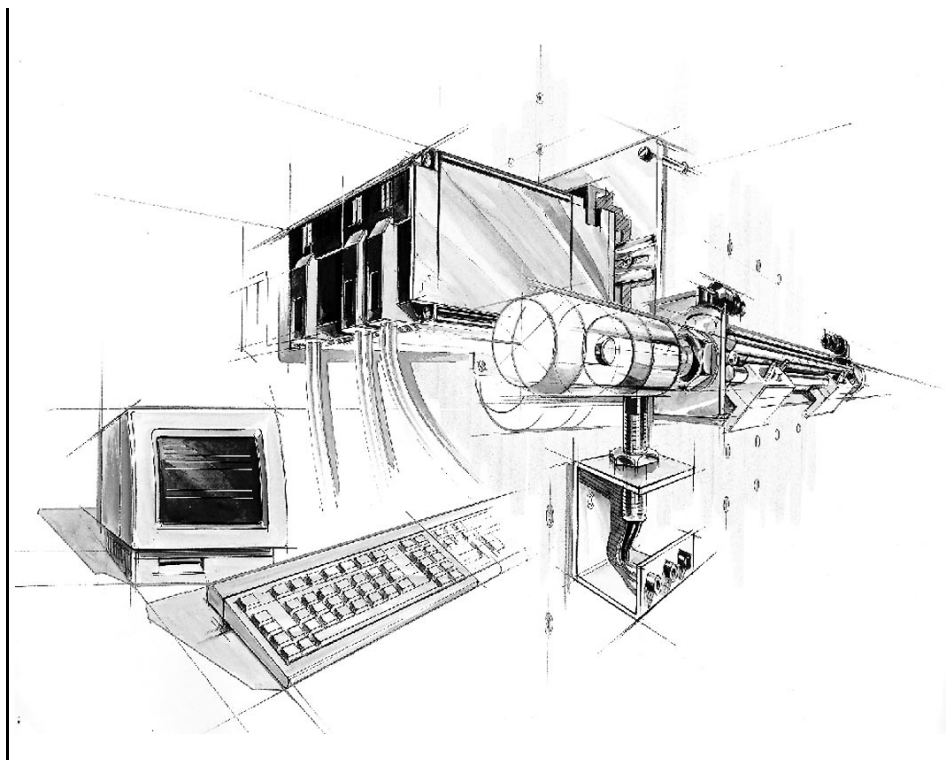


# Programmable logic Controllers

Workbook Basic level



Order No.: 093314  
Description: ARBB.SPS FPC GS  
Designation: D.S301-C-FPC-A-GB  
Edition: 12/1995  
Layout: 7.12.95, F. Ebel, M. Schwarz  
Graphics: D. Schwarzenberger  
Authors: E. v. Terzi, H. Regber, C. Löffler, F. Ebel

© Copyright by Festo Didactic KG, D-73734 Esslingen, 1995

All rights reserved, including translation rights. No part of this publication may be reproduced or transmitted in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of Festo Didactic.

## ***Preface***

The Festo Didactic Learning System for Automation and Communications is designed to meet a number of different training and vocational requirements, and the training packages are structured accordingly:

- Basic packages convey basic knowledge spanning a wide range of technologies
- Technology packages deal with important areas of open and closed-loop control technology
- Function packages explain the basic functions of automated systems
- Application packages provide basic and further training closely oriented to everyday industrial practice

The modular design of the learning system permits applications beyond the limits of the individual packages. PLC actuation, for example, is therefore possible of pneumatic, hydraulic and electrical actuators.

All learning packages have an identical structure:

- Hardware
- Teachware
- Software
- Courses

The hardware consists of industrial components and installations adapted for didactic purposes.

The courseware is matched methodologically and didactically to the training hardware. The courseware comprises:

- Textbooks (with exercises and examples)
- Workbooks (with practical exercises, worksheets, supplementary notes, solutions and data sheets)
- Overhead transparencies and videos (as a visual means of teaching support)

The teaching and learning media are available in several languages. They have been designed for use in classroom teaching, but can also be used for self-study purposes.

In the software field, computer-based training programs and programming software for programmable logic controllers are available.

Festo's Didactic range of products for basic and further training is completed by a comprehensive selection of courses matched to the contents of the technology packages.

### ***Layout of this workbook***

The workbook is structured as follows:

Section A – Course

Section B – Fundamentals

Section C – Solutions

Section D – Appendix

**Section A – Course** teaches the programming of programmable logic controllers with the help of a series of progressive exercises.

Any necessary technical knowledge required for the implementation of an exercise is provided at the beginning. Functions are limited to the most elementary requirements. More detailed knowledge may be gained in section B.

Section **C – Solutions** provides the solutions to the exercises with brief explanations.

**Section B – Fundamentals** contains generally applicable technical knowledge to supplement the training contents of the exercises in Section A. Theoretical links are established and the necessary technical terminology explained with the help of examples. An index provides an easy means of locating terminology.

**Section D – Appendix** which contains data sheets and a glossary serves as a means of reference.

Technology package TP301.....	11	<i>Table of contents</i>
Component/exercise table .....	12	
Equipment set .....	13	
Notes on safety .....	15	
Operating notes.....	16	

## **Section A – Course**

### **Components of a programmable logic controller**

Exercise 1: <b>Design and commissioning of a programmable logic controller</b>	
Components of a PLC.....	A-3

### **Programming to IEC 1131**

Exercise 2: <b>From problem to solution – taking into consideration IEC 1131-3</b>	
Practical steps for PLC programming.....	A-9

### **Basic logic operations**

Exercise 3: <b>Lamp circuit</b>	
The assignment function.....	A-15
Exercise 4: <b>Burglar alarm</b>	
The NOT function .....	A-25
Exercise 5: <b>Press with protective guard</b>	
The AND function .....	A-35
Exercise 6: <b>Bell system</b>	
The OR function .....	A-45

### **Logic control systems without latching properties**

Exercise 7: <b>Stamping device</b>	
Combination of AND/OR/NOT .....	A-55
Exercise 8: <b>Silo control system for two bulk materials</b>	
Combination circuit with branching.....	A-65

### **Logic control systems with latching properties**

Exercise 9: <b>Fire alarm</b>	
Setting an output.....	A-73
Exercise 10: <b>Drill breakage monitoring</b>	
Setting and resetting an output.....	A-81

Exercise 11: <b>Activating a cylinder</b>	
Signal edges.....	A-89

### **Logic control systems with time response**

Exercise 12: <b>Bonding of components</b>	
Pulse.....	A-101
Exercise 13: <b>Embossing device</b>	
Switch-on signal delay.....	A-111
Exercise 14: <b>Clamping device</b>	
Switch-off signal delay.....	A-121

### **Sequence control systems**

Exercise 15: <b>Lifting device for packages</b>	
Linear sequence.....	A-131
Exercise 16: <b>Lifting and sorting device for packages</b>	
Alternative branching.....	A-147
Exercise 17: <b>Stamping device with counter</b>	
Counting cycles.....	A-159

## **Section B – Fundamentals**

<b>Chapter 1 Automating with a PLC.....</b>	<b>B-1</b>
1.1 Introduction.....	B-2
1.2 Fields of application of a PLC.....	B-2
1.3 Basic design of a PLC.....	B-5
1.4 The new PLC standard IEC 1131.....	B-8

<b>Chapter 2 Fundamentals.....</b>	<b>B-11</b>
2.1 The decimal number system.....	B-12
2.2 The binary number system.....	B-12
2.3 The BCD code.....	B-14
2.4 The hexadecimal number system.....	B-14
2.5 Signed binary numbers.....	B-15
2.6 Real numbers.....	B-15
2.7 Generation of binary and digital signals.....	B-16

<b>Chapter 3 Boolean operations</b> .....	B-19
3.1 Basic logic functions .....	B-20
3.2 Further logic operations .....	B-24
3.3 Establishing switching functions .....	B-26
3.4 Simplification of logic functions .....	B-28
3.5 Karnaugh-Veitch diagram .....	B-30
<b>Chapter 4 Design and mode of operation of a PLC</b> .....	B-33
4.1 Structure of a PLC .....	B-34
4.2 Main processing unit of a PLC .....	B-36
4.3 Function mode of a PLC .....	B-38
4.4 Application program memory .....	B-40
4.5 Input module .....	B-42
4.6 Output module .....	B-44
4.7 Programming device / Personal computer .....	B-46
<b>Chapter 5 Programming of a PLC</b> .....	B-49
5.1 Systematic solution finding .....	B-50
5.2 IEC 1131-3 structuring resources .....	B-53
5.3 Programming languages .....	B-56
<b>Chapter 6 Common elements of programming languages</b> . . . .	B-61
6.1 Resources of a PLC .....	B-62
6.2 Variables and data types .....	B-66
6.3 Program organisation units .....	B-76
<b>Chapter 7 Function block diagram</b> .....	B-91
7.1 Elements of function block diagram .....	B-92
7.2 Evaluation of networks .....	B-93
7.3 Loop structures .....	B-94

<b>Chapter 8 Ladder diagram</b> .....	B-95
8.1 Elements of ladder diagram.....	B-96
8.2 Functions and function blocks .....	B-98
8.3 Evaluation of current rungs .....	B-99
<b>Chapter 9 Instruction list</b> .....	B-101
9.1 Instructions .....	B-102
9.2 Operators .....	B-103
9.3 Functions and function blocks .....	B-104
<b>Chapter 10 Structured text</b> .....	B-107
10.1 Expressions .....	B-108
10.2 Statements .....	B-110
10.3 Selection statements .....	B-112
10.4 Iteration statements .....	B-115
<b>Chapter 11 Sequential function chart</b> .....	B-119
11.1 Introduction.....	B-120
11.2 Elements of sequential function chart.....	B-120
11.3 Transitions .....	B-130
11.4 Steps.....	B-133
11.5 Example .....	B-143
<b>Chapter 12 Logic control systems</b> .....	B-147
12.1 What is a logic control system.....	B-148
12.2 Logic control systems without latching properties.....	B-148
12.3 Logic control systems with latching properties .....	B-154
12.4 Edge evaluation .....	B-157
<b>Chapter 13 Timers</b> .....	B-161
13.1 Introduction.....	B-162
13.2 Pulse timer .....	B-163
13.3 Switch-on signal delay .....	B-165
13.4 Switch-off signal delay .....	B-167



---

<b>Chapter 14 Counter</b> .....	B-171
14.1 Counter functions .....	B-172
14.2 Incremental counter .....	B-172
14.3 Decremental counter .....	B-176
14.4 Incremental/decremental counter .....	B-178
<b>Chapter 15 Sequence control systems</b> .....	B-179
15.1 What is a sequence control system .....	B-180
15.2 Function chart to IEC 848 .....	B-180
15.3 Displacement-step diagram .....	B-186
<b>Chapter 16 Commissioning and operational safety of a PLC</b> .....	B-187
16.1 Commissioning .....	B-188
16.2 Operational safety of a PLC .....	B-190
<b>Chapter 17 Communication</b> .....	B-195
17.1 The need for communication .....	B-196
17.2 Data transmission .....	B-196
17.3 Interfaces .....	B-197
17.4 Communication in the field area .....	B-198
<b>Appendix</b>	
<b>Bibliography of illustrations</b> .....	B-202
<b>Bibliography of literature</b> .....	B-203
<b>Guidelines and standards</b> .....	B-205
<b>Index</b> .....	B-209

**Section C – Solutions****Section D – Appendix****Data sheets**

Signal input, electrical . . . . .	011088
Signalling device and distributor, electrical . . . . .	030311
Single-acting cylinder . . . . .	152887
Double-acting cylinder . . . . .	152888
On/off valve with filter regulator valve . . . . .	152894
Manifold . . . . .	152896
Proximity sensor, inductive . . . . .	152902
Proximity sensor, capacitive . . . . .	152903
Proximity sensor, optical . . . . .	152904
Proximity sensor with cylinder mounting . . . . .	152905
5/2-way single solenoid valve . . . . .	152909
5/2-way double solenoid valve . . . . .	152910
Terminal unit for binary I/O . . . . .	159385
Glossary . . . . .	D-3

## **Technology package**

### **TP301 "Programmable logic controllers"**

The technology package TP301 "Programmable logic controllers" is a component part of the Festo Didactic Learning System for Automation and Communications and forms the basic level of TP300.

The training aims of TP301 are to learn how to program programmable logic controllers and to teach the fundamentals for creating programs in the programming languages 'ladder diagram' (LD), 'function block diagram' (FBD), 'instruction list' (IL), 'structured text' (ST) and 'sequential function chart' (SFC). Programming is effected in accordance with IEC 1131-3.

You have the option of using this workbook in conjunction with alternative programmable logic controllers by different manufacturers. Solutions are available for Festo FPC100 programmable logic controllers, Siemens S5-95U, AEG A120 and Mitsubishi Melsec A1S.

The exercises in this workbook may be carried out with either of two different equipment sets, i.e. a plug-in assembly board or a slotted assembly board version. A basic knowledge of electro-pneumatics and sensor technology is recommended to work through technology package TP301.

The exercises in TP301 deal with the following main topics:

- Components of a programmable logic controller
- PLC programming to IEC 1131
- Basic logic operations
- Logic control systems
- Sequence control systems

The allocation of components and exercises can be seen from the following component/exercise table.

## Allocation of component and exercise

Description	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Signal input, electrical			1	1	1	1		1	1	1	1	1	1	1		1	1
Signalling device and distributor, electrical			1	1		1			1	1							
Proximity sensor, optical							1			1					1	1	1
Proximity sensor, inductive					1		1				1		1				1
Proximity sensor, capacitive							1				1		1			1	1
Proximity sensor with cylinder mounting												1		4	4	4	4
5/2-way single solenoid valve					1		1	2			1	1	1	1	2	2	2
5/2-way double solenoid valve														1		1	1
Double-acting cylinder					1		1	2						1	2	2	2
Single-acting cylinder											1	1	1	1		1	1
On/off valve with filter regulator valve					1		1	1			1	1	1	1	1	1	1
Manifold					1		1	1			1	1	1	1	1	1	1

**Equipment set TP301, Slotted assembly board,  
Order No.: 080 261**

<b>Description</b>	<b>Order No.</b>	<b>Quantity</b>
Signal input, electrical	011 088	1
Signalling device and distributor, electrical	030 311	1
Proximity sensor, optical	152 904	1
Proximity sensor, inductive	152 902	1
Proximity sensor, capacitive	152 903	1
Proximity sensor with cylinder mounting	152 905	4
5/2-way single solenoid valve	152 909	2
5/2-way double solenoid valve	152 910	1
Double-acting cylinder	152 888	2
Single-acting cylinder	152 887	1
On/off valve with filter regulator valve	152 894	1
Manifold	152 896	1
Plastic tubing	151 496	
Quick push-pull distributor	036 315	
optional, <b>not</b> included in scope of delivery of equipment set		
Power supply unit	151 503	
Set of cables	030 332	
Plug-in adapter	035 651	

**Equipment set TP301, Plug-in assembly board,  
Order No.: 080 260**

<b>Description</b>	<b>Order No.</b>	<b>Quantity</b>
Signal input, electrical	011 088	1
Signalling device and distributor, electrical	030 311	1
Proximity sensor, optical	150 758	1
Proximity sensor, inductive	150 757	1
Proximity sensor, capacitive	150 759	1
Proximity sensor with cylinder mounting	030 331	4
5/2-way single solenoid valve	030 315	2
5/2-way double solenoid valve	030 317	1
Double-acting cylinder	013 415	2
Single-acting cylinder	011 711	1
On/off valve with filter regulator valve	011 758	1
Manifold	011 713	1
Plastic tubing	006 204	
Quick push-pull distributor	006 831	
optional, <b>not</b> included in scope of delivery of equipment set		
Power supply unit	151 503	
Set of cables	030 332	

**Notes on safety**

The following notes should be followed in the interest of safety:

- Mount all components securely on the board.
- Do not switch on compressed air until all line connections have been established and secured.
- Proceed with care when switching on the compressed air. Cylinders may advance or retract as soon as the compressed air is switched on.
- Switch off air supply immediately if air lines become detached. This prevents accidents.
- Do not disconnect air lines under pressure.
- Do not exceed the permitted working pressure of 8 bar.
- Observe general safety regulations in accordance with DIN 58 126 and VDE 0100.
- Use only extra-low voltages of up to 24 V DC.
- Observe the data sheets referring to the individual components, in particular all notes regarding safety.

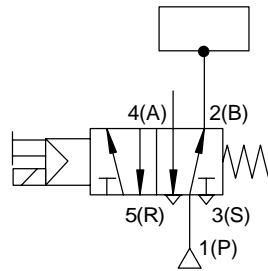


### Operating notes

The following rules should be observed when constructing a circuit:

- Block output 2 of the valve, if a single-acting cylinder is actuated by a 5/2-way single solenoid valve in a circuit.

*Plug for output 2  
of a 5/2-way valve*



- Input signals, which would result from an actual production process sequence, are reproduced in part by signals via push buttons or switches.



**Section A – Course**

**Components of a programmable logic controller**

Exercise 1: **Design and commissioning of a programmable logic controller** ..... A-3  
*Components of a PLC*

**Programming to IEC 1131**

Exercise 2: **From problem to solution – taking into consideration IEC 1131-3** ..... A-9  
*Practical steps for PLC programming*

**Basic logic operations**

Exercise 3: **Lamp circuit** ..... A-15  
*The assignment function*

Exercise 4: **Burglar alarm** ..... A-25  
*The NOT function*

Exercise 5: **Press with protective guard** ..... A-35  
*The AND function*

Exercise 6: **Bell system** ..... A-45  
*The OR function*

**Logic control system without latching properties**

Exercise 7: **Stamping device** ..... A-55  
*Combinations of AND/OR/NOT*

Exercise 8: **Silo control system for two bulk materials** ..... A-65  
*Logic control system with branching*

## Logic control systems with latching properties

- Exercise 9: **Fire alarm** ..... A-73  
*Setting an output*
- Exercise 10: **Drill breakage monitoring** ..... A-81  
*Setting and resetting an output*
- Exercise 11: **Activating a cylinder** ..... A-81  
*Signal edges*

## Logic control systems with time response

- Exercise 12: **Bonding of components** ..... A-101  
*Pulse*
- Exercise 13: **Embossing device** ..... A-111  
*Switch-on signal delay*
- Exercise 14: **Clamping device** ..... A-121  
*Switch-off signal delay*

## Sequence control systems

- Exercise 15: **Lifting device for packages** ..... A-131  
*Linear sequence*
- Exercise 16: **Lifting and sorting device for packages** ..... A-147  
*Alternative branching*
- Exercise 17: **Stamping device with counter** ..... A-159  
*Counting cycles*

Programmable logic controllers

Subject

### Design and commissioning of a programmable logic controller

Title

Components of a PLC

- To be able to explain the basic design and mode of operation of a PLC
- To be able to configure and commission a PLC

Training aim

Nowadays, programmable logic controllers form part of any automation process. Fig. A1.1 illustrates the typical configuration of an automation solution realised by means of a PLC. The control system shown represents the simpler, non-networked group of PLC applications.

Technical knowledge

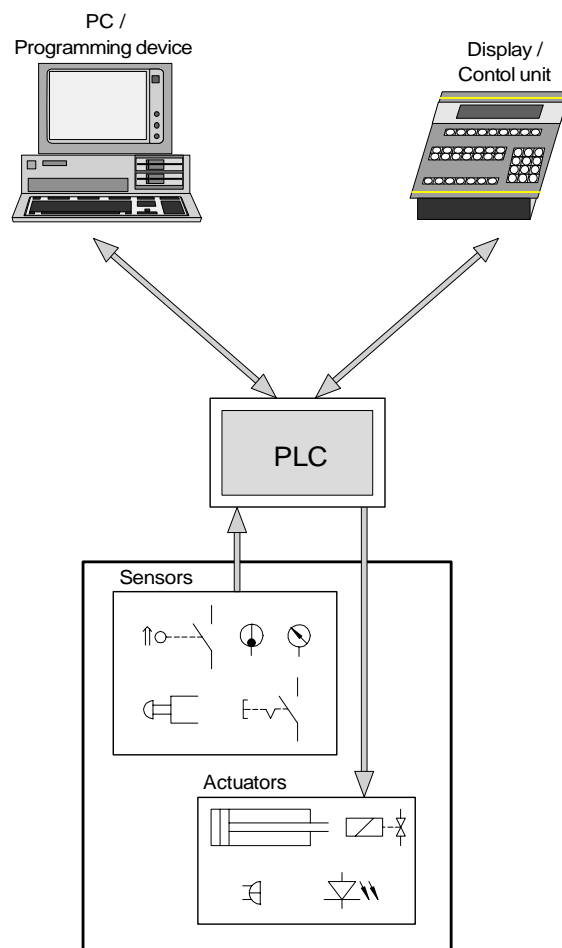


Fig. A1.1:  
Automation via PLC

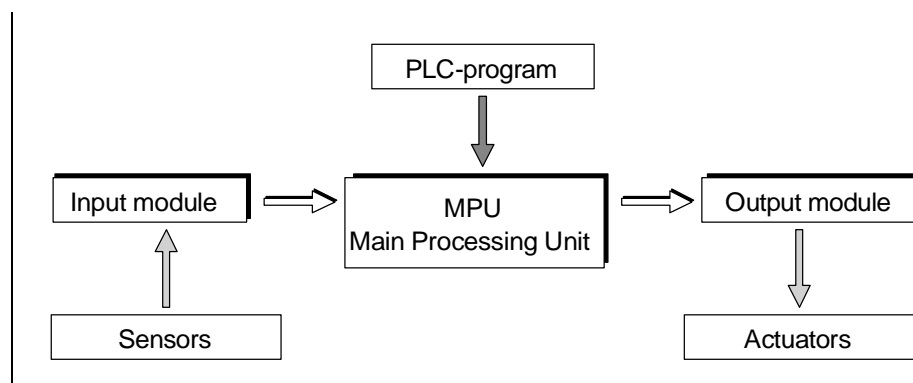
The basic components of the control system are:

- Programmable logic controller (PLC)  
By this, we understand the electronic modules through which all of the system or machine functions to be controlled are addressed and activated in a logic sequence.
- Sensors  
These components are located directly on the system or machinery to be controlled, through which the PLC is communicated actual statuses.
- Actuators  
These components are located directly on the system or machinery to be controlled, through which the PLC is able to change or influence statuses and as such the technical process.
- PC or programming device  
This is used to create the program containing the logic of the system or machinery to be controlled and to transfer this to the memory of the PLC. At the same time, these programming tools also provide supporting functions for the testing of the PLC program and commissioning of the controller.
- Display and control units  
These enables you to monitor and influence the operation of the system or machinery.

### Programmable logic controller

The most important component of a control system is the PLC and its program. Fig. A1.2 illustrates the system components of a PLC.

Fig. A1.2:  
System components  
of a PLC



A PLC is connected to the system to be controlled via input and output modules. The system to be controlled supplies input signals (mostly binary) via sensors to the input modules. These signals are processed within the main processing unit, the main component of the PLC. Prior to formulation of IEC standards, known as "central control unit" (CCU). The "specification" for the processing of signals is defined in the PLC program. The result of the processing is output to the actuators of the system to be controlled via the output module. Thus, the design of a PLC corresponds to that of a computer.



### PLC program

PLC programs consist of a logic sequence of instructions. The control program is stored in a special, electronic readable memory, the so-called program memory of the PLC. Special RAMs with back-up battery are used during the program development, since its contents can always be changed again very quickly.

After commissioning and error-free function of the controller it is a good idea to transfer the PLC program unerasably to a read-only memory, e.g. an EPROM. If the program is executed, it will be processed in continuous cycles.

### Signals

Input signals reach the PLC via sensors. These signals contain information about the status of the system to be controlled. It is possible to input binary, digital and analogue signals.



A PLC can only recognise and output electrical signals. For this reason, non-electrical signals are converted into electrical signals by the sensors. Sensor examples are:

- Push buttons, switches, limit switches, proximity sensors

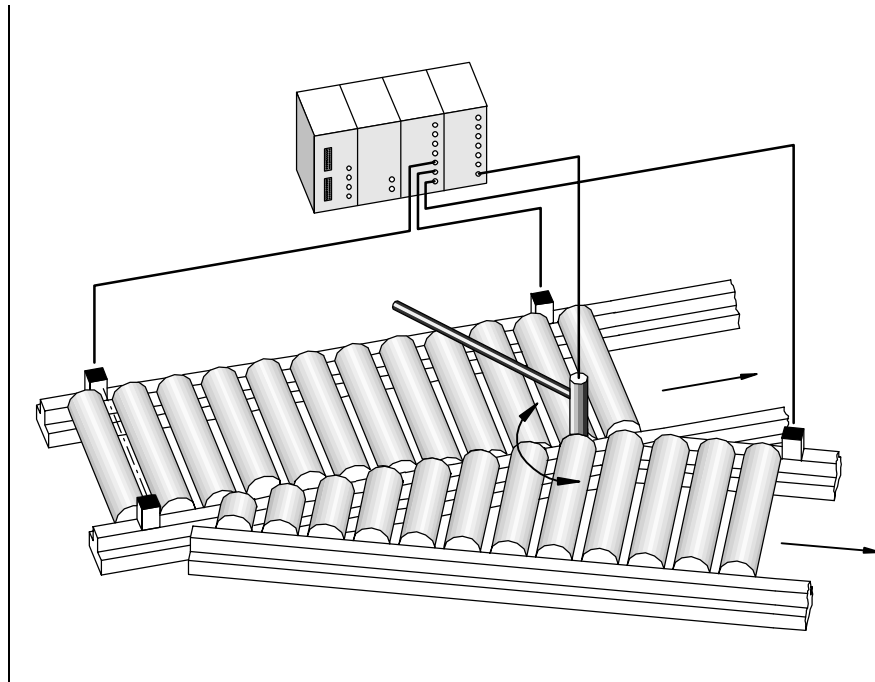
Output signals influence the system to be controlled. The signals can be output in the form of binary, digital or analogue signals. Output signals are amplified into switching signals via the actuators or converted into signals of other energy forms. Actuators examples are:

- Lamps, buzzers, bells, contactors, cylinders with solenoid valves, stepper motors

# A-6

## Exercise 1

*Problem description* A control task is to be solved via a programmable logic controller (PLC). Familiarise yourself with the basic design of a PLC.



*Positional sketch*

- Exercise definition*
1. Components of a PLC
  2. Design and commissioning of the PLC you have selected

*Implementation* To carry out the exercise using the worksheets, refer to Section B of the workbook and your PLC data sheet or manual.

**WORKSHEET**

**1. Components of a PLC**

**Question 1**

What are the basic components of a programmable logic controller?

---

---

---

---



**Question 2**

What are the basic modules making up the central control unit of a programmable logic controller?

---

---

---

---



**Question 3**

How is electrical isolation achieved between sensor/actuator signals and the PLC?

---

---

---

---



## WORKSHEET

### 2. Design and commissioning of the PLC you have selected

Enter the technical data of the selected programmable logic controller in the table below.

<i>Technical data</i>	<b>Operating voltage</b>	
	Nominal voltage	
	Permissible voltage range	
	Current consumption	
	<b>Inputs</b>	
	Number	
	Input current	
	Input level	
	<b>Outputs</b>	
	Number	
	Switching logic	
	Output voltage	
	Output current	

Configure the PLC in accordance with the notes in the relevant data sheet or manual.



Programmable logic controllers

Subject

### From problem to solution – taking into consideration IEC 1131-3

Title

Practical steps for PLC programming

- To familiarise yourself with the basic language resources for the configuration and structuring of a PLC program in accordance with IEC 1131-3
- To be able to declare variables for use in a PLC program
- To be able to apply a systematic procedure for the implementation of PLC exercises

Training aim

### Creating a PLC program

Technical knowledge

The practical steps for creating a PLC program are illustrated in fig. A2.1.

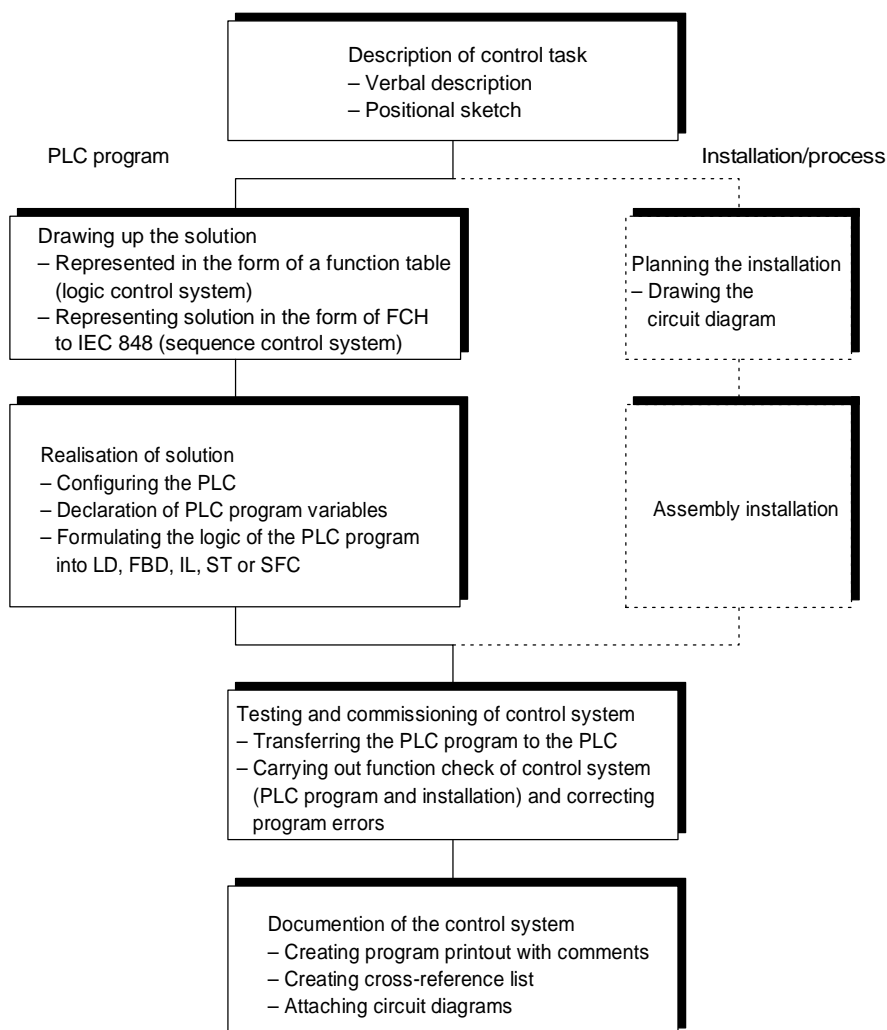
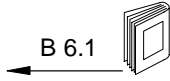


Fig. A2.1:  
Procedure for creating  
a PLC program



### Description of a control task

The basic requirements for describing a control task include a verbal description, a positional sketch and the definition of sensors and actuators to be used. This information is required for the development of circuit diagrams. Moreover, it is necessary to define the allocation of sensors to the PLC inputs or the allocation of actuators to PLC outputs. An example of an allocation list of this type is shown in table A2.1.

Table A2.1:  
Example of an  
allocation list

<i>Resource designation</i>	<i>Input/output address on PLC</i>	<i>Comment</i>
S1	I1.5	Push button START
Y5	Q2.7	Cylinder C to advance

Since this list forms a component part of PLC programs, it is not created separately in the case of smaller control tasks.

### Solution design

The designed solution is to give a clear representation of the function and behaviour of the controller independent of technology. The function table is used as a means of describing simple logic control systems. The function chart to IEC 848 is particularly suitable for the description of sequence control systems.

### Realisation of the solution

The realisation of the solution is divided into

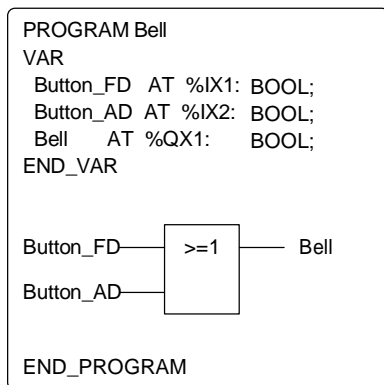
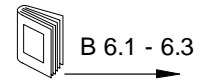
- Programming of the logic of the control system (PLC)
- Incorporating the PLC program in the PLC or PLC system

An example of this is shown in fig. A2.2 using a simple control task:

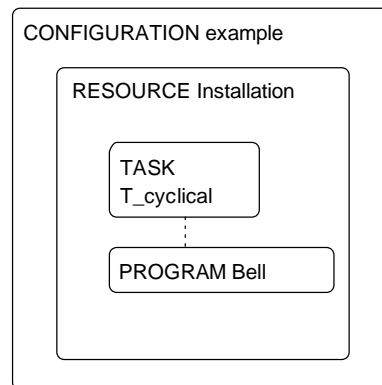
*Example* A bell is to ring either if the bell button on the front door or the bell button on the apartment door is actuated.

When commencing the production of a PLC program, the resources are to be declared in the syntax defined by IEC 1131-3. The PLC inputs and outputs declared in fig. A2.2a are local variables within the program "Bell". The use of global variables is only required for more complex control tasks.

Following the declaration, the program body is formulated. Functions and function blocks are available at this point in order to create a clearly arranged program. The example is programmed in function block diagram using the OR function.



a) Example of a PLC program



b) Example for the installation of a PLC program in a PLC or a PLC system

Fig. A2.2:  
Realisation of a solution

In order to execute a PLC program via the PLC, it is necessary to know how the program is to be processed: cyclically or in relation to certain events. IEC 1131-3 provides its own language resources for the assignment of such sequence characteristics and for incorporating the program in the PLC or the PLC system. These are the configuration language resources.

Fig. A2.2b illustrates the configuration "Example". This configuration represents the PLC. The configuration "Example" requires the resource "Processor\_1". This resource is assigned to the program "Bell". The task "T\_cyclical" defines that the program "Bell" is to be processed cyclically.

### **Testing and commissioning the controller**

The program is loaded from the PC or programming device to the PLC for the testing or commissioning of the control system. Following this, the interaction of PLC and system must be checked.

### **Control system documentation**

The system documentation is compiled as soon as the installation operates free of fault and the PLC program has been corrected accordingly. The documentation basically consists of:

- the positional sketch,
- the formal solution design and
- the program printout with comments.

*Problem description* PLC inputs and outputs and additional variables for storing information are to be incorporated in a PLC program. For this, you will need to familiarise yourself with the basic procedure required for PLC program generation.

*Exercise definition*

1. Procedures for creating a PLC program
2. Resources of a PLC according to IEC 1131-3
3. Declaration of variables according to IEC 1131-3

*Implementation* In order to carry out the exercise you will need the information from Section B of the workbook: Chapter 6, page B-65.

## WORKSHEET

### 1. Practical steps for creating a PLC program

Specify the five practical steps for creating a PLC program.

---

---

---

---

---

Answer the following questions:

*Question*

1. What activities are carried out in the step "Implementation of the solution"

---

---

---

---

### 2. Resources of a PLC in accordance with IEC 1131-3

The following resources are to be addressed directly.  
Specify the designations in accordance with IEC 1131-3:

Input bit 14	_____
Memory 9	_____
Output word 3	_____
Input 7 on 2nd input card	_____

## WORKSHEET

### 3. Declaration of variables to IEC 1131-3

The following data must be taken into consideration in a program declaration. Use the appropriate data type in your declaration. The declaration is to be valid locally only.

