Using LabVIEW Tool for Verifying the Functioning of an Electronic Programmable Circuit

Corina Daniela Cunțan, Ioan Baciu, Caius Panoiu, Raluca Rob

***Abstract*—** This paper presents an electronic circuit that identifies a code introduced using a keyboard. If this code is corresponding to the stored code, the system actuates an electromagnetic device. If the input code does not correspond to the stored one, the system blocks the inputs. The electronic circuit is realized in LabVIEW and permits an analogical command to the electromagnetic device using a data acquisition board.

***Keywords*—** programmable encoder, data acquisition board, electromagnetic device

I. INTRODUCTION

The programmable encoder that is presented in this paper realizes the coding in many variances in order not to permit the code breakage by an unauthorized person. If the person is dialling a wrong number, the execution device is blocked automatically.

The digits sequence is essential. Code number can contain the same digit more then once. The code can be changed using the input keyboard and is charged in three memory registers.

II. THEORETICAL PRESENTATION

The electric scheme contains three functional blocks: input block, command block and programming block.

Input block is composed by a keyboard for dialling the entering code and a diode matrix that realizes the binary codification. Using an eight input OR gate, the system approaches any key pressing, the output pulse is delayed using a trigger Schmitt delay circuit.

For describing the sequential system (command block) the authors use Boolean algebra and finite state automaton theory. There was adopted Moore model. The state register is represented using a temporary memory block realized with Delay latch.

The command block is depicted in figure 1. In figure 2 the authors present the system evolution in state space.

The synthesis of the combinational logic circuit (CLC) and the determination of the secondary excitation are realized using variation table of delay latch and the codification of the system states. There were used three delay latches. The transitions from one state to the other depend by an input variable. The secondary excitation function is obtained by superposition of some elementary functions:

$$f_i = \Phi_0 + \sum x_k \psi_k , \quad j = 0,1,2 \tag{1}$$

where:

- Φ_0 is the function of the transitions that are not conditioned by the input variables,
- ψ_k is function that is conditioned by xk variable.

Using Veitch Karnaugh diagrams, there can be obtained the other functions for the first delay latch:

$$\psi_1 = \overline{Q_0}\,\overline{Q_1}\,\overline{Q_2}$$

$$\psi_2 = \overline{Q_1}\,\overline{Q_0} \tag{2}$$

$$\psi_3 = \overline{Q_1}\,\overline{Q_0} \; ; \qquad \psi_4 = Q_2 Q_0$$

The input of the first delay latch can be calculated with the following relation:

$$D_0 = \Phi_0 + x_1 \Psi_1 + x_3 \Psi_2 + x_{diez} \Psi_3 + \overline{x_*} \Psi_4 =$$
$$x_1 \overline{Q}_0 \overline{Q}_1 \overline{Q}_2 + x_3 Q_1 \overline{Q}_0 + x_{diez} Q_1 Q_0 + \overline{x_*} Q_2 Q_0 \tag{3}$$

The functions ψ_k for the second delay latch are:

$$\psi_1 = Q_0 \overline{Q}_1 \overline{Q_2} \tag{4.a}$$

$$\psi_2 = Q_1 \overline{Q}_0 \tag{4.b}$$

The input of the second delay latch can be calculated with the following relation:

$$D_1 = \Phi_0 + x_2 \Psi_1 + x_3 \Psi_2 = x_2 Q_0 \overline{Q}_1 \overline{Q}_2 + x_3 Q_1 \overline{Q}_0 \tag{5}$$

The functions ψ_k for the third delay latch are:

$$\Psi_1 = \overline{Q_0}\,\overline{Q_1}\,\overline{Q_2}$$
$$\Psi_2 = Q_0 \overline{Q_1}\,\overline{Q_2} \tag{6}$$

Corina Cuntan - Electrical Engineering and Industrial Informatics Department, Timisoara Polytechnical University, Faculty of Engineering Hunedoara, (e-mail: corina.cuntan@fih.upt.ro)

Ioan Baciu - Electrical Engineering and Industrial Informatics Department, Timisoara Polytechnical University, Faculty of Engineering Hunedoara, (e-mail: ioan.baciu@fih.upt.ro)

Caius Panoiu - Electrical Engineering and Industrial Informatics Department, Timisoara Polytechnical University, Faculty of Engineering Hunedoara, (e-mail: caius.panoiu@fih.upt.ro)

Raluca Rob - Electrical Engineering and Industrial Informatics Department, Timisoara Polytechnical University, Faculty of Engineering Hunedoara, (e-mail: raluca.rob@fih.upt.ro)

$\psi_3 = Q_2$; $\psi_4 = Q_1 Q_0$

The input of the third delay latch can be calculated with the following relation:

$$D_2 = \Phi_0 + \overline{x_1}\Psi_1 + \overline{x_2}\Psi_2 + \overline{x_*}\Psi_3 + \overline{x_\#}\Psi_4 + \overline{x_3}\Psi_5 = \overline{x_1}\,\overline{Q_0}\,\overline{Q_1}\,\overline{Q_2} + \overline{x_2}Q_0\overline{Q_1}\,\overline{Q_2} + \overline{x_*}Q_2 + x_\# Q_0 Q_1 + \overline{x_3}\,\overline{Q_0}Q_1 \quad (7)$$

The synthesis of the primary excitation function is made following figure 1. Using VK diagrams, there are resulting the following relations:

