Development of a device for multiplying numbers by means of FPGA

N M Berezin, I E Chernetskaya, V S Panishchev and A M Shabarov
Department of Computer Engineering, South-West State University, 94, 50 let oktyabrya., Kursk, 305040, Russia
E-mail: nixberezin@yandex.ru, white731@yandex.ru, gskunk@yandex.ru

Abstract. The authors propose the description of the development of a device for multiplying numbers. The device for multiplying numbers on the field-programmable gate array (FPGA) includes two input and one output registers, fifty-six single-digit adders, sixty four logic elements AND, one exclusive OR gate. The main scientific and technical task in developing a device for multiplying numbers is to reduce hardware complexity using single-bit adders and logic elements. Introduction includes description of works of scientists and researchers whose publications are devoted to design and development of multiplier construction methods, multiplier FIR performance improvement by right-shift and addition method on FPGA (field-programmable gate array) basis. The implementation of MAC-block, hardware implementation of binary multiplier on the basis of multi operand adder, multiplier design by right-sliding and addition with control automaton in the FPGA basis is the actual research tasks presented in a number of papers. The description of features of multiplier implementation, high-speed multipliers with variable bit rate, studies of approaches for designing modular multipliers, FPGA image processing using Brown multiplier for performing convolution operation find application in problems of performance and speed. Also, a number of authors describe implementation of conveyorization method, design of dual multiplier, construction method of 8-bit multiplier with reduced delay, 8-bit high-density systolic multiplier arrays on FPGA and development of high-performance 8-bit multiplier using McCMOS technology. A fragment of a developed device for multiplying numbers is presented in the work by the authors. The principle of operation of a device for multiplication is described. The description of connected elements of the device is given. The timing diagrams of operation of a device for multiplication of numbers are presented.

1. Introduction
In the information age FPGA devices are actively used in research related to the development, testing, implementation and design of electronic tools. The development of new devices allows manufacturers and researchers to solve the problems of performance and speed of systems. The field of use of devices on the FPGA is significant, and includes tasks starting from the implementation of research work(R&D), research projects (RP) to the implementation of technical devices in the development of computer vision systems(CVC), digital and preprocessing images. It is also worth noting that the performance of a number of tasks related to the performance of calculations with digit capacity is now also a relevant research problem. Therefore, it is necessary to use multipliers to solve technical scientific and technical problems - devices for multiplying numbers based on FPGA.
Multiplier devices are actively used in signal processing and image processing tasks to maximize running speed to perform the operation in the device. The best known method of improving running speed is to implement finite impulse response filters (FIR-filters) in the basis of FPGA - multiplication by the method of right shift and addition. In this field multiplication of the number by the constant, which is the most common operation in digital signal processing (DSP), is widespread.

A number of researchers and developers are engaged in the design and development of methods for building multipliers. In the research [1], the author proposes a high-performance vector multi-format multiplier, based on complex functional (CF) unit (VMU) capable of executing a wide set of commands based on multiplication over different input types. Such a complex functional unit (CF) has a smaller area compared to non-universal devices of similar performance, capable of providing high computing performance. In the research [2], researchers consider designing a FIR filter on a multiplier using the method of right shift and addition on the basis of FPGA. The authors consider software multipliers. To improve the performance of the FIR filter on the ALTMEMMUL megafunction, we can use the option of loading coefficients from the FPGA block memory. In the work [3], the authors suggest an approach to improve multiplication speed in the Galois field, based on the application of pipelining method. This approach to improve multiplication performance can be used in the design of high-speed codecs. In [4], the authors consider a hardware implementation of a binary multiplier based on a multi operand adder. The method of summing shifted partial products is used for implementation. With the increase in the capacity parameter the estimates were made for hardware costs and speed. In research [5], the authors designed a multiplier by the method of right shift and addition with a control machine in the basis of the FPGA. The authors developed a MAC block for multiplying two 4-digit unsigned numbers using the multiplication and accumulation method. The implementation of the multiplier presented by the authors can be used in the design of FIR filters. In the article [6], the researcher describes the features of the implementation of multiplication by FPGA, thereby indicating that when increasing the multiplier bit, the use of the fast multiplication method allows you to gain both the required hardware resources of FPGA and calculation speed. In the research [3], the author describes high-speed multipliers with variable discharge on the FPGA. The author of the publication considers multiplication over data with variable discharge, a method of sequential formation of partial products, a vector and matrix scheme of a multiplier. Finally, the researcher concludes that the most productive is the matrix control scheme, which has a high speed of operations. In the paper [8], the authors investigate various approaches to designing modular multipliers that require maximum performance for use in digital signal processing tasks. The authors conclude that the best method in the general case for constructing modular multipliers for very large scale integration (VLSI) is the index method, and for programmable logic integrated circuits (FPGA) is the square difference method. Work [9] is devoted to image processing using an embedded system using a digital filter on a FPGA. To perform the convolution operation, the authors of the study use the Brown multiplier. Therefore, the development of multipliers for use in image processing tasks is a relevant task. Research [10] contains two implementations of multipliers with a pipeline method to reduce the critical path, as well as increase the quick-operating speed. The study [11] presents a multifunctional design of a binary precision floating point multiplier. This research topic is currently relevant in calculating the multiplication of complex numbers in digital signal processing problems. Researchers in [12] propose a method of constructing an 8-bit multiplier with a reduced delay, that is, to increase the speed of the multiplication operation. The proposed version is intended both for implementation on FPGA and for implementation in ASIC. The research [13] studies 8-bit systolic arrays of high-density multipliers on the FPGA. The authors developed a method for creating large, scalable and high-performance systolic arrays. The method is also relevant for the development and design of electronic tools. The research [14] is devoted to the development of a high-performance 8-bit multiplier using McCMOS technology. The technology is relevant for the development and design of multipliers on the FPGA.
2. Developing a number multiplier

