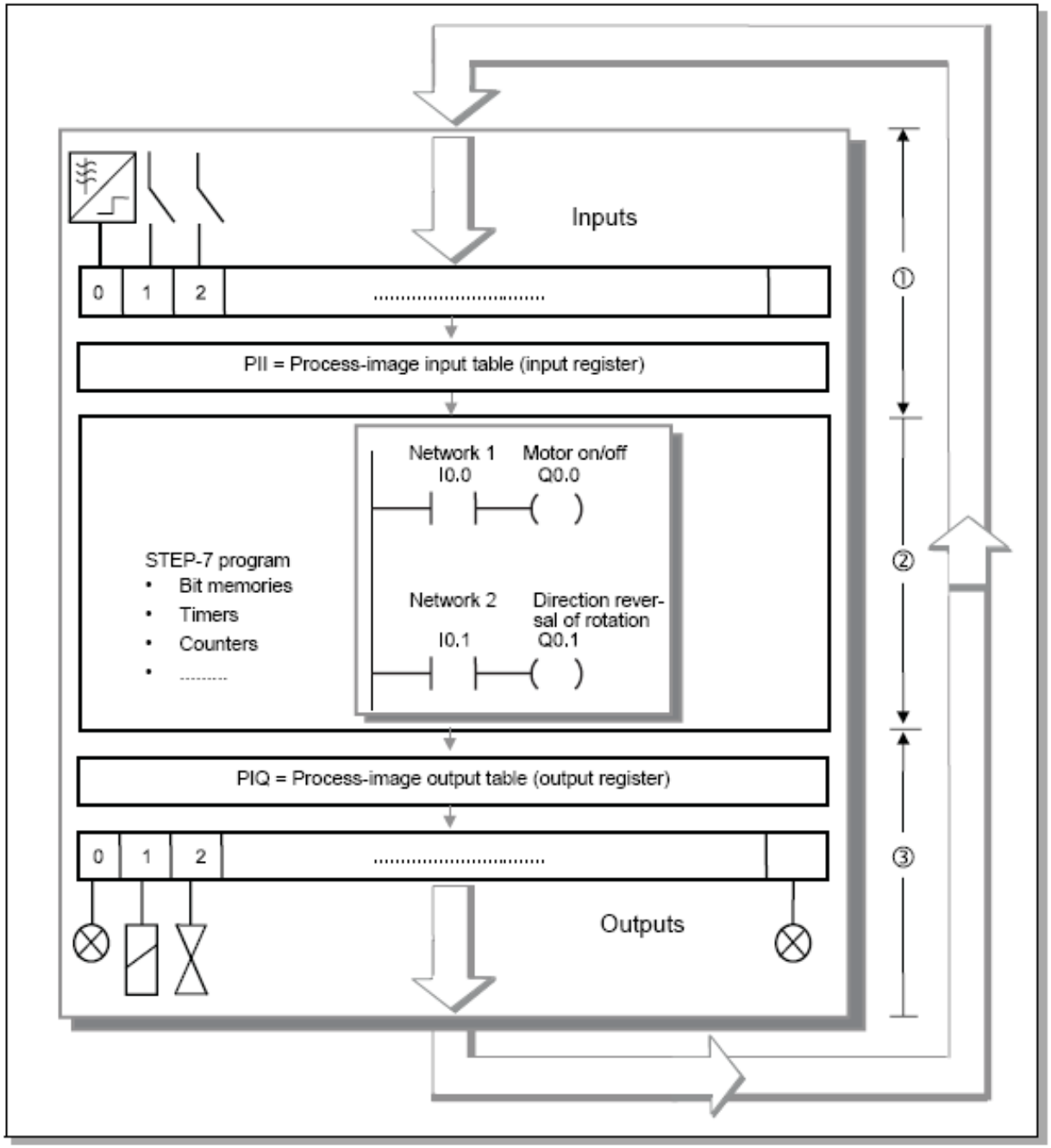
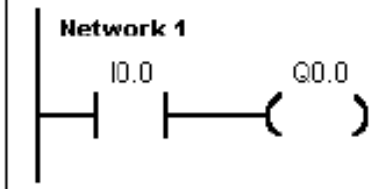
