

# SIEMENS

## SIMATIC S7-200

### S7-200 Getting Started - Advanced

#### Training Documents

---

#### Preface

---

#### Quick review

1

---

#### Latching

2

---

#### Pulse Operated Switch

3

---

#### Off Delay Timer

4

---

#### Sequencer

5

---

#### Learning More

6

---

#### Appendix A

A

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

### DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

### WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

### CAUTION

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

### CAUTION

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

### NOTICE

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

### WARNING

This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

## Trademarks

All names identified by © are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

# Preface

Dear SIMATIC S7-200 user,

We created the S7-200 Getting Started manuals so that first time users can quickly and easily learn to use the SIMATIC S7-200. Building on the Getting Started - Beginners manual the Getting Started - Advanced manual will in a short time familiarize you with the SIMATIC S7-200 Micro PLC. Using a few example tasks, you will learn how the controller operates and how to create control programs for basic automatic tasks.

After working through this manual, you will find it easy to solve typical controller tasks on your own. You can find and load the SIMATIC S7-200 program examples from the STEP 7-Micro/WIN Documentation CD or the Programming Examples CD provided in the back cover of this manual.





# Table of contents

	<b>Preface</b> .....	<b>3</b>
<b>1</b>	<b>Quick review</b> .....	<b>7</b>
1.1	A Few Words of Review .....	7
1.2	Here are the Bits.....	8
1.3	Power flow in a ladder diagram.....	9
1.4	Finding information about ladder instructions .....	10
1.5	The PLC scan cycle: Reading the inputs .....	11
1.6	The PLC scan cycle: Writing the outputs .....	12
1.7	The PLC scan cycle: Program logic.....	13
<b>2</b>	<b>Latching</b> .....	<b>15</b>
2.1	Introduction.....	15
2.2	Normally Closed (NC) Contact.....	16
2.3	Inserting a normally-closed contact .....	17
2.4	Testing your circuit.....	18
2.5	A faster way to create logic state latches.....	19
2.6	Making a Set/Reset network.....	20
2.7	Safety Tip: How to control a machine shutdown when a wire breaks.....	21
<b>3</b>	<b>Pulse Operated Switch</b> .....	<b>23</b>
3.1	Introduction.....	23
3.2	Solution Overview.....	24
3.3	Edge detection.....	25
3.4	Examples of edge detection .....	26
3.5	Bit memory (M addresses).....	27
3.6	Pulse-operated switch .....	28
3.7	Solution Description and Test.....	29
3.8	Time to Show What You Know .....	30
<b>4</b>	<b>Off Delay Timer</b> .....	<b>31</b>
4.1	Introduction.....	31
4.2	Save As.....	33
4.3	Solution Overview.....	34
4.4	Solution - Enter the Program .....	35
4.5	Solution Description.....	36
4.6	Entering comments.....	38
4.7	Time to Show What You Know .....	41

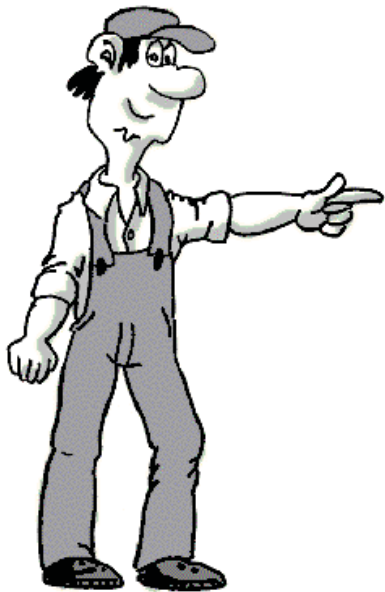
<b>5</b>	<b>Sequencer</b> .....	<b>43</b>
5.1	Introduction .....	43
5.2	Solution Starting Point .....	44
5.3	Sequencer control.....	45
5.4	Transition condition.....	46
5.5	Structure of the sequencer program .....	47
5.6	Control logic for the sequencer .....	48
5.7	Working with memory bits.....	49
5.8	Transition condition for the sequencer .....	50
5.9	Advantages of working with sequencers .....	51
5.10	Important safety considerations .....	52
5.11	Making safe transitions .....	53
5.12	Modification .....	54
5.13	Solution description: Transitions .....	55
5.14	Solution description: Activating the steps.....	56
5.15	Solution description: Activating a timer .....	57
5.16	Solution description: Activating the outputs .....	58
5.17	Test .....	59
<b>6</b>	<b>Learning More</b> .....	<b>61</b>
6.1	Do you Want to Learn More? .....	61
<b>A</b>	<b>Appendix A</b> .....	<b>63</b>
A.1	Bridge Circuit .....	63
A.2	Diode Circuit .....	64
A.3	Changeover Switch.....	65

## Quick review

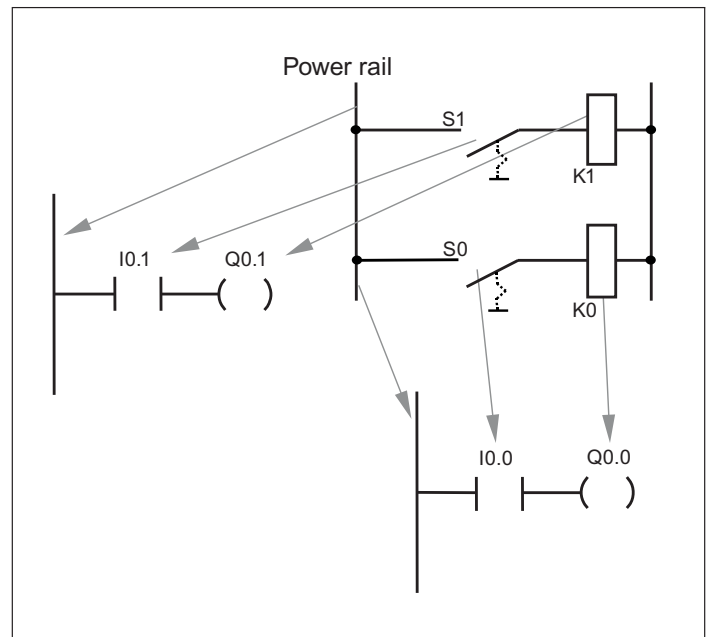
### 1.1 A Few Words of Review

In the S7-200 Getting Started - Beginners, you saw how the circuit diagram for a contactor controller relates to the ladder diagram used by a programmable controller. It is simply a representation with different symbols.

In addition, you were already able to program small logic operations yourself. You even learned to use timers.



Compare with Section 4.7 in the S7-200 Getting Started - Beginners manual

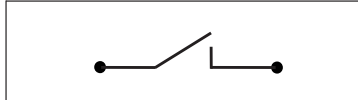


## 1.2 Here are the Bits

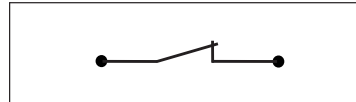
The smallest unit to be processed is the bit! The bit can assume two states:

- "1" meaning "bit set" or state is "true"
- "0" meaning "bit not set" or state is "false"

In a method familiar to you, the two binary states "1" and "0" can be represented as electrical circuits, that is, they can be represented by current flow (1) and no current flow (0).

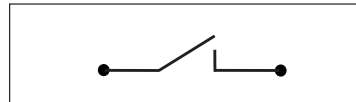


A closed switch:  
 Current flows so bit state = "1"



"1" = "true" =  
 Current flows

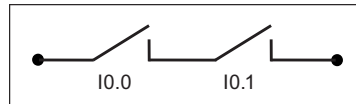
and an open switch:  
 No current flows so bit state = "0".



"0" = "false" =  
 No current flows

From here it requires only a short step to the representation of logic operations as circuits, e.g. series connection of two contacts.

The AND operation of inputs I0.0 and I0.1 is represented as shown on the right.



AND  
 operation

This is represented as follows in LAD:



Finally, a small convention:

The following applies for positive logic:  
 24V = high-level = "1" and  
 0V = low-level = "0".

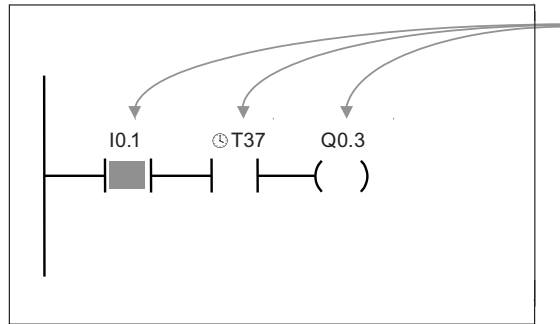
positive logic

The following applies for negative logic:  
 0V = low-level = "1"  
 24V = high-level = "0".

negative logic

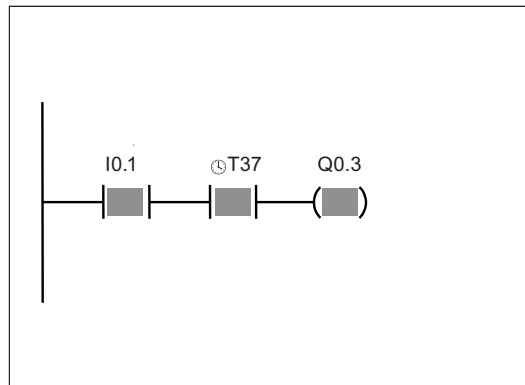


### 1.3 Power flow in a ladder diagram



In this example, output Q0.3 would be active or "1", if the contact at I0.1 is closed, or bit value "1" (24 V DC at input I0.1) AND simultaneously, the timer bit T37 is active, or "1".

Input I0.1 is now "1", and contact I0.1 is closed. T37 is not active in the figure, and it is a "0". Q0.3 remains inactive or "0"

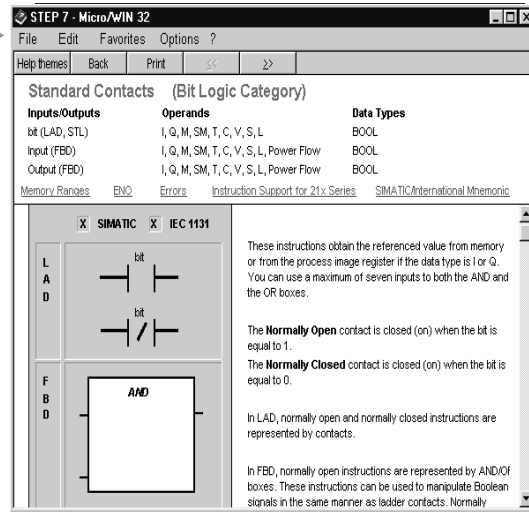
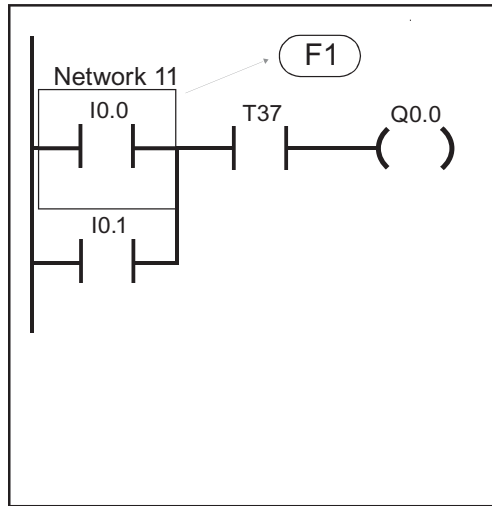


If timer T37 is also "1" (T37 has elapsed), the result of the AND operation is "1" and so output Q0.3 is also "1".

The output bit is then also "true", in other words, it takes the value "1" (gray background).

This corresponds to the LAD status view that you have already used in the Getting Started - Beginners for debugging your program.

## 1.4 Finding information about ladder instructions

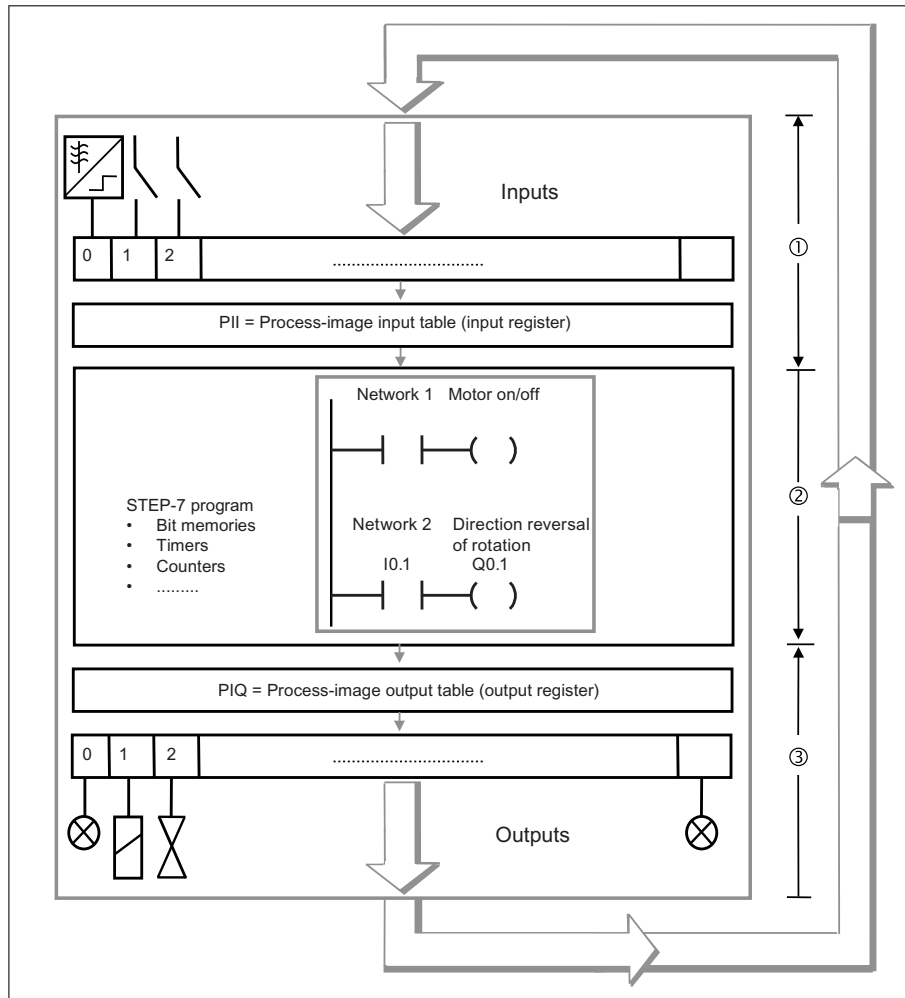


If you want to see the on-line help for a contact symbol or for other functions:  
Select the contact with the box cursor with a simple click of the mouse and then press F1.:

- in the Ladder Diagram (LAD) or
- in the Function Block Diagram (FBD)

## 1.5 The PLC scan cycle: Reading the inputs

All SIMATIC programmable controllers use a scan cycle. In this cyclical operation, the switch states are read at the inputs and stored in the process input image (PII). This information is interpreted by your control program.



