Development and Modeling of Remote Laboratory Works for Engineering Education

Development and Modeling of Remote Laboratory Works for Engineering Education

B B Sus[1]*, S P Zagorodnyuk[2], O S Bauzha[2] and A Kozinetz[1]

[1]Department of Nanophysics of Condensed Matter,
Taras Shevchenko National University of Kyiv, Kyiv 01033, Ukraine
[2]Department of Computer Engineering,
Taras Shevchenko National University of Kyiv, Kyiv 01033, Ukraine

*Email: bnsuse@gmail.com

Abstract. The basic principles of finite-state machine synthesis and simulation on Proteus CAD and VHDL are discussed. The main steps of designing an electronic circuit of the control and operating finite-state machines that perform the calculation of a given algebraic expression were described. The main steps of consistent performance of a complex task for the development of cross-cutting knowledge and skills, which must be performed by the student in the framework of the normative discipline ware analysed in detail. It is shown that the developed cycle of laboratory works can be successfully implemented in various training courses.

1. Introduction

The application and effectiveness of distance and virtual works in the educational process of engineering specialties have some new features and benefits such as clarity and convenience of modeling processes and phenomena, closer mutual integration of disciplines, standardized curriculum, the formalized procedure for assessing technical tasks, and flexible system of sufficient formation of assignments. It turns out that the solution to this class of technical problems is almost impossible without the widespread use of specialized computer-aided design systems.

One such effective computer-aided design system, which allows the synthesis, circuit modeling, and simulation of electronic circuits in real-time, is the software package Proteus Design Suite.
The Proteus simulator library has a large number of ready-made electronic components: power supplies, generators, signal analyzers (oscilloscope, voltmeter, ammeter, frequency analyzer), indicators for monitoring the parameters of the circuit in real-time (probes), discrete components, including resistors, capacitors, transformers, sensors and others. Simulation of the real behavior of electronic components and circuits based on the use of a specialized modeling language SPICE (Simulation Program with Integrated Circuit Emphasis). As a result of the evolution of software emulators, the SPICE language has become the major language for most programs that simulate the behavior of electronic components.

The software package Proteus VSM allows the student to simulate both analog and digital circuits, their combination, as well as simulation of complex sensory instruments and electronic systems on microcontrollers [1, 2]. The main advantage of the Proteus software simulator is the presence of libraries that include a large number of microcontrollers, microprocessors and logic elements. Using the Proteus environment allows you to visualize the functioning of the nodes of the computer system at the micro-level. In particular, an laboratory evaluation of the efficiency of using the Proteus virtual environment was discussed in [3]. The simulation of a digital device for measuring alternating current, voltage and power has been performed in [4]. Hardware implementations of the electric brushless motor driver circuit with three stator windings, as well as modeling of the driver circuit in the Proteus CAD have been discussed in [5] and visual 3-d modeling is described in [6]. In particular, performance evaluation of microcontroller driver implemented in the circuit was made by the means of well-known popular C-language model library. The article demonstrates that the logic of the microcontroller program can be constructed, debugged and compiled on an independent computer in a programming environment for any model of microcontroller available in Proteus, and then the binary file of the compiled program can be downloaded directly to the simulation unit of the selected microcontroller model.

Another alternative, but very effective and powerful way to form and test the logic of the electronic circuit is to describe the operation of the new synthesized device in a specialized language of equipment description. There are many hardware description languages, but the most popular are VHDL and System Verilog. The language VHDL (Eng. VHSIC - Very high speed integrated circuits Hardware Description Language) is the basic language in the development of devices for modern computer systems. For example, the problem of designing and modeling an electric motor driver, similar to [5], has also been successfully solved using the VHDL language in solar cell rotation systems [7] and the vehicle automation system [8].

VHDL is a tool for describing mostly digital systems, but there is a modification of this language called "VHDL AMS" (Analog and Mixed Signals), which allows us to describe both analog and combined digital-analog circuits [9]. Typically, the analog part of the circuit get the analog signal of the sensor set and digitizes it using an ADC. Then the digital signal is processed by the main digital section of the mixed circuit. Such the mechanism used by medical diagnostic equipment, for the design of which the VHDL AMS language is best fitted [10].

Composing an electronic circuit in hardware description languages opens the developer the opportunity to create circuits of arbitrary complexity, including developing new processors with a limited set of instructions [11], or new specific instructions [12]. A designed, debugged and successfully compiled electronic circuit in a VHDL or System Verilog application development environment can further be simulated using other automated design systems, such as ModelSIM or Simulink [13]. Also, the binary file of the compiled VHDL / Verilog program can also be downloaded to the programmable logic integrated circuit (Programmable Logic Device, PLD) - a specialized minicomputer, the logic of which is not determined during the manufacture of the device, but determined by the developer using programming languages equipment.

