Associative Co-processor on the Basis of Programmable Logical Integrated Circuits for Special Purpose Computer Systems

A.I.Martyshkin
Penza state technological University
PSTU
Penza, Russia
alexey314@yandex.ru

I.I.Salnikov
Penza state technological University
PSTU
Penza, Russia
iis@penzgtu.ru

D.V. Pashchenko
Penza state technological University
PSTU
Penza, Russia
dmitry.pashchenko@gmail.com

D.A Trokoz.
Penza state University
PSU
Penza, Russia
dmitriy.trokoz@gmail.com

***Abstract*—An option for implementing a co-processor for special-purpose computer systems on modern hardware components is offered in the article. The aim of the paper is to create an associative co-processor, based on the FPGA base for high-performance multiprocessor systems, the ones performing associative functions and data storage functions. Retrieval and sort operations are widely used both in users' and system programs. However, these operations are the most resource-intensive and time-consuming during the traditional implementation, when the necessary information is read in the following order: from the main memory of the computer system to the processor performing the corresponding operation with them. An associative co-processor, connected to a PCI Express bus of a specialized system that implements sort, retrieval, and "more-less" comparison operations immediately on 32 words, that are preloaded into associative memory, are considered in the article. The usage of the VHDL - hardware description language - gives the opportunity to obtain a flexible project, which can be easily corrected while checking the device operation. The operation study of the proposed device was carried out in the program Web pack ISE from Xilinx - one of the leading developers of FPGA. The efficiency of the proposed associative co-processor application is provided by performing time-consuming data retrieval, sort and comparison operations directly by the co-processor. It offloads the processor and improves the performance of the entire computer system. The main results obtained can be applied in various retrieval systems for various purposes: database servers, in the systems, used at railway stations, airports and for the efficient and rapid implementation of retrieval functions in operating systems.**

Keywords—associative storage device, addressing, associative co-processor, bus interface, computer system, hardware implementation, memory addressing, memory cell, multiple match analyzer, reaction memory, reading cycle, record cycle, top-priority analyzer

I. INTRODUCTION

Today one of the main applications of computers is processing of large amounts of data. The most time-consuming operations are likely to be searching and sorting. The memory addressing architecture is used in modern computer systems (CS), i.e. to search for the necessary data in memory one should read information in each memory address and compare it with the specified search argument. Searching for the necessary information takes a lot of time if we use such an algorithm. This, in its turn, negatively affects the speed of the whole CS. It is faster to provide an information access according to the content. The principle of this is as follows (Fig. 1): there is an array with N words. You are to search for all words beginning with the "A" character and ending with the "H" symbol. The search argument in this case is the word A *** N, where bits are marked with an asterisk. These bits do not affect the result. Speaking about the hardware, the memory array is built in such a way that at the output of the memory cells (MC), where the information matches with the inputted search argument, a signal appears recording the match. Then MCs are selected according to these signals.

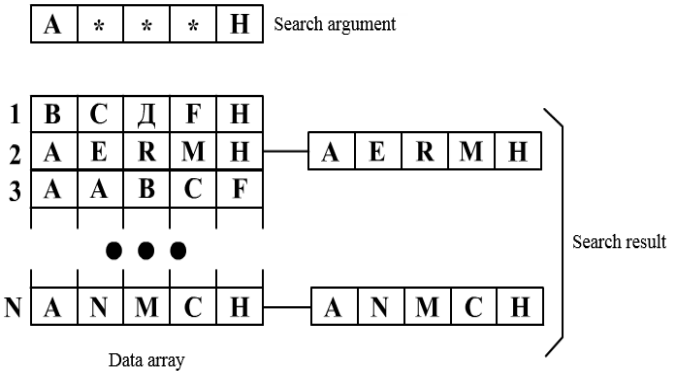

Fig. 1. The content addressable principle.

II. THEORETICAL ANALYSIS

When studying the subject area the authors analyzed some literature [1–9, 18] in order to find problems understudied. Some issues related to hardware implementation of a content addressable co-processor (coprocessor) for special-purpose computer systems were not covered in papers, but they were partially mentioned in works [10–20].

