High-speed devices for modular reduction with minimal hardware costs

S. Tynymbayev¹, R. Berdibayev¹, T. Omar¹, Y. Aitkhozhayeva¹, A. Shaikulova¹ and S. Adilbekkyzy¹

Abstract: Asymmetric cryptosystems have an important advantage over symmetric systems, since only the public key is transmitted. However, asymmetric cryptographic algorithms have a lower speed compared to symmetric ones. When encrypting and decrypting in asymmetric cryptographic algorithms, complex and cumbersome procedures are used to raise very large numbers to a power modulo (modular exponentiation). In this case, the most resource-consuming operation is the modular reduction operation. One of the solutions to improve performance is the development of high-speed circuit solutions for modular reduction, the main task of which is to obtain the remainder of the division of a reducible number by the module. The structure of a high-speed former of partial remainders based on one binary adder and three comparison circuits is proposed, which can significantly decrease the hardware costs of devices for reducing numbers of multi bits in modulus. Based on the proposed former of partial remainders, a block diagram of a high-speed device for reducing the number modulo with sequential action was developed. Using this principle, a structural block diagram of a device of sequential action of a matrix type is developed. Based on the matrix circuit, a pipelined matrix circuit for reducing the number modulo is designed to process the data stream. A formula is given for estimating the gain in time when processing data streams. Algorithmic validation and verification of the high-speed devices for modular reduction with minimal hardware costs of sequential action was carried out on programmable logic-integrated circuits (FPGAs). For this, The Nexys 4 board based...
on the Artix-7 Field Programmable Gate Array (FPGA) from Xilinx was chosen. Verilog HDL is used to describe the circuit for reducing a number modulo. The results of a timing simulation of the device are presented in the form of time diagrams for a given 8-bit and 16-bit numbers, confirming the correct operation of the device.

**Subjects:** Cryptographic hardware and embedded system; Public-key cryptography; Information security; Computer science

**Keywords:** modular reduction; former of partial remainders; comparators; high-speed hardware implementation; FPGA

### 1. Introduction

The wide use of asymmetric cryptosystems with high security in comparison with symmetric cryptosystems is constrained by their low speed, as encryption and decryption procedures use complex and cumbersome mathematical calculations over very large numbers.

Hardware encryption has a number of significant advantages over software encryption, one of which is high-speed performance. The hardware implementation of cryptography algorithm guarantees its integrity, encryption and storage of keys is performed in the encoder board itself rather than in the computer’s RAM. Thus, the security of the implementation of the algorithm is ensured, which is also an important advantage. Therefore, when designing hardware and software-hardware cryptosystems with a public key, the task of developing circuit solutions for implementing one of the basic operations—modular reduction becomes relevant (Aitkhozhayeva, 2014; Al-Haija, Smadi, Al-Jafari, & Al-Shua'ibi, 2014; Ismail & Nuray, 2014).

There are many different methods of calculating the remainder when dividing by the module $P$ (Erdem & Serdar, 2018; Hars, 2004; Safiullah, Khalid, & Yasir, 2018; Tengfei Wang, Wei Guo, & Jizeng Wei, 2019; Yu, Bai, & Hao, 2015). In (Petrenko, Sidorchuk, & Kuz'minov, 2009), a device for reduction of a $2n$-bit number to an $n$-bit module in $n/2$ steps was considered. In it, to form the next partial remainder $r_i$, eight binary adders were required, which leads to an increase in the complexity of the device.

In (Tynymbayev, Gnatyuk, Aitkhozhayeva, Berdibayev, & Namazbayev, 2019; Tynymbayev, Shaikulova, Imanbaev, & Ziro, 2017), the devices for reducing the number modulo are considered, where the former of partial remainders are constructed on three binary adders. Since the hardware costs of the $n$-bit adder is more than three times that of the $n$-bit comparator (Harris & Harris, 2012). By replacing the two $n$-bit binary adders with three $n$-bit comparators, it is possible to significantly reduce the hardware costs for constructing the formers of partial remainders. Especially it is strongly felt when constructing a device for modular reduction on matrix or pipelined circuits.