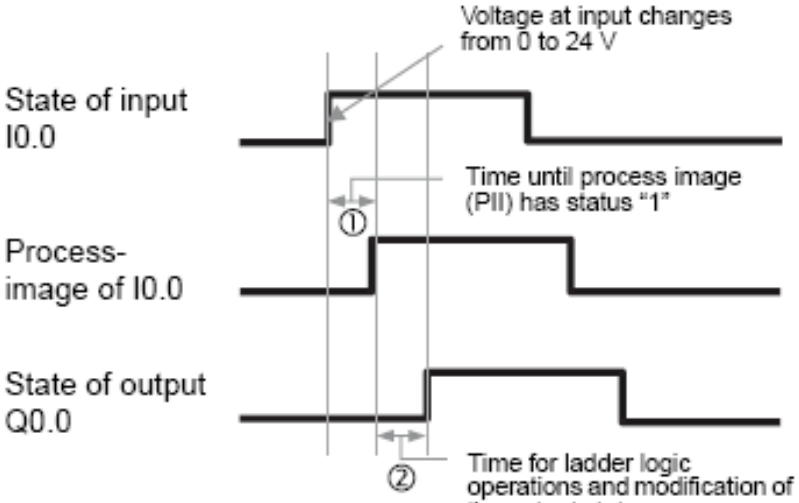
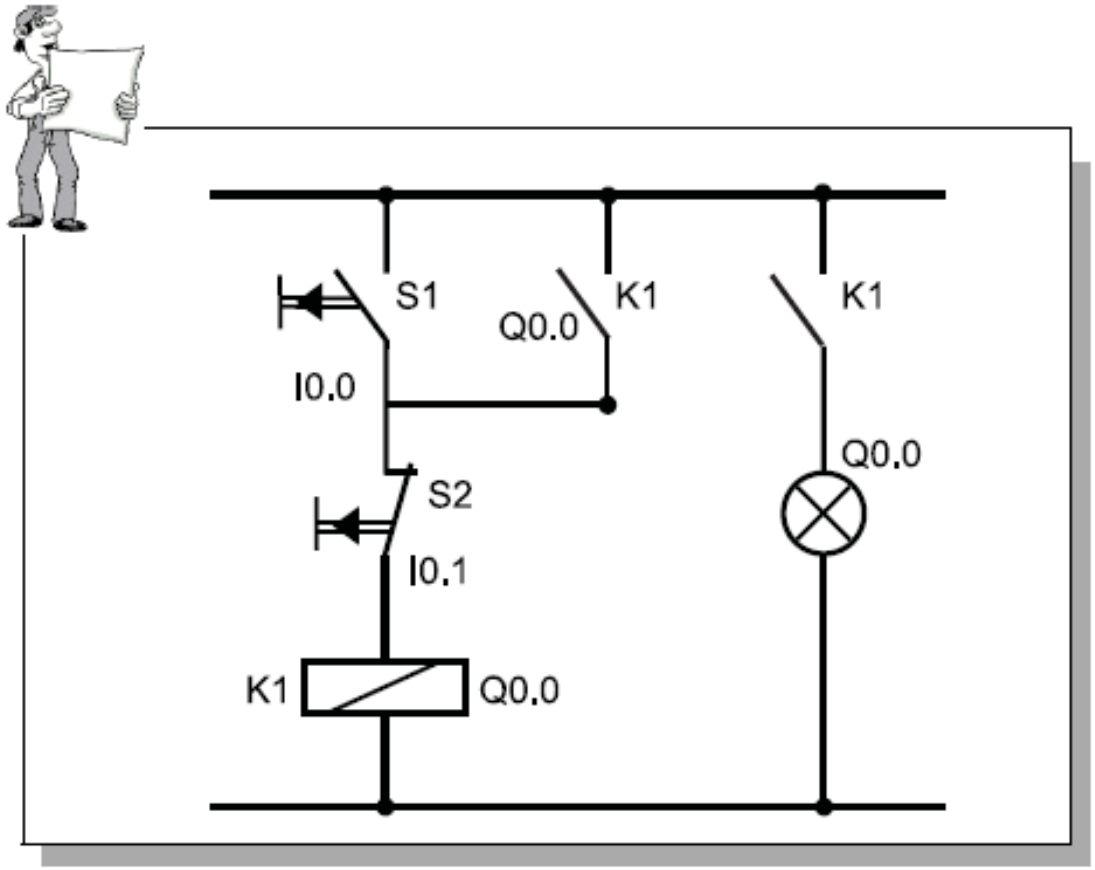