978-1-5386-7386-7/18/$31.00 ©2018 IEEE

The authors defined the purpose of the article. It aims at developing and studying a FPGA-based content addressable coprocessor for special-purpose computer systems. The issue under study is very relevant today due to the global informatization and the widespread processing of large amounts of various data.

The content addressable coprocessor proposed can perform an addressing and content addressable access to the data stored in the memory. The coprocessor has two parts: the main one performing the functions of the content addressable coprocessor, and coupling part with a PC forming the interface with the central processor (CPU).

Now it is known that a data content addressable access can be implemented by several ways, software and hardware being the main ones. It can be implemented as a parallel content addressable memory (CAM), where the search argument comes in parallel across all MCs. Then a comparison large-scale operation is performed, and the main result is as follows: the search is performed per one machine clock cycle. A variant of developing the hardware method of the data content addressable access is a serial bit content addressable memory (CAM), where the search is performed for each bit. Here, the search time depends greatly on the data bus width.

Processes being similar to biological mechanisms of storing and processing inputting data can be described via quite specific models of the content addressable memory (CAM) allowing to show associations of any complexity between objects. In addition, all these relationships can be implemented via a strictly ordered pair of O and V information objects and the A relation type: $O \xleftrightarrow{A} V$ [18]. One of the simplest models of the CAM for presenting such relationships is the mathematical model shown in Fig. 2, a, consisting of a content addressable medium that is connected with two input information channels and one output channel.

When recording input information is fed to the K input, and feature information is fed to the C input being the context in which the inputting information is stored in the memory. At the stage of a content addressable sampling, when the K key enters at the memory output, the R response is generated connected with the K key. Thus, the information stored in the memory can be selected using any of its fragments used as search engines. By specifying different C context, it is possible to specify more precisely the information to be sampled [3].

According to the CAM definition introduced there is an urgent problem concerning the organization of accumulating and searching for structured data. It should be provided in such a way that a data access would be possible on the basis of a content addressable sampling. Fig. 2, b shows the mathematical model of the CAM allowing to answer the questions: how can you record and select elements of structured data, and organize the circularity of the search process; where does the selected data element become the key for searching for a new piece of data [17, 18, 21]. The model considered implements the memory suitable for recording and selecting structured data.

Coupling of a content addressable coprocessor with a CS is achieved by several ways [19]: direct connection to the processor bus; connection via the USB interface; connection to the PCI Express bus. Let us define architecture and structure features based on the methods mentioned above of connecting the content addressable coprocessor to the CS. After having analyzed advantages and disadvantages of the methods being used for organizing the main part of the coprocessor and the coupling part with the CS, we decided to make the main part in the form of a parallel CAM as this method is the most efficient.

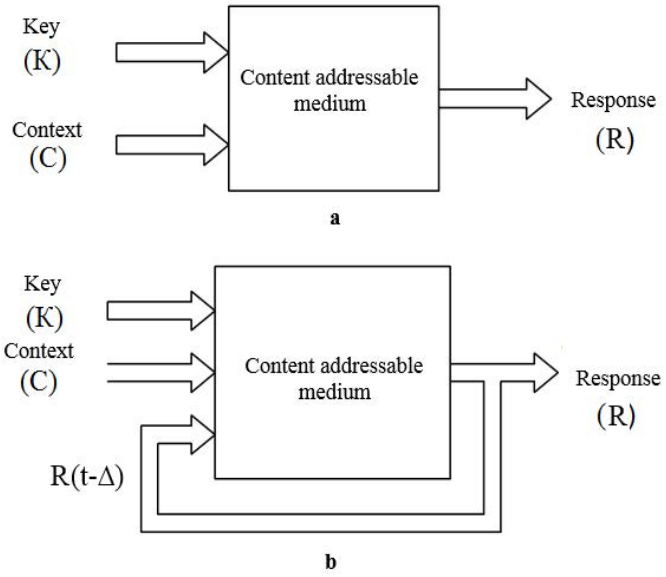

Fig. 2. The mathematical model with feedback (a) and without feedback (b)

Coupling with a special-purpose CS is organized on the PCI Express bus, because it has a quite sufficient bandwidth. It should also be mentioned that with such an organization the content addressable coprocessor will be placed in the I / O addressing space of the CS.