### 2. Materials and methods

#### 2.1. Structure of the former of partial remainders (FPR)

Figure 1 shows a functional diagram of the FPR consisting of one binary adder ADD and three comparators Com1, Com2 and Com3.

The inputs of the FPR the tripled values of the module $3P$ and $3\overline{P}$ and the values of the module $P$ in the true representation and one’s complement—$P$ and $\overline{P}$ from the corresponding registers are received. The values $2P$ and $2\overline{P}$ are formed by shifting the values of $P$ and $\overline{P}$ to the left by one bit, respectively. Besides, the value of the previous remainder with a shift of two bits to the left $4r_{i-1}$ is fed to the input of the FPR. At the output of the adder, as a result of one of the three additions $4r_{i-1} + 3P + 1$, $4r_{i-1} + 2\overline{P} + 1$ or $4r_{i-1} + P + 1$, a partial remainder—$r_i$ is formed.
Multiplied by four previous partial remainder $4r_{i-1}$ is fed to the first inputs of the adder ADD and to the first inputs of the comparators Com1, Com2, Com3. The input Com1 is fed with the value of $P$ and its one's complement $\bar{P}$, which simplifies the structure of Com1. Similarly, $2P$ and $2\bar{P}$ are fed to the other inputs of Com2. The inputs of Com3 are given the values $3P$ and $3\bar{P}$.

In the comparator Com1, the codes $4r_{i-1}$ and $P$ are compared. If $4r_{i-1} \geq P$, then a “1” signal is generated at its output 2. Conversely, if $4r_{i-1} < P$, then at the output 1 the unit impulse is generated.

The Com2 compares the value $4r_{i-1}$ with doubled module $2P$. Then at the output 1 of this circuit, the signal “1” is set, if $4r_{i-1} < 2P$, while “0” is set at output 2. If $4r_{i-1} \geq 2P$, output 1 is set to “0” and at the output 2 is signal “1”.

The Com3 compares the codes $4r_{i-1}$ and $3P$. If $4r_{i-1} < 3P$, then at the output 1 of this circuit a “1” signal is formed and “0” is set at the output 2. When $4r_{i-1} \geq 3P$, at the output 1 is formed by the signal “0” and at the output 2 the signal “1” is set.

Table 1 shows the executable operations, depending on the ratios of $4r_{i-1}$ with different values of the modules $P$, $2P$ and $3P$.

With the ratios $P \leq 4r_{i-1} < 2P$, a unit impulse is generated at the output of the AND1 gate, which is simultaneously fed to the inputs of OR2 and AND3 gates, on the second inputs of which are fed with the one’s complement module bits $\bar{P}$. Output AND3 gates are fed to the right inputs of the adder ADD via the OR1 gates. On the left inputs ADD, the codes of the value $4r_{i-1}$ are fed, and

| Ratios          | Operations                  |
|-----------------|-----------------------------|
| $P \leq 4r_{i-1} < 2P$ | $r_{i} = 4r_{i-1} + P$ 1    |
| $2P \leq 4r_{i-1} < 3P$ | $r_{i} = 4r_{i-1} + 2P$ 1  |
| $3P \leq 4r_{i-1}$ | $r_{i} = 4r_{i-1} + 3P$ 1  |
| $4r_{i-1} < P$    | $r_{i} = 4r_{i-1}$         |
through OR2 the signal “+1” is fed to the input of the lowest order bit position of this adder, the operation \( r_i = 4r_{i-1} + P + 1 \) is performed.

For the ratios \( 4r_{i-1} \geq 2P \) and \( 4r_{i-1} < 3P \), the outputs of the AND2 gate unit impulse is generated, which is fed to the input of the OR2 gate and the block of the AND4 gates. At the second data inputs of AND4 are fed the one’s complement doubled module bits. The value of the module \( 2P \) through the block of the OR1 gates is fed to the right inputs of the adder ADD, and the code “+1” is supplied to the input of the lowest order bit position and the operation \( r_i = 4r_{i-1} + 2P + 1 \) is performed in the adder.