- Input for a switch S1, applied to input 2 of the 4th input card
- Temperature TEMP, applied to output word No. 1
- Memory VALVE\_OPEN
- boolean memory with identifier PART\_PRESENT, preallocated initial value 0
- boolean memory with identifier ROBOT\_INIT, preallocated with initial value 1
- storage of a number (INT) under the name NUMBER, preassigned the value 0

---

---

---

---

---

---

---

---

---

---

Programmable logic controllers

*Subject*

### Lamp circuit

*Title*

The assignment function

*Training aim*

- To understand the actuation of a PLC output
- To be able to realise the logic assignment function with a PLC
- To be able to create a PLC program in accordance with IEC 1131-3

Each programmable logic controller has a certain number of inputs and outputs, through which it is connected with the sensors and actuators. The program transferred to the controller contains the commands which interconnect the individual inputs and assigns these to the corresponding outputs.

*Technical knowledge*

### The assignment function

The assignment function permits a PLC input signal to be directly transmitted to a PLC output. The behaviour can be clearly described with the help of a function table, which represents this for an input %IX1 and an output %QX2 in table A3.1.



%IX1	%QX2
0	0
1	1

Table A3.1:  
Function table for the  
assignment function

# A-16

## Exercise 3

In order to realise the assignment function in the individual programming languages, you will need the commands shown in table A3.2.







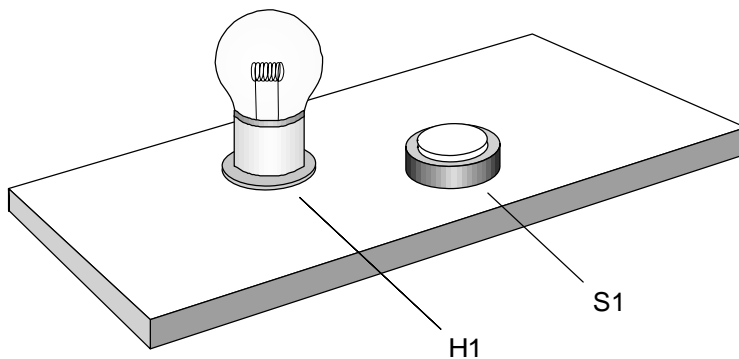
<p>B 8</p> 	<p>LD</p>
	 <p>Current rung with normally open contact and non-stored coil.</p>
<p>B 7</p> 	<p>FBD</p>
	 <p>Direct connection of specified input and output.</p>
<p>B 9</p> 	<p>IL</p>
	<p>LD %IX1</p> <p>Reading of value of specified input to accumulator.</p> <p>ST %QX2</p> <p>Storage of contents of accumulator to specified output.</p>
<p>B 10</p> 	<p>ST</p>
	<p>%QX2 := %IX1;</p> <p>Assignment of value of specified input to the right of " := " to the specified output on the left of " := ".</p>

Table A3.2:  
The assignment function



Actuation of a push button (S1) is to cause a lamp (H1) to be switched on. The lamp is to be illuminated as long as the push button is actuated.

*Problem description*



*Positional sketch*

1. Drawing up the circuit diagram and assembling the equipment
2. Describing the control task by means of the function table and the boolean equation
3. Declaration of PLC program variables
4. Formulation of the PLC program in the various programming languages
5. Testing and commissioning of the PLC program and system

*Exercise definition*

# A-18

## Exercise 3

### Implementation 1. Drawing up the circuit diagram and assembling the equipment

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

Components list

Quantity	Description
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Signal input, electrical
1	Signalling device



Prior to wiring:

- Switch off power supply!

⇒ Establish the electrical connections.

### 2. Describe the control task by means of the function table and the boolean equation

⇒ Describe the behaviour of the control system irrespective of technology by means of the function table and the associated boolean equation.

### 3. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

#### 4. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the programming languages supported by your PLC system. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

#### 5. Testing and commissioning of PLC program and system

**Prior to commissioning of the installation:**

- Check the assembled circuit with the help of the circuit diagrams!

**Commissioning of the installation:**

- Switch on power supply using a standard voltage of 24 V DC!



⇒ Load the program to the PLC.

⇒ Carry out a function check.

⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

# A-20

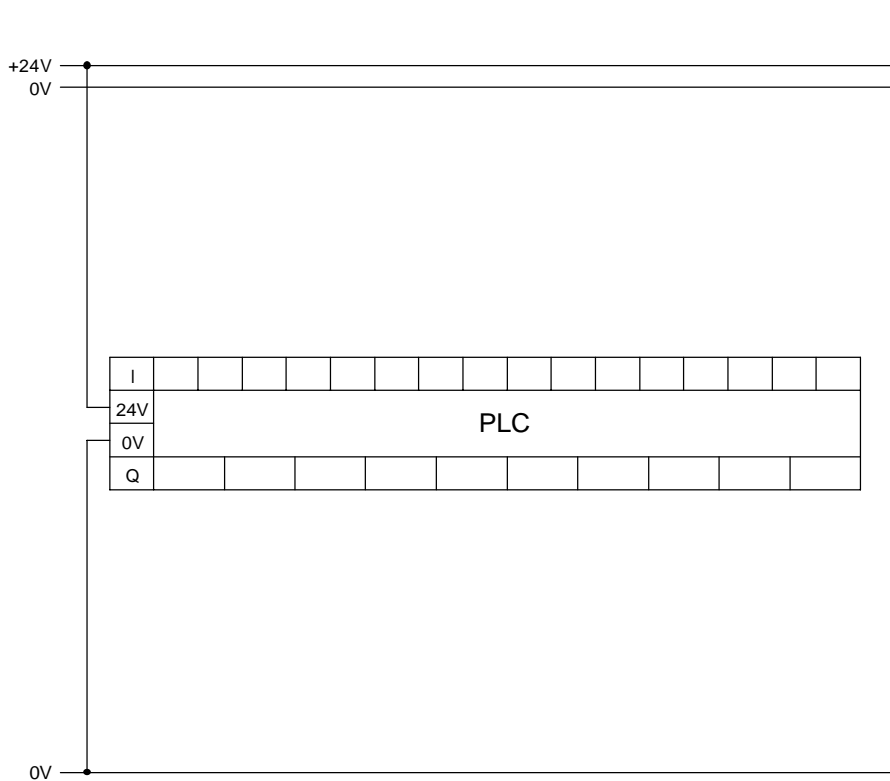
---

## Exercise 3

## WORKSHEET

### 1. Drawing up the circuit diagram and assembling the equipment

Complete the electrical circuit diagram and enter the available input and output addresses of your PLC.



Circuit diagram, electrical

### 2. Describing the control task by means of the function table and the boolean equation

Create the function table:

<b>S1</b>	<b>H1</b>
0	
1	

Function table

Derive the boolean equation from this:

\_\_\_\_\_

Boolean equation

# A-22

## Exercise 3

---

### WORKSHEET

#### 3. Declaration of PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

#### 4. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task in one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)


*PLC program*

**WORKSHEET**

Answer the following questions:

*Question*

1. What is the behaviour of a non-stored programmed output, if the input signal is no longer applied?

---

---

---

---

2. Describe the basic design of an output module.

---

---

---

---

# A-24

---

## Exercise 3



Programmable logic controllers

*Subject*

### **Burglar alarm**

The NOT function

*Title*

- To be able to realise the logic NOT function with a PLC

*Training aim*

### **The NOT function**

The NOT function is used to convert binary signals into opposing signals:

*Technical knowledge*

- If the signal is 0, it is evaluated as 1
- If the signal is 1, it is evaluated as 0



Table A4.1 illustrates an example of the behaviour of the NOT function for an input %IX1 and an output %QX2.

<i>%IX1</i>	<i>%QX2</i>
0	1
1	0

*Table A4.1:  
Function table for  
NOT function*

# A-26

## Exercise 4

Table A4.2 contains the commands for the implementation of the NOT function in the individual programming languages.

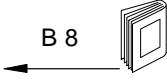

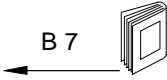
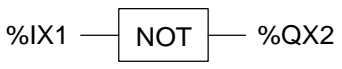
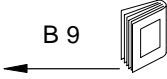
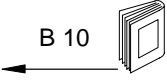
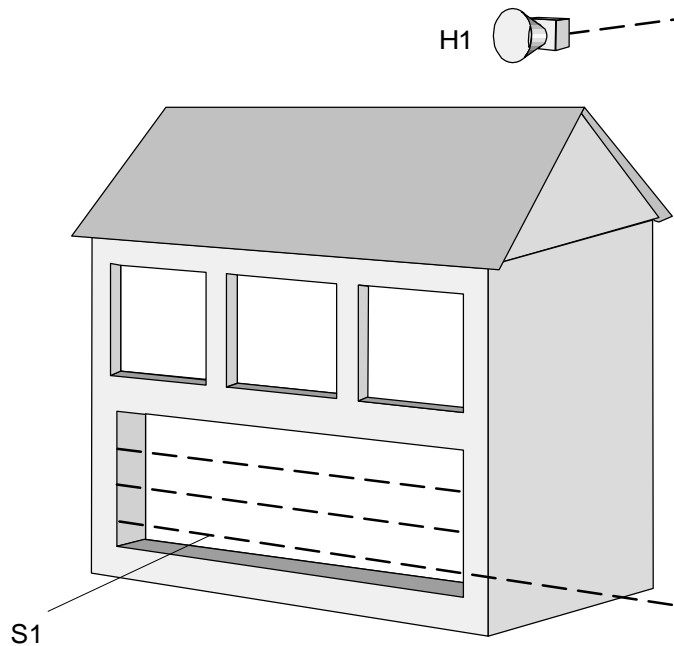
B 8 	<b>LD</b>		Current rung with normally closed contact and non-stored coil.
B 7 	<b>FBD</b>		Use NOT function at specified input.
B 9 	<b>IL</b>	LDN    %IX1  ST     %QX2	Reading of negated value of specified input to accumulator.  Storage of contents of accumulator to specified output.
B 10 	<b>ST</b>	<code>%QX2 := NOT %IX1;</code>	Assignment of negated value of specified input to the output on the left of " := ".

Table A4.2:  
The NOT function

A thin wire has been stretched behind a display window, which breaks if a burglary is attempted. A closed circuit is interrupted as a result of this and a buzzer is sounded.

*Problem description*



*Positional sketch*

1. Drawing up the circuit diagram and assembling the equipment
2. Describing the control task by means of the function table and the boolean equation
3. Declaration of the PLC program variables
4. Formulation of the PLC program into one of the PLC programming languages
5. Testing and commissioning of the PLC program and system

*Exercise definition*

### Implementation 1. Drawing up the circuit diagram and assembling the equipment

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted profile plate:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Signal input, electrical
1	Signalling device

*Components list*



Prior to wiring:

- Switch off power supply!

⇒ Establish the electrical connections.

### 2. Describing the control task by means of the function table and the boolean equation

⇒ Describe the behaviour of the control system irrespective of technology by means of a function table and the associated boolean equation.

### 3. Declaration of the PLC program variables

⇒ All variables must be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### **Note**

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

#### 4. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the programming languages supported by your PLC system. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

#### 5. Testing and commissioning of the PLC program and system

**Prior to commissioning of the installation:**

- Check assembled circuit with the help of the circuit diagrams!

**Commissioning of the installation:**

- Switch on power supply using a standard voltage of 24 V DC!



⇒ Load the program to the PLC.

⇒ Carry out a function check.

⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

# A-30

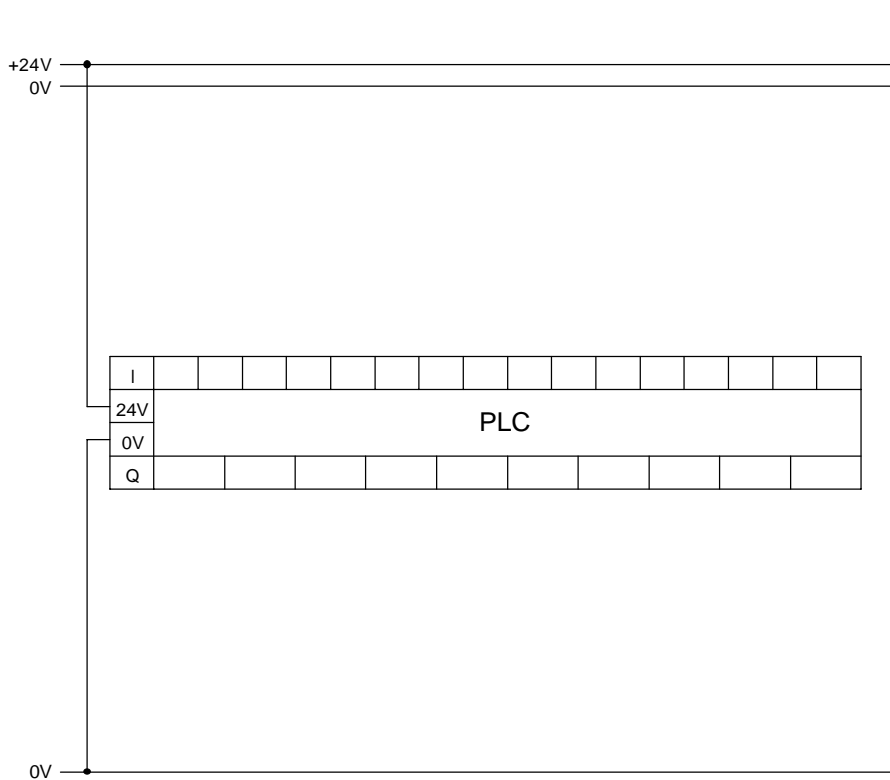
---

## Exercise 4

## WORKSHEET

### 1. Drawing up the circuit diagram and assembling the equipment

Complete the electrical circuit diagram and enter the available input and output addresses for your PLC.



Circuit diagram, electrical

### 2. Describing the control task by means of the function table and the boolean equation

Create the function table:

S1	H1
0	
1	

Function table

Derive the boolean equation from this:

\_\_\_\_\_

Boolean equation

## WORKSHEET

### 3. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

### 4. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task in one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)


*PLC program*



**WORKSHEET**

Answer the following question:

*Question*

1. The output is set non-stored. What is the effect on the output signal, if the wire is broken and has to be repaired?

---

---

---

---

# A-34

---

## Exercise 4

Programmable logic controllers

*Subject*

### Press with protective guard

*Title*

The AND function

- To be able to realise a logic AND function with a PLC
- To understand the term 'function' according to IEC 1131-3
- To be able to use standard functions to IEC 1131-3

*Training aim*

Functions are part of the program organisation units and therefore represent a means for configuring PLC programs. IEC 1131-3 provides standardised functions for the solution of basic control technology tasks.

*Technical knowledge*

### The AND function

Only when all AND connected signals are 1, is the result 1. If one of the connected signals is 0, then the result is also 0.

The function table for the AND function is illustrated below for the two inputs %IX1 and %IX2 as well as %QX3.



%IX1	%IX2	%QX3
0	0	0
0	1	0
1	0	0
1	1	1

*Table A5.1:  
Function table for the  
AND function*

# A-36

## Exercise 5

Table A5.2 contains the commands for the realisation of the AND function in the individual programming languages.



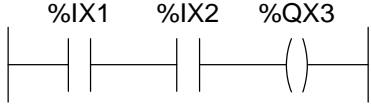


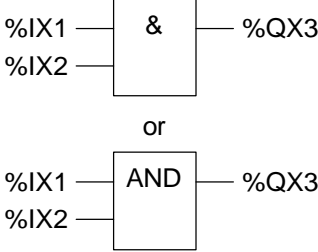




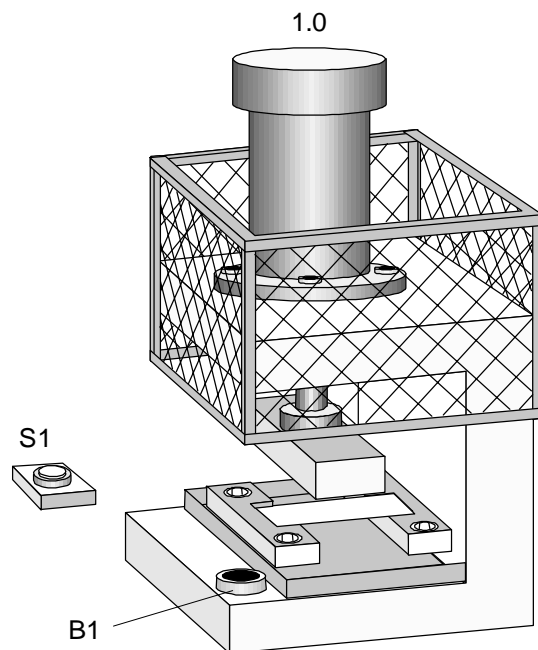
B 8  	<b>LD</b>		Rung with consecutively switching normally open contacts
B 7  	<b>FBD</b>		The inputs and the output of the AND function are connected with the specified current parameters.
B 9  	<b>IL</b>	<pre>LD    %IX1 AND   %IX2 ST    %QX3</pre>	Reading the value of the specified input to the accumulator. AND connection of current result with second input. Storage of contents of accumulator to specified output.
B 10  	<b>ST</b>	<pre>%QX3 := %IX1 &amp; %IX2;</pre> <p style="text-align: center;">or</p> <pre>%QX3 := AND(%IX1, %IX2);</pre>	The two specified inputs are connected with the "&" operator. The result is assigned to the specified output.  Invocation of AND function with specified inputs as current transfer parameters.

Table A5.2:  
The AND function

A press stamp 1.0 is to advance only if a push button S1 is actuated **and** a protective guard is closed. If one of these conditions is not met, the press tool is to return immediately.

The position of the closed protective guard B1 is monitored by a proximity switch B1. The press tool is advanced or retracted by means of a spring return solenoid valve (coil Y1).

*Problem description*



*Positional sketch*

1. Drawing up the electro-pneumatic and the electrical circuit diagram and assembling the equipment
2. Describing the control task by means of the function table and the boolean equation
3. Declaration of the PLC program variables
4. Formulation of the PLC program into one of the PLC programming languages
5. Testing and commissioning of the PLC program and system

*Exercise definition*

*Implementation* **1. Drawing the electro-pneumatic and the electrical circuit diagram and assembling the equipment**

⇒ Complete the electro-pneumatic and the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted profile plate:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Double-acting cylinder
	5/2-way single solenoid valve
1	Signal input, electrical
1	Proximity switch, inductive
	Plastic tubing

*Components list*



Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!

⇒ Establish the electrical and pneumatic connections.

**2. Describing the control task by means of the function table and the boolean equation**

⇒ Describe the behaviour of the control system irrespective of technology by means of a function table and the associated boolean equation.

### 3. Declaration of the PLC program variables

- ⇒ All variables are to be created as program-local variables
- ⇒ Specify only those parts of the declaration, which are required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC program system used.

### 4. Formulation of the PLC program into one of the PLC programming languages

- ⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 5. Testing and commissioning of PLC program and system

#### Prior to commissioning of the installation:

- Check the assembled circuit with the help of the circuit diagrams!

#### Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply at the service unit to operating pressure (see data sheets for pneumatic components)!

#### Operation of the installation:

- **Keep clear** of the operational parts of the installation!



- ⇒ Load the program to the PLC.
- ⇒ Carry out a function check.
- ⇒ Correct any errors occurring in the PLC program.
- ⇒ Document your solution.

# A-40

---

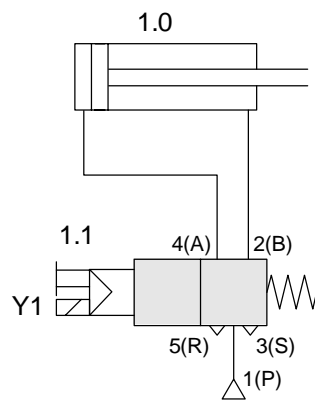
## Exercise 5



## WORKSHEET

### 1. Drawing up the electro-pneumatic and the electrical circuit diagram and assembling the equipment

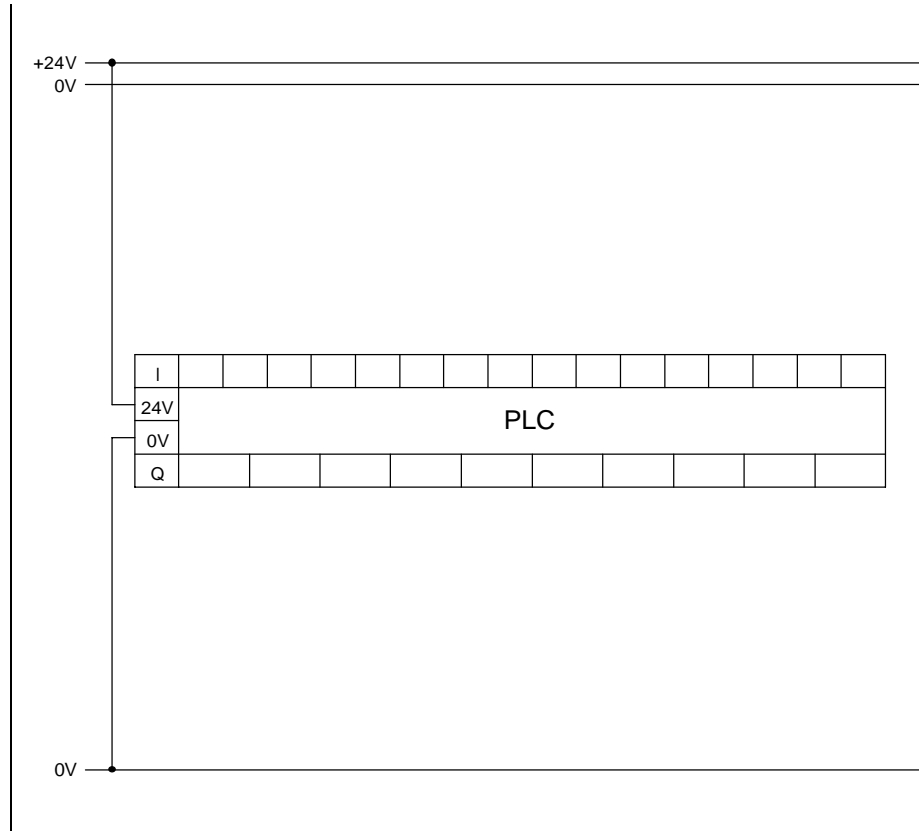
Complete the electro-pneumatic circuit diagram



Circuit diagram,  
electro-pneumatic

## WORKSHEET

Complete the electrical circuit diagram and enter the available input and output addresses for your PLC.



Circuit diagram, electrical

### 2. Describing the control task by means of the function table and the boolean equation

Create the function table:

	<b>S1</b>	<b>B1</b>	<b>Y1</b>

Function table

Derive the boolean equation from this:

Boolean equation

\_\_\_\_\_

**WORKSHEET****3. Declaration of the PLC program variables**

Declare the variables required in the PLC program:

<b>Designation</b>	<b>Data type</b>	<b>Address</b>	<b>Comment</b>

*Declaration of variables***4. Formulation of the PLC program into one of the PLC programming languages**

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*PLC program*

## **WORKSHEET**

*Question* Answer the following question:

1. Does output Y1 have to be set stored or non-stored?

---

---

---

---

Programmable logic controllers

*Subject*

### **Bell system**

The OR function

*Title*

- Realising the logic OR function with a PLC.

*Training aim*

### **The OR function**

If at least one of the connected signals is 1, the result is also 1. Only if all the connected signals are 0 is the result also 0.

*Technical knowledge*

Table A6.1 contains the function table for the OR connection of the signal from input %IX1 and input %IX2. The result is mapped to output %QX3.



<b>%IX1</b>	<b>%IX2</b>	<b>%QX3</b>
0	0	0
0	1	1
1	0	1
1	1	1

*Table A6.1:  
Function table for the  
OR function*

# A-46

## Exercise 6

Table A6.2 contains the commands for the realisation of the OR function in the individual programming languages.


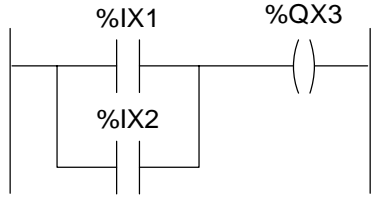

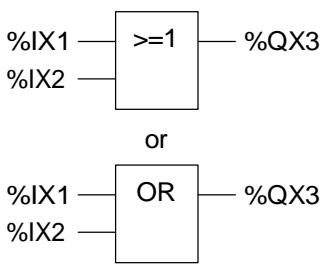


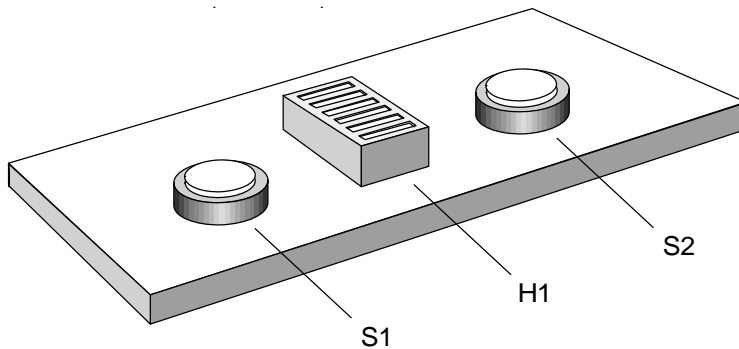
B 8  ←	<b>LD</b>		Rung with parallel switching normally open contacts.
B 7  ←	<b>FBD</b>		The inputs and the output of the OR function are connected with the current parameters specified.
B 9  ←	<b>IL</b>	<pre>LD   %IX1 OR   %IX2 ST   %QX3</pre>	Reading the value of the specified input to the accumulator. OR connection of current result with second input. Storage of contents of accumulator to specified output.
B 10  ←	<b>ST</b>	<pre>%QX3 := %IX1 OR %IX2;</pre>	The two specified outputs are connected with the operator "OR". The result is assigned to the specified output.

Table A6.2:  
The OR function

An apartment bell is to ring if bell button S1 at the front door is pressed or bell button S2 at the apartment door.

*Problem description*



*Positional sketch*

1. Drawing up the circuit diagram and assembling the equipment
2. Describing the control task by means of the function table and the boolean equation
3. Declaration of the PLC program variables
4. Formulation of the PLC program into one of the PLC programming languages
5. Testing and commissioning of PLC program and system

*Exercise definition*

### Implementation 1. Drawing up the circuit diagram and assembling the equipment

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted profile plate:

Quantity	Description
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Signal input, electrical
1	Signalling device

Components list



Prior to wiring the installation:

- Switch off power supply!

⇒ Establish the electrical connections.

### 2. Describing the control task by means of the function table and the boolean equation

⇒ Describe the behaviour of the control system irrespective of technology by means of a function table and the associated boolean equation.

### 3. Declaration of the PLC program variables

⇒ All variables must be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.



#### 4. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

#### 5. Testing and commissioning of PLC program and system

**Prior to commissioning of the installation:**

- Check the assembled circuit with the help of the circuit diagrams!

**Commissioning of the installation:**

- Switch on power supply using a standard voltage of 24 V DC!



⇒ Load the program to the PLC.

⇒ Carry out a function check.

⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

# A-50

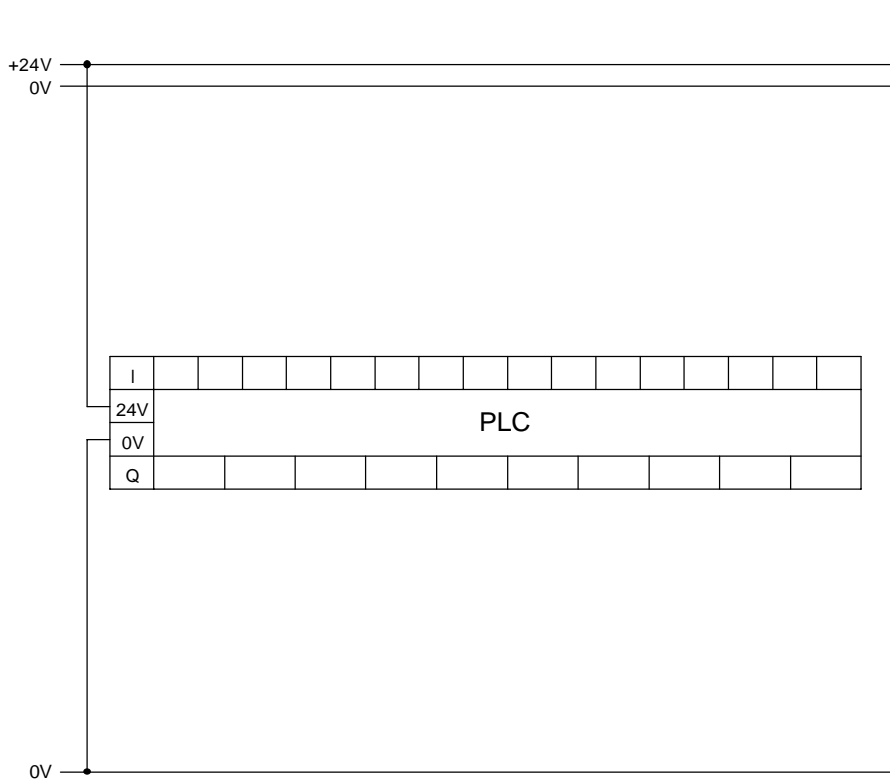
---

## Exercise 6

## WORKSHEET

### 1. Drawing up the circuit diagram and assembling the equipment

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



Circuit diagram, electrical

### 2. Describing the control task by means of the function table and the boolean equation

Create the function table:

S1	S2	H1

Function table

### WORKSHEET

Derive the boolean equation from this:

Boolean equation

---

#### 3. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

Declaration of variables

#### 4. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task in one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

PLC program

## WORKSHEET

Answer the following question:

1. In the case of the OR function, the output is set if at least one input signal is set. With the exclusive OR function, the output is set only if exactly one of the connected inputs is set. Complete the function table.

Question

<i>S1</i>	<i>S2</i>	<i>H1</i>
0	0	
0	1	
1	0	
1	1	

Function table

# A-54

---

## Exercise 6

Programmable logic controllers

*Subject*

### Stamping device

*Title*

Combinations of AND/OR/NOT

- To be able to realise combinations of logic connections with a PLC
- To understand the priorities of elementary operators in the individual programming languages

*Training aim*

### Combination of logic connections

*Technical knowledge*

Many control tasks require the programming of a combination of logic connections. The following are essential for drawing up a solution:

- Establishing a boolean equation which describes the logic of the control task
- Taking into consideration the priorities of the operators used for programming

The example below deals with a combination of AND, OR and NOT functions:



A lamp H1 is to illuminate if switch S1 and, in addition exactly one of the switches S2 or S3 is actuated.

The relevant function table is as follows:

S1	S2	S3	H1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

*Table A7.1:  
Function table*

The disjunctive normal form (DNF) can be derived from this table. The DNF describes the control task by means of a boolean equation, which can be easily converted into one of the programming languages.

# A-56

## Exercise 7

The solution method for the DNF is based on the lines in the function table, for which the result variable (H1) has the value 1. These lines are OR connected and lead to the boolean equation

$$H1 = (S1 \overline{S2} S3) \vee (S1 S2 \overline{S3})$$

The realisation of the control task in the individual programming languages is listed in table A7.2.

B 8	<b>LD</b> 
B 7	<b>FBD</b> 
B 9	<b>IL</b> <pre> LD      S1 ANDN   S2 AND    S3 OR(    S1 AND    S2 ANDN   S3 ) ST     H1         </pre>
B 10	<b>ST</b> <pre> H1 := S1 &amp; NOT S2 &amp; S3 OR S1 &amp; S2 &amp; NOT S3; or H1 := (S1 &amp; NOT S2 &amp; S3) OR (S1 &amp; S2 &amp; NOT S3);         </pre>

Table A7.2:  
Combination of  
logic operations



Since the processing of the OR command is a subsequent action in the statement list, an opening parenthesis follows the OR command. The closing parenthesis in the penultimate line causes the result of the parenthesized expression to be OR connected with the current result (in the accumulator).

The solution in Structured Text does not require any parenthesizing since the operators already have priorities. In order to obtain better readability it is however a good idea to use parentheses for more complex expressions.

### **Priorities with basic logic connections**

In the graphic programming languages **LD** and **FBD** the order of processing is implicitly specified by the graphics of the program or program part.

In this way, a series connection is evaluated first in a current rung before a simultaneously existing connection is "calculated".

In the function block diagram, the order of evaluation of a network is defined by blocks.

The instructions of an **IL** are processed line by line and the operators therefore all have the same priority. If the evaluation of an operator is to be a subsequent action, this must also be done by means of parenthesis.

The language **ST** defines a unique rule of precedence for the operators. The boolean AND has a higher priority than the boolean OR. With mathematical operations, the rule multiplication before addition or subtraction applies.

*Problem description* A stamping device can be operated from three sides. A workpiece is inserted via a guide, whereby it touches two of the three proximity switches B1, B2 and B3. This causes a pneumatic cylinder 1.0 to extend via a solenoid valve (coil Y1), whereby a recess is to be stamped into the workpiece. The stamping cycle is to be triggered only if two signal generators are addressed. For reasons of safety the cylinder must be prevented from advancing, if all three proximity sensors are contacted.

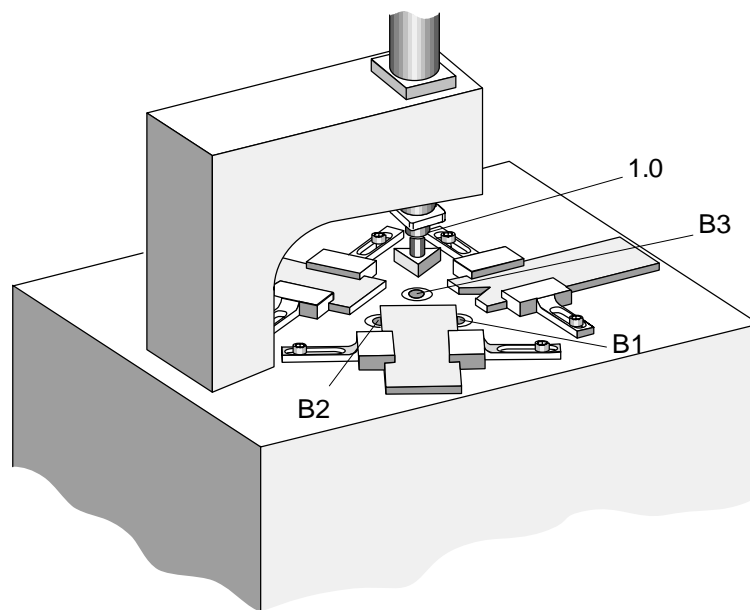


Fig. A7.1:  
Positional sketch

- Exercise definition*
1. Drawing up the electrical circuit diagram
  2. Assembling the equipment with the help of the electro-pneumatic and the electrical circuit diagram
  3. Describing the control task by means of the function table and the boolean equation
  4. Declaration of the PLC program variables
  5. Formulation of the PLC program into one of the PLC programming languages
  6. Testing and commissioning of the PLC program and system

### 1. Drawing up the electrical circuit diagram

⇒ Complete the electrical circuit diagram on the worksheet.

*Implementation*

### 2. Assembling the equipment with the help of the electro-pneumatic and the electrical circuit diagram

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Double-acting cylinder
1	5/2-way single solenoid valve
1	Proximity switch, inductive
1	Proximity switch, capacitive
1	Proximity switch, optical
	Plastic tubing

*Components list*

Prior to wiring and tubing of the installation:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 3. Describing the control task by means of the function table and the boolean equation

⇒ Describe the behaviour of the control system irrespective of technology by means of a function table and the associated boolean equation.