## 1.6 The PLC scan cycle: Writing the outputs

The outputs in the process-image output table (PIQ) are overwritten by the control logic in the program. The bit states in the PIQ are transferred to the physical outputs in the final step. The PLC scan cycle continuously repeats this process.



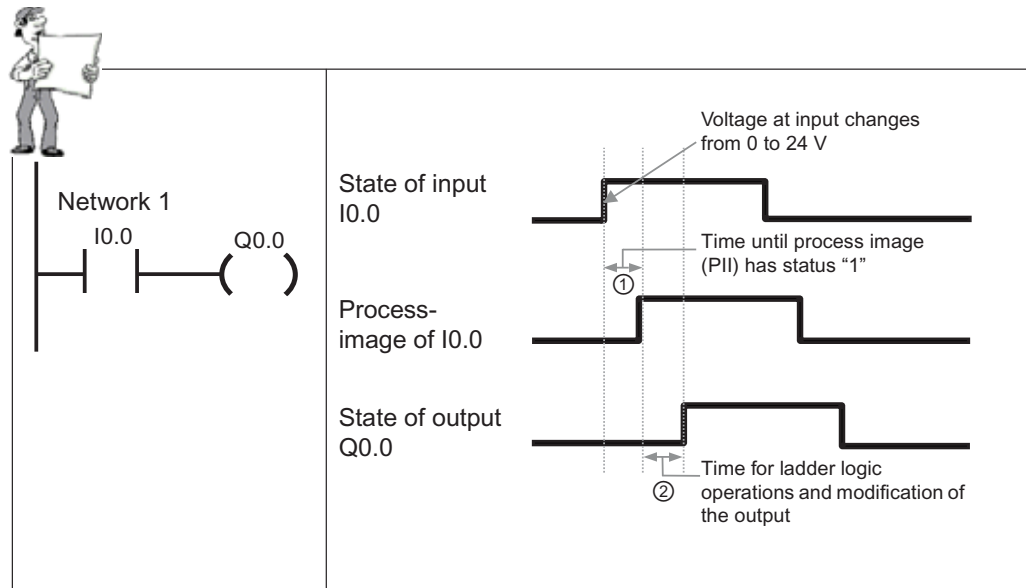
A typical scan cycle takes between 3 and 10 ms. The duration depends on the number and type of network logic instructions.

The cycle consists of two main components:

- 1) Operating system time, typ. 1 ms; corresponding to phase ① and ③ on the previous page.
- 2) Time for processing the commands; corresponding to phases ②, on the previous page.

In addition, the program logic is only processed when the PLC is operating, in other words, when it is in "RUN" mode.

## 1.7 The PLC scan cycle: Program logic



Signal state changes at the inputs which take place during a scan cycle are transferred to the input register in the next scan cycle. There, the signal states for this scan cycle are "frozen". This is the process-image input table PII (see ①).

In the next scan cycle, the transferred states are combined in accordance with the ladder logic (see ②) and the outputs are updated in accordance with the results of the logic operations.

Outputs modified only at the end of the next scan cycle.

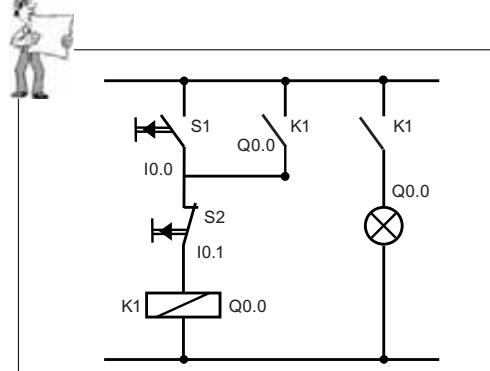


# Latching

# 2

## 2.1 Introduction

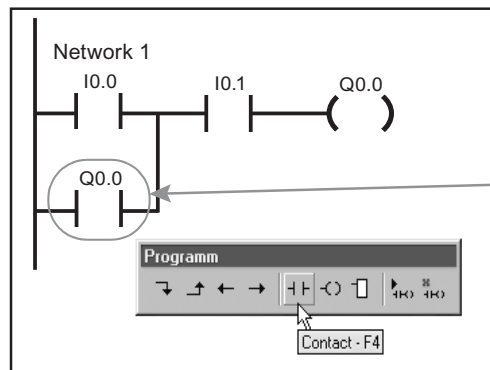
You should now be familiar with the standard latching function and here you will learn how to program it.



For example:

Output Q0.0 is to be activated as soon as S1 at input I0.0 is operated. With latching, Q0.0 is to remain active until S2 at input I0.1 is operated and thus interrupts the latch.

In STEP 7-Micro/WIN, open the first practice project "2h\_pr\_01.mwp" from the Programming Exercises CD. There are still a few elements missing in the program. Add the missing LAD elements now as a short exercise.



To allow the latching function to work, the output (Q0.0 in this case), must itself ensure, as soon as it is activated, that it retains its "true" state and therefore remains active.

This is achieved by switching the output (Q0.0 in this case) as a contact in parallel to the tripping input in the same way as a conventional contactor circuit (Q0.0 can be compared to our contactor K1).

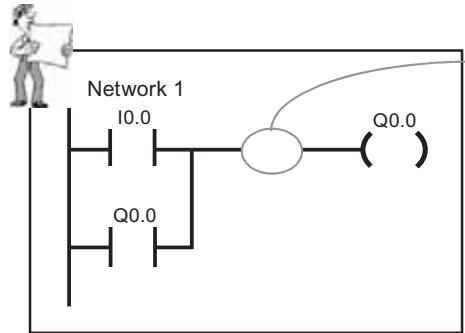
Output Q0.0 as an input ensures latching.

First add a contact Q0.0 at the point indicated as a parallel circuit to I0.0. To enter the contact:

1. Click on the ladder diagram, (in the blank space just below the I0.0 contact and just above "Network 2"). Then, click on the toolbar button for a contact (F4). Now click on the normally open contact in the drop list. (As indicated on the tooltip, you can also use function key F4 instead of the mouse). Finally, click on the address field, replace ??? with Q0.0, then press enter.
2. To enter the vertical line, place the box cursor on the I0.0 contact, and then click on the line down toolbar button.

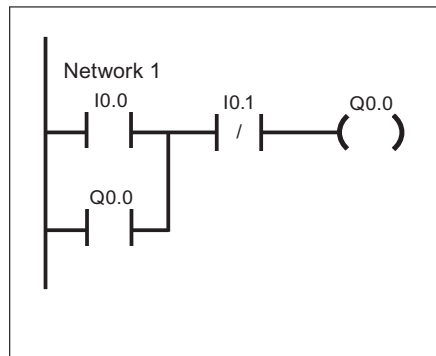
## 2.2 Normally Closed (NC) Contact

To disable the latching function, input I0.1 must break the current path when operated. If a current path is interrupted (i.e. state "0" exists) when a switch is operated, this is referred to as an NC contact.

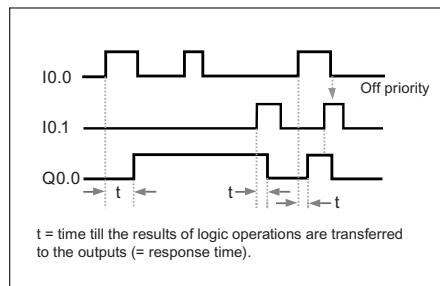


An element must be inserted which works as an NC contact in the ladder diagram when there is 24 V DC ("true") at input I0.1.

NC contact:

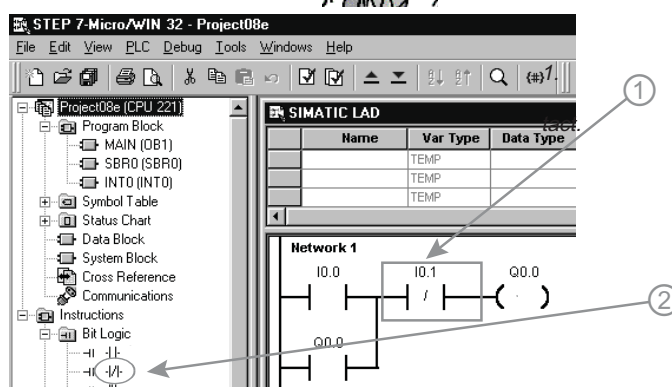
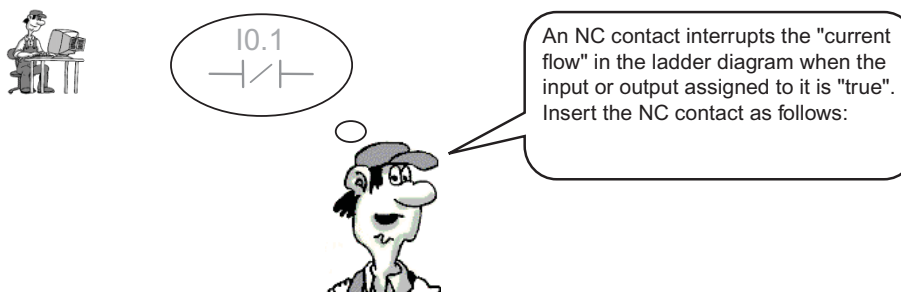


This is what the finished latching function looks like! Below is the principle of operation shown as a timing diagram.

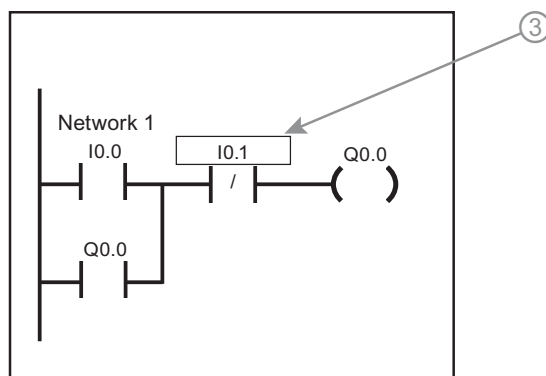




## 2.3 Inserting a normally-closed contact



Note: Press the insert key to toggle ON the Overwrite (OVR) mode. Look at the Status bar on the bottom-right corner of the STEP 7- Micro/Win window to make sure the OVR mode is set.



1. Click the mouse to mark the LAD diagram position with the box cursor that shows where the NC contact will be placed.

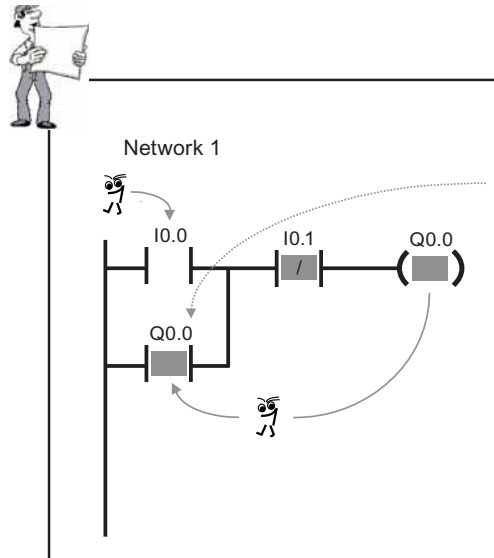
2. Open the Bit Logic folder in the Instruction Tree and double click the NC contact symbol. The NC contact is placed in the ladder diagram

3. Finally, enter the desired NC contact bit address (I0.1, in this case). This is done by selecting the address field with a mouse click and then typing the address.

4. Always terminate text field inputs by pressing Enter.

## 2.4 Testing your circuit

As in the contactor circuit, your LAD network has an output state contact (Q0.0) connected in parallel with the tripping element (I0.0).



If, during a scan cycle, output Q0.0 has been activated by operation of switch S1 at I0.0, contact Q0.0 parallel to I0.0 is closed in the very next cycle (a few milliseconds later). This brings about latching. NC contact I0.1 can terminate the latch when switch S2 at I0.1 is operated.



Save your completed program to hard disk. Then you can load it again at any time and continue to edit it. (We will need the program again for our OFF Delay example.)



Then transfer the program to the PLC to test the function.



For test purposes, switch the PLC to the "RUN" mode.