Modern PLD controllers have a wide range of additional hardware-implemented auxiliary components, including signal generators and analyzers, Digital Signal Processor, logic elements, addition and multiplication devices, input, output and information storage devices [14]. Similar to the library components of the Proteus CAD, optional components of the PLD minicomputer can be handled by the developer together with the new electronic circuit. This type of modern PLD device is called a "Field-programmable gate array" (FPGA). The developer can easily convert the FPGA controller into an experimentally functioning instance of its integrated circuit and can then supply the input of this controller with real electrical signals and receive the actual generated signals at the output of the controller [15]. Therefore, the logic of the FPGA-controller can be said that one part of it is implemented during the manufacture of the controller and can not be redefined by the developer, and the other part can be designed by the inventor, who, conversely, can repeatedly change, supplement and expand it.

The article talks about the procedure of designing a complex electronic circuit using Mill finite state machines using the Proteus automatic design system, as well as the implementation of this circuit in VHDL and execution of this circuit on an FPGA-controller Altera Cyclone-II. Similar problems in the designing of digital circuits based on Moore's and Mealy finite-state machines are fully in line with global trends and are actively solved by research engineers. [16].

2. Formulation of the problem.

During the research, a series of laboratory works were developed. It can be used in various training courses: "Digital Circuitry and Electronics", "Intelligent Systems, Control and Automation", "Computer Logic" and "Microprocessor Engineering", and therefore the development took into account the interdisciplinary approach. The study of these courses provides the student the opportunity to gain fundamental knowledge and experience in the development and synthesis of electronic circuits with an assigned functionality. These disciplines give students the opportunity to consistently expand and increase the valuable properties of new devices. Consequently, in the course of "Digital Circuitry and Electronics" simple logic gates and combinational circuits and triggers were considered in detail. In the course of "Intelligent Systems, Control and Automation" complex combinational circuits and digital finite-state machines are already considered with the elements of the previous course that are already used as ready modules. The course of "Computer Logic" analyzes the structure of the operating machine and the logic of its functioning. The course of "Microprocessor Engineering" examines the actual microcontrollers with the peculiarities of their architecture.

Mastering these courses requires a clear sequence of learning and also involves the use of modern CAD-systems for the design and simulation of devices synthesized by students. The article demonstrates in detail the main steps of consistent implementation of a complex task that must be performed by the student in the framework of the discipline "Computer Logic".

The task for the student is to develop an operating scheme of a computing device that implements a certain computational operation. This operation may consist of the following arithmetic operations: addition, subtraction, multiplication, division, root extraction, multiplication and division by 2 (offset left and right), synthesis and construction of the functional diagram of the operating and control machines, which together provide control of the computing device. During the laboratory work, the student is limited in the choice of the element base of logical devices, types of triggers as memory cells, as well as algorithms for operations of multiplication, division, and root extraction. The large variety of elements is excessive, excessive, and significantly reduces the versatility of the synthesized device. It also requires the student to perform their assignment, which differs from the tasks of other students. A total of 1024 modifications of tasks have been developed for this work.

During laboratory work, it is very important to optimize the schematic diagram, reduce the number of logical elements, since it forms the economic efficiency of the subsequent hardware-implemented circuit.

Assume that the computational operation to be implemented is the calculation of the expression 4C 2AB. Use the Mealy finite-state machine as the controller, and use the D-flip-flops as the memory cells. Also, the student should implement a multiplication algorithm: "Multiplication from the lower digits of the multiplier and the shift of the sum of partial terms to the right."

The multiplication operation is performed in direct codes. Sign and main digits are processed independently. To determine the sign of the result of multiplication, XOR operation on the sign bits of the multiplied numbers is performed.

Let the multiplied Y (Y= 0, y1 , y2 ,…, yn ; yi be {0,1}) and the factor X (X =0, x1 , x2 ,…, xn ; xi be {0,1}). Then the result of multiplication of the absolute values of Y та X can be expressed as:

$$Z = YX = Yx_1 2^{-1} + Yx_2 2^{-2} + \ldots + Yx_n 2^{-n}$$

Multiplication of two numbers Y and X can be realized by performing a cyclic process, the nature of which depends on the specific form of expression. Each multiplication cycle consists of adding the next partial product, which is the product of a multiplier multiplied by one digit, to the sum of the partial products. We present the multiplication as:

$$Z = YX = ((\ldots((0+Yx_n)\, 2^{-1} + Yx_{n-1})\, 2^{-1} + \ldots + Yx_i)\, 2^{-1} + \ldots + Yx_1) 2^{-1}$$

From the obtained expression it is seen that if we denote by $Z_i = (Z_{i-1} + Yx_{n-i+1}) * 2^{-1}$ where i is {1..n}, then then we can find the value of the result of multiplication by iteration. Initial values of i = 1, $Z_0 = 0$. At the end of n iterations $Z_n = Z = YX$.

