VOL. 13, NO. 18, SEPTEMBER 2018 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

# ASSOCIATIVE COPROCESSOR BLOCK BASED ON PLD FOR DIFFERENT COMPUTER SYSTEMS

Martyshkin A. I.
Penza State Technological University, Russia, Baydukov Proyezd / Gagarin Street, 1a/11, Penza, Penza Region, Russia
E-Mail: alexey314@yandex.ru

**ABSTRACT**

The article considers the possibility of implementing an associative coprocessor block on a modern element base for specialized computer systems. The goal of the article is the development and study of the associative co-processor block on PLD, for specialized computing, for example, multi-processor, systems, performing associative function and data storage functions. The object of research and development of this article is the associative coprocessor based on PLD. The search operation is widely used in different purposes as users and systems. However, this operation is one of the most labor-intensive and requires time-consuming when implemented in the traditional way, when the search data is sequentially read from the machine RAM to the CPU, where they perform the appropriate operation. The paper proposes associative co-processor connected to the PCI bus of a computer system, providing search and comparison of "more – less" simultaneously in 32 of said pre-loaded in the associative memory. Finally, conclusions have been made. The use of the VHDL language as a universal means of hardware description of integrated circuits provides the flexibility of the project and the ease of debugging the operation of the device. Modeling the associative co-processor was carried out in CAD Web pack ISE of Xilinx company. This allows evaluating the correct operation of the hardware coprocessor in the composition of computing systems without building the actual layout. Introduction efficiency of the module is assured by the fact that the coprocessor performs a time-consuming operation to search for and compare data and thus relieves the CPU and increases the performance of the computing system as a whole. The results obtained in the article can find application in the search engines for different purposes: servers, databases, search machines used at stations, airports and for fast implementation of the search task operating systems.

**Keywords:** module, block, associative memory, addressing, coprocessor, computer system, memory cell, fixation reactions, analyzer of multiple coincidence, memory addressing, bus interface, hardware implementation, priority analyzer, write cycle, read cycle.

## 1. INTRODUCTION

The main area of application of computers today is work with large amounts of data, in which the most laborious operations are all sorts of searches and sorting of data. Existing computing systems (Comps) use the address memory architecture, i.e. to search some data in memory, we need to read each memory module address and compare it with the search argument. As a result, the search for information in memory takes a lot of computer time. This circumstance has a very negative effect on the speed of the Comps as a whole. It is much faster to access the data by association (by content). The essence of the principle of addressing by content is as follows (Figure-1). There is an array of data with a capacity of N words, it requires we to find all the words beginning with the "A" symbol and ending with the "H" symbol. Here, the search argument is the word A***H, where the "*" mark denotes digits that do not affect the search result. The storage array on the hardware level is constructed so that the signal of coincidence appears on the output of the memory cells (MC) whose contents coincide with the value of the incoming search argument. Further, on the produced coincidence signals, a selection of the MC is performed.

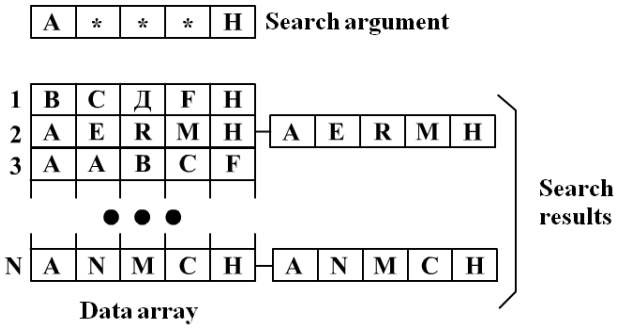


**Figure-1.** The essence of the principle of addressing by content.

## 2. GOAL SETTING

This article as a whole is of a research nature. In the course of studying the subject area, literature [1-4] was analyzed in order to search for unaffected and unresolved problems. Some issues related to the possibility of hardware implementation of an associative coprocessor for fast data retrieval have not been adequately reflected in publications, but this was partially considered in that works [5 - 8] and [9-11].

The purpose of this article is to develop and study the block of an associative coprocessor based on programmable logic device (PLD) for specialized computing, for example, multiprocessor systems. This issue is relevant today due to global informatization and the almost universal operation of colossal volumes of various data.

To achieve this goal, the article solves the problems of determining the structure of the device and the principles of its functioning.

