ISSN 2224-087X. Електроніка та інформаційні технології. 2024. Випуск 26. С. 3–15
Electronics and information technologies. 2024. Issue 26. P. 3–15

UDC 004.2 DOI: https://doi.org/10.30970/eli.26.1

DEVELOPMENT OF A MATHEMATICAL SPECIAL COMPUTER BASED ON THE ALTERA CYCLONE 3 FPGA USING THE NIOS2 CORE

R. Diachok[1], V. Lysiak[1], H. Klym[1,2]

[1]*Specialized Computer Systems Department,*
Lviv Polytechnic National University,
12 Bandery St., UA-79013, Lviv, Ukraine
roman.v.diachok@lpnu.ua, vitalii.Lysiak.KI.2020@lpnu.ua, halyna.i.klym@lpnu.ua

[2]*Radielectronic and Computer Systems Department,*
Ivan Franko National University of Lviv,
50 Drahomanova St., UA–79005 Lviv, Ukraine
halyna.klym@lnu.edu.ua

This article describes the research and analysis of the development process of a specialized mathematical computer based on the Altera Cyclone 3 FPGA using the NIOS2 core. The main goal is to study the technical aspects of the project, such as hardware architecture and algorithmic support. Particular attention is paid to the analysis of algorithms and the structural scheme of the computer to maximize its performance and use of FPGA resources. The key stages of the system development have been elaborated and described in detail. Among these stages are the development of the concept of the special computer architecture, the development of the software algorithm, and the development of the interaction of the created special computer with the user. Additionally, modeling of the calculation of basic mathematical operations on the NIOS II core was performed to confirm the concept and workability of the created device. The results of this study are intended to contribute to the further development of innovative technologies in the field of computing.

Key words: FPGA, NIOS2, Cyclone 3, Altera.

Introduction. Today full of various electronic devices, the issue of processing high-frequency signals, as well as the problem of quickly calculating complex mathematical expressions/formulas, is particularly acute. This is due to the fact that each new device has more and more analog/digital sensors, the accuracy of which is improving every time, and their management is becoming more and more difficult. The data received from such sensors usually needs to be processed very quickly and converted into a format that can be further processed by the rest of the system. This processing can include signal analysis and mathematical calculations. High-speed mathematical calculations can also be useful not only in analyzing sensor data, but also in calculating complex formulas that can be used, for example, in simulating the operation of electronic components, building virtual 3D models (and other applications).

The classic way to calculate mathematical expressions is to use a central processor and its corresponding instructions. This method is quite simple and does not require additional

electronic chips, but it also has a number of disadvantages. The first of them is low performance. Everything will depend on the number of processor cores/threads. Another important factor will be the presence of specialized instructions for processing certain expressions, plus the presence and set of instructions of the mathematical coprocessor. And if you need to calculate some complex non-standard expression/formula for which the processor does not have the appropriate instructions, then software emulation will be involved and performance will drop even more. [1]

There is another approach. All complex calculations can be transferred to an FPGA chip. This will allow you to make a specialized calculator that will calculate the value of the expression. This method has many advantages - there is no binding to the bit depth of standard data types - the input data buses can be made of the bit depth that is required. You can implement a calculator for absolutely any expression. But there is also a significant disadvantage: there are far fewer specialists on the market who can correctly create a calculator in the hardware description language than there are "programmers". Finding such a specialist in a company can take several weeks or even months. [2]

This article is aimed at researching and analyzing the development process of a mathematical specialized computer based on the Altera Cyclone 3 FPGA using the NIOS2 core. The main goal is to study the technical aspects of the project, such as hardware architecture, algorithmic support, and performance optimization.

Related work. Before developing their own computer, the authors familiarized themselves with existing solutions. They did not consider computers based on any microcontrollers and focused their attention on solutions specifically on FPGAs.

Source [8] illustrates a model of a calculator written in Verilog. This 8-bit calculator can add, subtract, multiply, divide, raise a number to the second and third powers, and calculate the square root and factorial. The simulation was performed on a CycloneV FPGA manufactured by Altera (now a division of Intel). The code was created and simulated in the Quartus II environment.

Source [9] presents a model of an FPGA-based square root calculator that uses very few resources and occupies a very small FPGA area. The model is designed to meet the needs of medium- and low-speed applications that do not require very high processing speeds, while optimizing the amount of resources used. A modified recoveryless algorithm is used to calculate the square root. The development was written in RTL VHDL and implemented on an Altera DE2 board for hardware validation. The implementation provided a very accurate square root calculation with low computation latency and low memory consumption for the input data of different widths tested.