$$y_2 = Q_1 Q_0 \quad (8.a)$$

$$y_1 = Q_2 Q_0 \quad (8.b)$$

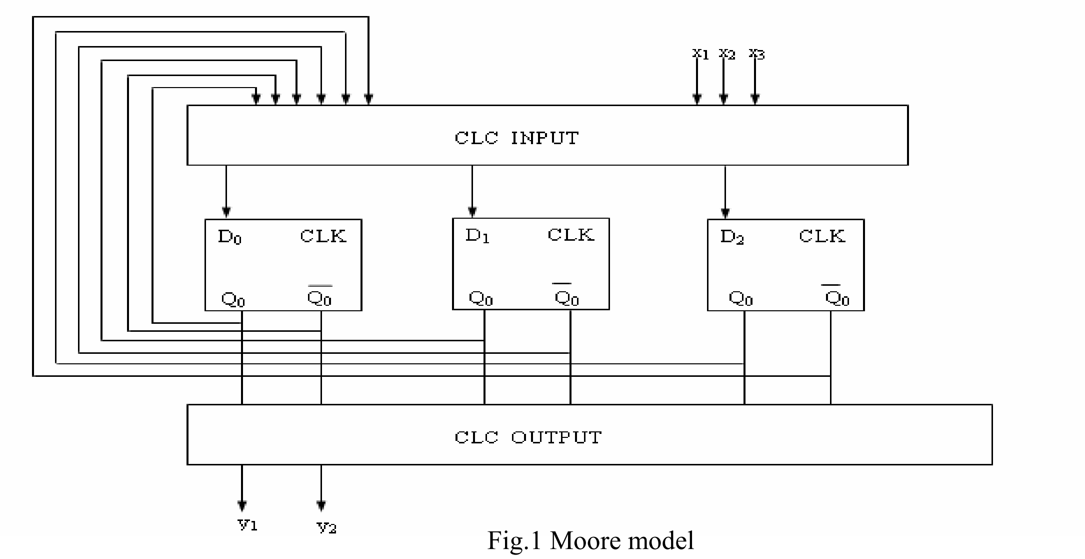

Fig.1 Moore model

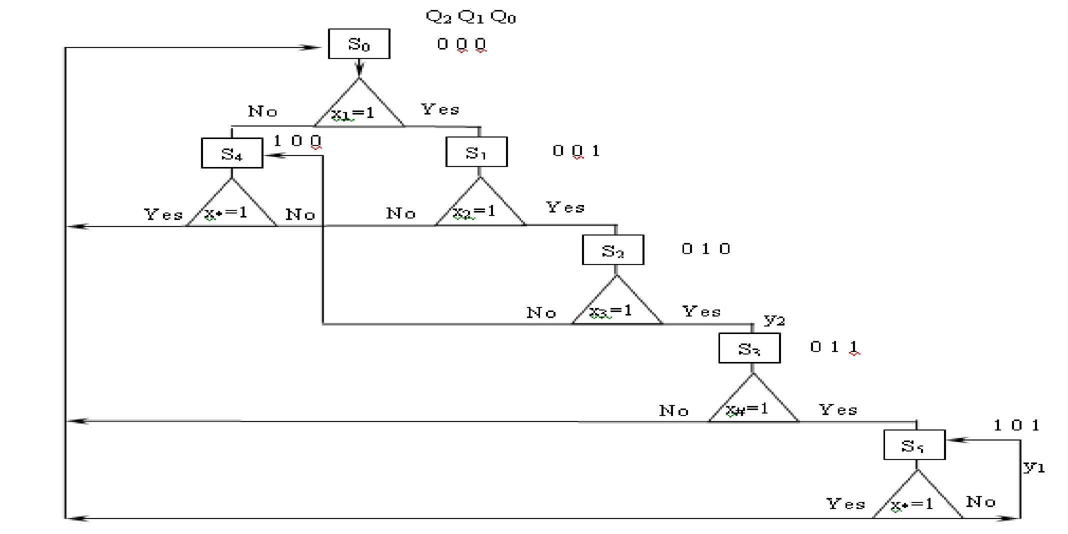

Fig.2.System evolution in state space

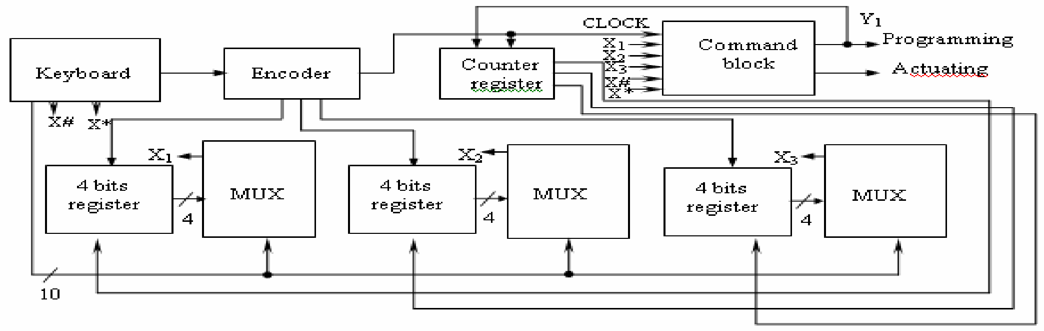

Fig. 3. Encoder block scheme.

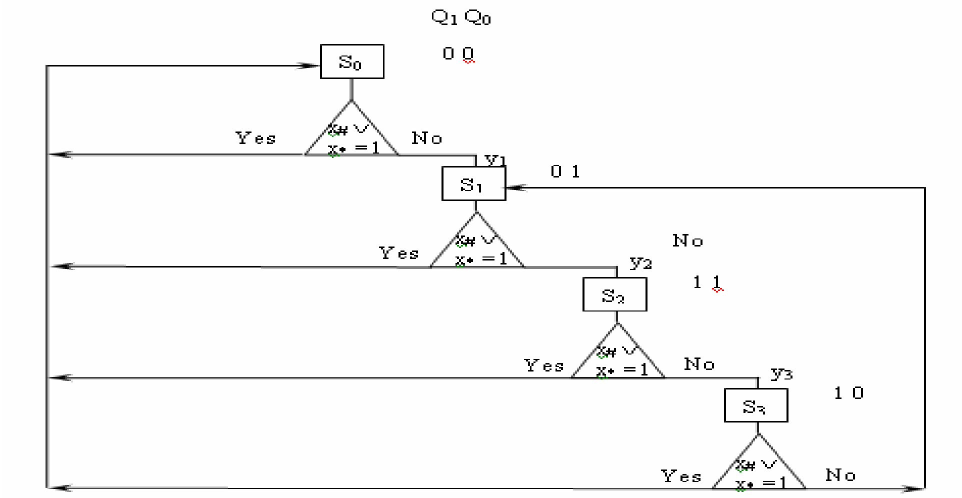

Fig. 4. Evolution algorithm for counter synthesis.

The programming block is presented in figure 3 and serves at code pre-establishing.

The electric scheme contains three 4 bits registers. The inhibit input of the inputs and outputs of the delay latches that are contained by the registers are on low state. The outputs maintain their state until the clock pulse turns to a new rising edge. The outputs of the register are connected to the control inputs of an analogical multiplexer.

The 10 numerical keys are connected directly to the multiplexer channels.

The clock pulses for all three registers are apportioned successively by a counter realized with two Delay latches.

For synthesis of the counter, the authors start from the evolution algorithm as in figure 4.

