EasyChair Preprint

№ 2989

The Minimizating of Hardware for Implementation of Pseudo-LRU Algorithm for Cache Memory

Vadym Puydenko and Vyacheslav Kharchenko

March 19, 2020

The 11h IEEE Conference on Dependable Systems, Services and Technologies, DESSERT'2020
Ukraine, Kyiv, May 14-18, 2020

The Minimizating of Hardware for Implementation of Pseudo-LRU Algorithm for Cache Memory

Vadym Puydenko
dept. Computer Ingineering
Kharkiv Radio Technical College
Kharkiv, Ukraine
VAPuydenko@gmail.com

Vyacheslav Kharchenko
dept. Computer Systems, Networks and Cybersecurity
National Aerospace University "KhAI"
Kharkiv, Ukraine
V.Kharchenko@csn.khai.edu

***Abstract*—Synthesis of logic LRU of the internal cache memory of the central processing unit is fulfilled in case of hit, miss and filling in the lines for unit of data of the internal cache memory. The logic models are analyzed by selecting the lines including unreliable multitude for filling and controlling logic for substitution of the lines while it was true the selected multitude by unit of data. The minimizing functions that are simple completely defined and composite not completely defined is carried out. They are switched as: L=f(B) by selecting multiple among reliable values and B^+=f(L,B) forming the values of bits for unit LRU considering the previous state. As a result the minimum discrete realization has been obtained by suggested hardware solutions substitution policy for the algorithm of pseudo-LRU of internal associative cache memory and associative translation look –a – side buffer.**

Keywords—algorithm pseudo-LRU, controlling logic, internal memory cache, translation look–a-side buffer, LRU unit.

I. INTRODUCTION

Over the last 40 years of the evolution for architecture processor cores the necessary component of the internal associative cache memory and associative the of translation look-a-side buffer of the module of LRU with the algorithm of pseudo-LRU of substituting for the lines multitude of unit of data. This module has a significant impact on characteristics of high-speed performance, complexity and energy consumption for integrated devices. According to research works on the substituting for the lines multitude by the algorithm pseudo-LRU there are conclusions about advantages of politics substituting LRU, namely: the best productivity as compared to the productivity of algorithms of casual substitution FIFO and LFU [1,2], high efficiency realization of LRU [3], optimal politics of substituting for pseudo-LRU [4-7] accompanied with general of algorithm, structural, functional schemes and program VHDL codes for the unit of controller LRU, where attention is not paid simplifying and evaluating hardware for algorithm realization. In the presented work the hardware solution of the LRU unit are synthesized according to the algorithm pseudo-LRU. On the basis of the model will be formed certain basic for computer modelling. The purpose is granting hardware to assure competitive characteristics, such as simplicity, productivity, energy consumption.

II. ARCHITECTURE OF THE INTERNAL MEMORY CACHE

Internal cache memory of level L1 (Fig. 1) stores the copies of the last instructions, operands and other data have been read.Iit is possible to apply to cache memory at every synchronization time. It works with physical addresses, that minimizes a number of times clearing of cache memory. Cache memory has to 8 directed associative by a multitude organization. The unit of data of 32 кbytes is divided into eight directions (Fig. 1), each of them has 64 64-bytes multitudes or lines of cache memory. Addressing of cache memory is carried out by a division low 36 bits of physical address on three parts. Six bits of the field of index determine the number of multitude from 64. High 24 bits are the field to the tag (signs): these bits are compared to the tags of each line of the index multitude and show whether the 64-bytes line of cache memory is kept according to the physical address. The low six bits of physical address elect a byte inside of the line of cache memory. The field of six bits, that is located in the unit of reliability, shows whether the data are present in current time period according to certain physical address, they are cached.

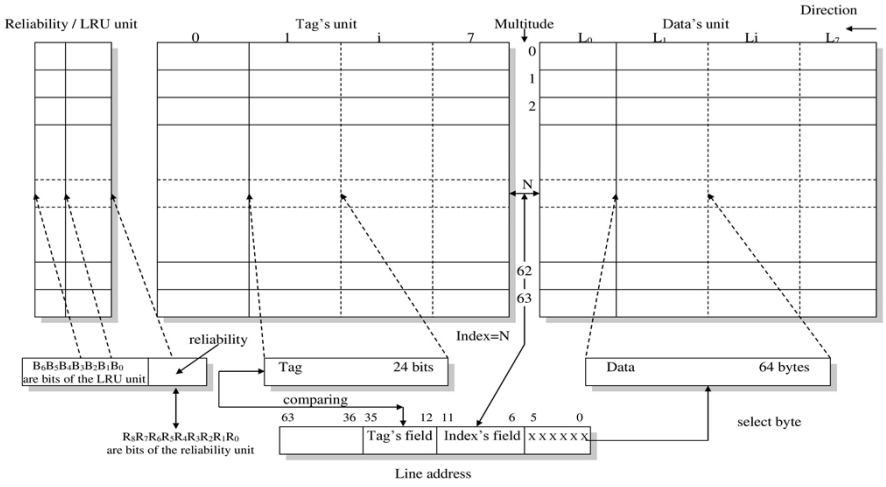


Fig. 1. Architecture of the internal memory cache

III. The Model of the Synchronous Digital Automaton for Representation Substitution Politics by the Algorithm Pseudo - LRU

