PLCs AND DCS IN PROCESS CONTROL AUTOMATION

Course Code: **20EI701**

Sharath H K, M.Tech.

Assistant Professor

Dept. of Mechanical Engineering,

MCE, Hassan.

PLC Instructions & Introduction to SCADA & DCS

Module – 4

Contents

Module – 4

PLC Instructions continued: Comparison and Data handling Instructions: comparison, Data handling and Logical Instructions. Sequencer Instructions: The Sequencer Instructions, Programming the output. Sequencer Instructions. Construction of Ladder diagram for analytical problems.

Introduction to SCADA and DCS: Supervisory Control and data Acquisition System: Channel Scanning, Data Processing, distributed SCADA Structure; Star and daisy Chain configuration. Distributed Control Systems: Distributed dedicated, Centralized and decentralized Computer Control Concept. Functional Requirements of DCS, System architecture, Functional Levels of DCS, Sub Systems; Presentation and Monitoring Devices, Communication links in DCS.

COMPARISON INSTRUCTIONS

1. The Equal Instruction (EQU)
2. The Not Equal Instruction (NEQ)
3. The Less Than Instruction (LES)
4. The Less Than or Equal Instruction (LEQ)
5. The Greater Than Instruction (GRT)
6. The Greater Than or Equal Instruction (GEQ)

The Equal Instruction (EQU)

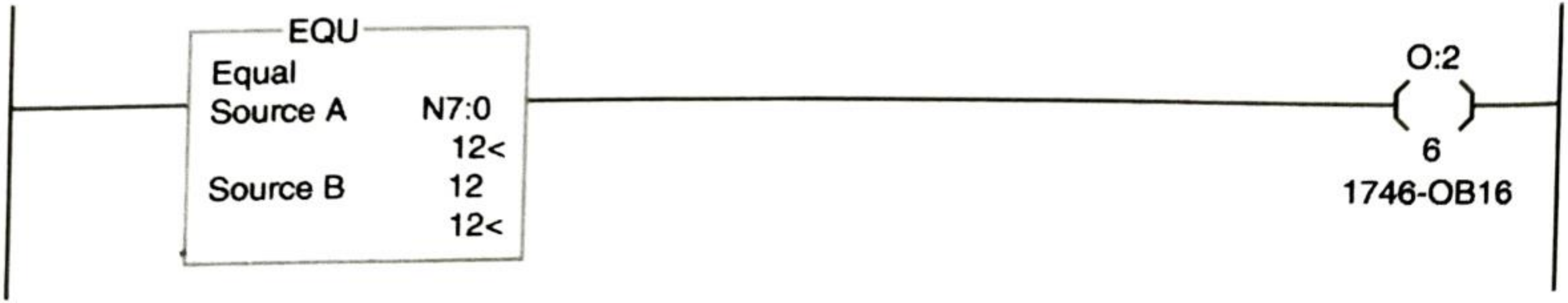


Figure 17-1 An equal instruction controlling OTE instruction address O:2/6.

The Not Equal Instruction (NEQ)

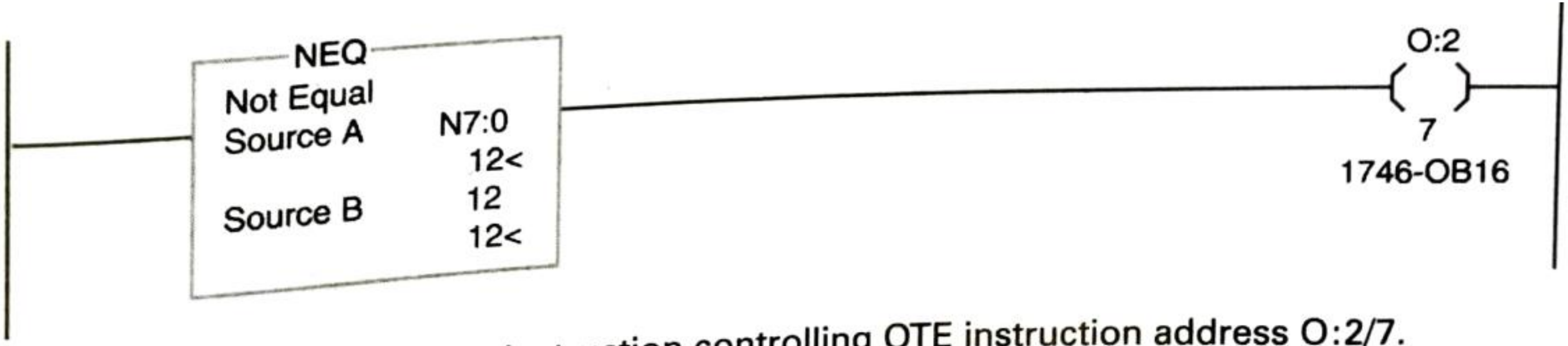


Figure 17-2 Not equal input instruction controlling OTE instruction address O:2/7.

The Less Than Instruction (LES)

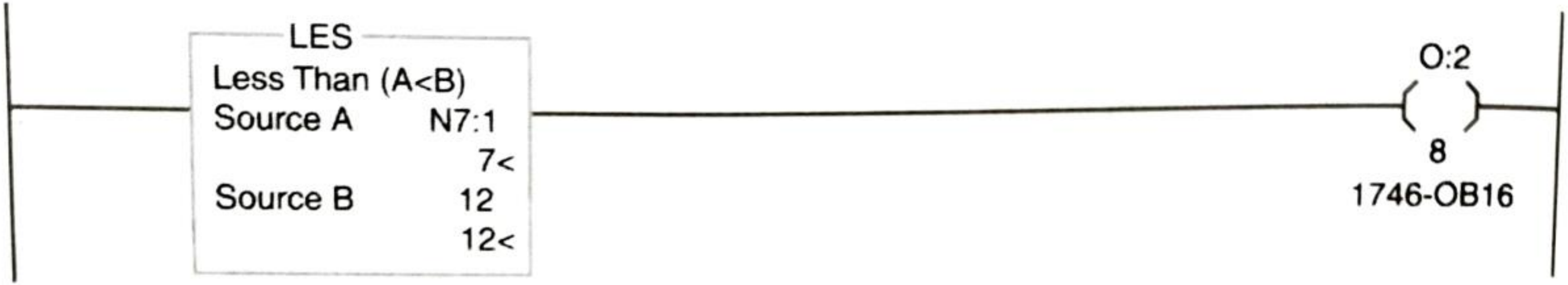


Figure 17-3 The less than instruction tests to see if the value in source A is less than the value stored in or addressed as source B.

The Less Than or Equal Instruction (LEQ)

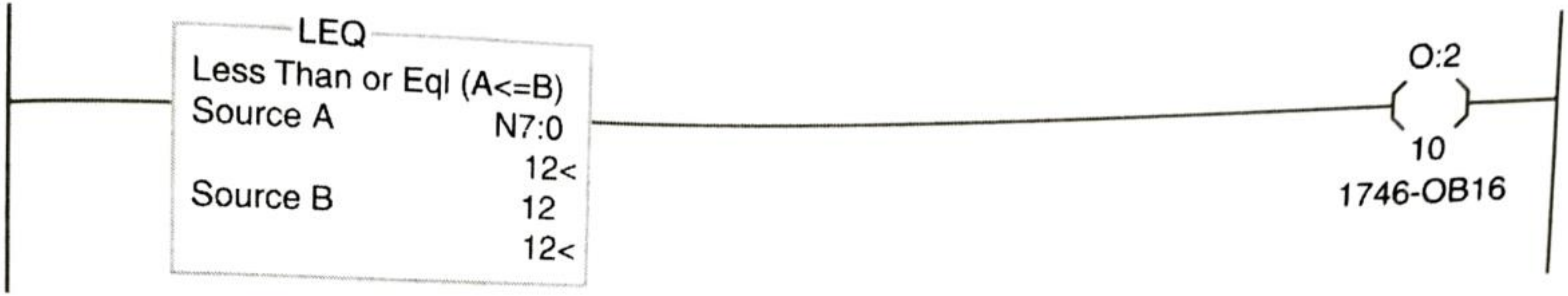
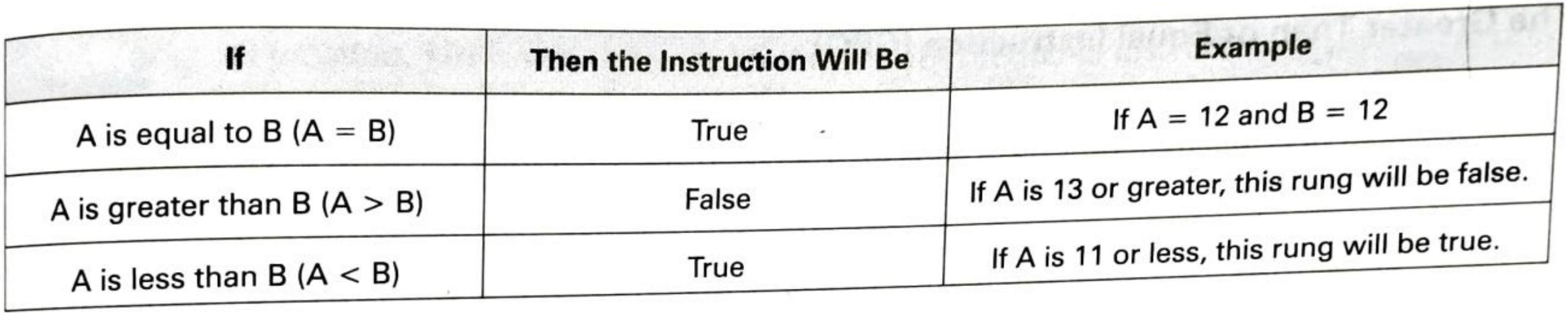


Figure 17-4 Less than or equal instruction tests to see if A ≤ B.



If	Then the Instruction Will Be	Example
A is equal to B (A = B)	True	If A = 12 and B = 12
A is greater than B (A > B)	False	If A is 13 or greater, this rung will be false.
A is less than B (A < B)	True	If A is 11 or less, this rung will be true.

Figure 17-5 Table showing A and B relationship for A ≤ B logic.

The Greater Than Instruction (GRT)

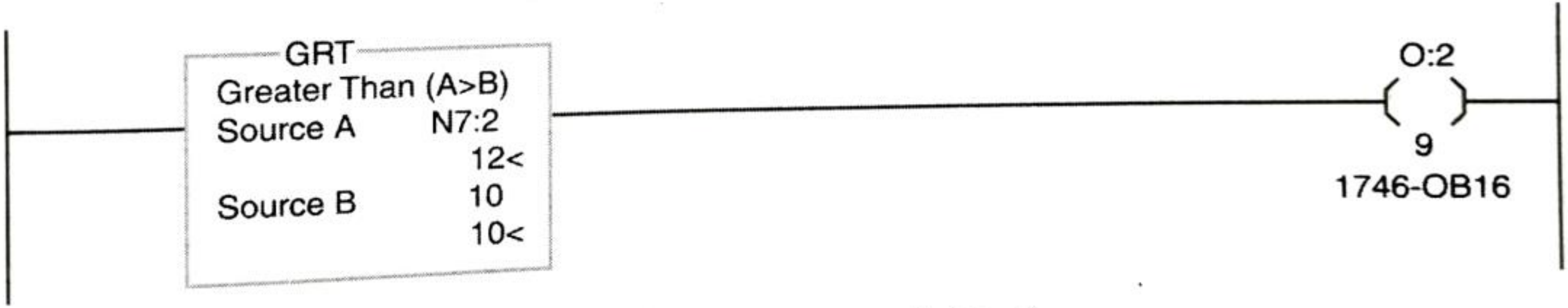


Figure 17-6 The greater than instruction tests to see if A > B.

The Greater Than or Equal Instruction (GEQ)

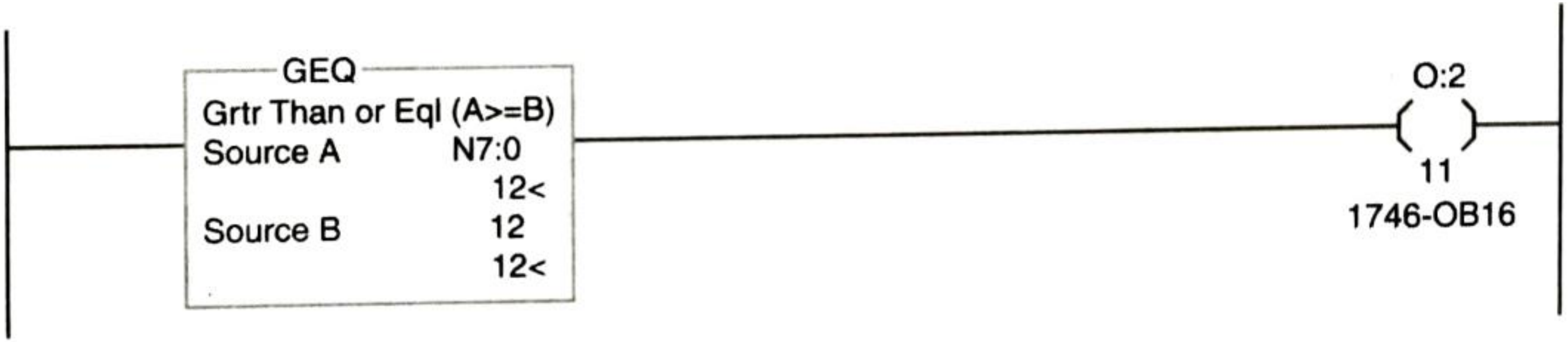
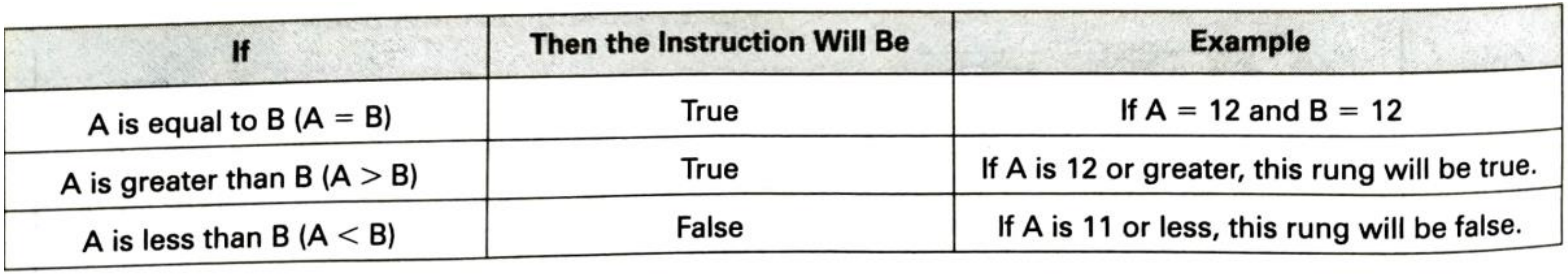


Figure 17-8 Greater than or equal instruction tests to see if A ⩾ B.



If	Then the Instruction Will Be	Example
A is equal to B (A = B)	True	If A = 12 and B = 12
A is greater than B (A > B)	True	If A is 12 or greater, this rung will be true.
A is less than B (A < B)	False	If A is 11 or less, this rung will be false.

Figure 17-9 Table showing A and B relationship for A ⩾ B logic.

DATA HANDLING INSTRUCTIONS

1. The Move Instruction
2. The Masked Move Instruction (MVM)
3. Converting from BCD to an Integer (FRD)
4. Converting an Integer to BCD (TOD)
5. The Copy Instruction

The Move Instruction

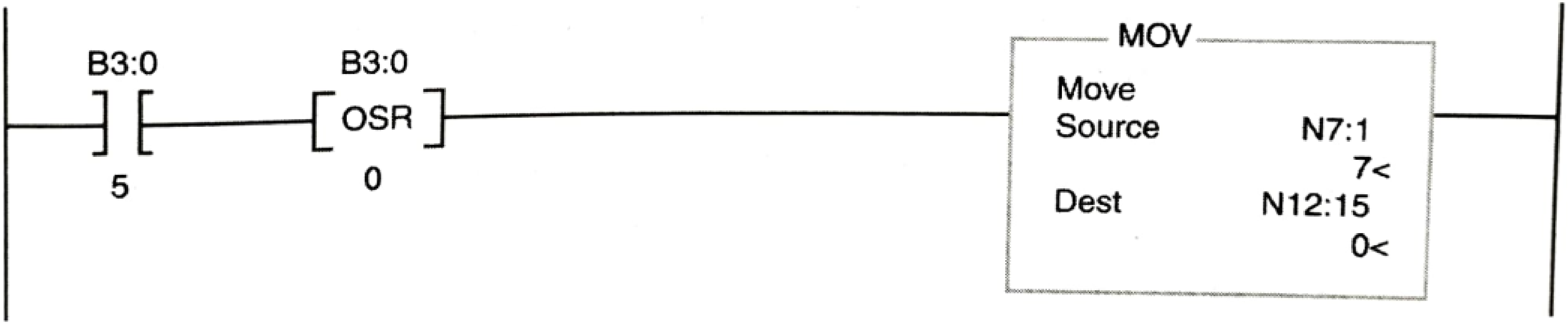


Figure 17-10 A move instruction, when true, will move a copy of the information stored in the specified source to the specified destination.

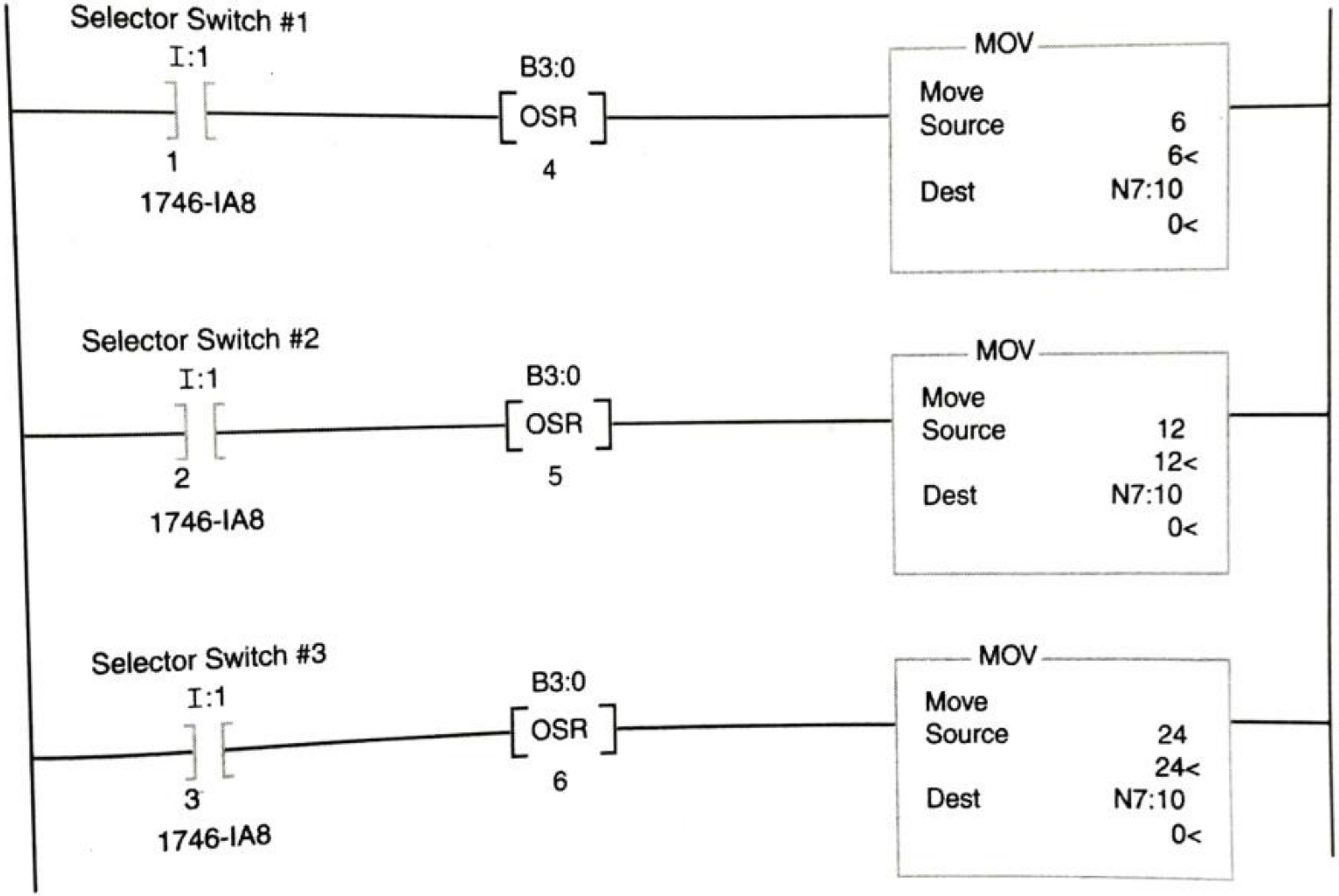


Figure 17-11 Selector switches used to select length specifications using a move instruction.