The developed hardware coprocessor has the ability to address and associative access to the data stored in memory. Addressable access is required to work with a specific record and to use test libraries developed for address memory.

The device consists of two parts: the main part that implements the functions of the associative coprocessor, namely: ordinary (address) write to the associative memory device (AMD); associative recording in AMD; normal reading from the AMD; associative reading from the AMD; search for coincidences and a part of the interface with the CompS on which the function of converting signals coming from the central processor unit (CPU) is assigned to the signals with which the coprocessor will operate, i.e. this part of the device organizes the interface with the CPU.

Today, associative access to data is realized in two ways: software based on the distribution of memory, depending on the content of data and implemented by software, and hardware, based on the use of special hardware designed to store and associatively search for data items. It can be implemented in the form of parallel AMD, where the search argument enters all the MCs in parallel. As a result, a massive comparison operation is performed, as a result - the search is performed in one clock cycle. Another option for implementing the hardware method of associative data access is consecutive bitwise AMD, where the search occurs bitwise. In this case, the search time directly depends on the number of bits of the data bus.

The processes analogous of the biological mechanisms for remembering and processing information can be represented with the help of various models of associative memory (AM), which allow to display relations (associations) of arbitrary complexity between information objects. However, all these relationships can be realized in the form of simple constructions - triples of components: an ordered pair of information objects O and V and the type of the relation A: $O \xleftarrow{A} V$ [12]. One of the simplest models of AM for displaying such relationships is shown in Figure-2, a. The model consists of an associative storage environment associated with two input channels and one information output channel.

In the write stage, the input information is fed to the input K, and the attribute information C is provided on the second channel, representing the context in which the input information is written into the memory.

At the associative sampling stage, when the key K appears on the output of the memory, a response R is formed associated with the key K. Thus, the information stored in the memory can be selected using any of its fragments used as search engines. By specifying a different context C, we can more precisely specify the information to be sampled.

In the context of the definition of AM introduced, there is the question of organizing the accumulation and retrieval of structured data in such a way that access to them is possible on the basis of an associative sampling. Figure-2, b shows the AM model, which allows to answer the questions: how to organize the recording and retrieval of structured information elements, and also perform a cycle of the search process, in which the selected information element becomes the key for searching for new information [12].

Three sets of values can be entered simultaneously on the three input channels. The output channel serves for sampling information. On the first channel, at the time t, the address information K (t) is supplied, and the second sign shows the sign C (t). The response of R (t) along the feedback channel is also fed to the input of the associative environment. In the operation of such an AM, the keys K (t) and the signs C (t) are fed through time intervals corresponding to the delay of the feedback channel.

The memory operation process will be considered under the assumption that the triple [K (t), C (t), R (t-Δ)] is a single static image, given at time t, and it is possible to simultaneously write it into memory in one operation. Suppose also that at the write stage R (t) and K (t) are the same.

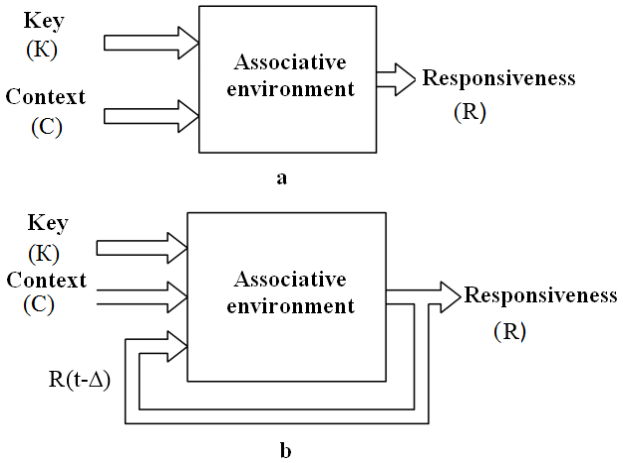


**Figure-2.** Model of associative memory without feedback (a) and with feedback (b).

At the write stage, K (t) and C (t) are fed to the inputs of the AP, and R (t), identical to K (t), is formed at the output. After this, with a delay Δ at the input, R (t-Δ) is formed. Each new triplet, appearing at the entrances, is wrote in memory.

