Modular System for Process Control Testing

Ioana FAGARASAN, S. St. ILIESCU, Iulia DUMITRU, Nicoleta ARGHIRA, I. BUCUR*
Department of Automatic Control and Industrial Informatics,
*Department of Computer Science,
University “POLITEHNICA” of Bucharest
Splaiul Independentei 313, 060042 Bucharest
ROMANIA
{ioana, iliescu}@shiva.pub.ro; dumitru.iulia@yahoo.com; arghira.nicoleta@gmail.com,
ion.bebe.bucur@gmail.com

Abstract: - A modular system for processes simulator and control diagram implementation can replace a vast number of expensive systems used to simulate, to design and to test the programming of programmable logical control (PLC) systems graphically and practically. Different processes are simulated and associated with different types of control application and with different complexity factors in order to develop control algorithms and implement them into the PLC unit. The process and control simulator is hardware and software intergraded system for multifunctional educational platforms that enables training and practical work.
The system’s modularity allows an easy development of the system and can be use in future development by adding FPGA (Field-Programmable Gate Array) components.

Key-Words: - electrical process, mechanical process, system simulator, control system design, PLC

1 Introduction

The design and testing of automatic control for some industrial processes represents modern and efficient direction for the multifunctional testing platforms designed for students training laboratories as well as for complex development of some technologic scenario for improvement and conception of control scheme in modern technologies. [6]
The majority of control and monitoring systems used in industry contains PLC units.
Therefore is important to study, develop and test their functioning.
The most of the simulators are composed of the command segment – the PC with an interface or a PLC – and the process simulator that is a micro plant or an electronic circuit.
The process simulator that is presented in this paper is hardware and software intergraded system.
A programmable logic controller (PLC) or programmable controller is a standard unit with no dedicated application [4,5,16]. The unit has to be connected to the various process input and output devices. The controller must be programmed with the tasks that the system must perform; this is achieved by a set of software instructions, [11]. The software contains program sequences that are initiated by inputs from the process, which then prompt the outputs to change the plant status [12].
The process control system has a hardware structure that assures stability, accuracy and good transitions. These performances are realized by interconnection of PLCs and a central digital language. The PLC is used for automation of industrial processes and replaces the circuits of sequential command in cable logic.

2 Control System and Process Simulator

The Simulator is sectionalized containing three parallel panels: *the PLC*, *a demonstration panel* to hold a mask (an applied mask corresponding to different processes) with a large image of the process that is being simulated and the connection panel between the PLC connection and *the demonstration panel*.

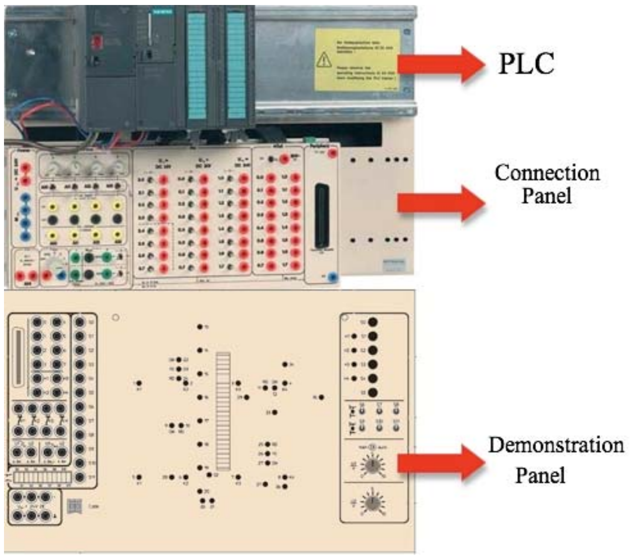


Fig. 1. The Process Simulator

The connected and correspondingly programmed PLC system allows the user to study automatic functions, for which even respective signals are generated automatically, while controlling the actuators. These automatic functions depend on the assignment process.
The large range of assignments allows you to study the field of automation without previous knowledge starting with small control circuits and also up market solution strategies by using the full scale of functions offered by the programmable logical control systems. The individual examples are obtained with masks that contain a process flow chart and are applied to the demonstration panel of the process simulator.
Three categories of exercises were developed:

- Projection and testing of the simplest assignments using classical contactor circuits
- Projection and testing of the medium assignments using industrial system models [2]
- Projection and testing of the complex assignments using industrial system models with the incorporation of the analog function [3]

2.1 The PLC component

All the inputs and outputs on the Simulator have been designed to meet industrial standards for control engineering: 24 V DC for digital inputs/outputs and from 0 to 10 V DC for analog inputs and outputs.
As a result, the simulator can be operated with the programmable logical control systems of all major manufacturers.