Presented model of the synchronous digital automaton (Fig.2) consists of combination schemes CS_1, CS_2 and D-triggers and is described by one complex switching function due to that the great number of simple switching functions $L_{q-1}(\mu)$,…, $L_0(\mu)$ replaces the input signals vector of the automaton $v = \{x_{a-1}…x_0\}$ and is simultaneously the output state of the automaton and thus we deal automatically with the original synchronous digital automaton of Moore - closed digital system with memory cells on the base of synchronous D-triggers with predicted combinational schemes, where: $\mu = \{B_{m-1}…B_0\}$ - multitude of the internal states; $v(t_д) = z(t_д)=\{L_{q-1}(\mu)$,…, $L_0(\mu)\}$ - multitude of the output states; $B_r = B_r(t_д)$, $B^+_r = B^+_r(t_д+1)$, r=0..m-1 - the value of the complex switching function f_r of updating internal state; f_r – complex switching function of updating internal state; $L_{q-1}…L_0$ – simple switching functions of selection directions L_i; $t_д$ – current discrete moment of time; $t_д+1$ – next discrete moment of time; $q=2^n$ – number of directions in a cache memory; n – natural number > 1; m=q – 1=D - number of synchronous D-triggers; $B^+_r = f_r (B_r, L_{q-1}(\mu)$,…, $L_0(\mu))$ - general view of the complex switching function.

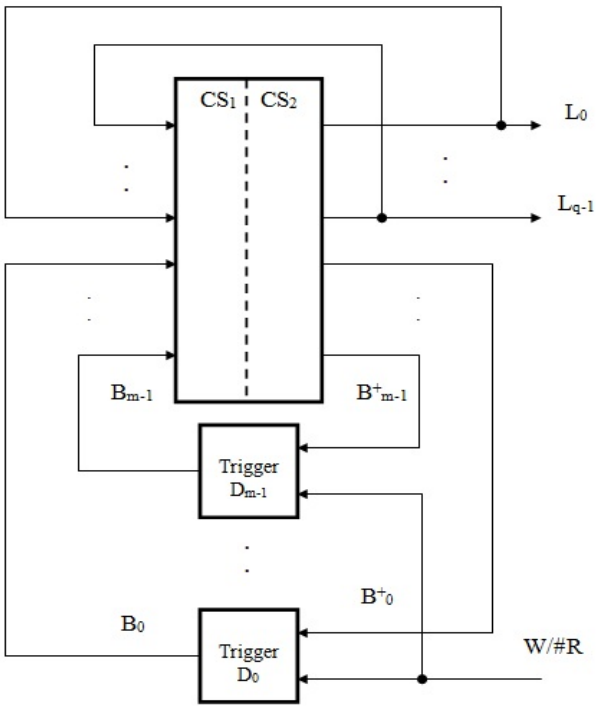


Fig. 2. Model of the synchronous digital automaton

IV. Synthesis of Automaton for Substitution Politics by the Algorithm Pseudo-LRU

A. The algorithm

Let's identify lines in multitude through L_0, L_1, L_2, L_3, L_4, L_5, L_6 and L_7. Every multitude in the unit LRU corresponds to seven bits of B_0, B_1, B_2, B_3, B_4, B_5 and B_6, that are modified at every hit or miss as follows (Fig. 3):

- if the last access in multitude is to the line of L_0 or L_1 or L_2 or L_3, then bit is B_0=1, else to the line of L_4 or L_5 or L_6 or L_7 - B_0=0.
- if the last access in multitude is to the line of L_0 or L_1, then bit is B_1=1, else to the line of L_2 or L_3 - B_1=0.
- if the last access in multitude is to the line of L_4 or L_5, then bit is B_2=1, else to the line of L_6 or L_7 - B_2=0.
- if last access in the pair of L_0 - L_1 is to the line of L_0, then bit is B_3=1, else to the line of L_1 - B_3=0.
- if last access in the pair of L_2 - L_3 is to the line of L_2, then bit is B_4=1, else to the line of L_3 - B_4=0.
- if last access in the pair of L_4 - L_5 is to the line of L_4, then bit is B_5=1, else to the line of L_5 - B_5=0.
- if last access in the pair of L_6 - L_7 is to the line of L_6, then bit is B_6=1, else to the line of L_7 - B_6=0

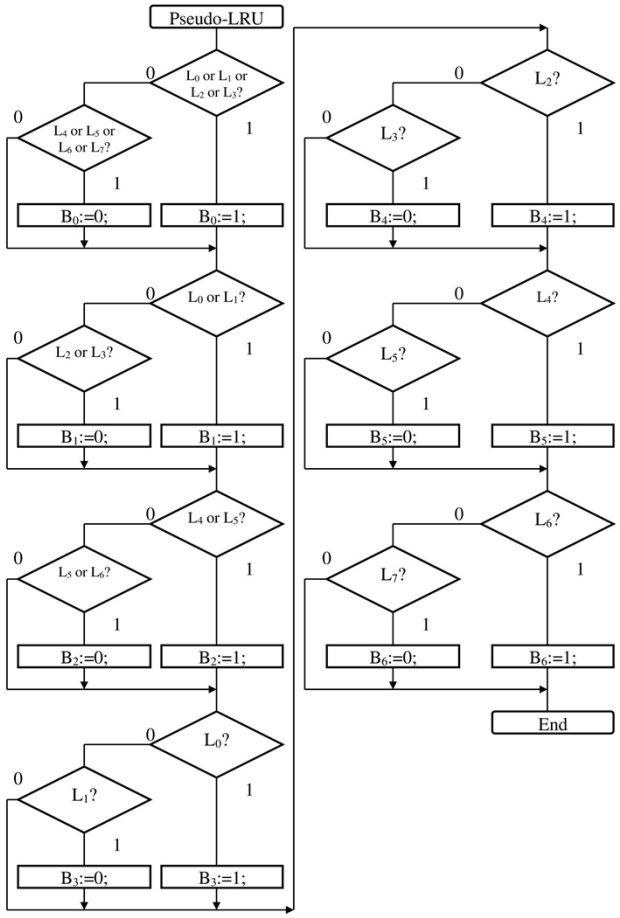


Fig. 3. The algorithm pseudo-LRU of substitution politics for associative cache memory to 8 directions

B. Program table of full state for bits $B_6B_5B_4B_3B_2B_1B_0$ of the LRU unit

Corresponding with the logic of the algorithm pseudo-LRU specially created program code will give the full table of the truth of states for bits $B_6B_5B_4B_3B_2B_1B_0$ which carrying information about the least recently used L_i line (Fig. 4).

