SIMATIC "LADDER" PROGRAM FOR A CONVEYOR BELT

Casavela Stelian Valentin, *lecturer dr. eng. University of Petrosani*
Casavela Cristofor, *medic*
Csavela Antonio, *medic*

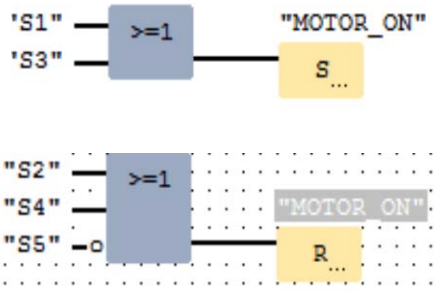
ABSTRACT: *The program may control one conveyor belt or two, using components of conveyor system like motors, push buttons switches and sensors, that are photoelectric barriers, which are designed to detect the direction in which a package is moving on the belt. The Ladder Logic Program is used also for temporary control of the area in between two conveyor belts.*

KEY WORDS: *conveyor, switches, photoelectric, ladder.*

1. BEGINNING OF THE PROGRAM

In Figure 1 it may be seen an image of a conveyor belt , which is moved due electrical power. We designed this system with two switching push buttons at the beginning of the belt: S1 for START and S2 for STOP. We designed also two switching push buttons at the end of the belt: S3 for START and S4 for STOP. We may so start or stop the belt from both ends. Also, the sensor noted with S5, is used to stop the belt when an object from on the belt arrives at the end of the conveyor. A first method of programming would be when we shall write a program to drive the conveyor belt, pictured in Figure 1, by employing symbols that represent the representative components of the conveyance system. Choosing this manner leads us to making a symbol table, used to correlate the defined symbols with absolute values (see Figure 2). The symbols are written in the symbol table of our program.

A second method would be when we shall write a program using absolute values that represent the different components of the conveyance system. Figure 3 comprises the first two sequences (networks) of our ladder logic program, called "Conveyor_belt", combining the two methods and where it may be controlled the starting and stopping of the conveyor belt. The same program sequence could be written in FBD, as below:



The STL code has the same effect.

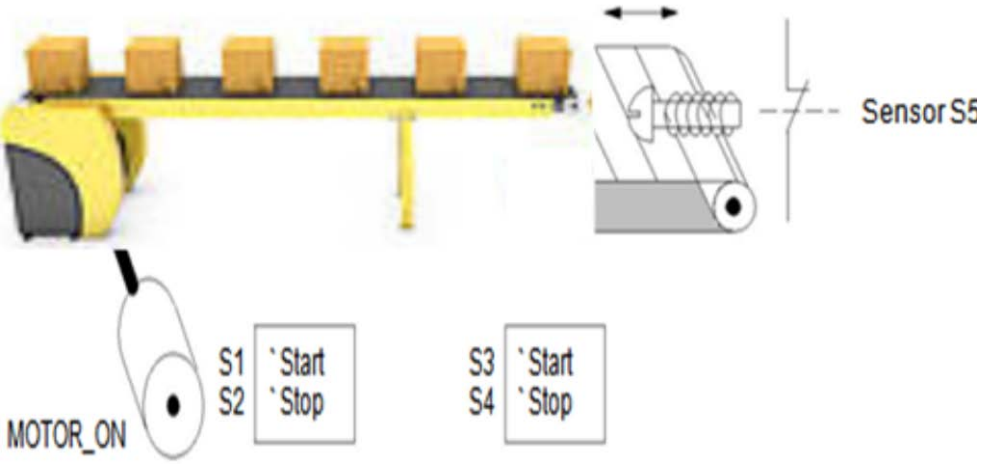
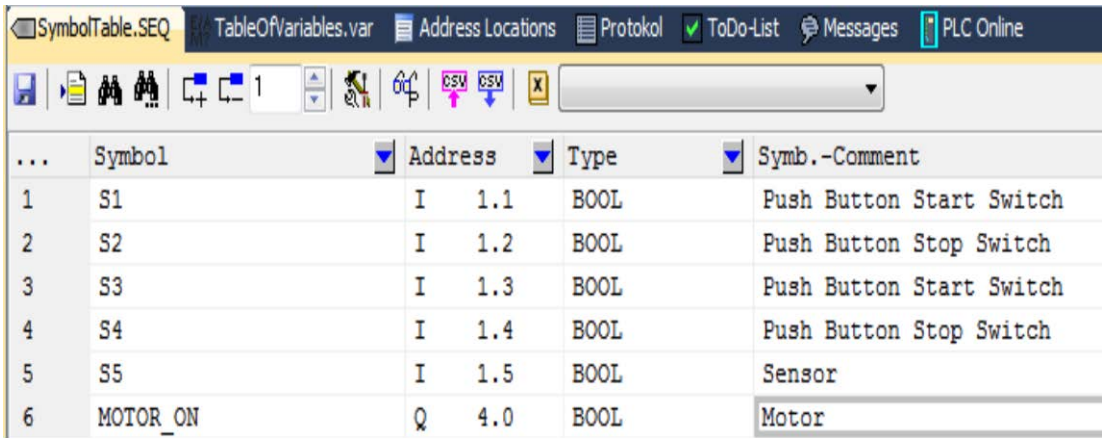