With the ratios \( 4r_{i-1} \geq 3P \) from second output of the comparator Com3, a unit impulse is applied to the input of the circuit of the AND5. At the data inputs of AND5 gates are fed with bits of the module \( 3P \). Codes \( 3P \) through the block of OR1 gates are transmitted to the right inputs of the ADD. In this case, the operation \( 4r_{i-1} + 3P + 1 \) is performed in the adder.

In sequential action devices for modular reduction under \( 4r_{i-1} < P \) condition previous remainder is stored in the remainder register.

In matrix circuits at \( 4r_{i-1} < P \), the value \( 4r_{i-1} \) through the AND0 gate and the block of the logic gates OR3 is transmitted to the next FPR.

### 2.2. Structure of devices of sequential action

Figure 2 shows the functional diagram of the device for modular reduction a sequential action, which consists:

- \( 2n+2 \)-bits register, where the number \( A \) is shifted by two bit positions to the left—RgA;
- \( m \)-bit registers Rg3P and RgP where values of the tripled module—\( 3P \) and the module \( P \) are taken, respectively, before the operations begin;
- former of partial remainders FPR;
- control block, which includes a subtracting counter.

The highest order bit positions of the register RgA through the block of the AND9 gates are connected with the FPR. Through the AND9 gates, under the control of the clock pulse CP from the controller, a value of \(4ri-1\) is transmitted from the register RgA. The inputs of the FPR are given the values \(3P, 3P, 2P, 2P\) and \(P, P\). From the output of the FPR via the next partial remainder is fed to the inputs RgA. By using the “End of Operation” signal the result of the calculation is output through the block of the AND10 gates. The inputs of the controller are fed with the signal “Start”, the clock pulses CP and the number of shifts \(n/2\), necessary for calculating \(R = A \mod P\).

The device works as follows. With the signal “Start”, the operands \(A, 3P\) and \(P\) are, respectively, received in the registers RgA, Rg3P and RgP. In addition, by the “Start” signal, the number of shifts \(n/2\) is fed in the counter of the clock pulses CCP. At each step of modular reduction, after receiving the operands, the controller sends its clock pulse to its output, which shifts the contents of RgA two bits to the left. After shifting, through the delayed CP on the delay lines DL, the value of the highest bits RgA, where the value \(4R_0\) is generated over the block of AND9 gates, is transferred to the inputs of the adder of the FPR, at the output of which the value of the partial remainder \(r_i\) is formed. This remainder via the OR4 gates will be written in RgA. With the clock pulse CP the counter reading is reduced by one. By this time, the next CP is fed into the circuit, which generates the next partial remainder in the FPR that is sent to RgA, and so on.

After the \(n/2^n\)th clock pulse is applied, the \(n/2^n\)th partial remainder is generated at the output of the FPR, which is stored in RgA. With this clock pulse, the CCP is set to zero and produces the signal “End of Operation”. With this signal, the result from the highest order bit positions of RgA is output by the AND5 gates.

### 2.3. Matrix and pipelined circuits for modular reduction

Now consider the matrix circuit for reducing the number modulo, the functional diagram of which is shown in Figure 3

The circuit is constructed for the number \(A = a_{11}a_{10} \ldots a_1a_0\) and \(P = a_7a_6 \ldots a_1a_0\) and consists of the registers RgA and Rg3P, RgP and the formers of partial remainders FPR1, FPR2 and FPR3. At the input of these formers, the values \(3P, 3P, 2P, 2P\) and \(P, P\) are applied.

At input FPR1 a value \(r_1\) is formed that with a shift by two bit positions to the left \(L(2) 4r_1\), is fed to the inputs of the FPR2. At the same time, the bits \(a_3\) and \(a_2\) of the number \(A\) are attached.