Inputs
0
1
2
PII = Process-image input table (input register)
Network 1
Motor on/off
I0.0
Q0.0
STEP-7 program
• Bit memories
• Timers
• Counters
•
Network 2
Direction reversal of rotation
I0.1
Q0.1
PIQ = Process-image output table (output register)
0
1
2
Outputs
①
②
③

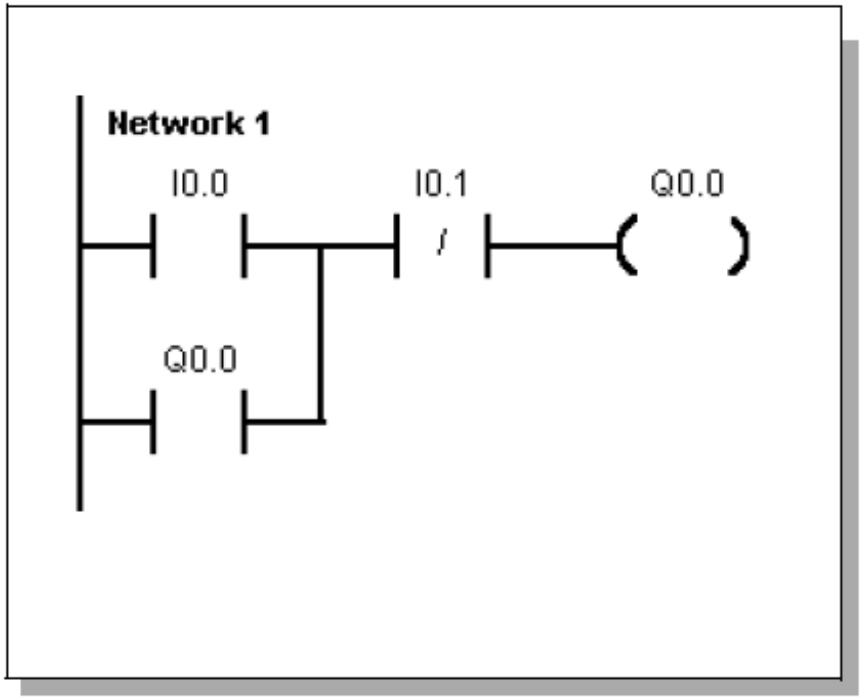

Network 1
I0.0
Q0.0


State of input
I0.0
Voltage at input changes
from 0 to 24 V
Time until process image
(PII) has status "1"
①
Process-
image of I0.0
State of output
Q0.0
②
Time for ladder logic
operations and modification of
the output status