At the sampling stage from AM the K key, associated with the context information C is fed to the input, after which K can be removed from the input. As a result, a copy of K appears at the output as a response. When a delayed signal R (t-Δ) appears at the memory input, the pair (C, R) becomes the new key sign, leading to an associative sample of the next image R (t), etc. Thus, the entire recorded sequence of images is selected together with the context information. That model considered implements a memory suitable for writing and sampling structured knowledge.

Connecting the described device to the Comps is possible in several ways [12]: direct connection to the processor bus, with the coprocessor included in the motherboard; Connection to the serial interface (such as, USB), with this method of connection, the coprocessor is implemented as a separate housing and is provided with a

separate power supply; connection to the computer's expansion bus (PCI), in which case the coprocessor is implemented as an expansion board.

Based on the mentioned possible ways of connecting the device being developed to the aircraft, we will determine the architectural and structural features of the associative coprocessor, having previously compared them with existing analogues. As mentioned above, with direct connection to the processor bus, the device will have to be included in the motherboard, which will lead to an increase in the cost of the coprocessor and its universality. The speed of such system is high.

When connected using the USB interface, the coprocessor will look like a separate external module, but it will work in sequence, which leads to a decrease in performance. But such a block is comparatively cheaper than the previous one.

Connecting the coprocessor to the PCI bus will allow it to be implemented as an expansion board, relatively inexpensive to implement, unlike the analogues mentioned. In this case, work with the module will be carried out on a parallel interface, which will achieve maximum performance in comparison with analogues.

Analyzing the advantages and disadvantages of the known and practiced methods of organizing the main part of the device and part of the interface with the CompS, it was decided to perform the main part in the form of parallel AMD, since this method has the maximum performance. Connection to the dedicated CompS is realized by the PCI bus, it has a sufficiently high throughput. Moreover, the associative coprocessor will be located in the address space of the I/O of the CompS.

## 3. DETERMINATION OF THE STRUCTURE OF ASSOCIATIVE COPROCESSOR BLOCK

The simplified block diagram of the coprocessor can be represented in the form of two blocks (Figure-3, a): the coprocessor (main part) and the PCI bus interface (part of the interface with the system). Analyzing the chosen method of implementing the main part of the coprocessor, it was decided that the main unit is the memory module of the AMD, which is an array of MC and performs the functions of storing data and searching for associations with an argument. The MC consists of a storage element that performs data storage functions and a comparison scheme that performs a search, i.e. generating signals indicating the equality or disparity of the contents of the cell with the argument (Figure-3, b).

To implement the functions of writing and reading data from the MC to the coprocessor, it is necessary to add a block of the multiplexer and address decoder to generate the "Cell Select" signal for a particular MC. Also it is necessary to include in the structure of the coprocessor block a block of the memory of reaction fixing for the preservation of the values of the responded cells. To calculate the number of responding cells, a block of coincidence counting is introduced into the coprocessor. The block of the multiple much analyzer (MMA) is necessary for priority selection, based on the results of which signals are generated for the encoder block necessary for converting the binary sequence into an address, with associative reading or recording, which goes to the inputs of the address selector. To store the search argument in the module, there is an argument register. Controls the operation of the coprocessor, the command decryption unit, which generates control signals. The register of the mask is excluded from the composition of the coprocessor, since the comparison circuit generates three signals ("EQUAL", "MORE" and "LESS") and the need to mask the search argument bits disappears (Figure-3, c).

