VOL. 13, NO. 17, SEPTEMBER 2018 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

PRINCIPLES OF FUNCTIONING OF THE ASSOCIATIVE COPROCESSOR MODULE FOR SPECIALIZED COMPUTER SYSTEMS BASED ON THE PLD

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

ABSTRACT

The main applications of computers today is working with large massive of data, where are all sorts of searches and sort the most labor-intensive operations. Existing computer systems used address memory architecture. In such an arrangement, to effect retrieval of data in the memory, it is necessary to make the reading of each memory module address and compare it with the search argument, resulting in finding the desired information in the memory takes a lot of computer time. This fact negatively affects the speed of a computer system as a whole. The paper considers the possibility of implementing a coprocessor module association on modern element base for specialized computer systems. The aim of this work is the development and research of associative co-processor module on PLDs, for specialized computing, for example, multi-processor, systems, performing associative function and data storage functions. The object of the development and research of this article is an associative co-processor based on PLD. In order to achieve the objectives used CAD Web pack ISE from Xilinx with the ability to create and simulate the operation of a device using the schematic editor, and in using VHDL language, which greatly facilitates the synthesis of projects for use on modern components - PLDs. Developed and debugged VHDL-code co-processor module association and its individual units. Functional and performance testing of devices tested in the CAD, obtained by the timing charts. The results of the study are obtained VHDL-code module of the associative co-processor, which is synthesized from broaching the file to configure the PLD.

Keywords: addressing, associative memory, block, bus interface, computer system, coprocessor, hardware implementation, memory cell, memory addressing, module, multiple match analyzer, read cycle, write cycle.

1. INTRODUCTION

Perhaps today is the most basic area of application of computing systems (CompS) and clusters is work with large amounts of data. The most laborious operations here are all sorts of searches and sorting of data. Existing CompS use the address memory architecture. This circumstance negatively effects of the speed of 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 described in [1, 2].

2. GOAL SETTING

This article as a whole is of a research. Literature sources were analyzed in the course of studying the subject area in order to find poorly worked questions [1, 3-7]. Some problem-oriented moments associated with the possibility of hardware implementation of the associative coprocessor module for fast data retrieval have not been adequately reflected in publications on this topic, but in part they have been covered and shown in [8-11].

The purpose of the article is to develop and study the functional organization of the associative coprocessor module based on PLD for specialized Comp S, for example, multiprocessor systems. This issue is topical today due to global informatization and the almost universal operation of huge amounts of data. To achieve this goal, the tasks are solved in determining the principles of the functional organization of the device. The developed 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 described device consists of two parts: the main one that implements the functions of the associative coprocessor (ordinary (address) write to the associative memory device (AMD), the associative record in the AMD, the normal reading from the AMD, the associative reading from the AMD, the search for coincidences), and a part of interface with the CompS performing the function of converting signals coming from the central processor (CPU) into signals with which the coprocessor will operate, that is this part of the device organizes the interface with the CPU.

Connection of the described device to the CompS is possible in the following ways [2]:

- direct connection to the CPU bus, while the coprocessor must be included in the system board;
- connection to the serial interface (USB); with this method of connection, the coprocessor must be made in the form of a separate module and be provided with a separate power supply;
- connection to the computer expansion bus (PCI); in this case the coprocessor will be executed in the form of an expansion board.

With direct connection to the CPU bus, the device must be included in the motherboard, which will lead to an increase in the cost of the coprocessor and its universality (this device will be designed to work with a certain class of CPU and certain architecture of the CompS). When connected using the USB interface, the coprocessor will look like a separate module, but it will work in sequence, which will lead to slow performance. Connecting the coprocessor to the PCI bus will allow it to be implemented as an expansion board. In this case, work with the module will be carried out on a parallel interface, which will achieve maximum performance. Having analyzed all the above-mentioned 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, because this method has the maximum performance. The connection to the system is realized via the PCI bus, because it has a sufficiently high throughput.