The Masked Move Instruction (MVM)

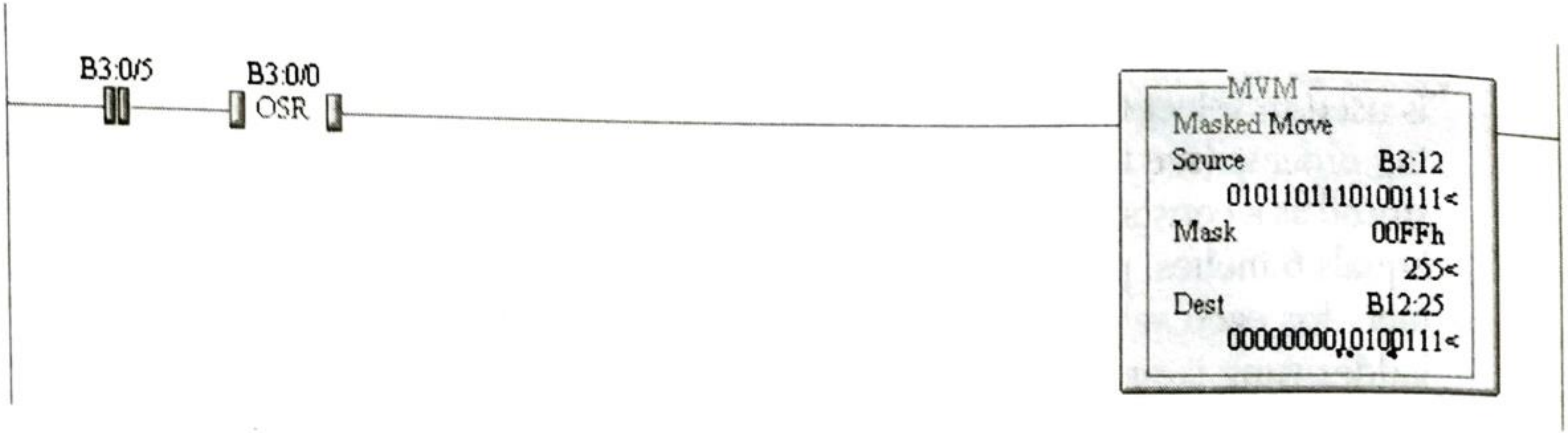
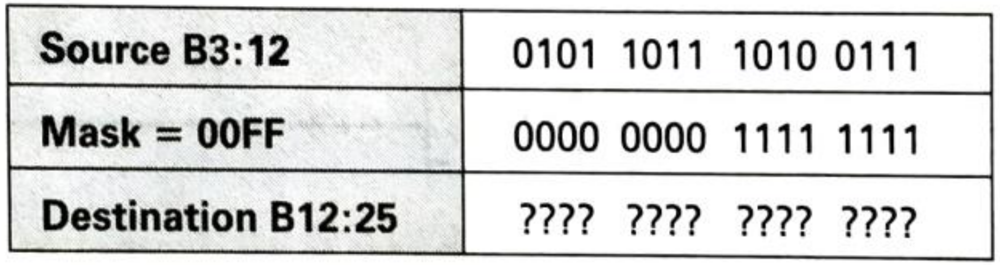
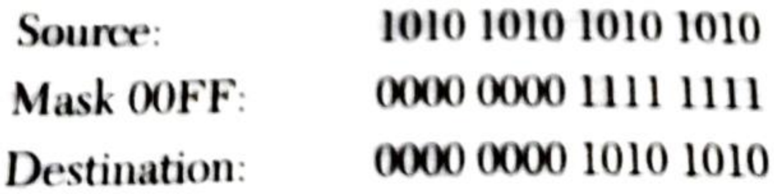


Figure 17-12 RSLogix 500 software masked move instruction.



Source B3:12	0101 1011 1010 0111
Mask = 00FF	0000 0000 1111 1111
Destination B12:25	???? ???? ???? ????



Source:	1010 1010 1010 1010
Mask 00FF:	0000 0000 1111 1111
Destination:	0000 0000 1010 1010

Figure 17-13 Source data moved through a mask to the destination.

Converting from BCD to an Integer (FRD)

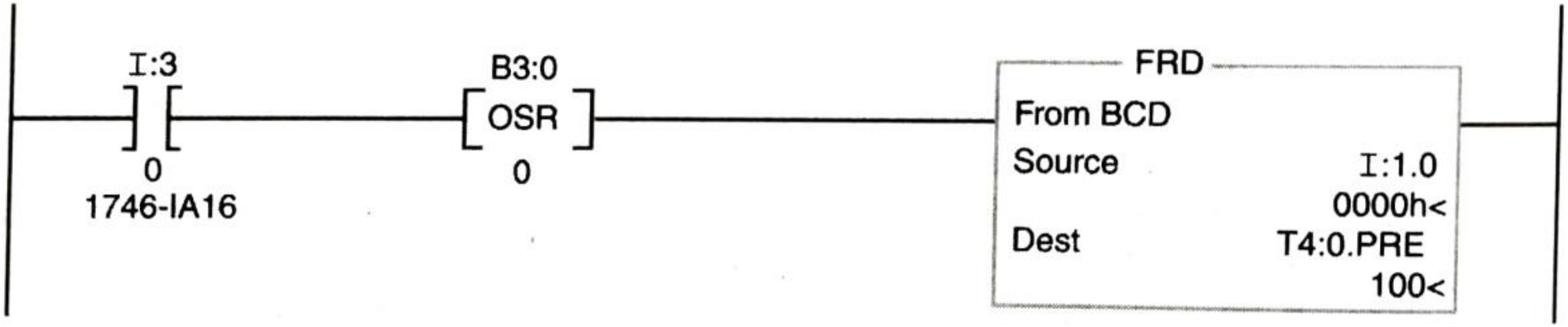


Figure 17-15 FRD instruction converting BCD input data for use in a timer preset value.

Converting an Integer to BCD (TOD)

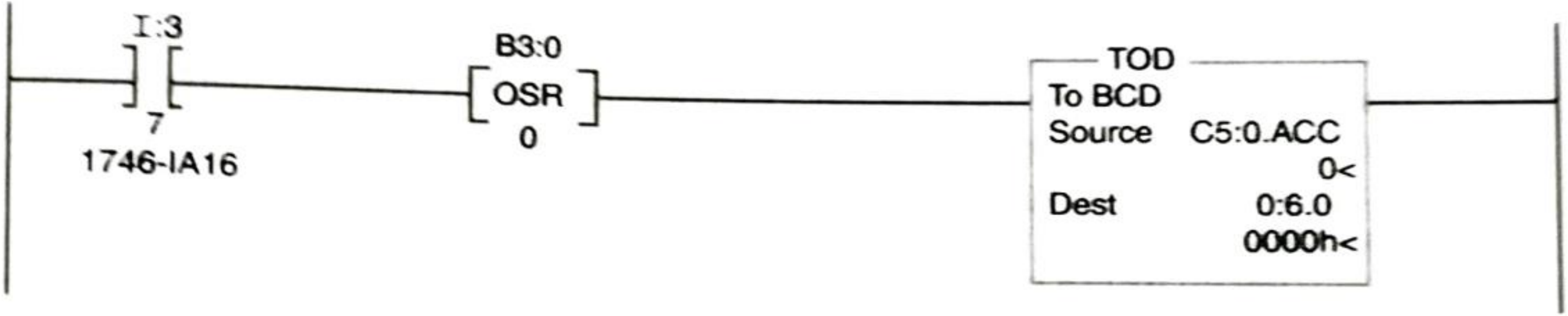


Figure 17-16 TOD instruction converting an integer value, counter file five, counter zero's accumulated value to BCD.

The Copy Instruction

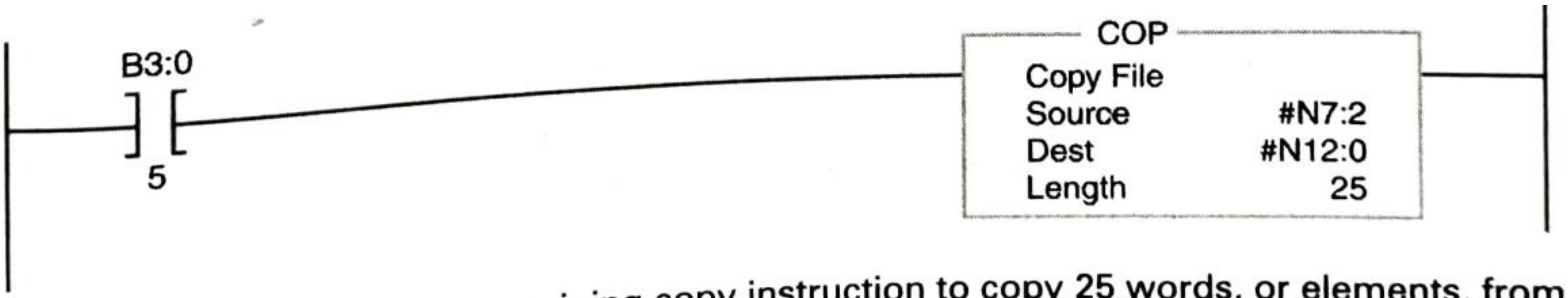
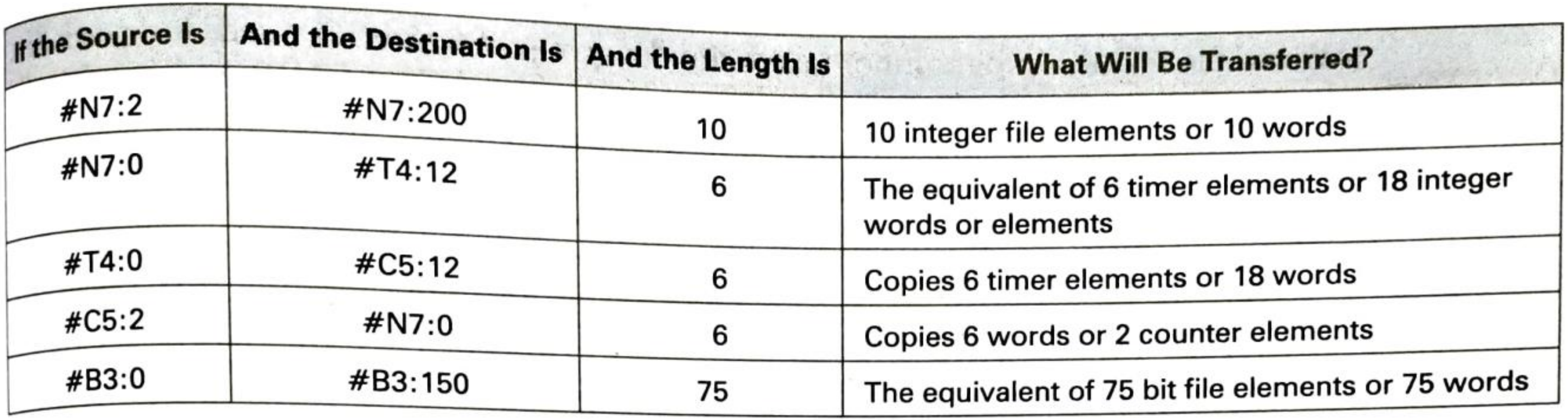
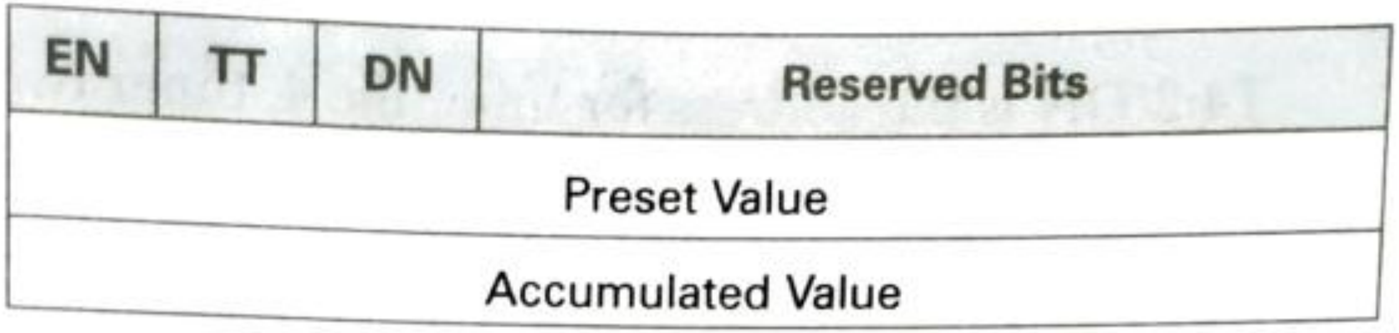


Figure 17-18 Ladder rung containing copy instruction to copy 25 words, or elements, from one integer file to another.



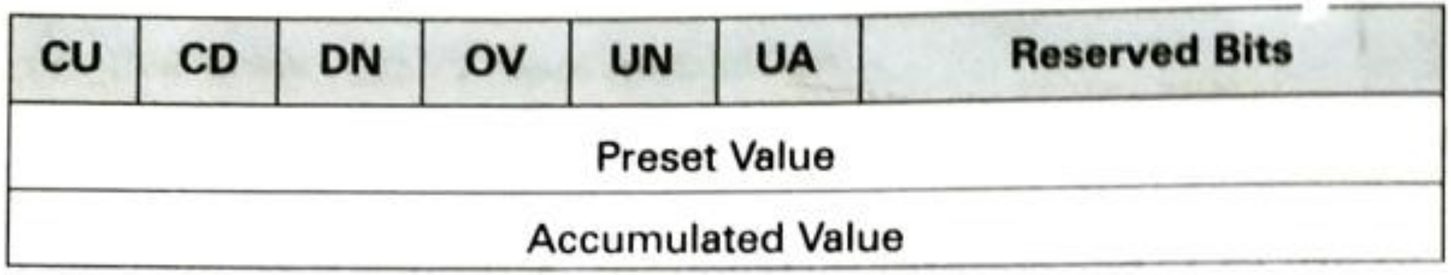
If the Source Is	And the Destination Is	And the Length Is	What Will Be Transferred?
#N7:2	#N7:200	10	10 integer file elements or 10 words
#N7:0	#T4:12	6	The equivalent of 6 timer elements or 18 integer words or elements
#T4:0	#C5:12	6	Copies 6 timer elements or 18 words
#C5:2	#N7:0	6	Copies 6 words or 2 counter elements
#B3:0	#B3:150	75	The equivalent of 75 bit file elements or 75 words

Figure 17-17 Elements transferred for mixed file copy examples.

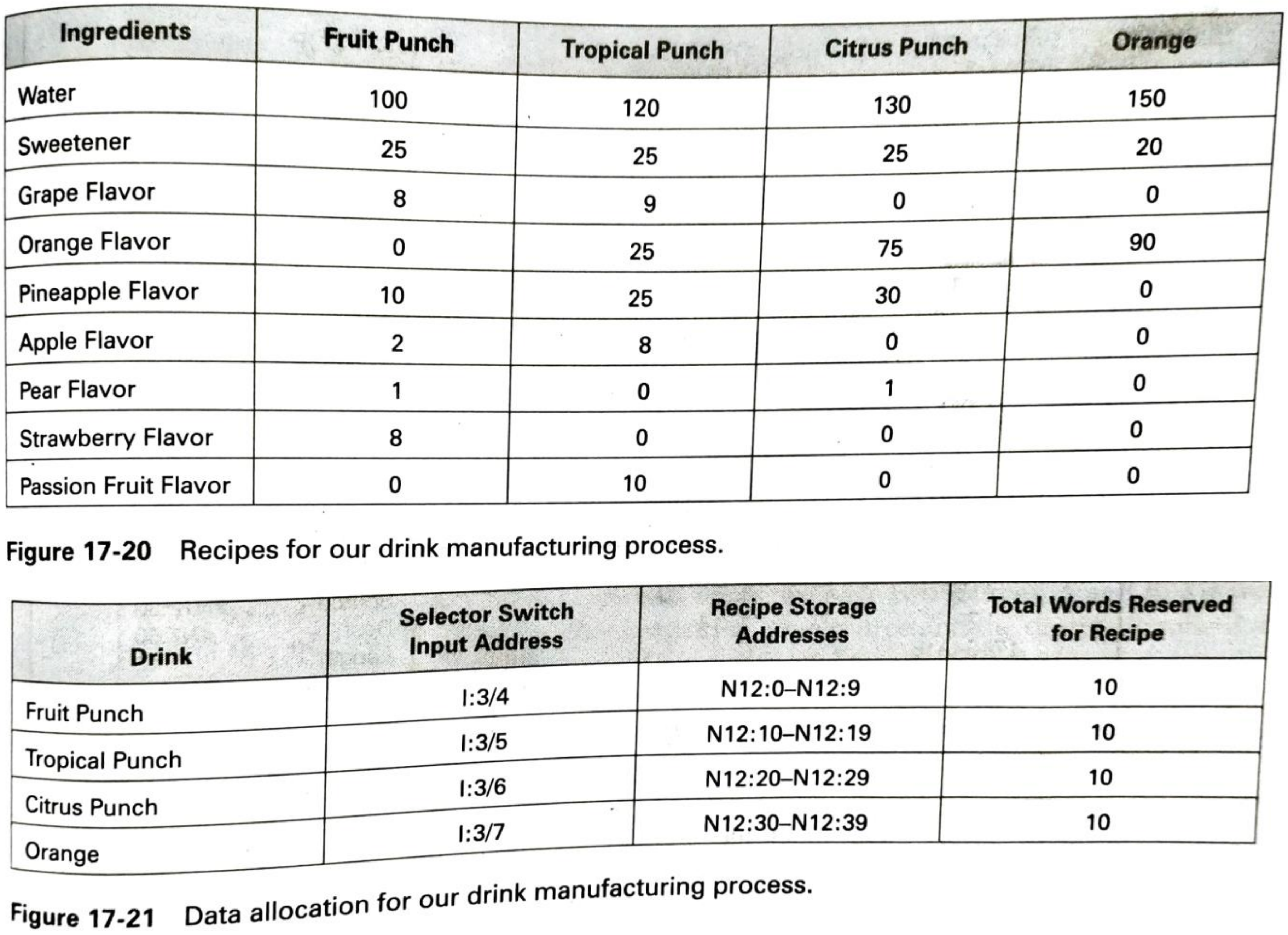


EN	TT	DN	Reserved Bits
Preset Value			
Accumulated Value			

One timer element is made of three 16-bit words.



CU	CD	DN	OV	UN	UA	Reserved Bits
Preset Value						
Accumulated Value						



Ingredients	Fruit Punch	Tropical Punch	Citrus Punch	Orange
Water	100	120	130	150
Sweetener	25	25	25	20
Grape Flavor	8	9	0	0
Orange Flavor	0	25	75	90
Pineapple Flavor	10	25	30	0
Apple Flavor	2	8	0	0
Pear Flavor	1	0	1	0
Strawberry Flavor	8	0	0	0
Passion Fruit Flavor	0	10	0	0

Figure 17-20 Recipes for our drink manufacturing process.