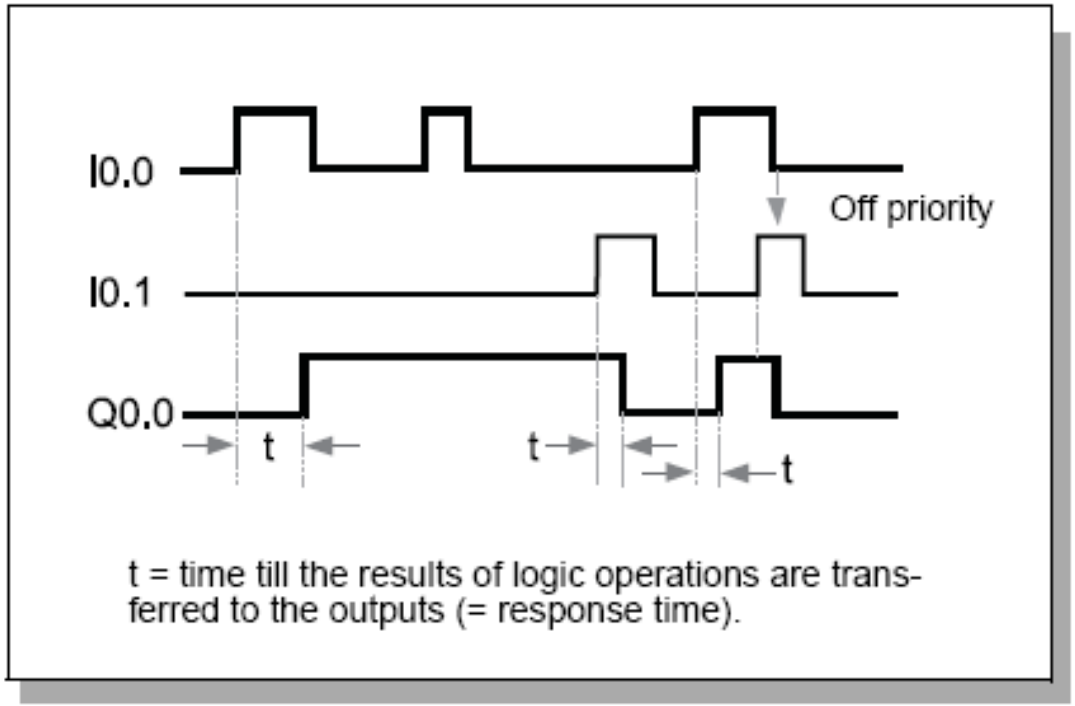
Samodržno stikalo



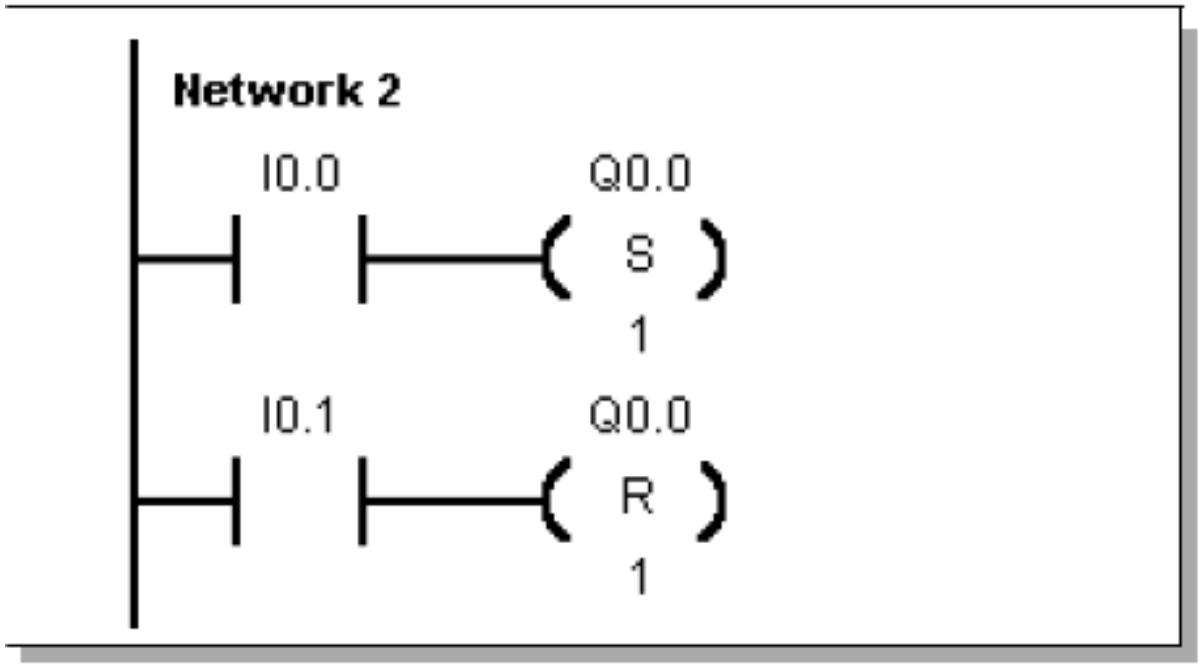
Samodržno stikalo



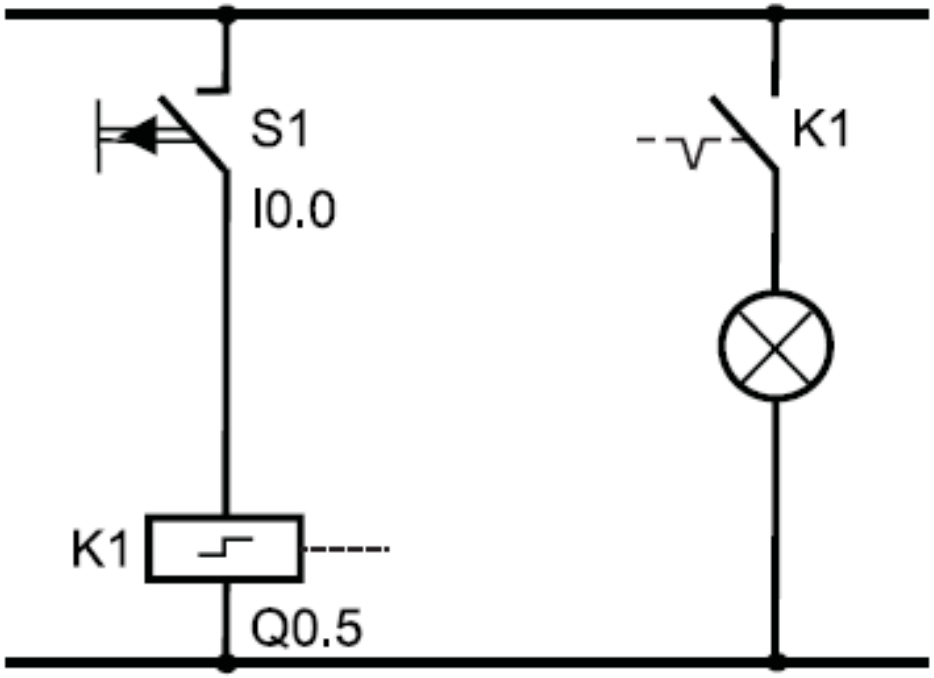
Samodržno stikalo



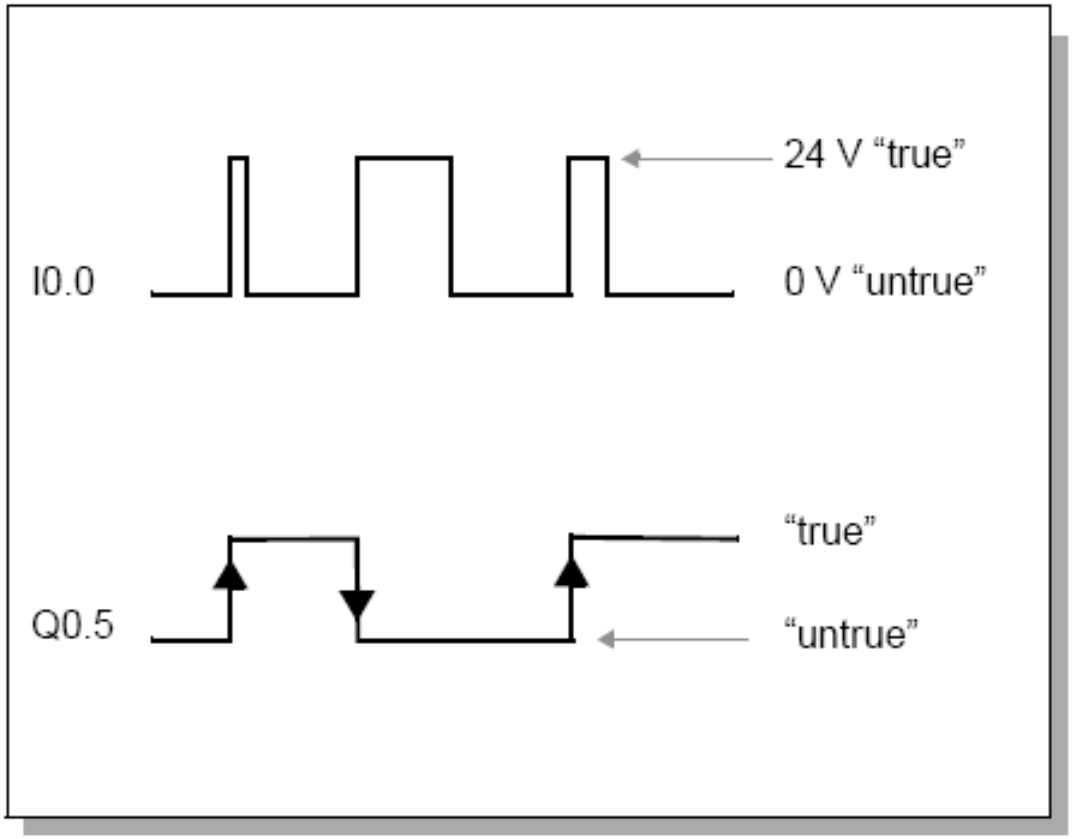