Figure 1



...	Symbol	Address	Type	Symb.-Comment
1	S1	I 1.1	BOOL	Push Button Start Switch
2	S2	I 1.2	BOOL	Push Button Stop Switch
3	S3	I 1.3	BOOL	Push Button Start Switch
4	S4	I 1.4	BOOL	Push Button Stop Switch
5	S5	I 1.5	BOOL	Sensor
6	MOTOR_ON	Q 4.0	BOOL	Motor

Figure 2

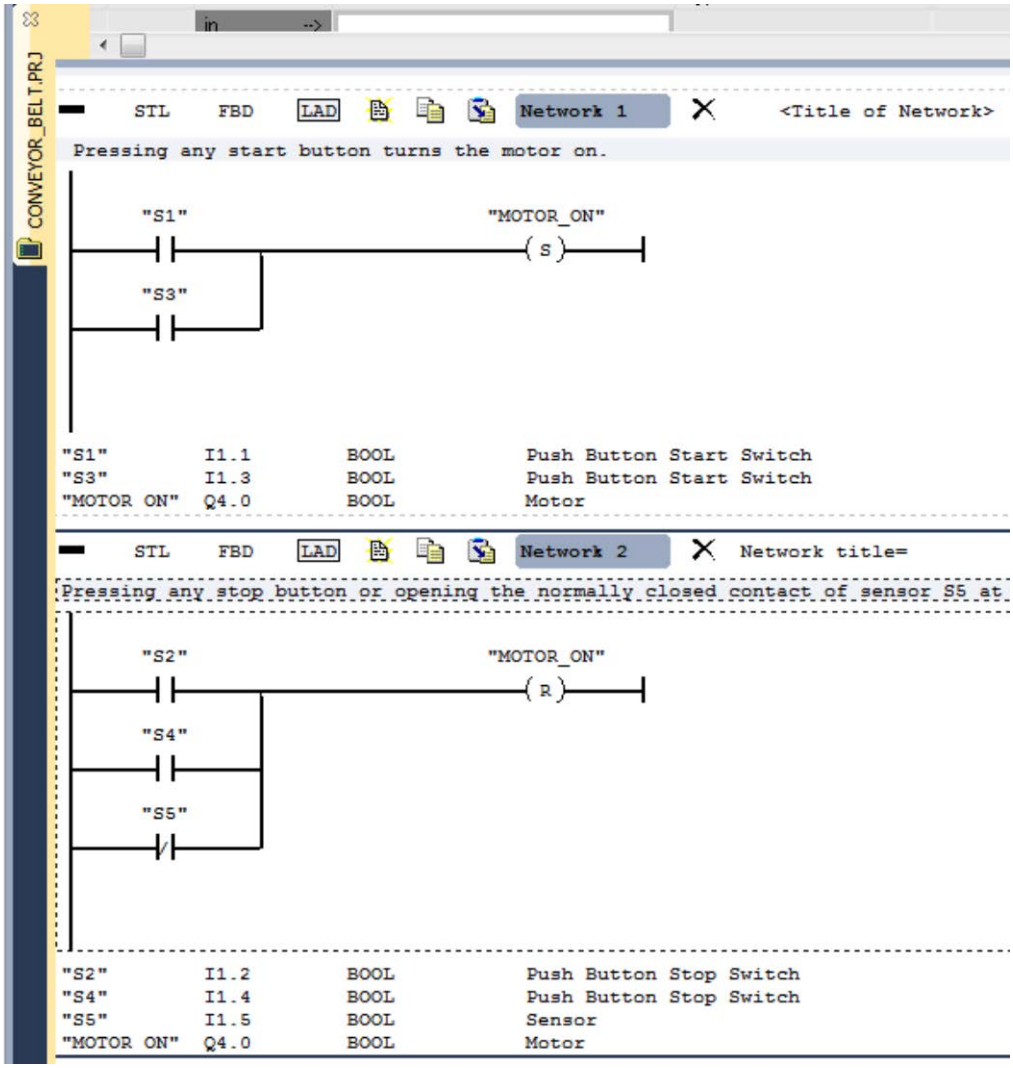


Figure 3

CHAPTER 2. DEVELOPING OF THE PROGRAM

Figure 4 presents a conveyance system, which is provided with two photoelectric devices (PED1 and PED2) that are employed to determine the direction in which an object is moving on the conveyance system. Each photoelectric device acts like a normally open contact.

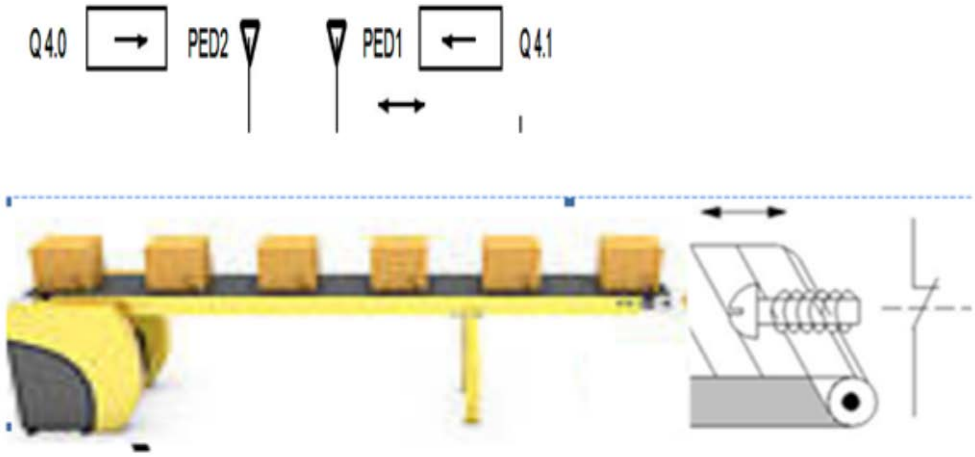


Figure 4

We write now the next sequence of our program, to design a direction display for the conveyance system shown in Figure 4, employing symbols, representing the principal components of the conveyor belt, including the photoelectric devices, for to detect the direction. Using this method, we had to make a symbol table to identify the symbols we chose, with absolute values (see Figure 5). There was created the symbol table, belonging to our program, with those defined symbols. The alternative method would be to write a program sequence, to design the direction display for the conveyance system, using absolute values, that represent the photoelectric devices for detecting direction. This was not used. Figure 6 reproduces a ladder logic program to control the direction display for the conveyance system.