This source [10] describes development of a complex module of an exponential function (ex) calculator. The developed module works with the 64-bit floating point format IEEE 754-1985. The module was coded in VHDL and synthesized for the Virtex-6 XC6VLX75T FPGA chip using the ISE 14.7 EDA tool from Xilinx. The results obtained by the module were compared to actual results and the sensitivity of the module was determined. The ability of this function to provide accurate results is crucial for the successful operation of function-dependent designs. For this reason, various methods are described in the literature for the logical calculation of the ex.

FPGA and Terasic DE0 board overview. FPGA chips (Field-Programmable Gate Arrays) are integrated circuits that have a large number of logic elements and are configured

after manufacturing. The main difference between FPGAs and conventional programmable logic integrated circuits (PLDs) is the ability to change the functionality and logic of the circuit after it is manufactured.

FPGAs offer several key advantages. First and foremost is their flexibility. These chips can be reprogrammed to execute a wide array of tasks, making them incredibly versatile across various applications. Additionally, FPGAs boast high performance capabilities. By executing numerous operations simultaneously, they can achieve remarkable processing speeds. Moreover, the manufacturing process for FPGA-based hardware is typically quicker than that for specialized integrated circuits, resulting in reduced development time. Many FPGAs also come equipped with built-in components such as memory blocks, multiplexers, and arithmetic logic units (ALUs), streamlining the development of intricate systems. Furthermore, utilizing FPGAs can lead to cost savings compared to the development and production of specialized integrated circuits [4].

However, FPGAs are not without their drawbacks. One significant disadvantage is their high resource consumption. Implementing certain functions on FPGAs may demand substantial resources, limiting scalability for some projects. Additionally, FPGAs tend to consume more energy than specialized integrated circuits when performing equivalent tasks. Another issue is the time delay associated with reprogramming FPGAs. The extensive configuration steps required can introduce operational delays, impacting overall performance [5].

Among the leading FPGA manufacturers are Xilinx Inc., renowned for their world-class products lauded for both their exceptional performance and extensive functionality, and Intel Corporation (formerly Altera), which, following its acquisition by Intel, continues to be a prominent player in the FPGA market, recognized for producing high-quality chips with a broad spectrum of applications. Lattice Semiconductor, on the other hand, specializes in manufacturing FPGAs with fewer logic elements, typically favored for their utilization in energy-efficient or embedded systems.

FPGAs find widespread application across various sectors. They are extensively employed in the development of embedded systems, including but not limited to medical devices, automotive electronic systems, and telecommunication equipment. Moreover, FPGAs play a crucial role in signal processing, facilitating the implementation of signal processing algorithms such as digital signal processing in electronics and imaging. Additionally, FPGAs are instrumental in cryptographic applications, enabling the rapid computation of cryptographic algorithms to safeguard sensitive information. Furthermore, some FPGAs excel in big computing tasks, capable of processing vast volumes of data and executing intricate computations vital to advancements in the realms of science and technology.

FPGAs are powerful tools for a wide range of applications, from embedded systems to signal processing and big data. The choice of a specific FPGA depends on the requirements of a particular project and the availability of built-in features, performance, and cost [6].

The Altera DE0 Board, manufactured by Terasic, was chosen to implement the specialized computer. The DE0 development and training board has a compact size and contains all the necessary tools for novice users to gain knowledge of digital logic, computer organization, and FPGAs. It is equipped with an Altera Cyclone III 3C16 FPGA that offers 15,408 LE blocks. The board has 346 user I/O pins and a rich feature set, making it suitable for use in university and college courses, as well as for developing complex digital systems. The DE0 combines Altera's low-power, low-cost, high-performance Cyclone III FPGA to drive the various functions of the DE0 board. The DE0 Development Board includes the software,

reference designs, and accessories needed to provide easy access for the user to evaluate their DE0 board.

To summarize the above, an FPGA is a user-programmable integrated circuit device that contains an array of programmable logic blocks (PLBs) connected by programmable routers. An FPGA can be programmed to perform any function or logical operation. They are commonly used to accelerate computing in large computing systems or to implement specialized computing devices. This allows you to prototype a variety of devices with minimal cost and time, because if there is an error during testing or you need to add new functionality, you just need to update the software. When the device has been fully tested and debugged, the manufacturer can order chips specifically for its tasks, based on the code previously created for the FPGA. It is up to each manufacturer to decide whether to do this - a specially created ASIC chip will run faster and consume less power, but launching such a chip into production can be very expensive, so even today, engineers often decide to leave the FPGA chip in the finished device [7].