3. DESCRIPTION OF THE ASSOCIATIVE COPROCESSOR AT THE FUNCTIONAL LEVEL

The basis of the associative coprocessor under consideration is a memory module, which is an array of memory cells (MC). Thus, the development of a memory module is reduced to the design of a MC and combining them into an array. The memory element of the MC can be implemented on a parallel register, which is an array of D-flip-flops, providing the maximum performance and minimum logic required to implement the storage of information. The main signals for the register are the 32-bit D-signal, through which the data is fed into the register, the CE-signal and the 32-bit Q-signal, from which the stored data is read from the register. The comparison scheme can be 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 MC to disconnect the output data bus of the MC from the common output bus. The main signals of the buffer element are the 32-bit D-signal, the 32-bit Q-signal and the T- signal. If the T-signal is equal to logical zero, then the data from D is fed to Q. Otherwise, Q switches to the third state. The functional diagram of the MC is shown in Figure-1,a.

The search argument is supplied to the ArgI input. Through the DataI input, the data enters the MC. From the DataO input, data is read from the cell. The WRITE and CS inputs serve to control the operation of the MC. With the CS-signal, the cell is selected, i.e. the buffer element BUFT on the output bus DataO passes the signals from the output of the register RG. On a single WRITE signal, data from the DataI input is written to the RG register. The outputs Equal, More and Less form the signals "EQUAL", "MORE" and "LESS", respectively.

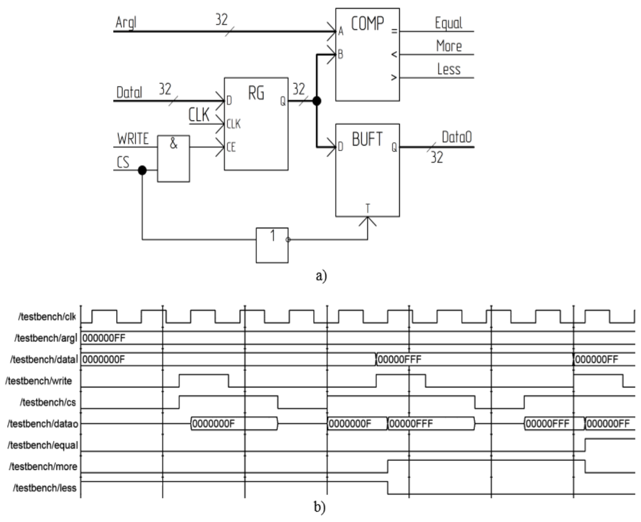

Figure-1. Memory cell: functional diagram (a); time diagrams of work (b).

The time diagrams of the MC's work, confirming its operability, are shown in Figure-1,b. Here is a write in the MC of a number of values: "F", "FFF" and "FF". The search argument is "F". As the figure shows, while the CS-signal 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, data stored in the register can be read from the data bus (DataO). The entry into the cell is done by sending a logical unit to the WRITE input. From the resulting time diagrams, we can see that as soon as a new value is written to the MC, the results of the search are set on the outputs Equal, More and Less.

The functional organization of the coprocessor block is considered in [10]. Here we describe in more detail the algorithms according to which some component parts of the associative coprocessor block and the device as a whole operate.

The command is sent to the coprocessor via a separate internal bus - the command bus, which is decoded and fed to the component blocks of the device. Coprocessor commands are listed in Table-1.

Table-1. Commands of an associative coprocessor.

Command	Command description
0000	readRAM - address reading
0001	WriteRAM - address writing
0010	WrArg - write of the search argument in the argument's register
0011	FixHit - command of fixation of responding cells
0100	rdAEqual - issuance of a cell address, the contents are equal of the argument
0101	rdAMore - issuance of a cell address, the contents are greater of the argument
0110	rdALess - issuance of a cell address, the contents are less of the argument
from 0111 to 1111	reserved