The first counter latch:

$$\psi_1 = \overline{Q_1} + \overline{Q_0} \tag{9}$$

$$D_0 = \overline{x_\#}\,\overline{x_*}\,\overline{Q_1}\,\overline{Q_2} = \overline{x_\# + x_* + Q_1 Q_0} \tag{10}$$

The second counter latch:

$$\Psi_1 = Q_0 \tag{11}$$

$$D_1 = \Phi_0 + \overline{x_\# + x_*\Psi_1} = \overline{x_\# + x_* Q_0} = \overline{x_\#}\,\overline{x_*}Q_0 = \overline{x_\# + x_*\overline{Q_0}} \tag{12}$$

The outputs can be followed with the relations:

$$y_1 = Q_0\overline{Q_1},\ y_2 = Q_0 Q_1,\ y_3 = \overline{Q_0} Q_1 \tag{13}$$

When the user press x# key, the electric scheme blocks the execution part and the clock pulses are applied to the counter register. The x* key resets the entire circuit. Simultaneously the counter register resets, too. The same effect (counter register reset) has a new x# key pressing. This operation is necessary when the user wants to set in circuit memory a new code number [1], [2], [3] and [4].

III. LABVIEW DESIGN OF THE FUNCTIONAL BLOCKS

General scheme of the encoder, presented in figure 5, contains the following functional blocks: encoder, counter register, multiplexer, 4 bits register and command block [5].

The encoder permits to obtain a 4 bits sequence for all 10 inputs that correspond to the selection buttons of the circuit code. It contains 10 Boolean controls and 4 Boolean indicators for displaying the output sequence.

The counter register, whose scheme is presented in figure 6, permits to obtain a 3 bits sequence that generates the clock pulse for all registers. The multiplexer, presented in figure 7, devolves at its output the input selected line corresponding to 4 bits code from the encoder.

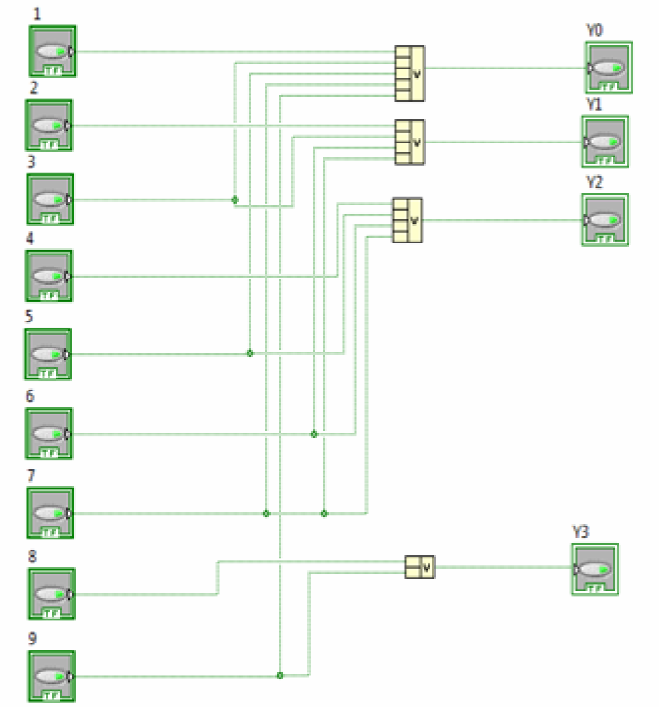

Fig. 5. LabVIEW functional block of the encoder.

The 4 bits register, presented in figure 8, memories the encoder output sequence and devolves to multiplexer when achieves the clock pulse from the counter register.

Moore automaton that is realised in LabVIEW (figure 9) permits the user to obtain the states that correspond to latch outputs command stages.

Front panel and the states that correspond to circuit (figure 10) block show the programming regime.

After using serially the three code numbers, it can be observed that the containing of the memory registers and the counter states are modifying (figure 11, 12, 13) reading the programming state. After that it is waiting for the user to introduce the code (figure 14).

After serially introducind of the memoried code, it can be observed the automaton latch states (figure 15, 16, 17).

If digit order is not respected, or the code is wrong dialled, the circuit blocks.

For modifying the code digits, the circuit has to turn on the programming state (figure 18), and the new code is dialled following the steps that were mentioned previously.

Command block realizes the impulse command for programming sequence, respective for the actuating sequence in function of the output states of the multiplexers. Programming sequence generates the selection for the clock pulses of the counter register block. As well, the command block permits the functioning reset for the entire circuit using x* key. At pressing the x# key, the execution part of the circuit is blocking and clock pulses are applied to counter register.

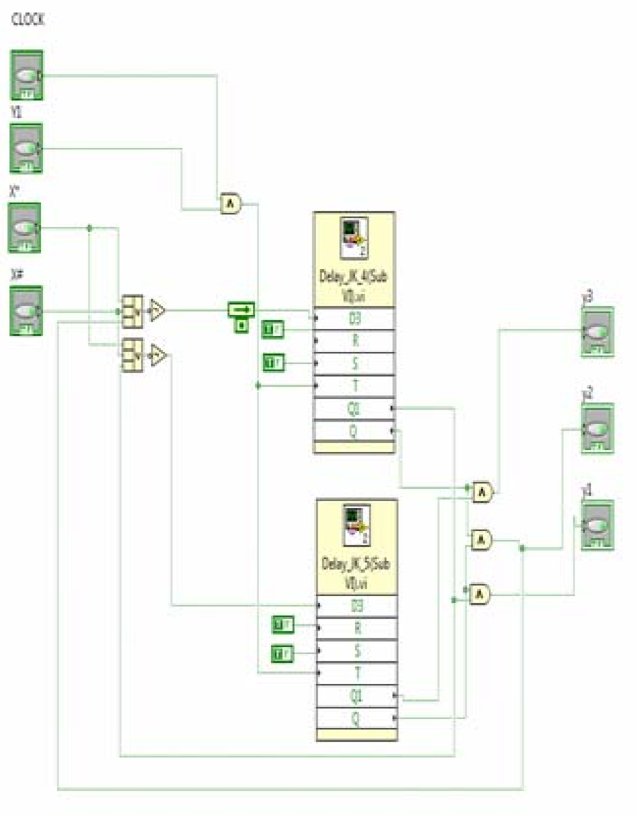

Fig. 6. LabVIEW functional block of the counter register.