With the help of hardware description languages, you can create hardware blocks of varying complexity on FPGA chips: from primitive (for example, a watchdog timer) to ultra-complex (for example, a driver for any modern high-speed video transmission line). Of course, there are large libraries of previously created blocks that you can use in your projects [8].

Chip manufacturers themselves are also trying to "simplify life" for engineers and offer many ready-made modules. One of them is worth a closer look. This is the NIOS II software processor.

The Nios II architecture encompasses several essential functional blocks, each playing a distinct role in the operation and functionality of the processor.

Firstly, the register file serves as a crucial component within the processor, facilitating the storage and manipulation of data during program execution. It provides a set of registers where data can be temporarily stored for processing [9].

The arithmetic logic unit (ALU) is responsible for executing arithmetic and logical operations on data stored in the register file. It performs tasks such as addition, subtraction, bitwise operations, and comparisons, enabling the processor to carry out various computational tasks.

The interface to the user command logic acts as a bridge between the processor and the user, facilitating communication and interaction with the system. It interprets commands from the user and translates them into actions or instructions that the processor can execute.

The exception controller manages exceptional conditions or events that may occur during program execution, such as interrupts, errors, or system faults. It ensures proper handling of these events to maintain the integrity and stability of the system.

An internal or external interrupt controller is responsible for managing interrupts generated by external devices or internal system events. It prioritizes and handles these interrupts, allowing the processor to respond promptly to critical events.

The instruction bus provides a pathway for transferring instruction data between the processor and memory. It enables the processor to fetch instructions from memory for execution.

Similarly, the data bus facilitates the transfer of data between the processor and memory or other devices. It allows the processor to read or write data to and from memory locations or peripheral devices [10].

The memory management unit (MMU) is responsible for managing memory access and address translation. It translates virtual addresses used by the processor into physical addresses

in memory, ensuring proper memory allocation and protection. The memory protection unit (MPU) is tasked with enforcing access control and security policies on memory regions. It prevents unauthorized access to memory locations and helps protect sensitive data from malicious or unintended actions [11].

A command and data cache improves processor performance by storing frequently accessed instructions and data in a faster, closer memory layer. This cache reduces the time and resources required to fetch instructions and data from main memory, enhancing overall system efficiency. Memory interfaces for hard-coded commands and data enable the processor to communicate with specialized memory modules or devices that store essential instructions or data required for system operation.

Finally, the JTAG debugging module provides debugging and testing capabilities for the processor. It allows developers to inspect and manipulate the internal state of the processor during runtime, facilitating the identification and resolution of software bugs or errors [12].

In the created special-purpose computer, this software processor performs the most important functions, namely, it is responsible for system operation, monitoring and allocation of hardware resources such as CPU time, i.e., the time to perform computational tasks. In fact, it allows engineers not to waste their time writing a module for calculating a function, but rather to use functions that are already built into the processor. This approach was also used in the process of working on a special purpose computer [13].

Created special computer. In the process of studying the NIOS2 core and the possibility of implementing a specialized calculator on its basis, the authors decided to develop a prototype calculator that performs basic mathematical operations. These operations include addition, subtraction, multiplication, and division. Of course, the NIOS2 software processor supports the calculation of much more complex mathematical functions, but the main goal of this study is to demonstrate the viability of the concept of a special-purpose calculator that uses the NIOS2 core with additional functionality implemented using the Verilog hardware description language.

The development of the computer begins with the creation of a project in the Quartus II environment. The entire architecture of the special-purpose computer was described in the Verilog language. At the same time, the NIOS2 software processor is being integrated using the Qsys environment. Another important stage was the creation of software to be executed on the program processor. This task was performed using NIOS II software build tools for Eclipse.

The first task is to initialize the NIOS 2 software processor and connect all the software peripherals necessary for its operation. Among the main applied program parts, it is worth highlighting the “clk_0” block, which is responsible for controlling the clock pulses for the rest of the program blocks. All peripherals are connected to the “nios2” software processor itself. The processor reads/stores data and instructions for its operation in the “onchip_memory” memory block. The System ID Peripheral block was used to set a unique system identifier (in the diagram it is the “sysid” block). One of the most important components of the system is a 32-bit input-output port, which serves to communicate the software processor with another part of the special computer written in Verilog. In the diagram, this block is called “pio_0”.