Drink	Selector Switch Input Address	Recipe Storage Addresses	Total Words Reserved for Recipe
Fruit Punch	I:3/4	N12:0–N12:9	10
Tropical Punch	I:3/5	N12:10–N12:19	10
Citrus Punch	I:3/6	N12:20–N12:29	10
Orange	I:3/7	N12:30–N12:39	10

Figure 17-21 Data allocation for our drink manufacturing process.

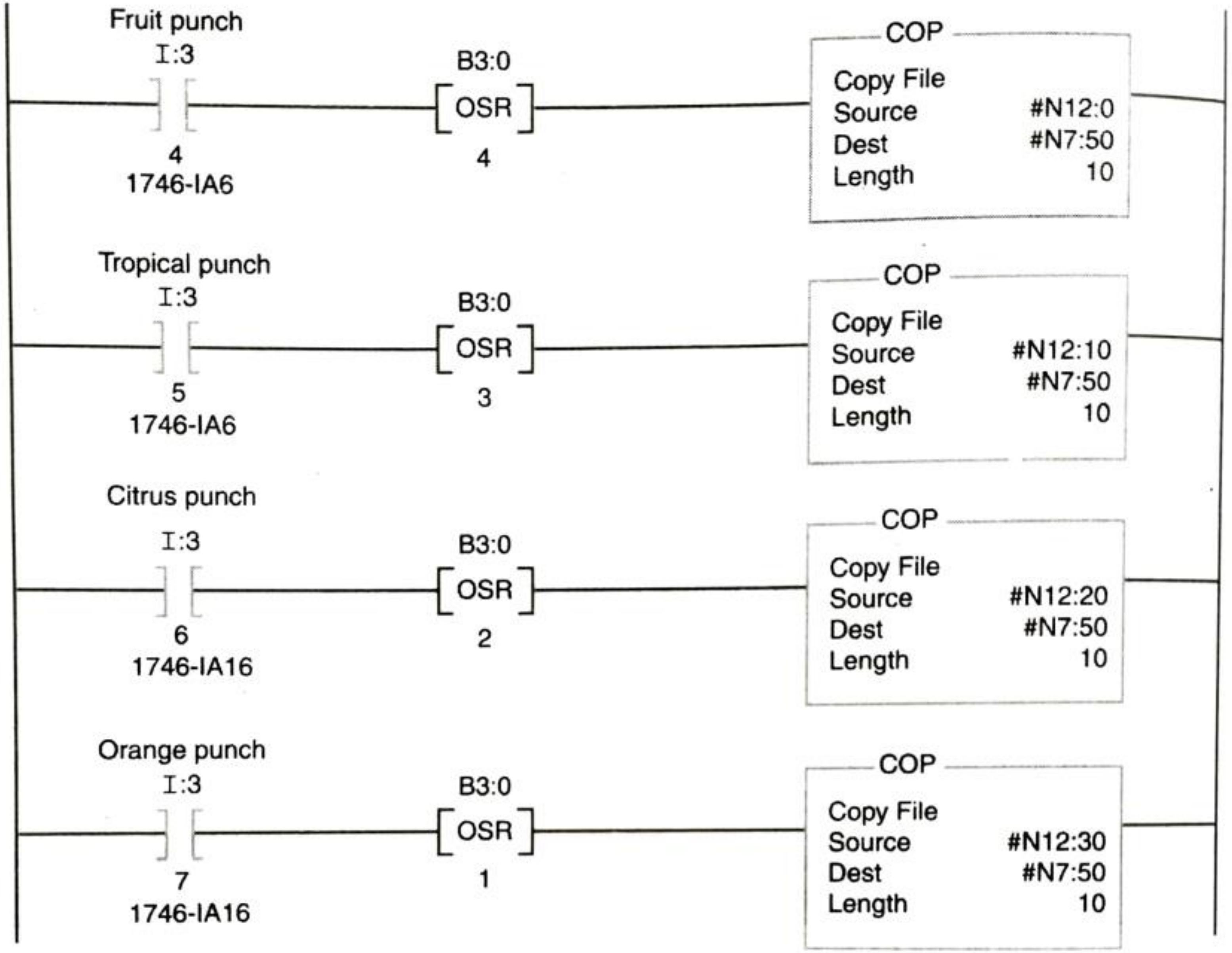
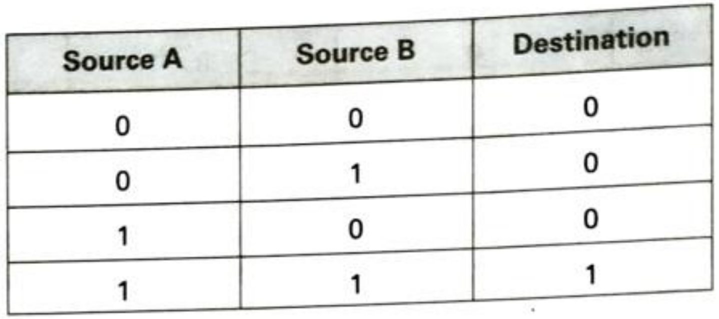


Figure 17-22 Drink flavor selection portion of our program.

LOGICAL INSTRUCTIONS

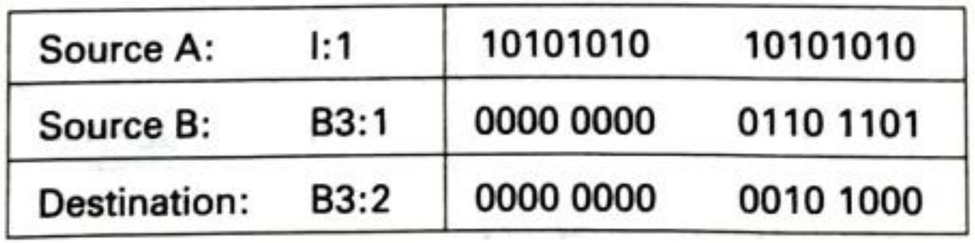
1. The AND Instruction
2. The OR Instruction
3. The Exclusive-OR Instruction
4. The NOT Instruction

The AND Instruction



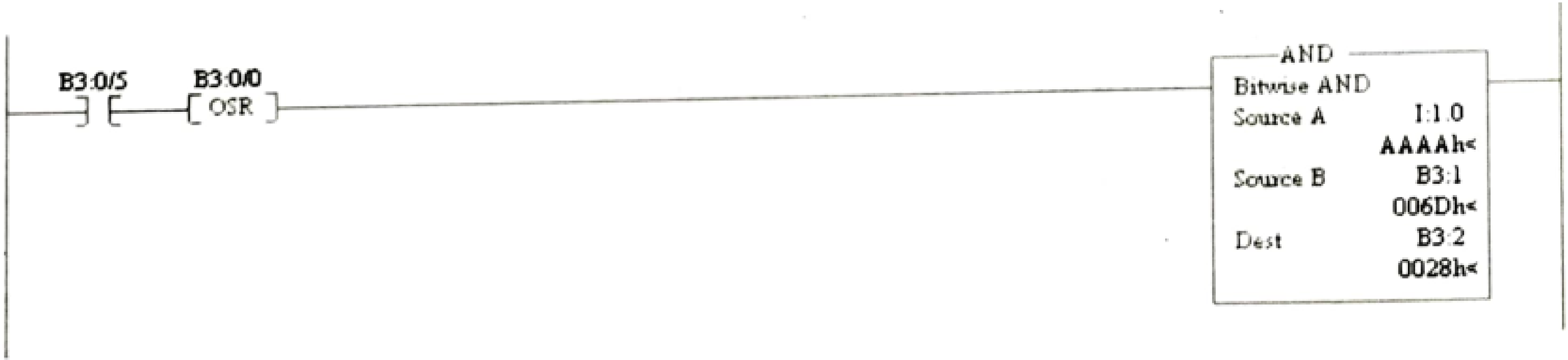
Source A	Source B	Destination
0	0	0
0	1	0
1	0	0
1	1	1

AND truth table.



Source A:	I:1	10101010	10101010
Source B:	B3:1	0000 0000	0110 1101
Destination:	B3:2	0000 0000	0010 1000

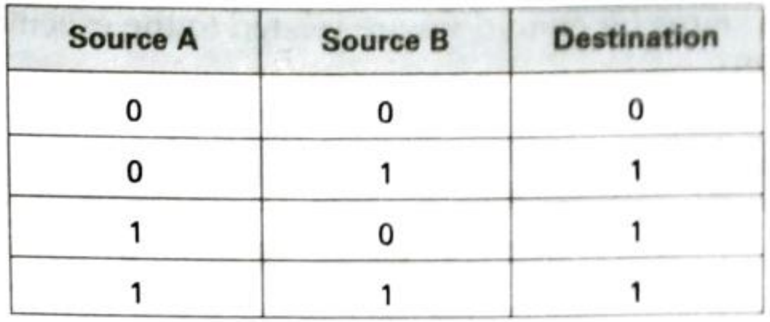
ANDing source A and source B and the resulting 16-bit word.



Bitwise AND
Source A I:1.0
AAAAh<
Source B B3:1
006Dh<
Dest B3:2
0028h<

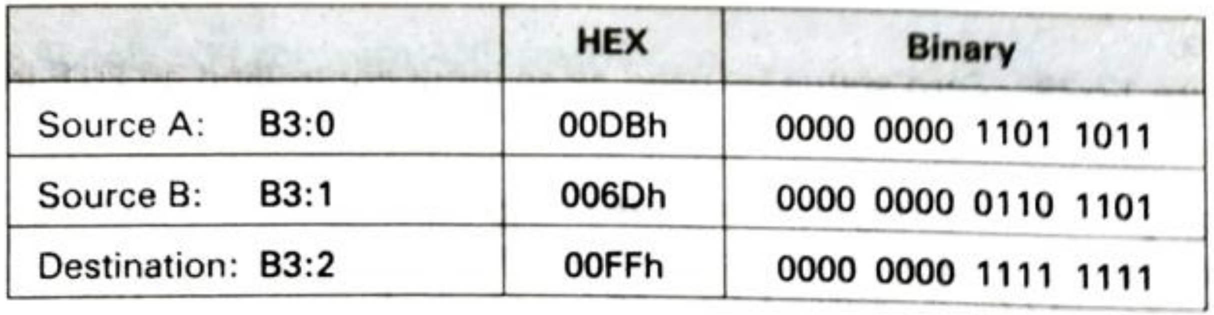
Figure 17-26 RSLogix 500 software AND instruction.

The OR Instruction



Source A	Source B	Destination
0	0	0
0	1	1
1	0	1
1	1	1

OR truth table.



		HEX	Binary
Source A:	B3:0	00DBh	0000 0000 1101 1011
Source B:	B3:1	006Dh	0000 0000 0110 1101
Destination:	B3:2	00FFh	0000 0000 1111 1111

ORing source A and source B and the resulting 16-bit word.

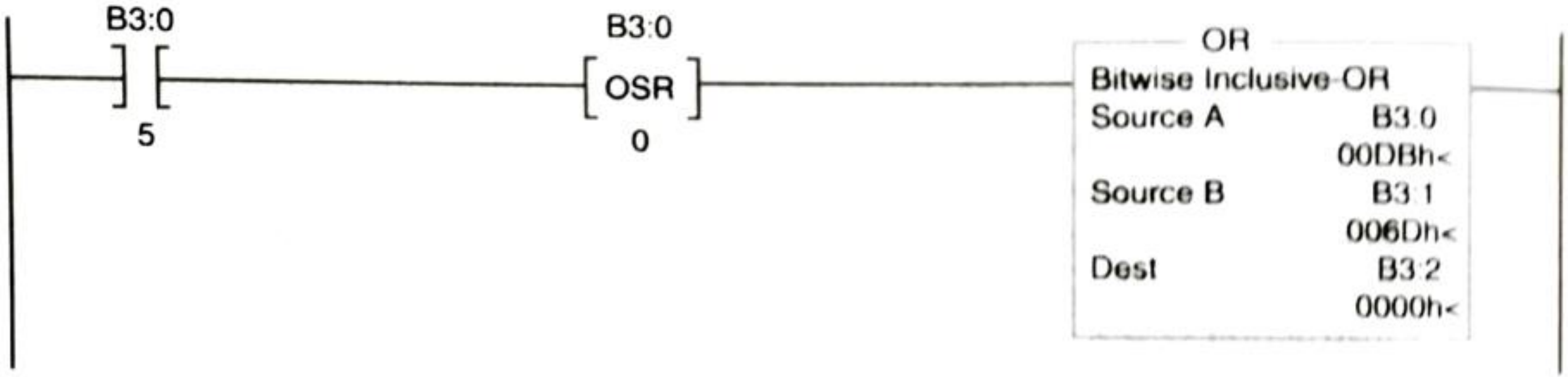
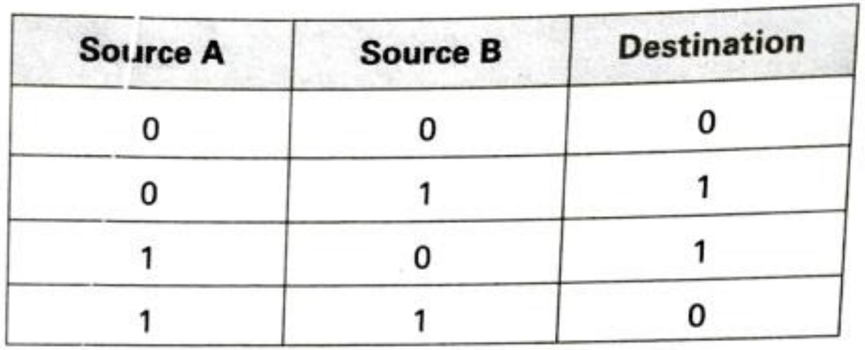


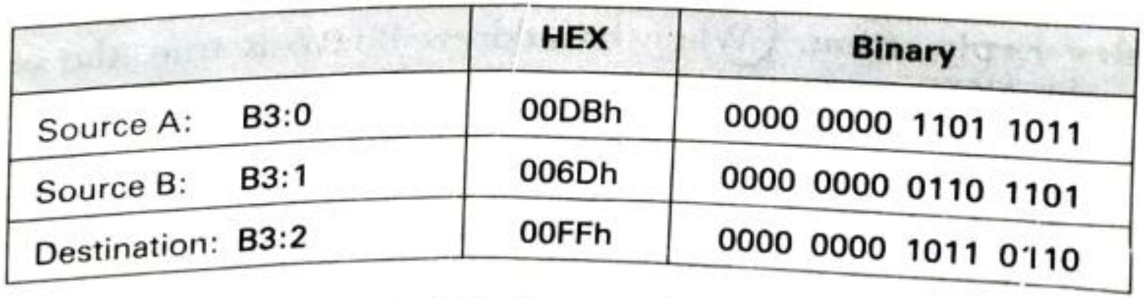
Figure 17-31 OR instruction on a ladder rung.

The Exclusive-OR Instruction



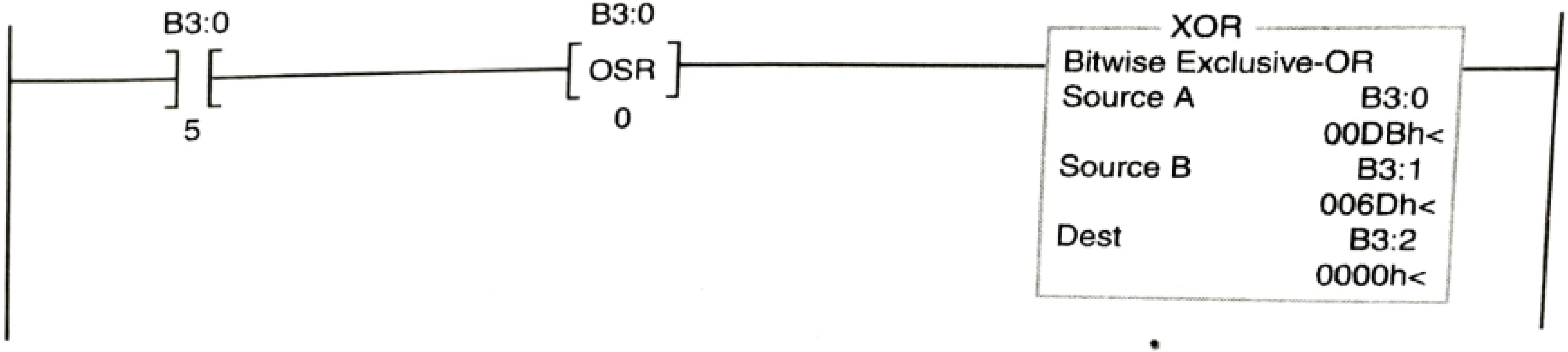
Source A	Source B	Destination
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-OR truth table.



	HEX	Binary
Source A: B3:0	00DBh	0000 0000 1101 1011
Source B: B3:1	006Dh	0000 0000 0110 1101
Destination: B3:2	00FFh	0000 0000 1011 0110

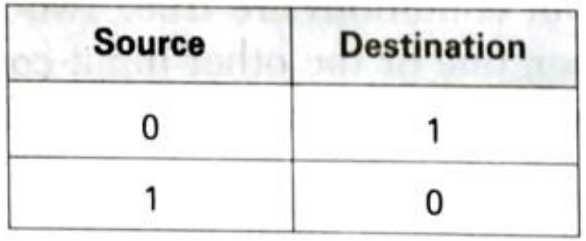
Exclusive ORing source A and source B and the resulting 16-bit word.



Bitwise Exclusive-OR
Source A B3:0
00DBh<
Source B B3:1
006Dh<
Dest B3:2
0000h<

Figure 17-34 Exclusive-OR instruction on a ladder rung.

The NOT Instruction



Source	Destination
0	1
1	0

Truth table for NOT logic.

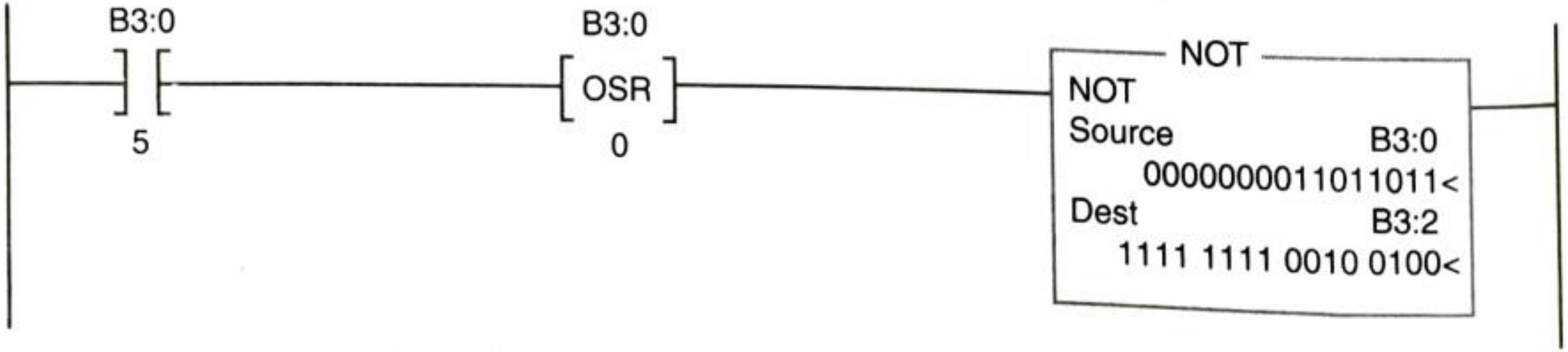


Figure 17-37 NOT logic as an output instruction on a PLC ladder rung. After execution, destination will contain 1111 1111 0010 0100.