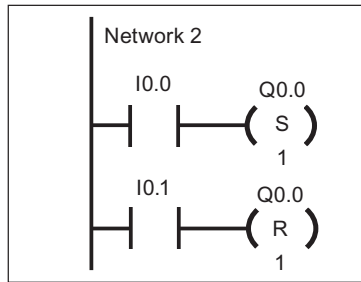
Test your program by operating the two switches on the input simulator connected at I0.0 and I0.1. Observe the lamps on the S7-200 or the LAD output status!

1. Begin by switching ON I0.0.
2. I0.1 must be switched OFF. The LED at I0.0 must light up.
3. Q0.0 will then light up.
4. As soon as I0.1 is switched ON, Q0.0 becomes ="0".

## 2.5 A faster way to create logic state latches



Instead of feeding back the output, as in the previous example, here the functions "Set" and "Reset" are used instead. Have a look at the ladder diagram below.



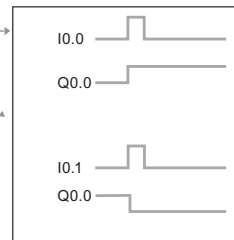
For the "Set" operation **-(S)**, a switching pulse at I0.0 has the effect that Q0.0 is activated in a steady state.

-(S) Set

For the "Reset" operation **-(R)**, a switching pulse at I0.1 has the effect that Q0.0 is deactivated again.

-(R) Reset

The "coils" **-(S) Set Q0.0 to "1"** and **-(R) Reset Q0.0 to "0"** are used frequently in PLC programs to switch outputs or bit memories ON or OFF to a steady state when a series-connected contact is briefly operated.



Steady-state setting of value with (S)

Resetting with (R)

-(S) ⇔ 1  
-(R) ⇔ 0

A "set" output or memory bit remains "set" until it is reset by the -(R) statement.

If the set coil and the associated reset coil of an output both have signal "1", the last operation in the program takes priority.

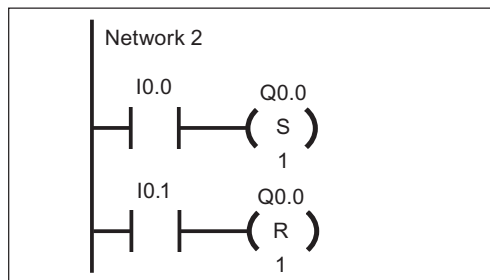
Last operation in cycle has priority



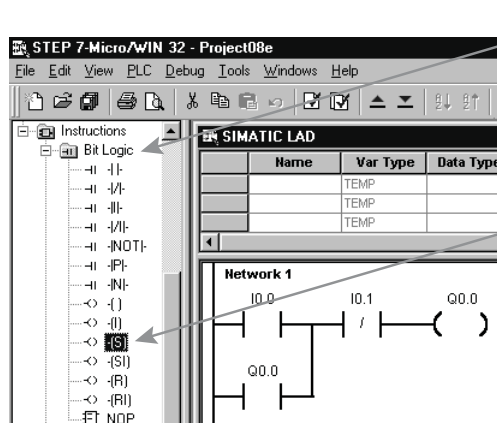
## 2.6 Making a Set/Reset network

To make sure you are using the least number of keystrokes and mouse clicks to create the LAD networks, do the following:

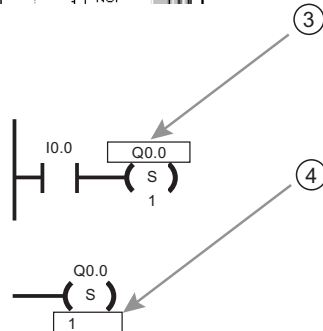
1. Select the menu command Tools > Options
2. Select the Program Editor tab
3. Verify the "Enable operand editing after placing instruction" checkbox is checked (enabled).



You have already learned how to enter the Normally Open contacts I0.0 and I0.1.

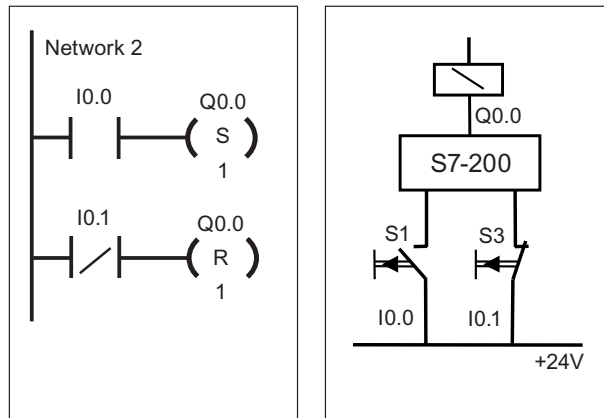


1. After marking the desired LAD field, select the Bit Logic folder with a single mouse click from the list for operation families.
2. Then double click "Set" (or "Reset") from the list of Bit Logic operations.
3. In the already selected address field, enter the output address you want, Q0.0 in this case.
4. Select the lower bit field and enter the number of bits to set starting from address Q0.0



## 2.7 Safety Tip: How to control a machine shutdown when a wire breaks

Because the circuit uses a NC switch as the input to the Reset logic, a broken wire on that input forces a reset.



Switch with NC contact that supplies the signal "0" when operated.

In LAD, this signal is reversed by the NC contact I0.1. This means that if you operate the switch S3, Q0.0 is reset.

### Note

#### Safety notes

- In the above example, an NC switch S3 was used for resetting.  
When I0.0 is operated, output Q0.0 is set with a steady state. If there is +24 V at I0.1, the "NC contact" supplies the state "0" in LAD. Output Q0.0 is not reset. The LAD "power flow" is interrupted, and the coil for resetting is deactivated. If there is no signal (0V) at I0.1 (S3 is open), the NC contact of I0.1 in LAD = "1" and the output is reset.  
When an NC switch is used at I0.1, the latching output Q0.0 is reset (switched off again) on either of the following conditions:
  - If switch S3 is operated (I0.1 = "0")
  - If there is a break in the connecting cable between I0.1 and the NC switch. Even in the event of wire break, it is guaranteed that a plant component will operate in a safe state, for example, a motor is switched off.
- The operation "Reset Q0.0" network is executed after the operation 'Set Q0.0'. This means that in the event of both switches being operated simultaneously, clearing the latch takes priority.

In STEP 7-Micro/WIN, open the exercise example 2hr\_pr\_02.mwp and test the functions!

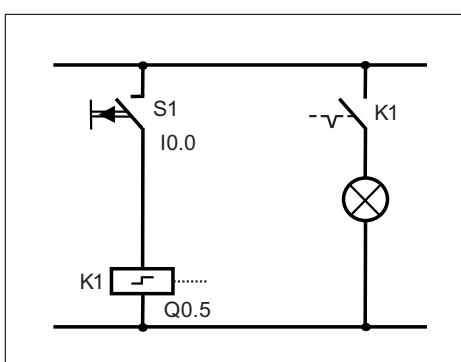


## Pulse Operated Switch

### 3.1 Introduction

You will implement a pulse-operated switch here. Within this context, you will learn about edge detection and bit memories.

#### Principle of Operation



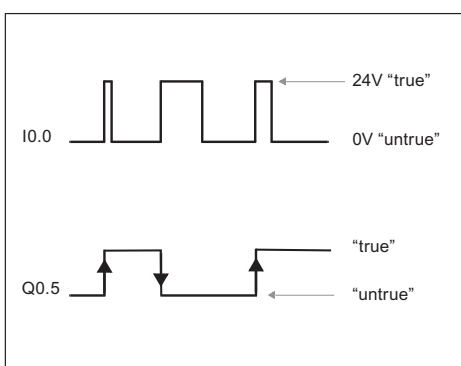
A lamp at output Q0.5 is to be switched on as soon as S1 at input I0.0 is momentarily closed.

If S1 (I0.0) is closed again, Q0.5 and the lamp are to go off.

Whenever switch S1 is operated, Q0.5 is to change its state.

This is a "pulse-operated switch".

#### Timing Diagram



Output Q0.5 is always to invert its current state when the switch at I0.0 changes from "open" to "closed".

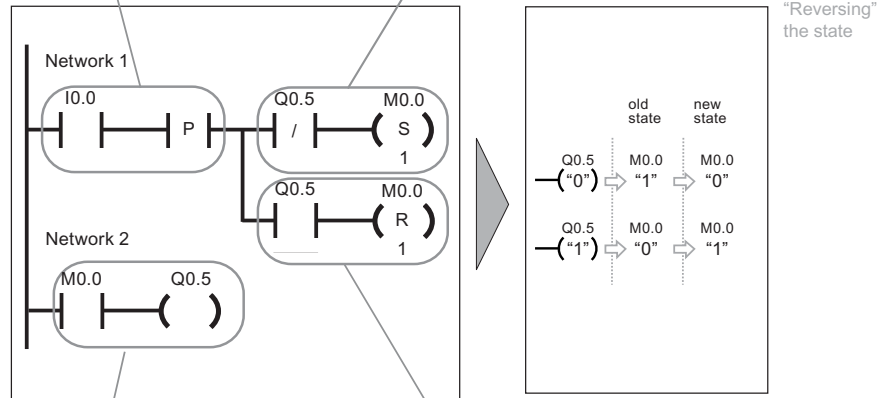
If the switch remains closed or open, no output change occurs.

### 3.2 Solution Overview

Before showing you the step-by-step solution of the task, we will show you the finished solution to provide you with an overview.

Detect whether a change of state from "0" to "1" (= positive edge) has taken place at I0.0.

If output Q0.5 is "0", bit memory M0.0 is set, this "flags" that Q0.5 in Network 2 is to become "1".

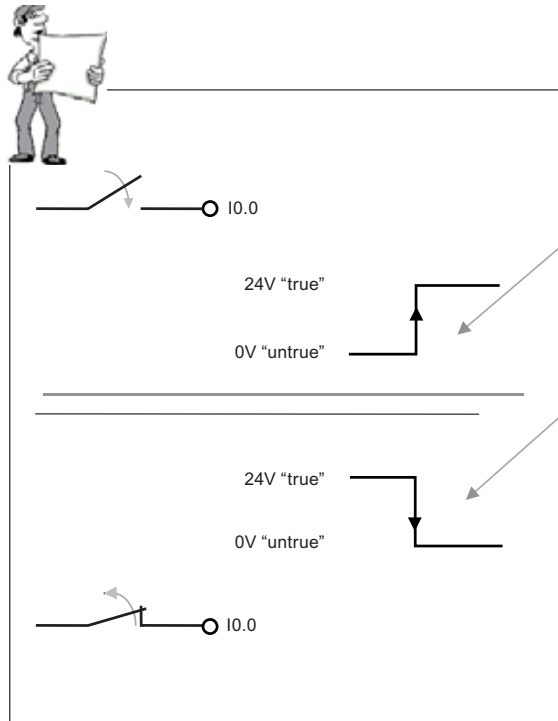


Assign the state of M0.0 to output Q0.5.

If output Q0.5 is "1", bit memory M0.0 is reset, this "flags" that Q0.5 in Network 2 is to become "0".



### 3.3 Edge detection

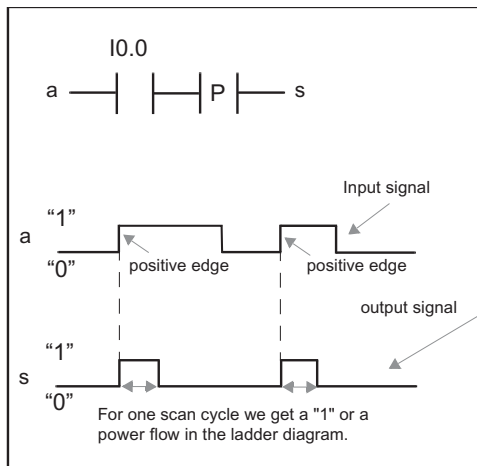


The moment of transition of a contact (input, output ...) from "open" to "closed" or from "false" to "true" is referred to as the rising or positive edge.

Correspondingly, the transition from "closed" to "open" or from "true" to "false" is referred to as the falling or negative edge.

The two Bit Logic functions -|P|- and -|N|- can detect rising and falling edges.