The last optional block is “jtag_uart_0”, which is used to send diagnostic messages during software development. Fig. 1 shows a diagram of the connections of program blocks in the Qsys environment.

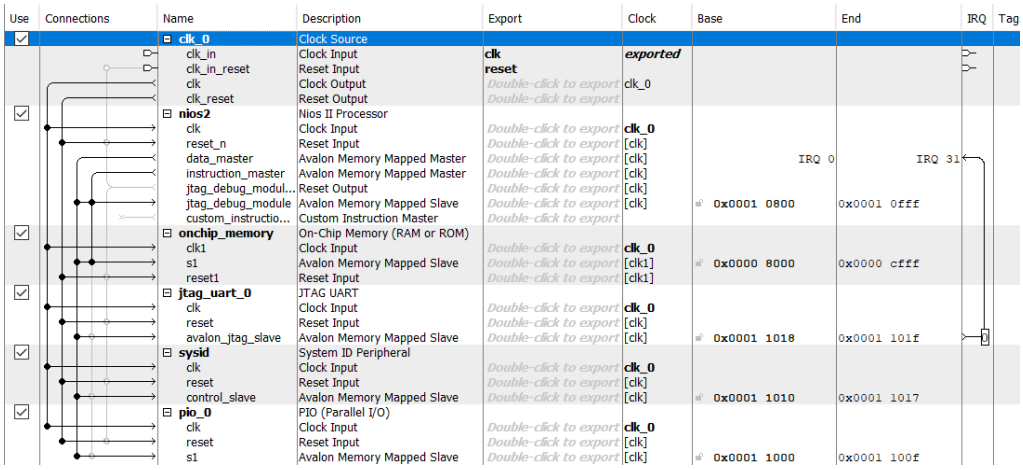

Fig.1. Scheme of connections of program blocks

The communication between the program processor and the rest of the special-purpose computer is carried out using the I/O port “pio_0”. Each of its bits is specially allocated for a certain signal. Fig. 2 shows the distribution of bits.

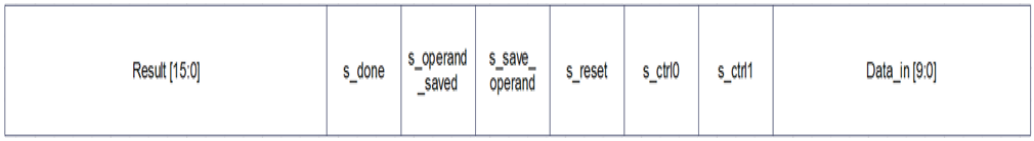

Fig. 2. Bits distribution in the port

The special-purpose calculator interacts with the user through the use of four seven-segment indicators, which display the result of calculations. Input data is set using ten switches. The action is selected by pressing one of the three push buttons.

The part of the special-purpose calculator written in Verilog performs important functions for decoding and displaying the result on seven-segment indicators. The “Debouncer” unit protects the clock button contacts from jitter. The “ALU_CONTROL_UNIT” unit reads input data from switches and pushbuttons, processes this data and generates operation codes and control pulses for indicators and the program processor. The “SEG7_DRIVER” block converts the result of calculations into signals required to control the seven-segment indicators. Fig. 3 shows the structural scheme of the created special purpose computer.

The next important task is to implement the software that the NIOS II processor executes. It functions as follows: first, the instruction code is read from the I/O port "pio_0". The processor decodes this instruction and starts calculating one of four mathematical operations or does nothing and moves on to the next input port read. After the calculation is complete, the processor sets the corresponding flag in the output port and outputs the result of the calculation to the high 16 bits of the I/O port. The processor then waits again for the next instruction. The

remaining bits of the I/O port are used for synchronization signals between the two parts of the computer. The principle of operation of the software is shown in Fig. 4.