2.1.1. The PLC hardware

All the inputs and outputs on the Simulator have been designed to meet industrial standards for control engineering: 24 V DC for digital inputs/outputs and from 0 to 10 V DC for analog inputs and outputs.
As a result, the simulator can be operated with the programmable logical control systems of all major manufacturers. For the PLC it is used the SIMATIC S7-300 modules, an industrial PLC, with a system which is able to offer real industrial conditions during the entire training and testing actions.[7,8]

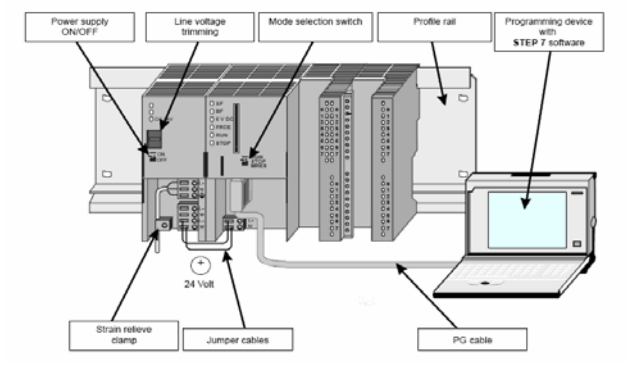


Fig. 2. The PLC

The S7-300 is a modular control system for system solutions with a main emphasis on production engineering [15]. The wide ranges of assemblies are ideally suited to cope with all required demands and a flexible application. The S7-300 contains the following types of modules:

- Central processing units (CPU)
- Power supply modules (PS)
- Interface modules (IM)
- Communications processors (CP); (for connecting to PROFIBUS)
- Function modules (FM); (for counting, positioning, closed-loop control)
- Digital and analog modules are now called “signal modules” (SM)

As in real application, the S7-300 modules are plugged into the profiled rail on the basic plate, fastened and connected to each other via the rear panel bus to be system compatible. The system is quickly and safely connected electrically to the inputs and outputs on the modules with the respective connector adapter. Due to its construction, the system is open to future development. The PLC is constructed from following modules: a power supply PS 307 2A, a CPU 313C, digital input module SM 321, digital output module SM 322, analog input/output module SM 334.

2.1.2. The PLC software

The standard software used for configuring and programming Simatic programmable logic controllers is Step 7 [7]. The programming languages for S7-300 are Ladder Logic, Statement List, and Function Block Diagram.
Ladder Logic Diagram (or LAD) is a graphic representation of the STEP 7 programming language. Its syntax for the instructions is similar to a relay ladder logic diagram: Ladder allows you to track the power

flow between power rails as it passes through various contacts, complex elements, and output coils.

A ladder diagram consists of 2 vertical lines called bus bars or power rails, and horizontal lines, called rungs or logic lines. The rungs contain one or several normally open contacts(-||-), and normally closed contacts (-|/|-) and at their right hand end, immediately before the right hand bus bar, an instruction that could be one of the commonly used PLC instructions.

The horizontal rungs are sometimes also called branching lines. These branching lines may be a singular rung, or they may be used in "OR" connection as parallel rungs, or they may form interim blocks of "OR" circuits. When necessary, such "OR" blocks may be "AND" connected into complex subcircuits. The rungs can branch apart, and they can join back together. Furthermore, such rugs may branch out into several output instructions. When logically "true", these subcircuits enable the output instruction is executed. The complete circuit arrangement ending in an output instruction is called a logic line or network.

Statement List (or STL) is a textual representation, similar to machine code. Its syntax for statements is similar to assembler language and consists of instructions followed by addresses on which the instructions act.

If a program is written in Statement List, the individual instructions correspond to the steps with which the CPU executes the program. To make programming easier, Statement List has been extended to include some high-level language constructions (such as structured data access and block parameters).

With reference to structure, an instruction statement belongs to either one of the following two basic groups, as shown in table 1 :

– A statement made up of an instruction alone(for example, NOT);
– A statement made up of an instruction and an address.