SymbolTable.SEQ | TableOfVariables.var | Address Locations | Protokol | ToDo-List | Messages | PLC Online

...	Symbol	Address	Type	Symb.-Comment
3	S3	I 1.3	BOOL	Push Button Start Switch
4	S4	I 1.4	BOOL	Push Button Stop Switch
5	S5	I 1.5	BOOL	Sensor
6	MOTOR_ON	Q 4.0	BOOL	Motor
7	PED1	I 2.1	BOOL	Photo electric device 1
8	PED2	I 2.2	BOOL	Photo electric device 2
9	RIGHT	Q 4.1	BOOL	Display for movement to right
10	LEFT	Q 4.2	BOOL	Display for movement to left
11	PMB1	M 0.0	BOOL	Pulse memory bit 1
12	PMB2	M 0.1	BOOL	Pulse memory bit 2

Figure 5

Pulse memory bit 1 or 2 are used here in conjunction with CPU Registers, especially with Status Word. The Status Word Register contains bits that can be referenced in the addresses of bit logic instructions. The figure 6 explains the significance of bits 0 through 8.

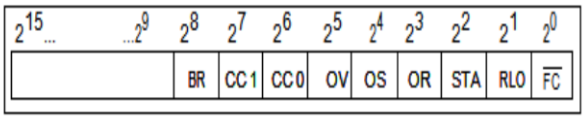


Figure 6

The changing of the bits in the Status Word leads to changes in the signal of ladder program flow.

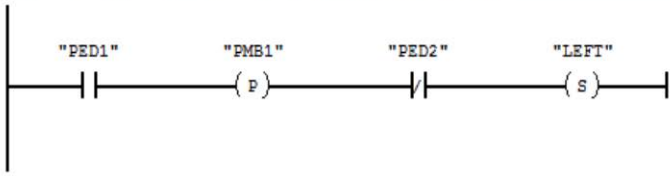
Value	Meaning
0	Sets the signal state to 0
1	Sets the signal state to 1
x	Changes the state
±	State remains unchanged

We shall use Bit 1 of the status word, which is called the Result of Logic Operation (RLO bit, in Figure 6). This bit retains the result of a flow of bit logic instructions or math comparisons. The signal state changes of the RLO bit provide information related to power flow. In the next program sequence (see figure 7), the first instruction in the network of ladder logic verifies the signal state of the PED1 contact and produces a result of 1 or 0. The instruction stores the result of this state check in the RLO bit. The second instruction in the rung of bit logic instructions (PMB1, state of the contents of a Boolean bit memory location- see figure 7) also verifies the signal state in this bit memory location, (or, maybe a contact) and produces a result. So, we use a bit logic instruction on a first check, storaged in the RLO and, relating it to the state of the contents of the Boolean bit memory location, we shall trigger the change of photoelectric display for left or right . The instruction combines this result with the value stored in the RLO bit of the status word according to the principles of Boolean logic. It means Positive RLO Edge Detection. The Edge memory bit PMB1

saves the old state of the RLO from the complete bit logic combination. If there is a signal change at the RLO from 0 to 1, the label Right or Left are changed . This logic operation is stored in the RLO bit of the status word, replacing the former value in the RLO bit. Each subsequent instruction in the rung performs a logic operation on two values: the result produced when the instruction checks the contact and the current RLO.