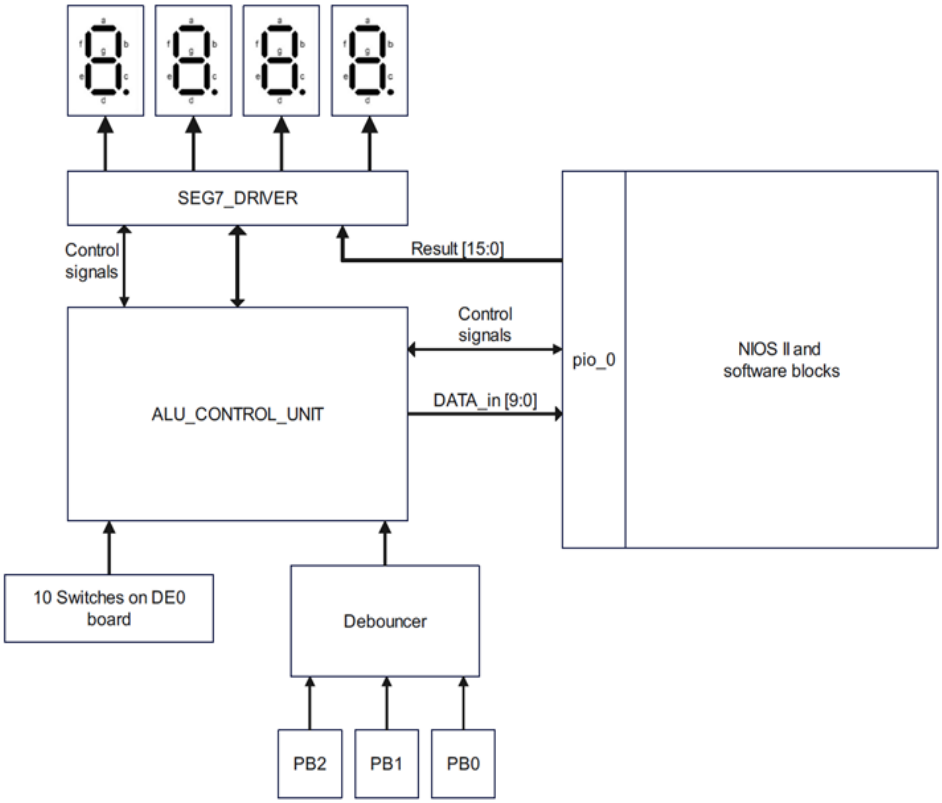

Fig.3. Structural scheme of the created special computer

Testing of the created special computer. After the authors had fully developed the prototype of the special-purpose computer, they created an algorithm for testing the resulting device. Testing of the device begins with the input of the first operand (10 bits) on the switches. The next step is for the user to press the button to save the variable. At this time, the seven-segment LED indicators display the entered and saved value of the first variable. Next, the combination corresponding to the second variable is set on the switches. Pressing the button saves the second variable to the register. Now it is displayed on the seven-segment indicators. The last step to start the calculator is to select the instruction code. Set the appropriate instruction code on the switches and start the calculation by pressing the corresponding button. The result of the calculation will appear on the seven-segment indicators.

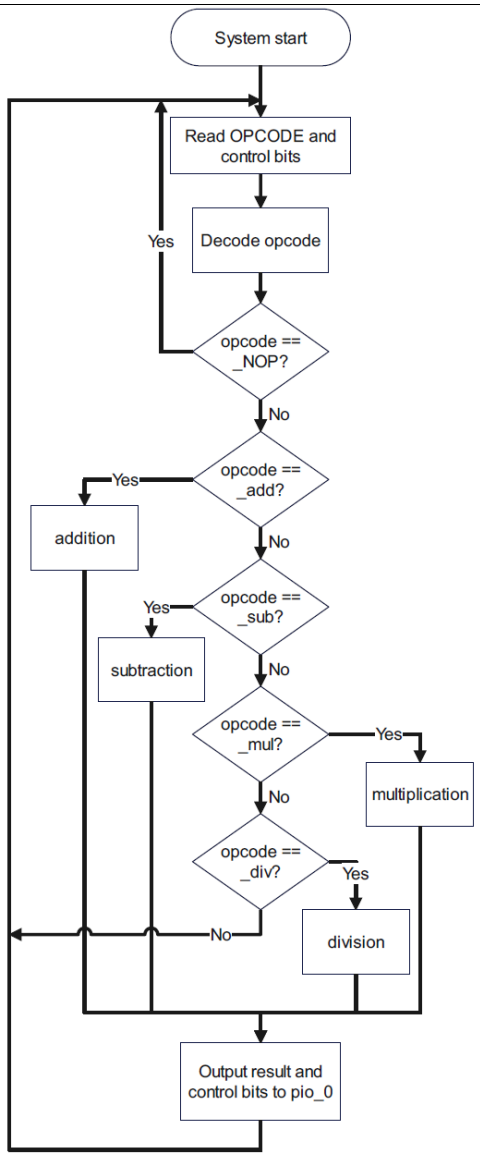

Fig. 4. The principle of operation of the software

Additionally, the calculation result will be output via the jtag interface to the console of the personal computer to which the board is connected. If the calculation process fails or registers are overflowed, the combination "----" will be displayed on the seven-segment indicators. Fig. 5 shows the sequence of testing a special-purpose computer.