The command decoder is implemented on the basis of a decoder, the bus of commands is connected to the four-input input. The readRAM command performs the address read function, i.e. the address received on the address bus is decoded, a logical unit is fed to the CS-input of the corresponding MC and the contents of this MC enter the output data bus. The WriteRAM command performs an address record function. The address is decoded; logical units are fed to the input CS and WRITE of the corresponding MC. The signals from the data input line are fed to the input of the MC's register, i.e. memorization is carried out. Signals from the input data bus are input to the argument register by command WrArg. Upon FixHit command, Equal, More and Less signals are fixed in the memory of fixation reactions (MFR) and signals containing information about the number of NLs whose contents are equal to the argument are greater than the argument and less than the argument appear on the MFR. The command rdAEqual analyzes the contents of the MFR and the address of the responding cell receives the address of the first MC whose contents are equal to the argument. After that, the MC is reset to the MFR corresponding to the given address. The command rdAMore analyzes the contents of the MFR and the address of the responding cell receives the address of the first MC, the contents of which are greater than the argument. After that, the MC is reset to the MFR corresponding to the given address. The rdALess command analyzes the contents of the MFR and sends the address of the first memory location whose contents are less than the argument to the address of the responding cell. After that, the MC is reset to the MFR corresponding to the given address.

The configuration memory block is executed on the register, which is responsible for selecting the address space of the PCI device. The allocation of PCI I/O space addresses is shown in Table-2.

Table-2. Distribution of PCI I / O addresses.

AD[31:10]	AD[9:7]	AD[4:0]	Description	
			Reading	Writing
BAR+	000	00000–11111	Reading from the MCs of associative coprocessor	Write to the memory cells of the associative coprocessor
	001	xxxxx	Reading the number of responding cells	Write of the search argument in the argument register
	010	xxxxx	Reading from a MC whose contents are equal to the argument	Write to a MC whose contents are equal to the argument
	011	xxxxx	Reading from a MC whose contents are larger than the argument	Write to a memory cell whose contents are larger than the argument
	100	xxxxx	Reading from a MC whose contents are less than an argument	Write to a MC whose contents are less than the argument
	101	xxxxx	–	Fixation of responding cells in the memory of reaction fixation

AMD can be implemented on the basis of the shift register (Figure-2,a) and the priority analyzer (Figure-2,b). The main elements of the AMD based on the shift register are the looped shift register and the address counter. The result of searching all the MCs is recorded in this looped shift register (MFR). 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 moment, the clock signal is automatically blocked. If in the initial state in the counter zeroes were written, then at the end of the count, its contents directly show 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.