The Limit Test Instruction

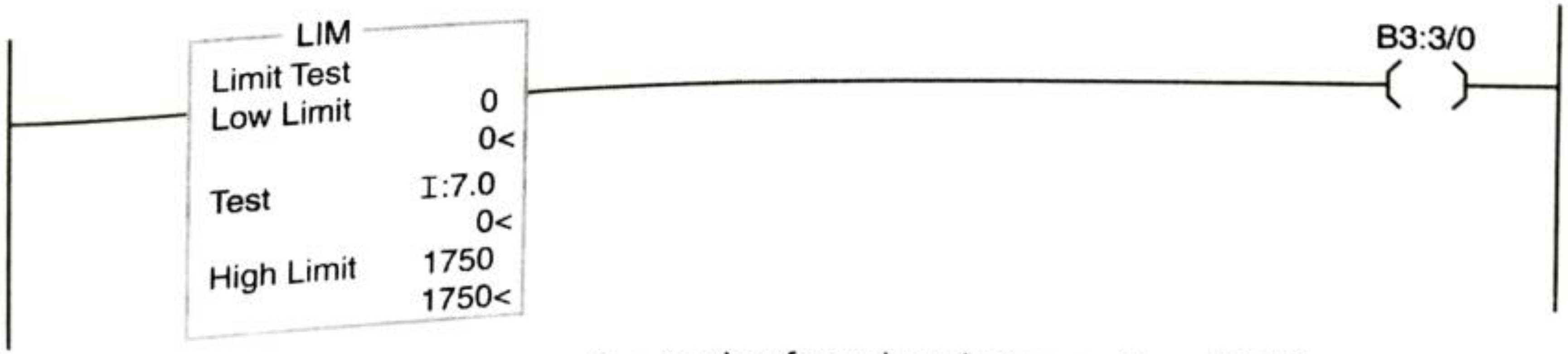


Figure 17-40 The limit test instruction testing for values between 0 and 1750.

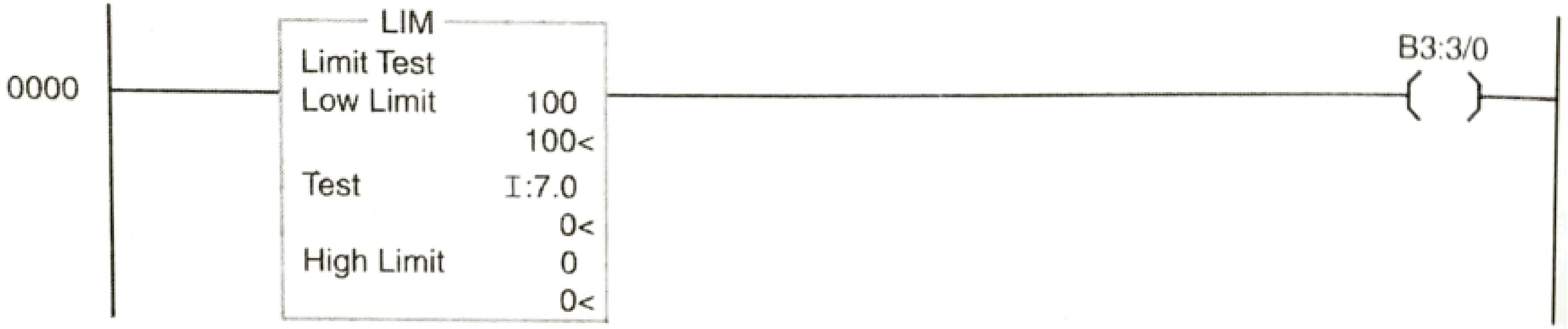


Figure 17-41 The limit test instruction testing for values outside the range of 0 to 100.

The Scale with Parameters Instruction

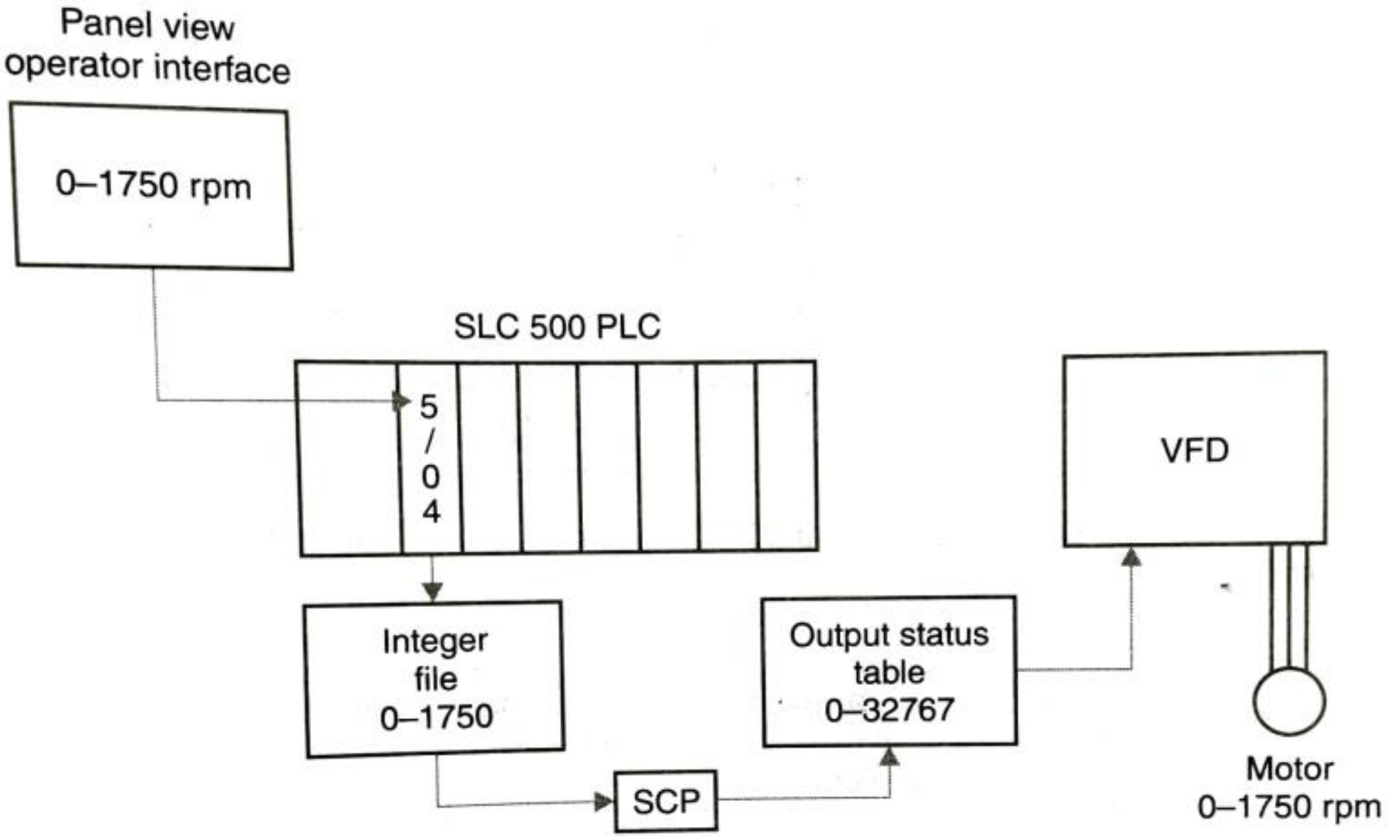
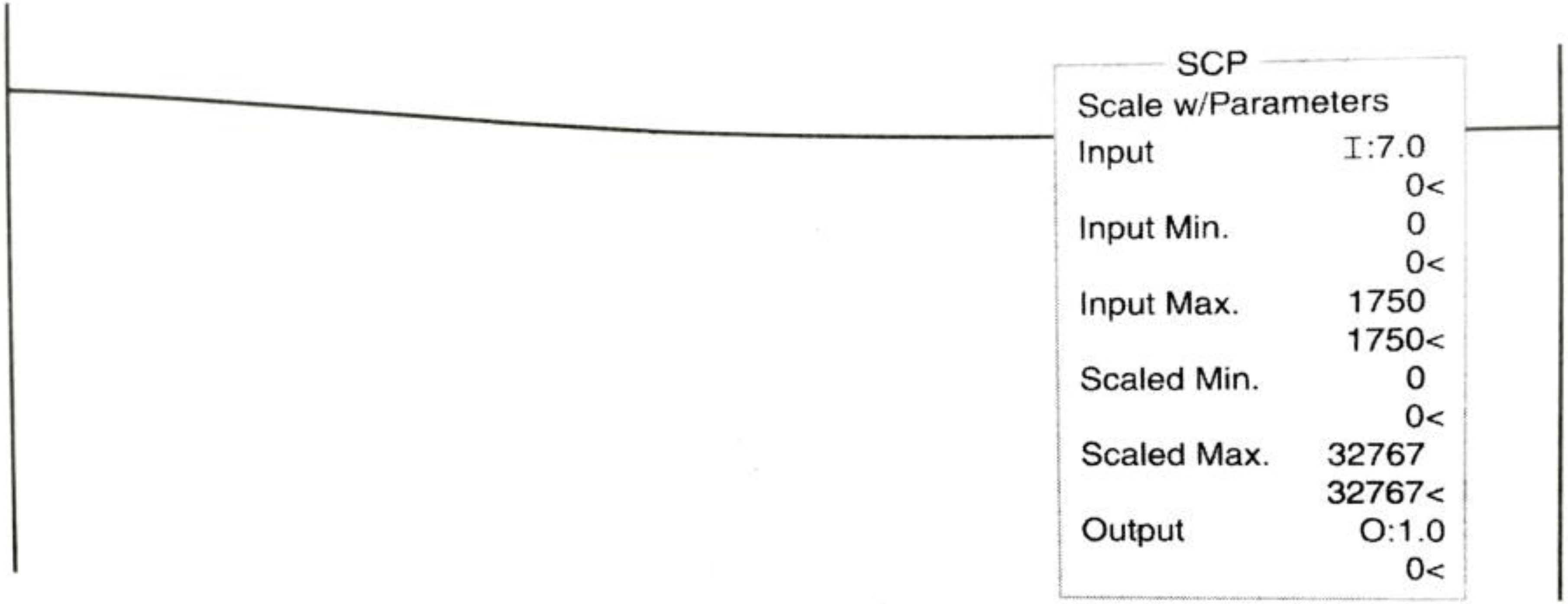


Figure 17-42 Data flow from a panel view operator interface terminal into the PLC and out to a VFD.

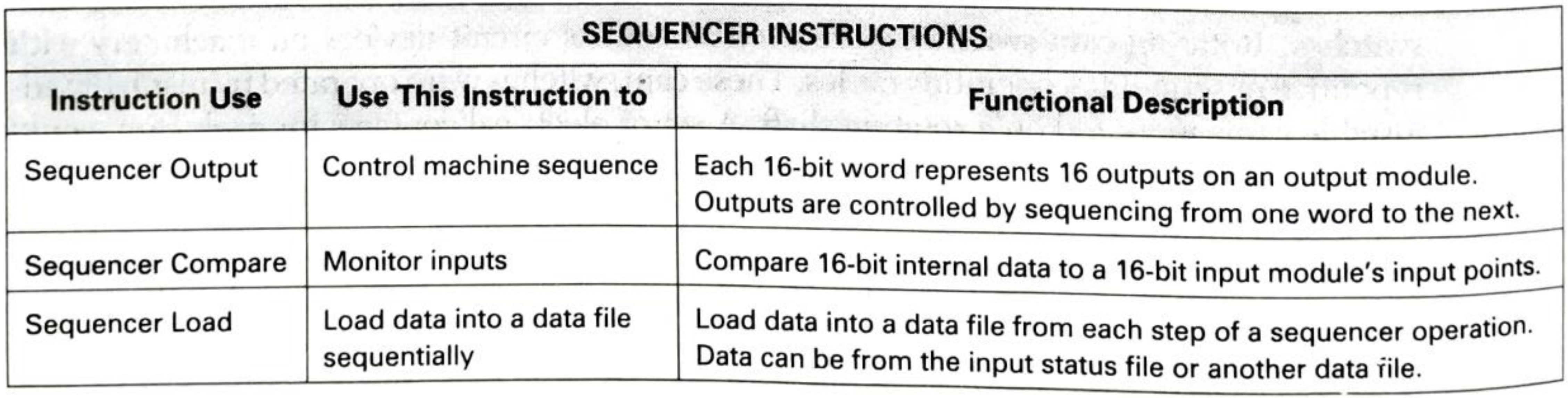


SCP
Scale w/Parameters

```
Input            I:7.0
                    0<
Input Min.          0
                    0<
Input Max.       1750
                 1750<
Scaled Min.         0
                    0<
Scaled Max.     32767
                32767<
Output          O:1.0
                    0<
```

Figure 17-45 The scale with parameters instruction scaling the input value of 0 to 1750 to 0 to 32767.

THE SEQUENCER



SEQUENCER INSTRUCTIONS		
Instruction Use	**Use This Instruction to**	**Functional Description**
Sequencer Output	Control machine sequence	Each 16-bit word represents 16 outputs on an output module. Outputs are controlled by sequencing from one word to the next.
Sequencer Compare	Monitor inputs	Compare 16-bit internal data to a 16-bit input module's input points.
Sequencer Load	Load data into a data file sequentially	Load data into a data file from each step of a sequencer operation. Data can be from the input status file or another data file.

Figure 18-1 Sequencer instructions available for the Allen-Bradley SLC 500 family of PLCs.

SEQUENCER INSTRUCTIONS

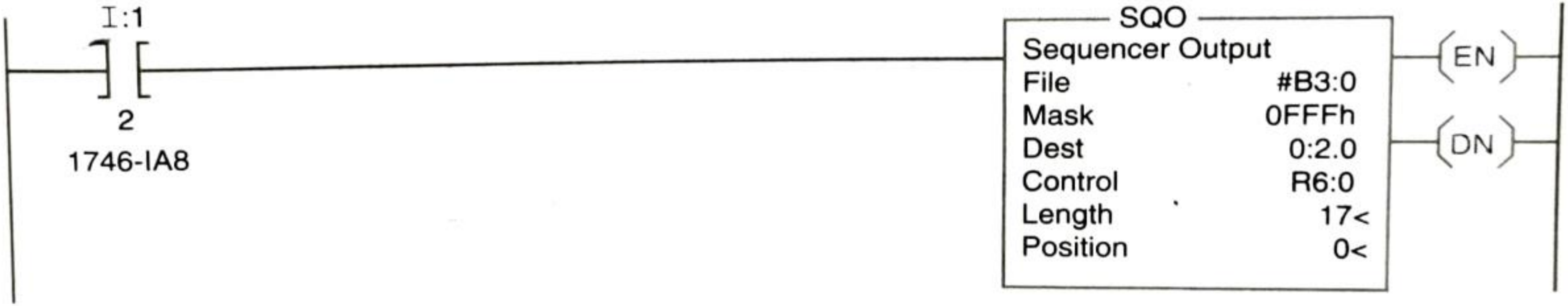
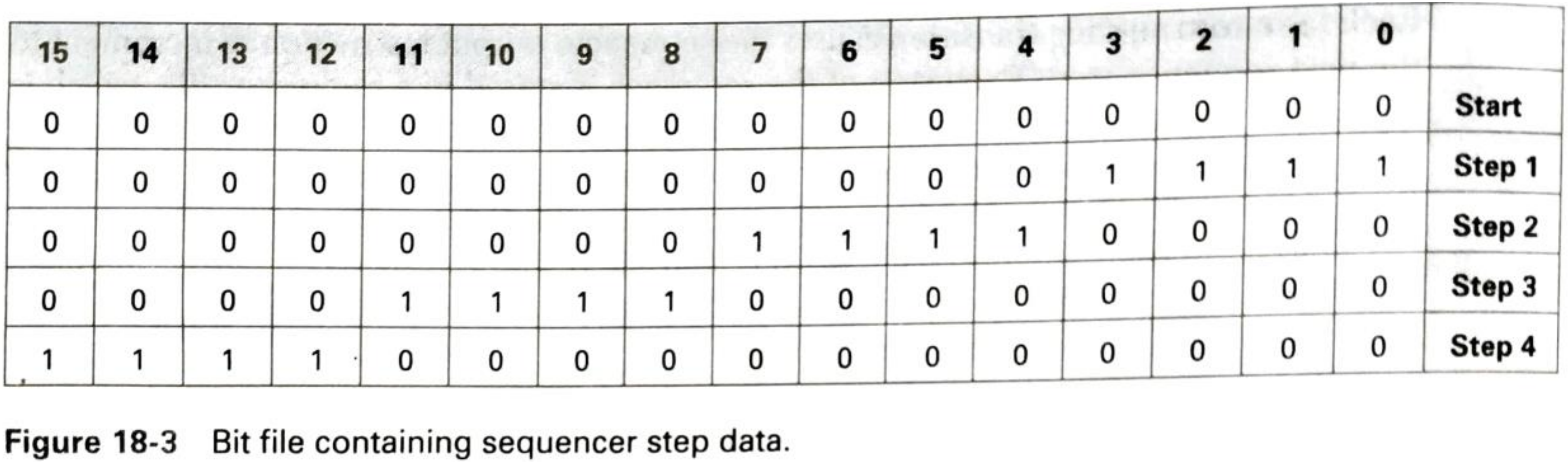
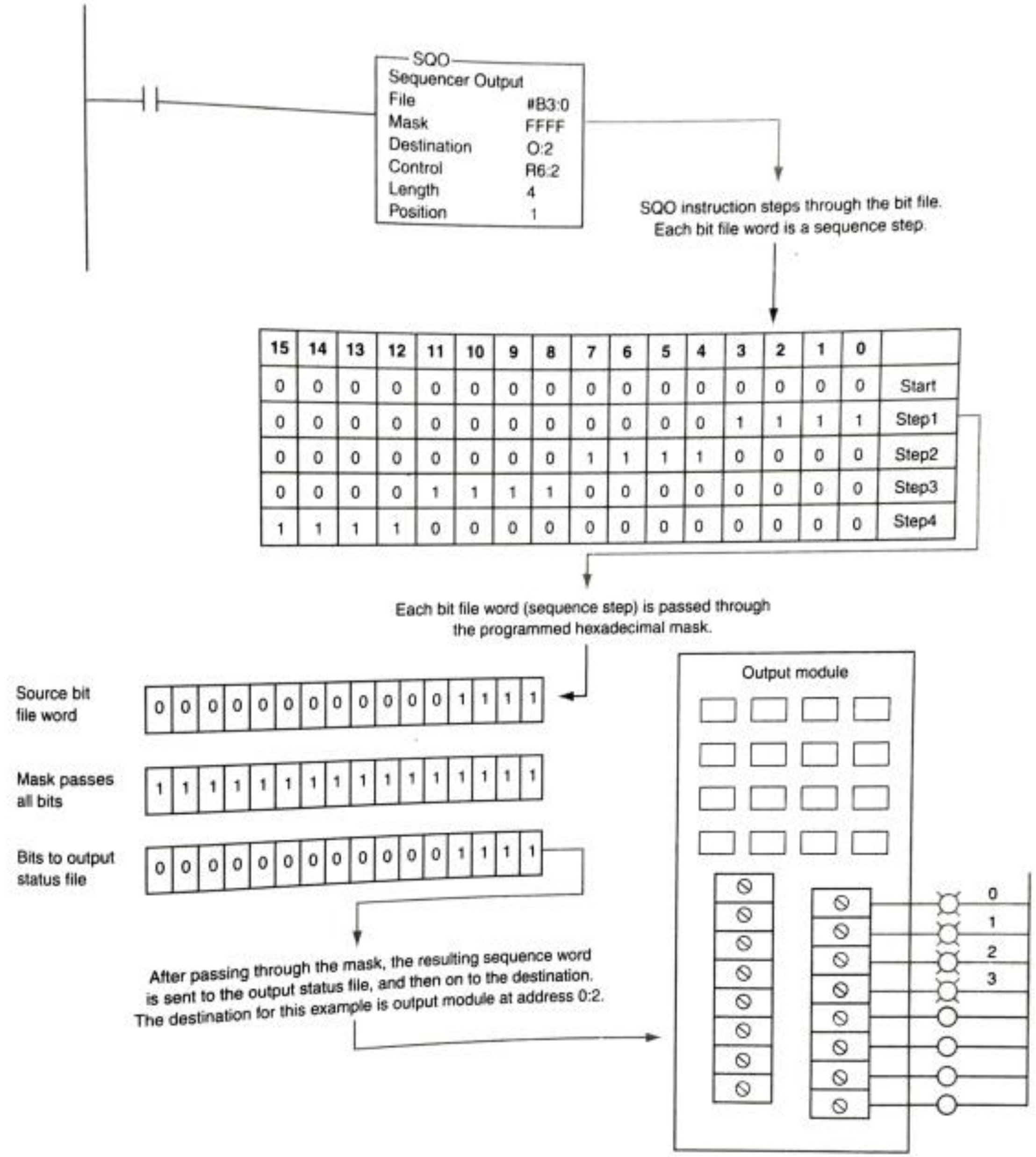


Figure 18-2 SLC 500 PLC family sequencer output instruction on a ladder rung.