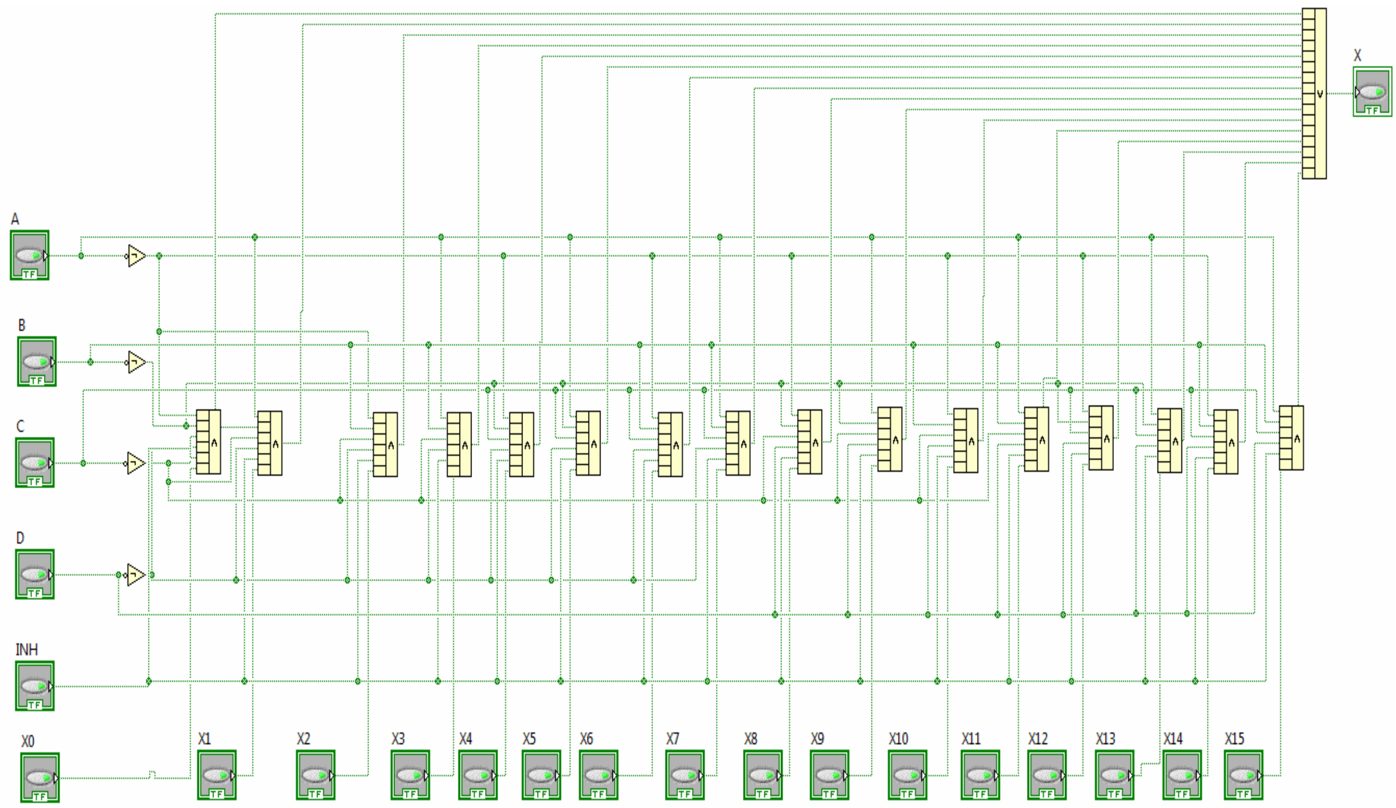

Fig. 7. LabVIEW functional block of multiplexer.

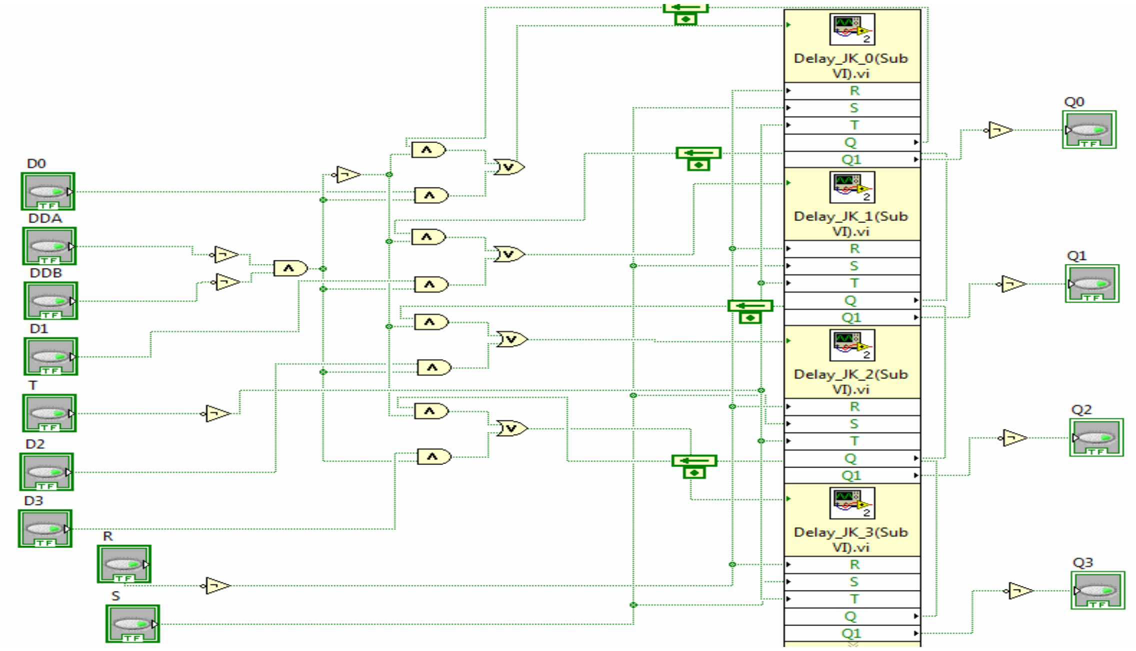

Fig. 8 LabVIEW functional block of 4 bits register.

