. Kinzeryavyy, M. Iavich, D. Prysiazhnyi, Kh. Yubuzova, High-Performance Reliable Block Encryption Algorithms Secured against Linear and Differential Cryptanalytic Attacks, CEUR Workshop Proceedings, Vol. 2104, pp. 657-668, 2018.
22. Iavich M., Gagnidze A., Iashvili G., Gnatyuk S., Vialkova V. Lattice based Merkle, CEUR Workshop Proceedings, Vol. 2470, pp. 13-16, 2019.