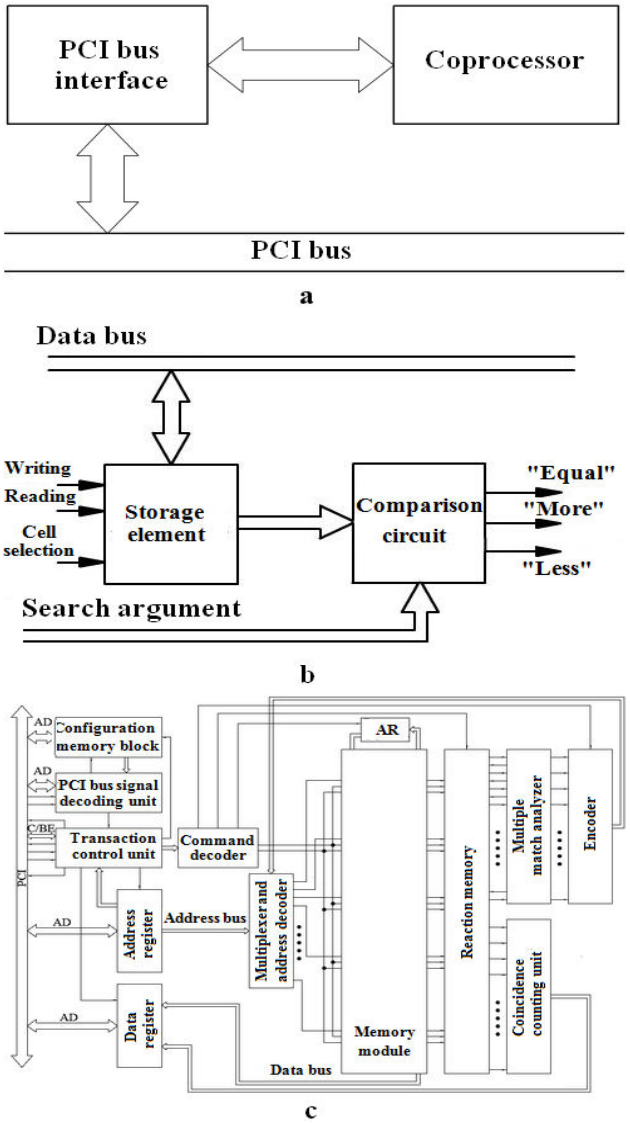


**Figure-3.** Simplified block diagram of the modular associative coprocessor (a); block diagram of the memory cell (b); a detailed block diagram of the associative coprocessor module (c).

Description of the associative coprocessor block at the functional level.

The basis of the associative coprocessor is the memory module, which is an array of MC. The storing element of the MC is implemented on a parallel register, which is an array of D-flip-flops, providing the maximum

performance and minimum logic necessary to ensure the storage of information. The main signals for the register are: a 32-bit D signal, which receives data, a CE signal and a 32-bit Q signal, from which the data stored in the register is read. The comparison scheme is implemented on the basis of a comparator. The main signals for it are: 32-bit signals A and B, which are fed with arguments for comparison, signals =, < and >, from which the results of comparison are read. A buffer element is included in the memory cell to disconnect the output data bus from the common bus. The main signals of the buffer element are: a 32-bit signal D, a 32-bit signal Q and a signal T. The functional diagram of the MC is shown in Figure-4, a. The principle of operation of the MC is as follows. The search argument is supplied to the ArgI input. Through the DataI input, the data falls into the MC. From the DataO output, the data is read from the SD. The Write and CS inputs are used to control the operation of the MC. The CS signal is selected MC by the CS signal, i.e. the buffer element BUFT passes the signals from the output of the register RG to the output DataO bus. On a single Write signal, data from the DataI input is written to the RG. The outputs Equal, More and Less form the signals "Equals", "More" and "Less" respectively.

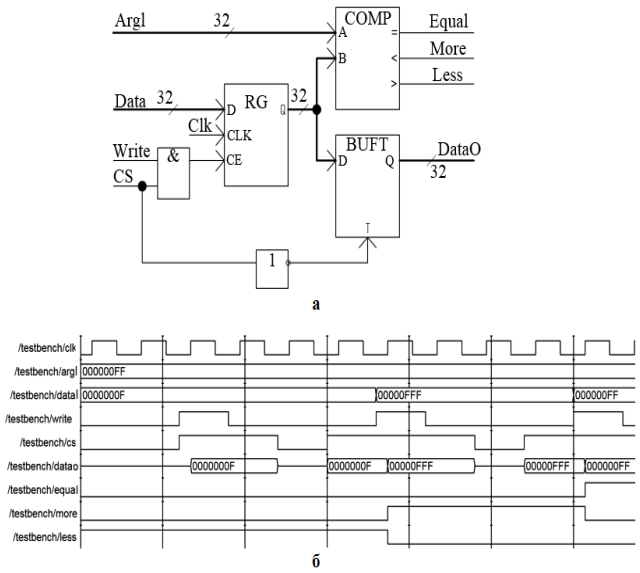


**Figure-4.** Functional diagram of the memory cell (a); time diagrams of memory cell operation (b).

The time diagrams of the operation of the MC, confirming its operability, are shown in Figure-4, b. Here is a record of a number of values: "F", "FFF" and "FF". The search argument is "F". From the obtained time diagrams, it is seen, while the signal CS is equal to logical zero, the output bus (DataO) is in the third state, i.e. Is disabled. When a logical unit is fed to the CS input, the values stored in the RG can be read from the data bus (DataO). The entry in the cell is performed by feeding a logical unit to the Write input. Figure-4, b shows that as soon as a new value is recorded in the MC, the results of the search are set on the outputs Equal, More and Less.