In this method, multiplication is performed from the lower digits of the multiplier, the sum of partial results of multiplication is shifted to the right, and the multiplied remains fixed. The operating scheme of the multiplication machine according to the first algorithm is presented in Fig. 1. The multiplied (Y) is expressed in the lower bits of the register RG3, the multiplier (X) is written in the register RG2. The result of the multiplication (Z) will be obtained in the registers RG1 (senior digits) and RG2 (junior digits). Before multiplication, the register RG1 is set to zero. Counting the number of multiplication cycles is provided by CT counters, according to which its bit rate q, 2q ≥ n is chosen.

The algorithm for obtaining the result in this method is as follows:

1. Enter the multiplied (Y) in the lower digits of the register RG3.
2. Enter the factor (X) in the register RG2.
3. The contents of the adder (register RG1) are reset.
4. Add to the register RG1 the contents of RG3 (when the least significant bit RG2 is 1), or 0 (when the least significant bit RG2 is 0). The bit matrix AND is used.
5. Move to the right (division by 2) registers RG1 and RG2. During the offset, the least significant bit of the register RG1 crosses to the most significant bit of the register RG2, and the least significant bit of the register RG2 is lost.
6. Repeat steps 4 and 5 (n-1) more times.

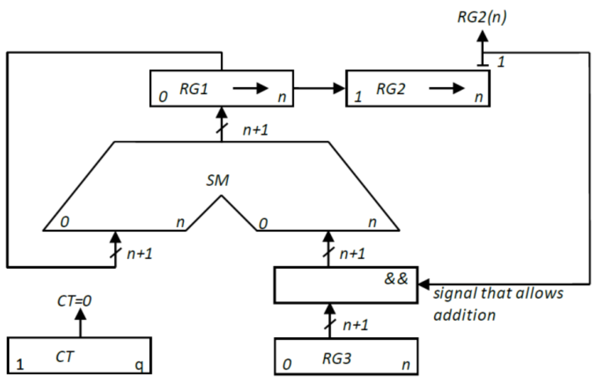


Figure 1. Operating diagram of the multiplication device.

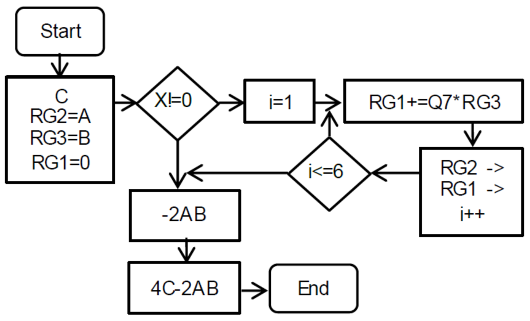


Figure 2. Essential scheme of the algorithm

A constructed table of coding of vertices and the graph-scheme of transitions of the datapath is presented in fig.3.

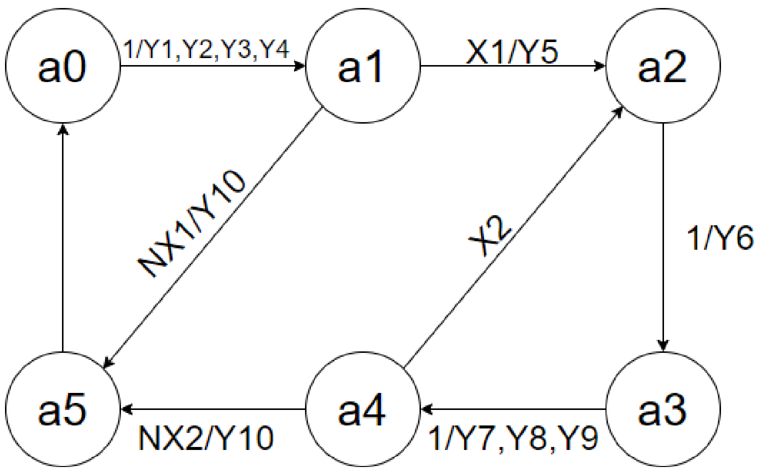


Figure 3. Graph diagram of the transitions of the datapath.

Table 1. Vertex coding table.