#### 4. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

#### 5. Formulation of PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

#### 6. Testing and commissioning of PLC program and system



**Prior to** commissioning of the installation:

- Check assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheets of pneumatic components)!

Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

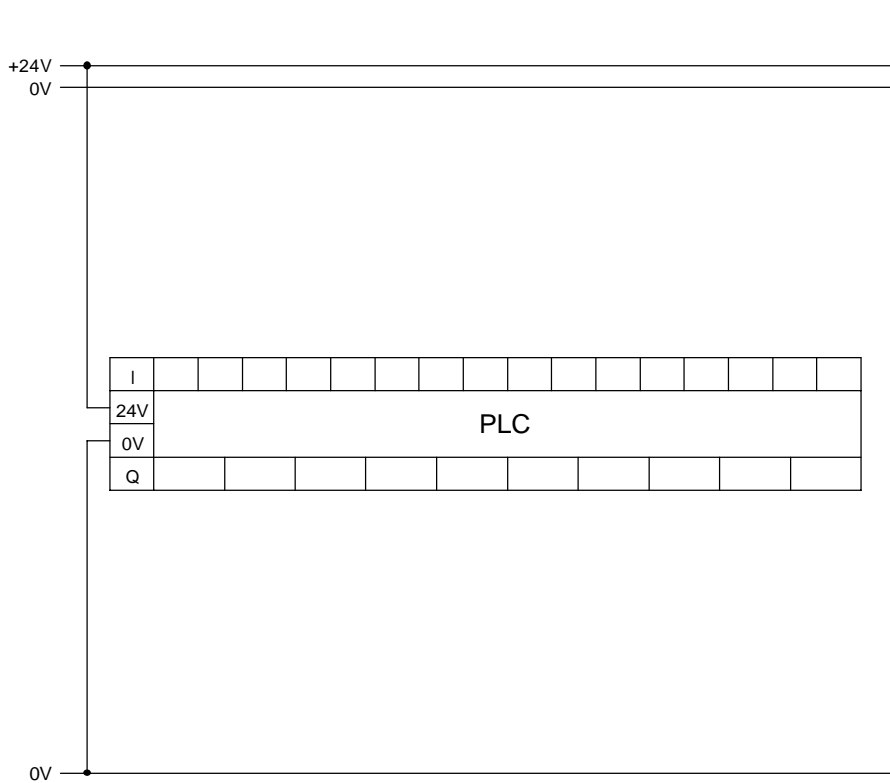
⇒ Correct any errors occurring in the PLC program.

⇒ Document the solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.

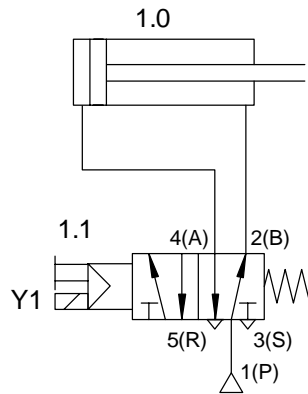


Circuit diagram, electrical

## WORKSHEET

### 2. Assembling the equipment with the help of the electro-pneumatic and the electrical circuit diagram

Configure the control system



Circuit diagram, electro-pneumatic

### 3. Describing the control task by means of the function table and the boolean equation

Complete the function table:

<i>B1</i>	<i>B2</i>	<i>B3</i>	<i>Y1</i>
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Function table

State the associated boolean equation.

Boolean equation

\_\_\_\_\_

**WORKSHEET**

**4. Declaration of variables of the PLC program**

Declare the variables required in your PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

**5. Formulation of the PLC program into one of the PLC programming languages**

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)


*PLC program*

## **WORKSHEET**

*Question* Answer the following question:

1. Why does the negated element have to occur in each parenthesis?

---

---

---

---



Programmable logic controllers

*Subject*

### **Silo control system for two bulk materials**

*Title*

Logic control system with branching

- To be able to solve a logic control system with branching

*Training aim*

More than one final control element is addressed even in the case of simple control tasks.

*Technical knowledge*

This requires the PLC to actuate not just one but several outputs. In the graphic languages this leads to the programming of several current rungs or networks. These networks or current rungs may be optionally provided with a network identifier. A network identifier is required if a jump is to be executed to this network within the program.

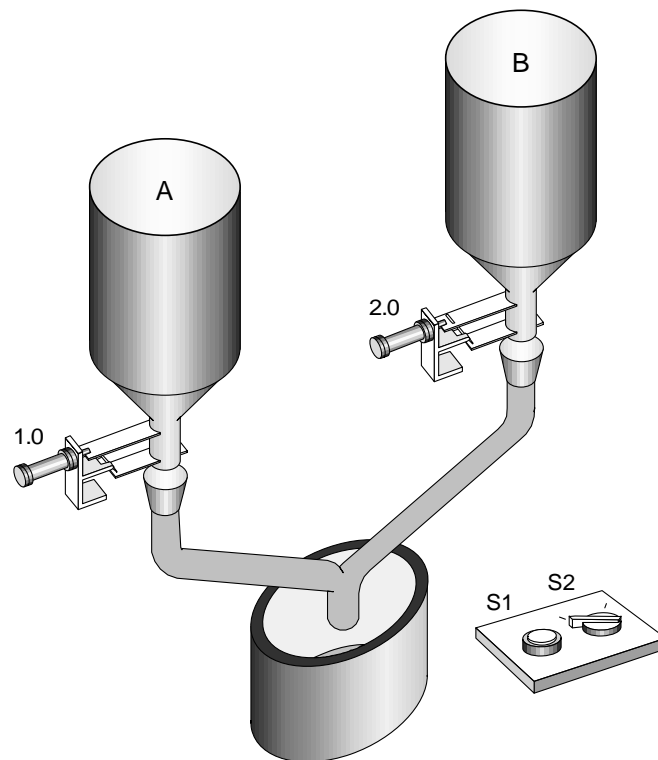


# A-66

## Exercise 8

*Problem description* A mixing plant permits a selection between two bulk materials per selector switch (S2). In switch position 1 (S2 = 0 signal), bulk material A reaches a mixing container, if push button S1 is actuated simultaneously.

Similarly, bulk material is conveyed, if selector switch S2 is in position 2 (S2 = 1 signal) and push button S1 is actuated. Silo A is opened via cylinder 1.0 (solenoid valve Y1), Silo B via cylinder 2.0 (solenoid valve Y2).



*Positional sketch*

- Exercise definition*
1. Drawing up the circuit diagram and assembling the equipment
  2. Describing the control task by means of the function table and the boolean equation
  3. Declaration of the PLC program variables
  4. Formulation of the PLC program into one of the PLC programming languages
  5. Testing and commissioning of PLC program and system

### 1. Drawing up the circuit diagram and assembling the equipment

*Implementation*

⇒ Complete the electrical circuit diagram on the worksheet.

### 2. Assembling the equipment with the help of the electro-pneumatic and the electrical circuit diagram

⇒ Assemble the required components on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
2	Double-acting cylinder
2	5/2-way single solenoid valve
1	Signal input, electrical
	Plastic tubing

*Components list*

Prior to wiring and tubing of the installation:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 3. Describing the control task by means of the function table and the boolean equation

⇒ Describe the behaviour of the controller irrespective of technology by means of a function table and the associated boolean equation.

#### 4. Declaration of the PLC program variables

⇒ All variables must be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

#### 5. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

#### 6. Testing and commissioning of the PLC program and system



##### **Prior to commissioning of the installation:**

- Check assembled circuit with the help of the circuit diagrams!

##### Commissioning of the installation:

- Switch on power supply using standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheets for pneumatic components)!

##### Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

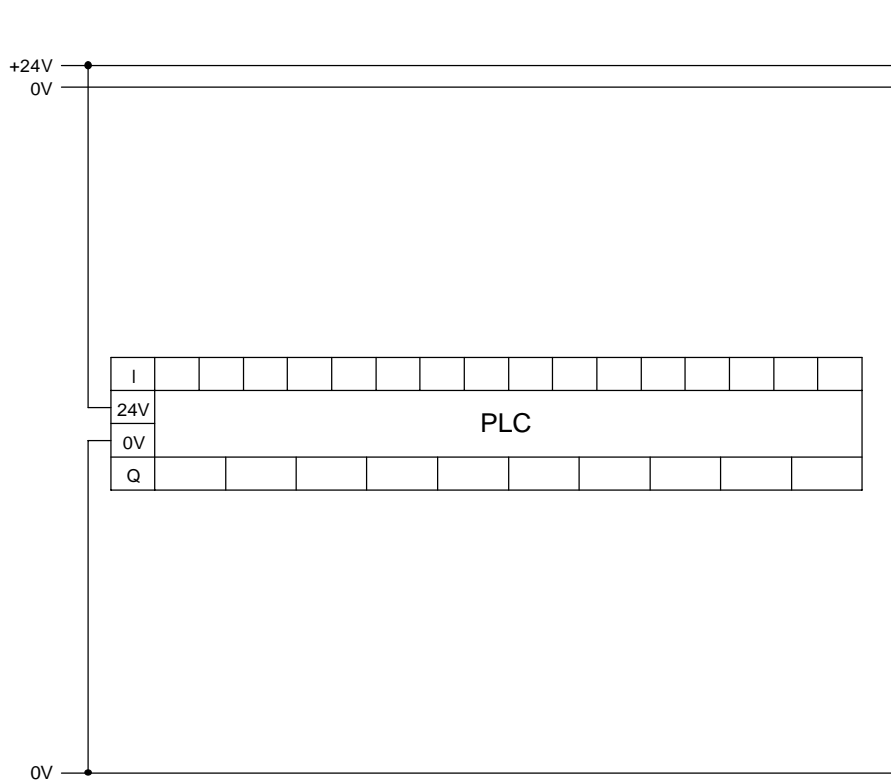
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



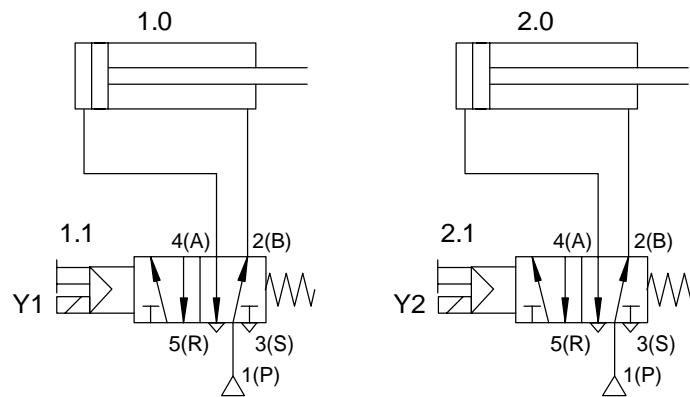
Circuit diagram, electrical

## WORKSHEET

### 2. Assembling the equipment with the help of the electro-pneumatic and the electrical circuit diagram

Configure the control system

Circuit diagram,  
electro-pneumatic



### 3. Describing the control task by means of the function table and the boolean equation

Complete the function table:

Function table

S1	S2	Y1	Y2
0	0		
0	1		
1	0		
1	1		

State the associated boolean equation.

Boolean equation

\_\_\_\_\_

## WORKSHEET

### 4. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

### 5. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*PLC program*

# A-72

---

Exercise 8



Programmable logic controllers

*Subject*

### Fire alarm

Setting an output

*Title*

- To be able to set and store an output of a PLC
- To be able to understand function blocks to IEC 1131-3
- To be able to use the standard function blocks SR flip-flop and RS flip-flop

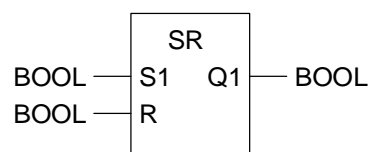
*Training aim*

Storage operations form part of the elementary PLC operations and apply in cases where a briefly occurring signal status is permanently stored. Typical examples of storage operations are the permanent setting or resetting of output signals. The standard function blocks SR and RS are available for the realisation of storage functions.

*Technical knowledge*

### Function block SR, dominant setting flip-flop

Function block SR (fig. A9.1) contains a dominant setting flip-flop.



*Fig. A9.1:  
Function block SR*

The typical behaviour of the SR function block is therefore as follows:

- A 1-signal at set input S1 sets the flip-flop, i.e. the value of Q1 becomes 1.
- A 1 signal at reset input R sets the value of Q1 at 0 only if a 0-signal simultaneously applies at the S1 input.
- If a 1-signal applies both at the S1 and the R input output Q1 is set.

### Realisation of storage function "Set" in the individual programming languages

The solutions for the following example are listed in table A9.1.

# A-74

## Exercise 9

*Example* Lamp H2 is to be illuminated via actuation of a push button S2.



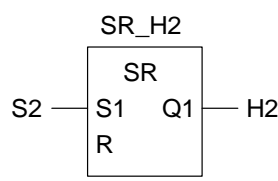



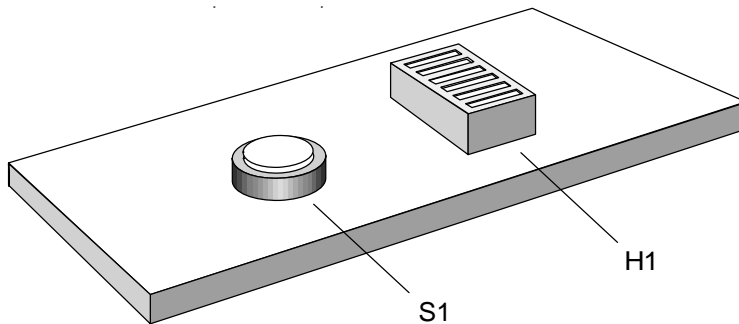
<p>B 7</p> 	<pre> VAR   S2 AT %IX3 : BOOL;      (* Push button S2 at input IX3 *)   H2 AT %QX4 : BOOL;      (* Lamp H2 at output QX4 *)   SR_H2      : SR;        (* SR-flip-flop named SR_H2 *)                                 (* For storage of status *)                                 (* of H2 *) END_VAR </pre>
<p>B 8</p> 	<p>FBD</p> <div style="display: flex; align-items: center;"> <div style="margin-right: 20px;">  </div> <div> <p>SR flip-flop SR_H2 for storage of status of variable H2. Direct assignment of value of Q1 at variable H2.</p> </div> </div>
<p>B 9</p> 	<p>LD</p> <div style="display: flex; align-items: center;"> <div style="margin-right: 20px;">  </div> <div> <p>Rung with normally open contact and set coil.</p> </div> </div>
<p>B 10</p> 	<p>IL</p> <pre> LD   S2           Reading of value of S2 S    H2           Setting of the variable H2 or CAL  SR_H2 (S1 := S2)  Invocation of flip-flop SR_H2 with                         current transfer parameter S2. LD   SR_H2.Q1     Reading of output value Q1 of flip-flop ST   H2           Assignment of read value to the                         variable H2. </pre>
	<p>ST</p> <pre> SR_H2 (S1 := S2);  Invocation of flip-flop SR_H2 using                     a current transfer parameter. H2 := SR_H2.Q1;   Assignment of output value Q1 of                     SR_H2 to the variable H2. </pre>

Table A9.1:  
The storage function Set

Buzzer H1 is to be switched on by pressing an indicator push button S1.

*Problem description*



*Positional sketch*

1. Drawing up and constructing the circuit diagram
2. Declaration of the PLC program variables
3. Formulation of the PLC program into one of the PLC programming languages
4. Testing and commissioning of the PLC program and system

*Exercise definition*

### 1. Drawing up the circuit diagram and assembling the equipment

*Implementation*

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Signal input, electrical
1	Signalling device

*Components list*



Prior to wiring the installation:

- Switch off power supply!

⇒ Establish the electrical connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 4. Testing and commissioning of the PLC program and system



Prior to commissioning of the installation:

- Check the assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

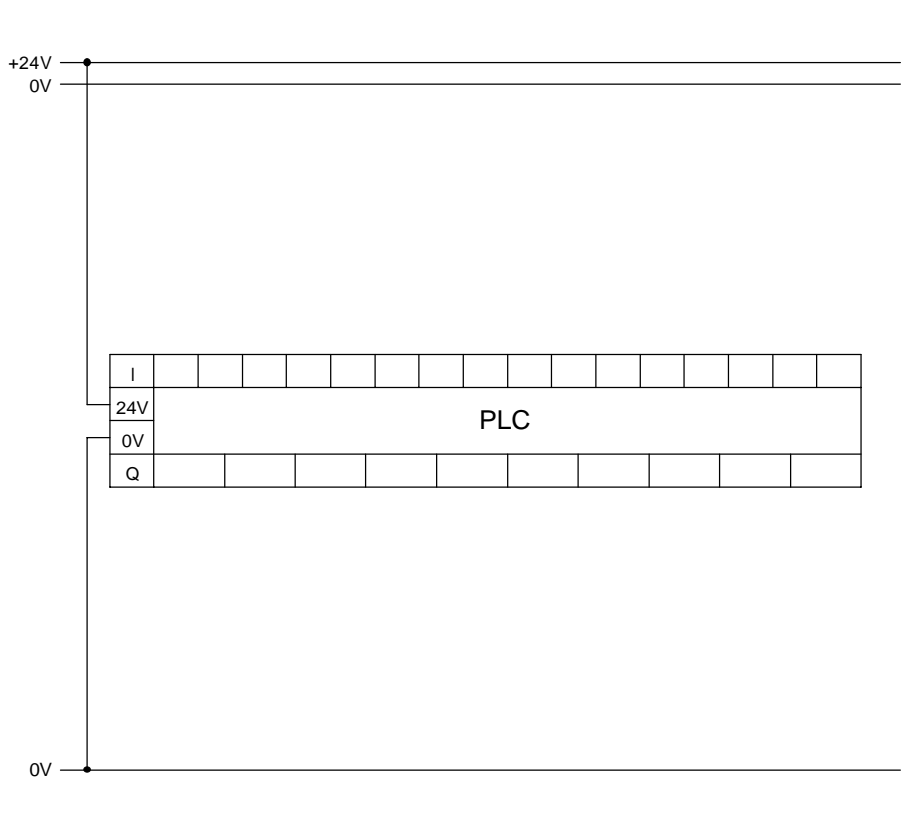
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

### WORKSHEET

#### 1. Drawing up the circuit diagram and assembling the equipment

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



*Circuit diagram, electrical*

#### 2. Declaration of the PLC program variables

Declare the variables required in your PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

## WORKSHEET

### 3. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task in one of these languages:

- ▣ Function block diagram (FBD)
- ▣ Ladder diagram (LD)
- ▣ Instruction list (IL)
- ▣ Structured text (ST)

PLC program

The image shows a large empty rectangular box with a vertical line on the left side and horizontal lines inside, intended for writing the PLC program. The box is positioned to the right of the 'PLC program' label.

## WORKSHEET

Mark the characteristics applicable to a function or function block.

<i>Characteristic</i>	<i>Function</i>		<i>Function block</i>	
	<input type="checkbox"/> yes	<input type="checkbox"/> no	<input type="checkbox"/> yes	<input type="checkbox"/> no
Name	<input type="checkbox"/> yes	<input type="checkbox"/> no	<input type="checkbox"/> yes	<input type="checkbox"/> no
Input parameters	<input type="checkbox"/> one	<input type="checkbox"/> several	<input type="checkbox"/> one	<input type="checkbox"/> several
Output parameters	<input type="checkbox"/> one	<input type="checkbox"/> several	<input type="checkbox"/> one	<input type="checkbox"/> several
Component part of the declaration of variables	<input type="checkbox"/> yes	<input type="checkbox"/> no	<input type="checkbox"/> yes	<input type="checkbox"/> no
Status information	<input type="checkbox"/> yes	<input type="checkbox"/> no	<input type="checkbox"/> yes	<input type="checkbox"/> no

Answer the following question:

*Question*

1. Which status information is stored in the SR function block?

---



---



---



---

# A-80

---

## Exercise 9



Programmable logic controllers

*Subject*

### Drill breakage monitoring

*Title*

Setting and resetting of an output

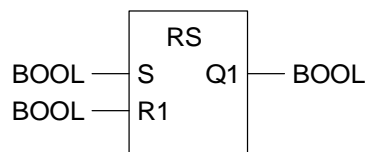
- To be able to set and reset a stored output of a PLC

*Training aim*

### Function block RS, dominant resetting flip-flop

*Technical knowledge*

Function block RS contains a dominant resetting flip-flop



*Fig. A10.1:  
Function block RS*

The behaviour of the block represented in fig. A10.1 is as follows:

- A 1-signal at reset input R1 sets the value of Q1 to 0, irrespective of which value applies at input S.
- A 1-signal at set input S sets output Q1 to 1 only if a 0-signal simultaneously applies at the R1 input.
- If 1-signals apply both at inputs S and R1, output Q1 is reset.



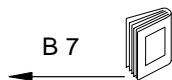
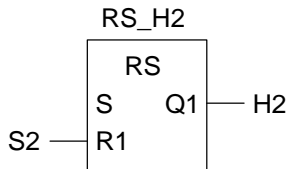
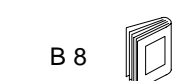

# A-82

## Exercise 10

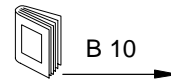
### Realisation of the "Reset" storage function in the individual programming languages

The solutions for the example below are listed in table A10.1.

*Example* Actuation of push button S2 is to cause lamp H2 to be switched off.

	<pre> VAR   S2 AT %IX5 : BOOL;      (* Switch S2 at input IX5      *)   H2 AT %QX6 : BOOL;      (* Lamp H2 at output QX6      *)   RS_H2      : RS;        (* RS flip-flop named RS_H2   *)                                    (* For storage of status      *)                                    (* of H2                      *) END_VAR         </pre>
<p>B 7</p> 	<p>FBS</p>
	<div style="display: flex; align-items: center;"> <div style="margin-right: 20px;">  </div> <div> <p>RS flip-flop RS_H2 for storage of status of the variable H2. Direct assignment of value of Q1 to the variable H2.</p> </div> </div>
<p>B 8</p> 	<p>LD</p>
<p>Table A10.1: The storage function reset</p>	<div style="display: flex; align-items: center;"> <div style="margin-right: 20px;">  </div> <div> <p>Rung with normally open contact and reset coil.</p> </div> </div>

IL		
LD	S2	Reading of value of S2
R	H2	Resetting of variable of H2
or		
CAL	RS_H2 (R1 := S2)	Invocation of flip-flop RS-H2 using current transfer parameter S2.
LD	RS_H2.Q1	Reading of output value Q1 of flip-flop RS-H2.
ST	H2	Assignment of the read value to the variable H2.
ST		
RS_H2 (R1 := S2);		Invocation of flip-flop RS_H2 using a current transfer parameter.
H2 := RS_H2.Q1;		Assignment of output value Q1 of RS_H2 to the variable H2.



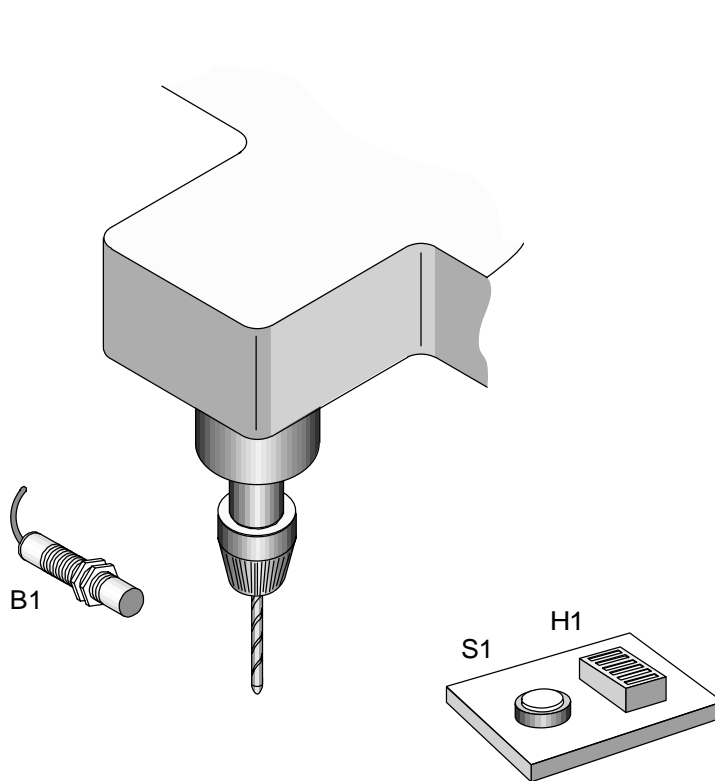
*Table A10.1:  
The storage function  
reset (continuation)*

# A-84

## Exercise 10

*Problem description* The drill on a drilling unit is monitored by means of a drill breakage sensor (B1).

If the drill is broken, the sensor interrupts the circuit. A buzzer (H1) is to sound in this event. The buzzer can only be switched off via push button S1.



*Positional sketch*

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Declaration of the PLC program variables
  3. Formulation of the PLC program into one of the PLC programming languages
  4. Testing and commissioning of the PLC program and system

### 1. Drawing up the circuit diagram and assembling the equipment

*Implementation*

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Signal input, electrical
1	Signalling device
1	Proximity switch, optical

*Components list*

Prior to wiring the installation:

- Switch off power supply!



⇒ Establish the electrical connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### **Note**

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LDR, FBD, STL and ST.

### 4. Testing and commissioning of the PLC program and system



**Prior to** commissioning of the installation:

- Check assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

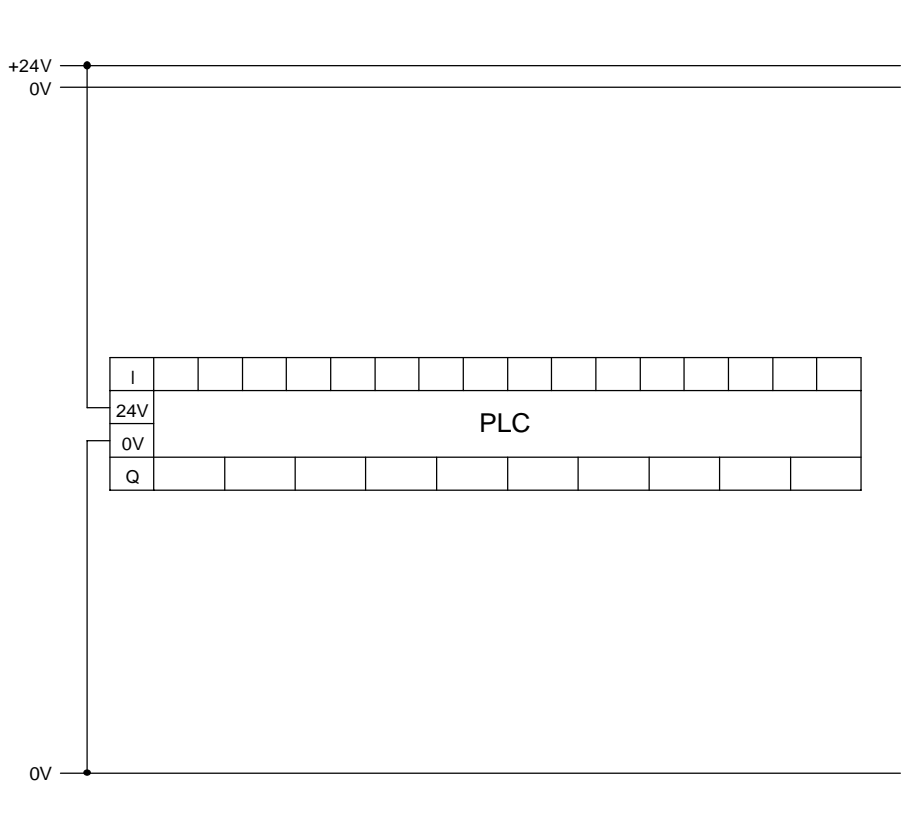
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the circuit diagram and assembling the equipment

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



*Circuit diagram, electrical*

### 2. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

## WORKSHEET

### 3. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

PLC program

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*Question* Answer the following question:

1. What is the resulting program sequence if a set dominant flip-flop is used instead of a reset dominant flip-flop?

---

---

---

---



Programmable logic controllers

*Subject*

### Activating a cylinder

*Title*

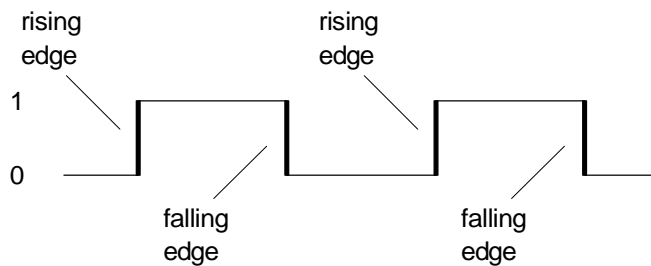
Signal edges

- To be able to describe the function of signal edges
- To be able to use the standard function block R\_TRIG for the recognition of a rising signal edge

*Training aim*

PLC applications frequently require the detection and evaluation not of a signal itself, but of the point of change of a signal. These signal changes are described as edges.

*Technical knowledge*



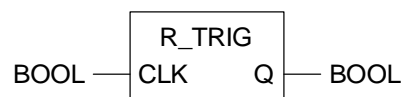
*Fig. A11.1:  
Edges*

Rising (positive) edges mark the instant, during which a signal change takes place from 0 to 1.

Falling (negative) edges mark the instant, during which a signal change takes place from 1 to 0.

### Function block R\_TRIG for rising edge detection

This standard function block is used for the detection of a rising edge.



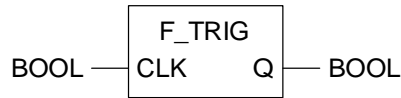
*Fig. A11.2:  
Function block R\_TRIG*

If a signal change takes place from 0 to 1 at input CLK, output Q assumes the value 1 during a program cycle.

### Function block F\_TRIG for falling edge detection

This standard function block is used for the detection of a falling edge.

Fig. A11.3:  
Function block F\_TRIG



If a signal change takes place from 1 to 0 at input CLK, output Q carries a 1-signal during a program cycle.

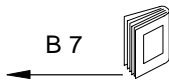
### Realisation of edge evaluation in the individual programming languages

Edge evaluation has been programmed in the languages FBD, LD, IL and ST for the example below.

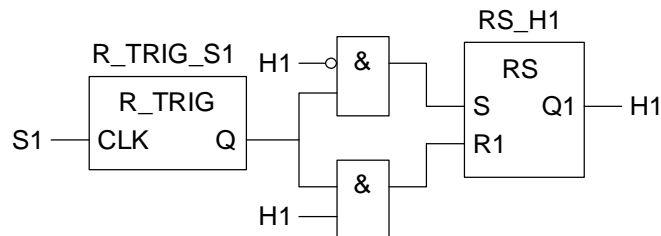
*Example* Actuation of a push button S1 causes a lamp H1 to be switched on. Repeat actuation of push button S1 switches off the lamp again.

```

VAR
  S1 AT   %IX1  : BOOL;   (* Push button S1      *)
  H1 AT   %QX1  : BOOL;   (* Lamp H1          *)
  R_TRIG_S1 : R_TRIG (* Rising edge of S1      *)
  RS_H1    : RS;    (* Flip-flop for H1 *)
END_VAR
  
```



FBD



Examining signal S1 with function block R\_TRIG\_S1 for rising edge. Depending on the status of lamp H1, a positive results leads to H1 being switched on or off.

Table A11.1:  
Evaluation of a rising edge

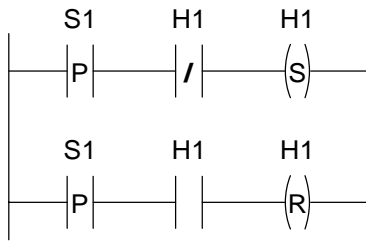
LD		
	<p>Detection of a positive edge by means of a special edge contact. Depending on the status of the lamp, this may be switched on in the first rung and switched off in the second rung.</p>	
IL		
CAL	R_TRIG_S1 (CLK := S1)	Invocation of function block R_TRIG_S1.
LD	R_TRIG_S1.Q	Depending on the result of the edge evaluation, the lamp switched on ...
ANDN	H1	
S	H1	
LD	R_TRIG_S1.Q	or off.
AND	H1	
R	H1	
ST		
	R_TRIG_S1 (CLK := S1);	Invocation of function block R_TRIG_S1.
	RS_H1 (S := R_TRIG_S1.Q & NOT H1, R1 := R_TRIG_S1.Q & H1);	Invocation of flip-flop RS_H1.
	H1 := RS_H1.Q1;	Status of flip-flop RS_H1 is mapped to H1.



Table A11.1:  
(Continuation)

Since the conditions for switching the lamp on and off are mutually exclusive, a set dominant flip-flop may also be used instead of a reset dominant one.

*Comment*

### Realisation of edge evaluation without special edge function blocks

If a PLC system does not support special function blocks for the detection of edges, memories may be used for the detection of signal changes.

PLC programs are continually cyclically processed. In order to detect a signal change, it is necessary to check whether the status of a signal has changed from one processing cycle to the next. To do this, the old signal status has to be stored and compared with the new current status.

Fig. A11.4 illustrates the method used to detect a rising edge.

```

VAR
  Signal    AT %IX1  : BOOL;      (* current input signal      *)
  S_Edge    AT %MX1  : BOOL;      (* detects edge of the       *)
                                     (* input signal               *)
  S_old     AT %MX2  : BOOL := 0; (* stores old status of     *)
                                     (* input signal               *)
  RS_S_old          : RS;        (* flip-flop for memory     *)
                                     (* S_old                      *)
END_VAR

```

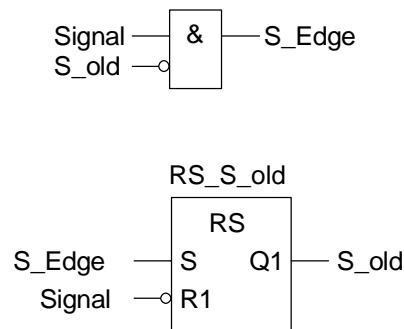


Fig. A11.4:  
Detecting a rising edge

The signal to be examined is represented by the variable "signal". The old status of the variable "signal" is stored in the memory "S\_Old". If a rising edge occurs, the memory "S\_Edge" assumes the value 1 for one processing cycle.

It should be noted that the memory "S\_Old" must maintain the value 0 at the program start (in the machining cycle).

The program parts shown have been formulated in the language FBD as an example.