The address of an instruction indicates a constant or the location where the instruction finds a value (data object) on which to perform an operation. The address can have a symbolic name or an absolute designation. The address can point to any of the following items:

– A constant, the value of a timer or counter, or an ASCII character string to be loaded into accumulator 1, table 2;
– A bit in the status word of the programmable logic controlle, table 3;
– A symbolic name, table 4;
– A data block and a location within the data block area, table 5;
– A function (FC), function block (FB), integrated system function (SFC), or integrated system function block (SFB) and the number of the function or block; table 6;
– An address identifier and a location within the memory area that is indicated by the address identifier.

Table 1. Basic Groups of statements

Statemenet group 1	Statement group 2
Instruction alone	Instruction + adress

Table 2. Adresses that point to a value or character string

Statement		Description
Instruction	Adress Constant	
L	+27	Load the integer 27 into accumulator 1
L	\`END\`	Load the ASCII characters \`END\` into accumulator 1

Table 3. Adresses that refer to a bit in the status word

Statement		Description
Instruction	Adress Bit in the status Word	
A	BR	The 1 or 0 in bit 8 of the status word is included in a Boolean logic combination.
A	UO	The instruction interprets the bit combination that it finds in bits CC 1 and CC 0 of the status word to see if if a certain condition has been fulfilled. For example, a combination of 1 and 1 indicates "unordered" that is, one of the values in a floating-point operation was not a valid floating - point number.

Table 4. Adresses that point to a symbolic name

Statement Instruction	Adress Symbol	Description
A	Motor.On	Perform an And logic operation on the bit whose symbolic name is "Motor.On". In this case, the symbolic name "Motor.On" can only represent a bit in the data block (D) area of memory or element of a structure "MOTOR" .
L	Speed	Load the byte, word, or double word value, whose symbolic name is SPEED, into accumulator 1 .

Table 5. Addresses that point to a data block and a location within the data block

Statement Instruction	Adress Data block and location	Description
L	DB4.DBD10	Load data double word DBD10 from data block DB4 into accumulator 1.
A	DB10.DBX4.3	Perform an And logic operation on data bit DBX4.3 from data block DB10.

Table 6. Addresses That Point to a Function, Function Block, System Function, or System Function Block

Statement Instruction	Adress FC, FB, SFC, SFB and number	Description
CALL	FB10, DB10	Call function block FB10 with instance data block DB10.
CALL	SFC43	Call integrated system function SFC43.

Function Block Diagram (FBD) is a graphic representation and uses the logic boxes familiar from Boolean algebra to represent the logic. Complex functions (for example, math functions) can be represented directly in conjunction with the logic boxes. Inputs and outputs of the blocks are wired together with connection lines, or links. Single lines may be used to connect two logical points of the diagram:

- An input variable and an input of a block;
- An output of a block and an input of another block;
- An output of a block and an output variable.

It is possible to disable part of the FBDs using enables. These are available for each function block but may not be displayed. Figure 3 shows an XOR calculation. Both of the Boolean AND functions have the enable inputs connected to 'enable'. If 'enable' is true, then the system works as expected and the output 'X' is the exclusive OR of 'A' and 'B'. However if 'enable' is off then the BAND functions will not operate. In this case the 'enable' input is not connected to the BOR function, but because it relies on the outputs from the BAND blocks, it will not function, and the output 'X' will not be changed.

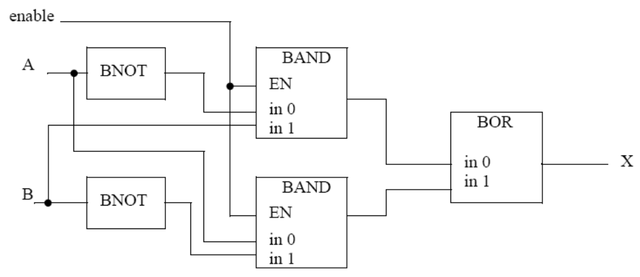


Fig. 3. Using Enables in FBDs

A FBD program is constructed using function blocks that are connected together to define the data exchange [9]. The connecting lines will have a data type that must be compatible on both ends. The inputs and outputs of function blocks can be inverted. This is normally shown with a small circle at the point where the line touches the function block, as shown in Figure 4.