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Start
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Step 1
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	Step 2
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	Step 3
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	Step 4

Figure 18-3 Bit file containing sequencer step data.



SQO instruction steps through the bit file. Each bit file word is a sequence step.

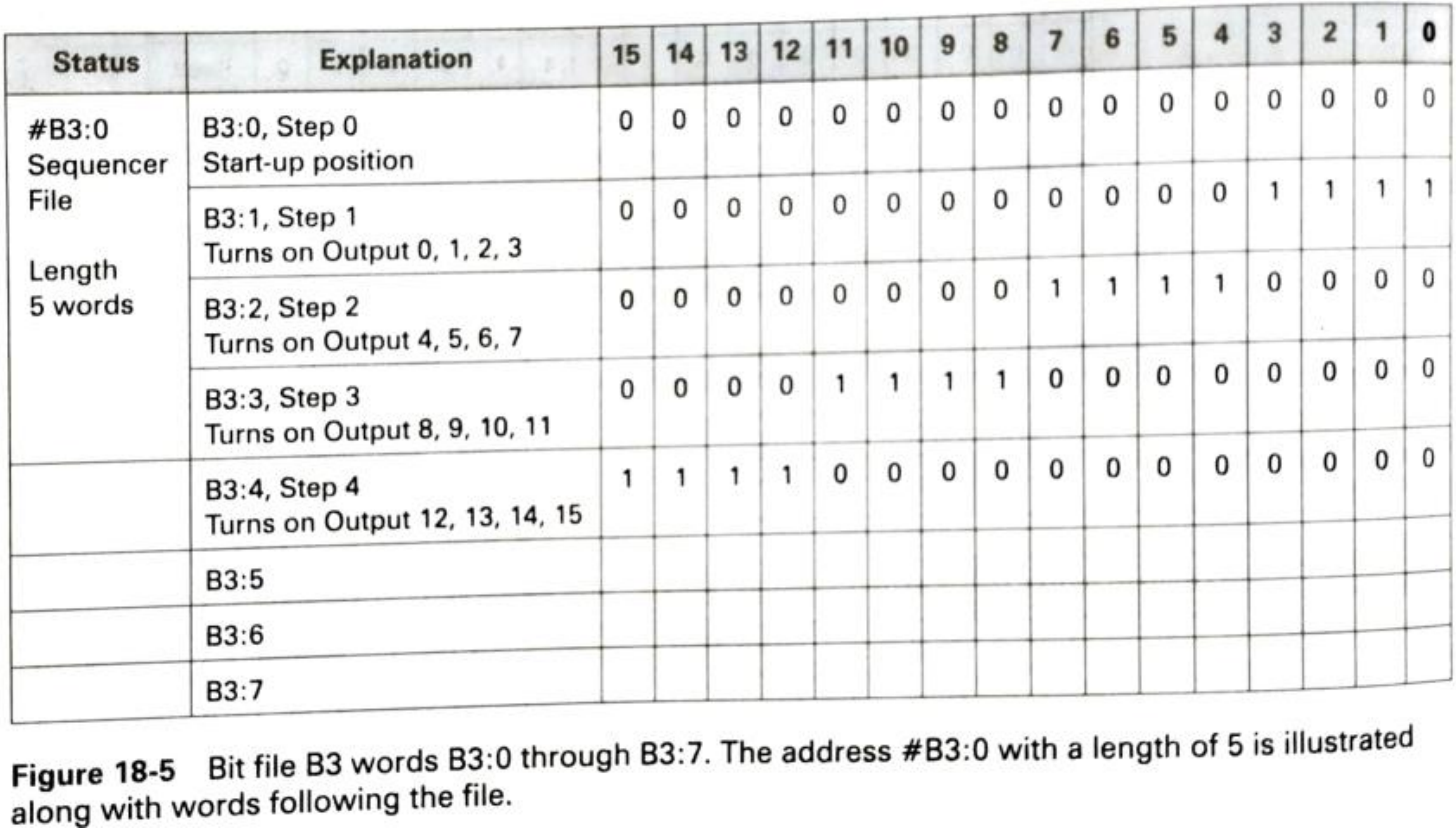
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Start
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Step1
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	Step2
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	Step3
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	Step4

Each bit file word (sequence step) is passed through the programmed hexadecimal mask.

Source bit file word	0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1
Mask passes all bits	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Bits to output status file	0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1

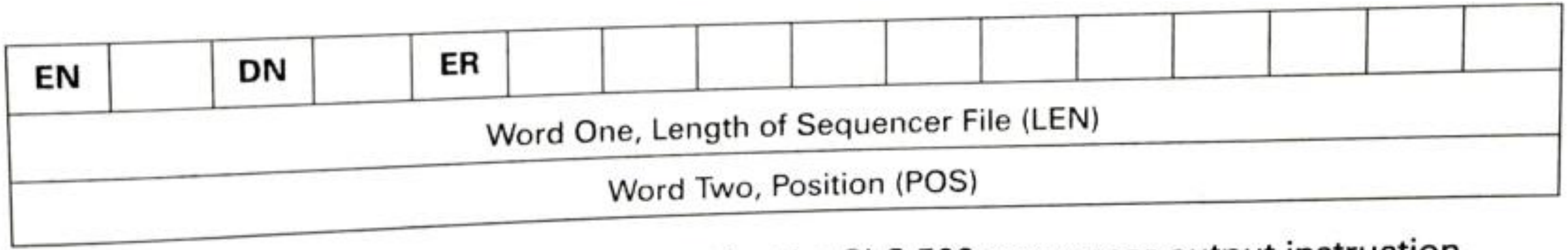
After passing through the mask, the resulting sequence word is sent to the output status file, and then on to the destination. The destination for this example is output module at address 0:2.

Figure 18-4 Sequencer at step one. Sequencer file word 0000 0000 0000 1111 will be sent to the output status file designated by the sequencer instruction, and then out to the output module.



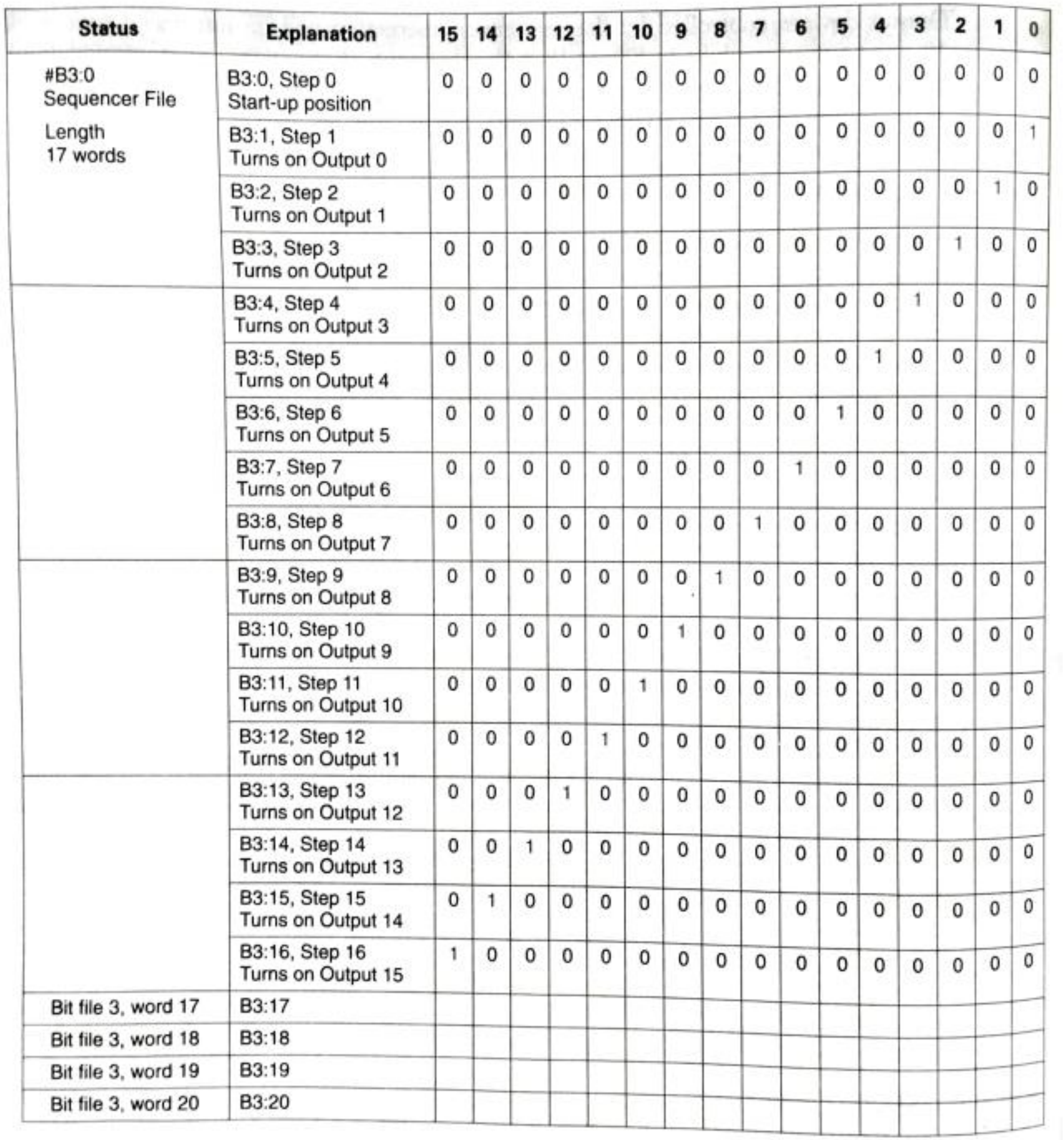
Status	Explanation	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
#B3:0 Sequencer File Length 5 words	B3:0, Step 0 Start-up position	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	B3:1, Step 1 Turns on Output 0, 1, 2, 3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
	B3:2, Step 2 Turns on Output 4, 5, 6, 7	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
	B3:3, Step 3 Turns on Output 8, 9, 10, 11	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
	B3:4, Step 4 Turns on Output 12, 13, 14, 15	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
	B3:5																
	B3:6																
	B3:7																

Figure 18-5 Bit file B3 words B3:0 through B3:7. The address #B3:0 with a length of 5 is illustrated along with words following the file.



EN		DN		ER											
Word One, Length of Sequencer File (LEN)															
Word Two, Position (POS)															

Figure 18-6 Three-word control element for the SLC 500 sequencer output instruction.



Status	Explanation	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
#B3:0 Sequencer File Length 17 words	B3:0, Step 0 Start-up position	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	B3:1, Step 1 Turns on Output 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	B3:2, Step 2 Turns on Output 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	B3:3, Step 3 Turns on Output 2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	B3:4, Step 4 Turns on Output 3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	B3:5, Step 5 Turns on Output 4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
	B3:6, Step 6 Turns on Output 5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	B3:7, Step 7 Turns on Output 6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
	B3:8, Step 8 Turns on Output 7	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	B3:9, Step 9 Turns on Output 8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	B3:10, Step 10 Turns on Output 9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	B3:11, Step 11 Turns on Output 10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	B3:12, Step 12 Turns on Output 11	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
	B3:13, Step 13 Turns on Output 12	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	B3:14, Step 14 Turns on Output 13	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	B3:15, Step 15 Turns on Output 14	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	B3:16, Step 16 Turns on Output 15	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit file 3, word 17	B3:17																
Bit file 3, word 18	B3:18																
Bit file 3, word 19	B3:19																
Bit file 3, word 20	B3:20																

Figure 18-7 Seventeen-word, sixteen-step sequencer file.

PROGRAMMING THE SEQUENCER OUTPUT INSTRUCTION

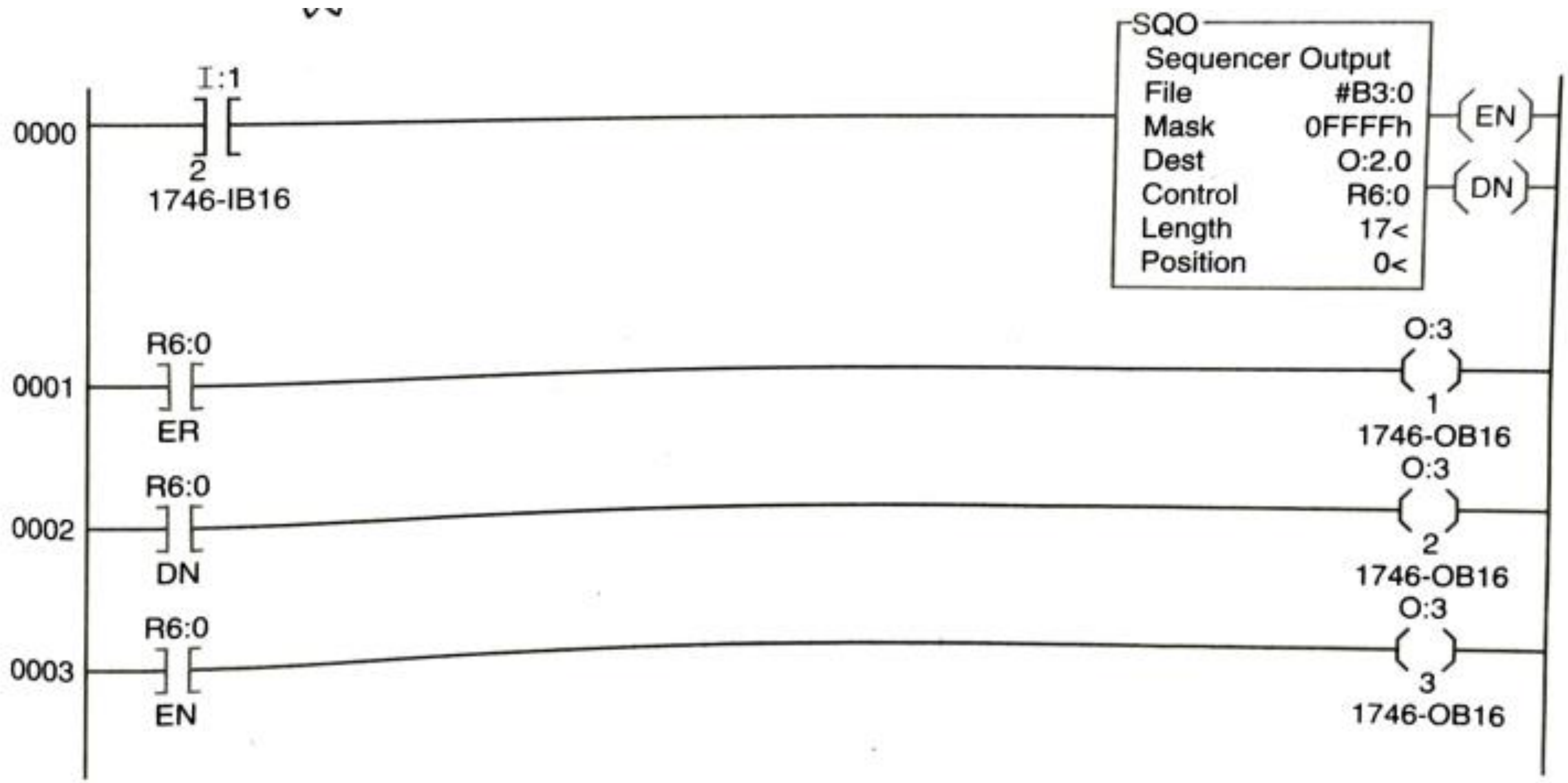
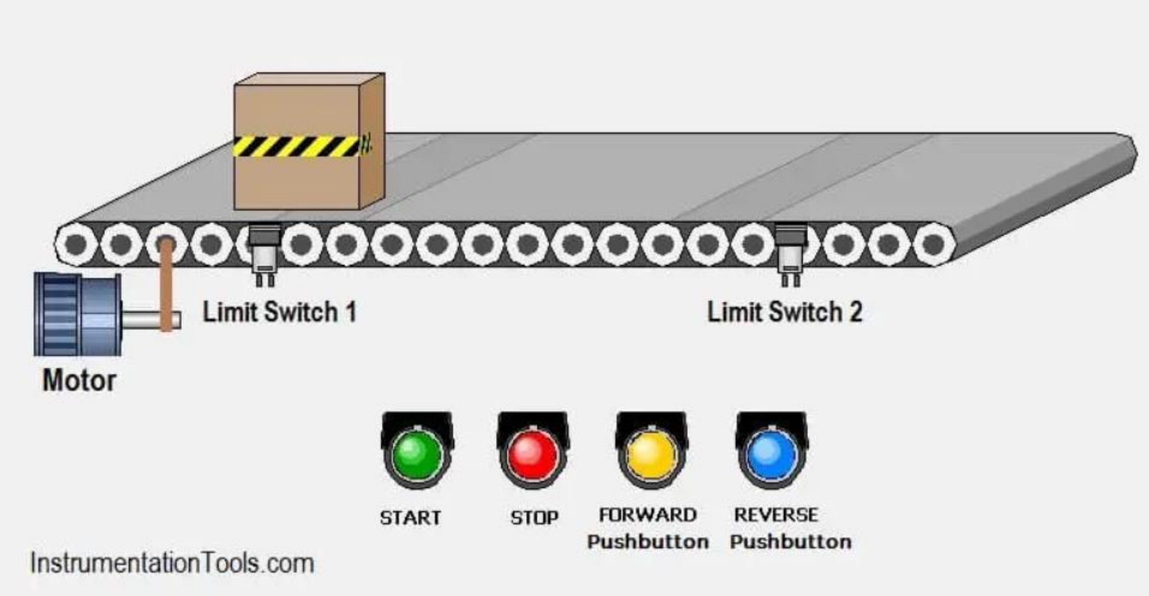


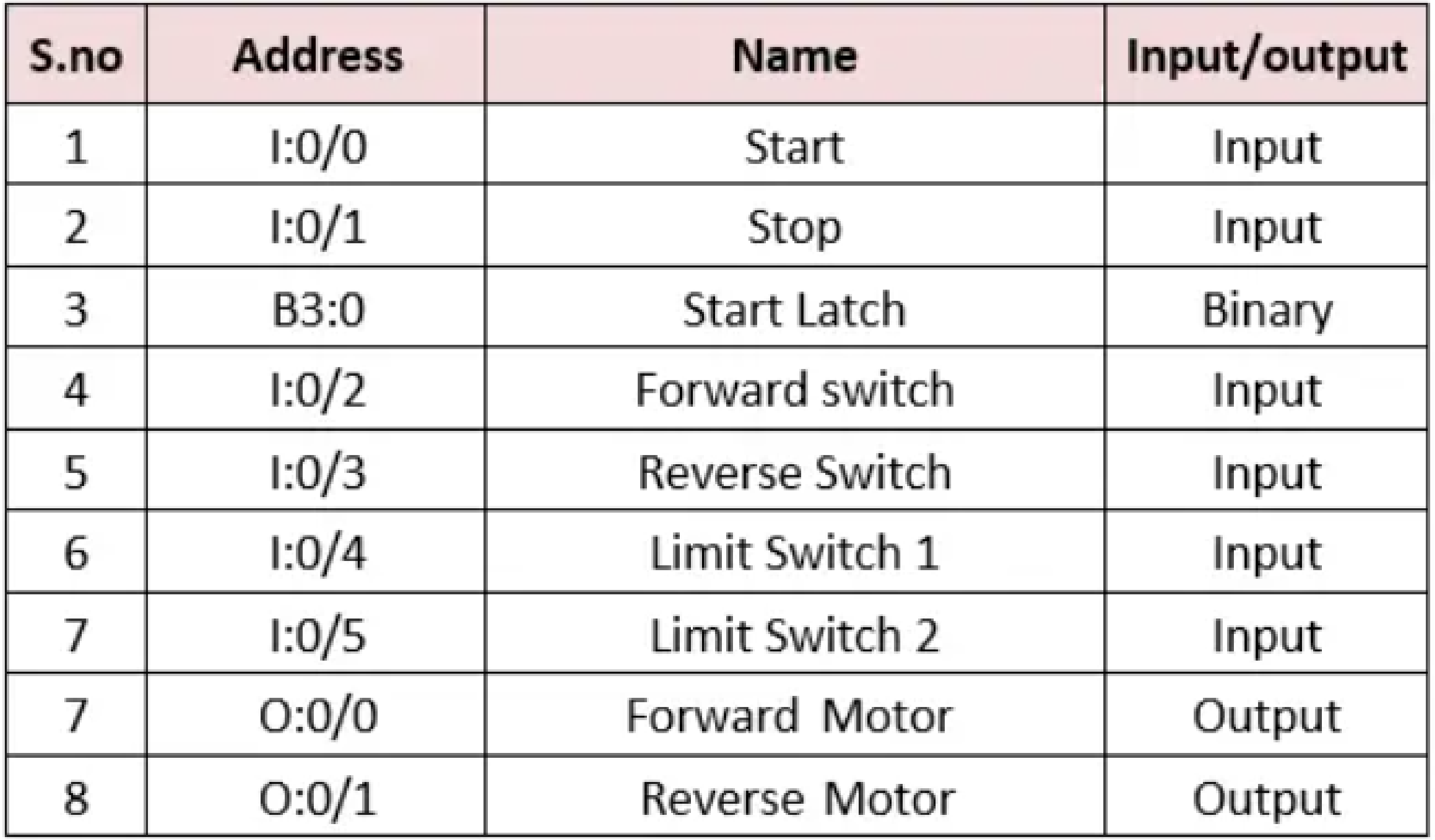
Figure 18-8 RSLogix 500 sequencer output instruction and associated status bit rungs.