Code	Content	Notes
Y1	C	Enter C
Y2	RG2=A	Write in RG2=A
Y3	RG3=B	Write in RG3=B
Y4	RG1=0	Fill RG1 with zero
Y5	i=1	Counter is set to zero
Y6	RG1+=Q7*RG3	If the las bit RG2[Q7]=0, then write the zeros into RG1, if 1, then write the bits from RG3 into RG1
Y7	RG2->	Shift RG2
Y8	RG1->	Shift RG1
Y9	i++	Counter increment
Y10	-2AB	Convert AB into the supplementary code
Y11	4C-2AB	Calculate 4C-2AB
X1	X!=0	Check the multiplier by 0
X2	i<=6	Check the counter at the end of the multiplication

Encoding of the state of the machine is presented it table 2:

Table 2. Encoding of the state of the finite-state machine.

Status	Code	Decimal notation
a0	101	5
a1	010	2
a2	100	1
a3	110	3
a4	001	4
a5	000	0

We obtain a structural table of transitions of the datapath, from which we build a system of equations of outputs and a system of equations of transitions of the finite-state control machine.

Table 3. Structural table of transitions-outputs of the Mealy finite-state machine.

S	E	X	Y	Ks	Ke	Ф3
a0	a1	1	Y1,Y2,Y3,Y4	101	010	D2
a1	a2	X1	Y5	010	100	D1
	a5	NX1	Y10	010	000	-
a2	a3	1	Y5	100	110	D1D2
a3	a4	1	Y7,Y8,Y9	110	001	D3
a4	a2	X2	-	001	100	D1
	a5	NX2	Y10	001	000	-
a5	a0	1	Y11	000	101	D1D3

System of transition equations:

$D1 = a_1x_1 \vee a_2 \vee a_4x_2 \vee a_5$

$D2 = a_0 \vee a_2$

$D3 = a_3 \vee a_5$

System of output equations:

$Y1=Y2=Y3=Y4=a_0$

$Y5=a_1x_1$

$Y6=a_2$

$Y7=Y8=Y9=a_3$

$Y10=a_1\bar{x}_1 \vee a_4\bar{x}_2$

$Y11=a_5$

3. Computer Simulations

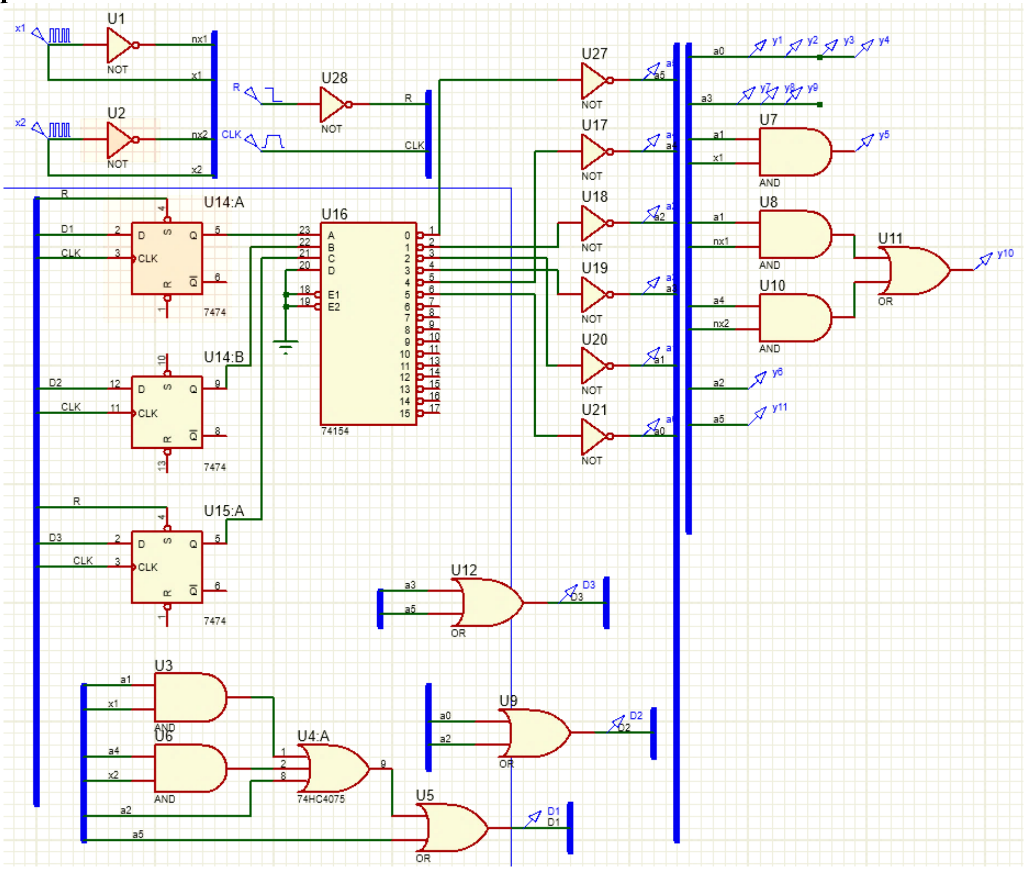


Figure 4. Functional diagram of the finite-state control machine.