The evaluation of a falling edge may be realised as illustrated below.

```

VAR
  Signal    AT %IX1  : BOOL;           (* current input signal *)
  S_Edge    AT %MX1  : BOOL;           (* detects edge of the *)
                                         (* input signal *)
  S_old     AT %MX2  : BOOL := 0;      (* stores old status of *)
                                         (* input signal *)
  SR_S_old          : SR;              (* flip-flop for memory *)
                                         (* S_old *)
END_VAR

```

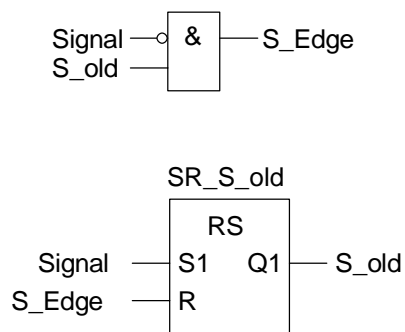
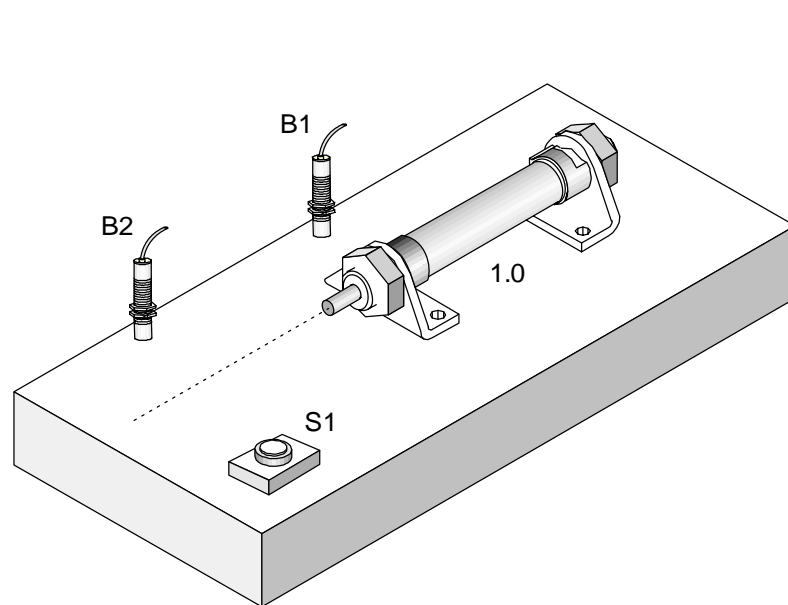


Fig. A11.5:  
Detecting a falling edge

# A-94

## Exercise 11

*Problem description* A cylinder is actuated by means of a spring-retained solenoid valve (coil Y1). Two proximity switches signal the positions "extended" (B2) and "retracted" (B1). Push button (S1) is used to actuate the cylinder in such a way that it advances from the retracted end position into the opposite direction. The cylinder must advance only once per push button actuation. To trigger a second movement of the cylinder, the push button must be released and actuated afresh.



*Positional sketch*

- Exercise definition*
1. Drawing up the electrical circuit diagram and constructing the circuit
  2. Declaration of the PLC program variables
  3. Formulation of the PLC program into one of the PLC programming languages
  4. Testing and commissioning of the PLC program and system

### 1. Drawing up the electrical circuit diagram and constructing the circuit

*Implementation*

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull distributor
1	Single-acting cylinder
1	5/2-way single solenoid valve
1	Signal input, electrical
1	Proximity switch, inductive
1	Proximity switch, capacitive
	Plastic tubing

*Components list*

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 4. Testing and commissioning of the PLC program and system



**Prior to commissioning of the installation:**

- Check assembled circuit with the help of the circuit diagrams!

**Commissioning of the installation:**

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply to operating pressure (see data sheets for pneumatic components)!

**Operation of the installation:**

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

⇒ Correct any errors occurring in the PLC program.

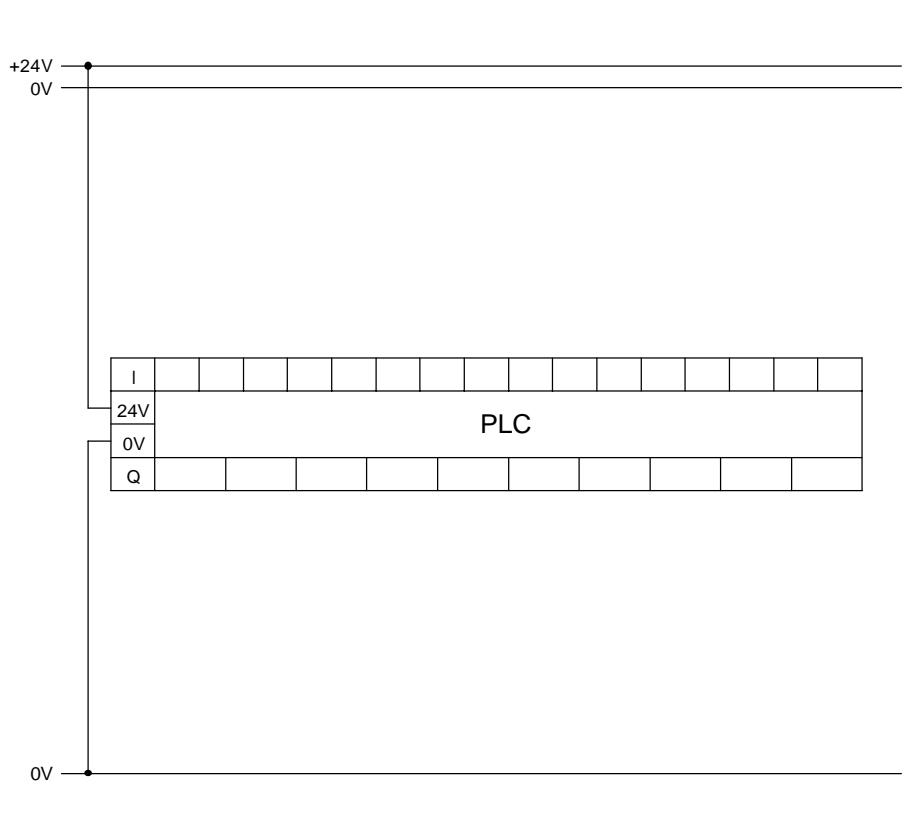
⇒ Document your solution.



## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.

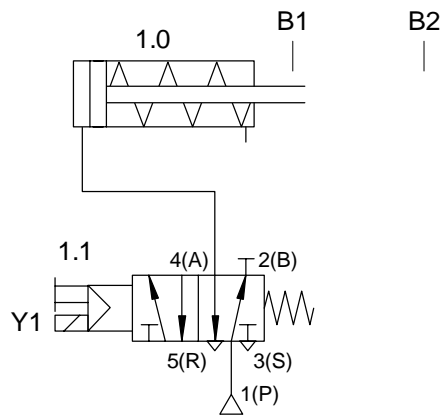


Circuit diagram, electrical

## WORKSHEET

Configure the control system

*Circuit diagram,  
electro-pneumatic*



### 2. Declaration of the PLC program variables

Declare the variables required in your PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

**WORKSHEET**

**3. Formulation of the PLC program into one of the PLC programming languages**

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*PLC program*

# A-100

---

## Exercise 11

### **WORKSHEET**

*Questions* Answer the following questions:

1. What is understood by a negative edge?

---

---

---

---

2. What effect does the period of actuation have on the program execution?

---

---

---

---

Programmable logic controllers

*Subject*

### **Bonding of components**

*Title*

Pulse

- To be able to use standard function block TP for pulse time response

*Training aim*

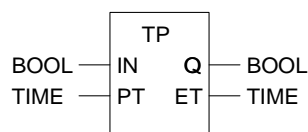
A large number of control tasks require the programming of time. Through IEC1131-3, standard function blocks are available for timers with different time response.

*Technical knowledge*

Timers are available for the realisation of a pulse time response, a switch-on signal delay and a switch-off signal delay.

### **Function block TP, pulse timer**

Standard function block TP (fig. A12.1) is a pulse timer



*Fig. A12.1:  
Function block R\_TRIG*

The response of function block TP is as follows:

- Function block TP is started via a short or long signal at input IN.
- Once the timer has started, a 1-signal applies at output Q for the time specified at input PT.
- The current timer value (the time, which has elapsed since the start) is available at output ET.
- The timer can only be started again once it has expired.

# A-102

## Exercise 12

### Programming of a pulse timer in the individual languages

The use of a pulse timer in the individual programming languages is illustrated with the help of the example given below.

#### Example

Workpieces are clamped securely for a period of 12 seconds for a machining process by means of a special device on a cylinder Y1. The process is triggered by actuating a start button S1.


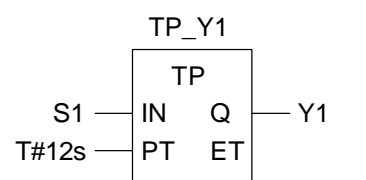

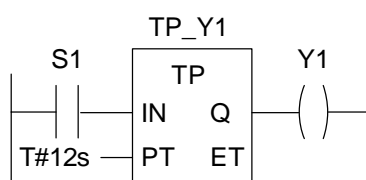
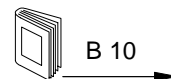
	<pre>VAR   S1 AT %IX1 : BOOL;      (* Push button S1      *)   Y1 AT %QX1 : BOOL;      (* Coil Y1 for cylinder *)   TP_Y1      : TP;        (* Pulse timer named    *)                           (* TP_Y1 for clamping process *) END_VAR</pre>
B 7 	<p>FBD</p> <div data-bbox="510 1008 877 1187"></div> <p>Timer function block TP_Y1, connected with the current parameters.</p>
B 8 	<p>LD</p> <div data-bbox="510 1344 877 1523"></div> <p>Interconnecting timer function block TP_Y1 into the rung.</p>

Table A12.1:  
Use of a pulse timer

IL		
CAL	TP_Y1 (IN := S1, PT := T#12s)	Invocation of function block TP_Y1.
LD	TP_Y1.Q	Reading of output Q of TP_Y1.
ST	Y1	Storage of current result to Y1.
ST		
	TP_Y1 (IN := S1, PT := T#12s);	Invocation of function block TP_Y1.
	Y1 := TP_Y1.Q;	Assignment of output Q of TP_Y1 to Y1.

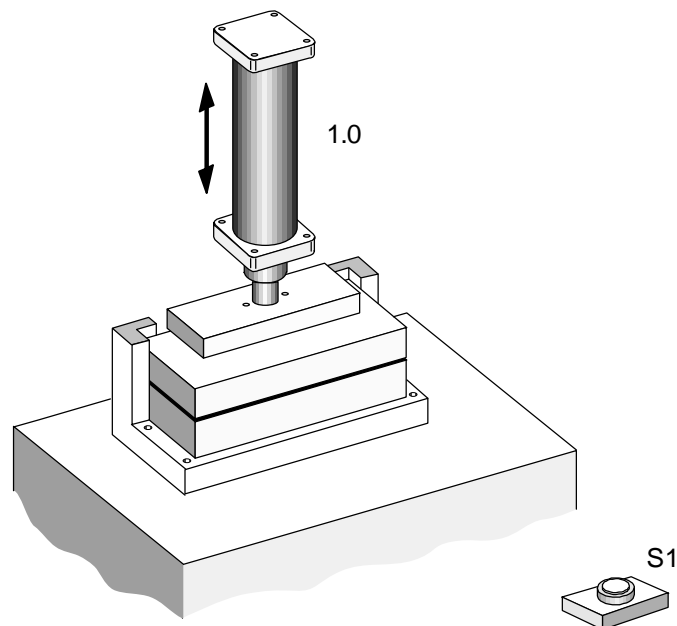


*Table A12.1:  
Use of a pulse timer  
(continuation)*

# A-104

## Exercise 12

*Problem description* Two components are to be bonded together with the help of a pneumatic cylinder 1.0. To do this, the bonding surfaces are pressed together with a defined force for 5 seconds. The time is commenced once the cylinder advances from its retracted end position (sensor B1 1). Once the 5 seconds have expired, the cylinder is to return to the initial position. The bonding process is started by a push button S1.



*Positional sketch*

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Declaration of the PLC program variables
  3. Formulation of the PLC program into one of the PLC programming languages
  4. Testing and commissioning of the PLC program and system



### 1. Drawing up the electrical circuit diagram and constructing the circuit

Implementation

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable of connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull distributor
1	Single-acting cylinder
1	5/2-way single solenoid valve
1	Signal input, electrical
1	Proximity switch, inductive-magnetic
	Plastic tubing

Components list

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comments.

#### Note

The component parts of the declaration of variables in this exercise section is represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 4. Testing and commissioning of the PLC program and system



#### Prior to commissioning of the installation:

- Check the assembled circuit with the help of the circuit diagrams!

#### Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheet for pneumatic components)!

#### Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

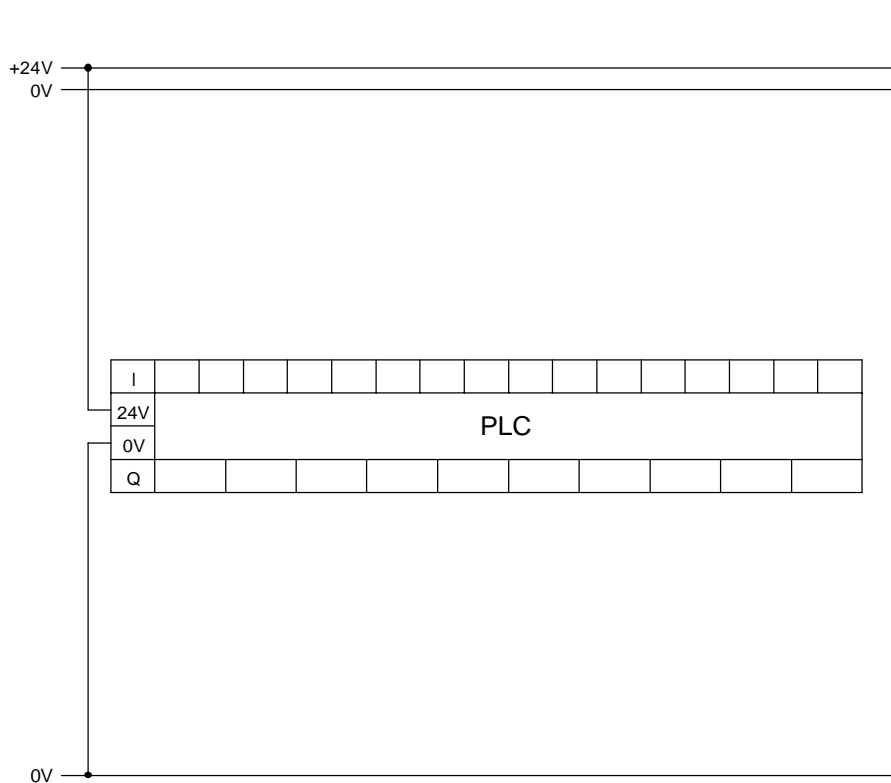
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

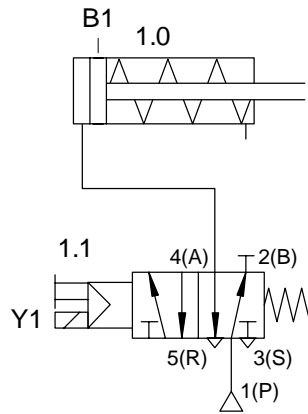
Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



Circuit diagram, electrical

## WORKSHEET

Configure the control system



Circuit diagram,  
electro-pneumatic

### 2. Declaration of the PLC program variables

Declare the variables required in your PLC program:

Designation	Data type	Address	Comment

Declaration of variables



# A-110

## Exercise 12

### WORKSHEET

Questions Answer the following questions:

1. Specify the name and the function of the parameters of the pulse timer.

---

---

---

---

2. What is the response of the timer, if a new start signal is given prior to the timer expiring ?

Complete the diagram.

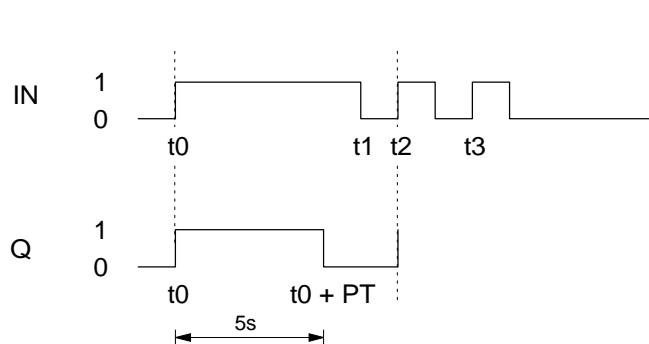


Fig. A12.2:  
Time response of  
pulse timer

Programmable logic controllers

*Subject*

### Embossing device

*Title*

Switch-on signal delay

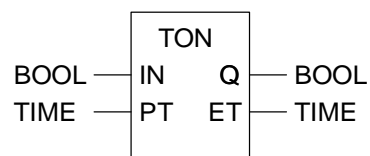
- To be able to realise a switch-on signal delay by using the standard function block TON

*Training aim*

### Function block TON, Switch-on signal delay

*Technical knowledge*

The standard function block TON is used to generate a switch-on signal delay.



*Fig. A13.1:  
Function block TON*

The behaviour of function block TON is as follows:

- Function block TON is started by means of a 1-signal at input IN.
- Upon expiry of the time specified at input PT, output Q carries a 1-signal. The 1-signal at output Q applies until the input signal IN reverts to the value 0.
- If the duration of the input signal IN is shorter than the specified time PT, the value of output Q remains a constant 0.
- The current timer value (the time, which has elapsed since the start) is available at output ET.



# A-112

## Exercise 13

### Programming of a switch-on signal delay in the individual languages

The use of a switch-on signal delay is demonstrated in the following example:

#### Example

The door of a bus will only close when the boarding area has been clear for a specified period (5 seconds). This is monitored by means of a light barrier.


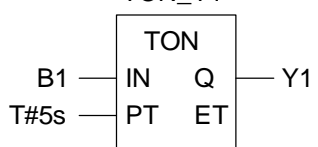
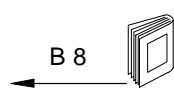
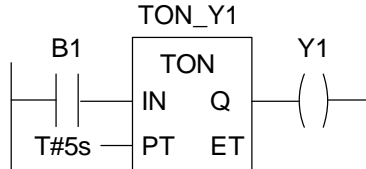
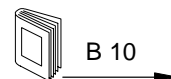
	<pre> VAR   B1 AT %IX1 : BOOL;    (* Light barrier *)   Y1 AT %QX1 : BOOL;    (* Coil Y1 for cylinder for closing *)                           (* the door *)   TON_Y1      : TON;    (* Switch-on signal delay named *)                           (* TON_Y1 for closing of door *) END_VAR         </pre>
<p>B 7</p> 	<p>FBD</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <p>TON_Y1</p>  </div> <div style="margin-left: 20px;"> <p>Connecting the inputs and outputs of function block TON_Y1 with current parameters.</p> </div> </div>
<p>B 8</p> 	<p>LD</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <p>TON_Y1</p>  </div> <div style="margin-left: 20px;"> <p>Interconnection of function block TON_Y1 into the rung.</p> </div> </div>

Table A13.1:  
Use of a  
switch-on signal delay



IL	
CAL TON_Y1 (IN := B1, PT := T#5s)	Invocation of function block TON_Y1.
LD TON_Y1.Q	Reading of output Q of TON_Y1.
ST Y1	Storing of current result to Y1.
ST	
TON_Y1 (IN := B1, PT := T#5s);	Invocation of function block TON_Y1.
Y1 := TON_Y1.Q;	Assignment of output Q of TON_Y1 to Y1.

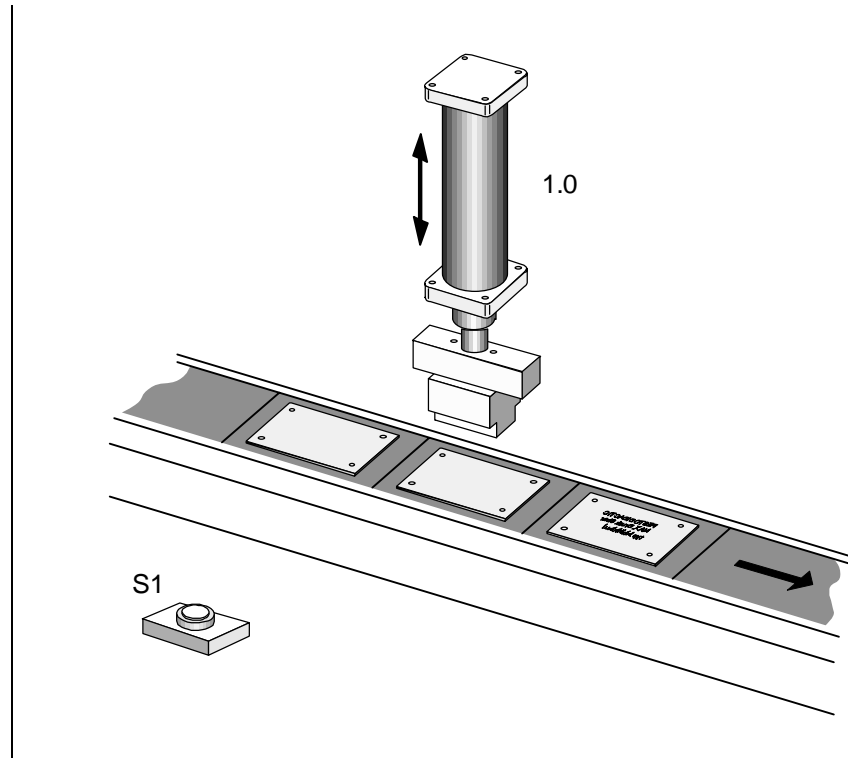


*Table A13.1:  
Use of a  
switch-on signal delay  
(continuation)*

# A-114

## Exercise 13

*Problem description* A workpiece is to be embossed by activating a start button (S1). In order to ensure that the embossing cycle is not triggered inadvertently, the embossing cycle is to be triggered only after 3 seconds have expired. During this time the start button must be permanently actuated. The position of the cylinder 1.0 is established by means of the proximity switches B1 (retracted) and B2 (extended).



*Positional sketch*

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Declaration of the PLC program variables
  3. Formulation of the PLC program into one of the PLC programming languages
  4. Testing and commissioning of the PLC program and system

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Implementation

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull distributor
1	Single-acting cylinder
1	5/2-way single solenoid valve
1	Signal input, electrical
1	Proximity switch, inductive
1	Proximity switch, capacitive
	Plastic tubing

Components list

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 4. Testing and commissioning of the PLC program and system



**Prior to** commissioning of the installation:

- Check assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheets for pneumatic components)!

Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

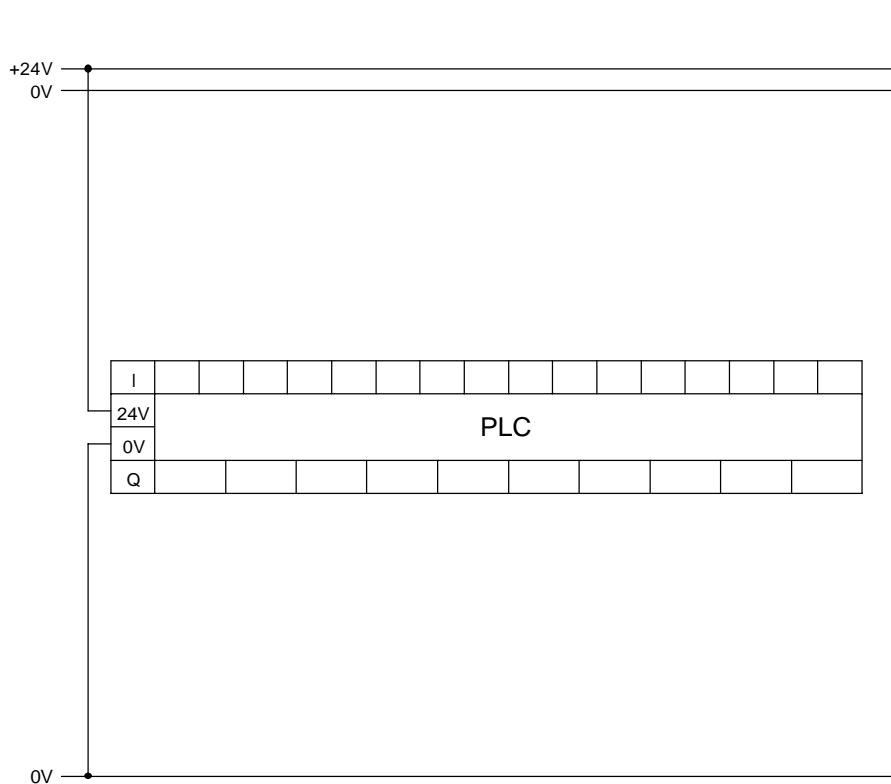
⇒ Correct any error occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



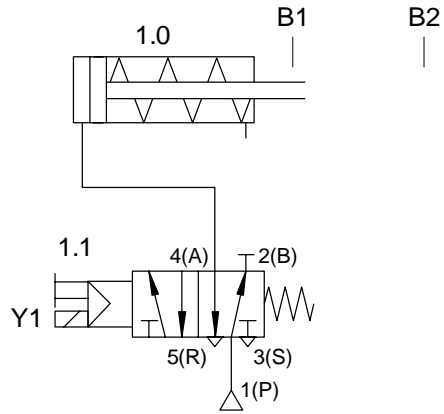
Circuit diagram, electrical

# A-118

## Exercise 13

Configure the control system.

*Circuit diagram,  
electro-pneumatic*



### 2. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

### 3. Formulation of the PLC program into one of the PLC programming languages

Formulate the solution of the control task into one of these languages:

- Function block diagram (FBD)
- Ladder diagram (LD)
- Instruction list (IL)
- Structured text (ST)

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*PLC program*

## **WORKSHEET**

*Question* Answer the following question:

1. The embossing cycle has been initiated. However, the start button is released before the 3 seconds have expired. What effect does this have on the program execution?

---

---

---

---



Programmable logic controllers

*Subject*

### Clamping device

*Title*

Switch-off signal delay

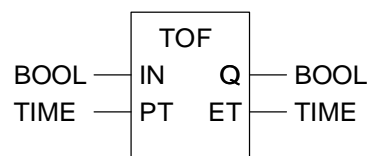
- To be able to realise a switch-off signal delay by using the standard function block TOF

*Training aim*

### Function block TOF, switch-off signal delay

*Technical knowledge*

The standard function block TOF (fig. A14.1) is used to generate a switch-off signal delay.



*Fig. A14.1:  
Function block TOF*

The behaviour response of function block TOF is as follows:

- Function block TOF is started via a 1-signal at input IN. Output Q simultaneously receives the value 1.
- After the input signal IN has reverted to the value 0, the 1 signal continues to be applied at output Q for the time specified at the PT input and then returns to the value 0.



# A-122

## Exercise 14

### Programming of a switch-off signal delay in the individual languages

The use of a switch-off signal delay is demonstrated in the following:

#### Example

The door of a furnace includes a lock so that it cannot be opened instantly during the burning process. If a signal is given to open the door, this will only be unlocked after 10 minutes has expired.


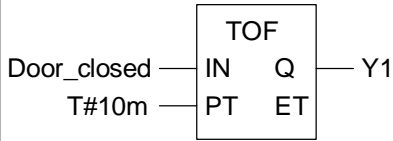

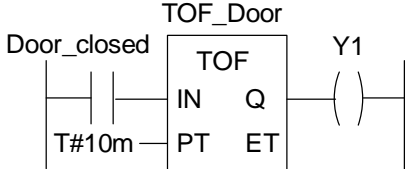
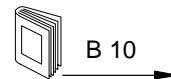
	<pre> VAR   Door_closed AT  %MX1: BOOL; (* Memory for latching of      *)                                 (* furnace door                *)   Y1          AT  %QX1: BOOL; (* Coil Y1 for cylinder    *)                                 (* opening of furnace door     *)   TOF_Door    : TOF; (* Switch-off signal delay      *)                                 (* named TOF_Door              *) END_VAR         </pre>
<p>B 7</p> 	<p>FBD</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <p>TOF_Door</p>  </div> <div style="margin-left: 20px;"> <p>Connection of Inputs and outputs of function block TOF_Door with current parameters.</p> </div> </div>
<p>B 8</p> 	<p>LD</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <p>TOF_Door</p>  </div> <div style="margin-left: 20px;"> <p>Interconnection of function block TOF_Door into the rung.</p> </div> </div>

Table A14.1:  
Use of a  
switch-off signal delay

IL	
CAL TOF_Door (IN := Door_closed, PT := T#10m)	Invocation of function block TOF_Door.
LD TOF_Door.Q	Reading of output Q of TOF_Door.
ST Y1	Storage of current result to Y1.
ST	
TOF_Door (IN := Door_closed, PT := T#10m);	Invocation of function block TOF_Door.
Y1 := TOF_Door.Q;	Assignment of output Q of TOF_Door to Y1.

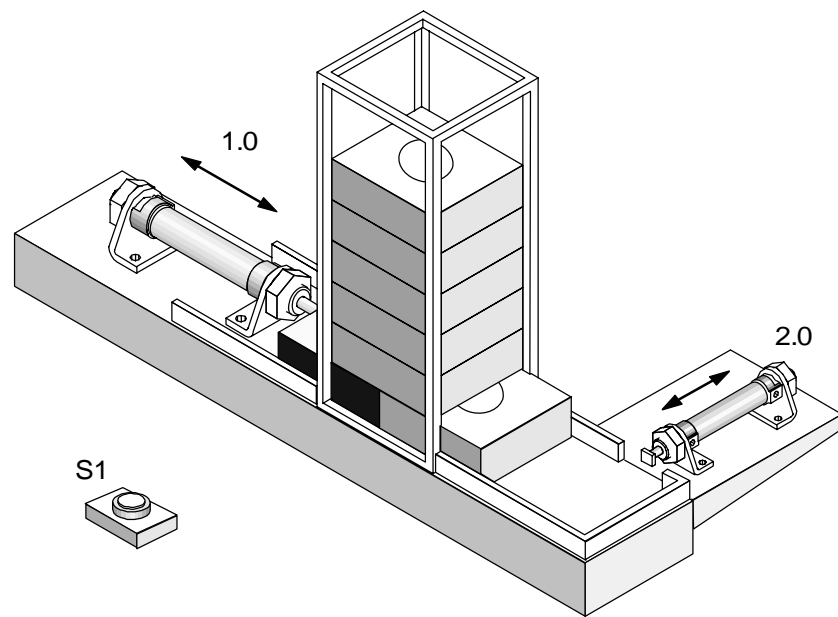


*Table A14.1:  
Use of a  
switch-off signal delay  
(continuation)*

# A-124

## Exercise 14

*Problem description* A workpiece is to be clamped by activating the start button S1. When the workpiece is clamped by cylinder 1.0, cylinder 2.0 extends and embosses the workpiece. Since the workpiece requires time to cool down, it remains clamped for a period of 3 seconds. This time is started with the advancing of cylinder 1.0.



*Positional sketch*

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Declaration of the PLC program variables
  3. Formulation of the PLC program into one of the programming languages
  4. Testing and commissioning of PLC program and system

### 1. Creating the electrical circuit diagram and constructing the circuit

Implementation

⇒ Complete the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

Quantity	Description
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull distributor
1	Single-acting cylinder
1	Double-acting cylinder
1	5/2-way single solenoid valve
1	5/2-way double solenoid valve
1	Signal input, electrical
4	Proximity switch, inductive
1	Quick push-pull distributor
	Plastic tubing

Table A14.1:  
Components list

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

### 2. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 3. Formulation of the PLC program into one of the PLC programming languages

⇒ Select one of the languages supported by your PLC system for programming. Suitable languages for the formulation of logic control systems are LD, FBD, IL and ST.

### 4. Testing and commissioning of the PLC program and system



**Prior to commissioning of the installation:**

- Check the assembled circuit with the help of the circuit diagrams!

**Commissioning of the installation:**

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheet for pneumatic components)!

**Operation of the installation:**

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

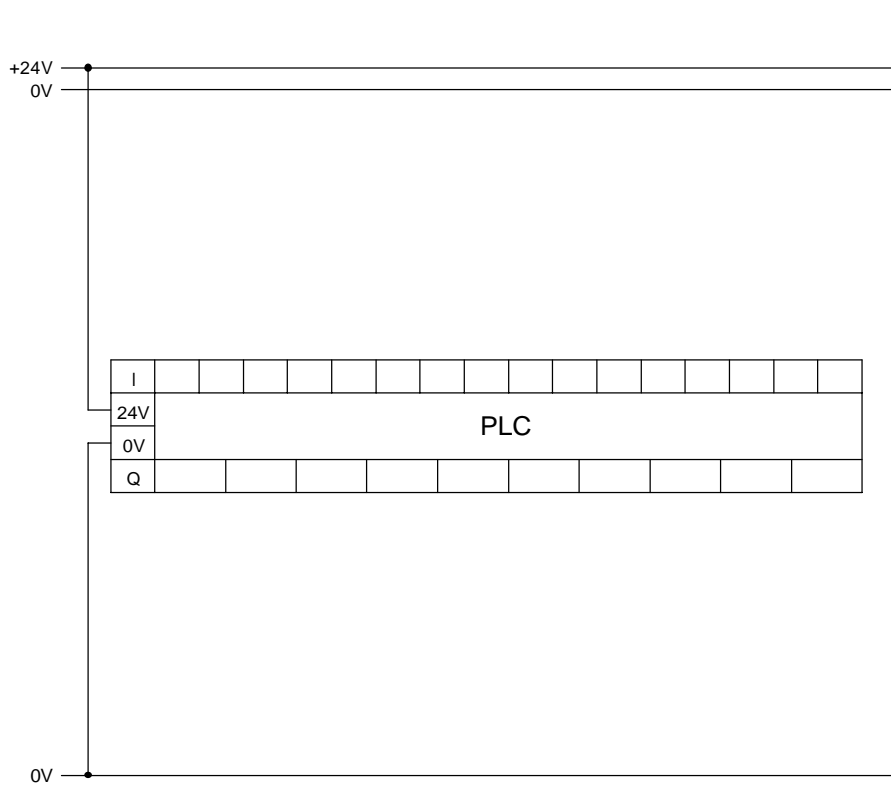
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



Circuit diagram, electrical







## **WORKSHEETT**

*Question* Answer the following question:

1. Through which signal is the time for the switch-off signal delay started ?

---

---

---

---

Programmable logic controllers

*Subject*

### Lifting device for packages

*Title*

Linear sequence

*Training aim*

- To be able to design and represent simple sequence control systems in accordance with IEC 848.
- To be able to program a sequence control system consisting of a linear sequence
- To be able to use the programming language Sequential Function Chart

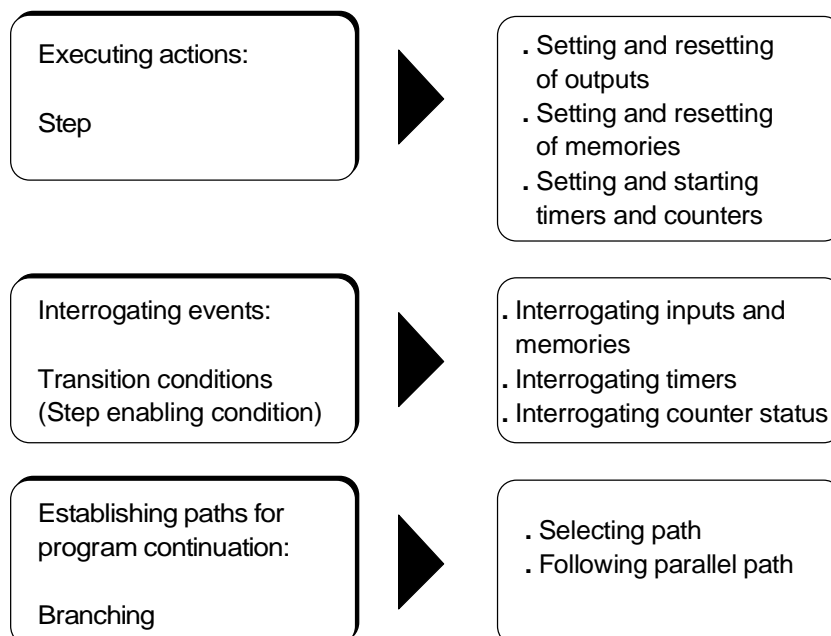
### Sequence control systems

*Technical knowledge*

Sequence control systems describe processes, which proceed in several clearly separated steps.

The transition from one step to the next is dependent on the process statuses. It is possible for the process to be branched into partial processes in relation to the process statuses established.