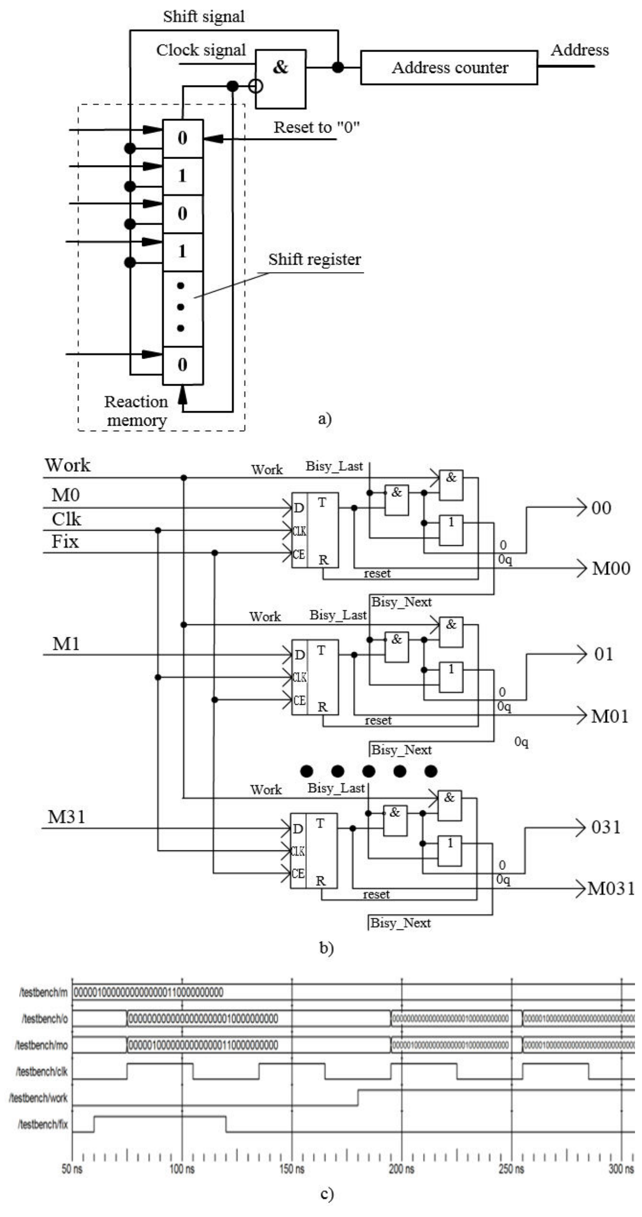

Figure-2. Analyzer of multiple coincidences based on the shift register (a) and on the basis of the priority analyzer (b); time diagrams of the work of memory fixation of reactions and the MMA (c).

The multiple much analyzer (MMA) based on the priority analyzer consists of D-flip-flops performing the functions of the MFR and combinational logic. This circuit operates on the edge of the CLK signal. The priority analyzer 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 to the unit, the signal from which automatically goes to the output, and the function of the signal "Reset" is reset of the first of the "answered" triggers. MFR and MMA are implemented on the basis of a priority analyzer scheme, because this circuit has a faster response time than a shift register-based scheme. The time diagrams of the circuit operation are shown in Figure-2, c.

The order of work with the block considered in the work is the following:

a) it is necessary to fill out the MC of AMD;

b) write the search argument in the argument register;

c) make a fixation of coincidences;

d) count the number of matches;

e) work with responding cells, i.e. read or write to these cells.

The block of the associative coprocessor is implemented on Xilinx PLDs. A project has been developed, consisting of four main modules that implement the main parts of the device: an associative coprocessor, a MFR and MMA, a description of the PCI interface and the interface of an associative coprocessor with it.

3. COMPUTATIONAL EXPERIMENT

The work of the associative coprocessor (Figure-3) begins with power on or with a hardware reset (the RST # signal is equal to logical zero), during which the BAR register and all device triggers are reset.

After reset, the device does not respond to access to I / O space. It starts configuring with the help of configuration write / read cycles. The IDSEL signal is set to a single value. Trigger T3 fixes access to the configuration memory. The device sets the DEVSEL and TRDY signals to zero, thereby confirming the readiness to receive / transmit data. Decodes the bus commands (CBE). If this is a configuration read command, the trigger T2 is set to a single state. From AD lines [7: 2] in the address phase, the address of the configuration memory register is read and decoded by the DCCS decoder. The decoded binary sequence is fixed to the RGCS register. In the case of configuration reading, a logical unit is sent from the zero-address register to the permissive input T of the BUFV buffer element and DEVICE ID and VENDOR ID ("01010101010101010101010101010101") are provided to the AD bus. In the case of configuration reading, a logical unit is sent from the register with address 10h to the permissive input T of the buffer element BUFB and the contents of the BAR register are output to the AD bus. If the command received via the CBE bus is a configuration record command, then the T1 trigger is set to a single state. As in the case of configuration reading, the address of the configuration memory register decodes the DCCS decoder. And by accessing the configuration memory register with address 10h, a logical unit is sent to the permitting input CE of the BAR register, the base address of the I / O area is written from the AD bus to the BAR register. Once the configuration is complete, the device is ready for work.