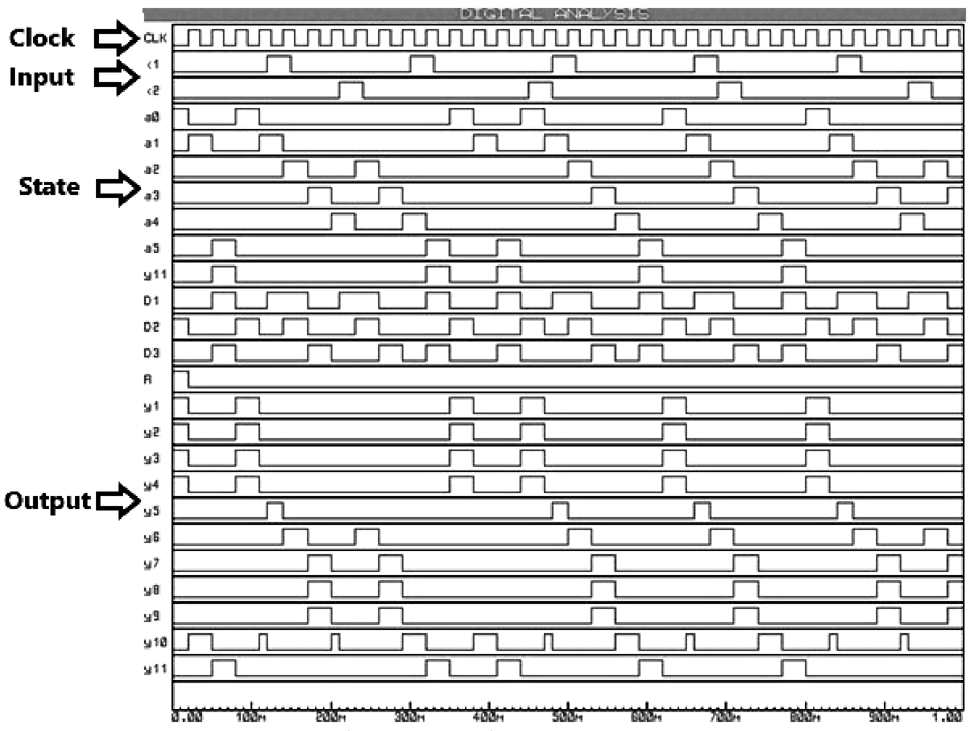
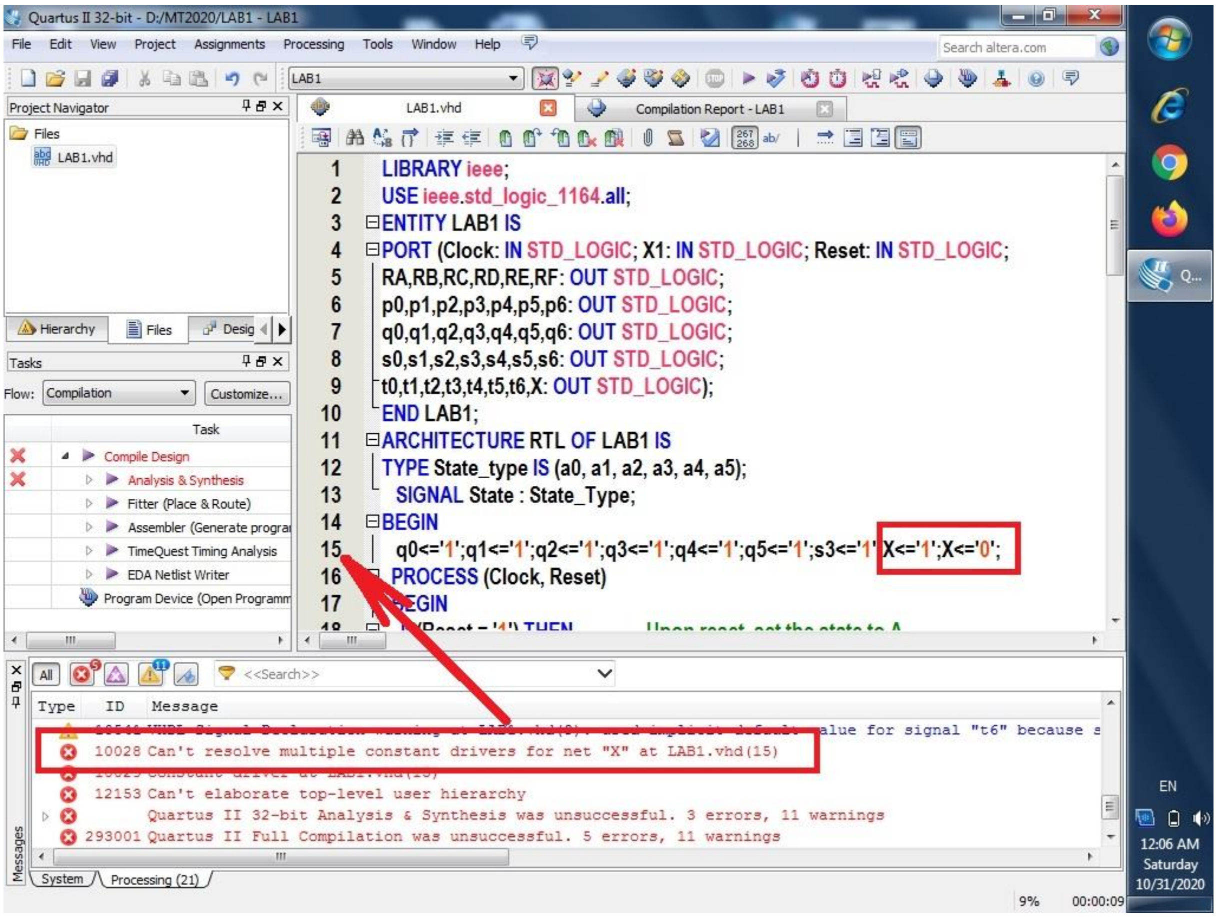