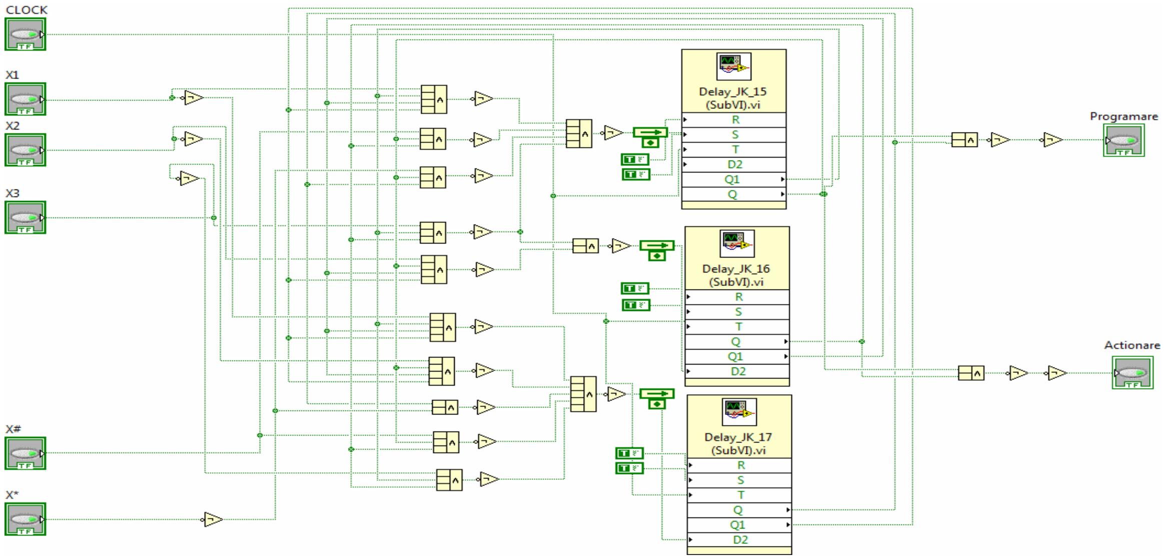

Fig. 9. State machine realised in LabVIEW

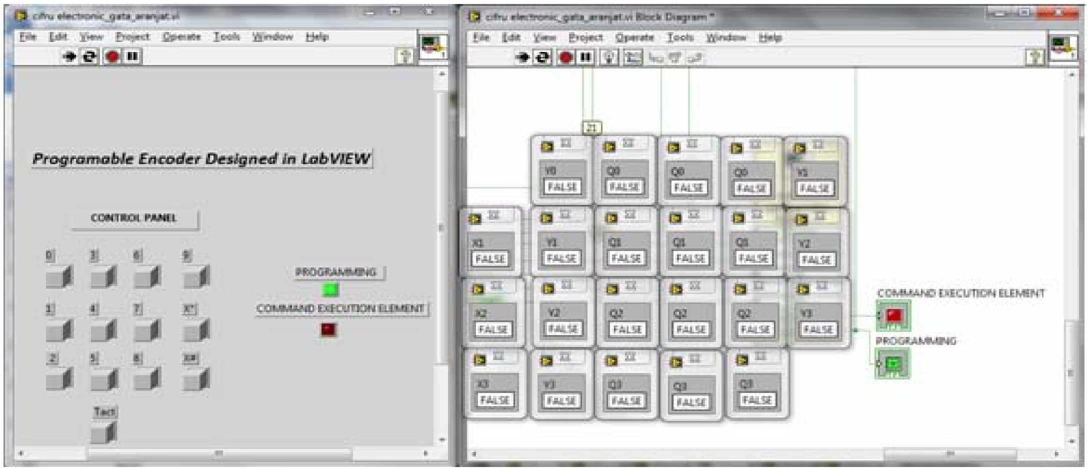
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
0
3
6
9
1
4
7
X*
2
5
8
X#
Tact
PROGRAMMING
COMMAND EXECUTION ELEMENT
cifru electronic_gata_aranjat.vi Block Diagram *
File Edit View Project Operate Tools Window Help
Y0
Q0
Q0
Q0
Y1
FALSE
X1
Y1
Q1
Q1
Q1
Y2
X2
Y2
Q2
Q2
Q2
Y3
X3
Y3
Q3
Q3
Q3
COMMAND EXECUTION ELEMENT
PROGRAMMING

Fig. 10. The front panel for programming situation.

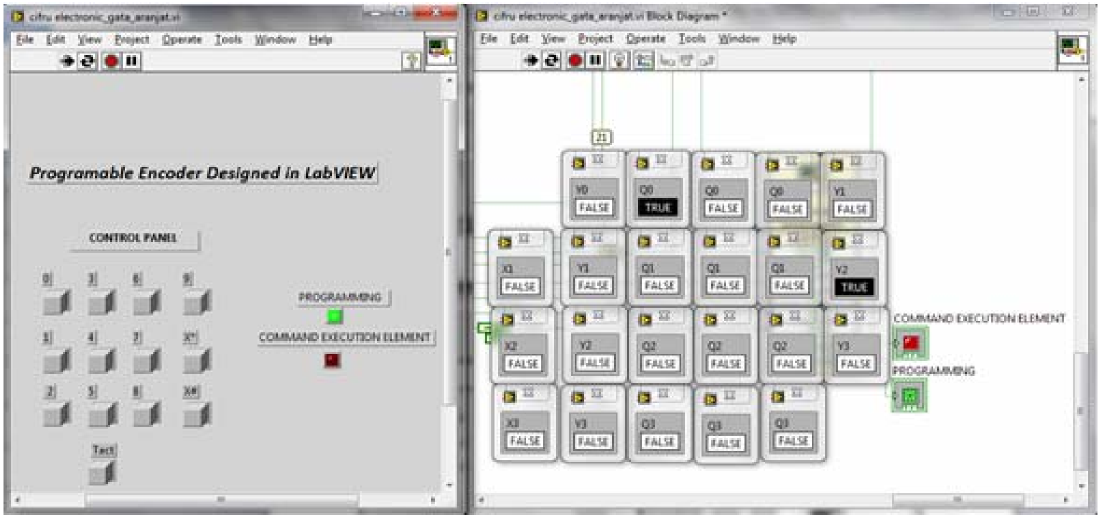
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
0
3
6
9
1
4
7
X*
2
5
8
X#
Tact
PROGRAMMING
COMMAND EXECUTION ELEMENT
cifru electronic_gata_aranjat.vi Block Diagram *
File Edit View Project Operate Tools Window Help
21
Y0 FALSE
Q0 TRUE
Q0 FALSE
Q0 FALSE
Y1 FALSE
X1 FALSE
Y1 FALSE
Q1 FALSE
Q1 FALSE
Q1 FALSE
Y2 TRUE
X2 FALSE
Y2 FALSE
Q2 FALSE
Q2 FALSE
Q2 FALSE
Y3 FALSE
X3 FALSE
Y3 FALSE
Q3 FALSE
Q3 FALSE
Q3 FALSE
COMMAND EXECUTION ELEMENT
PROGRAMMING

Fig. 11. Front panel and register states at writing 1.