The program of a sequence control system must therefore fulfil three basic exercises:



*Fig. A15.1:  
Functions of a  
control program*

### General representation of a sequence control system

Function chart in accordance with IEC 848 is for the description and planning of a sequence control system. This permits a clear, graphic representation of the behaviour and function of a sequence control system.

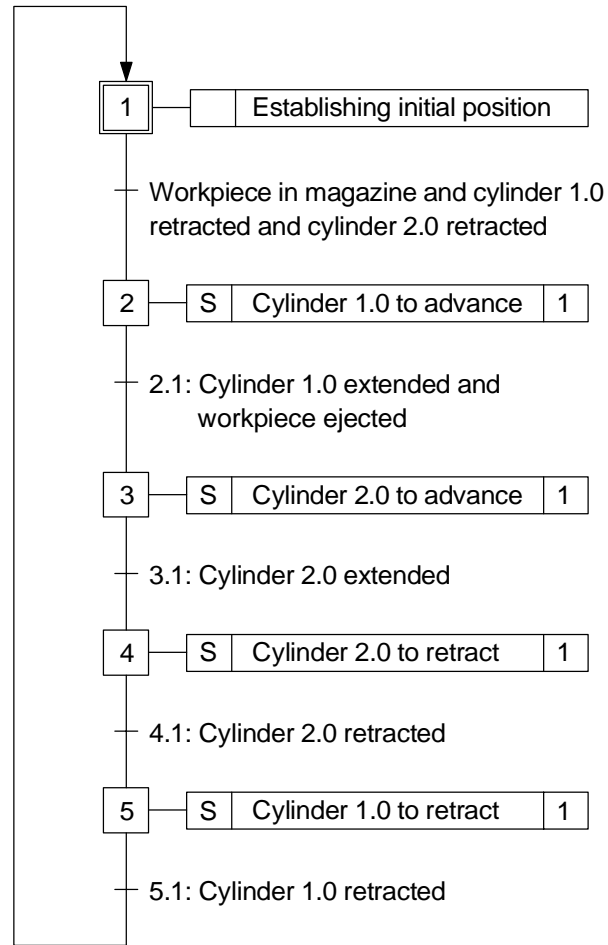


Fig. A15.2:  
Example of a  
sequence control system

The example shown above describes the following control task:

A workpiece is made available from a magazine for further machining. The workpiece is ejected from the magazine by a cylinder 1.0, and then transferred onto a conveyor belt via a slide by means of a second cylinder 2.0.

## Programming of a sequence control system in Sequential Function Chart

Sequence control systems can be easily and clearly programmed in a sequential function chart. The sequential function chart is derived from the function chart in accordance with IEC 848.

The example shown Fig. A15.3 illustrates the use of sequential function chart for the control task mentioned above.

- Step => Classification into actions
- Transition => Description by means of transition condition
- Alternative branch and junction
- Parallel branch and junction

When the PLC program is started, the step designated as initial step S1 automatically becomes active. A system is frequently moved into the initial position as a result of the initial step. In the example shown, step S1 is a void step. If the subsequent step enabling conditions – cylinder 1.0 and 2.0 are retracted and the magazine contains workpieces – are met, step S2 is set and step S1 reset. It should be noted that the step names represent names in the sense of IEC 1131-3. They must therefore start with a letter or an underline. In addition, insofar as this is possible for the process concerned, a feedback variable is specified in the third field of each action indicating the end of the action.

In step S2, cylinder 1.0 is extended by setting coil Y1. When this cylinder has reached its forward end position and the workpiece is in the correct position ( $B2 = 1$ ), step S2 is reset and step S3 activated. Cylinder 1.0 remains extended as a result of the S-qualifier. In step S3, cylinder 2.0 advances due to coil Y2 being set and transfers the workpiece to a slide. Cylinder 2.0 retracts again once it has reached its forward end position. If sensor B5 signals that the retracted end position of cylinder 2.0 has been reached, cylinder 1.0 also retracts. Sensor B3 now signals the end of the sequence and the program returns to the start. The complete step sequence is repeated again.

# A-134

## Exercise 15

```
VAR
  Y1 AT %QX1 : BOOL;      (* Coil Y1 for cylinder 1.0 *)
  Y2 AT %QX2 : BOOL;      (* Coil Y2 for cylinder 2.0 *)
  B1 AT %IX1 : BOOL;      (* Workpiece in magazine *)
  B2 AT %IX2 : BOOL;      (* Workpiece ejected *)
  B3 AT %IX3 : BOOL;      (* Cylinder 1.0 retracted *)
  B4 AT %IX4 : BOOL;      (* Cylinder 1.0 extended *)
  B5 AT %IX5 : BOOL;      (* Cylinder 2.0 retracted *)
  B6 AT %IX6 : BOOL;      (* Cylinder 2.0 extended *)
END_VAR
```

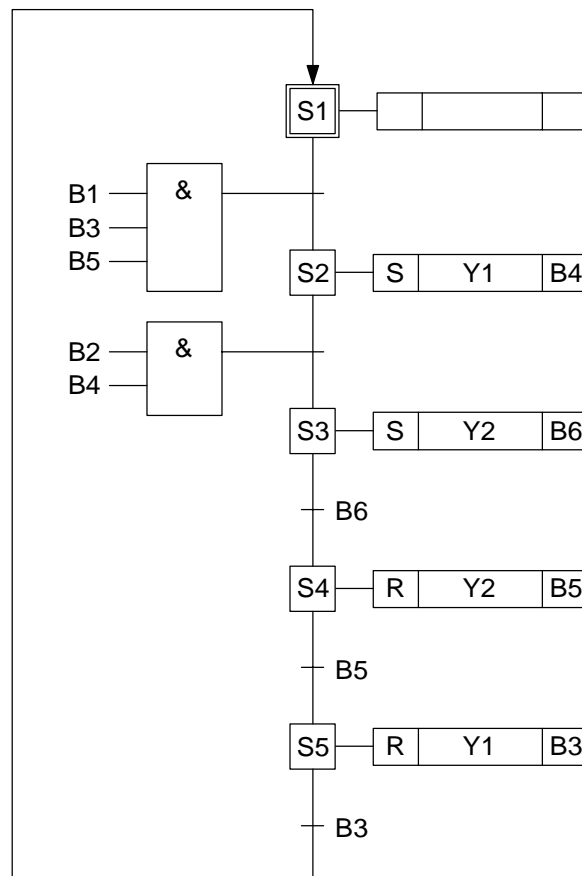


Fig. A15.3:  
Example of  
sequential function chart

### Generating a step sequence by means of RS storage elements

The step sequence may be generated by using storage elements if the programming language Sequential Function Chart is not supported by a PLC program.

Each step is assigned an RS flip-flop. This stores the status of the step. The relevant flip-flop is set, if the step is in the process of being executed; if the step is inactive, the flip-flop is reset.

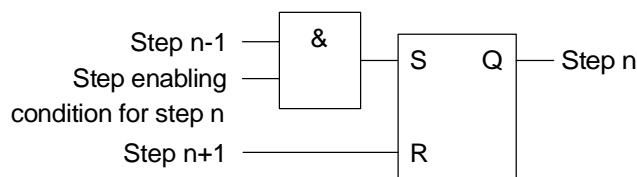


Fig. A15.4:  
Representation of a step

As shown in fig. A15.4, the start conditions for any step n within a step sequence) are:

- The preceding step n-1 is set
- The step enabling condition for step n is met

Each step is reset by the set subsequent step.

In this way, the individual steps of a step sequence are processed consecutively.

The structure of a step sequence is set out in detail in fig. A15.5. The language FBD is used for the programming of the control task in fig. A15.2. All actions occur as boolean actions.

# A-136

## Exercise 15

```

VAR
  Y1 AT %QX1 : BOOL;      (* Coil Y1 for cylinder 1.0 *)
  Y2 AT %QX2 : BOOL;      (* Coil Y2 for cylinder 2.0 *)
  B1 AT %IX1  : BOOL;      (* Workpiece in magazine *)
  B2 AT %IX2  : BOOL;      (* Workpiece ejected *)
  B3 AT %IX3  : BOOL;      (* Cylinder 1.0 retracted *)
  B4 AT %IX4  : BOOL;      (* Cylinder 1.0 extended *)
  B5 AT %IX5  : BOOL;      (* Cylinder 2.0 retracted *)
  B6 AT %IX6  : BOOL;      (* Cylinder 2.0 extended *)
  RS_S1      : RS;         (* Flip-flop for Step S1 *)
  RS_S2      : RS;         (* Flip-flop for Step S2 *)
  RS_S3      : RS;         (* Flip-flop for Step S3 *)
  RS_S4      : RS;         (* Flip-flop for Step S4 *)
  RS_S5      : RS;         (* Flip-flop for Step S5 *)
  RS_Y1      : RS;         (* Flip-flop for coil Y1 *)
  RS_Y2      : RS;         (* Flip-flop for coil Y2 *)

```

END\_VAR

(\* Programming of step sequence \*)

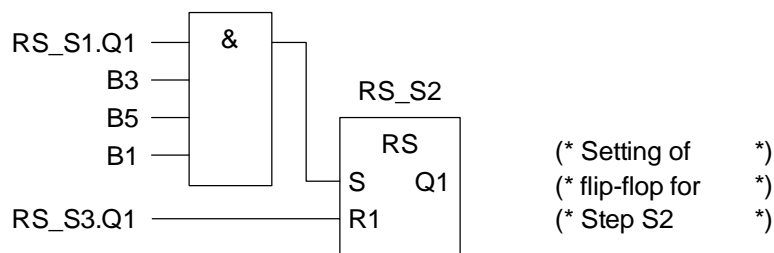
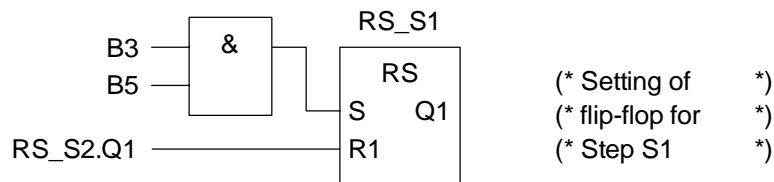
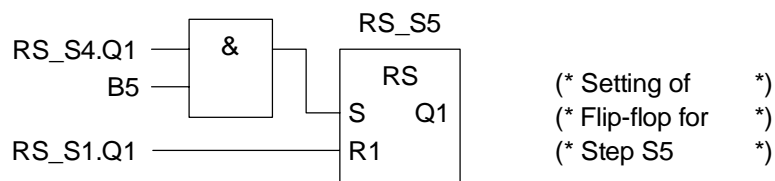
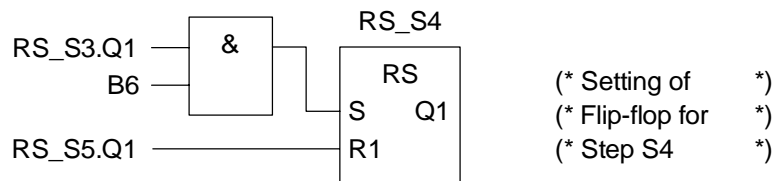
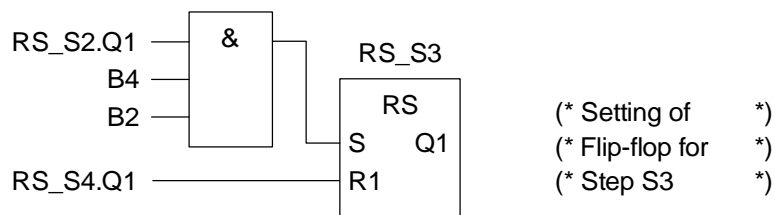


Fig. A15.5:  
 Example of a step  
 sequence with  
 RS storage elements





(\* Programming of power section \*)

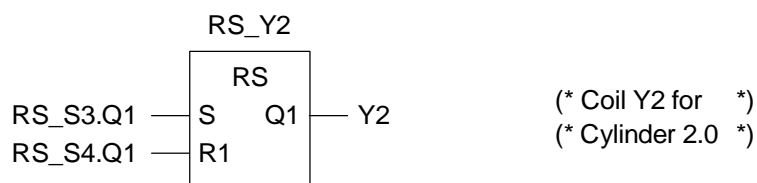
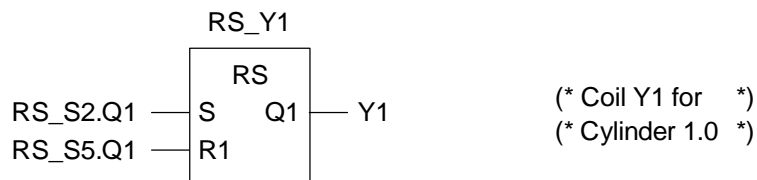


Fig. A15.5:  
 Example of a  
 step sequence with  
 RS storage elements  
 (continuation)

The programming of the step sequence requires an extension of the declaration section in fig. A15.3.

An RS flip-flop is additionally required for each step. Moreover, the statuses of coils Y1 and Y2 are stored by means of flip-flops.

The program consists of

- Step sequence
- Power section (for activation of outputs )

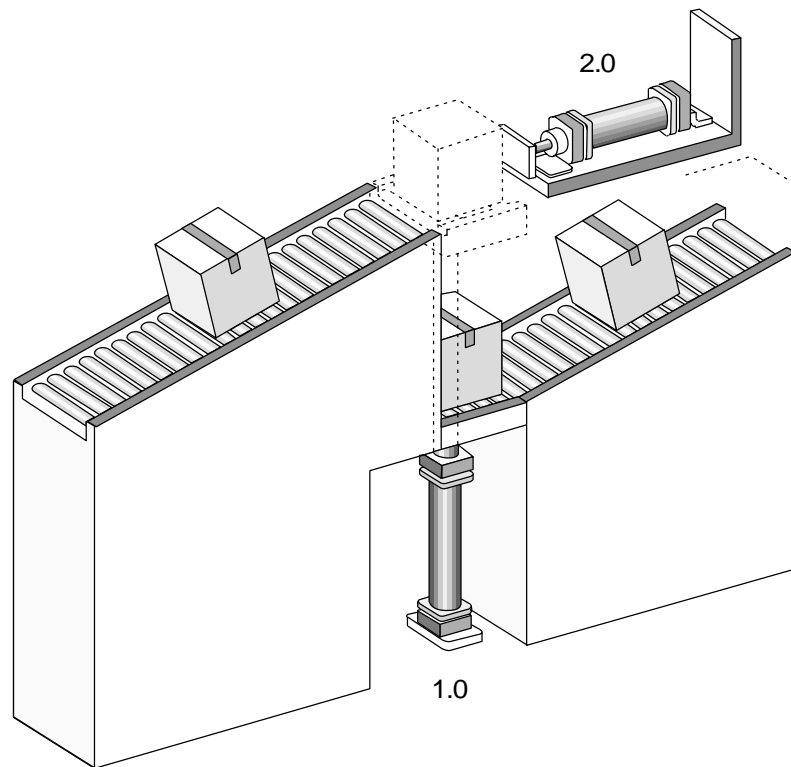
# A-138

## Exercise 15

*Problem description* A roller conveyor is monitored by a proximity switch B1 as to whether a package is present. If this is the case, the package is picked up by a cylinder 1.0 (lifting cylinder) and then transferred to another conveyor by means of cylinder 2.0 (transfer cylinder).

Cylinder 1.0 is to retract first, followed by cylinder 2.0. The cylinders are retracted and advanced by means of solenoid valves (coils Y1 and Y2). The cylinder positions are monitored by means of proximity switches B2 to B5.

On the feed side, packages have been previously arranged in such a way that they reach the lifting device individually.



*Positional sketch*

1. Drawing up and constructing the circuit diagram
2. Describing the control task in function chart to IEC 848
3. Declaration of the PLC program variables
4. Formulation of the program into a sequential function chart
  - Programming transition conditions directly in one of the languages FBD, LD or ST
  - Specifying actions as boolean actions
5. Testing and commissioning of the PLC program and system.

*Exercise definition*

### 1. Drawing up the electrical circuit diagram and constructing the circuit

*Implementation*

⇒ Completing the electrical circuit diagram on the worksheet.

⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
2	Double-acting cylinder
2	5/2-way single solenoid valve
1	Proximity switch, optical
4	Proximity switch, inductive
	Plastic tubing

*Components list*

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establishing the electrical and pneumatic connections.

### 2. Describing the control task in function chart to IEC 848

⇒ Create the program in function chart to IEC 848.

### 3. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

#### **Note**

The component parts of the declaration of variables in the exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

### 4. Formulation of the PLC program into the sequential function chart

⇒ Design the sequence structure consisting of steps and transitions.

⇒ Program the transition conditions directly in one of the languages FBD, LD or ST.

⇒ Formulate the actions associated with the steps directly as boolean actions.

⇒ Create the step structure by mapping the steps onto storage elements if the sequential function chart is not supported by your PLC.

## 5. Testing and commissioning of the PLC program and system

**Prior to** commissioning of the installation:

- Check the assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheets for pneumatic components)!

Commissioning of the installation:

- **Keep clear** of the operational parts of the installation!



⇒ Load the program to the PLC.

⇒ Carry out a function check.

⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

# A-142

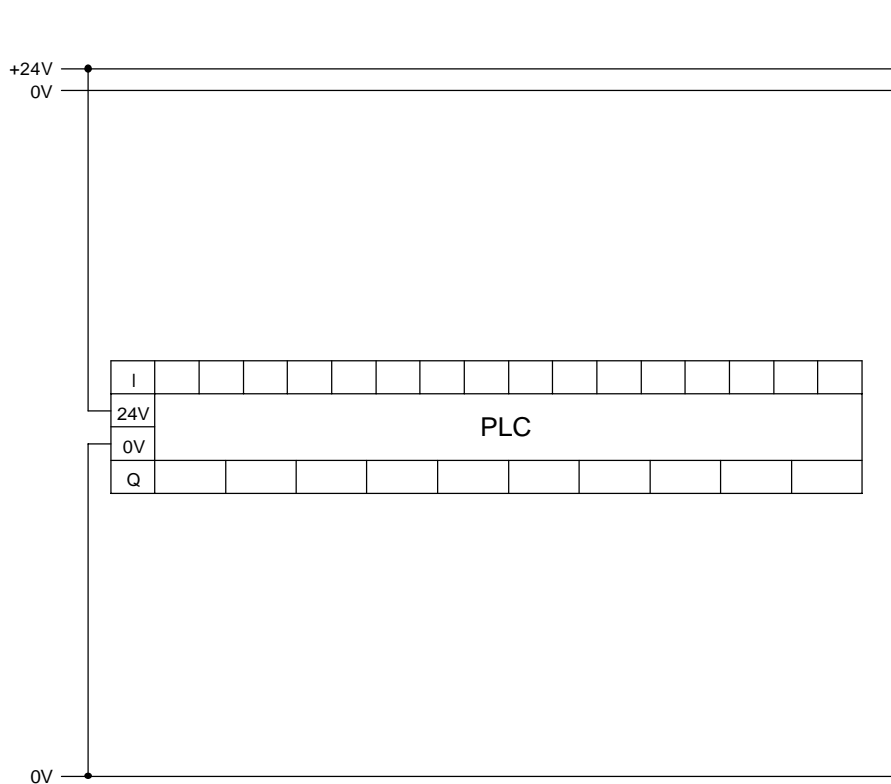
---

## Exercise 15

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



Circuit diagram, electrical

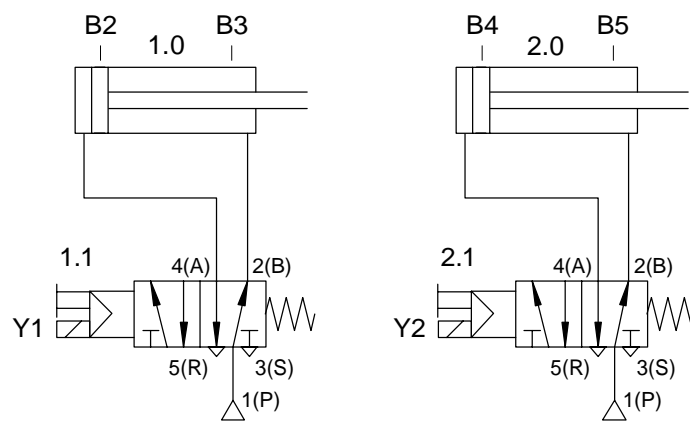
# A-144

## Exercise 15

### WORKSHEET

Configure the control system

Circuit diagram,  
electro-pneumatic



### 2. Describing the control task in function chart in accordance with IEC 848

⇒ Create the program in function chart in accordance with IEC 848.



## WORKSHEET

### 3. Declaration of the PLC program variables

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

### 4. Formulation of the PLC program into a sequential function chart

## **WORKSHEET**

*Questions* Answer the following questions:

1. What is the function of a step without associated actions ?

---

---

---

---

2. What is the response of the sequential function chart program if the action of step S3: "Cylinder 2.0 to advance" is programmed as a non-stored action ?

---

---

---

---

Programmable logic controllers

Subject

### Lifting and sorting device for packages

Title

Alternative branching

- To be able to program a sequence control system with alternative branching

Training aim

### Sequence control system with alternative branching

Technical knowledge

There are sequence control systems, where several different sequences may occur. A sequence is selected depending on the signals originating from the process applied.

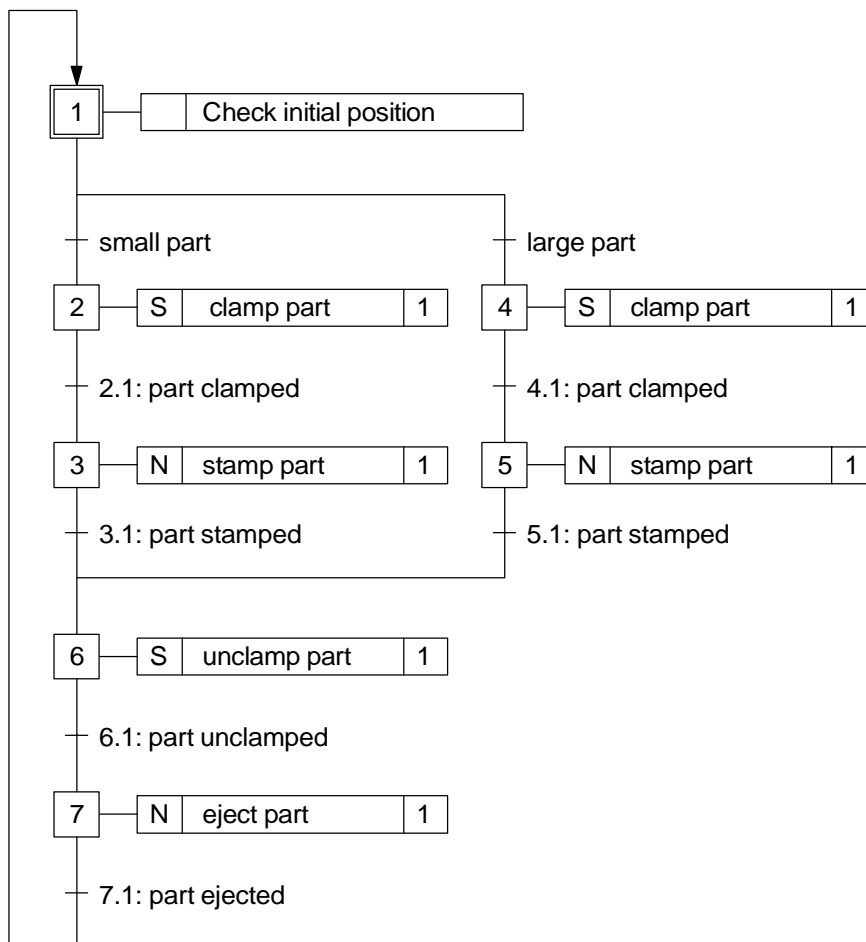


Fig. A16.1:  
Example of a sequence control system with alternative branching

A stamping tool, which stamps small or large parts by means of two different cylinders, represents an example of such a sequence control system.

Fig. A16.1 illustrates the function chart to IEC 848 for the above mentioned example.

Alternative branching is represented by as many transitions below the horizontal line as there are different sequences possible.

In order to select one option only, the transition conditions must be mutually exclusive.

Two sequences are available for selection in the example given. If a small part is detected, only steps 1, 2, 3, 6 and 7 are processed. If a large part is present, the program branches to step 4, 5, 6 and 7 after step 1.

### **Programming of a sequence control system with alternative branching**

Sequence control systems represented in function chart to IEC 848 are very easily programmed in sequential function chart. In the program listed below, the transition conditions have been formulated in the language ST .

The initial step S1 is active after the start of the program. S1 is a void step in this instance, i.e. no actions have been assigned to this step.

If all the cylinders are retracted and a small part is present, step S2 is set and step S1 reset. The part is then clamped, stamped, declamped and finally ejected.

If a large part has been detected (B1=1 and B2=1), steps S1, S4, S5, S6 and S7 are executed consecutively.

Following this, processing of the steps starts with step S1 again.

```

VAR
  B1 AT %IX1.0 : BOOL; (* small or large part *)
  B2 AT %IX1.1 : BOOL; (* for large part only *)
  B3 AT %IX2.0 : BOOL; (* cylinder 1.0 retracted *)
  B4 AT %IX2.1 : BOOL; (* cylinder 1.0 extended *)
  B5 AT %IX2.2 : BOOL; (* cylinder 2.0 retracted *)
  B6 AT %IX2.3 : BOOL; (* cylinder 2.0 extended *)
  B7 AT %IX2.4 : BOOL; (* cylinder 3.0 retracted *)
  B8 AT %IX2.5 : BOOL; (* cylinder 3.0 extended *)
  B9 AT %IX2.6 : BOOL; (* cylinder 4.0 retracted *)
  B10 AT %IX2.7 : BOOL; (* cylinder 4.0 extended *)
  Y1 AT %QX1.0 : BOOL; (* cylinder 1.0: clamping *)
  Y2 AT %QX1.1 : BOOL; (* cylinder 2.0: stamping small *)
  Y3 AT %QX1.2 : BOOL; (* cylinder 3.0: stamping large *)
  Y4 AT %QX1.3 : BOOL; (* cylinder 4.0: ejecting *)

```

END\_VAR

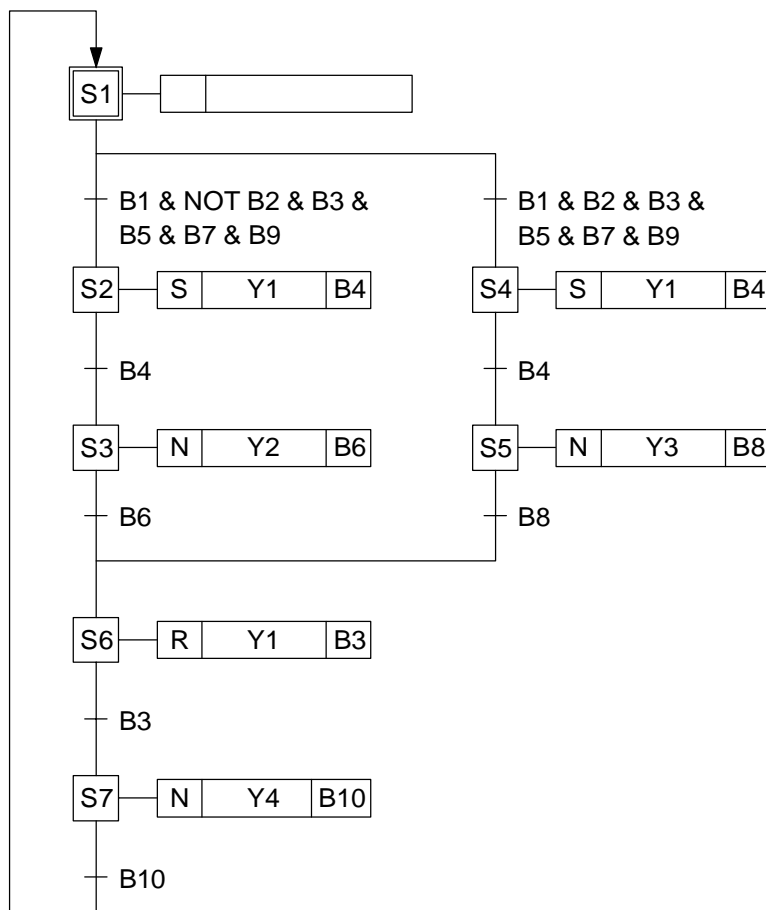


Fig. A16.2:  
Example of a sequential  
function chart  
with alternative branching

# A-150

---

## Exercise 16

In the above example, the actions for steps S3, S5 and S7 are programmed as non-stored. This is indicated by the qualifier N. The boolean variables therefore only carry a 1-signal while the associated steps are active.

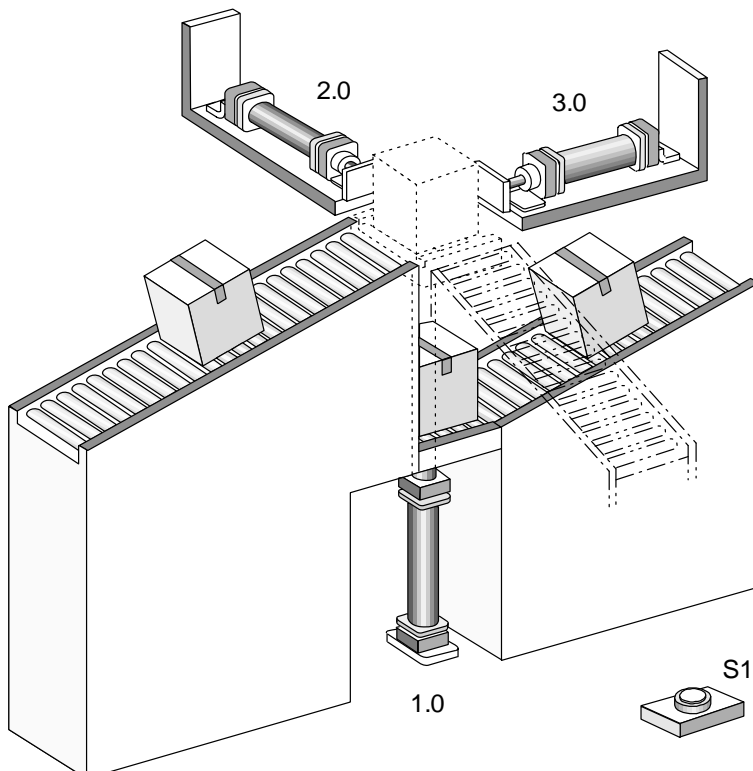
If the sequential function chart is not available for programming, the step sequences may also be generated in this instance by means of using storage elements.

Packages are conveyed past a linear measuring device on a roller conveyor in order to establish their size. There are two different package sizes: Short and long packages. The linear measuring device supplies an 0-signal for a short package and a 1-signal for a long package.

Following this, the packages reach a lifting platform. The sequence is started by means of START button S1. The packages are lifted by a lifting cylinder 1.0. The packages are then sorted: Short packages are transferred to a second conveyor via transfer cylinder 2.0, and long packages onto a third conveyor via cylinder 3.0. Lifting cylinder 1.0 is to retract again only after cylinders 2.0 or 3.0 have reached their end position.

The cylinder positions are monitored by means of proximity switches B1 to B6. Cylinder 1.0 is advanced and retracted by means of a double solenoid valve via coils Y1 and Y2. Cylinders 2.0 and 3.0 are advanced and retracted by means of solenoid valves (coils Y3 and Y4).

### Problem description



Positional sketch

# A-152

## Exercise 16

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Describing the control task in function chart to IEC 848
  3. Declaration of the PLC program variables
  4. Formulation of the program into a sequential function chart
    - Programming the transition conditions directly in one of the languages LD or ST
    - Specifying actions as boolean actions
  5. Testing and commissioning of the PLC program and system.

*Implementation*

### 1. Drawing up the electrical circuit diagram and constructing the circuit

- ⇒ Complete the electrical circuit diagram on the worksheet.
- ⇒ Assemble the required equipment on the slotted profile plate:

<i>Quantity</i>	<i>Designation</i>
1	Programmable logic controller
1	Interconnecting cable for connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull connector
1	Single-acting cylinder
2	Double-acting cylinder
2	5/2-way single solenoid valve
1	5/2-way double solenoid valve
1	Signal input, electrical
1	Proximity switch, capacitive
1	Proximity switch, optical
4	Proximity switch, inductive
	Plastic tubing

*Components list*



Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

## 2. Describing the control task in function chart to IEC 848

⇒ Create the program in function chart to IEC 848.

## 3. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC programming system used.

## 4. Formulation of the PLC program in sequential function chart

⇒ Design the sequence structure consisting of steps and transitions.

⇒ Program the transition conditions directly into one of the languages FBD, LD or ST.

⇒ Formulate the actions associated with the steps directly as boolean actions.

⇒ Create the step structure by mapping the steps onto a storage element, if the sequential function chart is not supported by your PLC.

## 5. Testing and commissioning of the PLC program and system



**Prior to** commissioning of installation:

- Check the assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC !
- Increase air supply on service unit to operating pressure (see data sheets for pneumatic components)!

Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

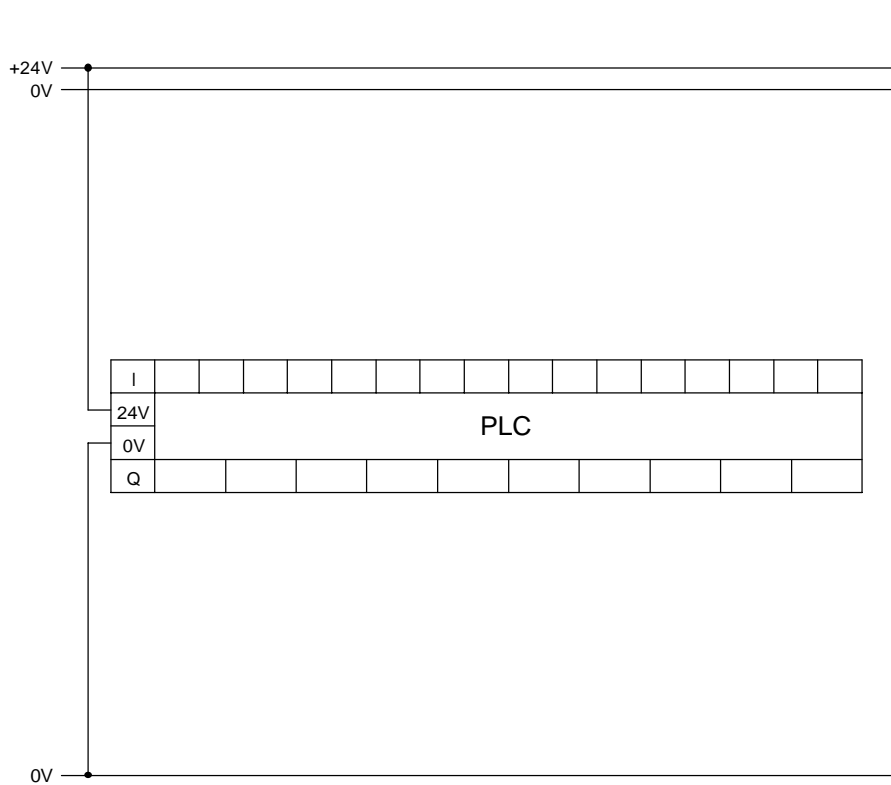
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



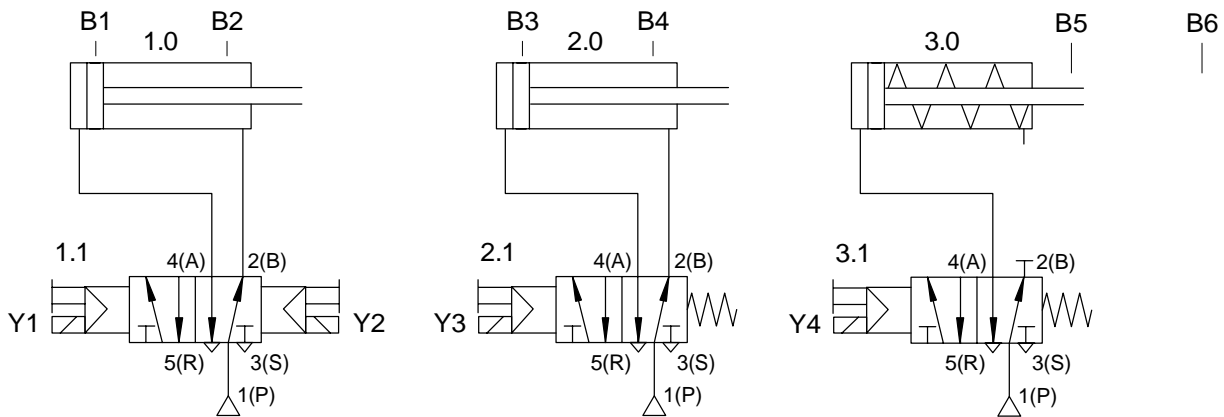
Circuit diagram, electrical

# A-156

## Exercise 16

### WORKSHEET

Configure the control system



Circuit diagram,  
electro-pneumatic

## 2. Describe the control task in function chart to IEC 848

⇒ Create the program in function chart to IEC 848.