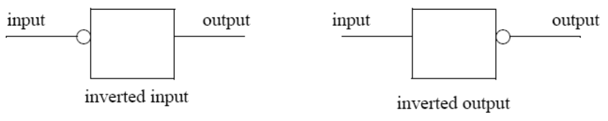
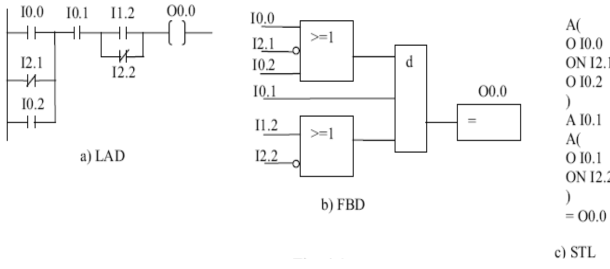


Fig. 4. Inverting Inputs and Outputs on Function Blocks

The basic functions used in FBD programs are equivalent to the basic set used in Structured Text (ST) programs.

FBD is most useful in those applications involving a high degree of information/data flow between control components, such as process control. A comparison between the three programming modes is shown in figure 5.



```
A(
O I0.0
ON I2.1
O I0.2
)
A I0.1
A(
O I0.1
ON I2.2
)
= O0.0
```

Fig. 5. Comparison between the three programming languages for S7-300

The programming languages are organized in different industrial standards, such as IEC 61131, IEC 61499, as shown in [10].

2.2 The connection panel

On the connection panel are presented 4 variable analog input voltage of 0 to 10 V DC, external analog inputs sockets to supply analog signals, e.g. from the sensors to the PLC system, two analog outputs, 20 rocker switches to simulate digital input signals, 16 outputs to access the digital output signals. All the inputs and outputs run to 2-mm or 4-mm safety sockets. It can be also seen rocker switches to either select the analog input signal via the analog simulation or via the external analog inputs or the external analog inputs via the 50-pin socket. The connection panel contains also an input to supply the external power voltage supply of 24 V to the outputs. When the selector is set to “int” position, the digital output signals are supplied with the voltage from the PS. When the selector is on “ext” position, it is possible to reach the maximum switching capacity for the output stages via an externally supplied DC voltage of 24 V.

2.3 The demonstration panel

The demonstration panel is operated with a stabilized DC voltage of 24 V, e.g. supplied with the power supply unit of the PLC system. The sockets are used to connect the sensors and switches on the system simulator to the PLC system. The safety sockets on the system simulator are connected to the sensors and switches according to the assignment. Each of these switches and sensors can physically act as make or break contact depending on the setting of the respective selector with the same marking. Sockets 0 to 7 are used to connect the actuators on the applied mask, e.g. relay and contactor coils to the PLC outputs.

The operating state of an actuator is indicated with an LED assigned to the symbol. Sockets H1 to H4 are directly connected to the signal lamps or LEDs H1 to H4 on the control panel.

All the switches, signal lamps and potentiometers required for operating the control circuit illustrated on the applied mask are clearly arranged on the control panel. Only the control elements labelled on the control panel are visible, corresponding to the applied mask.

The demonstration panel contains LEDs to indicate the operating states of the actuators, such as relay and contactor coils. The LED strip, for example, clearly indicates filling levels and motions. The PLC can only control the LEDs or the segments of the LED strip, which are assigned to the actuators on the applied mask.

3 Case studies

In this section there are described two processes that are simulated using the simulator. As different types of processes can be used, an electrical and a mechanical process are described using the current flow chart and the Step 7 program.

3.1 The electrical process

From the first specified category of applications in section 2, a case study regarding a three-phase squirrel-cage asynchronous motor starting in star-delta connection is presented in this selection [14]. The characteristic and the operative of the process is presented and depicted in a mask as in figure 6.

In order to build the command procedure for this process, some steps need to be followed: drawing of the pushbuttons and the sensor into the PLC terminal diagram (figure 7), drawing of the current flow chart for the control unit according to the assignment (figure 8), the process chart for the assignment (figure 9), the PLC program designed according to the current flow. The final step is downloading the program into the PLC and then starting the simulation.

The characteristic and the operative of the process is presented and depicted in a mask as in figure 6.

3.1.1 The assignment

A three-phase squirrel-cage asynchronous motor starts in star connection in order to avoid high starting currents and then automatically changes over to delta connection after the running up period.

The motor starts running in star connection when the start button S1 is activated. After a running period of approximately 10 s, the circuit is automatically changed over to delta connection.
The pilot lamp H1 indicates the operating state of the asynchronous motor. The activation of the motor protective relay is simulated by pressing pushbutton S5.