III. Content Addressable Coprocessor Structure Organization

The simplified structure of the content addressable coprocessor can be represented in the form of two blocks (Fig. 3, a): the coprocessor (the main part) and the PCI Express bus interface (the coupling part with the system). Having analyzed the selected method of implementing the coprocessor main part it has been found that the main unit is the memory module of the CAM. It is an array of memory cells (MCs) and it performs the functions of storing data and searching for matches with the argument. A MC consists of a storage element that performs data storage functions and a comparison diagram that performs the search, i.e. it generates signals pointing at matches or mismatches of the cell contents with the argument (Fig. 3, b).

To implement data write/read functions 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 of a particular MC. Also you should include the memory block for recording responses to keep the values of responding cells in the coprocessor module structure. The block of matches' counting is introduced into the coprocessor to count the number of responding cells. The multiple matches' analyzer (MMA) is necessary for the priority-oriented selection according to the

results of which the signals for the encoder block are generated. This block is necessary for converting the binary series to the address inputting at address selector inputs. To store the search argument in the module there is the argument register. The instructions decryption block controls the coprocessor operation. This block generates controlling signals. The mask register is excluded from the coprocessor structure, since the comparison circuit generates three signals ("EQUAL", "MORE" and "LESS") and there is no need to mask search argument bits (Fig. 3, c).