The developed device for multiplying numbers applies to computer engineering and can be used for hardware implementation of the 8-bit multiplier. The number multiplier consists of two input and one output registers, 56 single-bit adders and 64 AND gates, one exclusive OR.

The input numbers are an 8 digit sequence. The main scientific and technical task in developing a device for multiplying numbers is to reduce hardware complexity using single-bit adders and logic elements.

Figure 1 shows the development fragment of the number multiplier.
The principle of the device is as follows: input numbers are a digit sequence, where \( A = A (7; 0) \), and \( B = B (7; 0) \). Data is supplied to the input of sixty-four AND elements.

At the inputs of the device there are delay elements - registers. The data is clocked with one signal (clk). By the signal (clk), the registers store the input data - D1 and D2. After storing the input data, signals A and B appear at the output. After a certain time, the calculation will end. Thereafter, an output signal is generated by the timing signal clk.

As it is shown in Figure 1, the lower digits A (0) and B (0) are supplied to the inputs of the logical element AND1. Thus, the lower digit of the result \( ZZ (0; 0) \) is formed. The lower digit of the multiplier B (0) and the digit of the multiplied A (1) are supplied to the input of the logical element AND2. The output of the logic element I2 enters the input F1 of the adder S1, forming the output \( ZZ (0; 1) \).

At the output S of adder S56 the result of addition of data supplied to inputs F1 and F2 - Sum (7; 7) is generated. Sum (7; 7) addition result comes to output and generates P (14) digit.

The result of addition of transfers to the most significant bit PO_7 is supplied to the input of D15 register Reg3 and generates P (15) digit.

The high-order digit P (16) is the result of the XOR operation on the high-order digits of the input data, which stores the sign of the number - negative or positive.

3. Analysis of timing diagrams

Let us analyze the timing diagrams. Figure 2 is a timing diagram of the multiplier. Figure 3 is a resulting timing diagram of the multiplier.

```
| clk | D1(7:0) | D1_1 | D1_2 | D1_3 |
|-----|---------|------|------|------|
|     | D2(7:0) | D2_1 | D2_2 | D2_3 |
|     | A(7:0)  | A_1  | A_2  | A_3  |
|     | B(7:0)  | B_1  | B_2  | B_3  |
|     | P(15:0) | P_1  | P_2  | P_3  |
```

**Figure 2.** Multiplication device timing diagram

According to the present timing diagram of the multiplier (Figure 2), D1 (7:0) and D2 (7:0) are input data supplied to the calculation circuit.

D1_1, D1_2, D1_3 are three input data pairs. A (7:0) and B (7:0) – are outputs of two registers. That is, at the edge of the signal clk, the register stores data values, and after a certain amount of time, input data values are generated.
Figure 3. The resulting timing diagram of the multiplier

As it is shown in Figure 3, the data is input to the calculation circuit. During the action of the clk signal, a calculation must occur. Accordingly, upon arrival of the front, the result of calculating A_1; B_1 will result at the output of the circuit P_1.

Figure 4 is a time chart of the operation of the device.

Reliability of device multiplication operations is confirmed by the presented time chart. The device was simulated in XILINX ISE Design Suite 14.7. The operating time of the device at the frequency 19.893MHz was 50.268ns.

According to the table, these elements are found in our used chip. A computational project is built from the elements. It should be noted that when creating a project, developers very often rest on the so-called Slice, that is, in the part of a configurable FPGA unit containing several LUTs and Flip-Flop - there are register restrictions. Large projects require more computing resources. If multipliers with a lot of parallel calculations are created, then the built-in multipliers Number of DSP48A1s are used. None of it is used in our project. Built-in elements (LUT, slice, etc.) are the basic elements on which devices are built. More powerful chips contain global synchronization lines, memory, to improve performance. The selection of chips is not determined by the number of LUT registers on which all combinatorial logic can be created, namely, memory blocks, the number of multipliers, the number of synchronization blocks. In high-speed systems, the number of synchronization lines is a crucial point. Since the chip contains a certain number, i.e. global synchronization lines, to synchronize the entire chip. Therefore, when creating complex projects with calculations, it is always vital to be able to use not just the schema description language, but to understand which elements it is better to use when creating a schema.