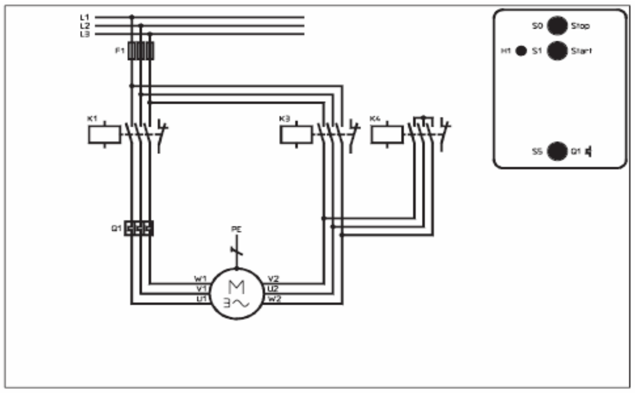


Fig. 6. The mask for the assignment automatic star-delta connection

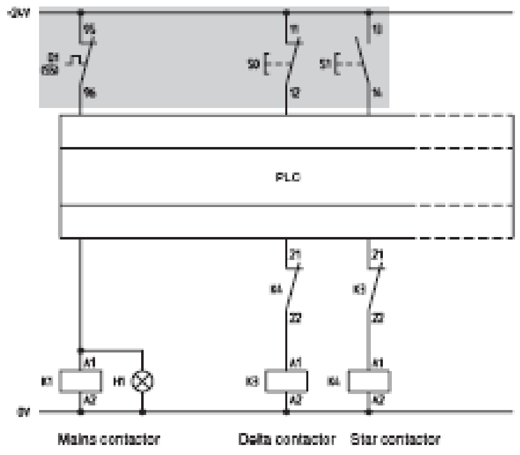


Fig. 7. The PLC terminal diagram

3.1.2 The solving procedure

The first step is to draw the contacts into the PLC terminal diagram considering the requirements for fail-safe circuits. The drawing of the current and the process flow chart is the next step, followed by the setting up of the assignment list according to the functional description and the used PLC. Next, the PLC program must be set up using the current or the process flow chart according to the functional description. The program will be downloaded into the PLC and then simulated.

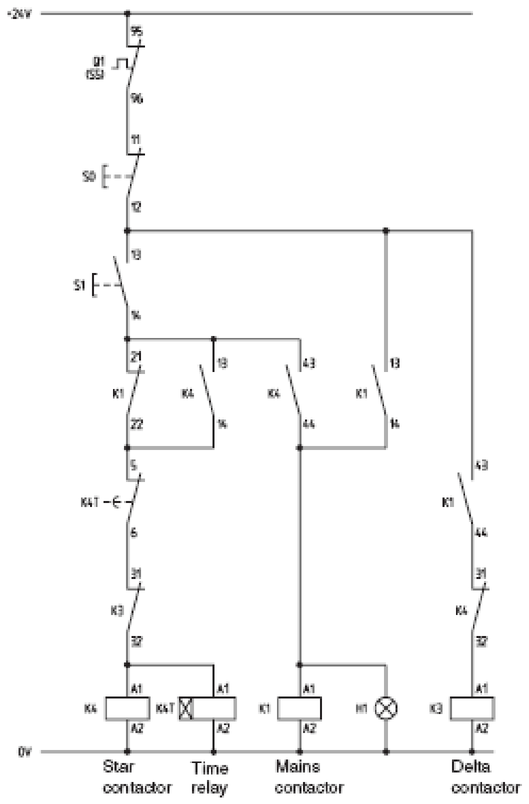


Fig. 8. The current flow chart for the control unit

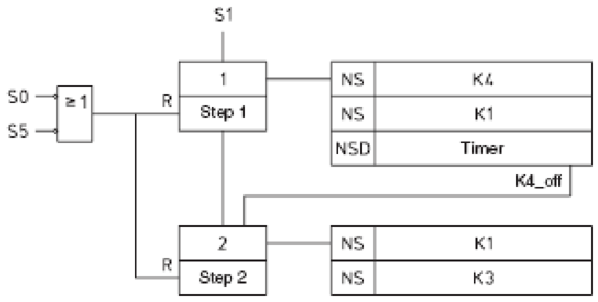


Fig. 9. The process chart for the assignment

The used PLC program is STEP 7. In table 7, the program using the Statement List Language is shown:

Table 7. The Step 7 Program

```
Network 1:  Description 1 on
A     ”S1”  I124.1   --Start
Network 2: Description 1 off
O      #Step 2
ON   ”S5”   I124.5   -- Motor protection
ON   ”S0”   I124.0   -- Stop
R      #Step 1
Network 3: Description 2 on
A    #Step 1
A    ”K4_off”   M1.0   --Auxiliary protection
S     #Step 2
Network 4: Description 2 off
ON   ”S5”   I124.5   -- Motor protection
ON   ”S0”   I124.0   -- Stop
R      #Step 2
Network 5: Star contactor
A     #Step 1
=     ”K4”   Q124.3   -- Star contactor
Network 6: Main contactor
O     #Step 1
O     #Step 2
=     ”K1”   Q124.0   -- Main contactor
Network 7: Star contactor
A    #Step 2
=     ”K3”   Q124.2   --
Network 8: Time delay relay
A     ”K4”   Q124.3   -- Star contactor
L     S5t#10s
SD  ”K2T”  T1          -- Time relay
NOP  0
NOP  0
NOP  0
A   ”K2T”      T1     -- Time relay
=    ”K4_off”  M1.0 -- Auxiliary protection
```

3.2 The mechanical process

From the specified category of exercises in section 2, a case study regarding the conveyor belt system is presented in this section.
In order to build the command proceed for the process, some steps need to be followed: the drawing the pushbuttons and the sensor into the PLC terminal diagram, the construction of the truth table and the Karnaugh-Veitch map, the drawing of the current flow chart according to the Karnaugh-Veitch map, the PLC program designed according to the current flow. The final step is downloading the program into the PLC and then his simulation.
The characteristic and the operative of the process is presented and depicted in a mask as in figure 10.

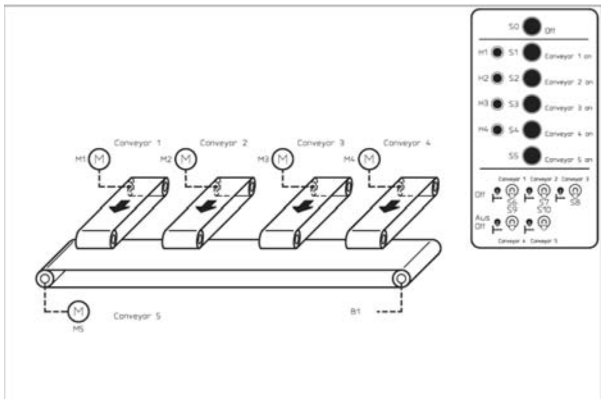


Fig. 10: The mask for the assignment collecting belt conveyor

3.2.1 The assignment

The collecting belt conveyor can be loaded via four feeding conveyors. The feeding conveyors must only be connected or switched on when the conveyor monitor indicates a correct closed circuit condition for the collecting belt conveyor.
The motor 2 automatically generates the B1 sensor signal “high” after the motor has started running and stays until the motor is switched off. The overload protection for the motors has not been taken into account in the following exercise.
To prevent the collecting belt conveyor from being overloaded, only two feeding conveyors must be switched on simultaneously. If two conveyors load the collecting belt conveyor simultaneously, the connection of the other two conveyors must be prevented with an appropriate lock. The closed circuit condition of the four feeding conveyors must be indicated with the pilot lamps H1 to H4.

3.2.2 The solving procedure

First it will be establish the maximum connection of two feeding conveyors, then it will be drawn the pushbuttons and the sensor into the PLC terminal diagram considering the requirements for fail-safe circuits. The next step is to complete the truth table accordingly with the assignment. Then, it will be designed the function equation for the contactor function by using the truth table (table 8) and completing the Karnaugh-Veitch map (KV map), figure 11, with minimized connections [17,18]. The drawing of the current flow chart (figure 12) according to the minimized function equation is the next step, followed by the setting up of the assignment list according to the functional description and the used

PLC. The PLC program will be designed according to the current flow. The program in table 9 will be downloaded into the PLC and then simulated.
The same steps can be followed for different applications.

Table 8. Truth table

K1	K2	K3	K4	K10
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	*
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	*
1	1	0	0	1
1	1	0	1	*
1	1	1	0	*
1	1	1	1	*

*Don't care fields, i.e. input signal combinations, which cannot occur during an uninterrupted sequence of functions. If these fields are included in minimization loops, then the circuit complexity is reduced.