Network 3

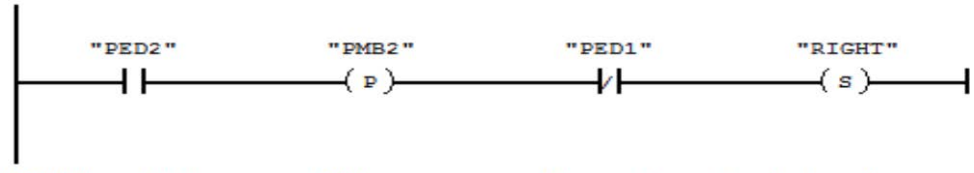
```
If there is a jump of signal from 0 to 1 ( rising) at input PED1 and, simultaneously,
the signal state at input PED2 is 0,
then the object on the belt is moving to the left.
```




```
"PED1"  I2.1    BOOL    Photo electric device 1
"PMB1"  M0.0    BOOL    Pulse memory bit 1
"PED2"  I2.2    BOOL    Photo electric device 2
"LEFT"  Q4.2    BOOL
```

Network 4

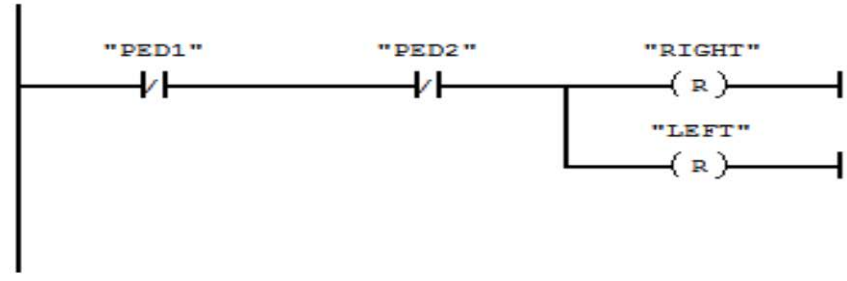
If there is a jump of signal from 0 to 1 (rising) at input PED2 and, simultaneously, the signal state at input PED1 is 0, then the object on the belt is moving to the right. If one of the photoelectric devices is blocked, this means that there is a object in the face of this.




```
"PED2"    I2.2    BOOL    Photo electric device 2
"PMB2"    M0.1    BOOL    Pulse memory bit 2
"PED1"    I2.1    BOOL    Photo electric device 1
"RIGHT"   Q4.1    BOOL    Display for movement to right
```

Figure 7

Network 5: If neither photoelectric device is broken, then there is no object in the face of these. The direction pointer shuts off.




```
"PED1"    I2.1    BOOL    Photo electric device 1
"PED2"    I2.2    BOOL    Photo electric device 2
"RIGHT"   Q4.1    BOOL    Display for movement to right
"LEFT"    Q4.2    BOOL
```

Figure 8

3. LADDER PROGRAM FOR TWO CONVEYANCE BELTS. The Figure 9 is a picture of a installation with two conveyances and a temporary storage area in between them, in which we used a counter and a comparator as ladder programing elements. The Conveyance belt 1 supplies objects to the storage area, . A photoelectric device, at the end of the conveyance belt 1, near the storage area, indicates the objects delivered to the storage area. The Conveyance belt 2 carries objects from the temporary storage area to a loading dock . A photoelectric device at the end of conveyance belt 2, near the storage area, indicates the objects which leave the storage area for to go to the loading dock. A display panel with five lights signals the fill level of the temporary storage area.