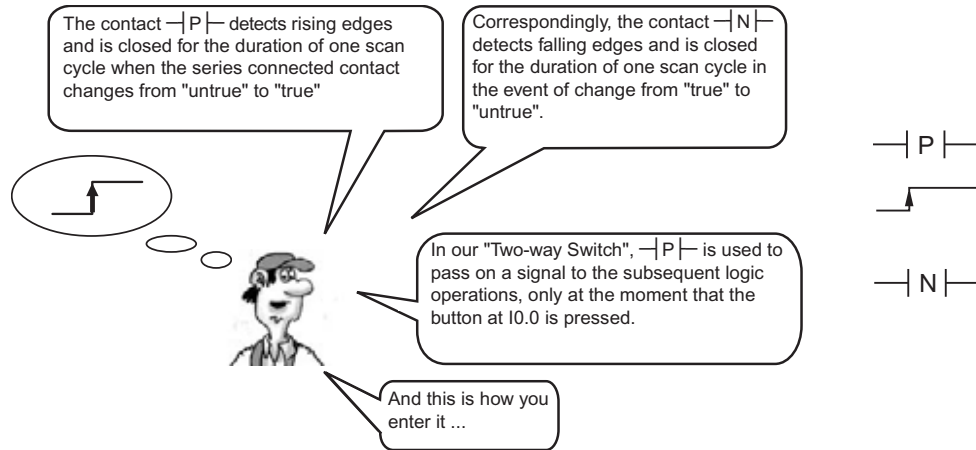
In our example, we use the -|P|- function as follows:



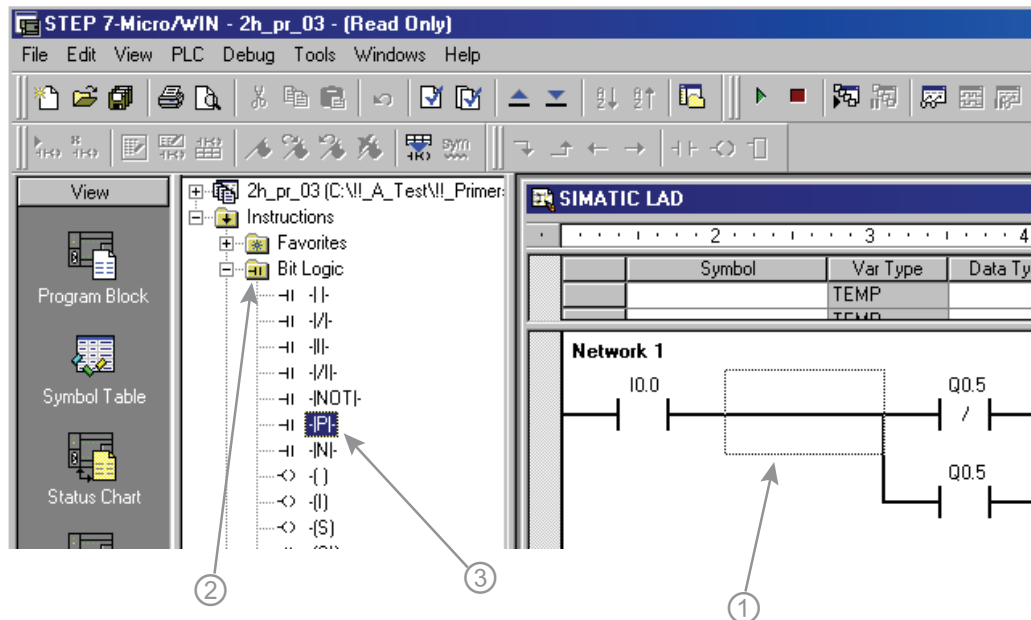
And this is what the -|P|- function output signal looks like.



### 3.4 Examples of edge detection



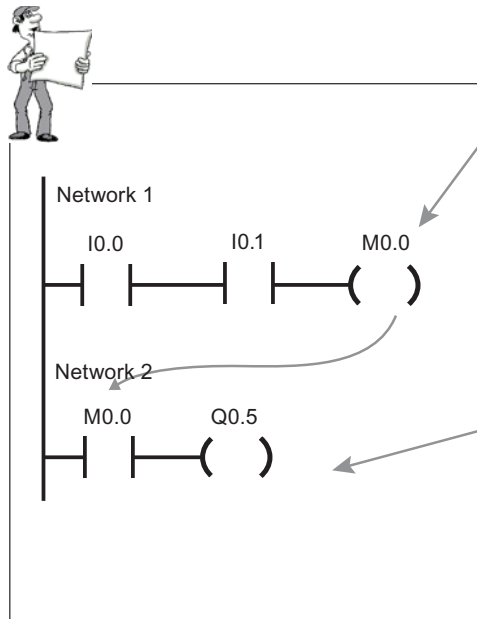
In STEP 7-Micro/WIN, open the exercise project 2h\_pr\_03.mwp from diskette. This project is also incomplete and will be finished step by step.



1. Use a mouse click to position the box cursor where the --|P|-- contact will be placed.
2. Select the Bit Logic folder with a single mouse click from the list for operation families.
3. Double click the Positive Transition or --|P|-- contact from the list that appears.

### 3.5 Bit memory (M addresses)

You can use bit memories for the pulse-operated switch. A brief example will serve here to show you how to work with them.



① Instead of being used as an output, the bit memory "M0.0" is used as a storage location within the PLC for the interim result of the logic operation "I0.0 AND I0.1".

② In this network, the bit memory is used as an "input NO contact" and so controls output Q0.5. The M0.0 bit memory address can still be used at any other location in the program.

Bit memories are used for storing interim results, as in the memory of a pocket calculator.

Bit memories can be used as outputs and can replace auxiliary contactors. A bit memory can be used as often as required at any location as a NC contact or as a NO contact.

The logic state of bit memory is immediately available (in the same cycle) for follow-on logic operations.

If the operating power is interrupted, bit memory contents are lost. The S7-200 "Retentivity" setting is designed to prevent this.

Memory is used if the (interim) result of a network is to be further processed in other networks (like sub-totals when adding numbers manually). They are also used to store follow-on states temporarily.



### 3.6 Pulse-operated switch

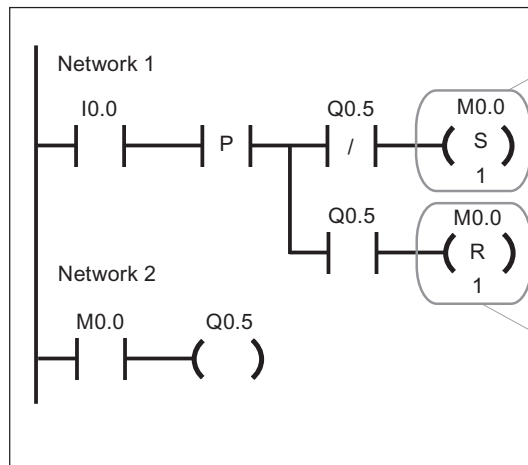
Now that you know the function of Bit Memory (M addresses), you will be able to understand the solution of the pulse-operated switch.

The  $\neg P$  function enables power flow (edge detection) in Network 1 for one scan cycle each time the contact at I0.0 closes.

Q0.5 is to change its state at each  $\neg P$  positive edge.

We do not write the inverted state (follow-on state) directly to output Q0.5, because the output would be set in the "upper" branch, then immediately reset again in the "lower" branch. For this reason, we write the follow-on state to bit memory M0.0 (to prevent overwriting).

In Network 2, the logic state of M0.0 bit memory is assigned to the output.



Place a coil here for setting bit memory M0.0. The number under the coil indicates how many bits are to be set counting from the specified starting address. 1 sets one bit at bit memory address M0.0.

M0.0 is set if Q0.5 was not active ("false")

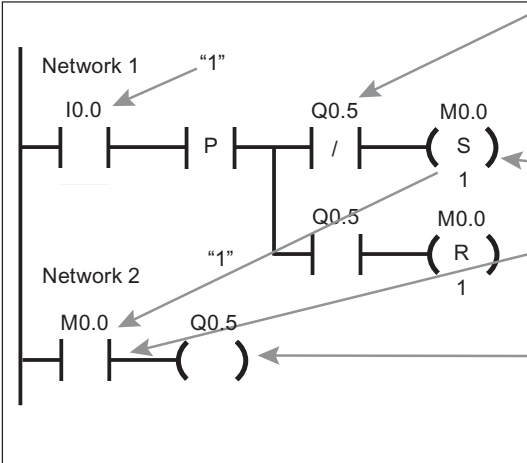
Since the lower branch implements the inverted function of the upper branch, bit memory address M0.0 is reset, or switched off, if power flows through this branch as the result of I0.0 contact closure.

M0.0 is reset, if Q0.5 was active ("true")

Finally, complete the example in your current exercise project in STEP 7-Micro/WIN as shown above.




### 3.7 Solution Description and Test

To summarize, the function of our now complete program is explained again below



If I0.0 is operated (P edge detection)  
**and**  
Q0.5 is "0" in the current cycle (upper branch is true on scanning with NC contact)  
**then...**  
Flag the follow-on state of Q0.5 by setting bit memory M0.0.  
M0.0 already has the follow-on state of Q0.5 here.  
Q0.5 is not assigned the new state until the end of the cycle and so does not appear as "true" or "1" in the LAD representation.



-  Save the completed program to hard disk.
  -  Transfer the program to the PLC.
  -  To test, switch the PLC to the "RUN" mode.
- Test your program: Operate the switch at I0.0 and observe output Q0.5.

## 3.8 Time to Show What You Know

### You've made some real progress!

Read and answer the questions below.

- What is the scan cycle of a PLC? What are the three main functions of the scan cycle? Section 1.6
- How is a latching function implemented in a PLC? Section 2.1
- Normally-closed contact: How is this represented in the ladder diagram, What effect does it have, which safety measures can be achieved using it? Section 2.2
- What is a logic signal edge, how is it detected and for what purpose? Section 3.3
- What are Memory Bits (M addresses), and what are they used for? Section 3.5
- How are the "Set" and "Reset" coils entered and what function do they have? Section 3.6

You're sure to know the answers to these questions, even if you have to look at the Sections again.



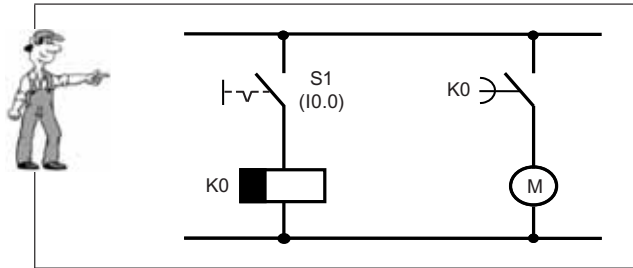
## Off Delay Timer

### 4.1 Introduction

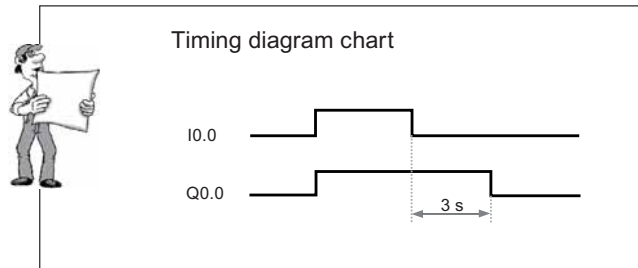


You are already familiar with the On-delay timer from the 1-Hour Primer. We will now implement an Off-delay timer together.

When S1 (I0.0) is operated, a fan motor at output Q0.0 is activated. If S1 (I0.0) is switched off, the fan continues to run for 3 seconds and then stops.



If S1 is switched off, the fan continues to run for 3 seconds.



## Procedure

In the next pages, we will work through these steps together to implement the off-delay timer safely.



1. First, load the complete latching circuit from our first example from the hard disk.
2. Then, save the example under a new name on the hard disk.
3. We will then work together to complete the off-delay timer with comments.
4. Finally, we will test the program together.



Let's get to work.



## 4.2 Save As...

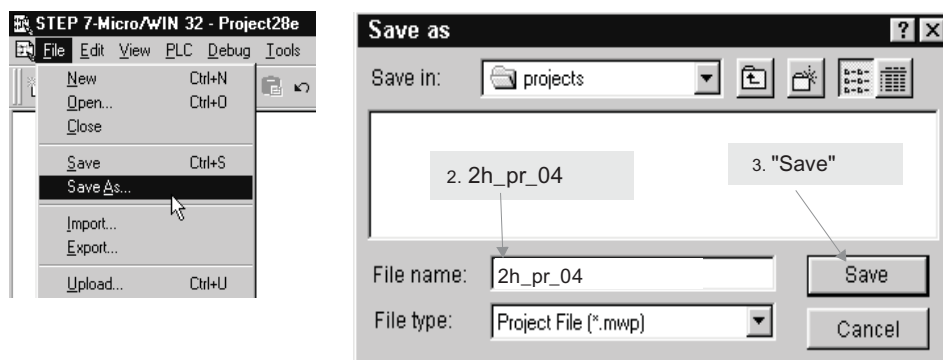
In STEP 7-Micro/WIN, load your project 2h\_pr\_01.mwp (latching circuit) from the hard disk. You stored it there in the first chapter.



We will use the latching circuit from the first chapter as the basis for our project. Duplicate the entire project by loading it and then immediately saving it under another name.

You can save the project under a new name. Save the project as described below under the new name 2h\_pr\_04.mwp.

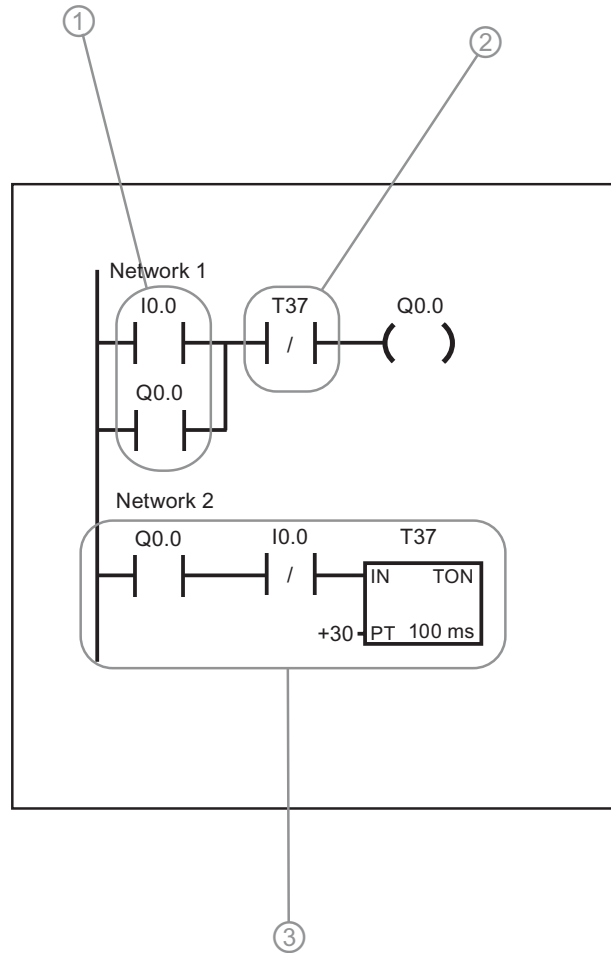
1. Select the "Project >Save As " menu command.