At the output of the FPR2, the value of the partial remainder \(r_2\) is formed. The inputs of the FPR3 are fed to the partial remainder \(r_2\) that shifted by two bit positions to the left \(4r_2\) and docked to the bits \(a_1\) and \(a_0\). At the output of the FPR3, the remainder \(R = r_3\) is formed. Delay time DL is determined by \(|DL = 3|\) FPR.

When pipelining, the whole process is divided into a sequence of completed steps. Each of the stages of the division procedure is performed at its stage pipeline, with all stages running in parallel. The results calculated at the \(i\)-th stage are transferred for further processing to the \((i + 1)\)-stage. The transfer of information from the stage to the stage occurs through the buffer register placed between them. The one that performs its operation puts the result in the buffer register and can start processing the next portion of these operations, while the next stage of the pipeline uses the data stored in the buffer registers located at its inputs as initial ones. Synchronization of the pipeline work is provided by clock pulses (CP), the period of which \(\tau\) is determined by the slowest stage of the pipeline \(\tau_i\) and the delay in the buffer register.
In the pipelined device of modular reduction with K stages, the input data that multiplied modulo can be fed to the input with an interval of K times smaller than in the case of the usual reduction of the numbers modulo. With the same frequency, the result appears at the output of the device.

For constructing such a pipelined device is required to have registers of reducible number, the registers of the partial remainders, the group of AND gates, formers of partial remainders, modulo-2 adders and result register.

Figure 4 shows the pipelined matrix circuit for reducing the number modulo, constructed on the basis of the matrix scheme (Figure 3). The pipeline consists of three stages. Each stage consists of a FPR and buffer registers of the partial remainder and bits of the number A that have not yet entered the operation and the registers Rg3P and RgP. The pipeline is controlled by the clock pulses CP. After each CP is applied to the input and pipeline is filled, the results of the pairs Ai and Pi are formed.
As you can see from Figure 4, the pipeline is synchronous linear. The performance of a synchronous pipeline basically depends on the correct choice of the duration of the clock period $T_p$. The minimum allowable value of $T_p$ can be defined as the sum of the largest of the processing time on a separate stage of the pipeline $T_{\text{max}}$ and the time of writing the results of the calculation in the buffer registers of the stage $T_{\text{Rg}}$, then

$$T_p = T_{\text{max}} + T_{\text{Rg}}$$

(1)

In the considered pipeline, $T_{\text{max}}$ is determined by processing time on the former of the partial remainders. It is determined by the time modulo-2 addition $T_{m2}$ and the switching time of the doubled partial remainder, or the result of adding the doubled remainder with the module $P$ to the outputs of the multiplexer, i.e. time delay on the multiplexer—$T_{m}$ and the comparison time $4 r_{i-1} c P$, $2 P$ and $3 P$—$T_{\text{com}}$. Then

$$T_{\text{max}} = T_{m2} + T_{m} + T_{\text{Rg}} + T_{\text{com}}$$

(2)

The processing time of the $N$ input data stream on a pipeline with $K$ stages with a clock period $T_p$ can be determined by the formula (Orlov & Tsilker, 2010).
Example. Let \( N = 20 \) and \( K = 3 \), then \( C = (20*3)-22 = 28T_p \)

From the considered example, it can be seen that the use of the pipeline will reduce the processing time to 28 units.

\[
C = |NK - (K + N - 1)|T_p
\] (3)

3. Implementation on FPGA

Algorithmic validation of the high-speed devices for modular reduction with minimal hardware costs of sequential action was carried out. For this, The Nexys 4 board based on the Artix-7 Field Programmable Gate Array (FPGA) from Xilinx was selected (Figure 5). To describe the circuit for modular reduction, the hardware description language Verilog was chosen (Digilent Nexys 4 Artix-7 FPGA Trainer Board, 2018; IEEE Standard for Verilog Hardware Description Language, 2018; Navabi, 2007).