MMA can be implemented based on the shift register (Figure 5, a) and on the basis of the priority analyzer (Figure-5, b). The main elements of the MMA based on the shift register are the looped shift register and the address counter. The result of the search of all MC is written to the circular shift register (memory of reaction fixation). Then a sequence of clock pulses is sent to the register and to the counter. The contents of the register are shifted toward the upper digits until the first one has a logical "1". At this point, the clock signal is automatically blocked. If in the initial state in the counter one zeros were written, then at the end of the count, its contents directly indicate the address of the first matched word. This code is entered in the address register, after which the word is read. Then the unit in the first digit of the register is reset and the clock pulses are automatically resumed. Again, the contents of the register are shifted up until the appearance of the next unit in its first digit. After that, the next matched word is read, etc., until the whole queue is served. It should be noted that with such a sampling organization, there is no need for an address coder.

MMA based on the priority analyzer consists of D-flip-flops, which perform the functions of the memory of reaction fixation and combinational logic. This circuit operates on the front of the Clk signal. The search results in the MC are fed to the inputs M0...M31, at the Fix signal they are fixed on the D flip-flops. The Work signal permits the operation of the MMA. The priority analyzer itself is a logical circuit that allows you to select the line with the lowest number among your inputs set to "1". It is built on the principle of consecutive connection of discharges. Each single input of this circuit blocks the action of lines with large numbers, as a result of which only the output corresponding to the first active line is set in the unit. A single signal corresponding to the first active line automatically goes to the output, and the function of the reset signal is reset of the first of the "responding" triggers.

The memory for fixing reactions and MMAs is implemented on the basis of a priority analyzer scheme; this circuit has a faster response time than a shift register-based scheme. The resulting time diagrams of the operation of the circuit are shown in Figure-5, c.

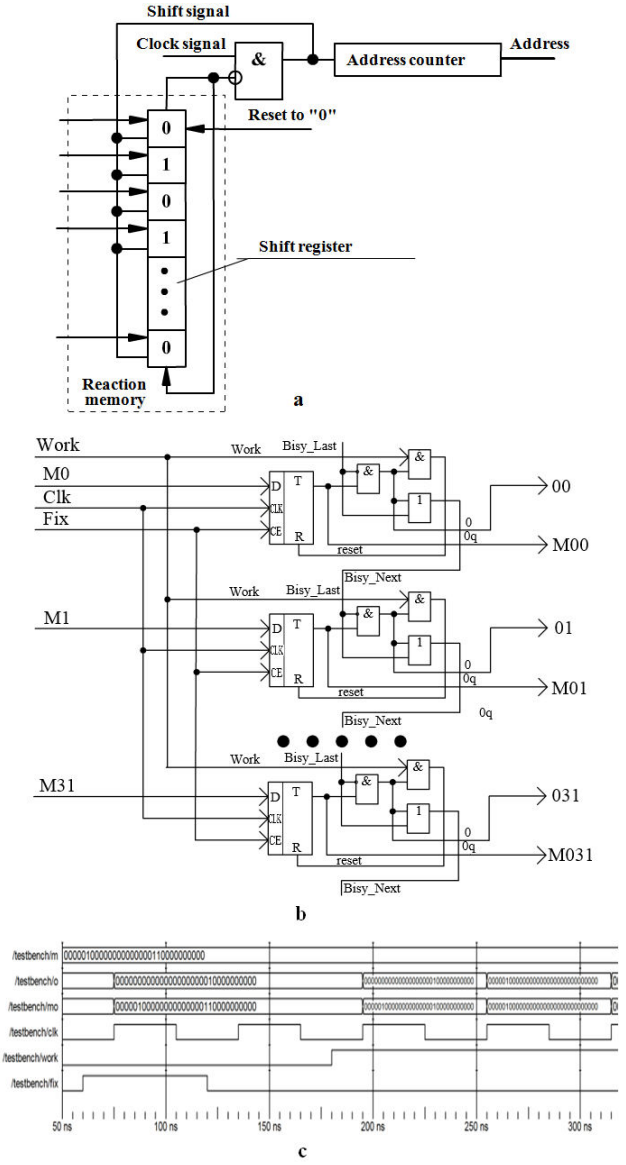


**Figure-5.** Multiple much analyzer based on the shift register (a) and on the basis of the priority analyzer (b); time diagrams of the work of the memory of the fixation of the reactions and the multiple much analyzer (c)

The coincidence-counting unit can be implemented in several ways: on the basis of the shift register and the counter and on the basis of cascades of adders.