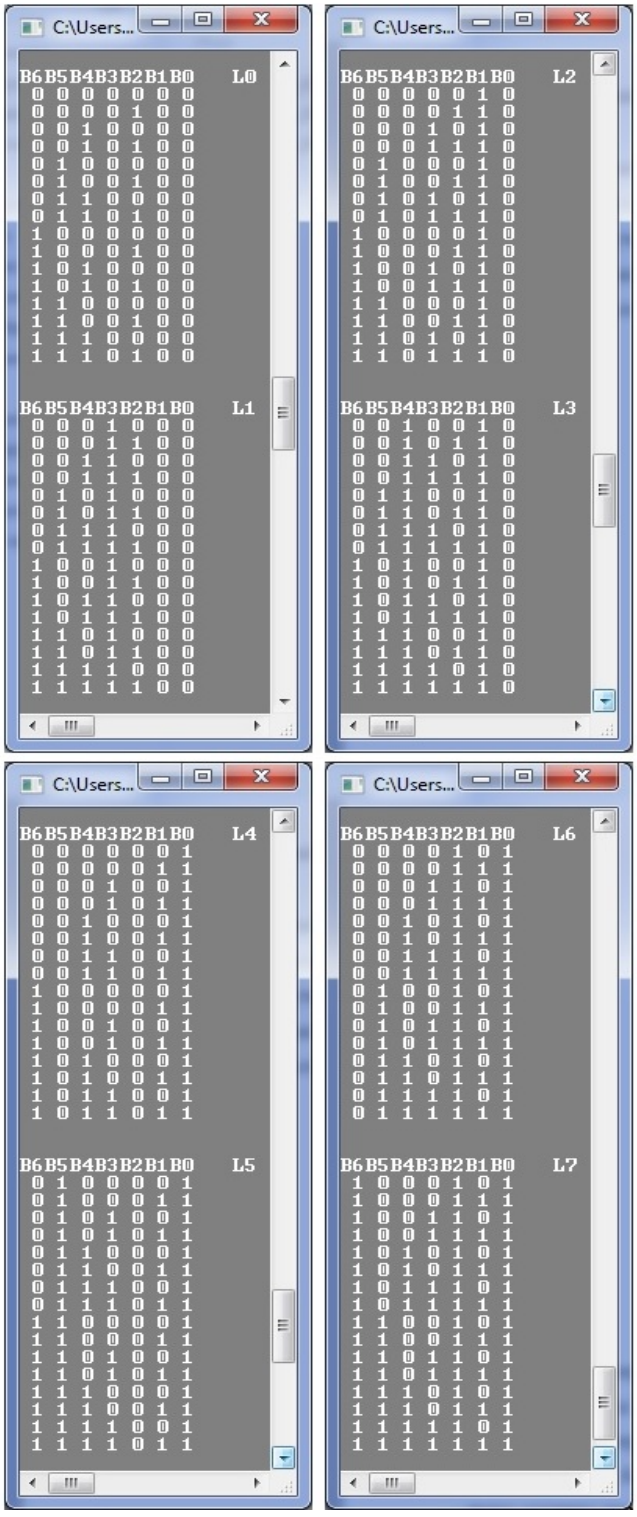


Fig. 4. Results of execution of the specially created program code of analysis of states of the bits B_0, B_1, B_2, B_3, B_4, B_5, B_6 witch define the least recently used the line L_i among lines L_0, L_1, L_2, L_3, L_4, L_5, L_6 L_7 of a mutitude block's data of the cache memory

C. The Karnaugh's diagram for minimazation of the simple completely defined switching function L=f(B)

Corresponding with the results on Fig. 5 to the purpose of receiving the minimum logical equations of simple completely defined function L=f(B) for selection of lines L we will make the corresponding diagram of Karnaugh on seven terms (Fig. 6).