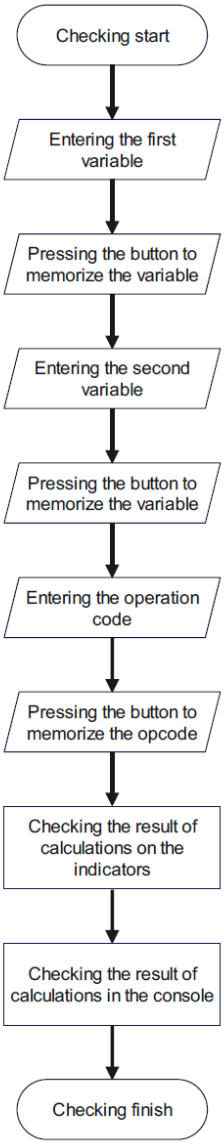

Fig. 5. Sequence of testing a special-purpose computer

The correct functioning of the four main mathematical operations was checked step by step. The numbers 831 and 275 were chosen as operands. First, the result of addition was checked, then subtraction, then multiplication and division. All the results were carefully checked by the authors and proved to be completely correct. This makes it possible to conclude that the created special calculator works completely correctly. Fig. 6 shows the terminal window with the program results.

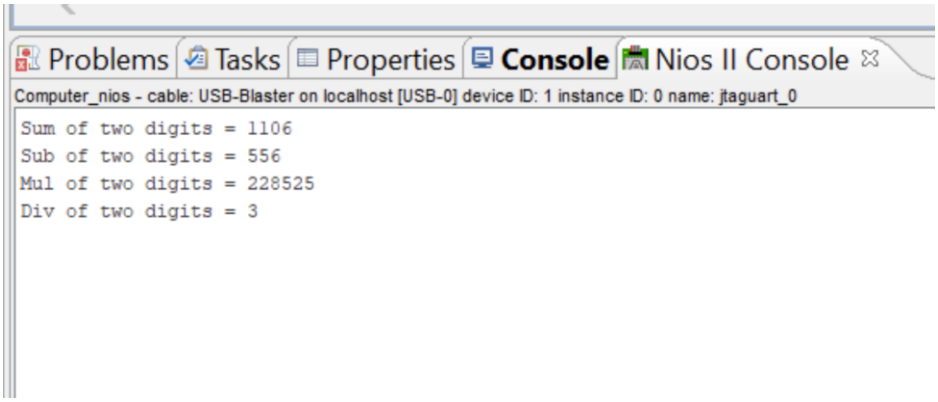

Fig. 6. The terminal window with the program results a special-purpose computer

Additionally, the authors tested the functioning of the NIOS II software processor separately. This was done by writing a separate code that calculates the same numbers as those used in the previous example. The results of the test were identical, which further confirmed the correct operation and settings of the software processor. Fig. 7 shows the code used to check the correct functioning of the program processor.

```c
#include <stdio.h>

int main()
{
    int a_num = 831, b_num = 275;
    int sum, div, mul, sub;

    sum = a_num + b_num;
    sub = a_num - b_num;
    mul = a_num * b_num;
    div = a_num / b_num;

    printf("Sum of two digits = %d\n", sum);
    printf("Sub of two digits = %d\n", sub);
    printf("Mul of two digits = %d\n", mul);
    printf("Div of two digits = %d\n", div);

  return 0;
```