Samodržno stikalo



Pulzno stikalo



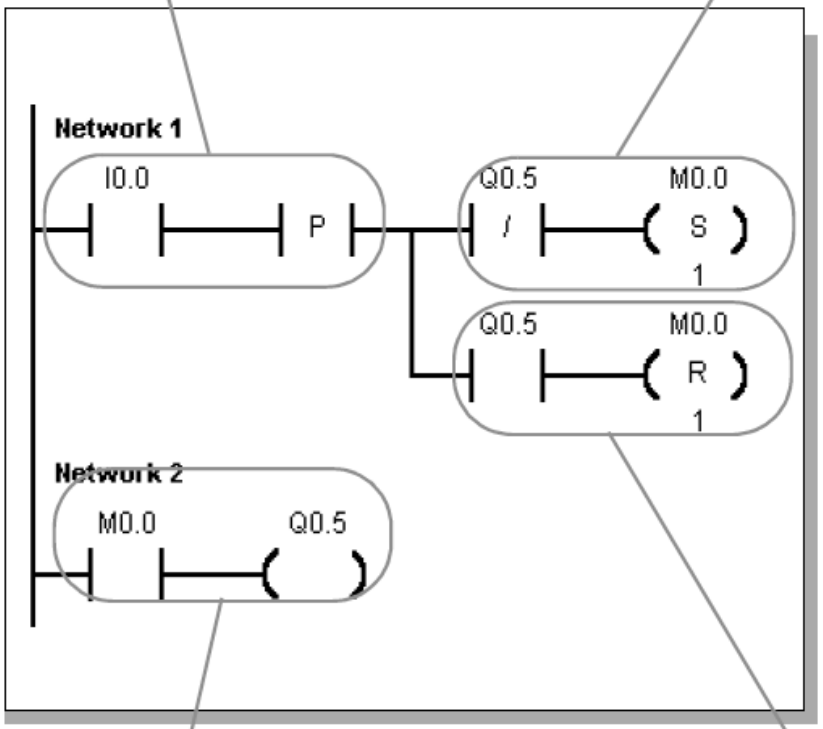
Pulzno stikalo

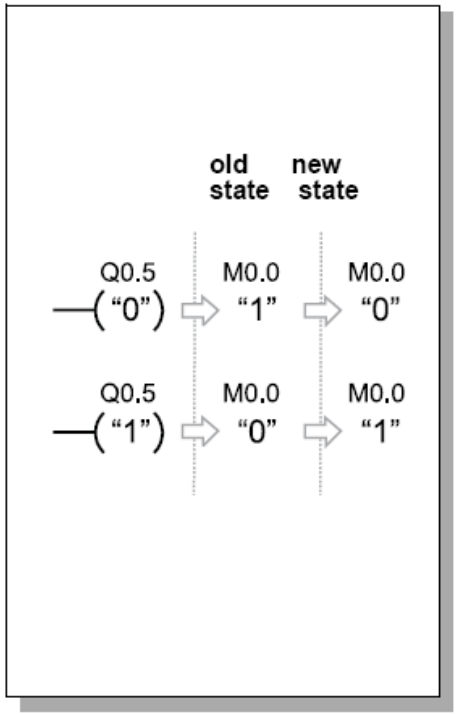


Pulzno stikalo

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".