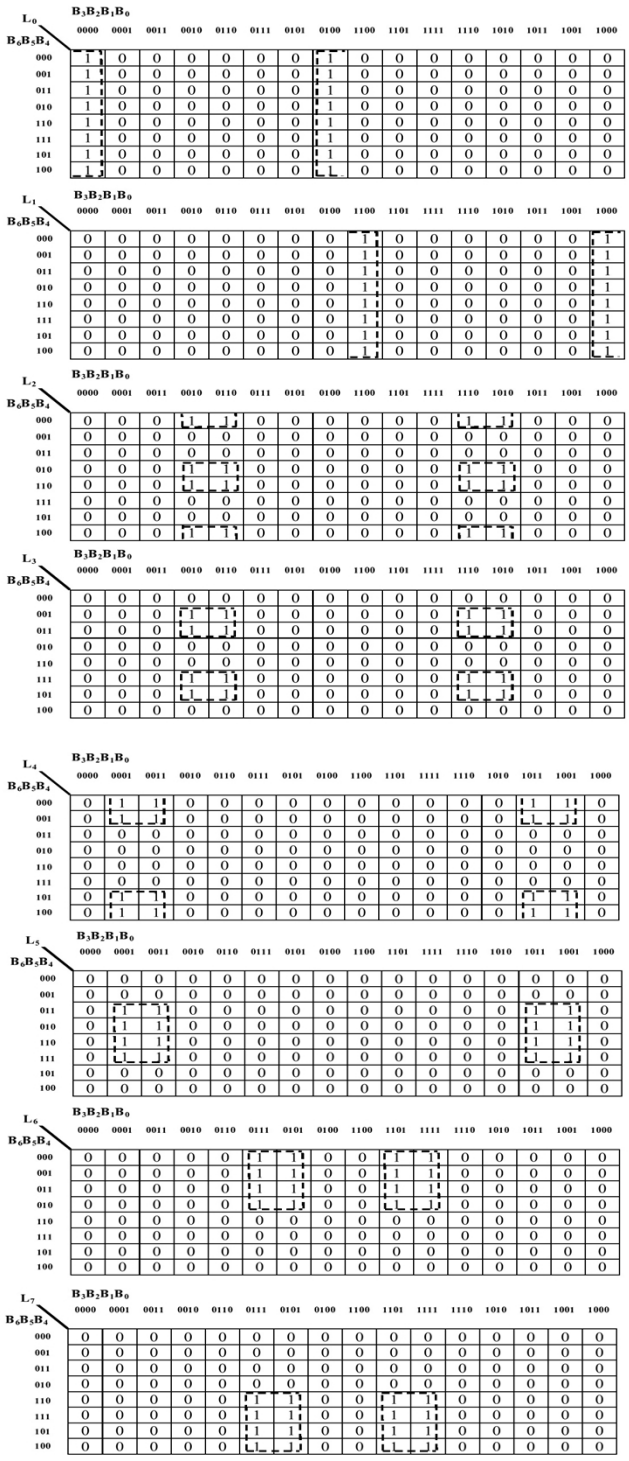
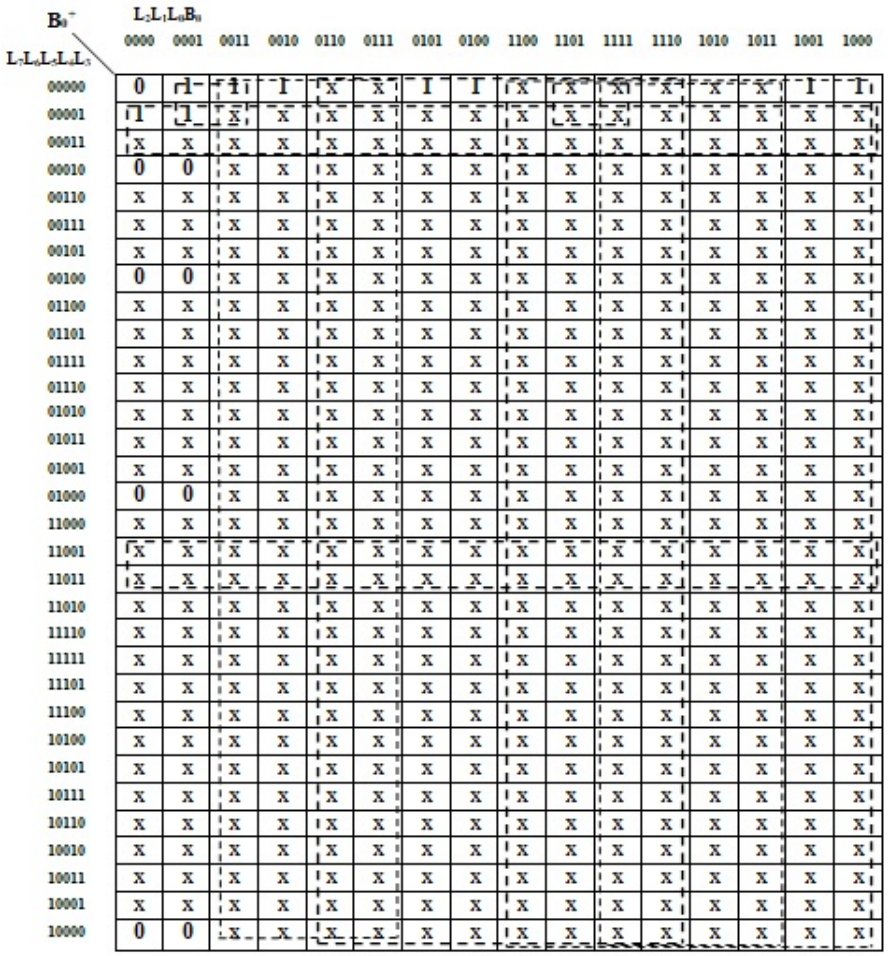


Fig. 5. The Karnaugh's diagram on eight terms for receiving the minimum logical equations of simple completely defined switching function L=f(B) for selection lines L_0, L_1, L_2, L_3, L_4, L_5, L_6 L_7 of a mutitude block's data

At event of hit or miss the bits B_0, B_1, B_2, B_3, B_4, B_5, B_6 of the LRU unit change own state corresponding with substitution politics by the algorithm pseudo-LRU. On the basis of above given let's make the diagram of Karnaugh on nine, five and three terms for receiving the minimum logical equations of simple completely defined function B^+=f(L,B) (Fig. 6).