Fig. 7. The code used to check the correct functioning of the program processor

Additionally, the authors noted the correctness of the created value decoder for seven-segment indicators. After conducting more than a hundred tests, the authors have never seen any errors in its functioning. Fig. 8 shows the code of the general data decoder for seven-segment indicators.

```
1  module SEG7_LUT_4 (  oSEG0,oSEG0_DP,oSEG1,oSEG1_DP,oSEG2,oSEG2_DP,oSEG3,oSEG3_DP,iDIG );
2  input [15:0]   iDIG;
3  output   [6:0] oSEG0,oSEG1,oSEG2,oSEG3;
4  output         oSEG0_DP,oSEG1_DP,oSEG2_DP,oSEG3_DP;
5
6  SEG7_LUT u0 (  oSEG0,oSEG0_DP,iDIG[3:0]   );
7  SEG7_LUT u1 (  oSEG1,oSEG1_DP,iDIG[7:4]   );
8  SEG7_LUT u2 (  oSEG2,oSEG2_DP,iDIG[11:8]  );
9  SEG7_LUT u3 (  oSEG3,oSEG3_DP,iDIG[15:12] );
10
11 endmodule
```

Fig. 8. The code of the general data decoder for seven-segment indicators

During the test, the created special computer fully proved its performance. The concept proved to be fully relevant and workable. As a further development of such a device, it is possible to realize the calculation of much more complex mathematical operations on the NIOS II program processor. It is also possible to describe the calculator modules for certain functions in the Verilog language. This will allow to perform relatively standard calculations on the software processor, and implement the functions that are difficult to calculate in hardware, which will significantly speed up the speed of calculations.

In the future, the authors plan to compare the performance of computing complex and non-standard mathematical operations on the NIOS II software processor and on a Verilog-based computing module. For maximum measurement accuracy, it is planned to use 4 pins on free I/O ports. At the start of the calculation, the module and the program processor will set a logical one on a pin, and after the calculation is completed, two other pins will also be set to a logical one. This will allow you to accurately measure the time of function calculation on the program processor on the Verilog module using an oscilloscope. It is expected that the calculation on the Verilog module will be much faster.

Conclusion. The results of the study demonstrate the importance and success of developing a special-purpose mathematical computer based on Altera Cyclone 3 FPGAs with the NIOS2 core. This project involves the implementation of high-performance algorithms and structural schemes, which significantly increases the performance of the computer system. The analysis of the structural scheme confirms its high performance and potential for further optimization. These results stimulate further development of innovative technologies and the introduction of new optimization methods to achieve even greater performance and efficiency in various applications. During testing, the created special-purpose computer confirmed its high reliability and correct functioning. This paves the way for processing much more complex mathematical expressions based on it, and will also allow in the future to compare the speed of mathematical operations on processors with conventional instructions (for example, on NIOS2), as well as on modules specially written for these operations in the Verilog language.

REFERENCES

[1] *Pinto C. F.* Development of Altera NIOS II soft-core system to predict total hemoglobin using multivariate analysis / C. F. Pinto, J. S. Parab, M. D. Sequeira, G. M. Naik // Journal of physics: conference series. – 2021. – Vol. 1921(1). – P. 012039.

[2] *Meyer-Baese U.* Embedded microprocessor system design using FPGAs / U. Meyer-Baese // Berlin/Heidelberg, Germany: Springer. – 2021. – P. 23-45.

[3] *Suárez-Gómez A. D.* FPGA-based custom ip for quadrature encoders decoding / A. D. Suárez-Gómez, W. J. Pérez-Holguín // Revista Politécnica. – 2020. – Vol. 16(32). – P. 68-76.

[4] *Romanov A. Y.* The usage of a simple SchoolMIPS Soft-Processor core for teaching students the computer microarchitecture / A. Y. Romanov, S. Zhelnio, L. G. Izmailova, A. E. Ryazanova // International Conference on Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS). - 2022 2022. – P. 382-387.

[5] *Savaş G.* FPGA-based remote accessible Laboratory Designs / G. Savaş, Z Albayrak // Electronic Letters on Science and Engineering. – 2022. – Vol. 18(1). – P. 21-30.

[6] *Mohammed N. Q.* Implementation dual parallelism cybersecurity architecture on FPGA / N. Q. Mohammed, A. Amir, M. H. Salih, H. Arrfou, Q. M. Hussein, B. Ahmad // J. Commun. 2022. – Vol. 17(5). – P. 386-392.

[7] *Zajic A.* Using analog side-channels for malware detection / A. Zajic, M. Prvulovic // Understanding Analog Side Channels Using Cryptography Algorithms. – 2023. – P. 151-209.

[8] *Jidin A. Z.* FPGA implementation of low-area square root calculator / A. Z. Jidin, T. Sutikno // TELKOMNIKA (Telecommunication Computing Electronics and Control). – 2015. – Vol. 13(4). – P. 1145-1152.