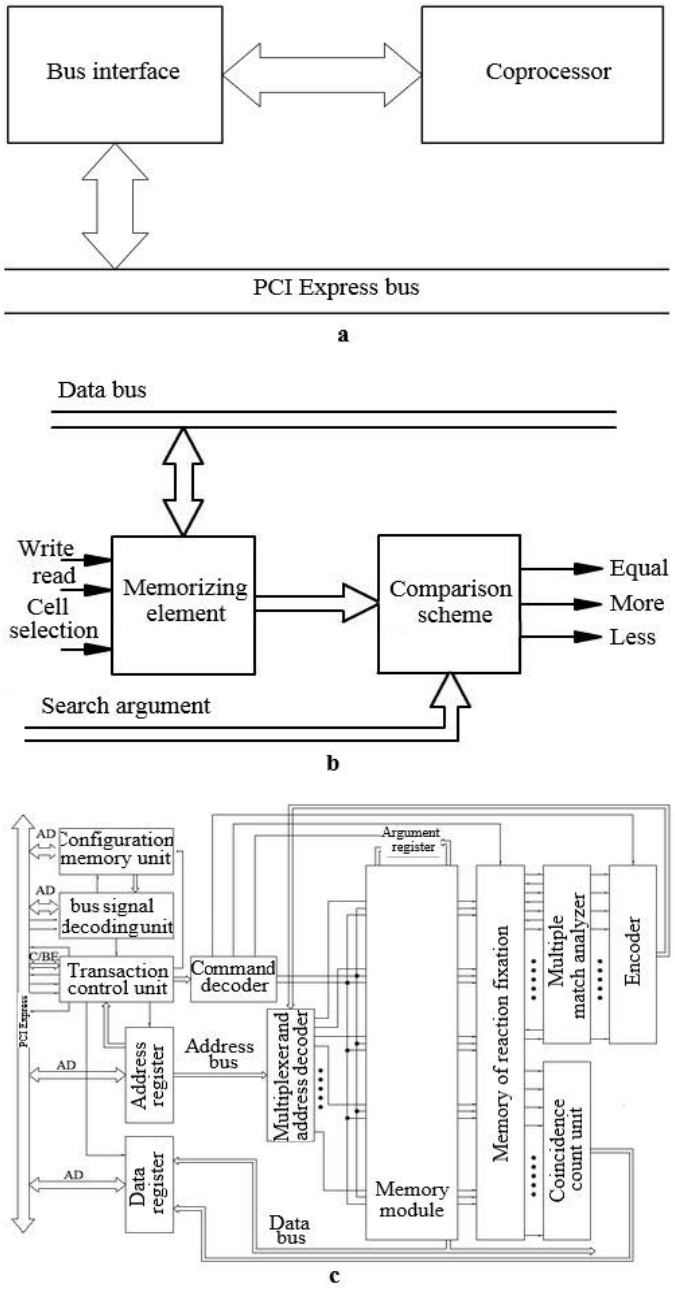

Fig. 3. The content addressable coprocessor simplified structure diagram (a); the memory cell structure (b); the content addressable coprocessor straight-line structure diagram (c)

IV. Content Addressable Coprocessor Functioning

The basis of the content addressable coprocessor is the memory block being an array of MCs. The memory of MCs is directly implemented by a parallel register being an array of D-flip-flops and providing the maximum performance and minimum logic required for the process of storing information. The main signals for the register are as follows: the D signal (32 bits) means that data is inputted, the CE signal and the Q signal (32 bits) mean that data stored in the register is read. The comparison diagram is made on the basis of a comparator. The main signals here are the A and B signals (32 bits each) receiving signals for comparison, = signals, <and> from which the comparison results are selected. The structure of a MC includes a buffer element which is necessary to disconnect the output data bus from the unified bus. The basic signals of the buffer element are the D signal (32 bits), the Q signal (32 bits) and the T signal. The circuit of MCs at the functional level is shown in Fig. 4, a. The functioning key point is as follows. The search argument inputs at the ArgI input. The data inputs in MCs via the DataI input, and using the DataO output they are read out of the MCs. The Write and CS inputs are required to control the operation of MCs. The MC is selected with the help of the CS signal, i.e. the BUFT buffer element allows signals from the RG register to pass to the DataO output bus. At the "1" signal on Write data from the DataI input is recorded in the RG. On the Equal, More and Less outputs signals "=", ">" and "<" are generated correspondently.

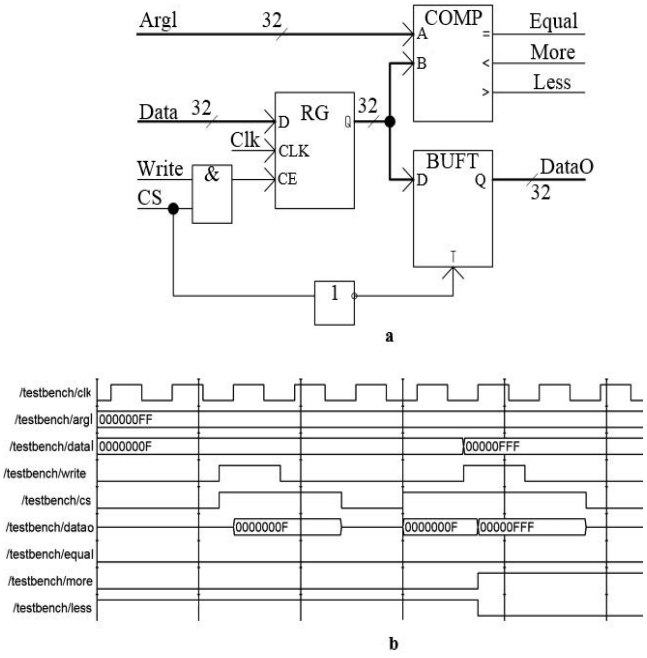

Fig. 4. The memory cell functioning diagram (a) and its operation temporary diagram (b)

The diagrams of the MCs functioning providing grounds for confirming its operability are shown in Figure 4, b. Here you can see a number of values: "F", "FFF" and "FF". The search argument is the "F" value. From these diagrams it is clear that while the CS signal is at logic zero, the (DataO) output bus is in a floating state. When the CS receives a logical one from the (DataO) output, the values stored in the register

(RG) can be read. The cell entry occurs when "1" is inputted to the Write input. Fig. 4, b shows that a new value is written in the MC and the search results are recorded at the Equal, More and Less outputs simultaneously.