B_0^+ \ $L_2L_1L_0B_0$	0000	0001	0011	0010	0110	0111	0101	0100	1100	1101	1111	1110	1010	1011	1001	1000
$L_7L_6L_5L_4L_3$ 00000	0	1	1	1	x	x	1	1	x	x	x	x	x	x	1	1
00001	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00011	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00010	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00110	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00111	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00101	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
00100	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01100	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01101	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01111	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01110	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01010	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01011	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01001	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
01000	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11000	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11001	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11011	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11010	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11110	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11111	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11101	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11100	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10100	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10101	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10111	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10110	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10010	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10011	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10001	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10000	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x

B_1^+ \ $L_1L_0B_1$	000	001	011	010	110	111	101	100
L_3L_2 00	0	1	1	1	x	x	1	1
01	0	0	x	x	x	x	x	x
11	x	x	x	x	x	x	x	x
10	0	0	x	x	x	x	x	x

B_2^+ \ $L_5L_4B_2$	000	001	011	010	110	111	101	100
L_7L_6 00	0	1	1	1	x	x	1	1
01	0	0	x	x	x	x	x	x
11	x	x	x	x	x	x	x	x
10	0	0	x	x	x	x	x	x

B_3^+ \ L_0B_3	00	01	11	10
L_1 0	0	1	1	1
1	0	0	x	x

B_4^+ \ L_2B_4	00	01	11	10
L_3 0	0	1	1	1
1	0	0	x	x

B_5^+ \ L_4B_5	00	01	11	10
L_5 0	0	1	1	1
1	0	0	x	x

B_6^+ \ L_6B_6	00	01	11	10
L_7 0	0	1	1	1
1	0	0	x	x

Fig. 6. The Karnaugh's diagram of composite not completely defined switching function $B^+=f(L,B)$ for changing values of B_6, B_5, B_4, B_3 B_2, B_1, B_0 bits of LRU unit

D. The logical model

At the above given diagrams of Karnaugh the conjunctive terms, which was found allow to receive the minimum logical equations of simple completely defined and composite not completely defined switching functions L=f(B) and B^{+}=f(L,B). These minimal logical equations (1) – (15) are describe the discrete mathematical model of the LRU unit for all multitudes block's data:

$$L_0 = \overline{B_3} \,\&\, \overline{B_1} \,\&\, \overline{B_0} \quad (1)$$

$$L_1 = B_3 \,\&\, \overline{B_1} \,\&\, \overline{B_0} \quad (2)$$

$$L_2 = \overline{B_4} \,\&\, B_1 \,\&\, \overline{B_0} \quad (3)$$

$$L_3 = B_4 \,\&\, B_1 \,\&\, \overline{B_0} \quad (4)$$

$$L_4 = \overline{B_5} \,\&\, \overline{B_2} \,\&\, B_0 \quad (5)$$

$$L_5 = B_5 \,\&\, \overline{B_2} \,\&\, B_0 \quad (6)$$

$$L_6 = \overline{B_6} \,\&\, B_2 \,\&\, B_0 \quad (7)$$

$$L_7 = B_6 \,\&\, B_2 \,\&\, B_0 \quad (8)$$

$$B_0+ = \overline{\overline{L_5} \,\&\, \overline{L_7} \,\&\, \overline{L_6} \,\&\, \overline{L_4} \,\&\, B_0 \,\&\, \overline{L_3} \,\&\, \overline{L_2} \,\&\, \overline{L_1} \,\&\, \overline{L_0}} \quad (9)$$

$$B_1+ = \overline{\overline{L_3} \,\&\, \overline{L_2} \,\&\, B_1 \,\&\, \overline{L_1} \,\&\, \overline{L_0}} \quad (10)$$

$$B_2+ = \overline{\overline{L_7} \,\&\, \overline{L_6} \,\&\, B_2 \,\&\, \overline{L_5} \,\&\, \overline{L_4}} \quad (11)$$

$$B_3+ = \overline{\overline{L_1} \,\&\, B_3 \,\&\, \overline{L_0}} \quad (12)$$

$$B_4+ = \overline{\overline{L_3} \,\&\, B_4 \,\&\, \overline{L_2}} \quad (13)$$

$$B_5+ = \overline{\overline{L_5} \,\&\, B_5 \,\&\, \overline{L_4}} \quad (14)$$

$$B_6+ = \overline{\overline{L_7} \,\&\, B_6 \,\&\, \overline{L_6}} \quad (15)$$

E. The hardware solution and computer model

The hardware solution for the substitution politics implementation by the algorithm pseudo-LRU is presented on Fig.7. The computer model is built to produce the minimized hardware for further research in the environment of computer modelling of EWB512 (Fig.8):