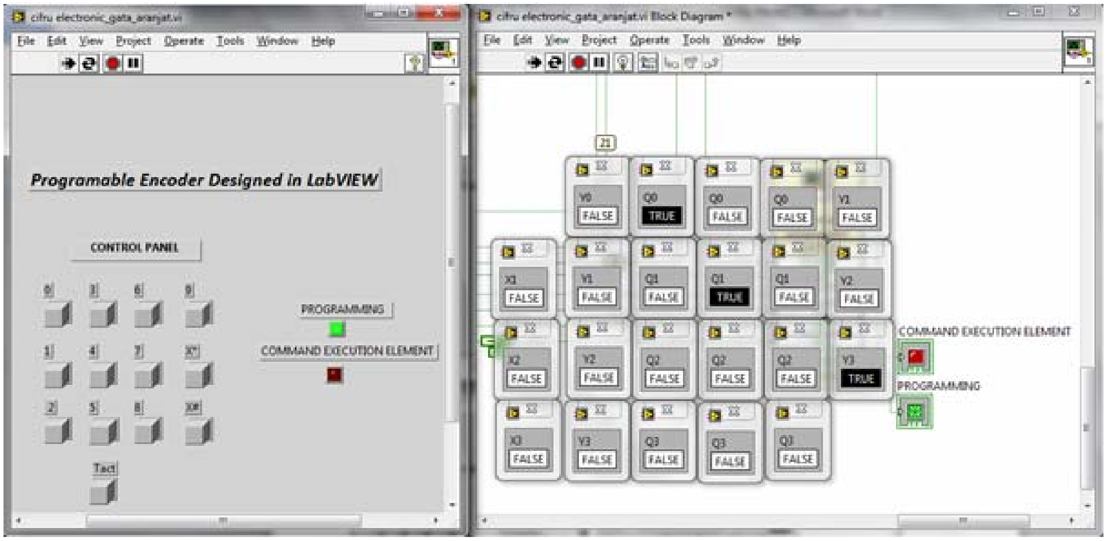
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
0
3
6
9
1
4
7
X*
2
5
8
X#
Tact
PROGRAMMING
COMMAND EXECUTION ELEMENT
cifru electronic_gata_aranjat.vi Block Diagram *
File Edit View Project Operate Tools Window Help
21
Y0 FALSE
Q0 TRUE
Q0 FALSE
Q0 FALSE
Y1 FALSE
X1 FALSE
Y1 FALSE
Q1 FALSE
Q1 TRUE
Q1 FALSE
Y2 FALSE
X2 FALSE
Y2 FALSE
Q2 FALSE
Q2 FALSE
Q2 FALSE
Y3 TRUE
X3 FALSE
Y3 FALSE
Q3 FALSE
Q3 FALSE
Q3 FALSE
COMMAND EXECUTION ELEMENT
PROGRAMMING

Fig. 12. Front panel and register states at writing 2.

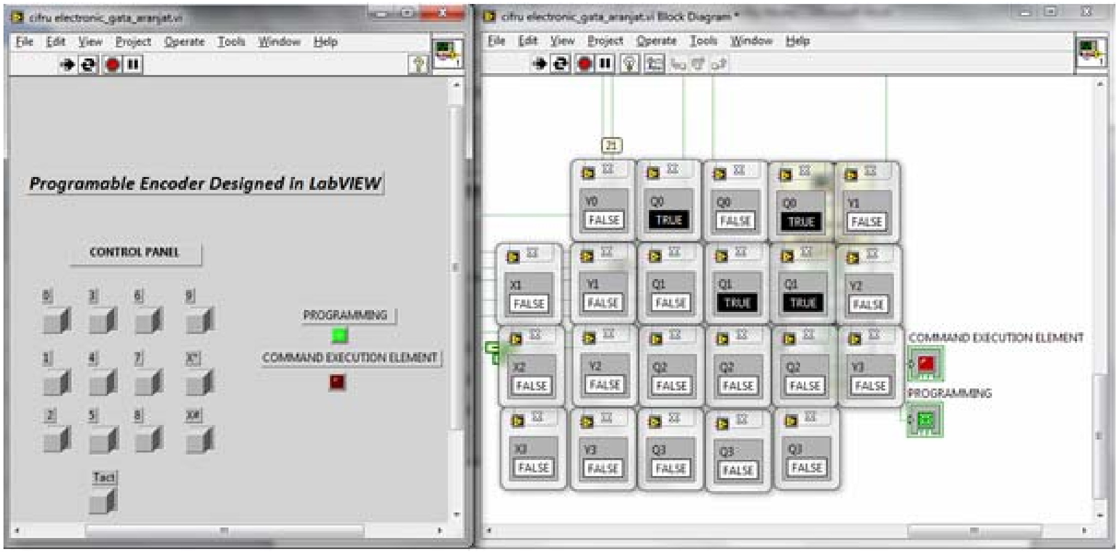
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
0
3
6
9
1
4
7
X*
2
5
8
X#
Tact
PROGRAMMING
COMMAND EXECUTION ELEMENT
cifru electronic_gata_aranjat.vi Block Diagram *
File Edit View Project Operate Tools Window Help
21
Y0 FALSE
Q0 TRUE
Q0 FALSE
Q0 TRUE
Y1 FALSE
X1 FALSE
Y1 FALSE
Q1 FALSE
Q1 TRUE
Q1 TRUE
Y2 FALSE
X2 FALSE
Y2 FALSE
Q2 FALSE
Q2 FALSE
Q2 FALSE
Y3 FALSE
X3 FALSE
Y3 FALSE
Q3 FALSE
Q3 FALSE
Q3 FALSE
COMMAND EXECUTION ELEMENT
PROGRAMMING

Fig. 13. Front panel and register states at writing 3.