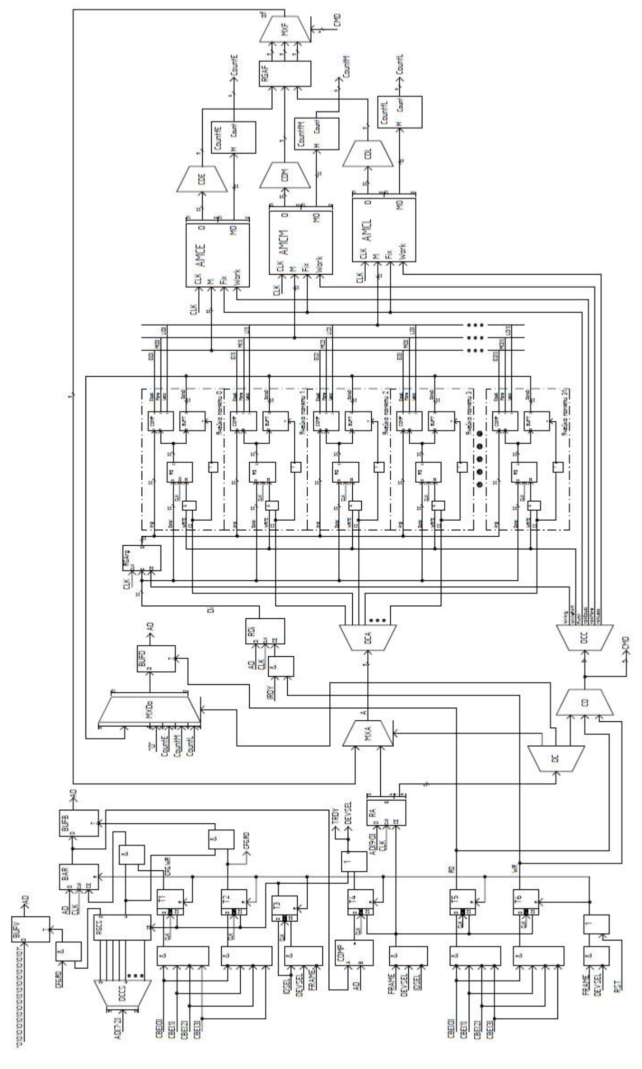

Figure-3. Functional diagram of the associative coprocessor.

The comparator COMP constantly compares the signals on the AD bus with the value written in the BAR register. If the signals on the AD bus and the value written in the BAR register are equal, a logical unit is fed to the input of the flip-flop T4, and if the signals FRAME and IDSEL are zero and the DEVSEL signal is one, then the trigger T4 fixes the unit at its input. The DEVSEL and TRDY signal is set to zero. Those. The device is ready to receive or transmit data. In the address phase, the AD register is fixed to the RA register [9: 0]. Based on the 9, 8 and 7 digits of the Q output of the RA register, the DC decoder and the CD encoder generate a command (CMD bus), which is deciphered to the DCC and enters the coprocessor blocks. Positions 0 through 4 of the Q address of the RA address register are input to the MXA multiplexer, which, depending on the internal command coming through the CMD (normal read / write commands), connects these signals to the input of the DCA decoder, which in turn decrypts the address and Generates a CS signal for the corresponding memory location.

With address (normal) reading from the MCs, the WriteRAM signal is zero. The CS signal generated by the decoder of the DCA signal for the corresponding cell allows the signals from the output of the register RG of the MC to pass to the output data bus (DataO). At other memory locations, at this time, BUFT buffer elements disconnect the register output from the data bus. Through the MXDo multiplexer and BUFD buffer element, data from the data bus is output to the AD bus. When the (usual) write to the MC with the decoder of the DCC command, the WriteRAM signal is transferred to a single state. The DCA address decoder generates a CS signal for the corresponding cell, and the signals from the AD bus through the register RDi are written to the register RG of the corresponding memory location. Asynchronously, the COMP comparators of all MCs compare the contents of the RG register to the argument and generate Equal, More and less signals for cases where the contents of the cells are equal, greater or less, respectively. And on the commit command (the FixHit command), these signals are fixed in the memory of the reaction fixation. The time diagrams of the operation of the associative coprocessor are shown in Figure-4,a-d). In Figure-4,a diagrams of configuration cycles are shown. In the first cycle, which takes 2 cycles, configuration reads from the configuration MC with address 0h. On the CBE bus, the configuration read command "1010" is transmitted, in hexadecimal notation it is A. The address of the configuration memory cell ("0") is transmitted via bits 7-2 of the AD bus. The DEVSEL and TRDY signals are set to zero, confirming the device is ready for operation. Next, in the next cycle, the data phase begins, and the device issues a DEVICE ID and a VENDOR ID is "55555555h". In the second cycle, which also takes 2 cycles, a configuration entry is made to the configuration MC with address 10h (the cell responsible for the base address of the I / O area). In it we write 5C00h. In the future, until the next reset or writing of another value to the address register of the I / O area (BAR), the device will respond to accessing I / O ports with addresses 5C00 - 5FFF. Since the base address is used with 31 to 10 bits, the rest are used to access the nodes of the device.