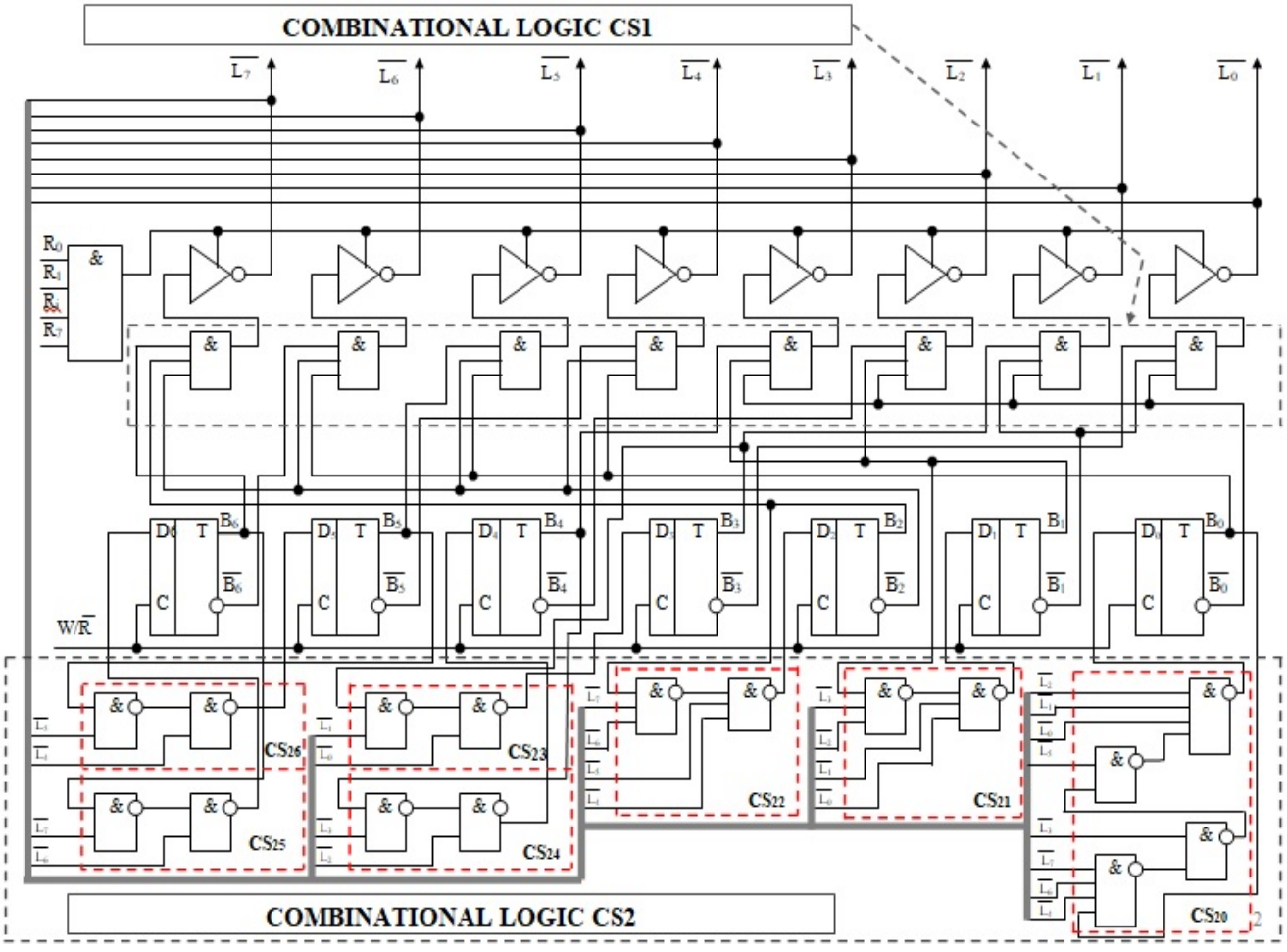


Fig. 7. The hardware solution of the substitution politics implementation by the algorithm pseudo-LRU