[9] *Guerrieri A.* Applications enabled by FPGA-based technology / A. Guerrieri, A. Upegui, L. Gantel // Electronics. – 2023. – Vol. 12(15). – P. 3302.

[10] *Ren W.* The design and implementation of high-speed codec based on FPGA / W. Ren, H. Liu // 10th International Conference on Communication Software and Networks (ICCSN). – 2018. – P. 427-532.

[11] *Sikka P.* Real time FPGA implementation of a high speed and area optimized Harris corner detection algorithm / P. Sikka, A. R. Asati, C. Shekhar // Microprocessors and Microsystems. – 2021. – Vol. 80. – P. 103514.

[12] *Moslehpour S.* Design of the Nios II system for the playing of wave files on an Altera DE2 Board / S. Moslehpour, K. Jenab, P. D. Weinsier, B. K. Matcha // International journal of engineering and technology. – 2013. – Vol. 5(3). – P. 361.

[13] *Cheng C. B.* Implementation of a camera system using Nios II on the Altera DE2-70 board / C. B. Cheng, A. B. Jambek, Indonesian Journal of Electrical Engineering and Computer Science. – 2019. – Vol. 13(1). – P. 513-522.

ISSN 2224-087X. Electronics and information technologies. 2024. Issue 26

РОЗРОБЛЕННЯ МАТЕМАТИЧНОГО СПЕЦОБЧИСЛЮВАЧА НА БАЗІ FPGA ALTERA CYCLONE 3 З ВИКОРИСТАННЯМ ЯДРА NIOS2

Р. Дячок[1], В. Лисяк[1], Г. Клим[1,2]
[1]кафедра спеціалізованих комп'ютерних систем,
Національний університет «Львівська політехніка»,
вул. С. Бандери, 12,79013 Львів, Україна
roman.v.diachok@lpnu.ua, vitalii.Lysiak.KI.2020@lpnu.ua, halyna.i.klym@lpnu.ua
[2]кафедра радіоелектронних і комп'ютерних систем,
Львівський національний університет імені Івана Франка
вул. Драгоманова, 50, 79005, Львів, Україна
halyna.klym@lnu.edu.ua

У роботі досліджено та проаналізовано процес розроблення математичного спеціалізованого обчислювача на основі FPGA Altera Cyclone 3 з використанням ядра NIOS2. Основна увага приділена вивченню технічних аспектів проєкту, таких як апаратна архітектура та алгоритмічне забезпечення. Також проаналізовано алгоритми та структурні схеми обчислювача для максимальної оптимізації його роботи та використання ресурсів FPGA.

В процесі дослідження ядра NIOS2 та аналізу можливості реалізації на його основі спеціалізованого обчислювача було розроблено прототип калькулятора, який виконує базові математичні операції. Основному задачею цього дослідження полягала у демонстрації життєздатності концепції спецобчислювача, реалізованого за допомогою мови опису апаратних засобів Verilog, у якому використовується ядро NIOS2 з додатковим функціоналом.

У роботі детально опрацьовані та описані ключові етапи розроблення системи. Серед цих етапів можна виділити розроблення концепції архітектури спецобчислювача, алгоритму роботи програмного забезпечення, а також взаємодії створеного спецобчислювача з користувачем. Додатково було здійснено моделювання роботи обчислення базових математичних операцій на ядрі NIOS2 для підтвердження концепції та працездатності створеного пристрою.

Під час перевірки реалізований спецобчислювач підтвердив свою працездатність. Як подальший розвиток такого пристрою буде реалізовано обчислення складніших математичних операцій на програмному процесорі NIOS2. Також буде описано обчислювач для певних функції на мові Verilog, що дозволить виконувати відносно стандартні обчислення на програмному процесорі, а складні функції реалізувати апаратно на модулі обчислення імплементованому за допомогою Verilog коду. Такі рішення сприятимуть збільшенню швидкодії, а також нададуть можливість здійснювати обчислення нестандартних виразів, які складно та часозатратно реалізовувати на звичайному комп'ютері.

Загалом, представлені результати сприятимуть подальшому розвитку інноваційних технологій в сфері обчислювальної техніки.

Ключові слова: FPGA, NIOS2, Cyclone 3, Altera.

The article was received by the editorial office on 15.05.2024.
Accepted for publication on 23.05.2024.