Figure 5. Timing diagrams of the Mealy finite-state control machine.

The functional diagram of the finite-state control machine is presented in Fig. 4. As the elementary base a family of 7400-series integrated circuits is used. Time diagrams are presented in Fig. 5. They show which signals are supplied to the control machine and which are sent to the operating machine for execution. The functional diagram of the datapath is quite complex and big, beyond the scope of the article. In the course of "Microprocessor engineering" students have the opportunity to implement both finite-state datapath and control machines by the means of the languages of description VHDL and System Verilog. Therefore, the student should formulate a task to synthesize according to his version a complex scheme that includes a control and operating machine, check its functionality in the CAD-environment Proteus Design Suite and then translate the synthesized complex scheme in both VHDL and System Verilog programming languages. It allows the student to program the FPGA controller with a description of its circuit. As an example, in the work the variant of the Mealy finite-state control machine with six states and logic of transitions between the states, described by the graph in fig. 3. The beginning of the VHDL-code of this machine is shown in Fig.6.

4. Hardware Implementation

Formalized hardware description language is fundamentally different from the classical programming language in that operators are not executed sequentially.At the same time it describes the logic of all signals of the electronic circuit, and consequently the logic of the device as a whole.



```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY LAB1 IS
PORT (Clock: IN STD_LOGIC; X1: IN STD_LOGIC; Reset: IN STD_LOGIC;
RA,RB,RC,RD,RE,RF: OUT STD_LOGIC;
p0,p1,p2,p3,p4,p5,p6: OUT STD_LOGIC;
q0,q1,q2,q3,q4,q5,q6: OUT STD_LOGIC;
s0,s1,s2,s3,s4,s5,s6: OUT STD_LOGIC;
t0,t1,t2,t3,t4,t5,t6,X: OUT STD_LOGIC);
END LAB1;
ARCHITECTURE RTL OF LAB1 IS
TYPE State_type IS (a0, a1, a2, a3, a4, a5);
  SIGNAL State : State_Type;
BEGIN
  q0<='1';q1<='1';q2<='1';q3<='1';q4<='1';q5<='1';s3<='1' X<='1';X<='0';
 PROCESS (Clock, Reset)
  BEGIN
    IF (Reset = '1') THEN          Upon reset, set the state to A
```

```
Type  ID     Message
      10541  VHDL Signal Declaration warning at LAB1.vhd(9): used implicit default value for signal "t6" because s
      10028  Can't resolve multiple constant drivers for net "X" at LAB1.vhd(15)
      10029  Constant driver at LAB1.vhd(15)
      12153  Can't elaborate top-level user hierarchy
             Quartus II 32-bit Analysis & Synthesis was unsuccessful. 3 errors, 11 warnings
      293001 Quartus II Full Compilation was unsuccessful. 5 errors, 11 warnings
```

Figure 6. VHDL code Compiling Error of in Altera Quartus II CAD environment v13.0 SP1.