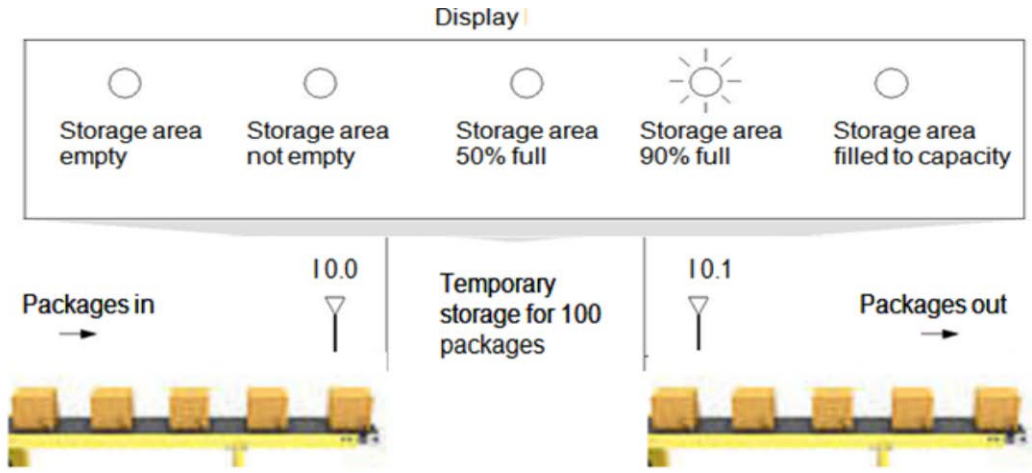
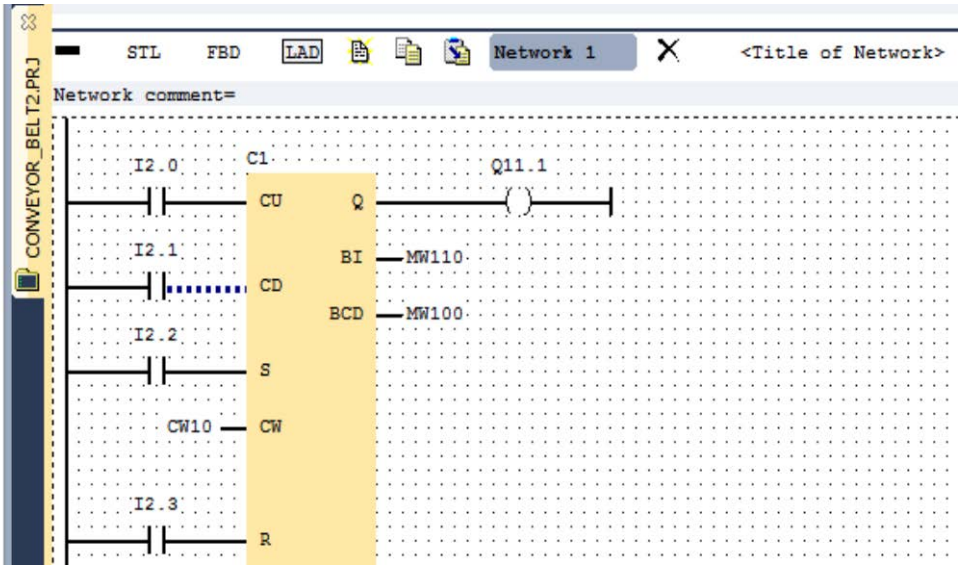


Figure 9

We creat an another program, clled Conveyor_belt2. Figure 10 reproduces the ladder program that controls the indicator lights on the display panel. In network 1, counter C1 counts up each rising from ª0º to ª1º at input CU and counts down at each decreasing from ª0º to ª1º at input CD. With a signal change from ª0º to º1º at input S, the counter value is set to the value CW. A rising from ª0º to ª1º at input R resets the counter value to ª0º. MW100 contains the current counter value of C1. Q11.1 indicates ªstorage area not emptyº. In the network 2 we used Q11.0 for to indicate ºstorage area emptyº.

In network 3, we control the indicator light for ªstorage area 50% fullº , which is lit if the current counter value is greater than or equal to 50. In the network 4, the indicator light for ªstorage area 90% fullº is lit, if the counter value is greater than or equal to 90.

In the network 5, the indicator light for ªstorage area fullº is lit, if the counter value is greater than or equal to 100 . The output Q17.3 is employed for to interlock conveyor belt 1.