A MMA can be implemented on the basis of a shift register (Fig. 5, a) and on the basis of a priority-oriented analyzer (Fig. 5, b). The main elements of a MMA based on a shift register are the looped shift register and the address counter. The search result for all MCs is recorded in the looped shift register (memory of responses recording). It should be noted that with such a sampling organization there is no need for an address coder.

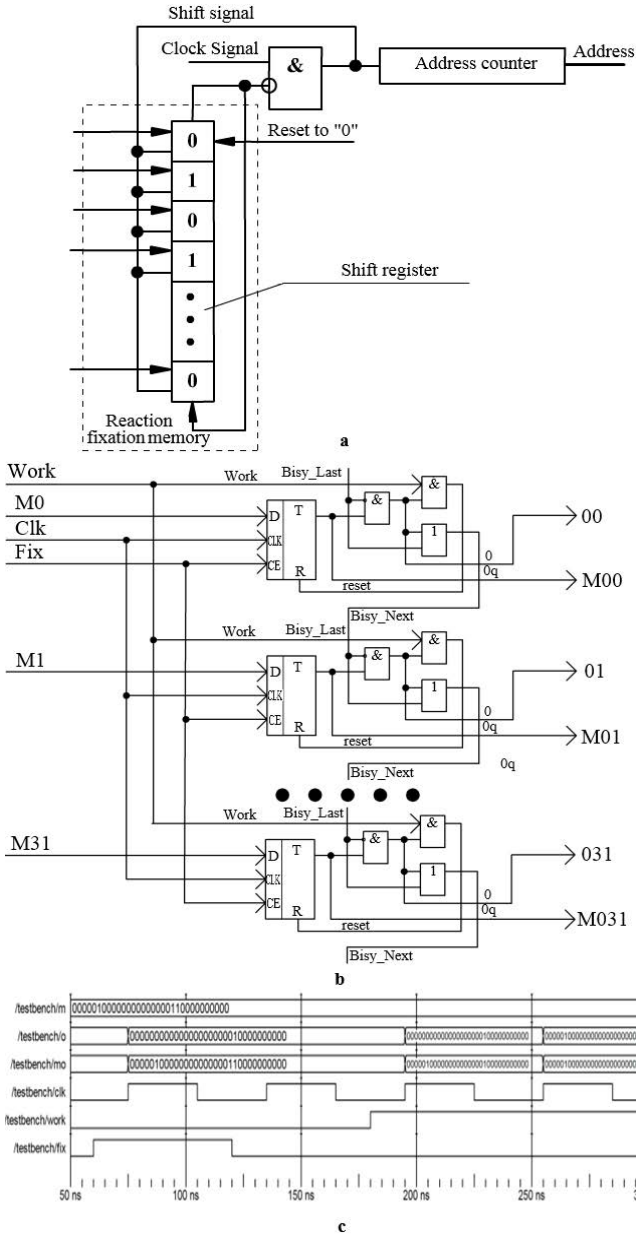

Fig. 5. The multiple matches' analyzer on the basis on the shift register (a) and on the basis of the priority-oriented analyzer (b); temporary diagrams of responses recording memory operation and the multiple matches' analyzer (c)

A MMA on a priority-oriented analyzer is built on D-flip-flops performing the memory functions of responses recording and combinational logic. This circuit operates on the Clk signal front. Search results in MCs are fed to the M0 ... M31inputs. At the Fix signal they are recorded on the D-flip-flops. The Work signal makes the operation of the MMA possible. The priority-oriented analyzer itself is a logical circuit allowing you to select a line with the lowest number among its inputs, which are set to "1". It is implemented in accordance with the bits-in-series principle. Each "1" input of this circuit blocks the effect from the lines with large numbers. As a consequence only the output being the first active line is set to one. The "1"signal corresponding to the first active line automatically passes to the output, and the reset of the first "responded" triggers corresponds to the reset signal function.