For example, if in the classical C-programming language two operators X = 1 are executed consecutively; X = 0; then the second operator ignores the result of the first statement and the variable X gets a final value of 0. Conversely, if the operators swap, the final value of the variable X will be 1. But if these two operators write in the hardware description language, where the first will set the signal

X units, and the second operator - logical zero, then for the CAD-environment the sequence of operators does not matter and in both cases it will return the compilation error "CAN'T RESOLVE MULTIPLE CONSTANT FOR 'X'" (Fig. 6). Only one of these statements must be transmitted for successful compilation.

```
. . . . . . . . . . . . . . . . . . . .
PROCESS (Clock, Reset)
  BEGIN
    If (Reset = '1') THEN
        State <= a0;
    ELSIF falling_edge(Clock) THEN
        CASE State IS
                WHEN a0 =>
                 State <= a1;
                WHEN a1 =>
                   IF X1='1' THEN
                        State <= a2; t2<='1'; t5<='1';  t1<='0'; t4<='0';
                        p0<='1';p1<='0';p2<='0';p3<='1';p4<='1';p5<='1';p6<='1';
                        ELSE
                        State <= a5;  t1<='1'; t4<='1'; t2<='0'; t5<='0';
                        p0<='0';p1<='0';p2<='0';p3<='0';p4<='0';p5<='0';p6<='1';
                END IF;
                . . . . . . . . . . . .
        END CASE;
    END IF;
  END PROCESS;
. . . . . . . . . . . . . . . . . . . .
```

In particular, the new value type a0-a5 is declared in line 12, and the internal signal State is declared in line 13 , which can get only one of the six declared values. Signals p0-p6, q0-q6, s0-s6, t0-t0 are used for visual output of information on the first, second, third and fourth seven-segment indicator, respectively, RA-RF signals - to indicate the status of the machine using green LEDs (Fig.7). From state a1 the finite-state machine has two conditional transitions to state a2 at the value of the input signal X1 = 1 with the generation of the output signal Y5, as well as to state a5 with the generation of the output signal Y10, as shown in the transition graph (Fig.3).
From the preceding fragment of the VHDL code it is indicated that the transition from state a0 to state a1 occurs unconditionally, only on the falling edge (falling_edge) of the Clock signal and at zero value of the Reset signal. On the contrary, from state a1 there is a transition to state a2 at X1 = 1 (Fig. 7a) or to state a5 at X1 = 0 (Fig. 7b).

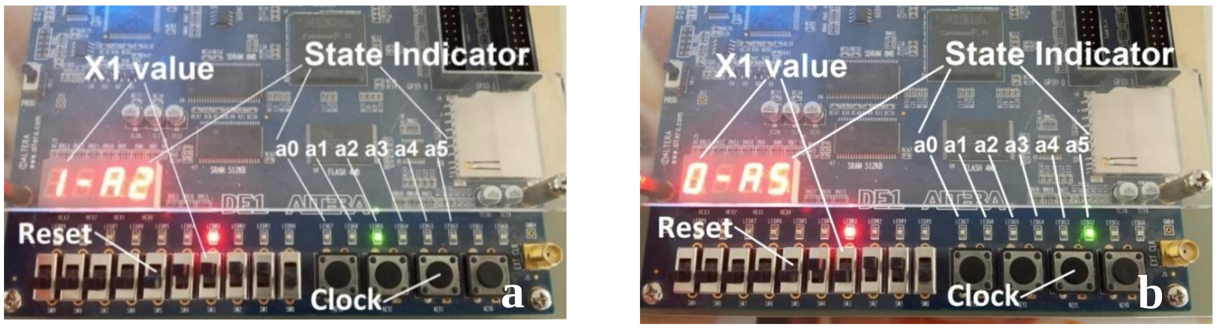


Figure 7. Demonstration of the functioning of the Mealy finite-state control machine implemented in the FPGA-controller Altera Cyclone II. The transition is from state a1 to state a2 (Fig. a) and from state a1 to state a5 (b).

To simplify the VHDL code, which must be written, compiled and run on the FPGA-controller during student's assignment, the signal Clock transition between states is generated manually by one of the four buttons of the controller (Fig.3). Modern FPGA-controllers also have built-in clock generators signals of almost arbitrary frequency, compatible in order of magnitude with the clock frequency of modern processors (109 Hz) and as a result of connecting the Clock signal to such a clock generator can synthesize a real functional circuit with very high speed digital finite-state machines that respond to input for several nanoseconds signals, performs a cycle of transitions between states and generates output signals.