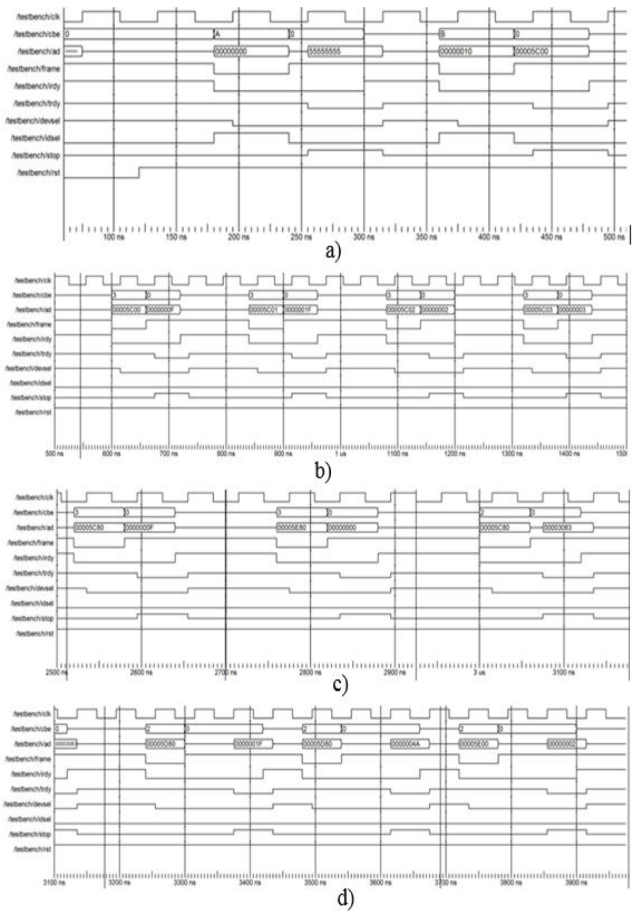

Figure-4. Time diagrams of the operation of the associative coprocessor: configuration of the device (a); writing to MCs (b); entry into the search argument, fixing the matches and reading the number of matches (c); reading from memory cells whose contents are greater and less than the argument (d).

In Figure-4,b there are diagrams of recording cycles in the associative memory (AM) cells. The cycle consists of the address phase and the data phase. As you can see from the diagrams, you need two measures to write to the device. In the first clock cycle (address phase), the address of the device ("00000000000000000010111xxxxxxxxxx") is transmitted to bits ADB-1 through 10. In bits 9 - 7 of the AD bus, the address of the node of the device ("000" - address reference to the MCs) is transmitted, and the address of the AD cell ("00000", "00001", "00010", "00011" "," 00100 "," 00101 "," 00110 "," 00111 "). Thus, in order to access the MC with address 5h on the AD bus, the address must be set to "00000000000000000101110000000101" (Figure-4,a). In the first measure on the CBE bus, the command to write to the input / output port ("0011") is also sent. In the second step, data is received via the AD bus. The cell with address 0h records "0000000F", in the cell with the address 1h - "0000001F", in the cell with the address 2h - "00000002", in the cell with the address 3h - "00000003" (Figure-4,b); In the cell with the address 4h - "0000004A", in the cell with the address 5h - "0000000F", in the cell with the address 6h - "0000000F" and in the cell with the address 7h - "0000000A".