The block of matches counting can be realized in several ways: on the basis of the shift register and the counter and on the basis of cascades of totalizers.

The diagram of matches counting based on the shift register and the counter (Fig. 6, a) consists of the shift register (a series of ones and zeros corresponding to responded and non-responded cells is loaded in it), the counter (it counts the number), and the "logical AND" element (it controls the counting process). A clock signal is fed to the circuit.

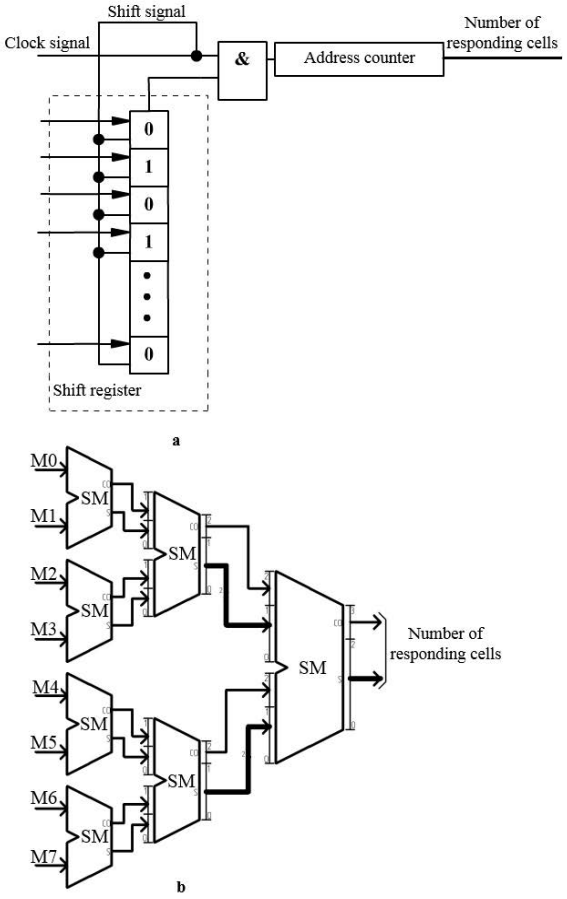

Fig. 6. The block of matches' counting on the basis of the shift register and counter (a) and on the basis of cascades of totalizers (b)

The shift register content is shifted one bit up. Depending on the value at the output of the shift register the value of the counter increases or does not increase, i.e. all register bits being equal to one are counted. The number of clock pulses required to count all the cells is equal to the number of all memory cells (in our case it is 32 clock cycles). In addition, this diagram requires an additional clock cycle for loading the shift register.

The circuit based on cascades of totalizers (Fig. 6, b) is a cascade of totalizers of different capacity, i.e. the input binary series is divided into groups of two and these groups are fed to the inputs of single-bit totalizers. Intermediate results obtained on the single-bit totalizer are also grouped in 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 diagram, this diagram is more quick-operating and in principle can count the number of cells per clock cycle, so the counter is implemented on the basis of cascades of totalizers.

The functioning accuracy of a content addressable coprocessor module was verified during the execution of a number of computer experiments. It has been experimentally proved that the usage of a content addressable coprocessor provides an increase in the performance of a CS by about 1/4 in comparison with systems using only classical processors. From this it is clear that a coprocessor should be used in practice.

V. CONCLUSIONS

The structure of a content addressable coprocessor is presented in the article. This structure differs from prototypes as it has been developed in the form of a functionally independent block in which algorithms have been implemented to perform time-consuming operations of searching and comparing data. In the traditional organization of such devices these operations were performed directly by the CPU. Based on the description of device blocks at the functional level the VHDL-code of the content addressable coprocessor has been developed and debugged.

The module mentioned in the paper can be implemented as a CS expansion board for connecting via the PCI Express connector. The device is implemented by hardware. It makes it possible to perform a number of time-consuming operations unloading the CPU and improving the CS performance as a whole.

The content addressable coprocessor described in the article performs the following functions:

1) address recording and address reading in the CAM;

2) search for data being equal, larger and less than search arguments;

3) content addressable reading (larger, less or equal to the argument);

4) content addressable recording (recording to memory cells which content is larger, less or equal to the argument).