The scheme of the coincidence counting unit based on the shift register and the counter (Figure 6, a) consists of a shift register into which a sequence of ones and zeros is loaded, corresponding to the responded and non-responding cells, the counter that counts the quantity, and the element "logical AND", which controls the counting process. The clock signal is applied to the circuit. The contents of the shift register are shifted one up and depending on the value at the output of the shift register, the counter value increases or does not increase, i.e. all counts of the register equal to one are counted. The number of clock pulses required to count all cells is equal to the number of all memory cells (in this article it is 32 cycles). In addition, this scheme requires an additional step for loading the shift register.

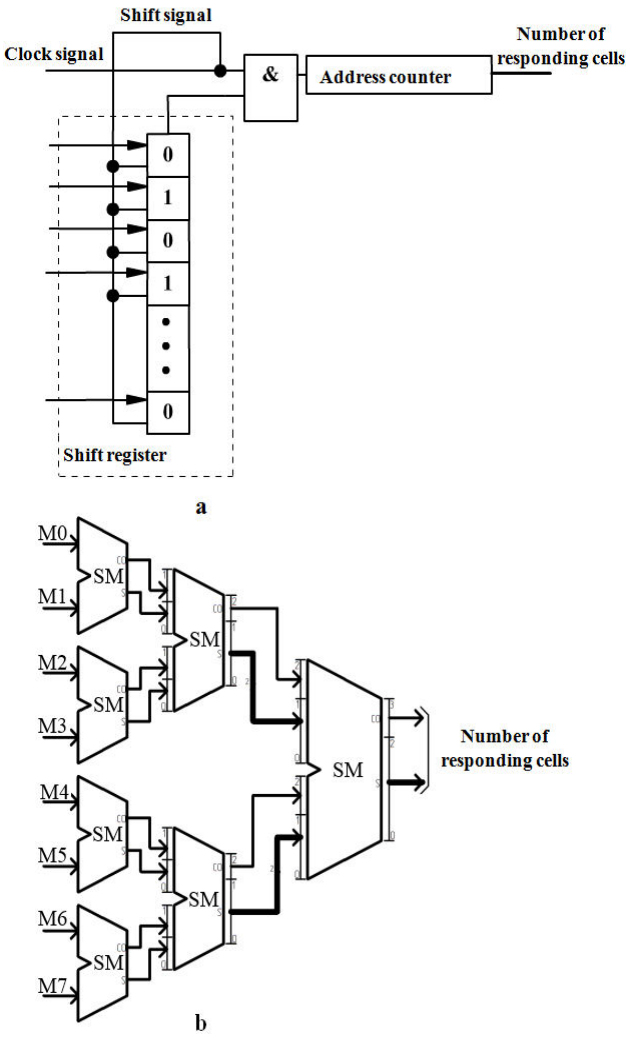


**Figure-6.** Block of counting coincidences on the basis of the shift register and the counter (a) and on the basis of cascades of adders (b).

The circuit based on the cascades of the adders (Figure-6, b) is a cascade of adders of different digits, i.e. we divide the input binary sequence into groups of two and apply these groups to the inputs of single-digit adder. Intermediate results obtained on single-digit adorers are also grouped into groups of two and are connected together with the transfer signal to the input of two bit totalizers and so on. In comparison with the previous scheme, this is faster and in principle can count the number of cells per clock cycle, so the counting unit is implemented on the basis of cascades of adders.