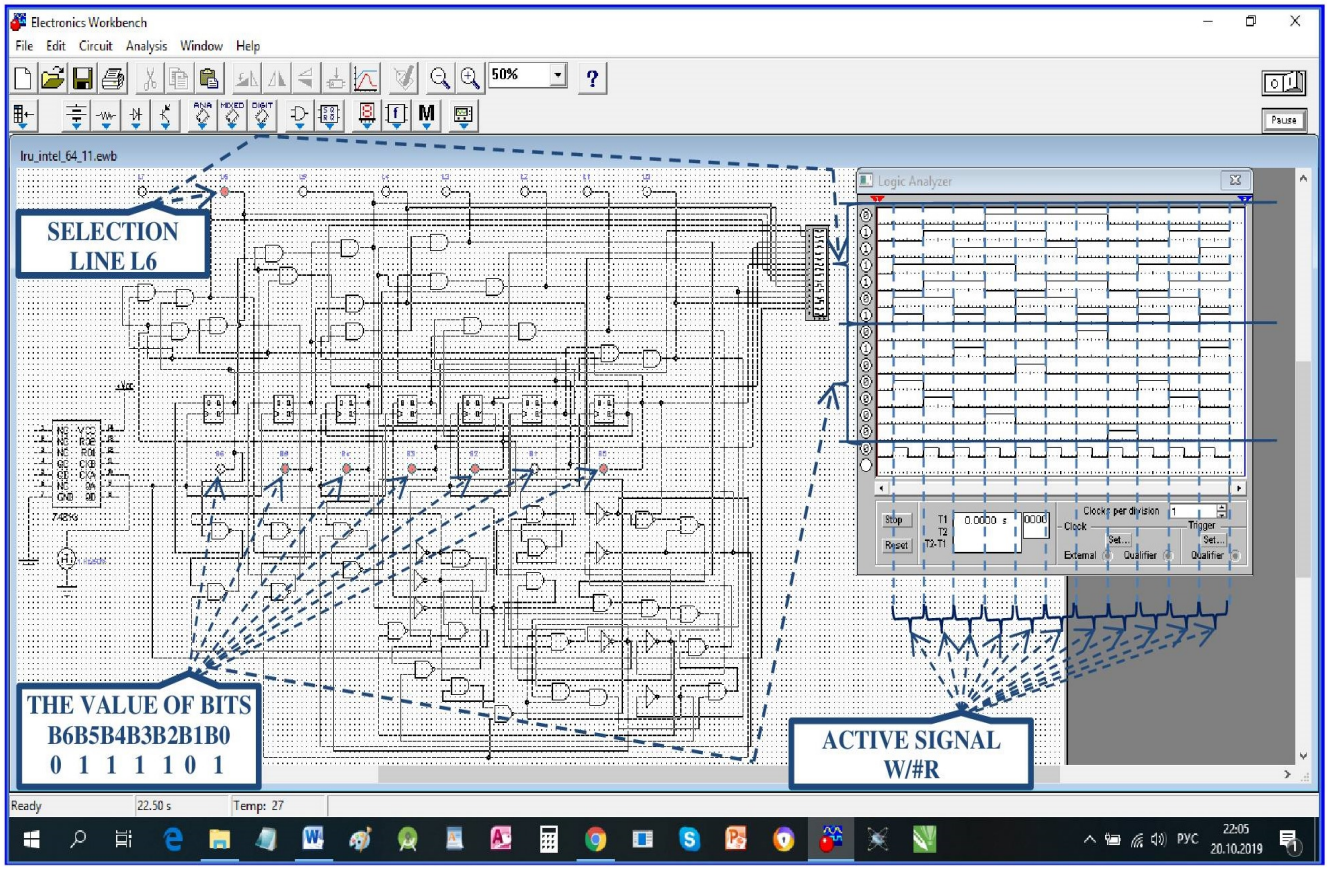


Fig. 8. Computer model of the hardware solution of the substitution politics implementation by the algorithm pseudo-LRU

The results in form of time diagram and the value of logic analyzers were obtained in the research process of the obtained model of the LRU unit in the environment of computer modelling of EWB512, where the certain sequence of proceeding in the values of bits is determined recreated, B_6, B_5, B_4, B_3, B_2, B_1, B_0: 0000000, 0001011, 0101110, 0111101, 1111000, 1110011, 1010110, 1000101 (0, 0xb, 0x2e, 0x3d, 0x78, 0x73, 0x56, 0x45) at every activity of writing signal W/R, that answers filling of lines L_0, L_4, L_2, L_6, L_1, L_5, L_3, L_7 of corresponding multitude for the unit of data of cache memory in case of hit or miss.

V. Assessment of Hardware Complexity

To exactly defined the expenses of building the combinational schemes we will use such method, as "by the Quine's estimation". It provides the total quantity of inputs of all logical elements for the digital combinational schemes built in the corresponding base:

$$K = \sum_{i=1}^{N} Ei \qquad (17)$$

where: N – quantity of inputs according to the scheme that is estimated; $E_i=1$ – direct input; $E_i=2$ – inverse input;
The combinational scheme CS_1 (Fig. 7) contains 8 logical elements of "i" for 3 inputs. The combinational scheme CS_2 consists of combinational schemes CS_{20}, CS_{21}, CS_{22}, CS_{23}, CS_{24}, CS_{25}, and CS_{26} (Fig. 7).
Combinational schemes CS_{20}, CS_{21}, CS_{22}, CS_{23}, CS_{24}, CS_{25}, CS_{26} form a value for the informative inputs D of the synchronous D - triggers D_0, D_1, D_2, D_3, D_4, D_5, D_6, forming complex switching functions Br = fr (Br, $L_{q-1}(\mu)$,., $L_0(\mu)$) of renewal bits $B_6 \ldots B_0$. The typical combinational schemes CS_{23}, CS_{24}, CS_{25}, CS_{26} consist of the consistently including logical elements of base "and –not" for 2 inputs. The general view of the Quine's estimation for the combinational schemes CS_1 is such as:

$$Kcs_1=p*2^p; \qquad (18)$$

where: p=3 – quantity of inputs

The general view of the Quine's estimation for combinational schemes CS_{23}, CS_{24}, CS_{25}, CS_{26} is such as:

$$K_{CS2[3456]}= 2^n*n*2^{n-1}+4= n*2^{2n-1}+4; \qquad (19)$$

where: n=2 – quantity of inputs, 4 –accounting the inverse inputs – outputs.
The typical combinational schemes CS_{21} and CS_{22} consist of the consistently included logical elements of base "and – not" on 3 inputs. Thus, a general view of Quine's estimation of combinational schemes CS_{21} and CS_{22} will be such:

$$K_{CS2[12]}=2*p*2^{p-2}+2=p*2^{p-1}+2; \qquad (20)$$

where: p=3 – quantity of inputs, 2 – accounting the inverse inputs – outputs.

The combinational scheme CS_{20} consists of consistently included logical elements of base "and – not" for 2 and 4 inputs. Thus, the general view of the Quine's estimation for combinational schemes CS_{20} will be the following:

$$K_{CS20}=n*2^{n-1}+1+m*2^{m-3}+2= n*2^{n-1}+1+2*n*2^{2*n-3}+2, \quad (21)$$

where: n=2 – quantity of inputs, m=2*n=4 - quantity of inputs, 1 accounting the inverse inputs - outputs, 2 – accounting the inverse inputs – outputs.

$$K = K_{CS1} + K_{CS2}+D = K_{CS1} + K_{CS2[3456]} + K_{CS2[12]} + K_{CS20} + D= p*(2^{p} +2^{p-1})+n*(2^{2*n-1}+2^{n-1}+2^{2*(n-1)}+2^{2*n.}$$

Thus, total Quine's estimation of the solution (22):

$$K = p*(2^{p} +2^{p-1})+n*(2^{2*n-1}+2^{n-1}+2^{2*(n-1)}+2^{2*n.} \quad (22)$$

Expressions (17) and (22) are identical and produce the same result by Quine's estimation. With expressions (17) combinational schemes CS1 (Fig. 7) consist of the consistently including 8 logical elements of base "and" for 3 inputs. Thus, Quine's estimation of combinational scheme CS1 will be such:

$$K_{CS}1=3*8=24;$$

The combinational scheme CS2 (Fig. 7) consist next logical elements of basis «and – not»: logical elements of 10 units for 2 inputs, additional outputs – inputs of 5 units taking into account inverters, logical elements of 4 units for 3 inputs, additional outputs – inputs of 2 units taking into account inverters, logical elements of 2 units for 4 inputs, additional outputs – inputs of 2 units taking into account inverters.