Problem-1

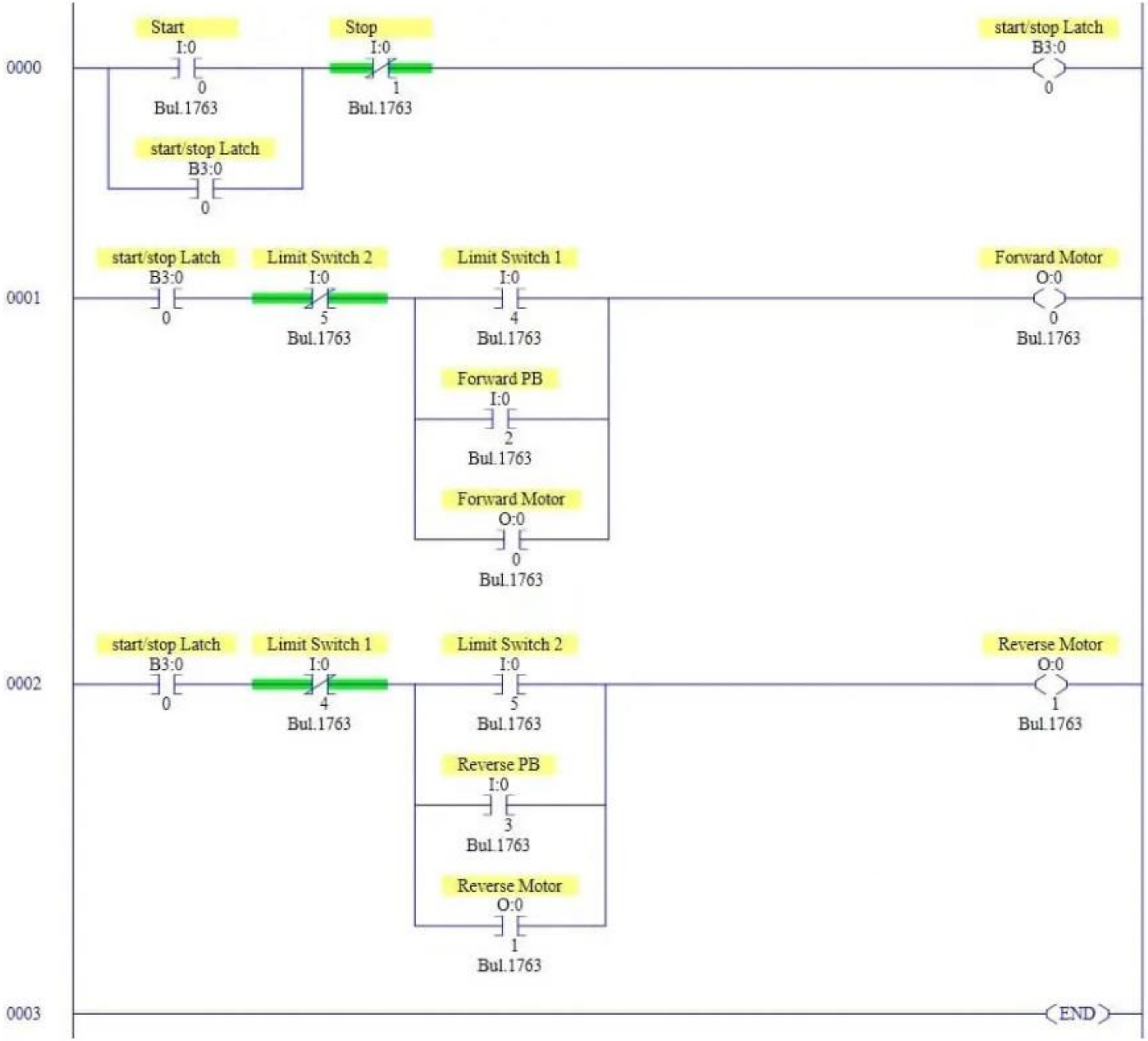
The Workpiece starts moving on the left side and moves to the right when the start button is pressed. When it reaches the rightmost limit, the drive motor reverses and brings the workpiece back to the left most position again and the process repeats. The forward and reverse push buttons provides a means of starting the motor in either forward or reverse so that the limit switches can take over automatic control. Draw ladder diagram to execute this using a PLC.

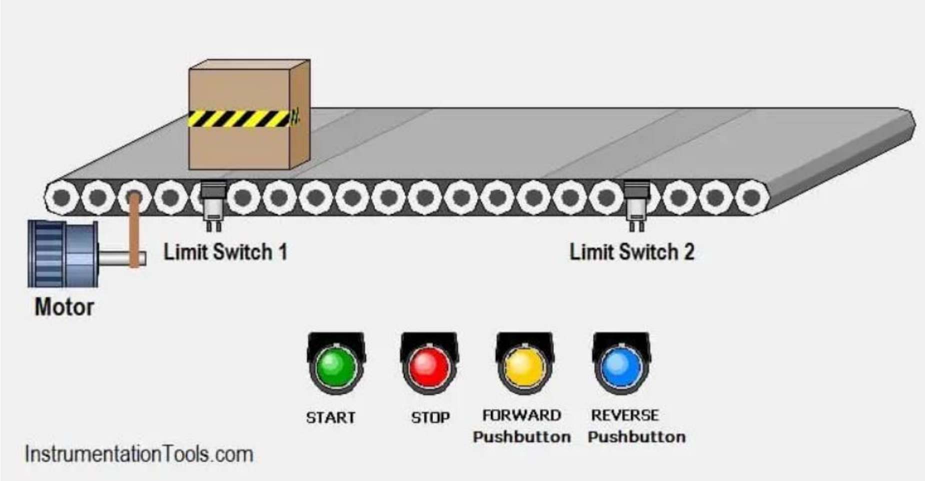


List of Inputs and Outputs



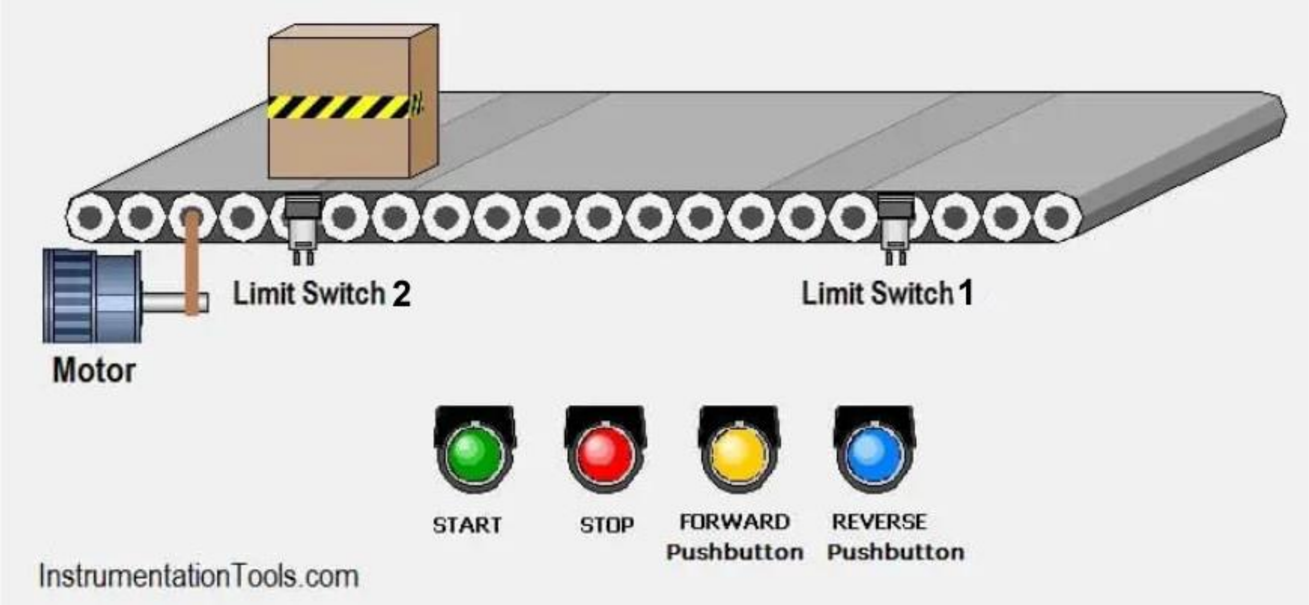
S.no	Address	Name	Input/output
1	I:0/0	Start	Input
2	I:0/1	Stop	Input
3	B3:0	Start Latch	Binary
4	I:0/2	Forward switch	Input
5	I:0/3	Reverse Switch	Input
6	I:0/4	Limit Switch 1	Input
7	I:0/5	Limit Switch 2	Input
7	O:0/0	Forward Motor	Output
8	O:0/1	Reverse Motor	Output


0000
Start
I:0
0
Bul.1763
Stop
I:0
1
Bul.1763
start/stop Latch
B3:0
0
start/stop Latch
B3:0
0
0001
start/stop Latch
B3:0
0
Limit Switch 2
I:0
5
Bul.1763
Limit Switch 1
I:0
4
Bul.1763
Forward PB
I:0
2
Bul.1763
Forward Motor
O:0
0
Bul.1763
Forward Motor
O:0
0
Bul.1763
0002
start/stop Latch
B3:0
0
Limit Switch 1
I:0
4
Bul.1763
Limit Switch 2
I:0
5
Bul.1763
Reverse PB
I:0
3
Bul.1763
Reverse Motor
O:0
1
Bul.1763
Reverse Motor
O:0
1
Bul.1763
0003
END


Limit Switch 1
Limit Switch 2
Motor
START
STOP
FORWARD Pushbutton
REVERSE Pushbutton
InstrumentationTools.com

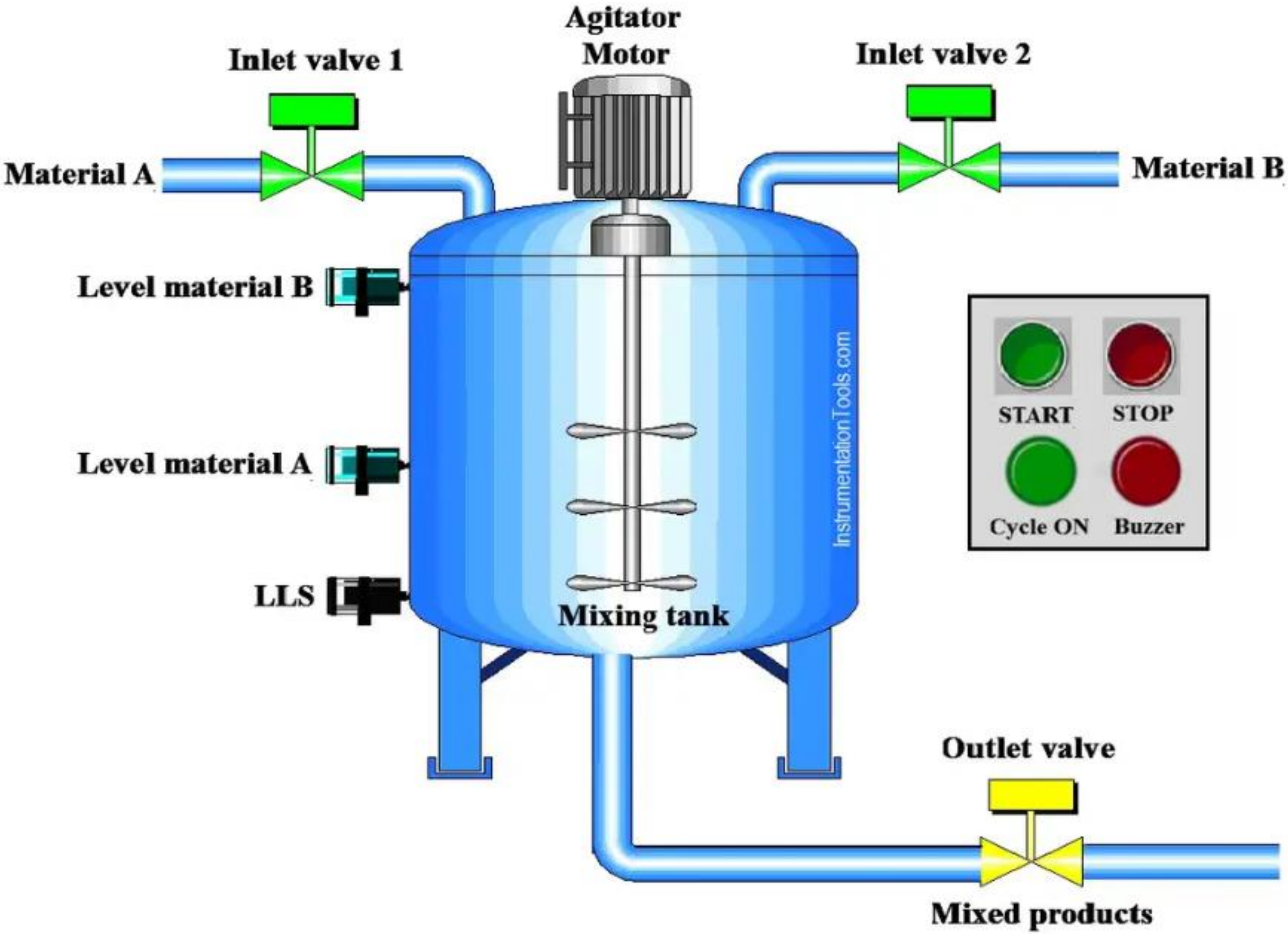
Problem-2

A motor coupled to a work table is used to move workpiece between two limit switches. When the process starts, the workpiece should move from its position in the forward direction until it encounters limit switch LS1. Once it contacts LS1 it should start to move in reverse direction until it encounters limit switch LS2. This process should repeat until a manual stop button is pressed. A start button should be pressed to turn on the process. Draw ladder diagram to execute this using a PLC.

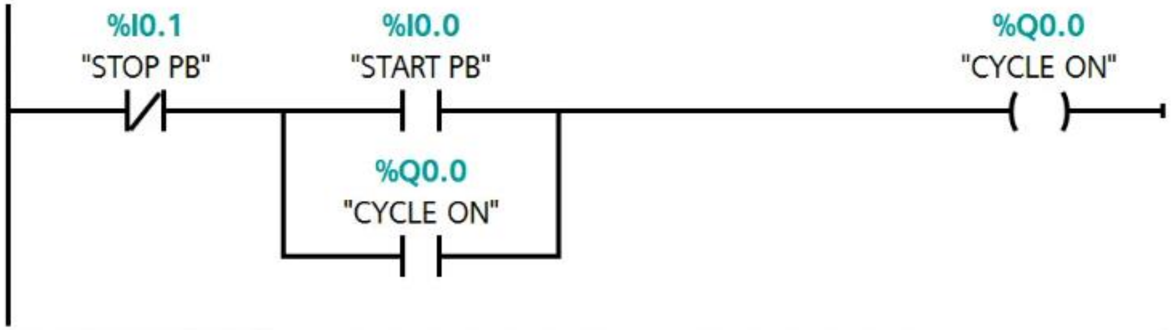


Problem-3

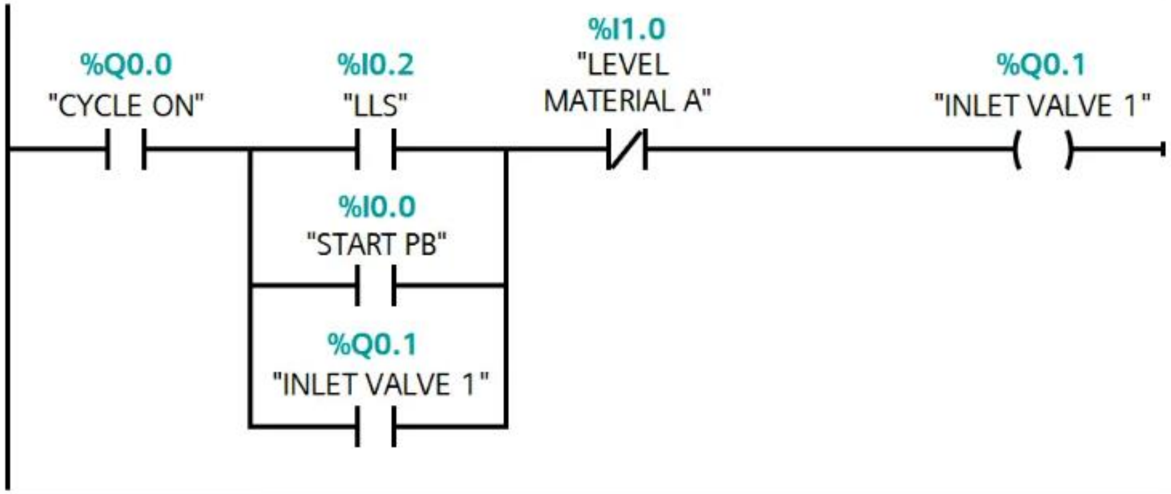
Material A and material B are collected in a tank. These materials will be mixed for 20 sec and then the mixed product drained out through the outlet valve. Two level sensors are used for detecting the level of material A and material B. Also one low level sensor used for detecting the bottom level. To control level of this system, valve is used which has two states, either fully opened or fully closed. After successfully completion of mixing, outlet valve is operated to drain the mixed material. When mixing process is completed, buzzer will be activated and it will remain ON and after 5 sec it will be automatically OFF. Draw ladder diagram to execute this using a PLC.