Table 2 shows the number of main resources of FPGA Artix-7 (XC7A100T-1CSG324C):

To input the data and visualize the intermediate results, the FPGA Board is equipped with all necessary ports and peripherals, the main of which are 16 switches, 16 LEDs, as well as a USB-UART bridge, DDR2 128MB, etc.

Figure 6 shows the timing diagram for the formation of the values of partial remainders with the reduction of the number \( A_{a7\ldots a0} = 187_{10} = 1011011_{2} \), \( P = 14_{10} = 1110_{2} \), \( 2P = 28_{10} = 11100_{2} \), \( 3P = 42_{10} = 101010_{2} \) and the highest bits \( A_{a7\ldots a0} \) represent \( r_0 = 11_{10} = 1011_{2} \).

In Figure 6, on the rising edge of the clock pulse CP1, the contents of register RgA is shifted by two bit positions to the left and in this register on the highest six bits positions \( 4r_0 + a_3a_2 \) is formed, which corresponds to the binary code 1,011,102 = 4610. At the outputs of AND9 gates (Figure 2), the number 46 is compared with the numbers \( P = 14_{10} \), \( 2P = 28_{10} \), \( 3P = 42_{10} \) and signal “1” is generated at the

Figure 5. Nexys 4 FPGA board.
output 2 of Com3, which leads to perform operations in the adder $46 - 3P = 46 - 42 = 4 = r_1$ and the binary number $410 = 1002$ is transmitted to the highest bit positions of the register RgA.

On the rising edge of the clock pulse CP2 is shifted by two bit positions to the left register RgA and in it is established $4r_1 + a_10_0 = 16 + 3 = 19_{10}$. The number 19 is compared to P, 2P and 3P and signal “1” is generated at the output 2 of the Com1, which leads to perform operations $19 - P = 19 - 14 = 5$, which can be seen in Figure 6, i.e. $R = 5$ is the result of the operation $187 \mod 14 = 5$.

Figure 7 shows the timing diagram for the formation of partial remainders for the numbers $A_{15 - 00} = 27317_{10} uP = 209_{10}; 2P = 418_{10} u3P = 627_{10}$.

$A_{15 - 00} = 0110101010110001_2$ and $P = 11010001_2$, it is easy to determine from this that $r_0 = 0110101010 = 106_{10}$ from the diagram, we can see $r_1 = 8, r_2 = 0$ and $r_3 = 0, r_4 = 147$.

When calculating $r_2 = r_3$ at the outputs of the adder ADD, we get negative differences, which were blocked, keeping the “old” remainders in RgA.

4. Conclusion
In devices for modular reduction in former of partial remainders, replacing two binary adders with three comparators, which results in minimizing the structure of a high-speed device for modular reduction. The presented device allows to accelerate the calculation by reducing the $2n$-bit number $A$ modulo $P$ by two times. Not all calculations go beyond the bit grid of the module.

When processing a large amount of data on the same algorithm, the most productive are the pipeline structures. When encrypting data, the modular reduction operation is performed for a large
amount of different numbers. Therefore, to increase the speed, it is advisable to use pipeline structures. On the base of the modifications, it is possible to implement the pipelined matrix circuit for reducing the number modulo on the FPGA.

Funding
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 info-communication systems and general purpose networks”).

Author details
S. Tynymbayev
E-mail: tynym@mail.ru
ORCID ID: http://orcid.org/0000-0001-9634-143X
R. Berdibayev
E-mail: r.berdibayev@aes.kz
ORCID ID: http://orcid.org/0000-0002-8341-9645
T. Omar
E-mail: at.ujan@mail.ru
ORCID ID: http://orcid.org/0000-0002-5961-8556
A. Shaikulova
E-mail: shaikulova_ok_ali@mail.ru
ORCID ID: http://orcid.org/0000-0001-9634-143X
S. Adilbekkyzy
E-mail: saraon.02.95@mail.ru
ORCID ID: http://orcid.org/0000-0002-3929-7070
1 Almaty University of Power Engineering and Telecommunication, 126/1 Baitursyuly Street, Almaty 050013, Kazakhstan.