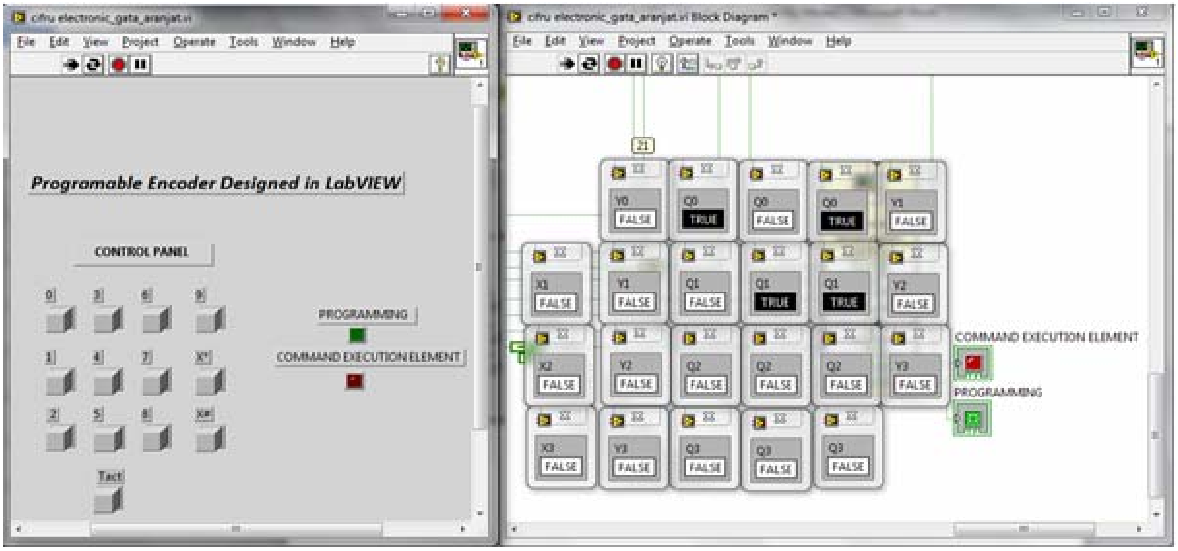

Fig. 14. Front panel for waiting state.

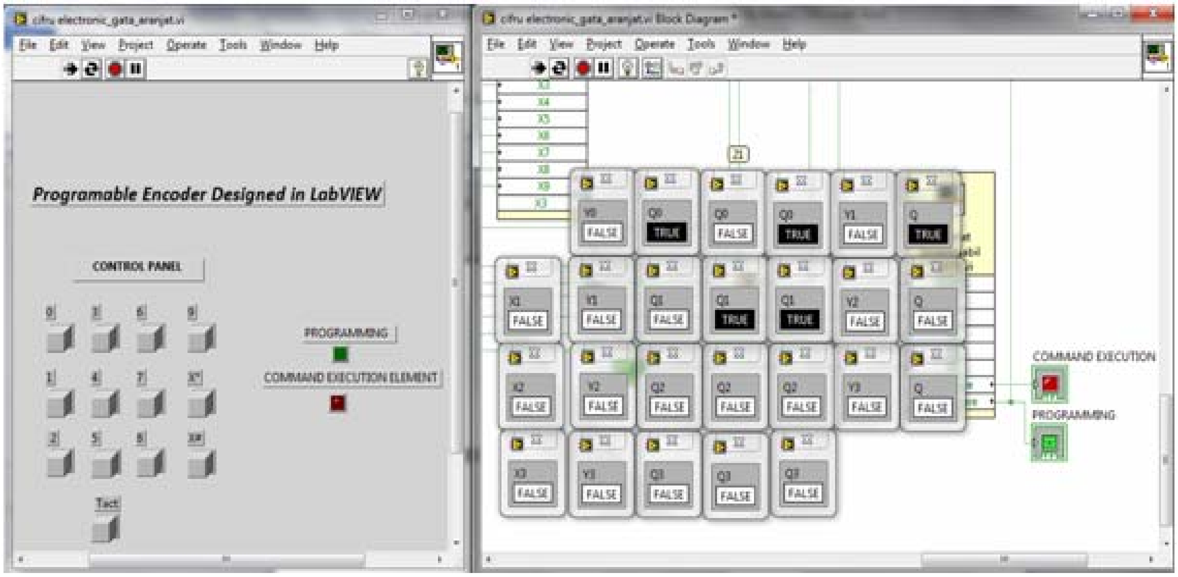

Fig. 15. Front panel and latch state at dialling the first memoried digit.

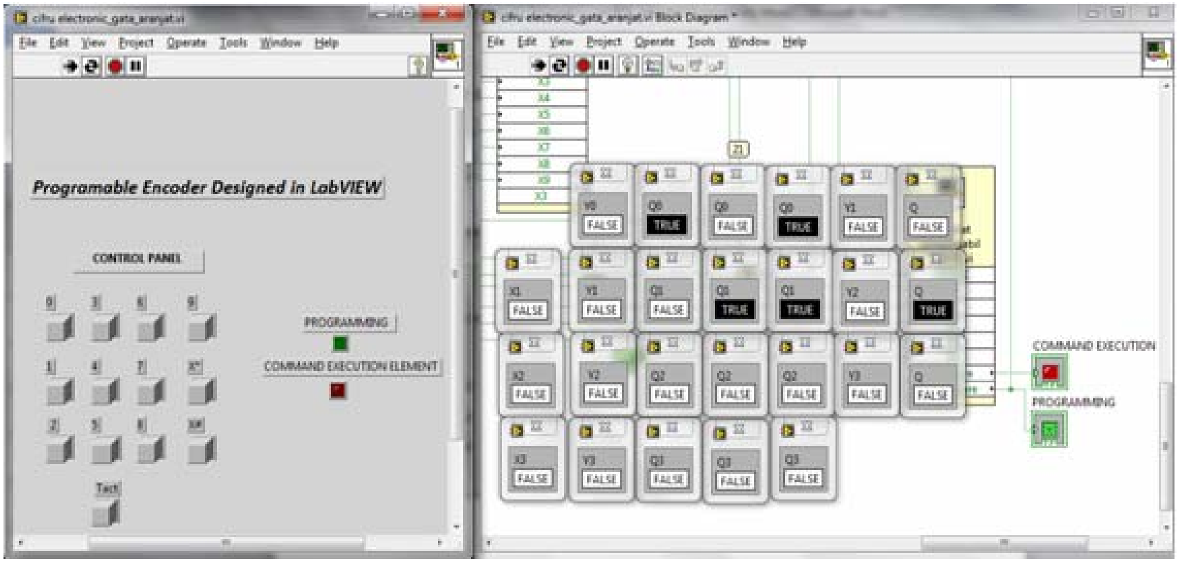

Fig. 16. Front panel and latch state at dialling the second memoried digit.