The overview of the resources of the chip is given in Table1.
Table 1. The overview of the resources of a chip

| №  | Slice Logic Utilization                        | Used     | Available | Utilization |
|----|-----------------------------------------------|----------|-----------|-------------|
| 1  | Number of Slice Registers                     | 32       | 18,224    | 1%          |
| 2  | Number of Slice LUTs                          | 225      | 9,112     | 2%          |
| 3  | Number used as logic                          | 224      | 9,112     | 2%          |
| 4  | Number using O6 output only                   | 224      |           |             |
| 5  | Number used as Memory                         | 0        | 2,176     | 0%          |
| 6  | Number used exclusively as route-thrus        | 1        |           |             |
| 7  | Number with same-slice register load          | 1        |           |             |
| 8  | Number of occupied Slices                    | 71       | 2,278     | 3%          |
| 9  | Number of MUXCY's used                       | 0        | 4,556     | 0%          |
| 10 | Number of LUT Flip Flop pairs used            | 225      |           |             |
| 11 | Number with an unused Flip Flop              | 197      | 225       | 87%         |
| 12 | Number with an unused LUT                    | 0        | 225       | 0%          |
| 13 | Number of fully used LUT-FF pairs            | 28       | 225       | 12%         |
| 14 | Number of unique control sets                | 1        |           |             |
| 15 | Number of bonded IOBs                        | 34       | 232       | 14%         |
| 16 | Number of BUFG/BUFGMUXs                      | 1        | 16        | 6%          |
| 17 | Number used as BUFGs                         | 1        |           |             |
| 18 | Average Fanout of Non-Clock Nets              | 2.36     |           |             |

4. Conclusion
Thus, the development of a device for multiplying numbers based on FPGA was presented in the work, which included two input and one output registers, fifty-six single-digit adders, sixty-four AND gates, one exclusive OR gate. This development can be used in pipeline circuits. Thus, the use of programmable logic integrated circuits in computational tasks is a highly popular direction in the field of electronics. The authors presented a fragment of a developed device for multiplying numbers. The principle of operation of device for multiplication was described. The description of connected elements of the device was given. The timing diagrams of operation of device for multiplication of numbers were presented. The reliability of device multiplication operations was confirmed by the presented timing diagram. The number multiplier was modeled in XILINX ISE Design Suite 14.7. The operating time of the multiplier at frequency 19.893MHz was 50.268ns.

Acknowledgments
The author is grateful to the scientific advisor Doctor of Technical Sciences, Professor I.E. Chernetskaya, PhD of Sciences V.S. Panishchev, head of the department of Computer Engineering of South West State University V.S. Titov for assistance in this research.

References
[1] Belyaev I A 2019 Problems of radio electronics 8 31-36
[2] Strogonov A M, Bystritsky A V and and Jyoti A 2014 Components and technologies 3(152) 63-68
[3] Amelyutin E V and Somov A M 2016 Proceedings of the Radio Research Institute 1 37-40
[4] Yakunin A N and Aung M S 2017 Actual problems of informatization in science and education 126-131
[5] Strogonov A M, Bystritsky A V 2013 *Components and technologies* **12(149)** 52-56

[6] MO Ch Ch 2008 *Modern high technologies* **4** 100-102.

[7] Listopad E V 2018 *Reports of the Belarusian State University of Informatics and Radioelectronics* **3(113)** 5-12.

[8] Telpukhov D V, Soloviev R A, Balaka E S, Rukhlov V S and Mikhmel A S 2016 *Problems of the development of promising micro- and nanoelectronic systems (MES)* **1** 249-254

[9] Egunov V A, Konchenkov V I and A. V. Starostina 2016 Bulletin of the Volgograd State Technical University **11(190)** 37-44

[10] Rashidi B, Rezaeian Farashahi R and Sayedi S M 2015 *The ISC Int’l Journal of Information Security* **2** 101-114

[11] Liu D, Wang M, Wang Y and Su H A 2015 *IEEE 9th International Conference on Anti-Counterfeiting, Security, and Identification (ASID)* 56–60

[12] Dhanabalan and Tamil S 2013 *International Journal of Computer and Communication Engineering* **6** 665-668

[13] Langhammer M, Gribok S and Baeckler G 2020 *IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)* 84-92

[14] Kayal D, Mostafa P, Dandapat A and Sarkar C K 2013 *Journal of Signal Processing Systems* **1** 1-9