Citation information
Cite this article as: High-speed devices for modular reduction with minimal hardware costs, S. Tynymbayev, R. Berdibayev, T. Omar, Y. Aitkhozhayeva, A. Shaikulova & S. Adilbekkyzy, Cogent Engineering (2019), 6: 1697555.

References
Aitkhozhayeva, E. (2014). Aspects of hardware reduction modulo in asymmetric cryptography. Bulletin of National Academy of Sciences of the Republic of Kazakhstan, 5, 88–93. (In Russian).

Al-Hojjo, Q. A., Smadi, M. A., Al-Jaf’i, M., & Al-Sharif, A. (2014). Efficient FPGA implementation of RSA coprocessor using scalable modules. Procedia Computer Science, 34, 647–654. doi:10.1016/j.procs.2014.07.092

Digilent Nexys 4 Artix-7 FPGA Trainer Board. (2018). https://www.xilinx.com/products/boards-and-kits/1-3ymp5.html

Erdem, Ö., & Serdar, S. E. (2018). A fast digit based Montgomery multiplier designed for FPGAs with DSP resources. Microprocessors and Microsystems, 62, 12–19. ISSN 0141-9331. doi:10.1016/j.micpro.2018.06.015.

Harris, D. M., & Harris, S. L. (2012). Digital design and computer architecture (2nd ed.). San Francisco, CA: Morgan Kaufmann Publishers Inc.

Hars, L. (2001). Long modular multiplication for cryptographic applications. In M. Joye & J. J. Quisquater (Eds.), CHES 2004. LNCS (Vol. 3156, pp. 45–61). Berlin, Heidelberg: Springer. doi:10.1007/3-540-28632-5_4

IEEE Standard for Verilog Hardware Description Language. (2018). IEEE Standard 1364–2005 (Revision of IEEE Standard 1364–2001). doi:10.1109/IEEESTD.2006.99495.

Ismail, S., & Nuray, A. (2014). Improving the computational efficiency of modular operations for embedded systems. Journal of Systems Architecture, 60(5), 440–451. doi:10.1016/j.sysarc.2013.10.013

Navabi, Z. (2007). Embedded core design with FPGAs. US: McGraw-Hill Education.

Petrenko, V. I., Sidorchuk, A. V., & Kuz’mov, J. V. (2009). Device for generating remainder from arbitrary modulus. Russian Federation Patent 2368942. (In Russian) http://www.freepatent.ru/patents/2368942

Safiullah, K., Khalid, J., & Yasir, A. (2018). Highly-parallel hardware implementation of full-word Montgomery multiplier for ECC applications. Microprocessors and Microsystems, 62, 91–101. ISSN 0141–9331. doi:10.1016/j.micpro.2018.07.005.

Tengfei Wang, A., Wei Guo, B., & Jizeng Wei, C. (2019). Efficient modular reduction based on the divider by blocking negative remainders. NEWS of National Academy of Sciences of the Republic of Kazakhstan., 2, 238–248. doi:10.32014/2019.2518-170X.60

Tynymbayev, S., Gnatyuk, S., Aitkhozhayeva, Y., Berdibayev, R., & Namazbayev, T. (2019). Modular reduction based on the divider by blocking negative remainders. NEWS of National Academy of Sciences of the Republic of Kazakhstan., 2, 238–248. doi:10.32014/2019.2518-170X.60

Tynymbayev, S. T., Shaikulova, A. A., Imamboev, A. Z., & Ziro, A. A. (2017). Matrix schemes for reducing the number modulo. Vestnik KazNRTU special edition. Satbayev University: Almaty, 534–539. (In Russian).

Yu, H., Bai, G., & Hao, H. (2018). Efficient modular reduction algorithm without correction phase. In J. Wang & C. Yap (Eds.), FAW 2015. LNCS, 9130 (pp. 304–313). Cham: Springer. doi:10.1007/978-3-319-19647-3_28