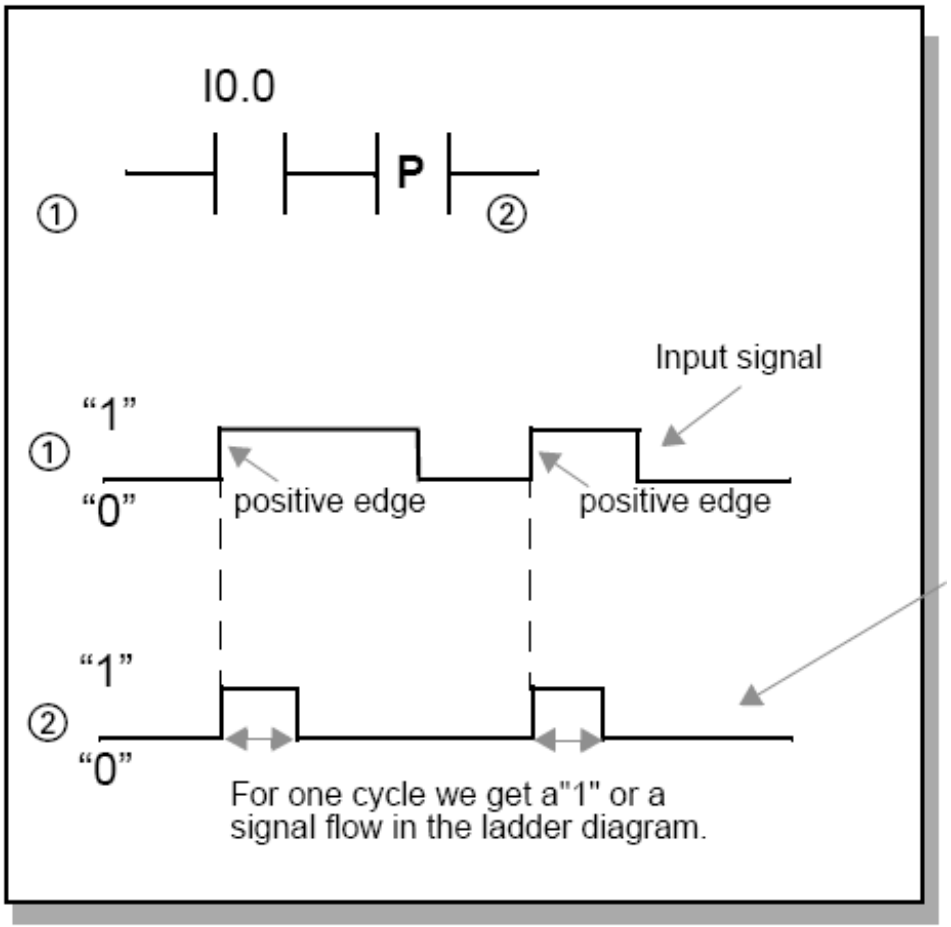


"Reversing" the state

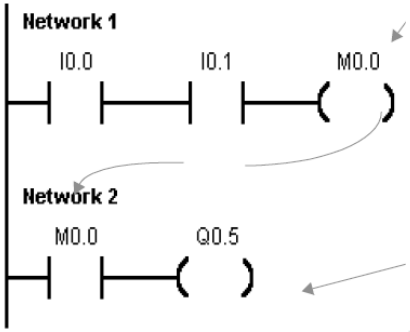
Assign the state of M0.0 to output Q0.5.

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

Pulzno stikalo



Pulzno stikalo



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

In PLC technology, bit memories are used as outputs and have an effect comparable with auxiliary contactors. A bit memory can be used as often as required at any location as an NC contact or an NO contact.

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

If the operating power is interrupted, bit memory contents are lost.

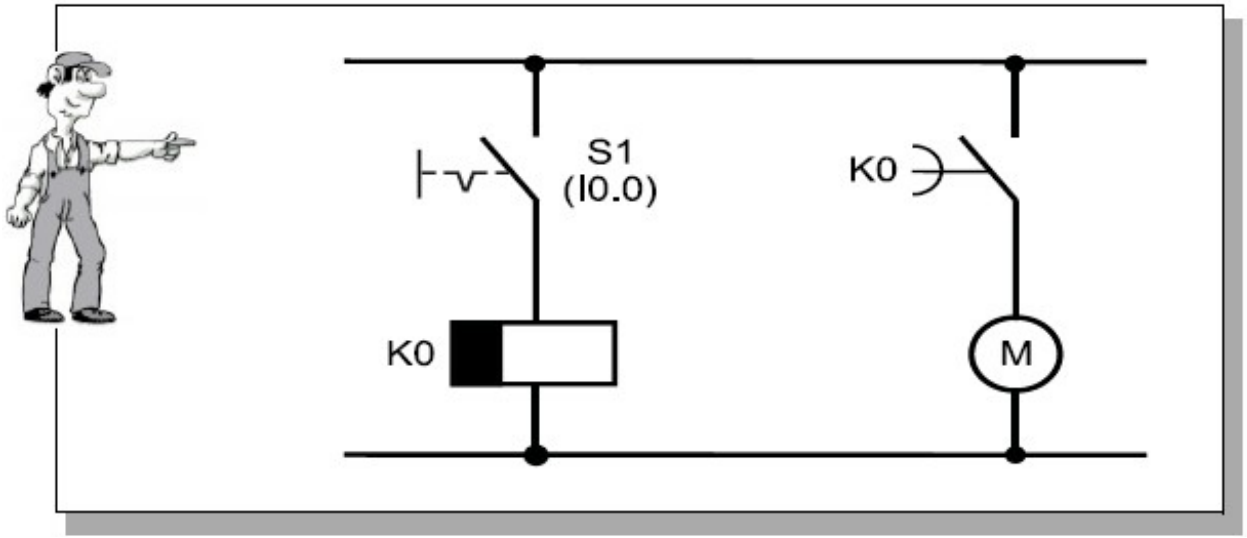
"Retentivity" is designed to prevent this.