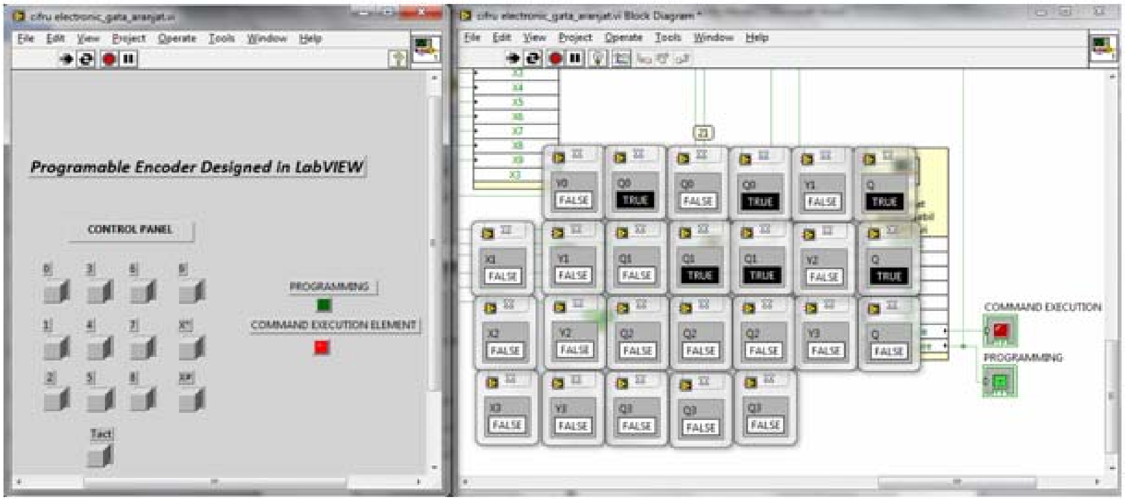
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
PROGRAMMING
COMMAND EXECUTION ELEMENT
Tact
cifru electronic_gata_aranjat.vi Block Diagram
COMMAND EXECUTION
PROGRAMMING

Fig. 17. The situation of command the execution element.

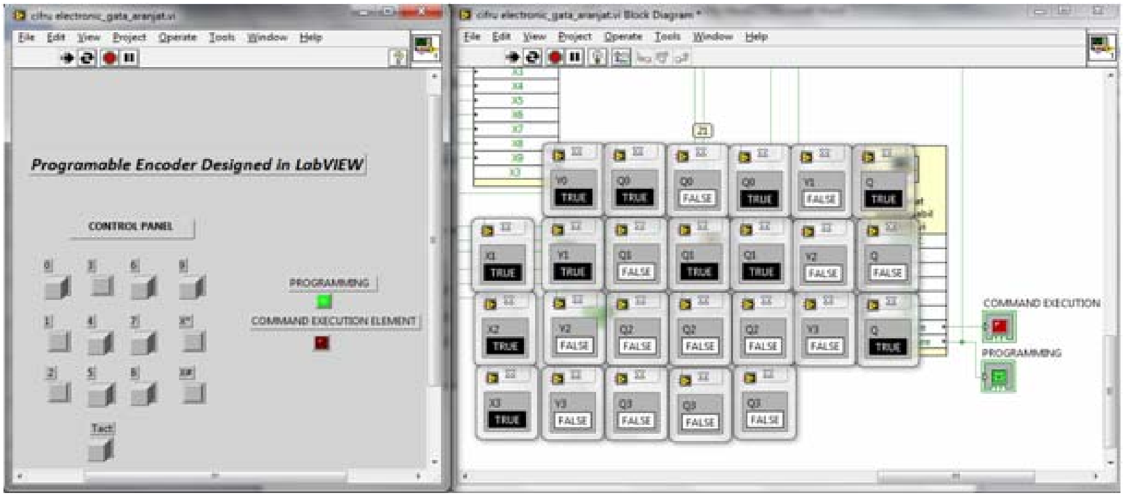
cifru electronic_gata_aranjat.vi
File Edit View Project Operate Tools Window Help
Programable Encoder Designed in LabVIEW
CONTROL PANEL
PROGRAMMING
COMMAND EXECUTION ELEMENT
Tact
cifru electronic_gata_aranjat.vi Block Diagram
COMMAND EXECUTION
PROGRAMMING

Fig. 18. Front panel and latch state at changing from the action state to the programming state.

IV. Conclusions

The scheme of the electronically circuit is realized using functional blocks. Because the scheme contain many identical blocks, the authors was used an important facilities of the LabVIEW [6]-[7], consisting in possibility of using Sub Virtual Instrument blocks. Each block was realized and tested individually, the authors checking his functionality.

Taking into consideration the complexity of the electronically circuit, in this paper are presented the internal structures for each block and the entire scheme being presented as block scheme.

Based on LabVIEW scheme and using an adequate data acquisition board it can be realised a command to an external execution element.

References

[1]Cristian Pîrvu, Note de Aplicatie în Electronica Digitală, Editura Albastră, Cluj Napoca, 2007.

[2] S., I. Spanulescu, Circuite integrate digitale, sisteme cu microprocesoare. Editura Victor. Bucuresti, 1996 2.

[3]Ardelean I. (colectiv), Cicuite integrate CMOS. Manual de utilizare, Editura Tehnică, Bucuresti, 1986.

[4] Stojanov I. (colectiv), De la poarta TTL la Microprocesor, Editura Tehnică, 1987.

[5]Cottet F., Ciobanu O., Bazele programării în Labview, Editura Matrix, Bucuresti, 1998.

[6] R. A. Jabbar Khan, A. Mohammed, M. Junaid, M. A. Masood, A. Iftkhar, LabVIEW based Electrical Machines Laboratory for Engineering Education, WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION, Issue 5, Volume 7, May 2010.

[7] B. Ando', A. Ascia, S. Baglio, N. Pitrone , Experimental Training on Advanced Technologies, WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION Issue 8, Volume 5, August 2008.

[8] Ertrugul, N. – LabVIEW for Electric Circuits, Machines, Drives, and Laboratories, Pretince Hall PTR, NJ, 2002.