### 4.3 Solution Overview

I0.0 activates Q0.0  
Q0.0 maintains its state (latches)  
since it is also switched  
simultaneously in parallel with  
I0.0.

When T37 has elapsed, the latch function is  
broken via this contact.  
The motor stops.  
If T37 has not elapsed, the latch remains in  
effect.



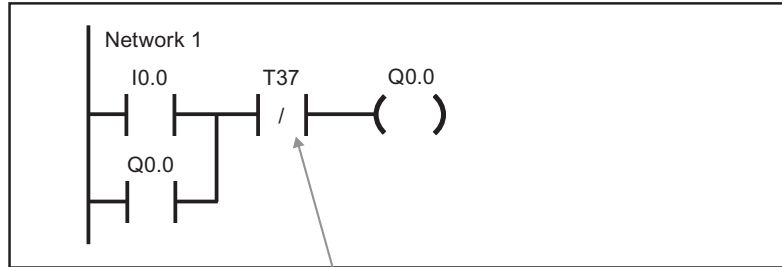
This is how the finished  
program appears..



When Q0.0 is ON (1) and I0.0 is OFF (0) again (S1 no longer operated), timer T37 starts to run.

## 4.4 Solution - Enter the Program

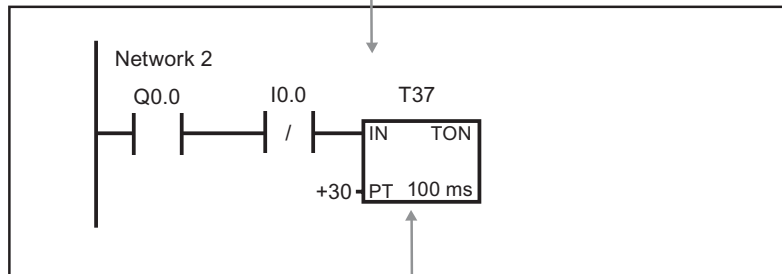
Network 1 must look like this:



Overwrite I0.1 of the latching circuit with T37.

Enter the following program in Network 2:

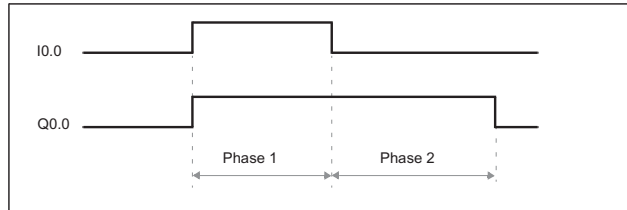
- Enter T37 with
- 1). F9 Shows a list of all box instructions
  - 2). Type TON and press the Enter Key



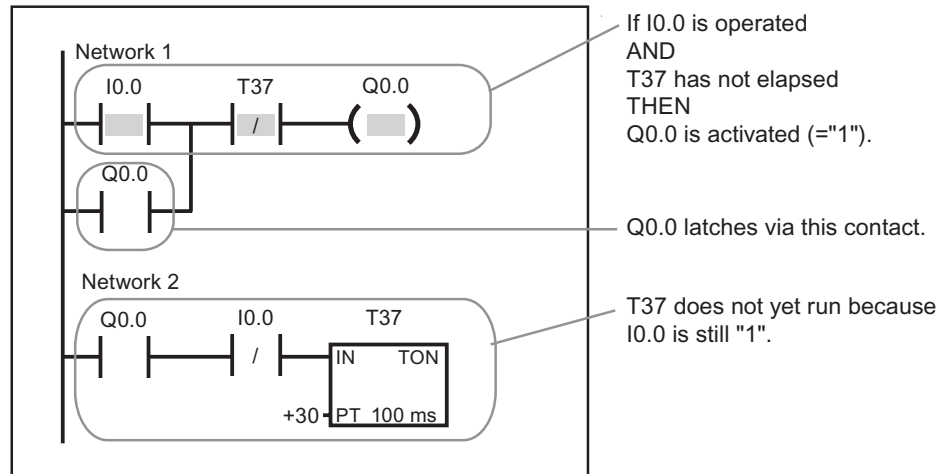
T37 has a timebase of 100 ms  
The time value is therefore  $30 * 100 \text{ ms} = 3 \text{ s}$ .

## 4.5 Solution Description

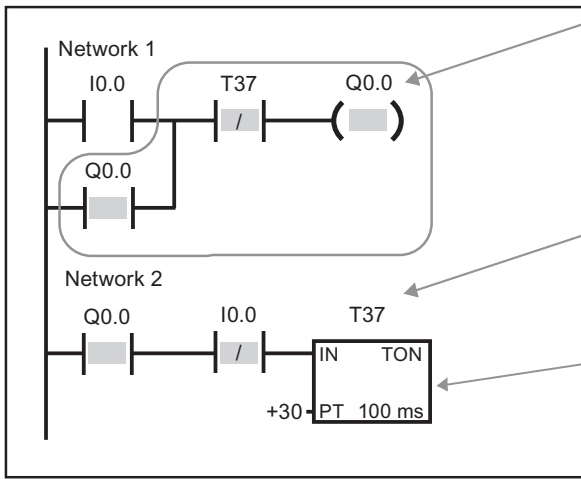
This is how our program functions. It has two active phases.



Phase 1: Activation of the latching circuit, I0.0 is "1" (we assume that Q0.0 is not active).



Phase 2:



I0.0 is no longer operated. The latch remains in force until T37 has elapsed. While the timer is running, T37 is "0" and the NC contact lets power through and keeps Q0.0 latched.

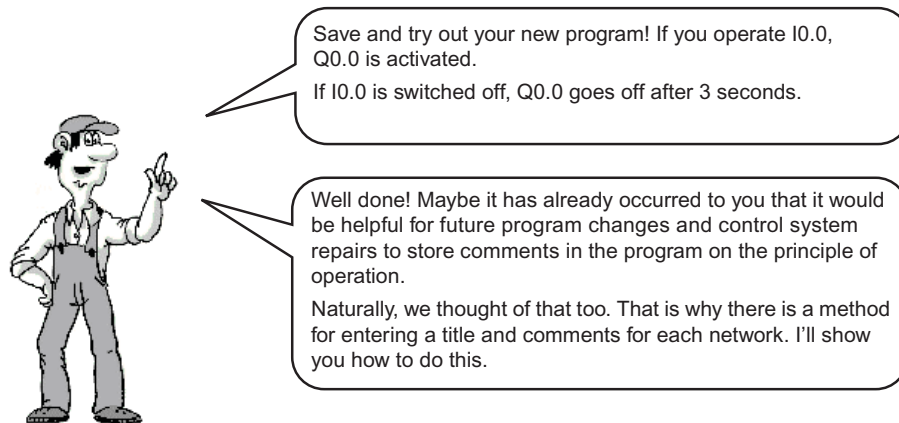
The current value of the timer can be monitored when you use the Debug > Start Program Status menu command.

If Q0.0 is active AND I0.0 is no longer operated, timer T37 runs.

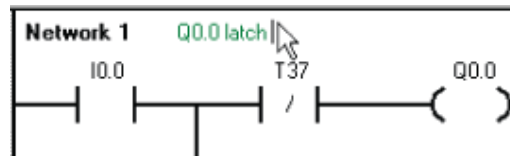


## 4.6 Entering comments

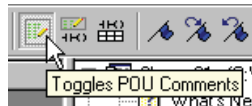
You can use comments to document your program.



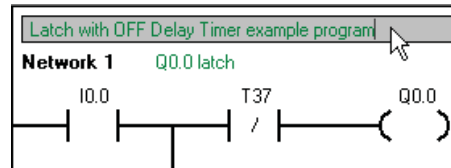
You can add a network title by using a mouse click to select the network title field. Then, type in your title text .



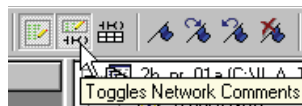
Use the View > POU Comments menu command or toggle ON POU comments with the toolbar button.



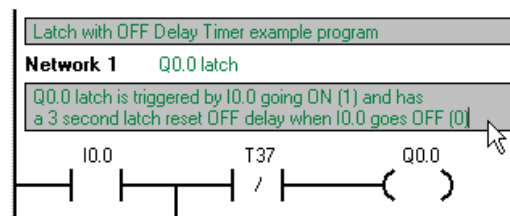
Select the first line of your program with a mouse click and then type in your POU comments.



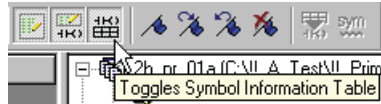
Use the View > Network Comments menu command or toggle ON network comments with the toolbar button



Select the network comment field with a mouse click and then type in your network comments.



Use the View > Symbol Information Table menu command or toggle ON symbol information tables with the toolbar button



If you already assigned symbol names and comments in the Symbol Table and you successfully compiled your program, then you can display the Symbol Information tables within the LAD editor. The symbol comments can be edited in the Symbol Table window or the LAD editor window. Double click the mouse while it is placed over a Symbol Information Table row in the LAD editor to edit symbol comments.

Latch with OFF Delay Timer example program

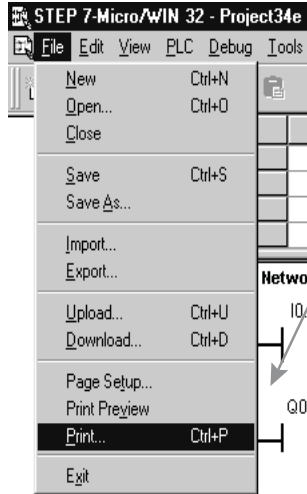
**Network 1**    Q0.0 latch

Q0.0 latch is triggered by I0.0 going ON (1) and has a 3 second latch reset OFF delay when I0.0 goes OFF (0)

The diagram shows a single network with two parallel branches. The top branch contains a normally open contact labeled 'I0.0' in series with a normally closed contact labeled 'Q0.0'. The bottom branch contains a normally open contact labeled 'Q0.0'. Both branches lead to a coil labeled 'Q0.0'. A timer symbol 'T37' is connected to the coil with a slash and a vertical line, indicating a 3-second ON delay timer.

Symbol	Address	Comment
Latched_Output	Q0.0	Positive Logic 0
ON_Delay	T37	3 Second TON
Trigger_Input	I0.0	Normally Open c

### Printing comments



If you want your program comments to be included in a program printout, then you must enable the comment views and see the comments in the LAD editor. What you see in the LAD editor is what you get in the printout.

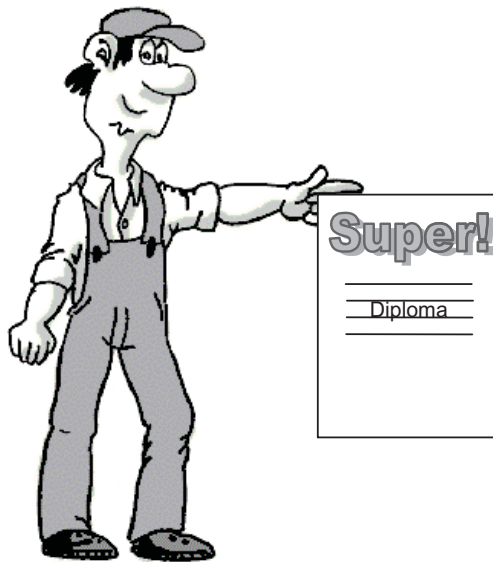




## 4.7 Time to Show What You Know

Please read and answer the questions below.

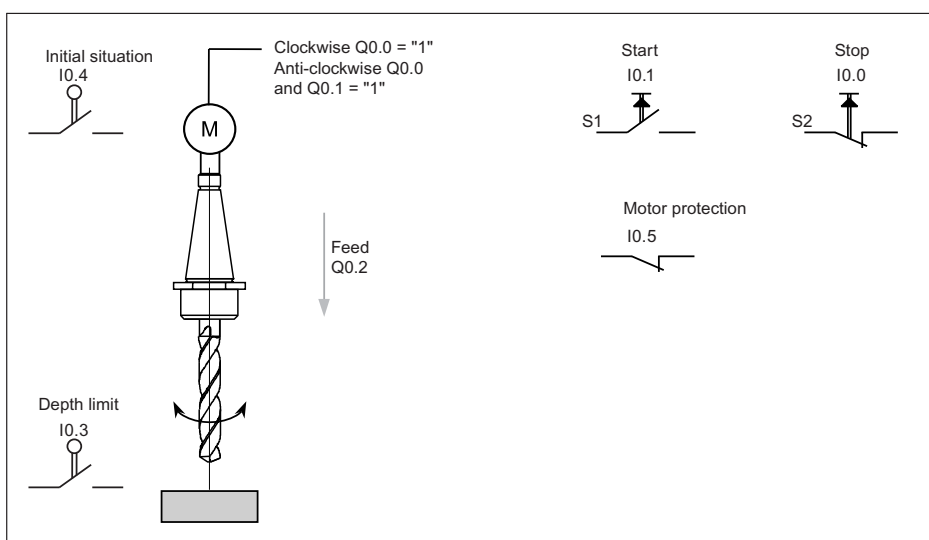
- How can you implement an off-delay timer? Draw the ladder diagram for two possible solutions: Once with the normal coil —( ) and once with (S) and (R).
- How do you save a project?
- How do you determine the value of a timer?
- What comments can be made to your program?