Thus, Quine's estimation of combinational scheme CS2 will be such:

$$K_{CS}2=(2*10+5)+(3*4+2)+(4*2+2)=25+14+10=49;$$

The total Quine's estimation of the hardware solution of the substitution politics by the algorithm pseudo – LRU will be such:

$$K= K_{CS}1+ K_{CS}2 + D=24+49+7=73+7=80;$$

The expression (22) has such argument as binary digit. On the basis of expression (22) the diagram on fig.14 allows to see increasing complexity in dependence from arguments n=2, p=3, m=2*n=4 (1 - additional 2^{2*n} outputs-inputs units taking into account inverters):

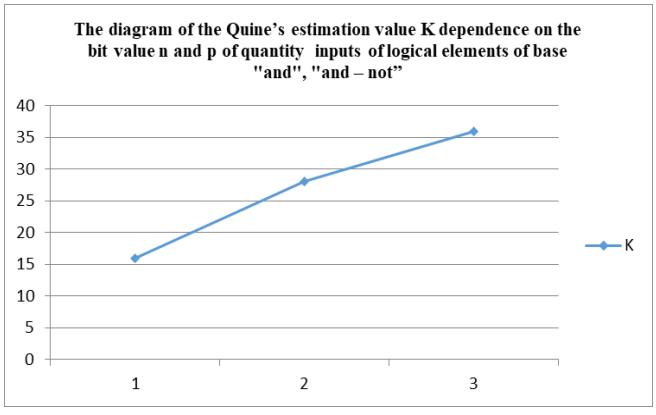


Fig. 9. The increasing complexity in dependence from arguments n=2, p=3, m=2*n=4 (1 - additional 2^{2*n} outputs – inputs units taking into account inverters)

VI Conclusions

Results obtained from synthesis of the minimum hardware solutions of the substitution politics by the pseudo algorithm – LRU associative cache memory to 8 directions and the associative translating look – a – side buffer are competitive according to the following characteristics. The computer models of the minized hardware solutions satisfy fully to politics of the substitution politics by the algorithm pseudo – LRU using the obtained in form of time diagram in the environment of computer modelling EWB512.

In the presented work the synthesized decisions of algorithm pseudo - LRU is hardware, so they take precedence over software code VHDL [2] according to the productivity: updating B bits into 6τ and selecting of corresponding line of multitude associative cache memory to 8 directions into τ, where τ - is the transition process of switching logical element "and - not" base.

Synthesized hardware of the algorithm pseudo - LRU in base "and – not" is considerably more simple unlike the realization hardware solutions LRU [3], where in particular certain combination logic such complex digital devices are additionally involved, as counters, registers and comparators, they have negative effects on such features, as: simplicstic implementation, productivity, reliability, high efficience and the Quine's estimation.

As for Quine's estimation of the hardware solution, the diagram on Fig. 9 shows that with increasing parameter of binary digit, the complexity of hardware solution will increase linearly.

Future research will be connected with development and minimizing hardware for implementation other algorithms (LFU, MRC).

References

[1] Vadym Puydenko, "The computer's model of the memory cache LRU unit of the processor's core of the architecture IA-32". 2018 International Conference Information Technologies and Computer modelling», Ivano-Frankivsk, Ukraine, May, 18-23, pp. 363-365.

[2] Safaa S. Omran, Ibrahim A. Amory, "Implementation of LRU Replacement Policy for Reconfigurable Cache Memory Using FPGA", 2018 International Conference on Advanced Science and Engineering, Kurdistan Region, Iraq, November, 12-14, pp.13-18.

[3] T.S.B. Sudarshan, Rahil Abbas Mir, S.Vijayalakshmi, "Highly Efficient LRU Implementations for High Associativity Cache Memory". Birla Institute of Technology and Science, Pilani, Rajasthan 330331 INDIA, 2017.

[4] Burhan Ul Islam Khan, Rashidah F. Olanrewaju, Roohie Naaz Mir, Abdul Raouf Khan, S. H. Yusoff A Computationally, "Efficient P-LRU based Optimal Cache Heap Object Replacement Policy". International Journal of Advanced Computer Science and Applications, Vol. 8, No. 1, 2017.

[5] Swadhesh Kumar and P K Singh. "An Overview of Modern Cache Memory and Performance Analysis of Replacement Policies". 2016 2nd IEEE International Conference on Engineering and Technology, India, pp. 4145-4148.

[6] Jaafar Alghazo, Adil Akaaboune, Nazeih Botros, "Cache Replacement Algorithm Records". 2004 International Workshop on Memory Technology, Design and Testing, Illinois, USA, August, pp. 19-24.

[7] J. Reineke, D. Grund, C. Berg, and R. Wilhelm, "Timing predictability of cache replacement policies," Real-Time Syst., vol. 37, no. 2, Nov. 2007, pp. 99–122.