**Questions** Answer the following questions:

1. What is the sorting criteria according to which the packages are evaluated?

---



---



---



---

2. How do you ensure that just one sequence step is selected during program execution ?

---



---



---



---

**WORKSHEET**

**3. Declaration of the PLC program variables**

Declare the variables required in the PLC program:

<i>Designation</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>

*Declaration of variables*

**4. Formulate the PLC program in sequential function chart**

## WORKSHEET

*Question* Answer the following question:

3. Specify the transition condition, which is always true. Why are such transition conditions formulated ?

---

---

---

---

Programmable logic controllers

*Subject*

### Stamping device with counter

*Title*

Counting cycles

- To be able to realise counting cycles by means of using the standard function modules CTU or CTD

*Training aim*

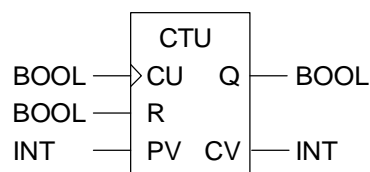
Counting cycles form part of the basic operations of a PLC. IEC 1131-3 defines the three standard function blocks CTU (incremental counter), CTD (decremental counter) and CTUD (incremental/decremental counter) for the realisation of these operations.

*Technical knowledge*



### Function block CTU, incremental counter

Function block CTU (fig. A17.1) realises an incremental counter. Its interface is defined by means of three input and two output parameters.



*Fig. A17.1:  
Function block CTU*

The characteristic behaviour of an incremental counter is as follows:

- A 1-signal at reset input R sets the current counter status CV at 0.
- Thereafter, the value CV of the counter is increased by 1 with each rising edge at input CU.
- A 1-signal applies at output Q as soon as the current value CV is equal or greater than the preselect value PV. Output Q carries a 0-signal as long as current counter status CV is less than the preselect value PV.

# A-160

## Exercise 17

### Function block CTD, Decremental counter

Function block CTD (fig. A17.2) being a decremental counter is counterpart to function block CTU.

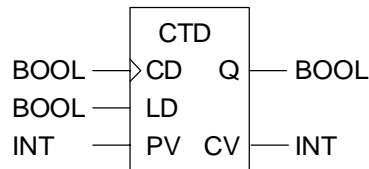


Fig. A17.2:  
Function block CTD

The behaviour of a decremental counter is as follows:

- A 1-signal at the LD input sets the current counter status CV equal to the preselect value PV.
- Each rising edge at the CD input decreases the current counter status CV by 1.
- Output Q carries a 0-signal as long as the current counter status CV is greater than 0. Only if the current value is less or equal to 0, does a 1-signal apply at output Q.

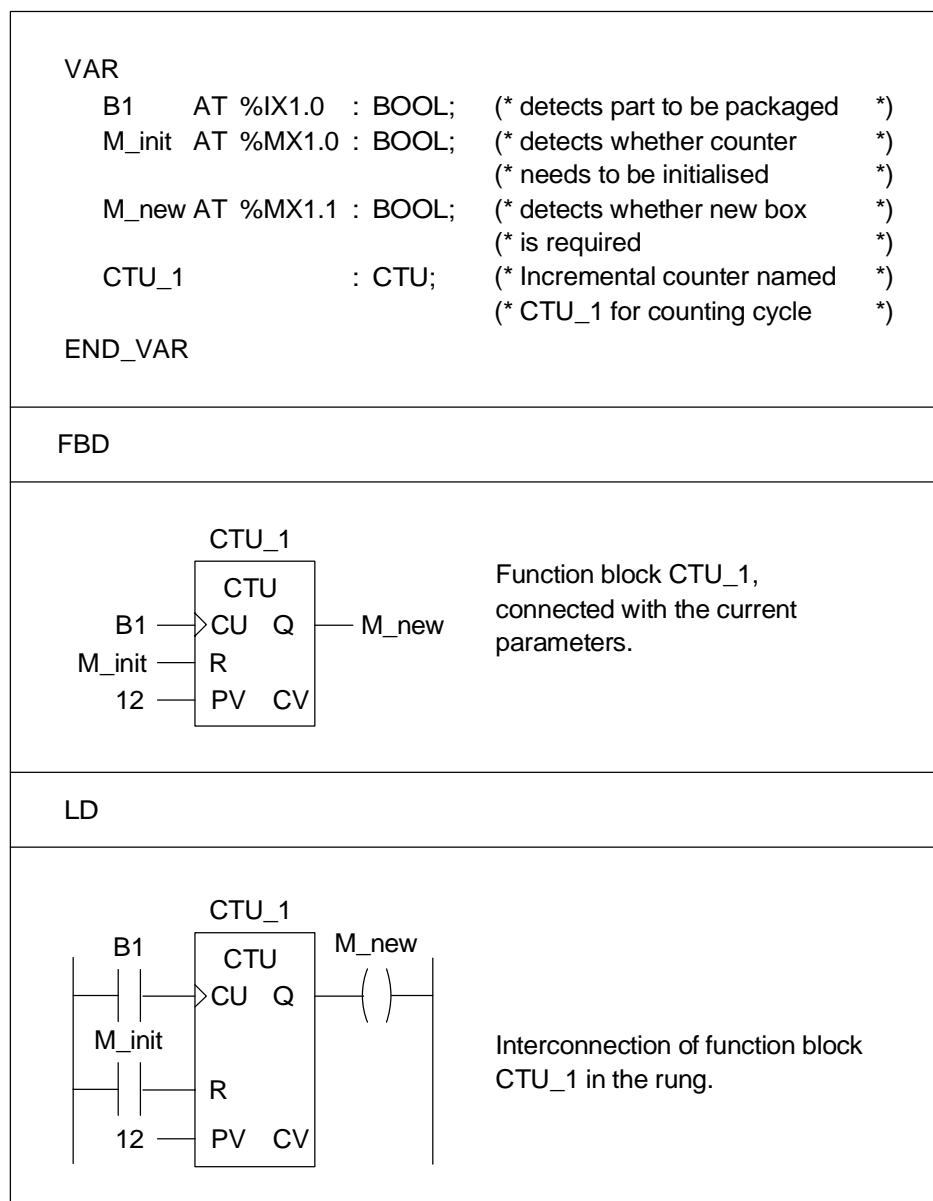


### Use of function block CTU in the individual programming languages

The use of the function block is demonstrated on a small packaging task.

12 parts each are to be packed into one box. When a box has been filled, another is made available. Each packaging cycle is triggered via a memory M\_init. The individual parts are detected by means of a sensor B1. The status of the counter is mapped onto a memory M\_new.

*Example*



*Table A17.1:  
Application of an  
incremental counter*

# A-162

## Exercise 17

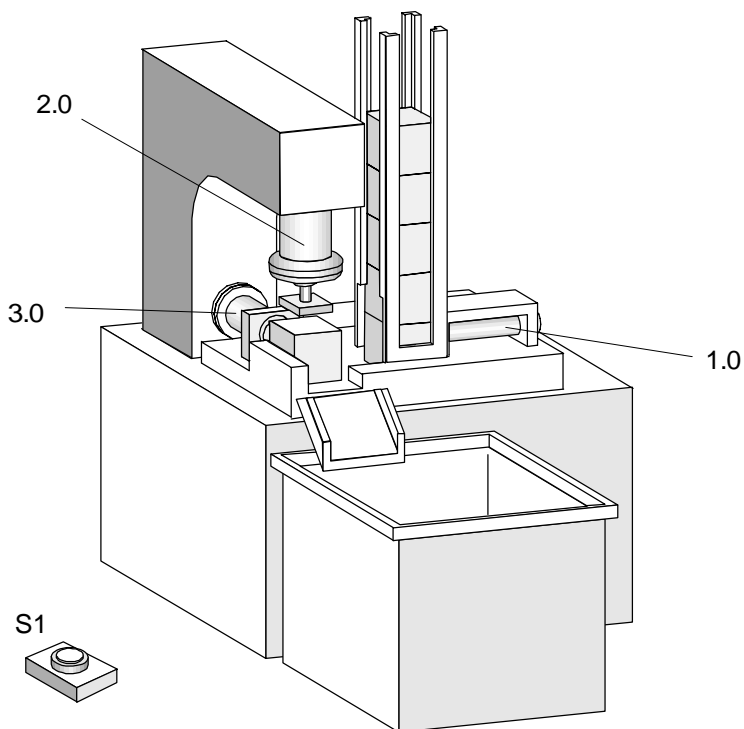
IL		
CAL	CTU_1 (CU := B1, R := M_init, PV := 12)	Invocation of function block CTU_1
LD	CTU_1.Q	Reading of output Q of CTU_1
ST	M_new	Storage of current results to M_new
ST		
	CTU_1 (CU := B1, R := M_init, PV := 12);	Invocation of function block CTU_1
	M_new := CTU_1.Q;	Assignment of output Q of CTU_1 to M_new

Table A17.1:  
(Continuation)

10 parts at a time are stamped on a machine. The program cycle is started by means of a push button S1. The proximity switch B7 signals "Part in magazine". A part is fed towards the machine by means of a cylinder 1.0 and clamped. It is then stamped via cylinder 2.0, and subsequently ejected by means of an ejecting cylinder 3.0.

### *Problem description*

The clamping cylinder 1.0 operates via a double solenoid valve with two coils Y1 (clamping) and Y2 (unclamping). Cylinders 2.0 and 3.0 are powered by spring-return solenoid valves with the coils Y3 and Y4. The cylinder positions are monitored by means of the proximity switches B1 to B6.



*Positional sketch*

# A-164

## Exercise 17

- Exercise definition*
1. Drawing up and constructing the circuit diagram
  2. Describing the control task in function chart to IEC 848
  3. Declaration of the program variables
  4. Formulation of the program in sequential function chart
    - Programming the transition conditions in one of languages FBD, LD or ST
    - Specifying the actions
  5. Testing and commissioning of the PLC program and system.

*Implementation*

### 1. Drawing up the electrical circuit diagram and constructing the circuit

- ⇒ Complete the electrical circuit diagram on the worksheet.
- ⇒ Assemble the required equipment on the slotted assembly board:

<i>Quantity</i>	<i>Description</i>
1	Programmable logic controller
1	Interconnecting cable of connection unit
1	Connection unit
1	Service unit
1	Manifold
1	Quick push-pull distributor
1	Single-acting cylinder
2	Double-acting cylinder
2	5/2-way single solenoid valve
1	5/2-way double solenoid valve
1	Signal input, electrical
1	Proximity switch, capacitive
1	Proximity switch, optical
1	Proximity switch, inductive
4	Proximity switch, inductive
	Plastic tubing

*Components list*

Prior to wiring and tubing:

- Switch off power supply!
- Switch off air supply at service unit!



⇒ Establish the electrical and pneumatic connections.

## 2. Describing the control task in function chart to IEC 848

⇒ Create the program in function chart to IEC 848.

## 3. Declaration of the PLC program variables

⇒ All variables are to be created as program-local variables.

⇒ Specify only those parts of the declaration required for your PLC application. These are: Designation, data type, address – only if directly addressed variables are used – and variables comment.

### Note

The component parts of the declaration of variables in this exercise section are represented in tabular form. If actual PLC systems are used, the input and representation of the variables declaration is dependent on the PLC program system used.

## 4. Formulation of the PLC program in sequential function chart

⇒ Design the sequence structure consisting of steps and transitions.

⇒ Program the transition conditions directly in one of the languages FBD, LD or ST.

⇒ Formulate the actions associated with the steps. For actions consisting of more than once boolean variable, it is mandatory to input an action name.

⇒ Create the step structure by mapping the steps onto storage elements if the sequential function chart is not supported by your PLC.

### 5. Testing and commissioning of the PLC program and system



**Prior to** commissioning of the installation:

- Check the assembled circuit with the help of the circuit diagrams!

Commissioning of the installation:

- Switch on power supply using a standard voltage of 24 V DC!
- Increase air supply on service unit to operating pressure (see data sheets for pneumatic components)!

Operation of the installation:

- **Keep clear** of the operational parts of the installation!

⇒ Load the program to the PLC.

⇒ Carry out a function check.

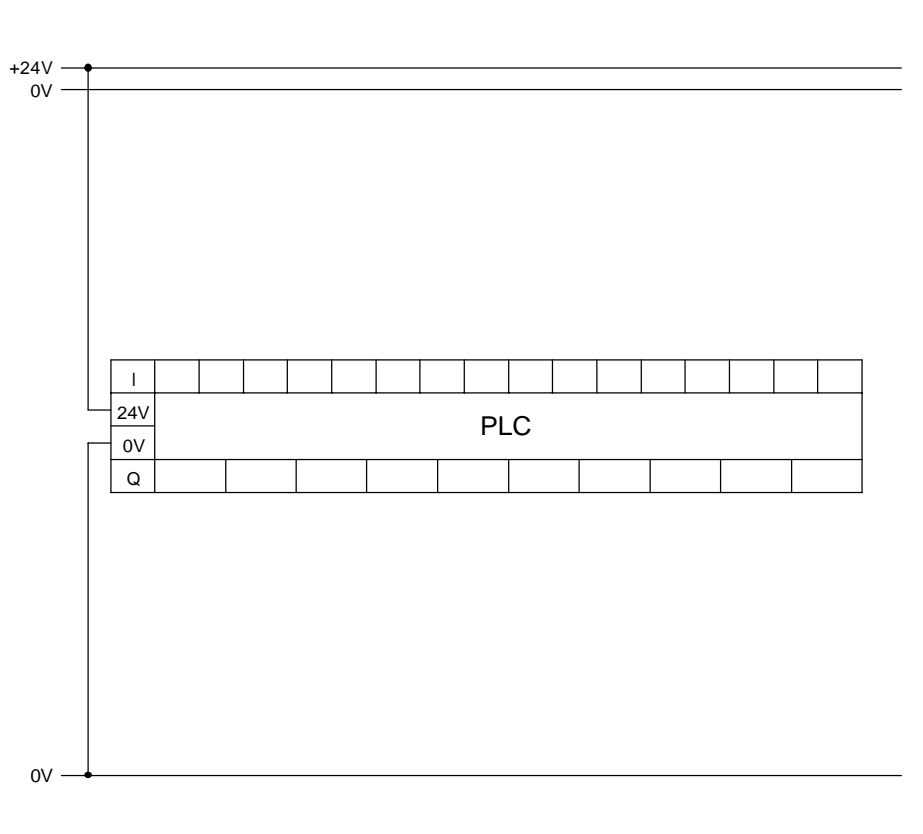
⇒ Correct any errors occurring in the PLC program.

⇒ Document your solution.

## WORKSHEET

### 1. Drawing up the electrical circuit diagram and constructing the circuit

Complete the electrical circuit diagram and enter the input and output addresses available for your PLC.



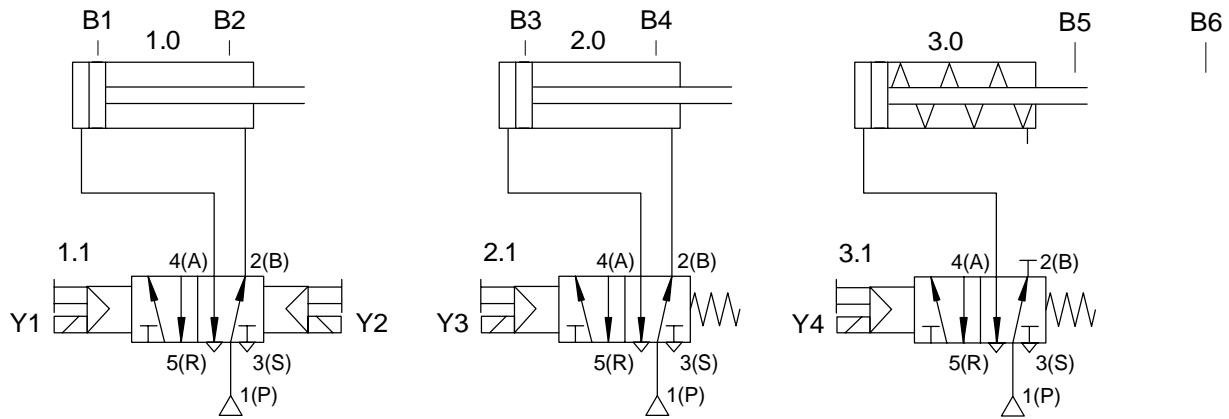
Circuit diagram, electrical

# A-168

## Exercise 17

### WORKSHEET

Configure the control system.



Circuit diagram,  
electro-pneumatic

## 2. Describing the control task in function chart to IEC 848

⇒ Create the program in function chart to IEC 848.





## **WORKSHEET**

4. Formulate the PLC program into a sequential function chart

## WORKSHEET

Answer the following question:

Questions

1. When does the status of a counter change?

---

---

---

---

# A-172

---

## Exercise 17

## **Section C – Solutions**

### **Components of a programmable logic controller**

- Solution 1: **Design and commissioning of a programmable logic controller**  
Components of a PLC..... C-3

### **Programming to IEC 1131**

- Exercise 2: **From problem to solution – taking into consideration IEC 1131-3**  
Practical steps for PLC programming..... C-5

### **Basic logic operations**

- Solution 3: **Lamp circuit**  
The assignment function..... C-7
- Solution 4: **Burglar alarm**  
The NOT function ..... C-9
- Solution 5: **Press with protective guard**  
The AND function ..... C-11
- Solution 6: **Bell system**  
The OR function ..... C-13

### **Logic control systems without latching properties**

- Solution 7: **Stamping device**  
Combination of AND/OR/NOT ..... C-15
- Solution 8: **Silo control system for two bulk materials**  
Combination circuit with branching..... C-19

### **Logic control systems with latching properties**

- Solution 9: **Fire alarm**  
Setting an output..... C-21
- Solution 10: **Drill breakage monitoring**  
Setting and resetting an output..... C-23
- Solution 11: **Activating a cylinder**  
Signal edges ..... C-25

## Logic control systems with time response

Solution 12: <b>Bonding of components</b>	
Pulse .....	C-29
Solution 13: <b>Embossing device</b>	
Switch-on signal delay .....	C-31
Solution 14: <b>Clamping device</b>	
Switch-off signal delay .....	C-35

## Sequence control systems

Solution 15: <b>Lifting device for packages</b>	
Linear sequence .....	C-39
Solution 16: <b>Lifting and sorting device for packages</b>	
Alternative branching .....	C-43
Solution 17: <b>Stamping device with counter</b>	
Counting cycles .....	C-47

## Design and commissioning of a programmable logic controller

### Components of a PLC

Title

#### 1. Components of a PLC

*What are the basic components of a programmable logic controller?*

Question 1

The basic components of a PLC are:

Answer

- the main processing unit, formerly (central control unit)
- the input modules
- the output modules
- the program memory
- the PLC program

*What are the basic modules making up the main processing unit of a programmable logic controller?*

Question 2

The basic modules of an MPU are:

Answer

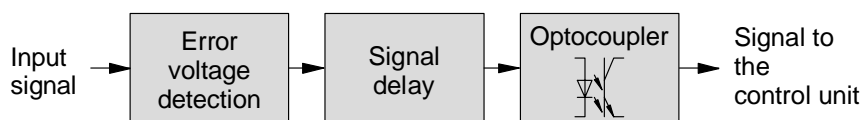
- the control unit
- the data memory
- the arithmetic and logic unit (ALU)

*How is electrical isolation achieved between sensor/actuator signals and the PLC?*

Question 3

The sensor/actuator signals and the PLC are electrically isolated via an optocoupler. The main processing unit is thus separated from the external circuit of the sensors and actuators. Interferences in these circuits therefore cannot damage the controller.

Answer



Block diagram of an input module

## 2. Design and commissioning of your selected PLC

The following table lists the technical data of a Festo FPC 101B programmable logic controller as an example.

<b>Operating voltage</b>	
Nominal voltage	24 V DC
Permissible voltage range	16 to 30 V DC
Current consumption	approx. 160 mA
<b>Inputs</b>	
Number	21 of which 1 is a counter input
Input current	6 mA
Input level	log. 0 = 0 to 5 V log. 1 = 11 to 30 V
<b>Outputs</b>	
Number	14 Transistor outputs
Type	positive switching
Output voltage	Operating voltage – 2 V
Output current	max. 300 mA/output Total output current max. 2.5 A

Technical data



**From problem to solution – taking into consideration  
IEC 1131-3**  
PLC programming procedures

*Title*

### **1. Practical steps for creating a PLC program**

*List the five practical steps for creating a PLC program.*

The five steps to create a PLC program are:

- Description of the control task
- Planning a solution
- Implementing the solution
- Testing and commissioning the control system
- Control system documentation

*What activities are carried out in the step "Implementation of the" solution"?*

*Question 1*

The following activities are carried out in this step:

*Answer*

- PLC configuration
- Declaration of the PLC program variables
- Formulation of the PLC program logic into one of the languages LD, FBD, IL, ST or SFC

### **2. Resources of a PLC in accordance with IEC 1131-3**

*The following resources are to be addressed directly. Specify the designations in accordance with IEC 1131-3:*

Input bit 14	%IX14 or %I14
Memory 9	%MX9 or %M9
Output word 3	%QW3
Input 7 on 2nd input card	%I2.7

### 3. Declaration of variables to IEC 1131-3

*The following data must be taken into consideration in a program declaration. Use the appropriate data type in your declaration. The declaration is to be valid locally only.*

- *Input for a switch S1, applied to input 2 of the 4th input card*
- *Temperature TEMP, applied to output word No. 1*
- *Memory VALVE\_OPEN*
- *boolean memory with identifier PART\_PRESENT, preallocated initial value 0*
- *boolean memory with identifier ROBOT\_INIT, preassigned initial value 1*
- *Storage of one number (INT) under the name NUMBER, preassigned the value 0*

```
VAR
  S1 AT %I4.2           :BOOL;
  TEMP AT %QW1         :INT;
  VALVE_OPEN           :BOOL;
  PART_PRESENT         :BOOL:=0;
  ROBOT_INIT           :BOOL:=1;
  NUMBER               :INT:=0;
END_VAR
```

### Lamp circuit

The assignment function

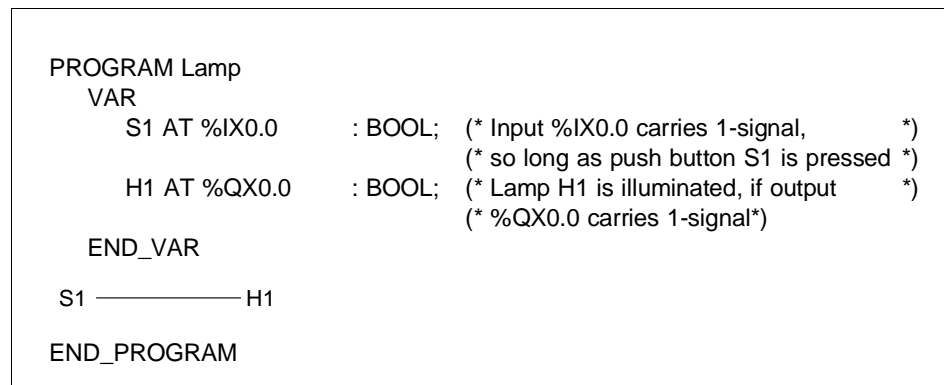
Title

### 3. Declaration of variables of a PLC program

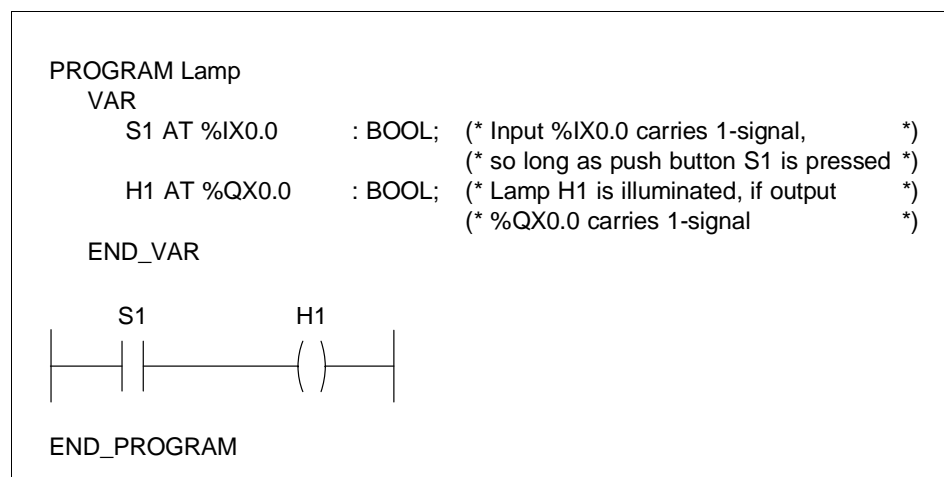
Name	Data type	Address	Comment
S1	BOOL	%IX0.0	Input % IX0.0 carries 1-signal, so long as push button S1 is pressed
H1	BOOL	%QX0.0	Lamp H1 is illuminated, if output % QX0.0 carries 1-signal

### 4. Formulation of the PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of the program descriptor – this also includes the declaration of variables – and main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



*Function block diagram*



*Ladder diagram*

# C-8

## Solution 3

---

*Instruction list*

```
PROGRAM Lamp
VAR
  S1 AT %IX0.0      : BOOL; (* Input %IX0.0 carries 1-signal, *)
                    (* so long as push button S1 is pressed *)
  H1 AT %QX0.0      : BOOL; (* Lamp H1 is illuminated, if output *)
                    (* %QX0.0 carries 1-signal *)
END_VAR

LD   S1
ST   H1

END_PROGAM
```

*Structured text*

```
PROGRAM Lamp
VAR
  S1 AT %IX0.0      : BOOL; (* Input %IX0.0 carries 1-signal, *)
                    (* so long as push button S1 is pressed *)
  H1 AT %QX0.0      : BOOL; (* Lamp H1 is illuminated, if output *)
                    (* %QX0.0 carries 1-signal *)
END_VAR

H1 := S1;

END_PROGRAM
```

## Burglar alarm

The NOT function

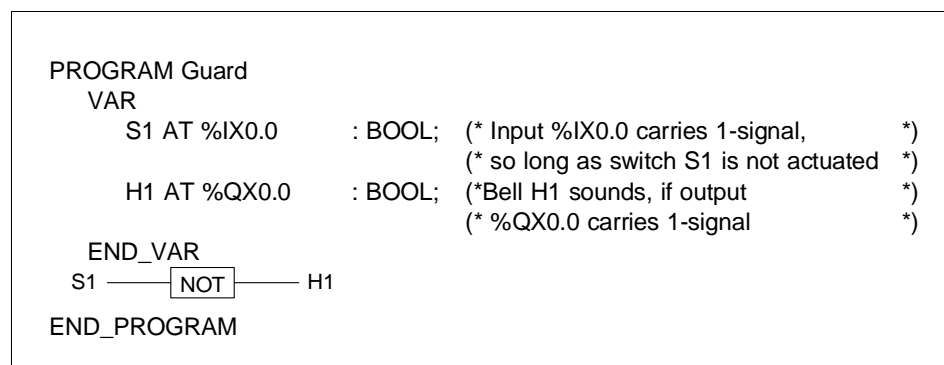
Title

### 3. Declaration of PLC program variables

Name	Data type	Address	Comment
S1	BOOL	%IX0.0	Input %IX0.0 carries 1-signal, so long as switch S1 is not actuated
H1	BOOL	%QX0.0	Bell H1 sounds, if output %QX0.0 carries 1-signal

### 4. Formulation of PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



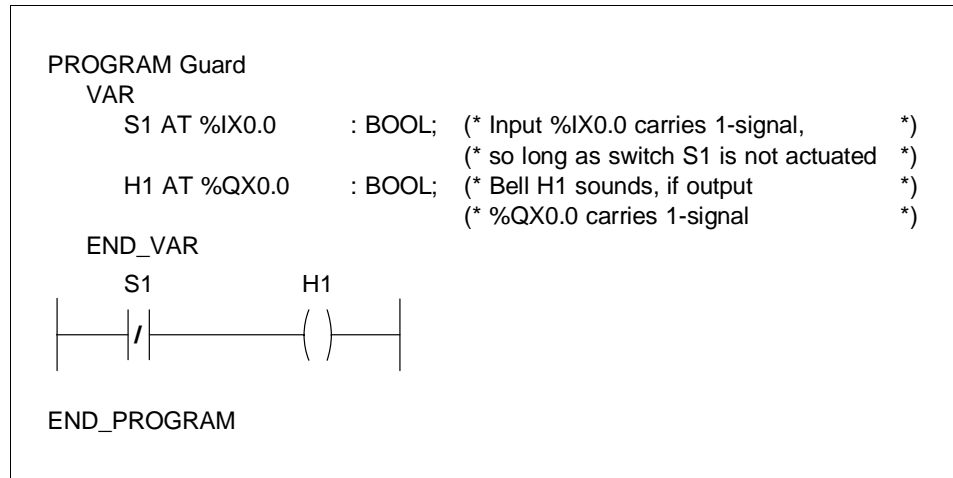
Function block diagram

If the value of a boolean variable is to be directly mapped negated onto another variable, this can only be done via the NOT function. The negating of a boolean variable via the character "o" is only possible directly at inputs or outputs of functions or function blocks.

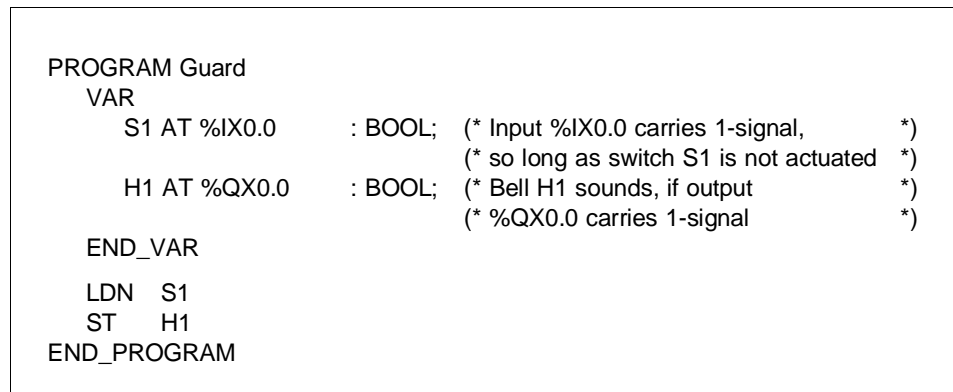
# C-10

## Solution 4

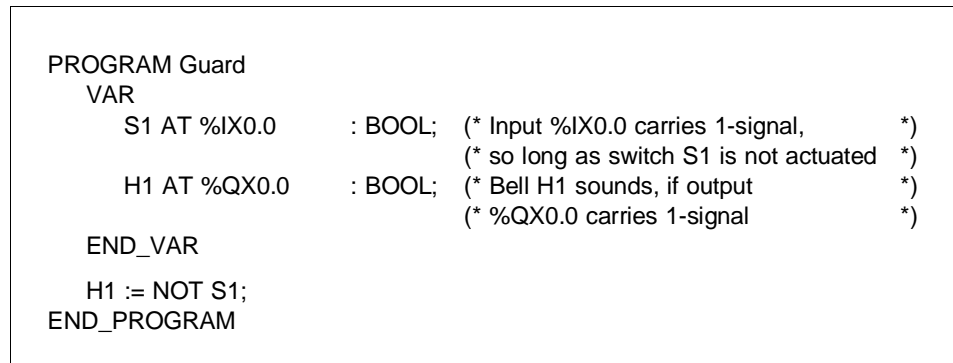
Ladder diagram



Instruction list



Structured text



**Press with protective guard**  
The AND function

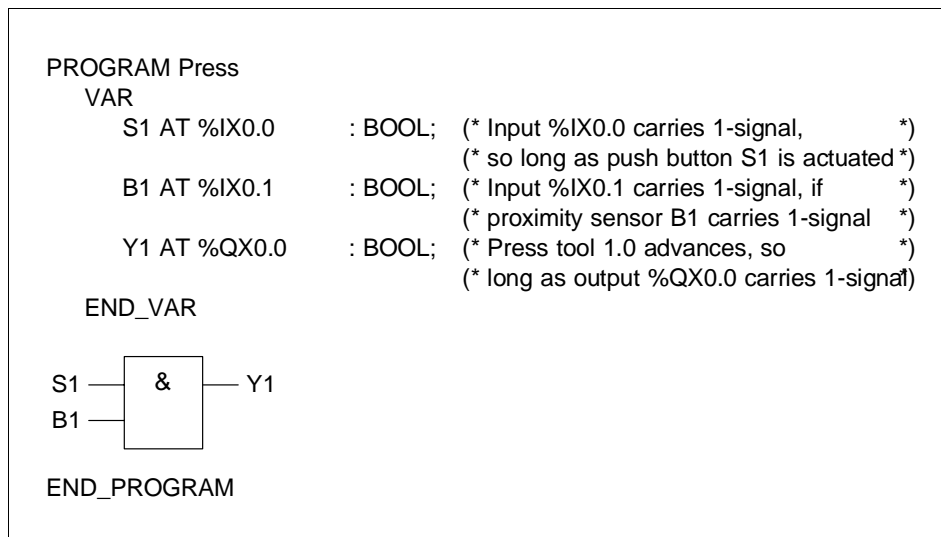
Title

### 3. Declaration of PLC program variables

<i>Name</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>
S1	BOOL	%IX0.0	Input %IX0.0 carries 1-signal, so long as push button S1 is actuated
B1	BOOL	%IX0.1	Input %IX0.1 carries 1-signal, if proximity sensor B1 carries 1-signal
Y1	BOOL	%QX0.0	Press tool 1.0 advances, so long as output %QX0.0 carries 1-signal

### 4. Formulation of PLC program in the various PLC programming languages

According to the IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.