# Sequencer

## 5.1 Introduction



Now we will implement a sequencer together.

A drill motor is started in the clockwise direction with S1. After 3s, the feed is activated.

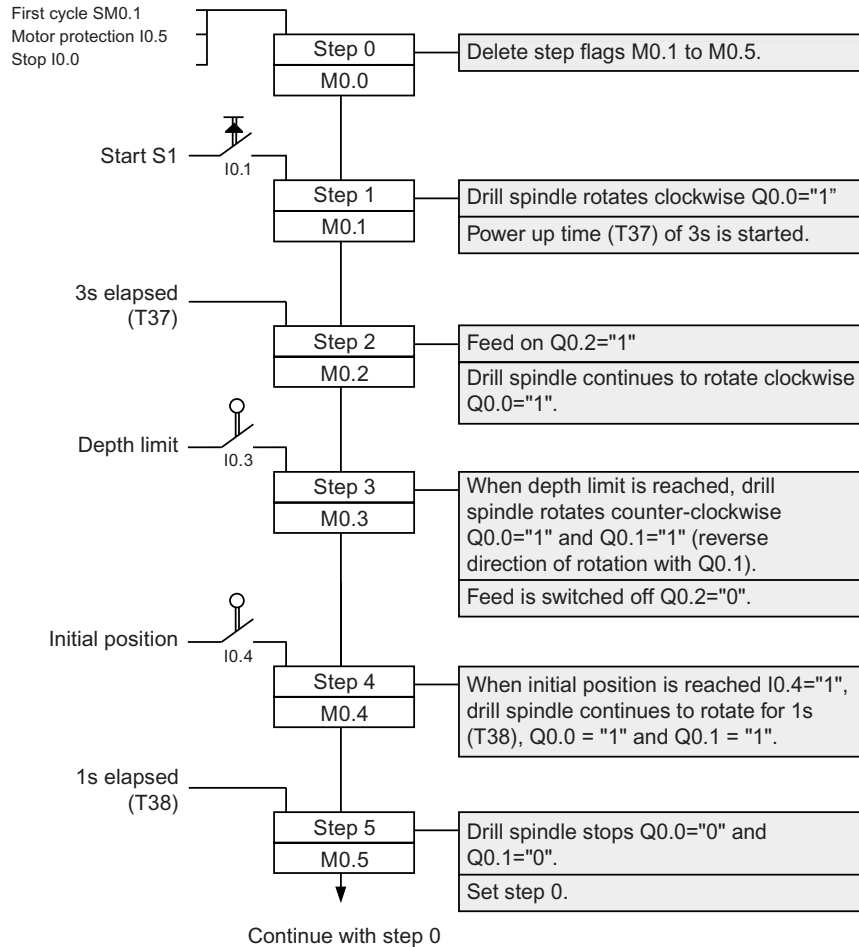
When the depth limit at I0.3 is reached, the feed is deactivated. A spring returns the drill to the initial position. In doing so, the drive turns counter-clockwise (Q0.0 and Q0.1 are "1").

When the initial position I0.4 = "1" is reached, the drive continues to operate for 1s until the drill is fully switched off. The drill can always be switched off with Stop (activation with I0.0 = "0").

## 5.2 Solution Starting Point



This is what the solution for the sequencer of the drill example looks like.



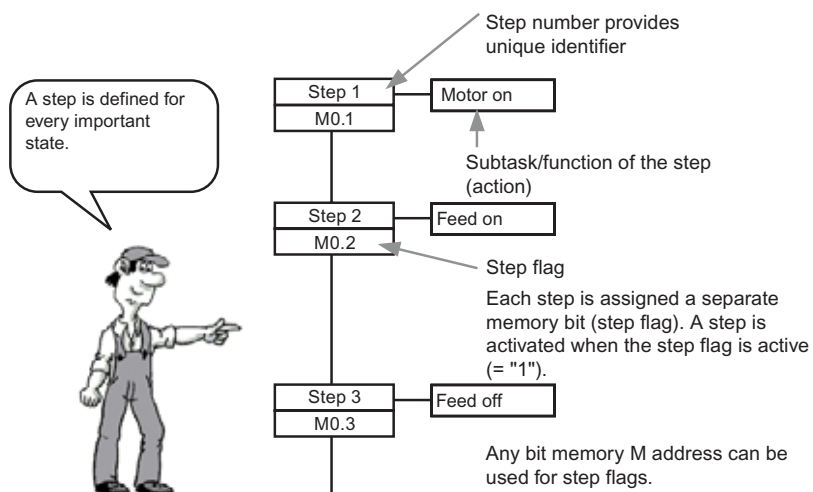
## 5.3 Sequencer control

We will now solve the drill example with a sequencer.



### What is a sequencer control?

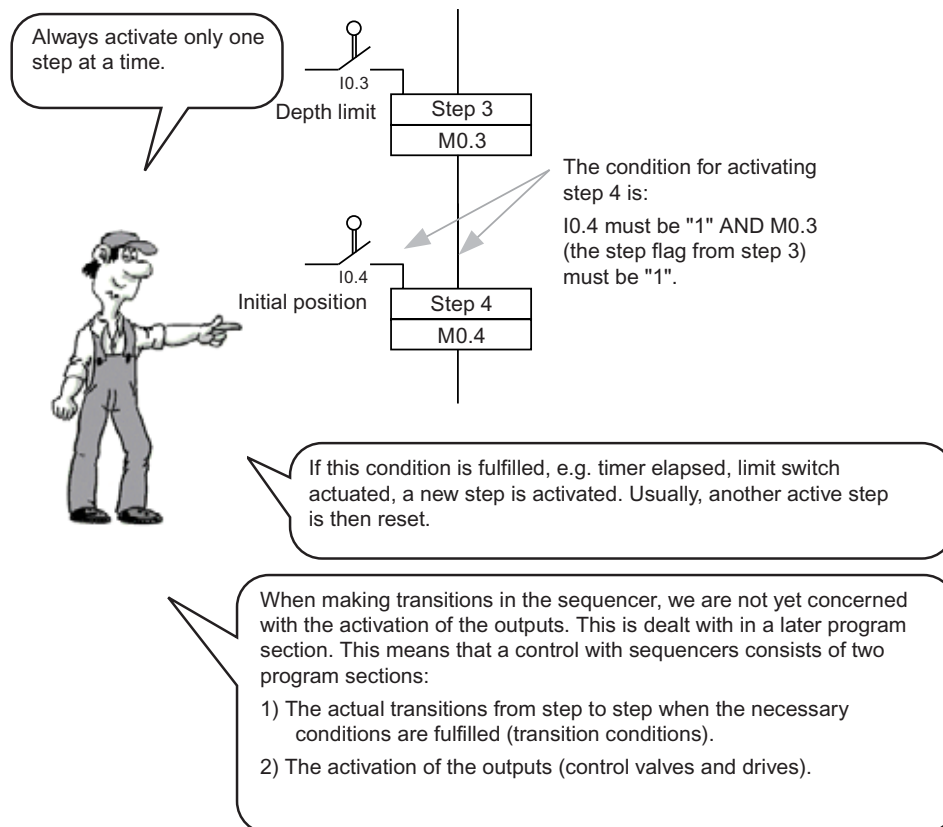
- A control method in which a task is broken down into very small, usually sequential, subtasks (such as Motor on, feed on, or feed off).
- The subtasks (functions) are called steps.
- Usually one step has to be completed before the next one is started.
- A new step becomes active when the relevant transition condition is active.
- A step is active when the associated step flag, for example: M0.1 = "1".



## 5.4 Transition condition

### What is a transition condition?

- Each step is started (activated) by a condition. The condition is usually derived from the states of the machine. These can include actuated limit switches, operator keys, temperatures reached or timers.
- An active preceding step is almost always part of the condition.
- If a new step flag is set, the step flag of the preceding step is reset.



## 5.5 Structure of the sequencer program

### The two program sections of a sequencer control

1. The conditions for activating the individual steps (sub-tasks) are logically combined with the individual step flags.

If flags M0.1... become active in sequence, the entire sequencer is processed.

**This defines the overall sequence of the sub-tasks.**

2. The active memory bits are assigned to the outputs of the PLC which then control contactors or valves, for example.

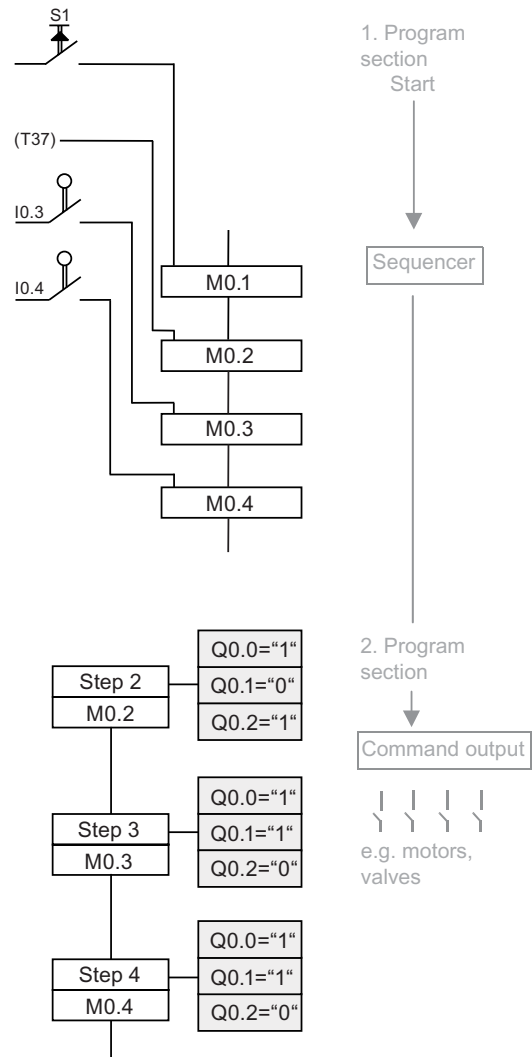
**This is the interface to the plant /machine.**