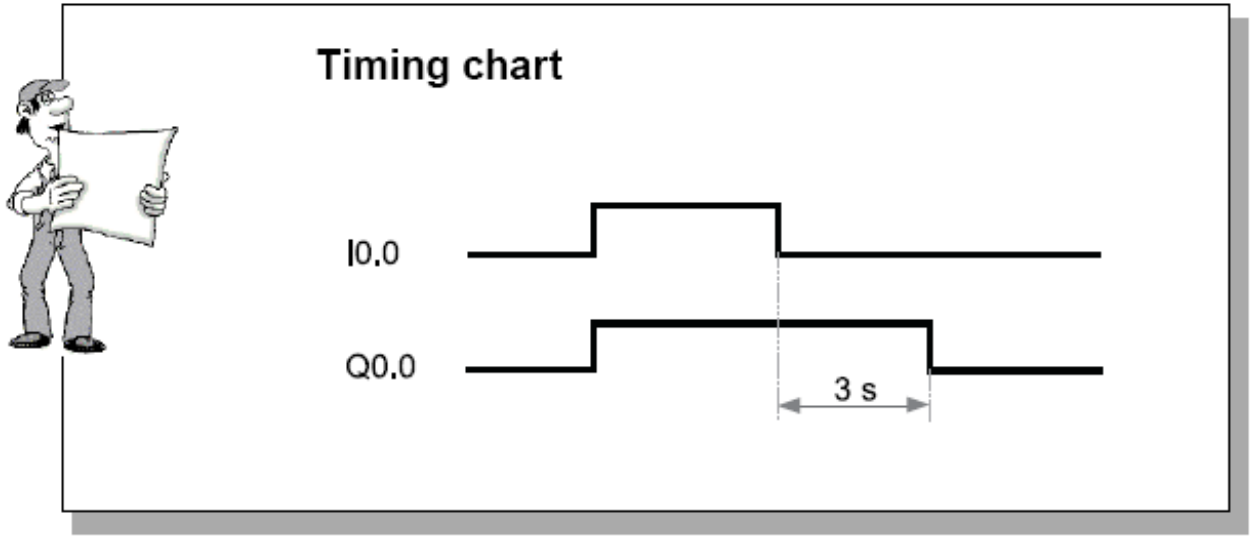
Bit memories are used if the (interim) result of a network is to be further processed in other networks (like sub-totals when adding numbers manually). They are also used to store evaluated follow-on states temporarily.

Off-delay stikalo (zakasnilno)

Po vklopu S1 se prične ventilator vrteti.

Po izklopu S1 se ventilator še vrti določen čas 3 s, nato se ugasne.

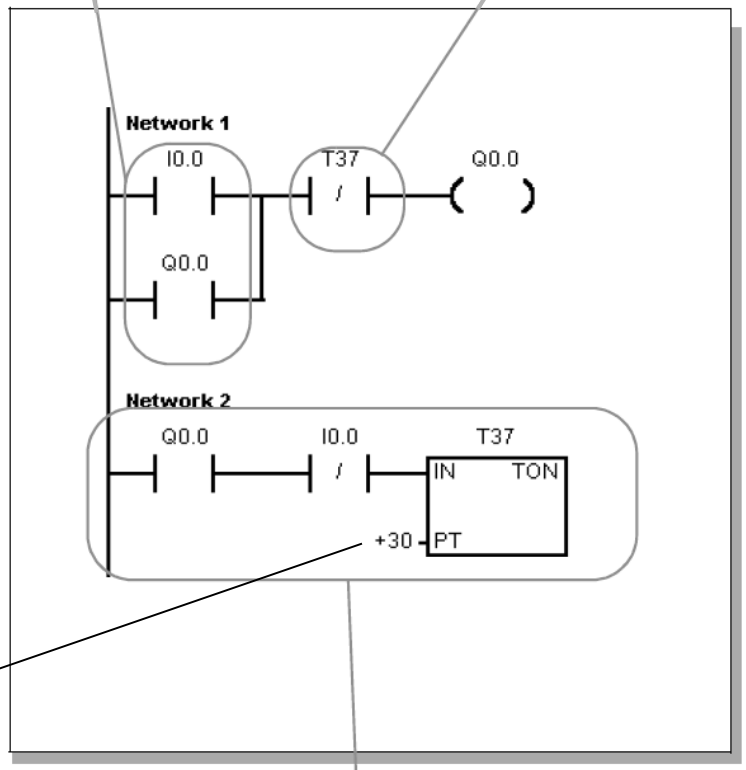




Off-delay stikalo (zakasnilno)

I0.0 aktivira Q0.0.
Q0.0 drži aktivno stanje (vzporedna vezava)

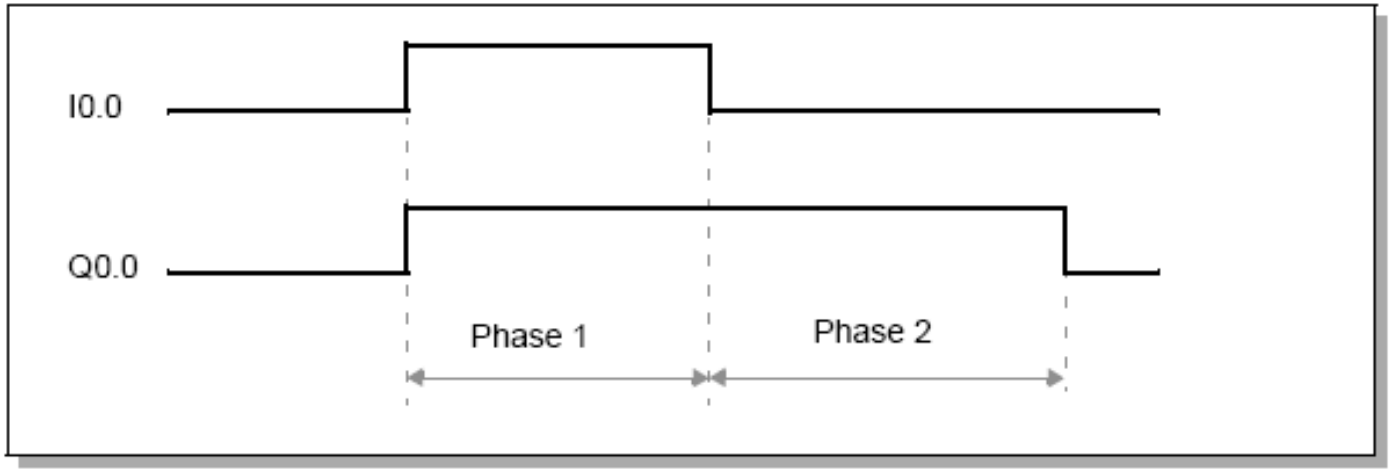
Po izteku časovnika T37 se ta kontakt prekine in motor se ostavi (Q0.0 postane neaktiven)