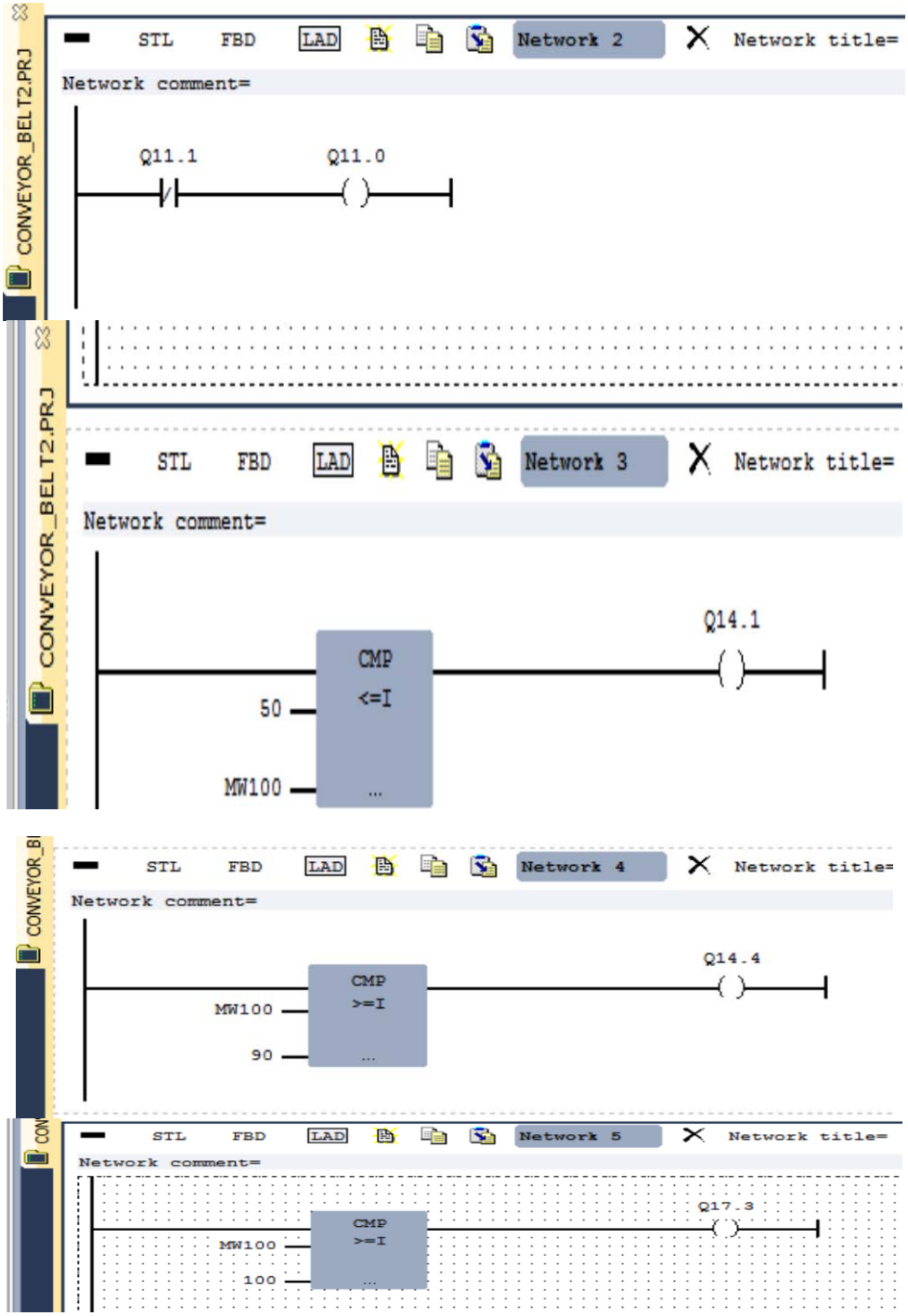


Figure 10

REFERENCES

[1].*www.prosoft.ru/cms/f/367435.pdf*
[2].*http://www.scribd.com/doc/209692479/Manual-Winplc7*
[3].*http://automatizace.hw.cz/software winplc7*
[4].*www.int-technics.pl/media/2011/07/HB140E_CPU_315-4NE12_09-45.pdf*
[5].*www.factoryintel.com.au/index.php/vipa-plcs*