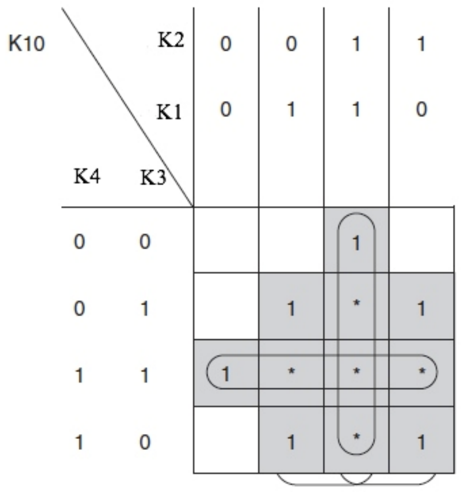


Fig. 11: The Karnaugh-Veitch map

K10 = K1*K2+K1*K3+K2*K3+K3*K4+K1*K4+K2*K4
= K1(K3+K4)+K2(K3+K4)+K1*K2+K3*K4
= K1*K2 + K3*K4 + (K3+K4)(K1+K2)

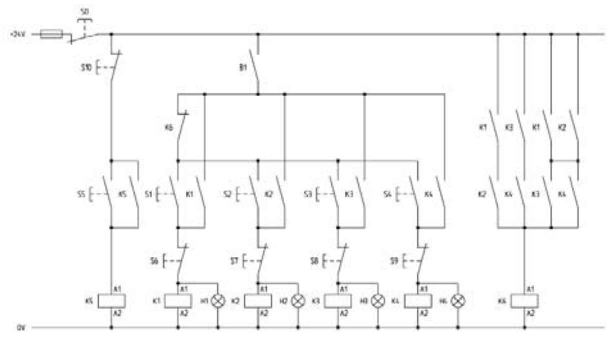


Fig. 12: The current flow chart

4 Conclusion

The System Simulator can be used for vocational training in order to practice programming of PLCs. It can be used for testing and analyzing command diagrams before the actual deployment. The PLC systems can simulate the real processes with several masks, as shown in the case studies. PLCs have the great advantage that the same basic controller can be used with a wide range of control systems. To modify a control system and the rules that are to be used, all that is necessary is for an operator to set a different set of instructions. There is no need to rewire. PLCs are optimised for control and command tasks and the industrial environment [13].
An important aspect that has to be mentioned is the system's modularity. This characteristic allows an easy development of the system and can be use in future development by adding FPGA (*Field-Programmable Gate Array*) components. In this case the commands procedure or process simulator could be implemented using FPGA technology [1] and the system will be used for training and testing using both technologies PLC and FPGA.
The presented simulator has a hardware, as well as a software component, used for better transitions and stability during process functioning. The software component allows programming for different control assignments using one of the three languages: Ladder Logic, Statement List, and Function Block Diagram.
The simulator is an "open" equipment as it can be developed further on future technologies.

Table 9. The Step 7 program (The Statement List Language)

```
Network 1: Collector Band
O     "S5"    I 124.5 -- Band 5 on
O     "K5"    Q124.4 -- Band 5
A     "S10"   I 125.2 -- Band 5 of
A     "S0"    I 124.0 -- OFF
=     "K5"    Q 124.4-- Band 5
Network 2: Band Activation
A     "K5"    Q 124.5-- Band 5
L     S5T#2s
SD    "Timer1"       T1
A     "Timer1"       T1
=     "B1"
Network 3: Protection
A     "K1"    Q 124.0-- Band 1
A     "K2"    Q 124.1-- Band 2
O (
A     "K3"    Q 124.2-- Band 3
A     "K4"    Q 124.4-- Band 4
 )
O (
O     "K1"    Q 124.0-- Band 1
O     "K2"    Q 124.1-- Band 2
A (
O     "K3"    Q 124.2-- Band 3
O     "K4"    Q 124.4-- Band
)
)
=     "K6"    M1.0    -- Protection
Network 4: Band 1
A     "S1"    I 124.1 -- Band 1 on
AN    "K6"    M1.0    -- Protection
O     "K1"    Q 124.0-- Band 1
A     "S6"    I 124.6 -- Band 1 off
A     "B1"    I 125.3 --Collector Band
A     "S0"    I 124.0 -- Off
=     "K1"    Q 124.0-- Band 1
=     "H1"    Q 125.0-- Pilot lamp
Network 5: Band 2
A     "S2"    I 124.2 -- Band 2 on
AN    "K6"    M1.0    -- Protection
O     "K2"    Q 124.1-- Band
A     "S7"    I 124.7 -- Band 2 off
A     "B1"    I 125.3 --Collector  Band
A     "S0"    I 124.0 -- Off
=     "K2"    Q 124.1-- Band 2
=     "H2"    Q 125.1-- Pilot lamp
Network 6: Band 3
A     "S3"    I 124.3 -- Band 3 on
AN    "K6"    M1.0    -- Protection
O     "K3"    Q 124.2-- Band 3
A     "S8"    I 125.0 -- Band 3 off
A     "B1"    I 125.3 -- Collector Band
A     "S0"    I 124.0 -- Off
=     "K3"    Q 124.2-- Band 3
=     "H3"    Q 125.2-- Pilot lamp
Network 7: Band 1
A     "S4"    I 124.4 -- Band 4 on
AN    "K6"    M1.0    -- Protection
O     "K4"    Q 124.3-- Band 4
A     "S9"    I 125.1 -- Band 4 off
A     "B1"    I 125.3 -- Collector Band
A     "S0"    I 124.0 -- Off
=     "K4"    Q 124.3-- Band 4
=     "H4"    Q 125.3-- Pilot lamp
```