In Figure-4,c shows the cycles of writing the argument, fixing the matches and reading the number of matches. The loop of writing a search argument is no different from writing to a memory location. It also requires two measures. In the address phase, an address consisting of the base address ("00000000000000000010111xxxxxxxxxx") and the node address ("001") is transmitted. The address of the memory location is ignored. In the argument register write "0000000F". Thus, we have 3 cells whose contents are equal to the argument, 2 cells - more than the argument and 3 cells - less than the argument (Figure-5,b).

Base device address	Device node address	Reserve	Memory address
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1	0 0 0	0 0 0	0 0 1 0 1

a)

Argument	0000000F	
Cell №0	0000000F	Equals argument
Cell №1	0000001F	More argument
Cell №2	00000002	Less argument
Cell №3	00000003	Less argument
Cell №4	0000004A	More argument
Cell №5	0000000F	Equals argument
Cell №6	0000000F	Equals argument
Cell №7	0000000A	Less argument

b)

Figure-5. Operation of the coprocessor: addressing of the device (a); the contents of associative memory (b).

The coincidence loop starts at address 5E80h. This address was obtained as a result of combining the base part of the address (it is, as in all other cases, "00000000000000000010111xxxxxxxxxx"), the address of the device node is "101" and the address of the MC that does not participate in the coincidence loop. Fixing of coincidences occurs only at the command of writing to the input / output port ("0011"). The value passed in the data phase does not matter, since it is not taken into account anywhere. This value is transmitted because the PCI bus transaction must have at least one data phase. The cycle of reading the number of matches begins with the address of the identical cycle of writing the argument. The difference is that the number reading cycle starts with a read command from the I / O port, and the write cycle of the argument is from the write command to the I / O port. In the data phase, the value of the number of elements equal to the argument (AD [17:12]), greater than the argument (AD [11: 6]) and less than the argument (AD [5: 0]) is read. In our case this value is 3083h ("000011 000010 000011"), i.e. The number of MCs whose contents equals the argument is 3. The number of MCs whose contents are larger than the argument is 2. The number of MCs whose contents are less than the argument is 3.

In Figure-4,d there are cycles of reading MCs, the contents of which are larger and smaller than the argument. One cycle of reading requires three measures. This is due to the fact that the clock is required to read the address of the responding cell. Those. In the first measure the address consisting of the base address and the address of the node is transmitted (for reading cells whose contents are greater than the argument, it is "011", for cells whose contents are larger than the argument, it is "100"), the DEVSEL signal is set to zero. In the second measure, the address of the responding cell is read. In the third clock, the read data reads the data, and the TRDY signal is set to zero, indicating that the target device is ready to output data.

The hardware module under consideration is implemented on Xilinx PLDs. The VHDL codes of the device are developed, the description of which consists of four modules, including the main blocks of the implemented device: an associative coprocessor, a reaction memory and a multiple coincidence analyzer, a description of the PCI interface, and the interface of the associative coprocessor with it.

4. CONCLUSIONS

The paper discusses the principles of functioning of the associative coprocessor block based on PLDs. Based on the description of the associative coprocessor at the functional level, a VHDL device code was developed, followed by a firmware file for configuring the PLD. The performance of the device and individual units is verified by testing and debugging of the developed VHDL codes.

The data bus width, described in the operation of the coprocessor, is 32 bits (a double machine word). The capacity of the AM array is 32 double machine words. The associative coprocessor is implemented in hardware, which allows performing laborious search and comparison operations, thereby unloading the CPU and increasing the overall performance of the CS.

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

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] Martyshkin A.I. 2016. Block of the associative coprocessor based on FPGA for specialized computer systems // Fundamental research. (12-1): 73-79.

[9] Martyshkin A.I. 2016. The module of the associative coprocessor based on FPGA for specialized computer systems. Bulletin of the Ryazan State Radio Engineering University. (58): 75-82.

[10] Martyshkin A.I. 2016. Functional organization of the associative coprocessor module for specialized computer systems. Models, systems, networks in economics, technology, nature and society. (3) (19): 157-167.

[11] Martyshkin A.I. 2016. Structural organization of the block of the associative coprocessor on the basis of a programmable logic integrated circuit for specialized computer systems. Models, systems, networks in economics, technology, nature and society. (4) (20): 128-138.