Start S1 I0.1,  
3s delay,  
depth limit I0.3,

initial position  
I0.4

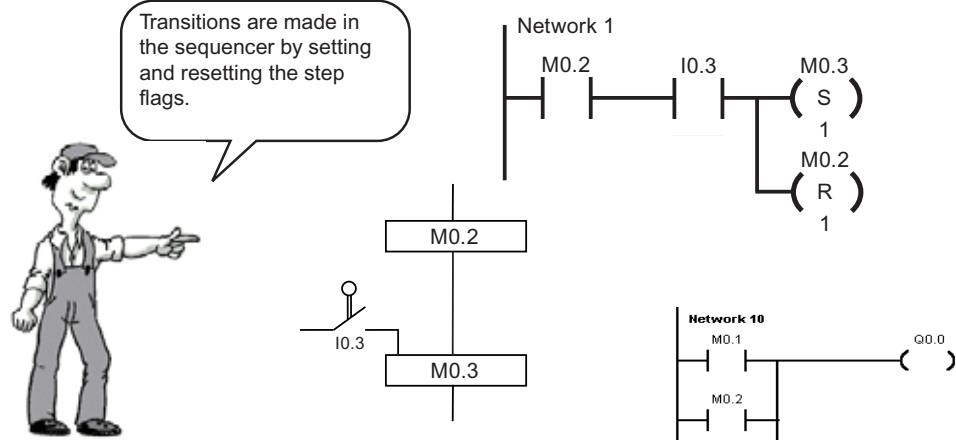
Step flags  
M0.1, M0.2,  
M0.3, M0.4

Q0.1, Q0.2, Q0.0

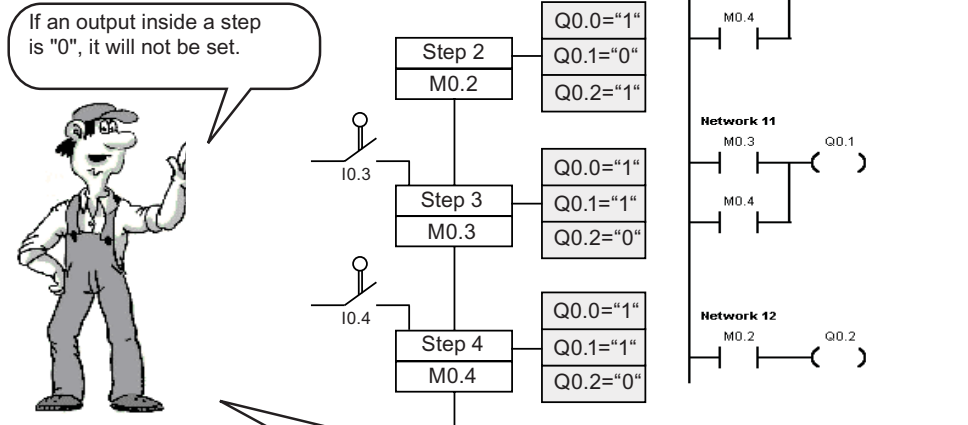


## 5.6 Control logic for the sequencer

### 1) Controlling the sequencer/making transitions in the sequencer



### 2) Setting the outputs via the step flags



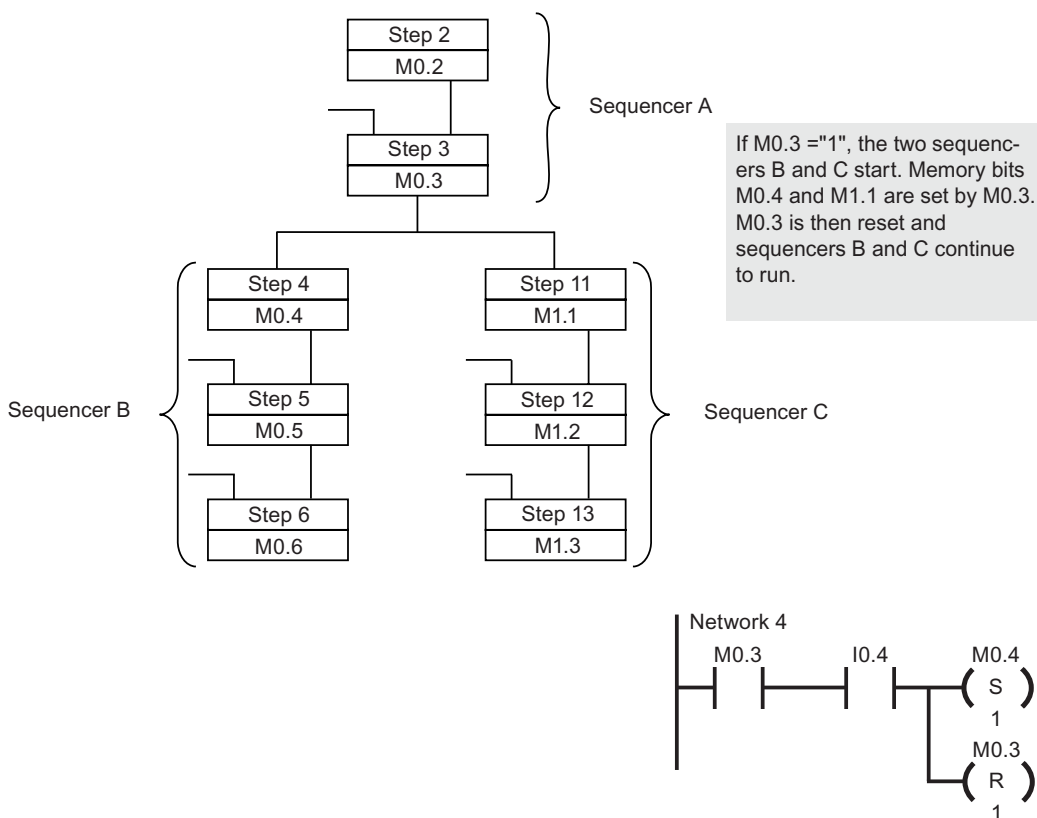
Outputs are set only by the step flags. Assigning outputs with a normal coil  $\text{—}(\ )$  ensures that the output is activated only in the one given step.

If an output has to be "1" in several steps (e.g. Q0.0), the step flags are configured using "OR" logic and assigned to the output.



## 5.7 Working with memory bits

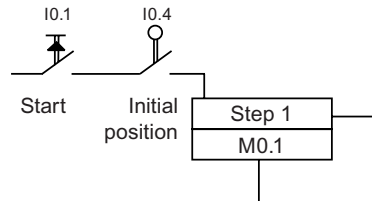
- A separate memory bit (step flag) is assigned to each step. This is "1" if the step is active.
- For the sake of clarity, only one step in a sequencer should be active at any time. This means only one step flag should be "1".
- If the task is more complex, it is best to use an additional sequencer.
- If two or more processes must be controlled simultaneously and independently, separate sequencers are used. See the following diagram.



## 5.8 Transition condition for the sequencer

The transition condition is in practice also made up of several contacts.

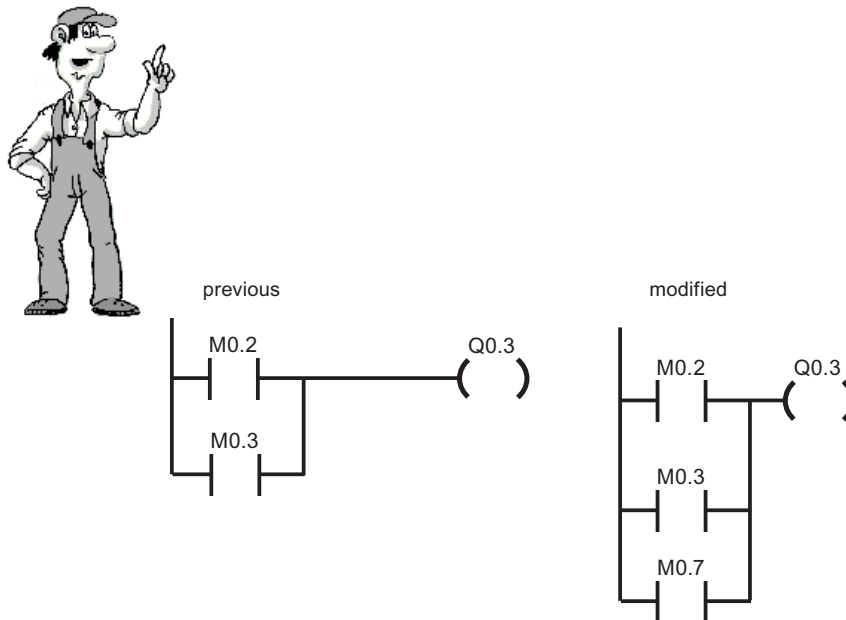
Our example can be expanded in such a way that, for example, the start can only take place if the drill is in the initial position. The sequencer then looks like this at this point:



## 5.9 Advantages of working with sequencers

### Separate sections for the step sequence control and the output control

If an output is now to be active in step 7 in addition to step 2 and 3, the program only needs to be modified at one point.



Modifications to the control section of the sequencer do not affect the setting of the outputs.

### Easy-to-test program

The program is easy to test:

- Each step can be traced easily on the programming device.
- If transitions do not function, it is easy to detect which condition is missing.

### Detecting missing transition conditions for machines

If a machine ceases to operate, it is easy to detect the missing transition condition from the mechanical position of the machine and the active step flag.

### Fewer programming errors and faster start-up

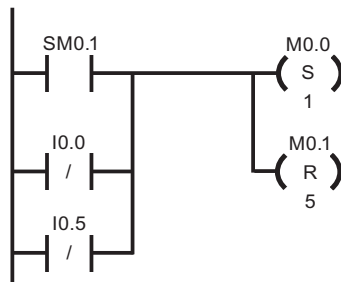
Using sequencers forces you to structure your programs which in turn minimizes programming errors.

## 5.10 Important safety considerations



There should be no drives or valves active in the first step flag (initial position). In our example, this is step 0 or step flag M0.0.

When "STOP" is operated or a motor protector is active, the first step flag (M0.0 in our example) need only be set for all drives to come to a stop. At the same time, all other step flags must be reset.



M0.0 is set, M0.1 to M0.5 are reset by any of the following conditions:

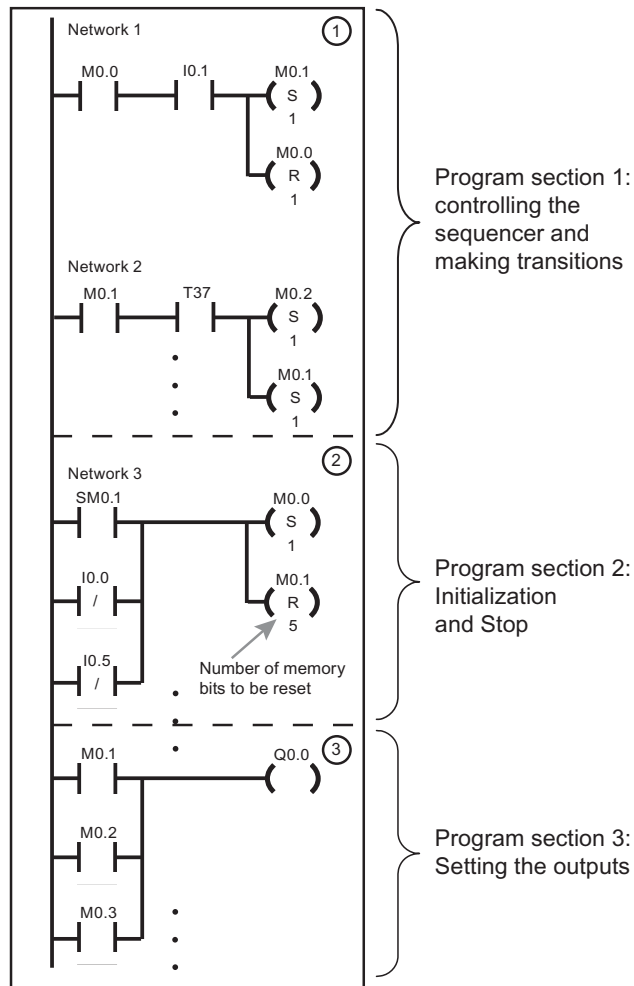
- In the first cycle by SM0.1, after PLC power is restored
- If I0.0="0"
- If I0.5="0"

The program section shown in the example must be at the end of the "normal" transition conditions of the sequencer. This ensures that any necessary shutdown can take place prior to activating the outputs.

## 5.11 Making safe transitions

Before assigning the first output, the program section for activating the initial position must be in place. This ensures that activation of the initial position has the highest priority.

Program section 1 – Making transitions in the sequencer:



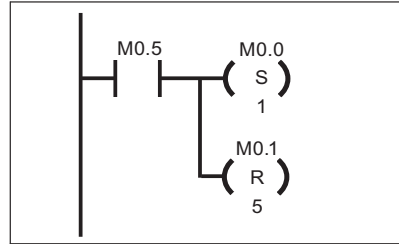
## 5.12 Modification



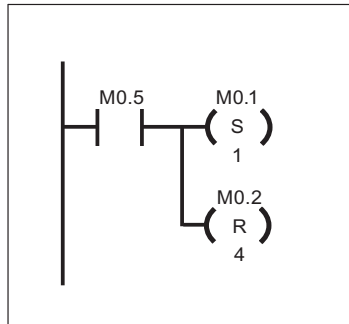
Network 6 determines in which step the program jumps to step 5. In the example, it jumps in step 0.

This is controlled by:

Setting M0.0 and resetting M0.1 to M0.5.



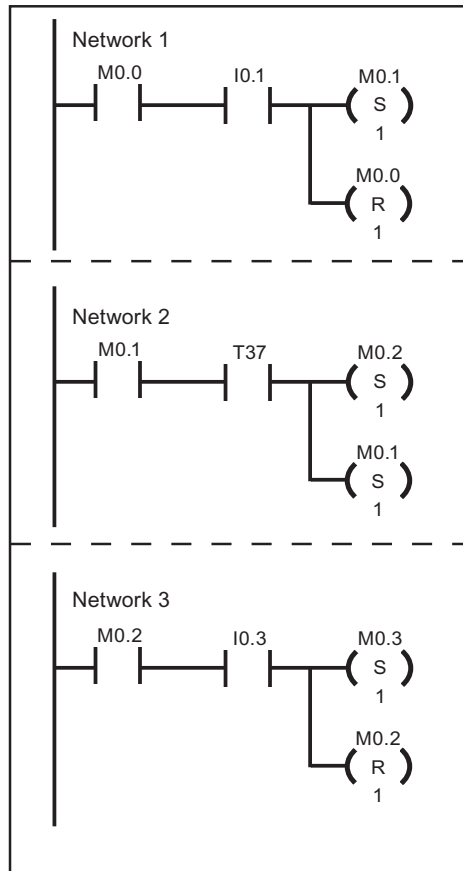
If the program is to jump automatically to step 1 following step 5, Network 6 must look like this.



This modification causes the drill to run automatically until stopped by I0.0 or I0.5.

## 5.13 Solution description: Transitions

### Program section 1 - Making transitions in the sequencer



#### Activating step 1

Step flag M0.1 is set when the sequencer is in the initial position (M0.0 = "1") AND I0.1 is operated. At the same time, M0.0, the step flag of the initial position, is reset.

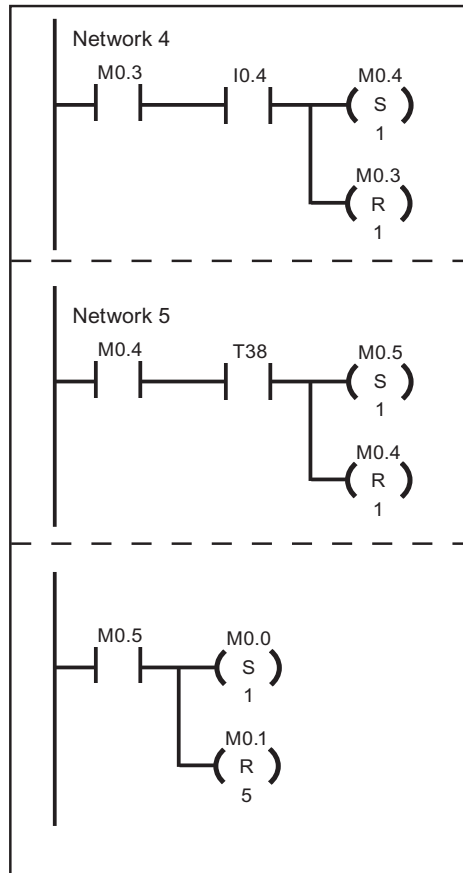
#### Activating step 2

Step flag M0.2 is set if the sequencer is at step 1 (M0.1 = "1") AND timer T37 has elapsed. At the same time, step flag M0.1 is reset.

#### Activating step 3

Step flag M0.3 is set if the sequencer is at step 2 (M0.2 = "1") AND input I0.3 depth limit becomes "1". At the same time, M0.2 is reset.

## 5.14 Solution description: Activating the steps



### Activating step 4

Step flag M0.4 is set if the sequencer is at step 3 (M0.3 = "1") AND input I0.4 (initial position) becomes "1". At the same time, M0.3 is reset.