References:

[1] I. Bucur, I. Făgărăşan, C. Popescu, C.A. Boiangiu, G. Culea – *On K-LUT Based FPGA Optimum Delay and Optimal Area Mapping*, Proceedings of de 10th WSEAS International Conference on Mathematic and Computational Methods in Science and Engineering (MACMESE'08), ISBN 978-960-474-019-2, ISSN 1790-2769, November 2008, Bucharest

[2] I. Fagarasan, S.St. Iliescu, I. Dumitru, Process Simulator using PLC technology, *UPB Scientific Bulletin*, Series C: Electrical Engineering, vol. 71, nr. 3, ISSN 1454-234X, 2009

[3] I. Fagarasan, S.St. Iliescu, Applications of Fault Detection Methods to Industrial Processes, *WSEAS TRANSACTIONS on SYSTEMS*, Issue 6, Volume 7, June 2008, ISSN: 1109-2777

[4] D. Popescu, *Automate programabile*, Editura Matrix Rom, 2005.

[5] N. Ivanescu, T. Borangiu, S. Brotac, *Automate programabile Teorie si probleme rezolvate*, Editura Printech, 2002

[6] *** Elwe – Automation, Electrical machines, Power electronics, 2007

[7]. *** Siemens – Programmable Logic Controllers S7-300 Module Data, 2004

[8]. *** Siemens – Central processing units, 1999

[9]. H. Jack, Automating Manufacturing Systems with PLCs, march 2008, http://www.eod.gvsu.edu/~jackh/books/plcs/pdf/plcbook5_1.pdf

[10] FJ.Molina, J. Barbancho, C. Leon, A. Molina, A. Gomez, Using industrial standards on PLC programming learning, Proceedings on MEDITERRANEAN CONFERENCE ON CONTROL & AUTOMATION, Volumes 1-4, Pages 390-395, 2007

[11] M. Dobriceanu, A. Bitoleanu, M. Popescu, S. Enache, E. Subtirelu - SCADA System for

Monitoring Water Supply Networks, *WSEAS TRANSACTIONS on SYSTEMS*, Issue 10, Volume 7, pp. 1070-1079, October 2008,

[12] S. K. Subramaniam, S. H. Husin, S. A. Anas, A. H. Hamidon, Multiple Method Switching System for Electrical Appliances using Programmable Logic Controller, *WSEAS TRANSACTIONS on SYSTEMS and CONTROL,* Issue 6, Volume 4, June 2009

[13] W. Bolton, Programmable Logic Controllers, Elsevier Ed., July, 2006, ISBN-13: 978-0-7506-8112-4, ISBN-10: 0-7506-8112-8, 304 pages

[14] I. Făgărăşan, S. St. Iliescu,I. Dumitru, N. Arghira, I. Bucur, Control System Simulator with PLC, Proceedings of the 11th WSEAS Int. Conf. on Sustainability in Science Engineering, Timişoara, 2009

[15] Costianu, D., Arghira, Nicoleta, Făgărăşan, Ioana, Iliescu, S. St., Power Plants Automation and Control using PLC Tehnology, Proceedings of *Journal ISOM* Vol. 2 No.2, 2008

[16] Frank D. Petruzella, Programmable Logic Controllers, 3rd edition, McGraw-Hill Ed. US 2005, ISBN 0078298520 / 9780078298523, 482 pages

[17] B. Wilkinson, The Essence of Digital Design, Prentice Hall Europe Ed., 1997

[18] G. Toacse, D. Nicula, Electronica digitala – Dispozitive, circuite, proiectare, Ed. Tehnica, bucuresti 2005