Functionality and accuracy of the content addressable operation and its individual parts have been verified and confirmed by testing the developed modules using VHDL language.

REFERENCES

[1] T. Kokhonen, Content addressable memories: translation from English. Moscow: Mir Publishers, 1982.

[2] Lawrence Chisvin and R. J. Duckworth, “Content-addressable and associative memory: Alternatives to the ubiquitous RAM,” IEEE Computer, 22(7), pp. 51–64, 1989.

[3] T. B. Pei and C. Zukowski, “Putting routing tables in silicon,” IEEE Network Magazine, no. 6(1), pp. 42–50, 1992.

[4] I. N. Robinson, “Pattern-addressable memory,” IEEE Micro, no. 12(3), pp. 20–30, 1992.

[5] M. Ruiz-Sanchez, E. W. Biersack, and W. Dabbous, “A survey and taxonomy of IP lookup algorithms,” IEEE Network, no. 15(2), pp. 8–23, 2001.

[6] K. J. Schultz, CAM-Based Circuits for ATM Switching Networks, PhD thesis, University of Toronto, 1996.

[7] S. Stas, “Associative processing with CAMs,” In Northcon/93 Conference Record, pp. 161–167, 1993.

[8] D. Wright and M. Sachdev, “Tansistor-Level Fault Analysis and Test Algorithm Development for Ternary Dynamic Content Addressable Memories,” IEEE International Test Conference, pp. 39–47, 2003.

[9] N. Mohan, M. Sachdev, “Low Power Dual Matchline Ternary Content Addressable Memory,” Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), 2004.

[10] N. Mohan, M. Sachdev, “A Static Power Reduction Technique for Ternary Content Addressable Memories,” Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2004.

[11] W. W. Fung, M. Sachdev, “High Performance Priority Encoder for Content Addressable Memories,” Proceedings of the Micronet Annual Workshop 2004, Aylmer, Quebec, Canada, 2004.

[12] K. Pagiamtzis, A. Sheikholeslami, “Content-addressable memory (CAM) circuits and architectures: A tutorial and survey,” IEEE Journal of Solid-State Circuits, vol. 41, no. 3, pp. 712–727, 2006.

[13] E. Tanenbaum, H. Bos, Modern operating systems. St. Petersburg: Peter, 2015.

[14] A. I. Martyshkin, “A FPGA-based content addressable coprocessor module for special-purpose computer systems,” Bulletin of the Ryazan State Radiotechnical University, no. 58, pp. 75–82, 2016.

[15] A. I. Martyshkin, “Functional organization of a content addressable coprocessor module for special-purpose computer systems,” Models, systems, networks in economy, machinery, environment and society, no. 3(19), pp. 157–167, 2016.

[16] A. I. Martyshkin, “A FPGA-based content addressable coprocessor block for special-purpose computer systems,” Fundamental Sciences, no. 12–1, pp. 73–79, 2016.

[17] A. I. Martyshkin, “Basic operation principles of associate coprocessor module for specialized computer systems based on programmable logical integral schemes,” Journal of Fundamental and Applied Sciences, no. 10(6S), pp. 1449–1463, 2018.

[18] I. V. Ognev, S. A. Borisov, Content addressable media. Moscow: Radio and Communication, 2000.

[19] B. Ya. Tsil’ker, S. A. Orlov, Computers and computer systems organization (the 2-nd edition). St. Petersburg: Peter, 2011.

[20] I. I. Salnikov, M. Yu. Babich, M. M. Butaev, A. I. Martyshkin, “Investigation of the memory subsystem of information systems,” The International Journal of Applied Engineering Research, vol. 11, no. 19, pp. 9846–9849, 2016.

[21] A. I. Martyshkin, “Structural organization of the associative coprocessor unit on the basis of a programmable logic integral circuit for specialized computer systems,” Models, systems, networks in economics, technology, nature and society, no. 4(20), pp. 128–138, 2016