Agitator Motor
Inlet valve 1
Inlet valve 2
Material A
Material B
Level material B
Level material A
LLS
InstrumentationTools.com
START
STOP
Cycle ON
Buzzer
Mixing tank
Outlet valve
Mixed products

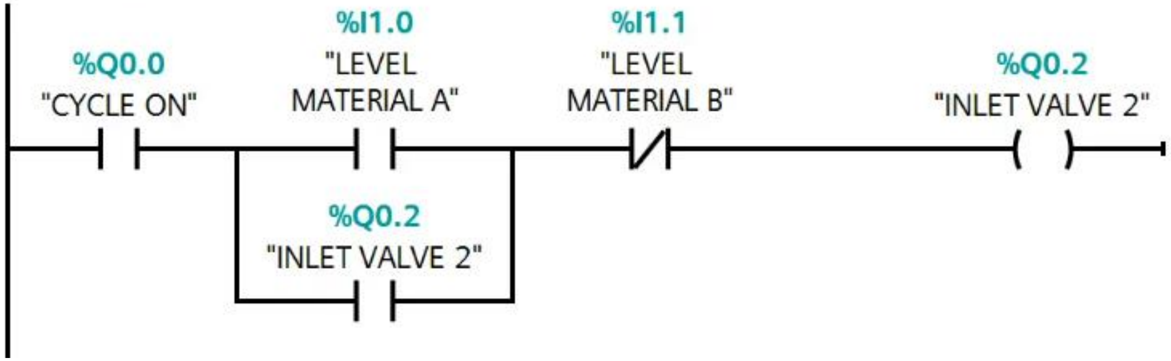
Network 1 : Cycle ON InstrumentationTools.com

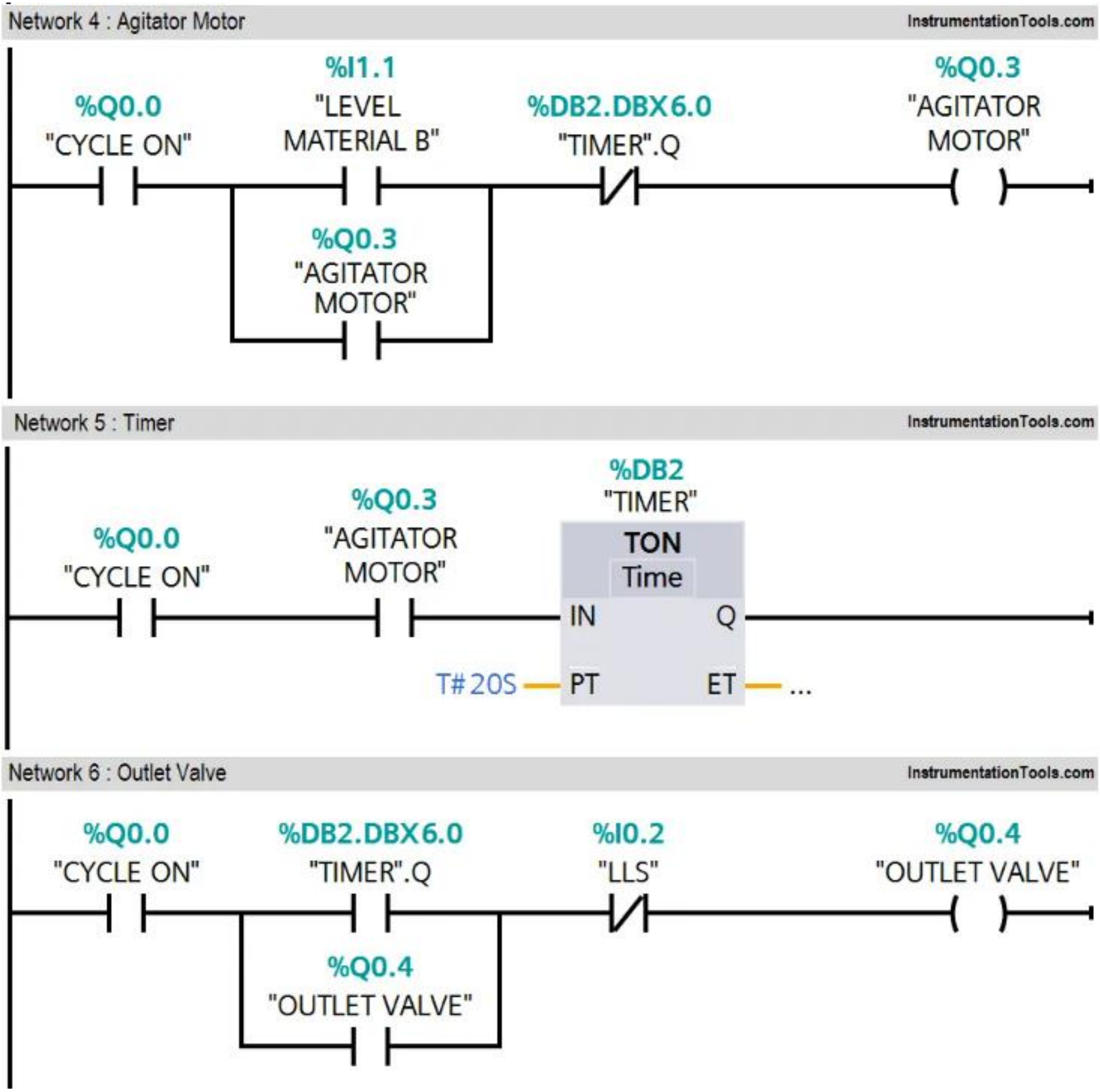


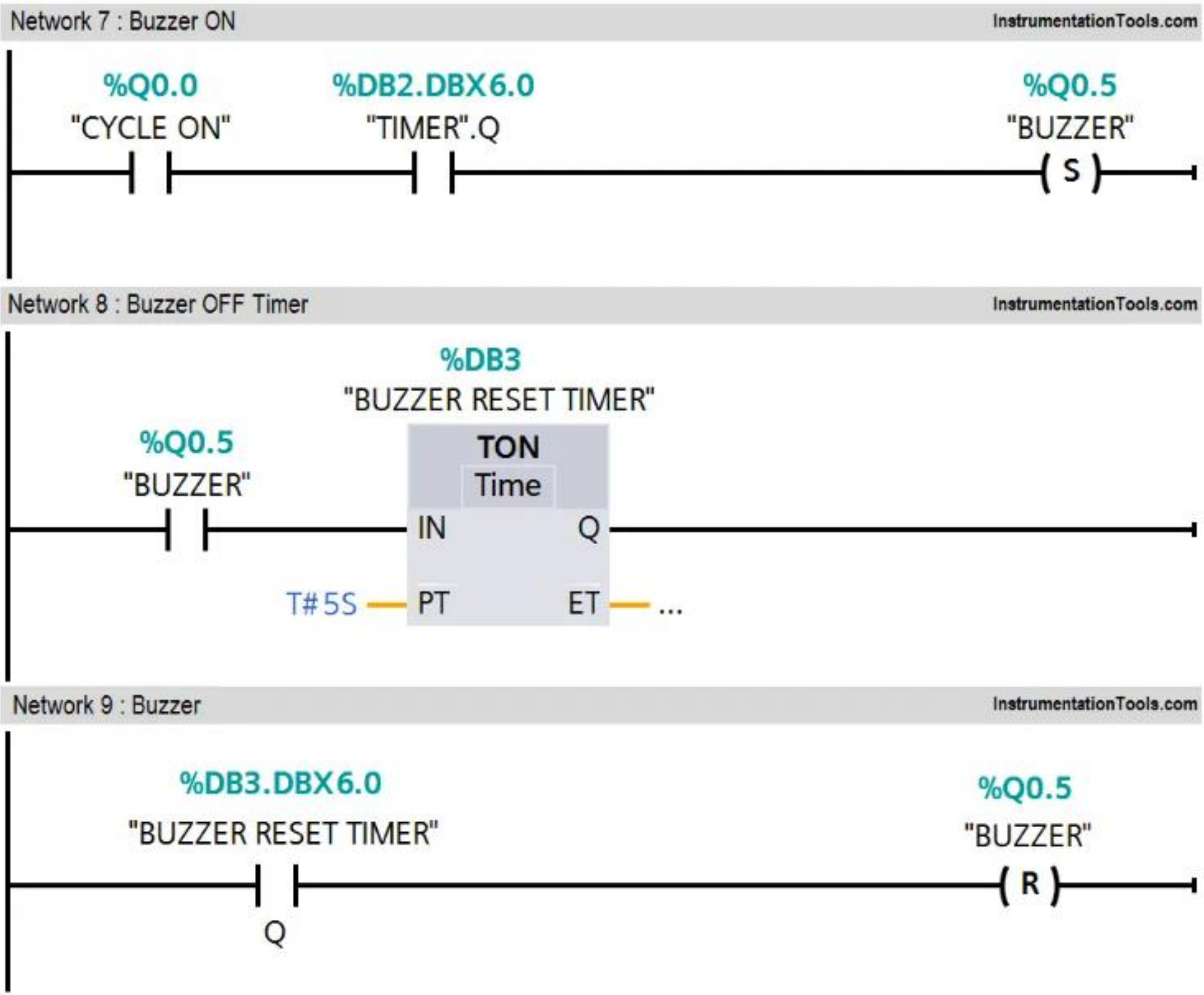
Network 2 : Inlet Valve 1 InstrumentationTools.com



Network 3 : Inlet Valve 2 InstrumentationTools.com




Network 4 : Agitator Motor
InstrumentationTools.com
%Q0.0
"CYCLE ON"
%I1.1
"LEVEL MATERIAL B"
%Q0.3
"AGITATOR MOTOR"
%DB2.DBX6.0
"TIMER".Q
%Q0.3
"AGITATOR MOTOR"
Network 5 : Timer
InstrumentationTools.com
%Q0.0
"CYCLE ON"
%Q0.3
"AGITATOR MOTOR"
%DB2
"TIMER"
TON
Time
IN
Q
T#20S
PT
ET
...
Network 6 : Outlet Valve
InstrumentationTools.com
%Q0.0
"CYCLE ON"
%DB2.DBX6.0
"TIMER".Q
%Q0.4
"OUTLET VALVE"
%I0.2
"LLS"
%Q0.4
"OUTLET VALVE"



Network 7 : Buzzer ON

Network 8 : Buzzer OFF Timer

Network 9 : Buzzer

Problem-4

In a soft drink plant a tank will be filled with two fluids, mixed and then the tank is to be drained. When the start button is pressed pump A will be turned on and valve A will be opened. Pump A runs for 15 seconds fill the tank with fluid A and will be turned off and the valve A will be closed, after pump A turns off pump B turns on and Valve B will open. Pump B runs for 10 seconds fill the tank with fluid B and will be turned off and Valve B get closed. After pump B turns off mixer motor will get started, runs for 90 seconds. After mixing outlet valve C opens, pump C turns on runs for 25 seconds. After mixing outlet valve C opens, pump C turns on runs for 25 seconds and will be turned off. This process repeats for 10 cycles and then the process stops. A manual stop button is also provided in the panel for safety. Develop a PLC ladder program.

Supervisory Control and Data Acquisition System

The basic functions carried out by a SCADA system are:

1. Channel scanning
2. Conversion into engineering units
3. Data processing

SCADA (Continued..)

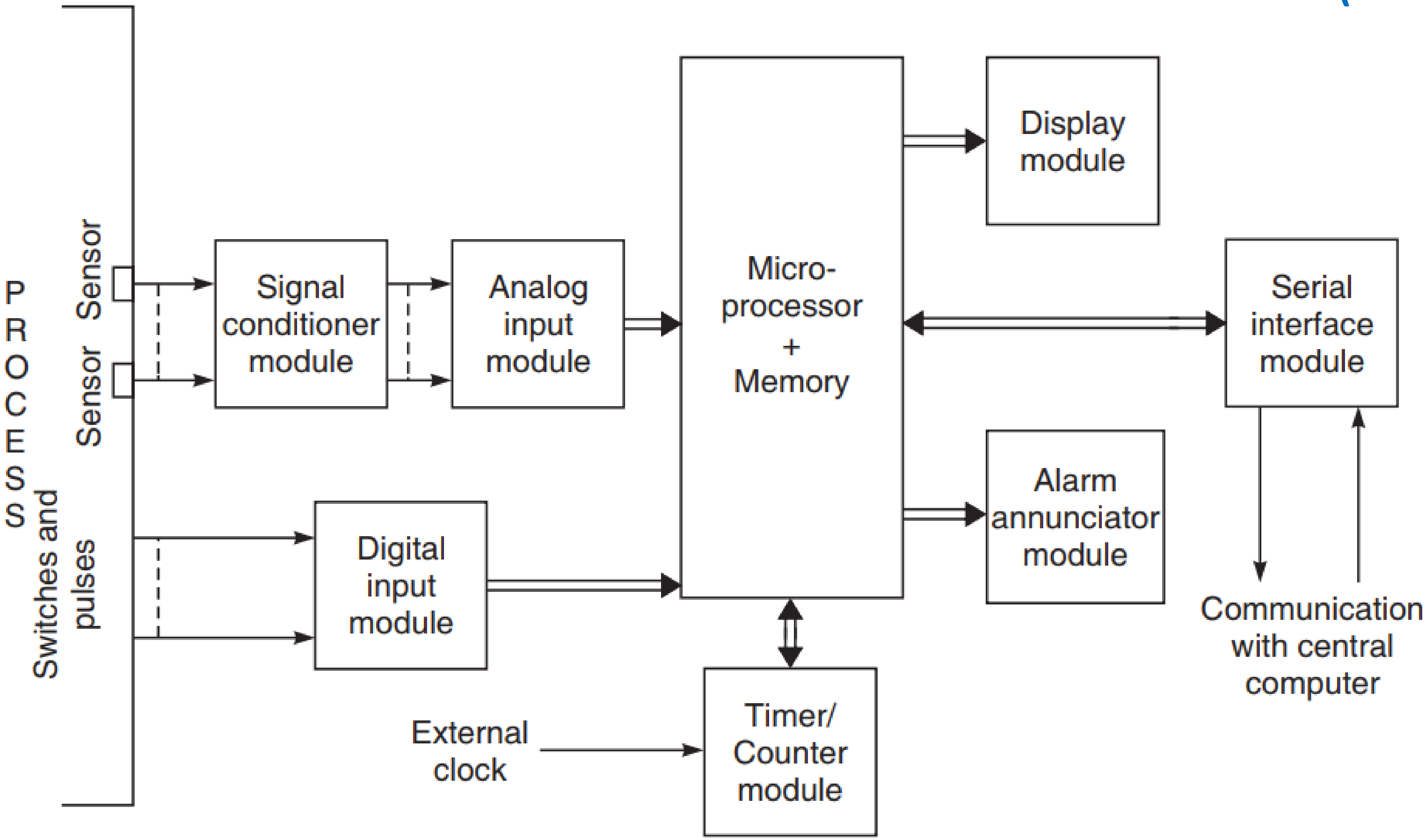


Figure 3.51 Supervisory control and data acquisition system.

SCADA (Continued..)

1. Channel Scanning

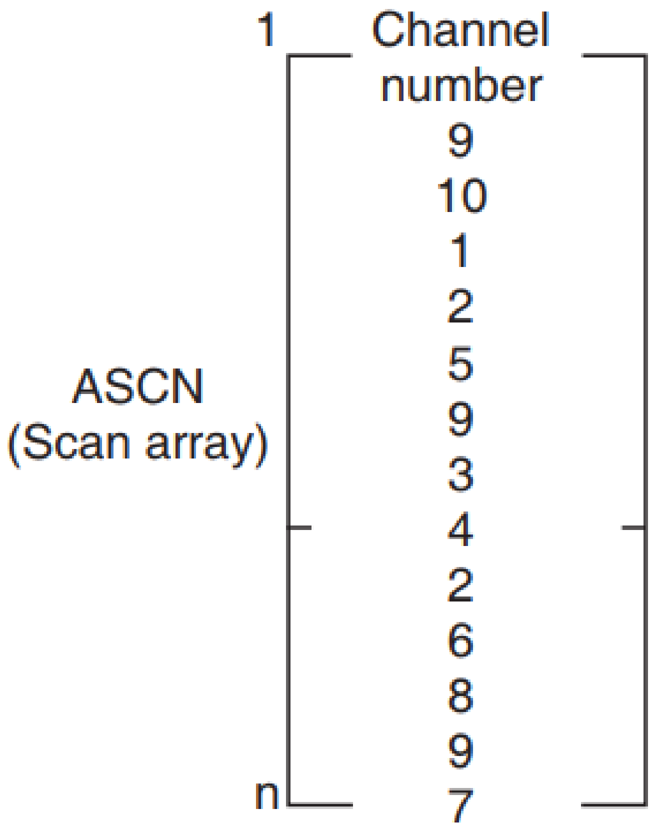
- There are many ways in which microprocessor can address the various channels and read the data.
- The channel scanning and reading of data requires, the following actions to be taken:
    1. Sending channel address to multiplexer
    2. Sending start convert pulse to ADC
    3. Reading the digital data.

SCADA : Channel Scanning (Continued..)

Polling

- The microprocessor scans the channels to read the data, and this process is called polling.
- In polling, the action of selecting a channel and addressing it, is the responsibility of processor.
- The channel selection may be sequential or in any particular order decided by the designer.
- It is also possible to assign priority to some channels over others, i.e. some channels can be scanned more frequently than others.
- It is also possible to offer this facility of selecting the order of channel addressing and channel priorities to the operator level.

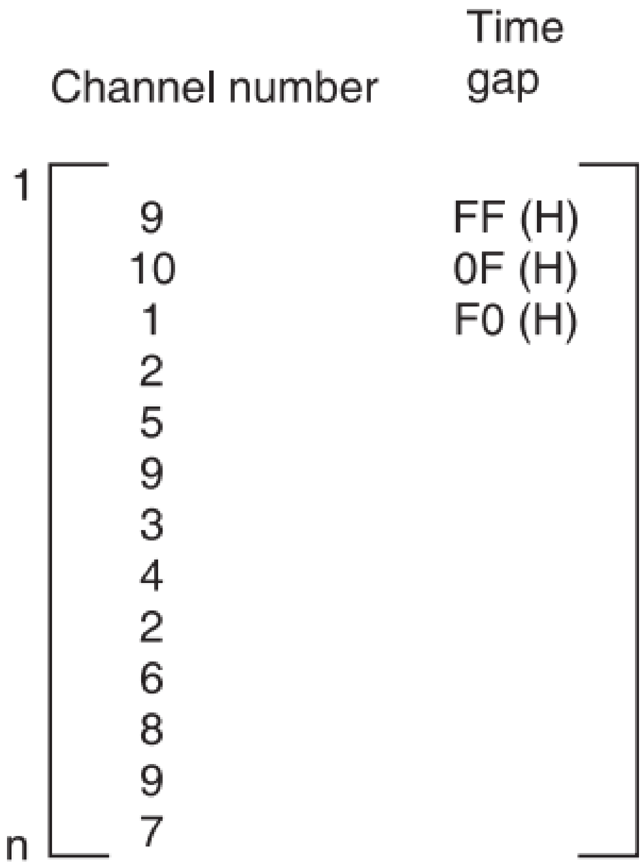
SCADA : Channel Scanning (Continued..)



ASCN
(Scan array)

1 … n	Channel number
1	9
	10
	1
	2
	5
	9
	3
	4
	2
	6
	8
	9
n	7

Figure 3.52 Channel scan array.



1 … n	Channel number	Time gap
1	9	FF (H)
	10	0F (H)
	1	F0 (H)
	2	
	5	
	9	
	3	
	4	
	2	
	6	
	8	
	9	
n	7	

Figure 3.54 Scan array with time.

SCADA : Channel Scanning (Continued..)