The correct functioning of the block of the associative coprocessor was checked during the execution of a number of computational experiments. It was experimentally obtained that the use of a hardware coprocessor (when performing search and comparison operations) allows increasing the productivity of the computer system by an average of approximately 25% in

comparison with systems that include only traditional processors. In other words, the system with the associative coprocessor shows a 1, 25 times higher performance than the analogs. Hence the expediency of applying this development to practice. An expansion board with an associative coprocessor block can be used to improve the performance of servers (by processing information during the reading from memory, and also reducing the load on the bus) databases such as Oracle and SQL Server located on x86 machines, as well as for processing graphics, While the use of an expansion board should be economically justified by the criterion of "cost-effectiveness".

The article describes the structure of the block of the associative coprocessor based on PLDs, which differs from the analogs in that the device under consideration is executed in the form of a functionally independent block, in which algorithms are implemented to perform laborious operations of searching and comparing data. In the traditional organization of such devices, these operations were performed directly by the processor. It is also necessary to note one more advantage of this development, which is connected with the implementation of the block under consideration on the modern element base (PLD).

The considered hardware block of the associative coprocessor is implemented on the PLD of Xilinx. The VHDL codes of a device consisting of four modules are included, including the main blocks of the device under consideration: an associative coprocessor, a reaction memory and a multiple coincidence analyzer, a description of the PCI interface, an associative coprocessor interface, and a PCI interface.

## 4. CONCLUSIONS

The paper discusses the structure of the associative coprocessor block based on PLDs. Based on the above description of the device blocks at the functional level, the VHDL code of the associative coprocessor was developed and debugged.

The device mentioned in the work can be physically realized in the form of an expansion board Comps, connected via the PCI interface. The associative coprocessor is implemented in hardware, which allows performing laborious searches and comparisons, thereby unloading the CPU and increasing the overall performance of the Comps.

The associative co-processor described in the article performs the following functions:

a) the address record in the AMD;

b) address reading from the AMD;

c) search for data of equal, larger and smaller search arguments;

d) associative reading (reading elements greater than, less than or equal to the argument);

e) Associative record (writing to memory cells whose contents are larger, less than or equal to the argument).

The performance of the device and individual units is verified by testing and debugging of the developed VHDL codes.

The work has been done with the financial support of RFBR (Grant No. 16-07-00012 A).

## REFERENCES

[1] Kohonen T. 1982. Associative storage devices: translation from English. Moscow: Mir. p. 384.

[2] Hamaher K., Vraneshich Z., Zaki S. 2003. Computer organization. 5th edition. Translated from English by O. Zdir. SPb. Peter; Kiev. Publishing group BHV. p. 848.

[3] Tsilker B. Y., Orlov S. A. 2011. Organization of computers and systems: Textbook for universities. 2nd edition. SPb. Peter. p. 688.

[4] Tanenbaum E., Bos X. 2015. Modern operating systems. SPb: Peter. p. 1120.

[5] Martyshkin A.I., Yasarevskaya O.N. 2015. Mathematical modeling of the Task Managers for Multiprocessor systems on the basis of open-loop queuing networks. ARPN Journal of Engineering and Applied Sciences. 10(16): 6744-6749.

[6] Martyshkin A.I. 2016. Mathematical modeling of Tasks Managers with the strategy in space with a homogeneous and heterogeneous input flow and finite queue. ARPN Journal of Engineering and Applied Sciences. 11(19): 11325-11332.

[7] Martyshkin A.I. 2016. Development and research of open-loop models the subsystem "processor-memory" of Multiprocessor systems architectures UMA, NUMA and SUMA // ARPN Journal of Engineering and Applied Sciences. 11(23): 13526-13535.

[8] Salnikov I.I., Babich M. Yu., Butaev M.M., Martyshkin A.I. 2016. Investigation of the memory subsystem of information systems. The International Journal of Applied Engineering Research. 11(19): 9846-9849.

[9] Martyshkin A.I. 2015. The development of the hardware buffer memory of Multiprocessor system. // Fundamental Research. (12-3): 485-489.

[10] Martyshkin A.I. 2016. Functional organization and algorithms of operation of a hardware buffer memory of a Multiprocessor computer system // Fundamental Research. (12-3): 518-522.

[11] Martyshkin A.I. 2015. Implementation of the hardware memory buffer for Multiprocessor system. // Proceedings of the 12th International Scientific and Technical Conference "New Information Technologies and Systems". Penza: PSU. pp. 96-99.

[12] Ognev I.V., Borisov V. V. 2000. Associative environment. M.: Radio and Communication. p. 312.