This means that the control and operating circuit synthesized by the student and which runs on an FPGA controller is a universal blank for the development of new computing devices, including microprocessors with new specific instructions. This is the deep applied content of the problem of synthesis of such circuits, which include digital Moore or Mealy finite-state machines. Such specific computing devices are necessary to control complex devices not manufactured in series but in single copies, such as space satellites or devices operating in conditions of huge pressure (sea depths), high temperatures (volcanoes), ionizing radiation (reactors).

Even if the student or teacher does not have an FPGA controller, the Altera Quartus II integrated environment includes a built-in powerful CAD simulator ModelSIM, which analyzes the operation of the circuit, successfully compiled in the form of VHDL or System Verilog program, can simulate all possible modes of the electronic circuit. Similar to the Proteus Design Suite simulator, it also checks the correct operation and signal generation of the circuit.

Conclusions

An electronic circuit development is a complex technological procedure that contains a sequence of certain steps, including simulation of the circuit. As example of such procedure the article describes the designing steps of finite-state machine that performs the algebraic expression calculation.

The designed circuit was modeled using the Proteus Design Suite program, as well as compiled in the VHDL hardware description language and run on an Altera Cyclone II FPGA controller.

Developed remote laboratory work allows students to use modern CAD packages to design logical electronic circuits and perform computational experiments.

Created and proposed assignments for laboratory work in various disciplines, provide students with the opportunity of consistently expanding the functionality of the designed devices.

Laboratory works help students to master experience in the development and debugging of electronic circuits with a given functionality.

References

[1] Motahhir S Chalh A El Ghzizal A Sebti E G and Derouich A 2017 Modeling of Photovoltaic Panel by using Proteus *Journal of Engineering Science and Technology Review* **10(2)** 8-13

[2] Chaikivskyi T Sus B Bauzha O and Zagorodnyuk S 2020 Multicomponent analyzer of volatile compounds characterization based on artificial neural networks *CMIS-2020* **2608** 819-31

[3] Shamonia V H Semenikhina O V Proshkin V V Lebid O V Kharchenko S Ya and Lytvyn O S 2019 Using the Proteus virtual environment to train future IT professionals EducDim **53** 181-98

[4] Klimenko X A 2020 The development of digital device for current, voltage, power measuring and simulation results in proteus environment *J. Phys.: Conf. Ser.* **1515** 052050

[5] Mukherjee A Ray S and Das A 2014 Development of Microcontroller Based Speed Control Scheme of BLDC Motor Using Proteus VSM Software *IJEEE* 1-7

[6] Chaikivskyi T Sus B Tmienova N Bauzha O S and Zagorodnyuk S P 2019 3D simulation of virtual laboratory on electron microscopy *CEUR Workshop Proceedings* **2533** 282-91

[7] Tatar G Bayar S and Alkan M 2019 FPGA Based Step Motor Control For Solar Panels *2019 IEEE 13th International Conference on Application of Information and Communication Technologies (AICT)* 1-5

[8] Tang Z and Chen X 2021 VHDL Design of Motor Control Module on FPGA *Application of Intelligent Systems in Multi-modal Information Analytics* **1234** 237-45

[9] Otmani R and Benmoussa N 2020 Micro Electro Mechanical Systems Modeling by VHDL-AMS: Application to a Piezoresistive Pressure Sensor *Mechatronics 4.0* 31–41

[10] Bauzha O Sus B Zagorodnyuk S and Stuchynska N 2019 Electrocardiogram Measurement Complex Based on Microcontrollers and Wireless Networks *IEEE International Scientific-*

Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T) 345-9

[11] Ayeh E Agbedanu K Morita Y Adamo O and Guturu P 2008 FPGA Implementation of an 8-bit Simple Processor *2008 IEEE Region 5 Conference* 2008 1-5

[12] Dudhane T M 2020 Architectural Enhancement of Processor with 8 Bit Multiplier and 16 Bit Co-operative ALU using VHDL *IJETER* **8** 2977–83

[13] Kung Y-S Risfendra 2016 ModelSim/Simulink co-simulation of a sensorless control for PMSM drives based on I-F startup and EKF *International Conference on Applied System Innovation (ICASI)* 1–4

[14] Shen L Zhou G and Zhai J Li H 2019 The Waveform Generator and Monitor Based on FPGA&DSP for the Pulsed Power Supply FSDM

[15] Kumar A Kumar P Modi S and Nath V 2021 Study and Implementation of Ladder Logic Conversion to VHDL for Field Programmable Gate Array (FPGA)-Based Programmable Logic Controllers (PLC) *Proceedings of the Fourth International Conference on Microelectronics, Computing and Communication Systems* **673** 31–42

[16] Garapati P and Musala S 2020 Moore and Mealy Negative Edge detector A VHDL Example for Finite State Machine *2020 International Conference on Communication and Signal Processing (ICCSP)* 1159–61