- The processor may scan the channels continuously in the particular order or the channels may be scanned after every fixed time period.
- The fixed time period approach requires a timer/counter circuit whose output is connected to interrupt request input.
- The scan routine for one channel is incorporated in ‘Interrupt Service Routine’.

SCADA : Channel Scanning (Continued..)

Interrupt scanning

- In interrupt scanning, transducer check for violation of limits.
- It sends interrupt request signal to processor when the analog signal from transducer is not within High and Low limits boundary set by Analog High and Analog Low signals.
- This is also called Scanning by Exception.
- When any parameter exceeds the limits, then the limit checking circuit would send interrupt request to microprocessor which in turn would monitor all parameters till the parameter values come back within pre-specified limits.
- This allows for a detailed analysis of the system and the problems by the SCADA system.

SCADA : Channel Scanning (Continued..)

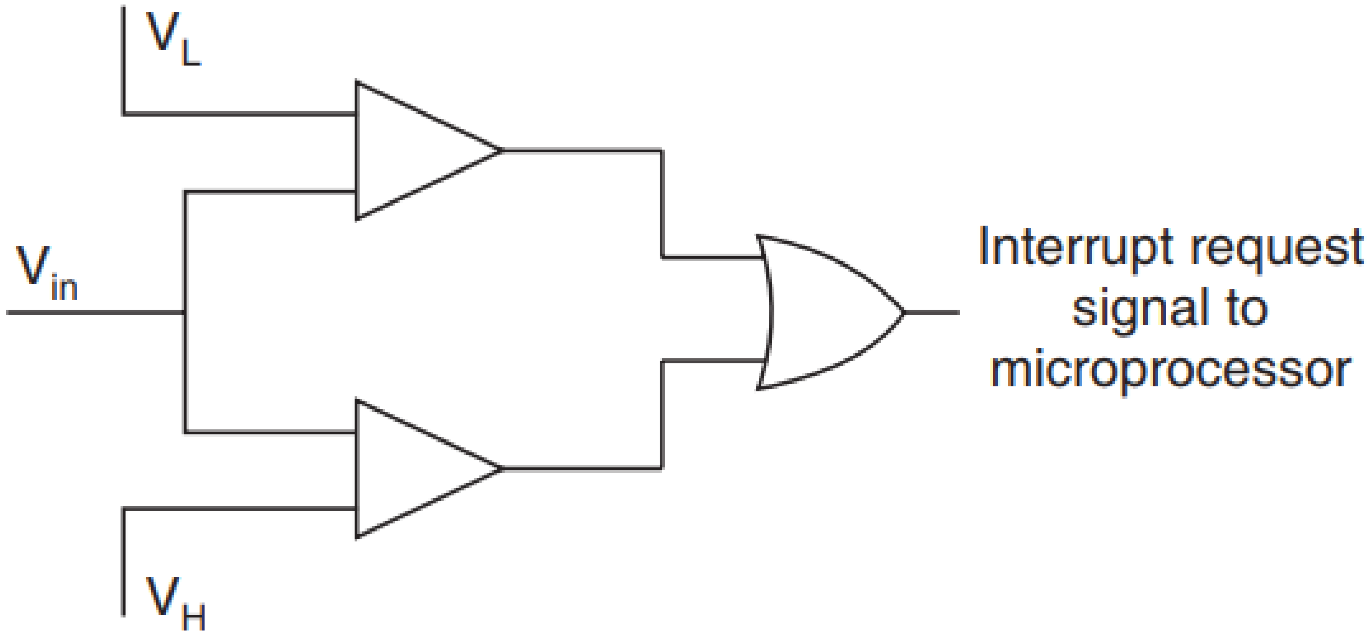


Figure 3.55 Interrupt request generation on limit violation.

SCADA (Continued..)

2. Conversion into engineering units

- The data read from the output of ADC should be converted to the equivalent engineering units before any analysis is done or the data is sent for display or printing.
- For an 8-bit ADC working in unipolar mode the output ranges between 0 and 255.
- An ADC output value will correspond to a particular engineering value based on the following parameters.
    1. Calibration of transmitters
    2. ADC mode and digital output lines

SCADA : Conversion into engineering units (Continued..)

- The conversion of ADC output to engineering units involve multiplication by conversion factor.
- The conversion factor is based on the ADC type, mode and the transmitter range.

SCADA (Continued..)

3. Data Processing

- The data read from the ADC output for various channels is processed by the microprocessor to carry out limit checking and performance analysis.
- For limit checking the Highest and Lowest Limits for each channel are stored in an array.
- When any of the two limits is violated for any channel, appropriate action like alarm generation, printing, etc. is initiated.

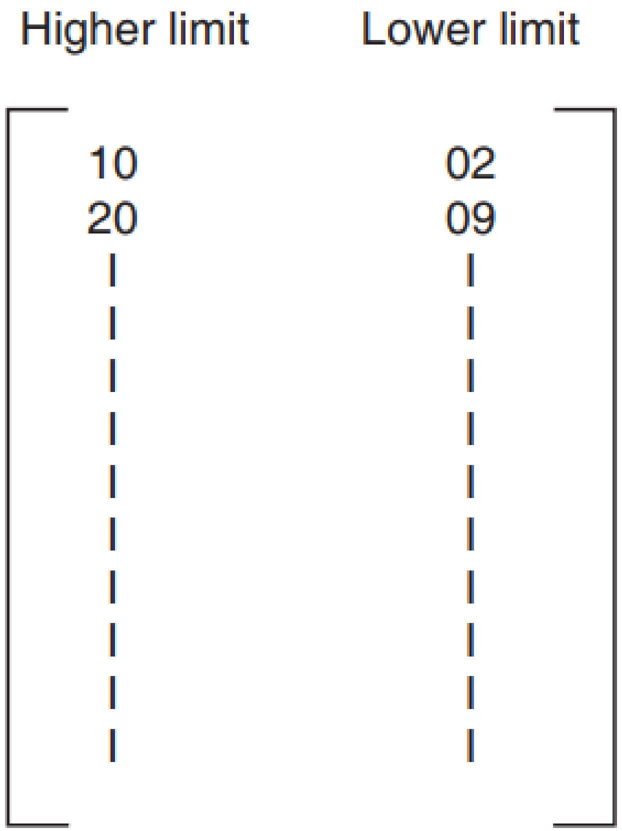


Figure 3.56 Limit array.

SCADA (Continued..)

- In addition to limit checking, the system performance may also be analysed and report could be generated for the manager level.
- This report will enable the managers to visualise the problems in the system and to take decisions regarding system modification or alternate operational strategy to increase the system performance.

Distributed SCADA Structure

- In any application, if the number of channels are quite large then more than one SCADA system is used and the channels are distributed among them.
- But, for performance analysis on the process plant, it is mandatory that the data from various channels should reach a central location where it can be consolidated and analysed to generate the reports on plant performance.
- Figures 3.58(a) and (b) show the interfacing of number of SCADA systems with central computer in star configuration and Daisy chain configuration respectively.

Distributed SCADA Structure (Continued..)

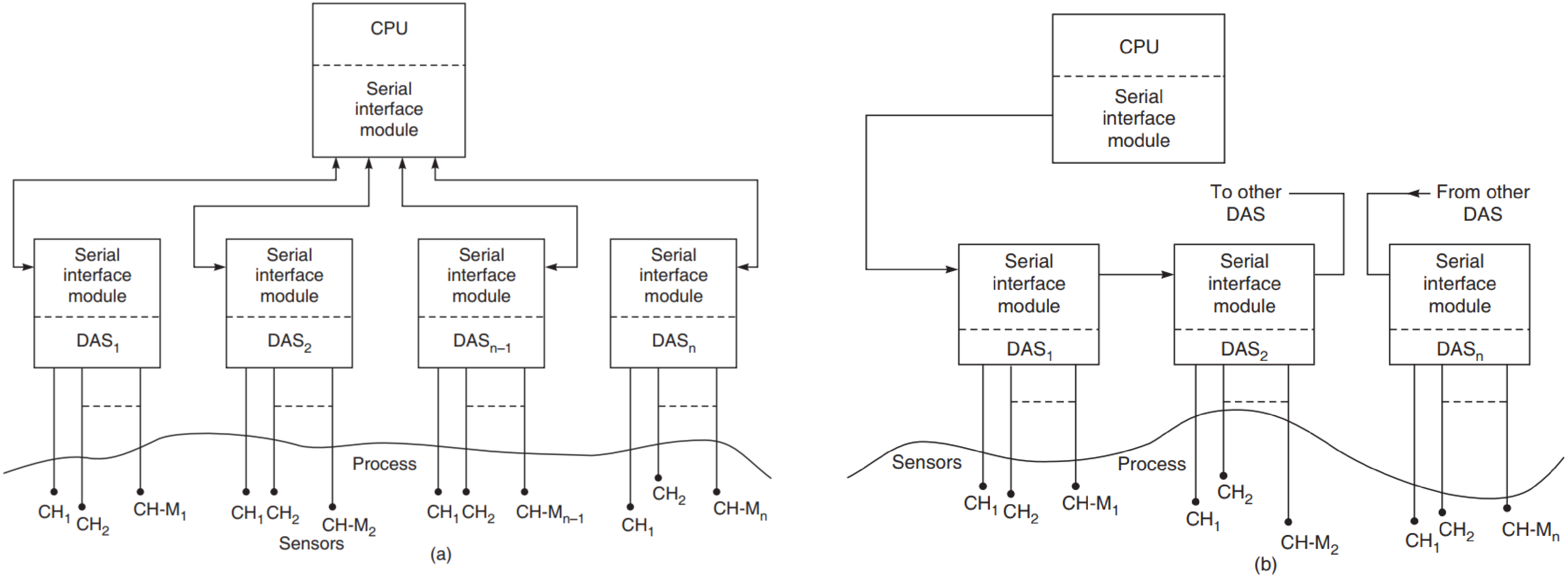
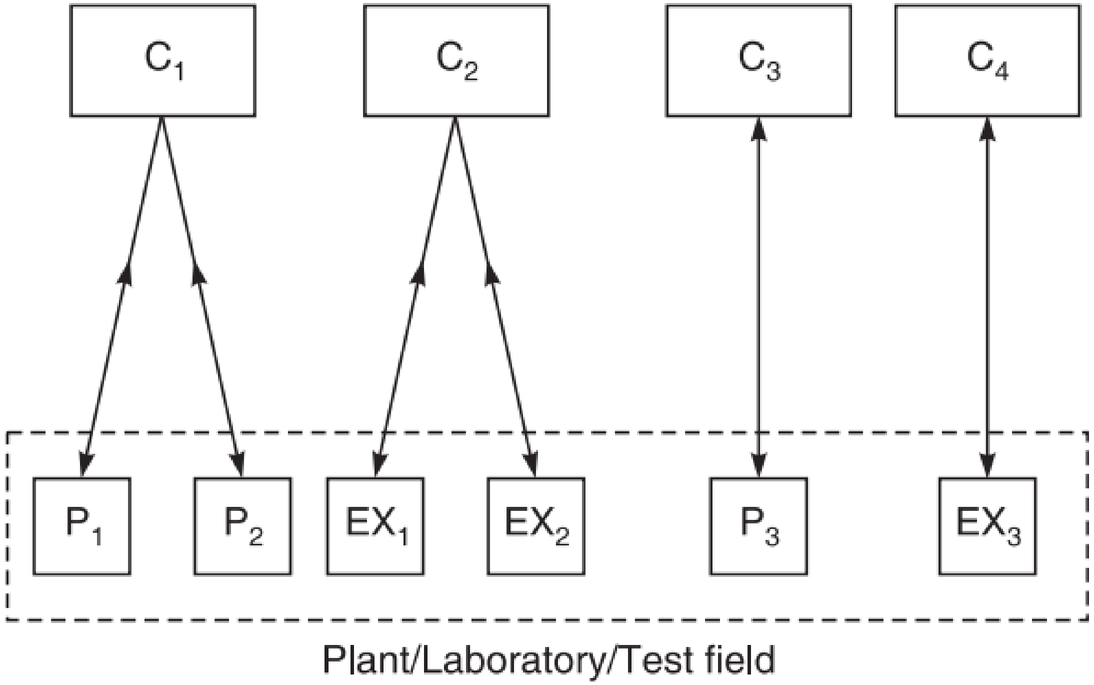


Figure 3.58 (a) Distributed SCADA structure (Star configuration), and
(b) Distributed SCADA structure (Daisy chain configuration).

DISTRIBUTED CONTROL SYSTEMS

Dedicated computers concept

- In the earlier years, the individual computers had been attached to the different parts of the plants to be automated, which has led to an assembly of distributed, mutually independent and dedicated small computers.



C_1 ·· C_4 = Dedicated Computers

P_1, P_2, P_3 = Partial Processes

EX_1, EX_2, EX_3 = Experiments

Figure 7.1 Dedicated computers concept.

Centralised computer concept

- Due to the fact that even the small dedicated computers were relatively costly, a centralised, single computer automation structure was introduced, containing a middle-scale or large-scale process control computer as its central part. Following essential functions were concentrated in the computer itself:
    - Process monitoring
    - Data acquisition
    - Alarming and logging
    - Data processing
    - Data archiving
    - Process control

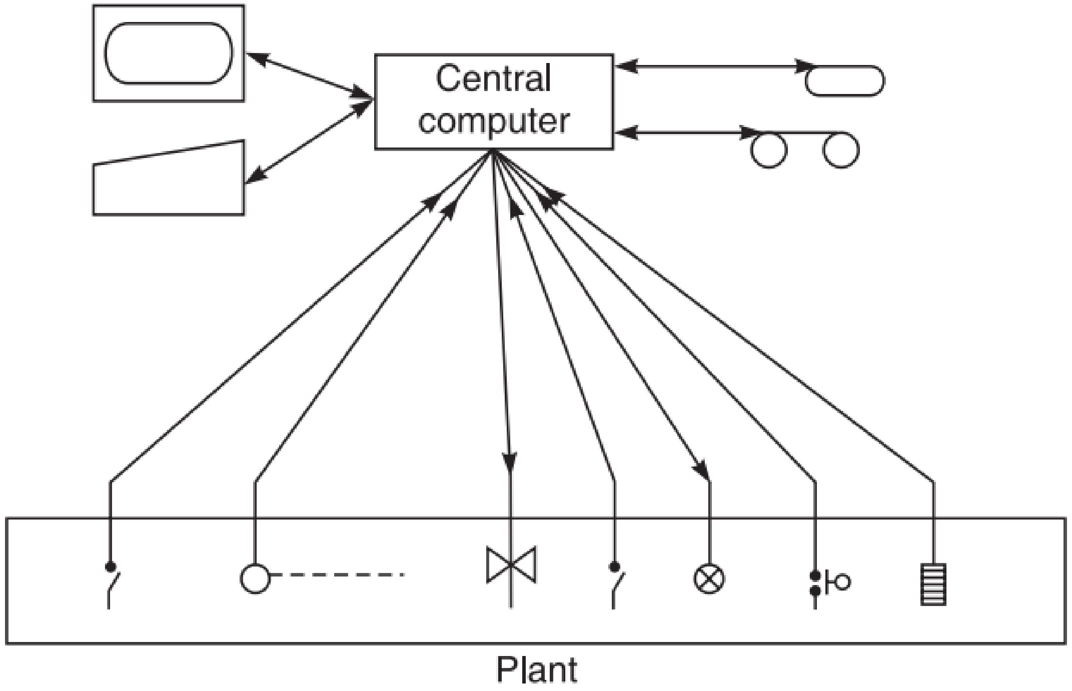


Figure 7.2 Centralised computer control concept.

Decentralised computer concept

- The main reason for transition from decentralized to the totally distributed automation concept lies in the rather inherent implementation difficulties of a decentralised system.
- These may be due to
  a. hardware problems (mainly interfacing problems), or
  b. software problems (compatibility and transportability problems).

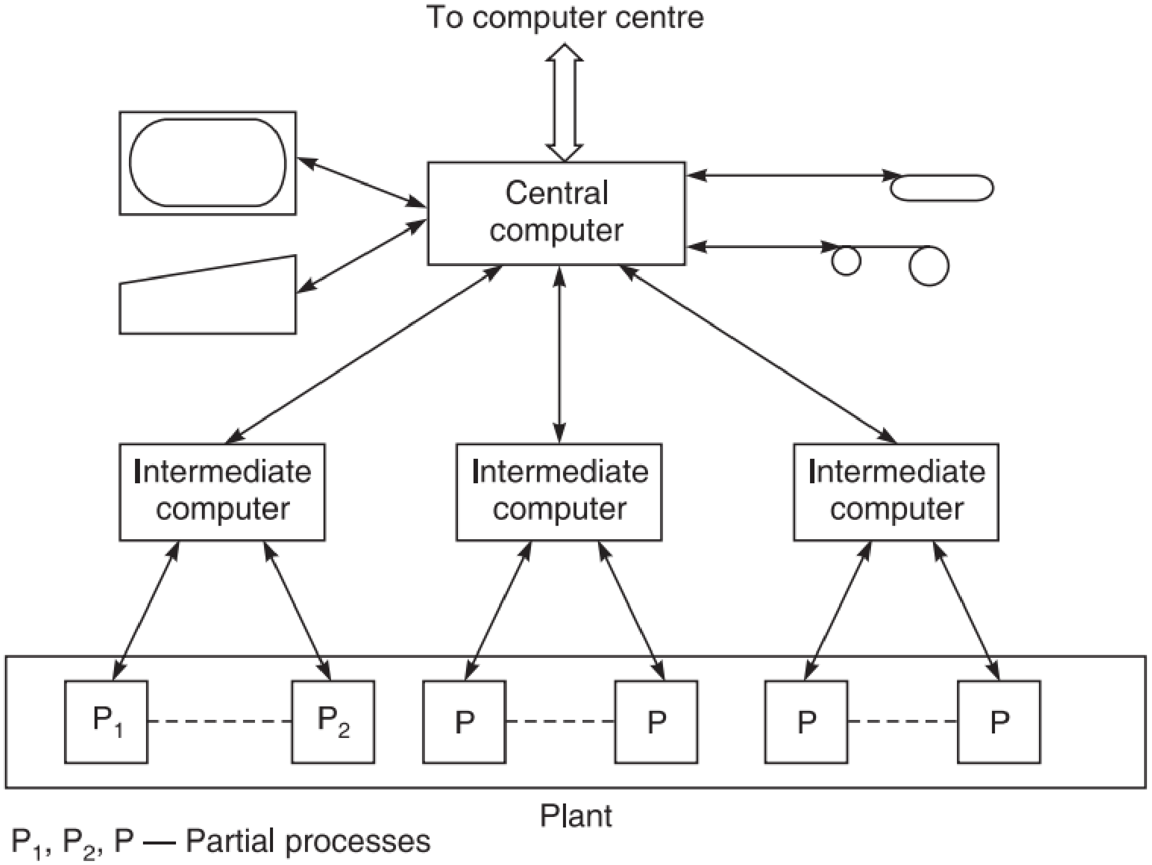


Figure 7.3 Decentralised computer control concept.

Functional Requirements of a Distributed Process Control System

- The basic functional requirements of a distributed process control system are:

    1. it should have a consistent and uniform system approach
    2. it should fulfill and perform all operational, process and plant control functions
    3. it should automatically control the process and plant in normal operation within the specified limits and tolerances but also permit manual operation

Functional Requirements of a Distributed Process Control System (Continued..)

4. it should provide at any time, the operating personnel with comprehensive information on the status of plant and process for control and maintenance purposes, fault detection and localisation
5. it should permit the manipulation without specialised programming knowledge of parameters and process control functions, as may be required or desirable from time to time
6. it should be so designed that future expansions can be easily and economically implemented
7. it should enable highly economical plant operation

Reference

1. **Introduction to Programmable Logic Controllers**, Garry Dunning, 3rd edition, Cengage learning.
2. **https://instrumentationtools.com/plc-programming-example-for-motor-forward-and-reverse/**