### Activating step 5

Step flag M0.5 is set if the sequencer is at step 4 (M0.4 = "1") AND timer T38 has elapsed. At the same time, step flag M0.4 is reset.

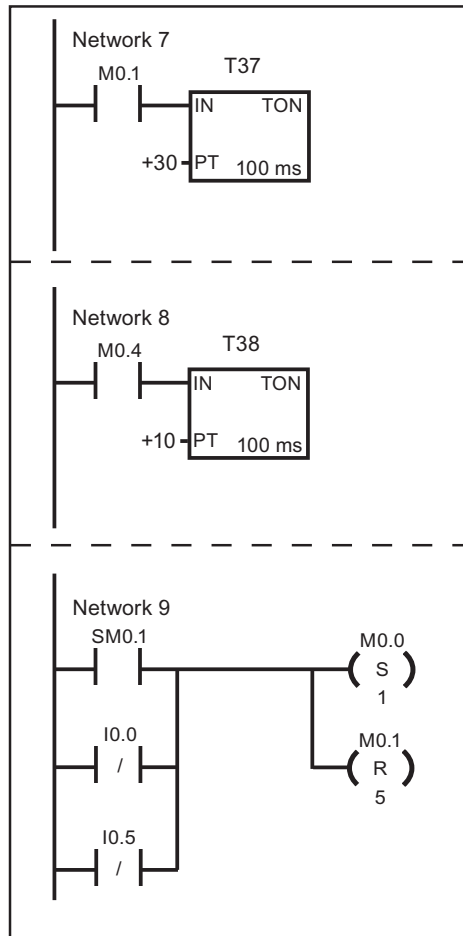
### Activating step 0

If step flag M0.5 is active (overshoot time T38 is finished), step 0 (initialization step) is activated from the sequencer. This step in Network 6 has been included deliberately so that further conditions such as removal of the work piece could be scanned at this point before re-activation of step 0.

This condition would then have to be switched in parallel to contact M0.5.



## 5.15 Solution description: Activating a timer

**Activating timer T37**

If step 1 is active (M0.1 = "1"), timer T37 is started.

**Activating timer T38**

If step 4 is active (M0.4 = "1"), timer T38 is started.

**Initialization of a sequencer**

Step flag M0.0 is set

In the first cycle (SM0.1 is "1" here for one scan cycle)

OR

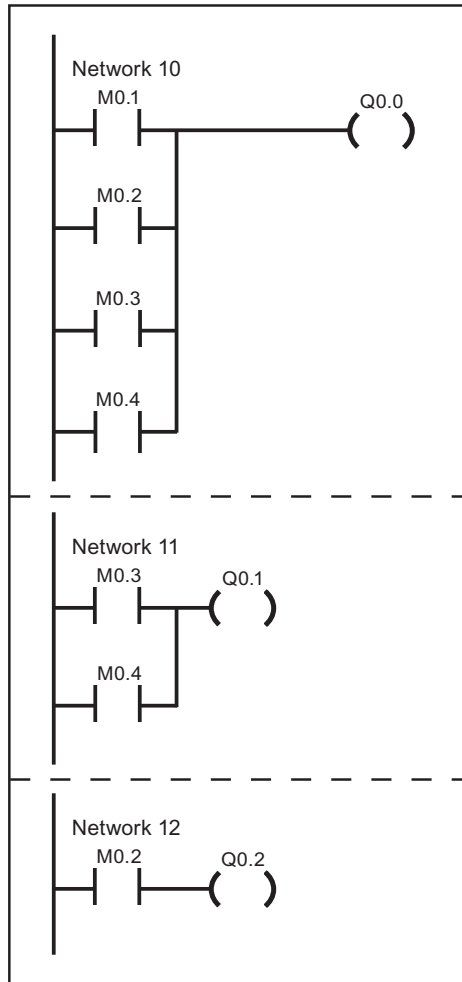
if Stop is operated (I0.0 = "0")

OR

if the motor protection is active (I0.5 = "0"). At the same time, step flags M0.1 to M0.5 are reset.

## 5.16 Solution description: Activating the outputs

Program section 2 - Setting the outputs



### Activate output Q0.0 (drive clockwise)

Output Q0.0 is "1" in steps 1, 2, 3, 4, i.e. if M0.1 or M0.2 or M0.3 or M0.4 are "1".

### Activate output Q0.1(direction reversal)

Output Q0.1 is "1" in steps 3 and 4, i.e. if M0.3 or M0.4 are "1".

### Activate output Q0.2(feed on)

If memory bit M0.2 = "1" output Q0.2 will become "1".

## 5.17 Test

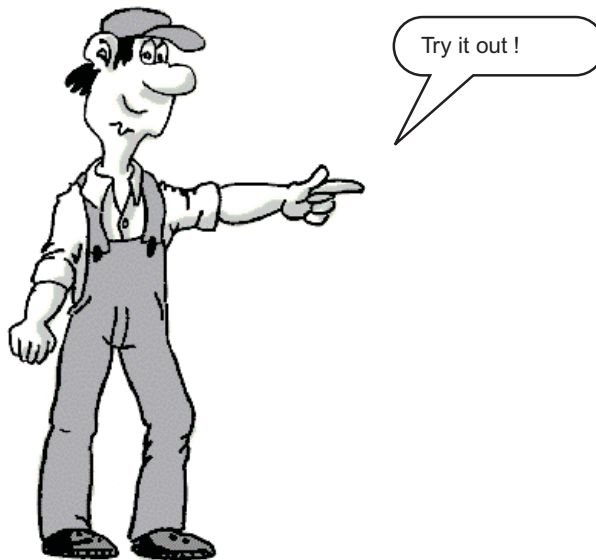
You can enter the program yourself or load the file 2h\_pr\_05.mwp from the Programming Exercises CD. Please note that the stop switch I0.0 and the motor protection I0.5 are "normally-closed (NC) contacts". This has been implemented in this way for safety reasons. A broken wire between the switches and the PLC stops the machine!

I0.5 and I0.0 must be "1" for test purposes, that is, the input LEDs must light up.

Briefly operating I0.1 starts the drive. The feed Q0.2 switches on after 3 s. After I0.3 is operated, the drive reverses its direction of rotation and the feed Q0.2 stops.

If the initial position is reached (brief operation of I0.4), the drive stops after 1s. I0.0 and I0.5 stop the drive in every phase.

Use the Debug > Start Program Status menu command to watch the program run. You will see exactly which input is required in each case for making the transitions in the sequencer.





We made it.

Now you can solve tasks yourself using the SIMATIC S7-200. If you want to implement complex contactor circuits, you can find some useful tips in the Appendix.

## Learning More

### 6.1 Do you Want to Learn More?

Use the Help > S7-200 on the Web menu command to access Siemens S7-200 support on the Internet. You can find more examples in the "Samples" sub-directory in your STEP 7-Micro/WIN directory.

In addition, a CD ROM with "Tips & Tricks" for the S7-200 is available. You can obtain the "Tips & Tricks" from your SIMATIC representative. You can find additional information in the S7-200 manuals. For further training you can attend a S7-200 course at your Siemens Training Center or from your SIMATIC representative.



Please contact the SIMATIC representative from whom you purchased the Starter Kit. If you cannot contact your SIMATIC representative, please call the Siemens Technical Support:

Worldwide (Nurnberg): 49 (180) 5050-222

United States (Johnson City):

1 (423) 262-2522

1 (800) 333-7421 (USA only)

Asia/Australia (Beijing): 86 10 64 75 75 75

We have put together a few examples to make it easy for you to implement even complex "switching operations" in ladder logic.



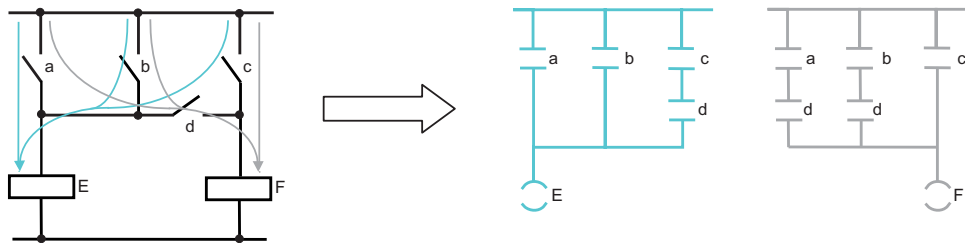
## Appendix A

### A.1 Bridge Circuit

If you are changing over from contactor technology to PLC technology you will probably encounter switch combinations that cannot be converted directly into ladder diagram representation. Included among these is the bridge circuit. Solutions are provided here both for the simple and the more complex bridge circuit.

#### Simple bridge circuit

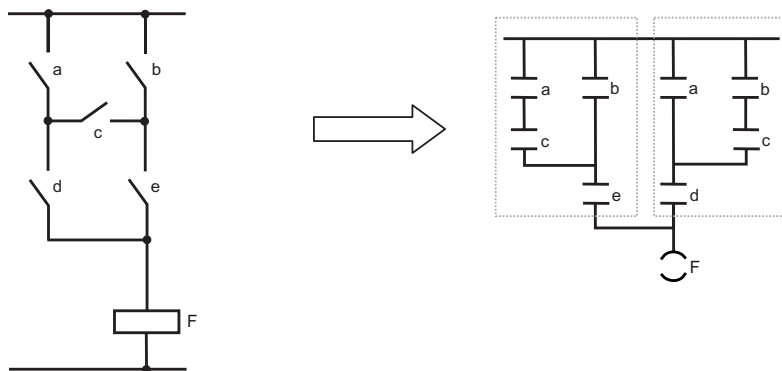
The simple bridge circuit (left) is implemented with two networks. The individual possible current paths are simply split up. For ease of comparison, we have likewise arranged the ladder diagram vertically.



#### Complex bridge circuit

The two possible current paths have been converted again and recombined. On the one hand, a,c parallel b, on the other b,c parallel a. For ease of comparison, we have arranged the ladder diagram vertically.

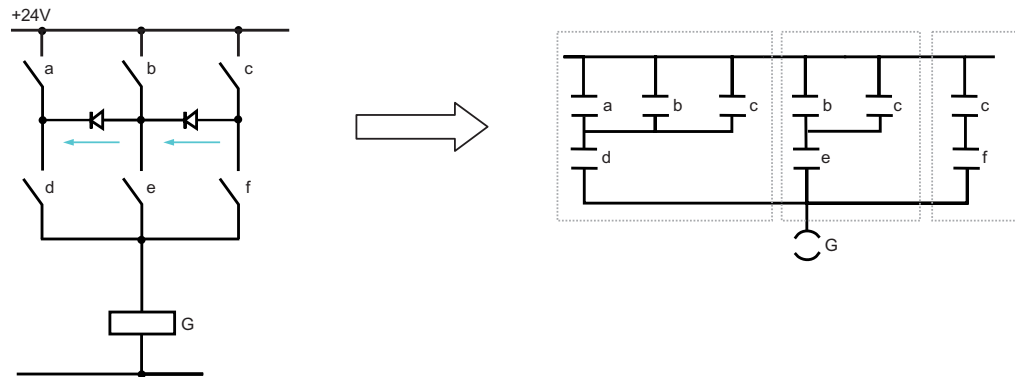
In new projects, avoid using the bridge circuit in the circuit diagram where possible! Think "in ladder diagram" right from the start.



## A.2 Diode Circuit

When diodes have been used in "old" circuit diagrams converting them into ladder diagram terms is not an altogether simple matter.

Since diodes represent connection lines in principle but only conduct current in one direction, a similar solution is adopted here as with the bridge circuit. For ease of comparison with the circuit diagram, the ladder diagram is arranged vertically again.



Three current paths are possible with this circuit: Over switch d, switch e and switch f.

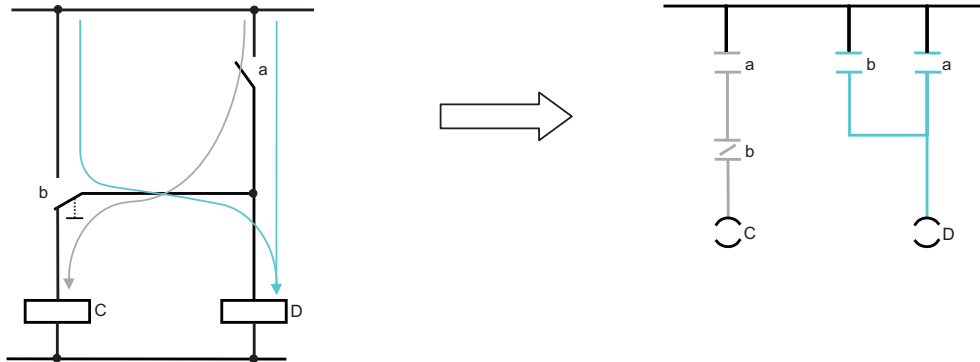
The current through the diodes can only flow from b to d or from c to e.

The three current paths result in the three framed sub-networks in the ladder diagram solution. Since switches d, e and f are on the same rail as output G, these three sub-networks have also been linked to form one network.



### A.3 Changeover Switch

Changeover switches should likewise not cause you any problem when you are converting a circuit diagram into a ladder diagram. This transformation is explained below.



The current paths are highlighted.

Changeover switch b is then divided into a normally closed (NC) contact that is switched in series and contributes to the effect at output C, or a normally open (NO) contact that takes effect in parallel with a and switches D.

In principle it is possible to convert a changeover switch using an NC contact and an NO contact with the same input address in the ladder diagram.