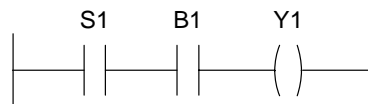


Function block diagram

# C-12

## Solution 5

```
PROGRAM Press
VAR
  S1 AT %IX0.0      : BOOL; (* Input %IX0.0 carries 1-signal, *)
                    (* so long as push button S1 is actuated *)
  B1 AT %IX0.1      : BOOL; (* Input %IX0.1 carries 1-signal, if *)
                    (* proximity sensor B1 carries 1-signal *)
  Y1 AT %QX0.0      : BOOL; (* Press tool 1.0 advances, so long as *)
                    (* output %QX0.0 carries 1-signal *)
END_VAR
```



```
END_PROGRAM
```

Ladder diagram

```
PROGRAM Press
VAR
  S1 AT %IX0.0      : BOOL; (* Input %IX0.0 carries 1-signal, *)
                    (* so long as push button S1 is actuated *)
  B1 AT %IX0.1      : BOOL; (* Input %IX0.1 carries 1-signal, if *)
                    (* proximity sensor B1 carries 1-signal *)
  Y1 AT %QX0.0      : BOOL; (* Press tool 1.0 advances, so long as *)
                    (* output %QX0.0 carries 1-signal *)
END_VAR
```

```
LD   S1
AND  B1
ST   Y1
END_PROGRAM
```

Instruction list

```
PROGRAM Press
VAR
  S1 AT %IX0.0      : BOOL; (* Input %IX0.0 carries 1-signal, *)
                    (* so long as push button S1 is actuated *)
  B1 AT %IX0.1      : BOOL; (* Input %IX0.1 carries 1-signal, if *)
                    (* proximity sensor B1 carries 1-signal *)
  Y1 AT %QX0.0      : BOOL; (* Press tool 1.0 advances, so long as *)
                    (* output %QX0.0 carries 1-signal *)
END_VAR
```

```
Y1 := S1 & B1;
END_PROGRAM
```

Structured text



**Bell system**  
The OR function

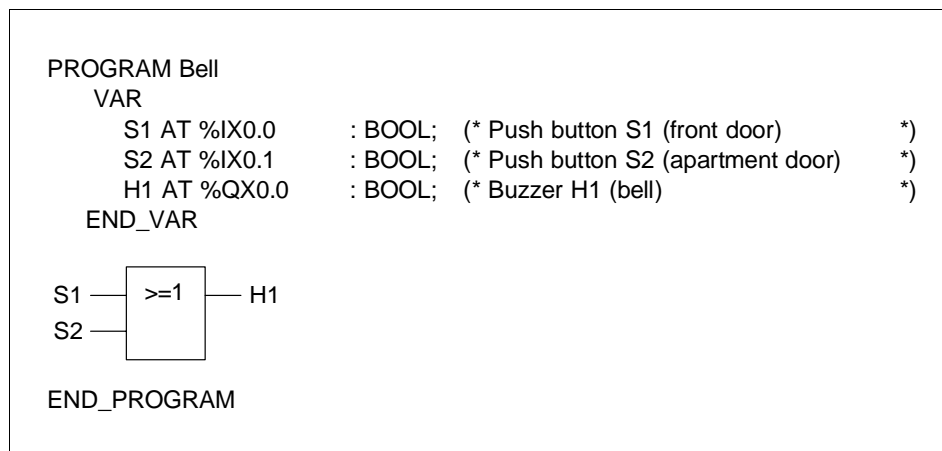
*Title*

### 3. Declaration of PLC program variables

<i>Name</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>
S1	BOOL	%IX0.0	Push button S1 (front door)
S2	BOOL	%IX0.1	Push button S2 (apartment door)
H1	BOOL	%QX0.0	Buzzer H1 (bell)

### 4. Formulation of the PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



*Function block diagram*

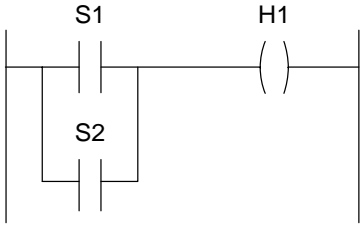
# C-14

## Solution 6

```
PROGRAM Bell
VAR
  S1 AT %IX0.0      : BOOL; (* Push button S1 (front door)      *)
  S2 AT %IX0.1      : BOOL; (* Push button S2 (apartment door)   *)
  H1 AT %QX0.0      : BOOL; (* Buzzer H1                        *)
END_VAR

  S1
  S2
  H1

END_PROGRAM
```



Ladder diagram

```
PROGRAM Bell
VAR
  S1 AT %IX0.0      : BOOL; (* Push button S1 (front door)      *)
  S2 AT %IX0.1      : BOOL; (* Push button S2 (apartment door)   *)
  H1 AT %QX0.0      : BOOL; (* Buzzer H1                        *)
END_VAR

LD   S1
OR   S2
ST   H1

END_PROGRAM
```

Instruction list

```
PROGRAM Bell
VAR
  S1 AT %IX0.0      : BOOL; (* Push button S1 (front door)      *)
  S2 AT %IX0.1      : BOOL; (* Push button S2 (apartment door)   *)
  H1 AT %QX0.0      : BOOL; (* Buzzer H1                        *)
END_VAR

H1 := S1 OR S2;
END_PROGRAM
```

Structured text

**Stamping device**  
Combination of AND/OR/NOT

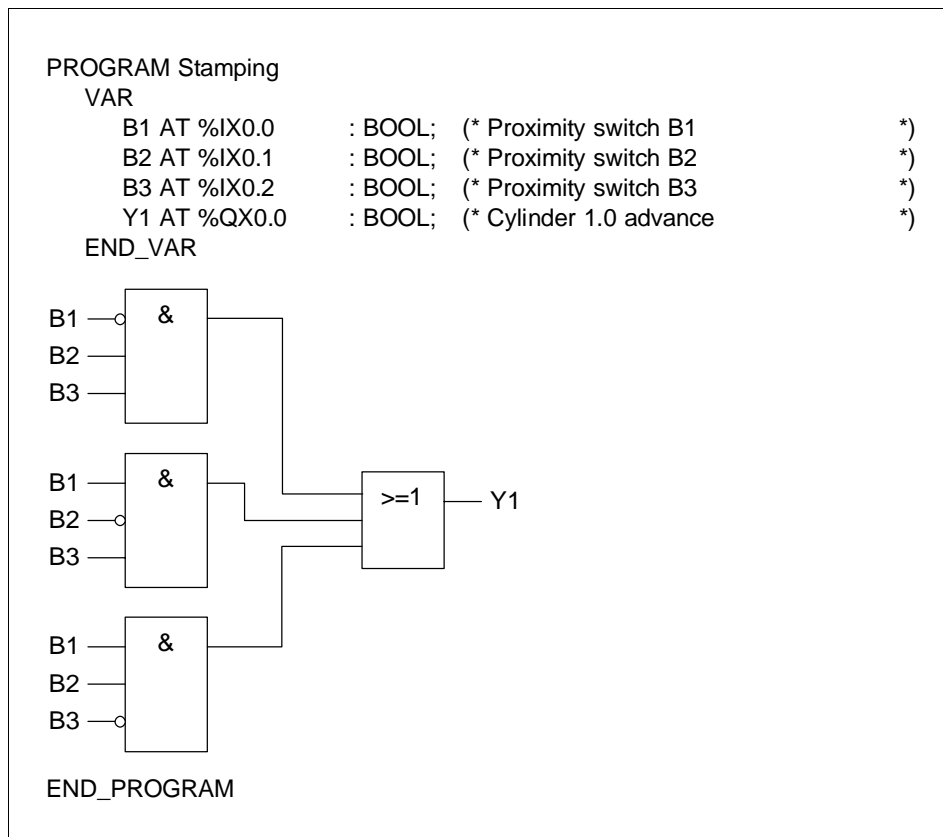
Title

### 3. Declaration of variables

Name	Data type	Address	Comment
B1	BOOL	%IX0.0	Proximity switch B1
B2	BOOL	%IX0.1	Proximity switch B2
B3	BOOL	%IX0.2	Proximity switch B3
Y1	BOOL	%QX0.0	Cylinder 1.0 advance

### 4. Formulation of PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram



```
PROGRAM Stamping
VAR
  B1 AT %IX0.0      : BOOL; (* Proximity switch B1      *)
  B2 AT %IX0.1      : BOOL; (* Proximity switch B2      *)
  B3 AT %IX0.2      : BOOL; (* Proximity switch B3      *)
  Y1 AT %QX0.0      : BOOL; (* Cylinder 1.0 advance     *)
END_VAR

Y1 := ( NOT B1 & B2 & B3 ) OR ( B1 & NOT B2 & B3 )
      OR ( B1 & B2 & NOT B3 );

END_PROGRAM
```

*Structured text*

The parenthesising of AND expressions is not a mandatory requirement, since an AND operation has a higher priority than an OR operation. The use of parenthesis, however, makes it easier and quicker to understand a complex expression.

# C-18

---

Solution 7

**Silo control system for two bulk materials**  
Logic control system with branching

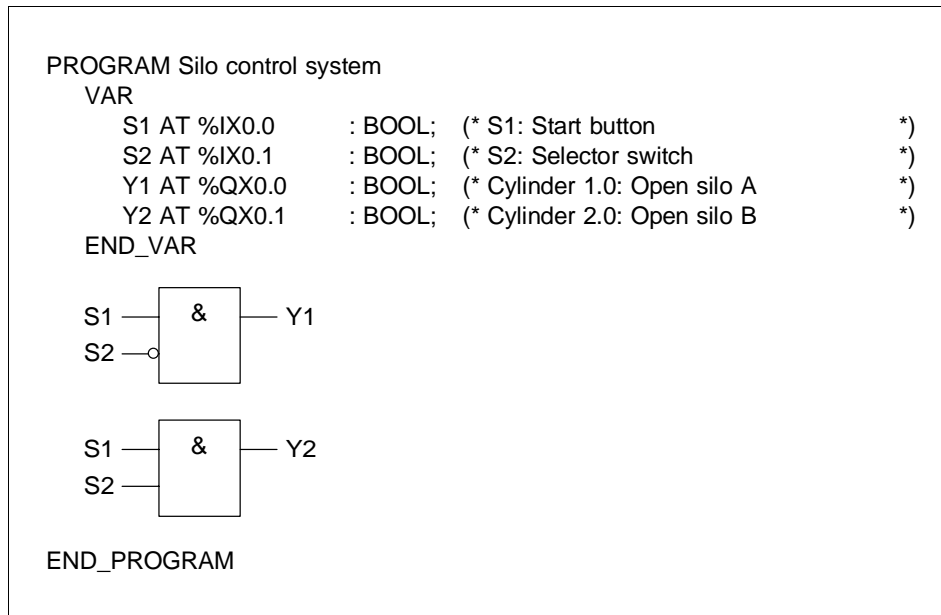
Title

**3. Declaration of variables**

<i>Name</i>	<i>Data type</i>	<i>Address</i>	<i>Comment</i>
S1	BOOL	%IX0.0	S1: Start button
S2	BOOL	%IX0.1	S2: Selector switch
Y1	BOOL	%QX0.0	Cylinder 1.0: Open silo A
Y2	BOOL	%QX0.1	Cylinder 2.0: Open silo B

**4. Formulation of a PLC program in the various PLC programming languages**

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram

# C-20

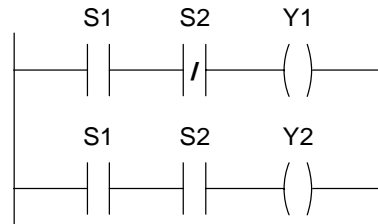
## Solution 8

PROGRAM Silo control system

VAR

S1 AT %IX0.0 : BOOL; (\* S1: Start button \*)  
S2 AT %IX0.1 : BOOL; (\* S2: Selector switch \*)  
Y1 AT %QX0.0 : BOOL; (\* Cylinder 1.0: Open silo A \*)  
Y2 AT %QX0.1 : BOOL; (\* Cylinder 2.0: Open silo B \*)

END\_VAR



END\_PROGRAM

Ladder diagram

PROGRAM Silo control system

VAR

S1 AT %IX0.0 : BOOL; (\* S1: Start button \*)  
S2 AT %IX0.1 : BOOL; (\* S2: Selector switch \*)  
Y1 AT %QX0.0 : BOOL; (\* Cylinder 1.0: Open silo A \*)  
Y2 AT %QX0.1 : BOOL; (\* Cylinder 2.0: Open silo B \*)

END\_VAR

LD S1 (\* Push button S1 actuated \*)  
ANDN S2 (\* Switch S2: Bulk material A \*)  
ST Y1 (\* Discharge bulk material A \*)  
  
LD S1 (\* Push button S1 actuated \*)  
AND S2 (\* Switch S2: Bulk material B \*)  
ST Y2 (\* Discharge bulk material B \*)

END\_PROGRAM

Instruction list

PROGRAM Silo control system

VAR

S1 AT %IX0.0 : BOOL; (\* S1: Start button \*)  
S2 AT %IX0.1 : BOOL; (\* S2: Selector switch \*)  
Y1 AT %QX0.0 : BOOL; (\* Cylinder 1.0: Open silo A \*)  
Y2 AT %QX0.1 : BOOL; (\* Cylinder 2.0: Open silo B \*)

END\_VAR

Y1 := S1 & NOT S2; (\* Discharge bulk material A \*)  
Y2 := S1 & S2; (\* Discharge bulk material B \*)

END\_PROGRAM

Structured text

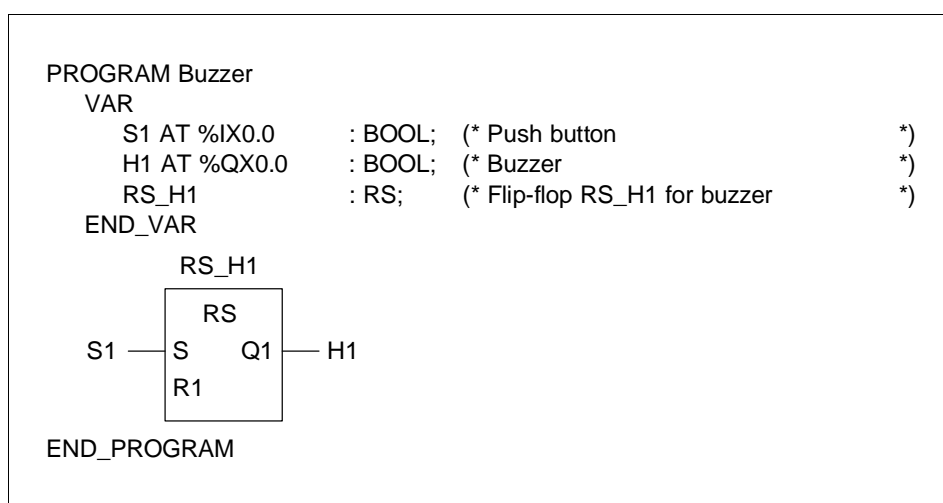


**Fire alarm**  
Setting a PLC output

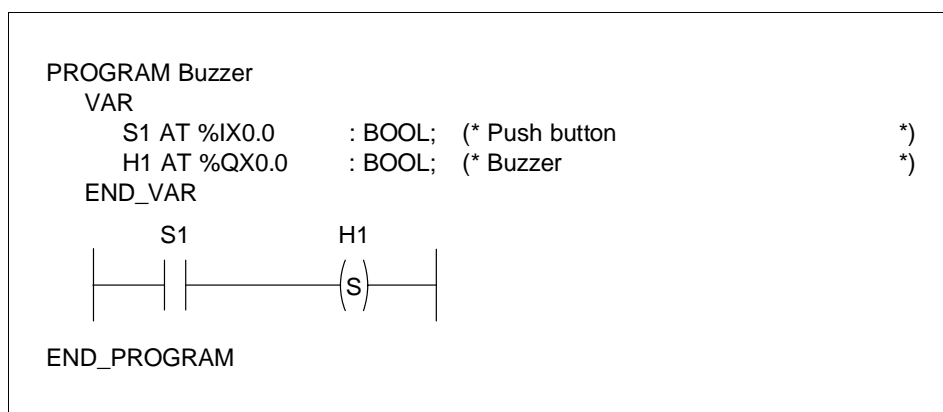
*Title*

**3. Formulation of a PLC program in the various PLC programming languages**

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



*Function block diagram*



*Ladder diagram*

In the language Ladder Diagram, the set operation is effected via a set coil. The linking of an RS function block is therefore not required.

*Instruction list*

```
PROGRAM Buzzer
VAR
  S1 AT %IX0.0      : BOOL; (* Push button *)
  H1 AT %QX0.0      : BOOL; (* Buzzer *)
END_VAR

LD  S1
S   H1

END_PROGRAM
```

The language Instruction List has its own set operator S. The use of an RS function block is therefore not required.

*Structured text*

```
PROGRAM Buzzer
VAR
  S1 AT %IX0.0      : BOOL; (* Push button *)
  H1 AT %QX0.0      : BOOL; (* Buzzer *)
  RS_H1              : RS;   (* Flip-flop RS_H1 for buzzer *)
END_VAR

RS_H1( S := S1 );
H1 := RS_H1.Q1;

END_PROGRAM
```

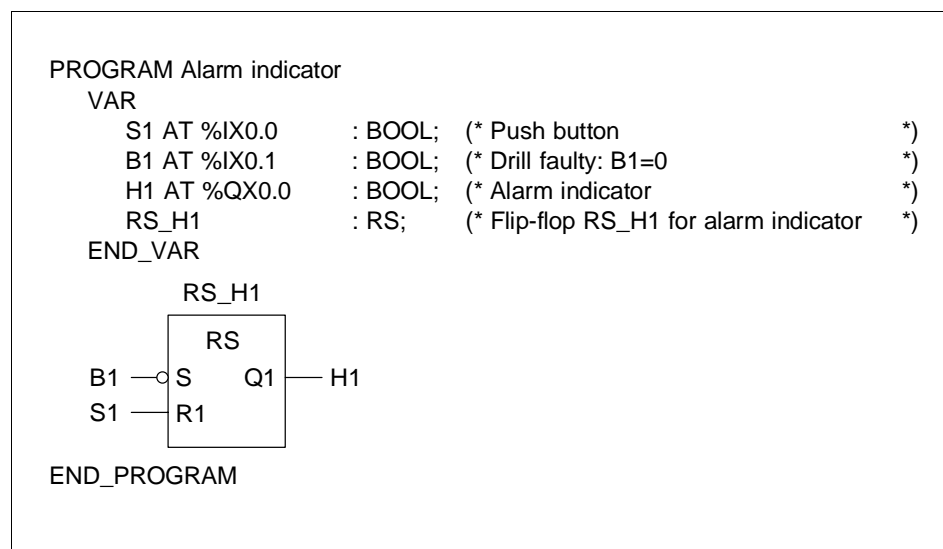
To invoke a function block it is not mandatory for all transfer parameters to be specified. In this case, the function block entity RS\_H1 only receives a current value for the input parameter S, i.e. the value of push button S1.

### Drill breakage monitoring Setting and resetting an output

Title

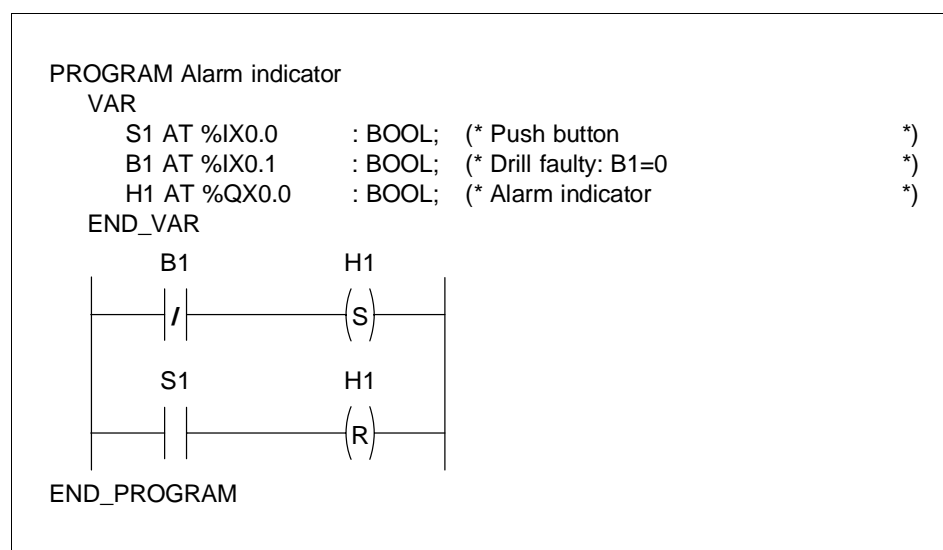
### 3. Formulation of a PLC program in the various PLC programming languages

According to the IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram

The desired behaviour is obtained by means of using an RS function block (reset dominant).



Ladder diagram

# C-24

## Solution 10

```
PROGRAM Alarm indicator
VAR
  S1 AT %IX0.0      : BOOL; (* Push button *)
  B1 AT %IX0.1      : BOOL; (* Drill faulty: B1=0 *)
  H1 AT %QX0.0      : BOOL; (* Alarm indicator *)
END_VAR

LDN  B1
ST   H1
LD   S1
R    H1

END_PROGRAM
```

*Instruction list*

The languages Ladder Diagram and Instruction list have their own operations for the stored setting or resetting of a variable, whereby the use of an RS flip-flop does not apply. The sequence of set and reset commands is crucial for the behaviour of the PLC. The command, which must be dominant, – in this case reset command – must be the last to be processed.

```
PROGRAM Alarm indicator
VAR
  S1 AT %IX0.0      : BOOL; (* Push button *)
  B1 AT %IX0.1      : BOOL; (* Drill faulty: B1=0 *)
  H1 AT %QX0.0      : BOOL; (* Alarm indicator *)
  RS_H1              : RS;   (* Flip-flop RS_H1 for alarm indicator *)
END_VAR

RS_H1( S := NOT B1, R1 := S1 );
H1 := RS_H1.Q1;

END_PROGRAM
```

*Structured text*

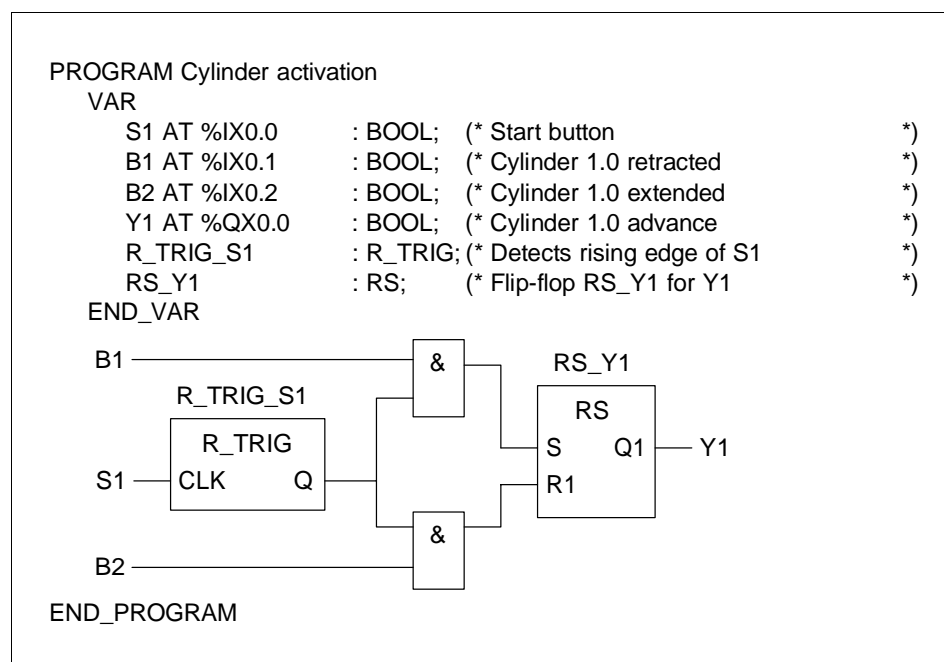
### Activating a cylinder

Signal edges

Title

### 3. Formulation of a PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.

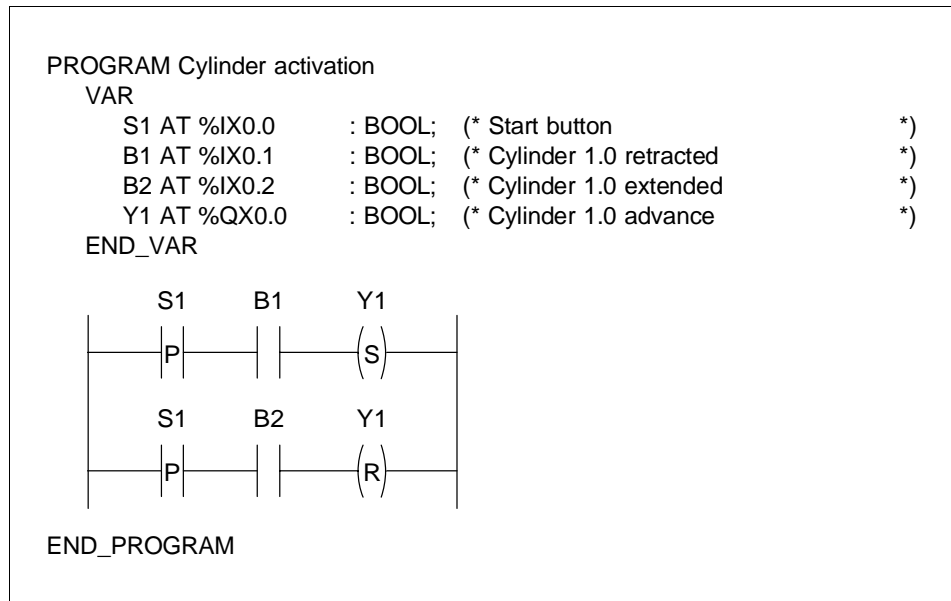


Function block diagram

In the function block diagram, edge evaluation is effected by means of using a R\_TRIG function block.

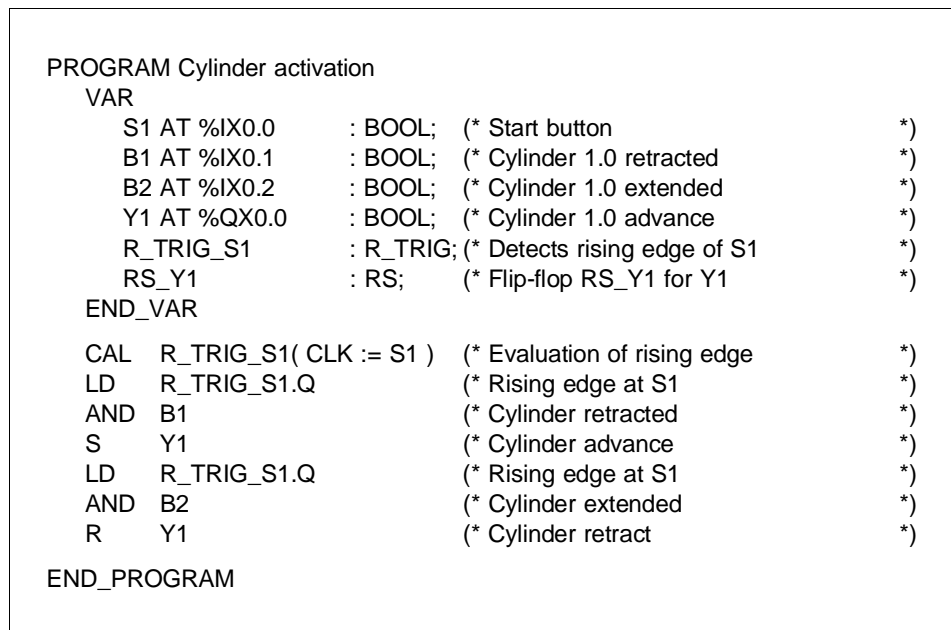
# C-26

## Solution 11



Ladder diagram

The language Ladder Diagram has special contacts for the detection of edges.



Instruction list

```
PROGRAM Cylinder activation
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Cylinder 1.0 retracted *)
  B2 AT %IX0.2      : BOOL; (* Cylinder 1.0 extended *)
  Y1 AT %QX0.0      : BOOL; (* Cylinder 1.0 advance *)
  R_TRIG_S1         : R_TRIG; (* Detects rising edge of S1 *)
  RS_Y1             : RS;     (* Flip-flop RS_Y1 for Y1 *)
END_VAR

R_TRIG_S1( CLK := S1 );      (* Evaluation of rising edge *)
RS_Y1( S := R_TRIG_S1.Q & B1; (* Invoke flip-flop for Y1 *)
R1 := R_TRIG_S1.Q & B2 );
Y1 := RS_Y1.Q1;             (* Status of flip-flop to Y1 *)

END_PROGRAM
```

*Structured text*

In the languages Instruction list and Structured Text, edge detection takes place by invoking an R\_TRIG function block.

# C-28

---

Solution 11



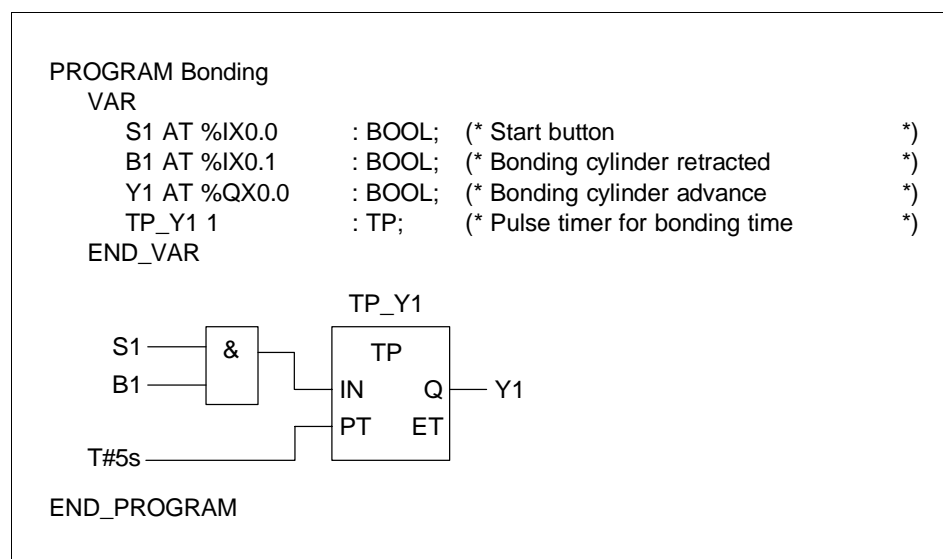
### Bonding of components

Pulse

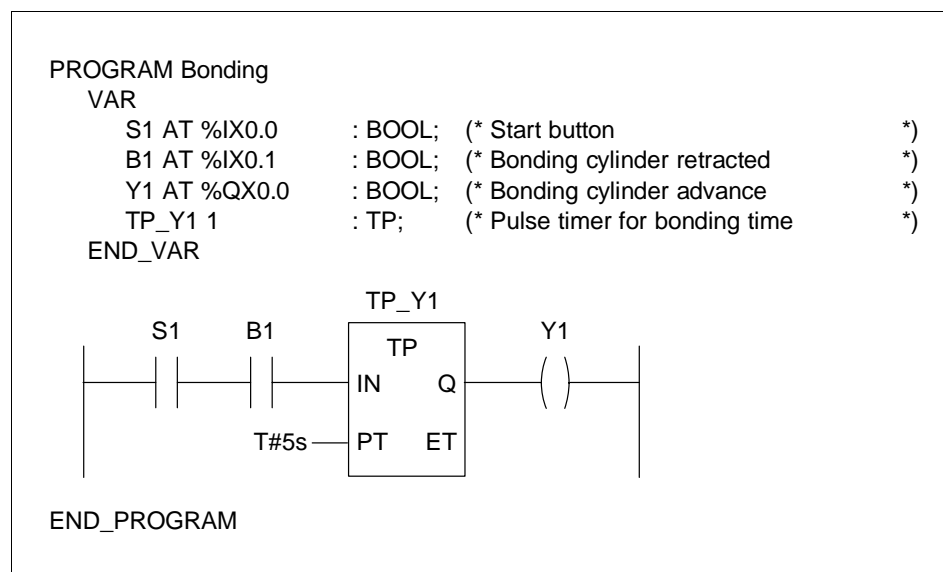
Title

### 3. Formulation of a PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram



Ladder diagram

```
PROGRAM Bonding
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Bonding cylinder retracted *)
  Y1 AT %QX0.0      : BOOL; (* Bonding cylinder advance *)
  T_Start AT %MX0.0 : BOOL; (* Start condition for TP_Y1 *)
  TP_Y1 1           : TP;   (* Pulse timer for bonding time *)
END_VAR

LD   S1
AND  B1
ST   T_Start
CAL  TP_Y1( IN := T_Start, PT := T#5s )
LD   TP_Y1.Q
ST   Y1

END_PROGRAM
```

*Instruction list*

In the language Instruction List, transfer parameters for a function block invocation may consist of only one individual variable. To achieve this, the variable T\_Start is declared in the program.

```
PROGRAM Bonding
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Bonding cylinder retracted *)
  Y1 AT %QX0.0      : BOOL; (* Bonding cylinder advance *)
  TP_Y1 1           : TP;   (* Pulse timer for bonding time *)
END_VAR

TP_Y1( IN := S1 & B1, PT := T# 5s );
Y1 := TP_Y1.Q;

END_PROGRAM
```

*Structured text*

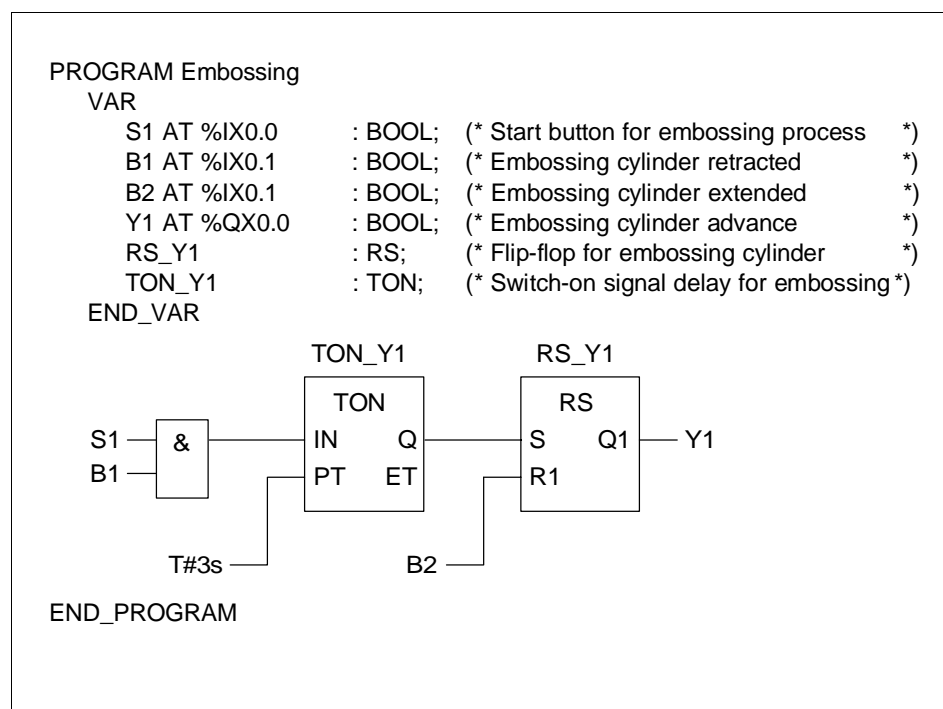
### Embossing device

Switch-on signal delay

Title

### 3. Formulation of a PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram

# C-32

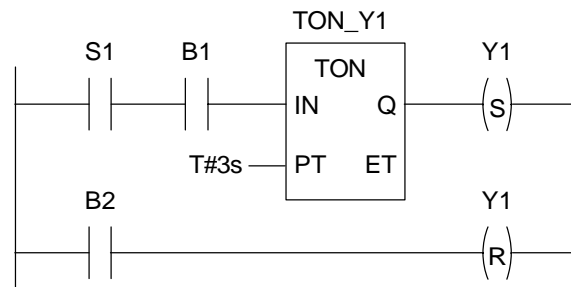
## Solution 13

PROGRAM Embossing

VAR

S1 AT %IX0.0 : BOOL; (\* Start button for embossing process \*)  
B1 AT %IX0.1 : BOOL; (\* Embossing cylinder retracted \*)  
B2 AT %IX0.1 : BOOL; (\* Embossing cylinder extended \*)  
Y1 AT %QX0.0 : BOOL; (\* Embossing cylinder advance \*)  
RS\_Y1 : RS; (\* Flip-flop for embossing cylinder \*)  
TON\_Y1 : TON; (\* Switch-on signal delay for embossing \*)

END\_VAR



END\_PROGRAM

Ladder diagram

PROGRAM Embossing

VAR

S1 AT %IX0.0 : BOOL; (\* Start button for embossing process \*)  
B1 AT %IX0.1 : BOOL; (\* Embossing cylinder retracted \*)  
B2 AT %IX0.1 : BOOL; (\* Embossing cylinder extended \*)  
Y1 AT %QX0.0 : BOOL; (\* Embossing cylinder advance \*)  
T\_Start AT %MX0.0 : BOOL; (\* Start condition for TON\_Y1 \*)  
RS\_Y1 : RS; (\* Flip-flop for embossing cylinder \*)  
TON\_Y1 : TON; (\* Switch-on signal delay for embossing \*)

END\_VAR

LD S1  
AND B1  
ST T\_Start  
CAL TON\_Y1( IN := T\_Start, PT := T#3s )  
LD TON\_Y1.Q  
S Y1  
LD B2  
R Y1

END\_PROGRAM

Instruction list

```
PROGRAM Embossing
VAR
  S1 AT %IX0.0      : BOOL; (* Start button for embossing process *)
  B1 AT %IX0.1      : BOOL; (* Embossing cylinder retracted *)
  B2 AT %IX0.1      : BOOL; (* Embossing cylinder extended *)
  Y1 AT %QX0.0      : BOOL; (* Embossing cylinder advance *)
  RS_Y1             : RS;   (* Flip-flop for embossing cylinder *)
  TON_Y1            : TON;  (* Switch-on signal delay for embossing *)
END_VAR

TON_Y1( IN := S1 & B1, PT := T#3s );
RS_Y1( S := TON_Y1.Q, R1 := B2 );
Y1 := RS_Y1.Q1;

END_PROGRAM
```

*Structured text*

# C-34

---

Solution 13

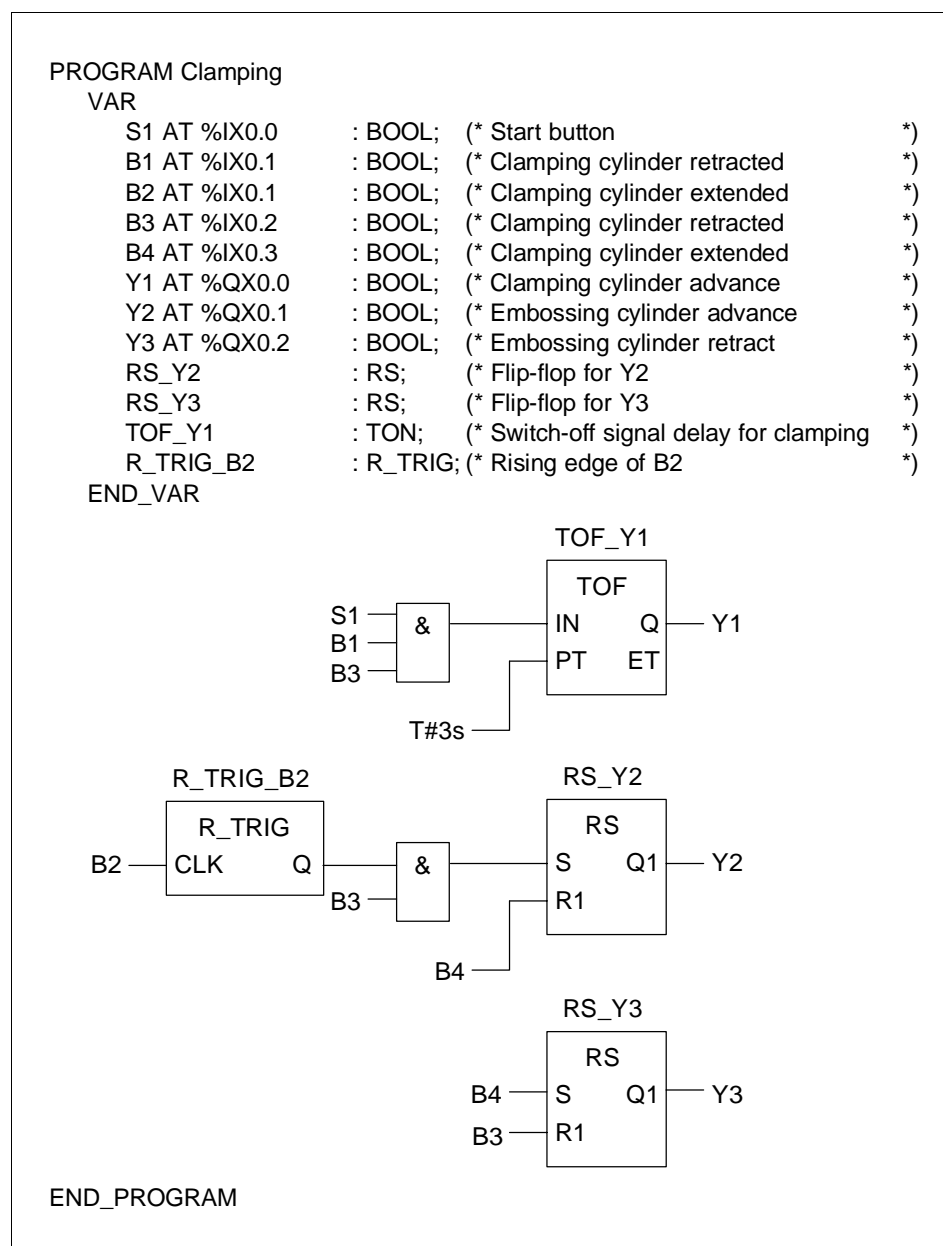
### Clamping device

Switch-off signal delay

Title

### 3. Formulation of a PLC program in the various PLC programming languages

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Function block diagram

# C-36

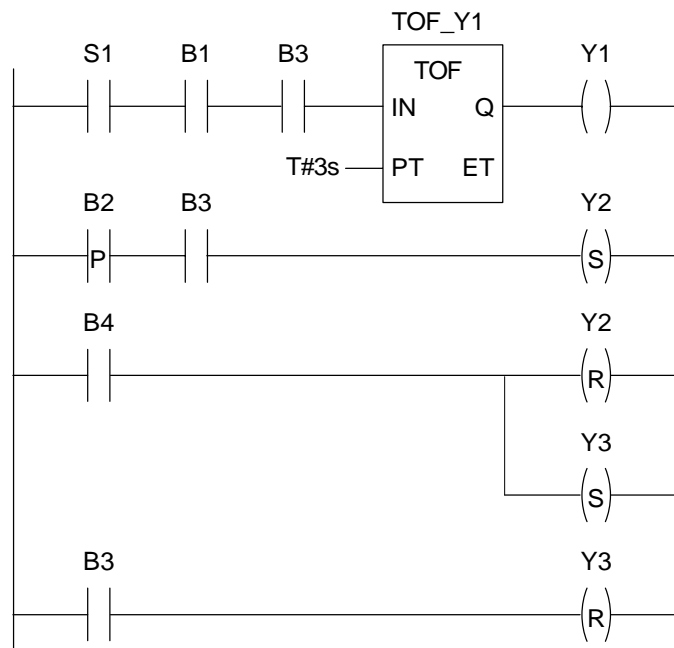
## Solution 14

PROGRAM Clamping

VAR

S1 AT %IX0.0 : BOOL; (\* Start button \*)  
B1 AT %IX0.1 : BOOL; (\* Clamping cylinder retracted \*)  
B2 AT %IX0.1 : BOOL; (\* Clamping cylinder extended \*)  
B3 AT %IX0.2 : BOOL; (\* Embossing cylinder retracted \*)  
B4 AT %IX0.3 : BOOL; (\* Embossing cylinder extended \*)  
Y1 AT %QX0.0 : BOOL; (\* Clamping cylinder advance \*)  
Y2 AT %QX0.1 : BOOL; (\* Embossing cylinder advance \*)  
Y3 AT %QX0.2 : BOOL; (\* Embossing cylinder retract \*)  
TOF\_Y1 : TON; (\* Switch-off signal delay for clamping \*)

END\_VAR



END\_PROGRAM

Ladder diagram



```
PROGRAM Clamping
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Clamping cylinder retracted *)
  B2 AT %IX0.1      : BOOL; (* Clamping cylinder extended *)
  B3 AT %IX0.2      : BOOL; (* Embossing cylinder retracted *)
  B4 AT %IX0.3      : BOOL; (* Embossing cylinder extended *)
  Y1 AT %QX0.0      : BOOL; (* Clamping cylinder advance *)
  Y2 AT %QX0.1      : BOOL; (* Embossing cylinder advance *)
  Y3 AT %QX0.2      : BOOL; (* Embossing cylinder retract *)
  T_Start AT %MX0.0 : BOOL; (* Start condition for TOF_Y1 *)
  TOF_Y1            : TON; (* Switch-off signal delay for clamping *)
  R_TRIG_B2        : R_TRIG; (* Rising edge of B2 *)
END_VAR

LD  S1
AND B1
AND B3
ST  T_Start
CAL TOF_Y1( IN := T_Start, PT := T#3s )
LD  TOF_Y1.Q
ST  Y1
CAL R_TRIG_B2( CLK := B2 )
LD  R_TRIG_B2.Q
AND B3
S   Y2
LD  B4
R   Y2
S   Y3
LD  B3
R   Y3

END_PROGRAM
```

*Instruction list*

```
PROGRAM Clamping
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Clamping cylinder retracted *)
  B2 AT %IX0.1      : BOOL; (* Clamping cylinder extended *)
  B3 AT %IX0.2      : BOOL; (* Embossing cylinder retracted *)
  B4 AT %IX0.3      : BOOL; (* Embossing cylinder extended *)
  Y1 AT %QX0.0      : BOOL; (* Clamping cylinder advance *)
  Y2 AT %QX0.1      : BOOL; (* Embossing cylinder advance *)
  Y3 AT %QX0.2      : BOOL; (* Embossing cylinder retract *)
  RS_Y2             : RS;   (* Flip-flop for Y2 *)
  RS_Y3             : RS;   (* Flip-flop for Y3 *)
  TOF_Y1            : TON;  (* Switch-off signal delay for clamping *)
  R_TRIG_B2         : R_TRIG; (* Rising edge of B2 *)
END_VAR

TOF_Y1( IN := S1 & B1 & B3, PT := T#3s );
Y1 := TOF_Y1.Q;
R_TRIG_B2( CLK := B2 );
RS_Y2( S := R_TRIG_B2.Q & B3, R1 := B4 );
Y2 := RS_Y2.Q1;
RS_Y3( S := B4, R1 := B3 );
Y3 := RS_Y3.Q1;

END_PROGRAM
```

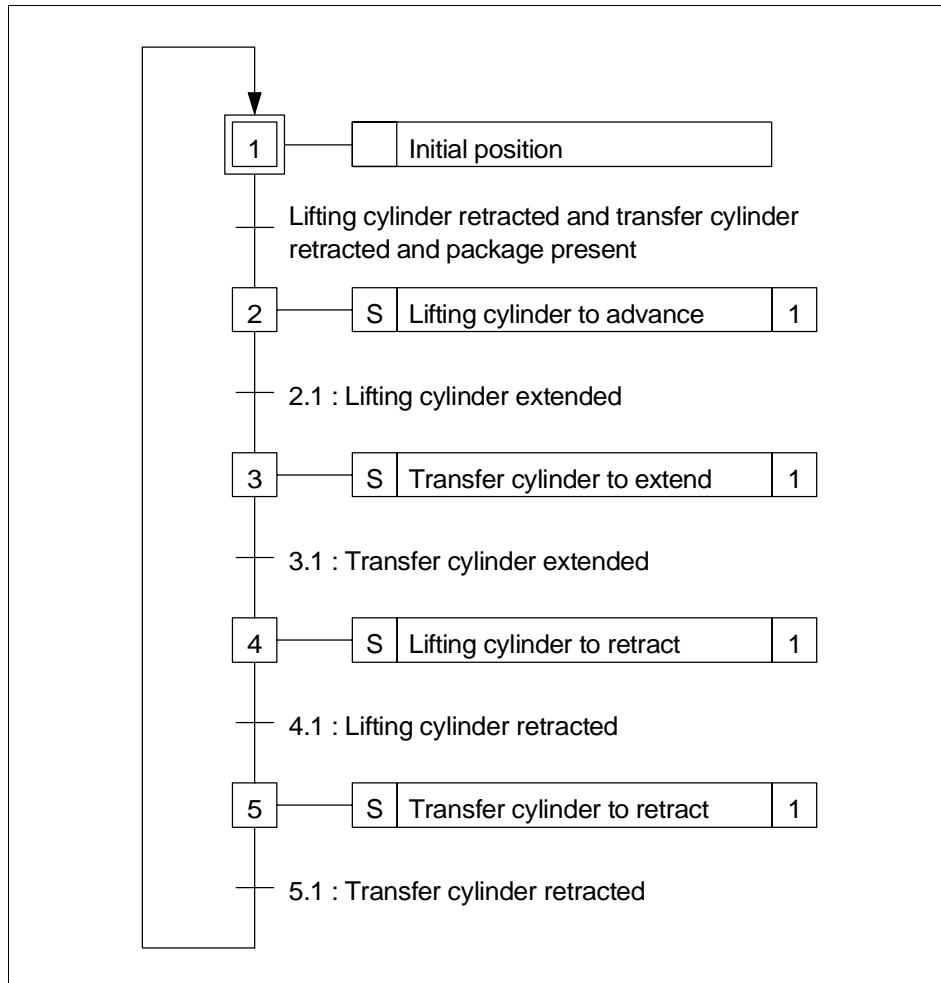
Structured text

## Lifting device for packages

Linear sequence

Title

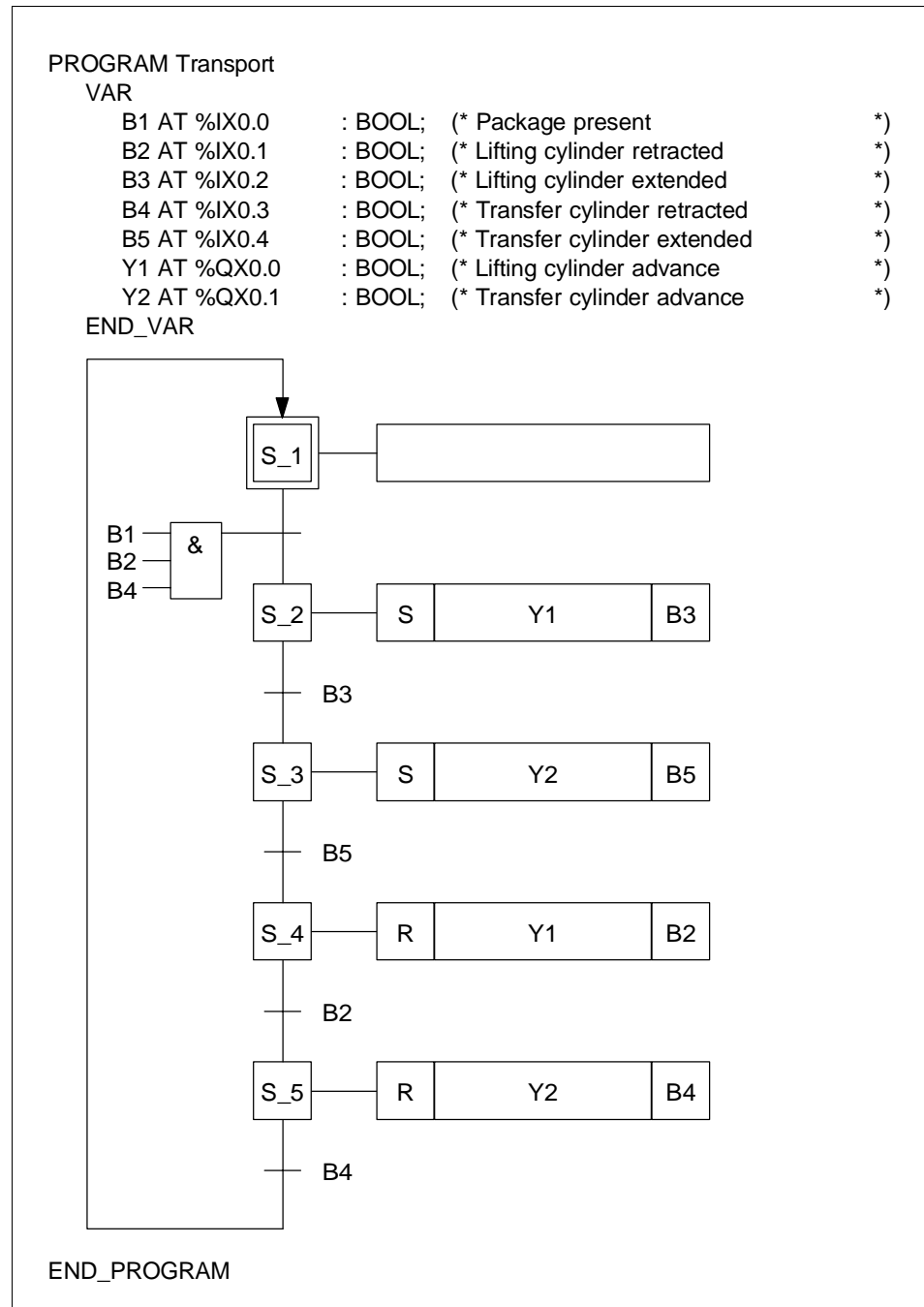
### 2. Description of the control task in function chart to IEC848



Function chart  
IEC 848

## 4. Formulation of PLC program

According to IEC 1131-3, a program consists of a program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Sequential function chart

The main body of the program is structured in sequential function chart. The associated transition conditions are formulated in function block diagram. If a transition condition consists of one boolean variable only, this is represented in structured text. The steps consist of simple boolean actions with the qualifiers S (stored set) and R (stored reset). Step S\_1 is a void step.

# C-42

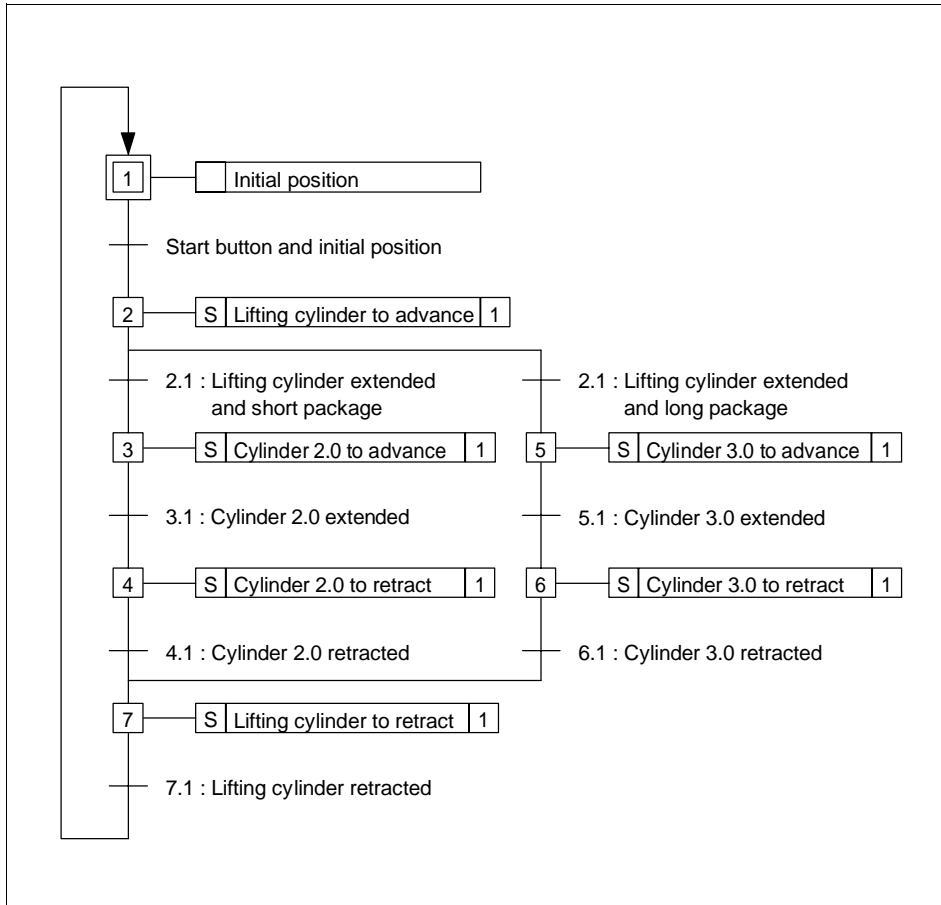
---

Solution 15

**Lifting and sorting device for packages**  
Sequence with alternative branching

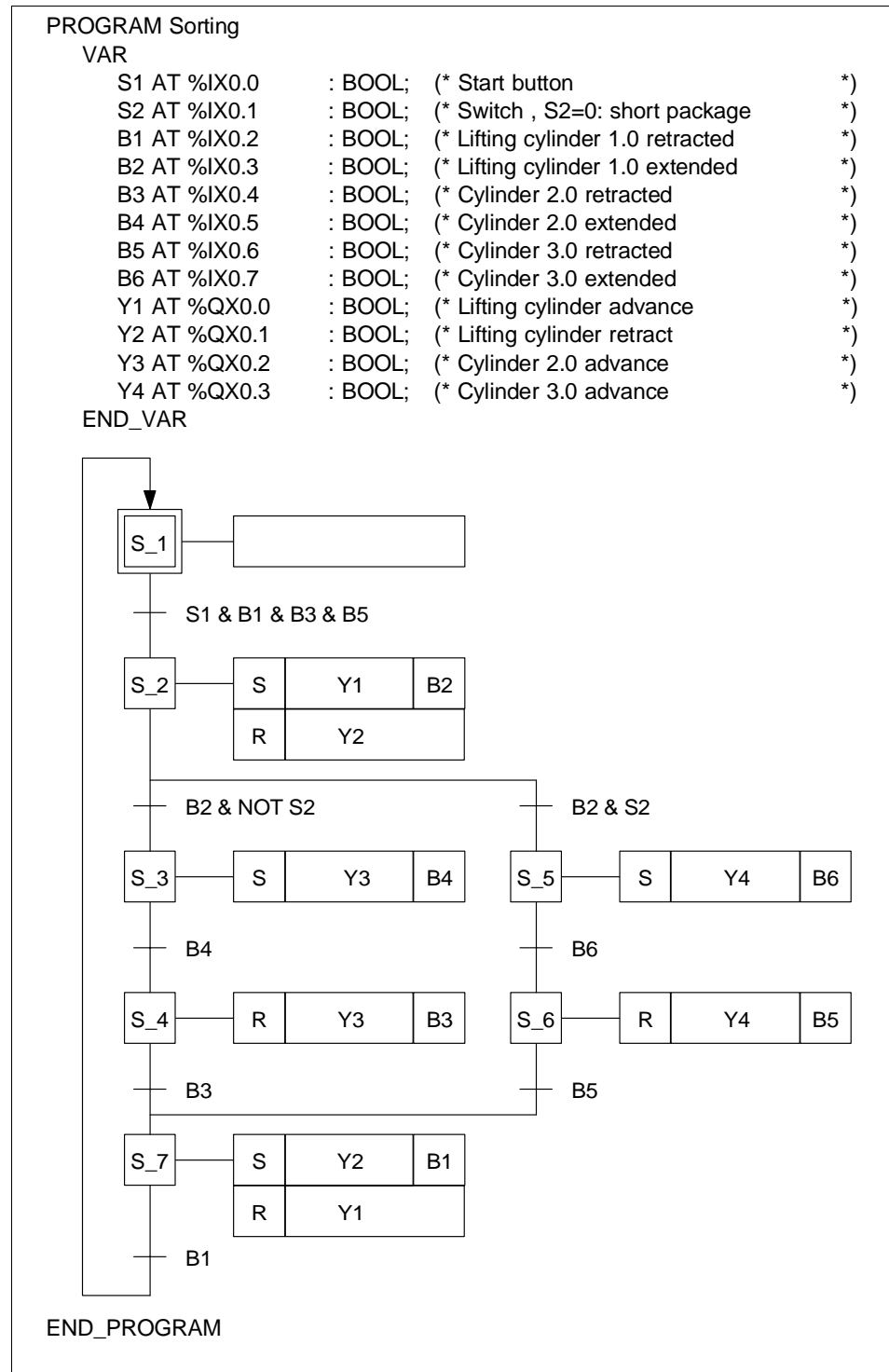
Title

**2. Description of the control task in function chart to IEC848**



## 4. Formulation of PLC program

According to IEC 1131-3, a program consists of the program descriptor – this also includes the declaration of variables – and the main body of the program. This is why the declaration of variables in textual form is a component part of every represented solution.



Sequential function chart



The main body of the program is structured in sequential function chart. The associated transition conditions are represented in the structured text language. The steps consist of simple boolean actions with the qualifiers S and R. The initial step S\_1 is a void step. After step S\_2 the program branches into two alternative sequences.

# C-46

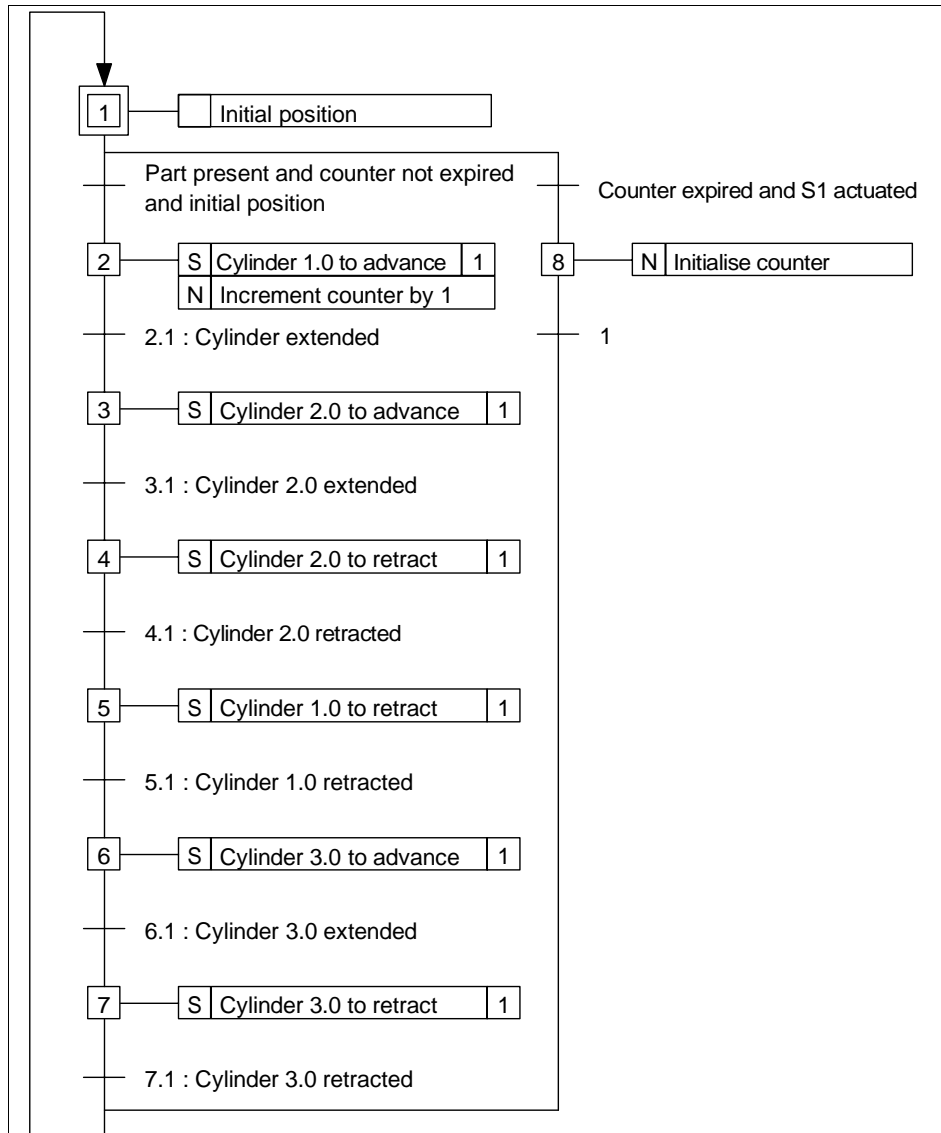
---

Solution 16

**Stamping device with counter**  
Counting cycles

Title

**2. Description of the control task in function chart to IEC848**

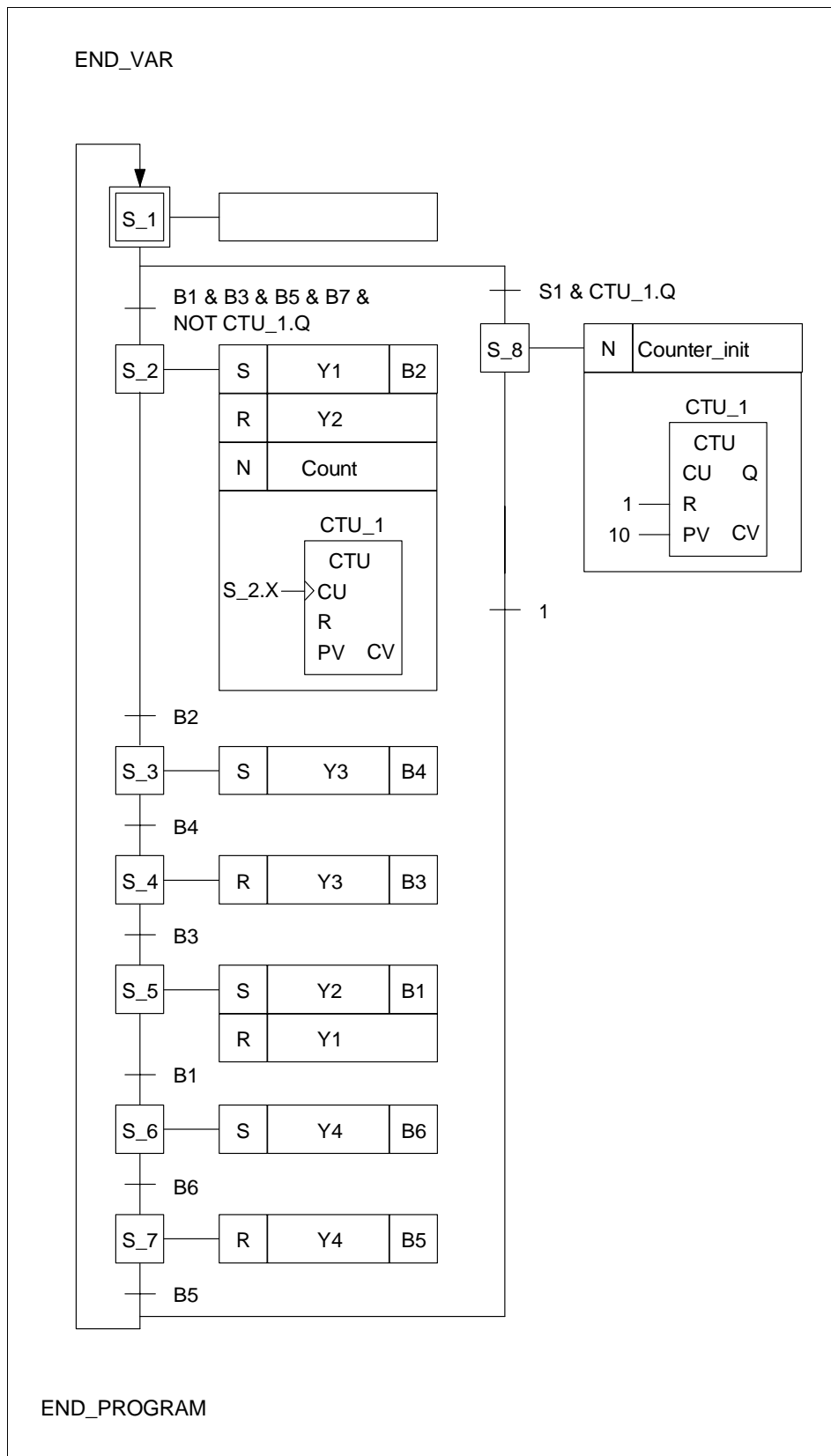


Function chart to IEC 848

## 4. Formulation of PLC program

```
PROGRAM Stamping
VAR
  S1 AT %IX0.0      : BOOL; (* Start button *)
  B1 AT %IX0.1      : BOOL; (* Cylinder 1.0 retracted *)
  B2 AT %IX0.2      : BOOL; (* Cylinder 1.0 extended *)
  B3 AT %IX0.3      : BOOL; (* Cylinder 2.0 retracted *)
  B4 AT %IX0.4      : BOOL; (* Cylinder 2.0 extended *)
  B5 AT %IX0.5      : BOOL; (* Cylinder 3.0 retracted *)
  B6 AT %IX0.6      : BOOL; (* Cylinder 3.0 extended *)
  B7 AT %IX0.7      : BOOL; (* Part in magazine *)
  Y1 AT %QX0.0      : BOOL; (* Cylinder 1.0 advance *)
  Y2 AT %QX0.1      : BOOL; (* Cylinder 1.0 retract *)
  Y3 AT %QX0.2      : BOOL; (* Cylinder 2.0 advance *)
  Y4 AT %QX0.3      : BOOL; (* Cylinder 3.0 advance *)
  CTU_1              : CTU; (* Incremental counter named CTU_1 *)
END_VAR
```

*Declaration of variables*



The main body of the program is structured in sequential function chart. The associated transition conditions are represented in the structured text language. Following the initial step S\_1, the program branches into to alternative sequences.

If push button S1 is actuated for the first time, the used counter function block copy CTU\_1 is initialised. This always occurs within the action named "Counter\_init". The program returns to step S\_1 via a permanently true transition condition. The conditions of the lefthand sequence chain are now fulfilled. This is cyclically processed until 10 parts have been stamped and ejected. The actual counting procedure is not programmed in the boolean action "Count". The edge triggering of the counting procedure is created via the step flag S\_2.X.

When the function block copy CTU\_1 has been initialised, the value 0 applies at input CU. If step S\_2 is now executed, step flag S\_2.X assumes the value 1, whereby a rising edge briefly applies at input CU.

If the transition condition B2 has now been met, the action "Count" is executed as a non-boolean action for the last time. For this final evolution process, step flag S\_2.X already has the value 0 and as such also input CU of CTU\_1. If the program reaches step S\_2 one further time, a change takes place in the status of input CU from 0 to 1: the rising edge for the resolution of the counting cycle applies.