Časovna baza
30*100ms=3s

Ko je Q0.0 aktiven in I0.0 ponovno neaktiven (S1 izklopljeno), se štarta časovnik T37

Off-delay stikalo (zakasnilno)



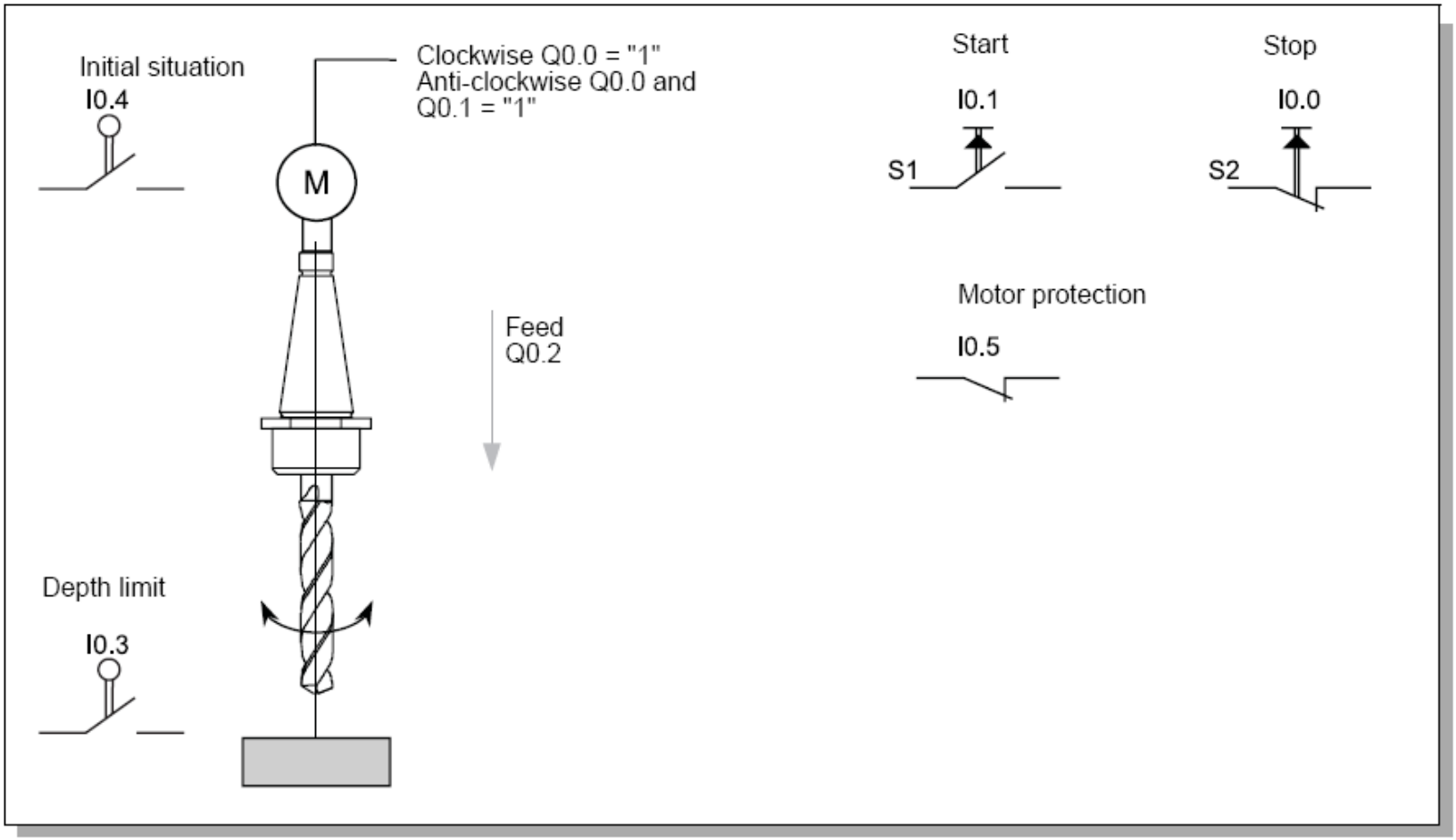
Sekvencer

Motor svedra se prične vrteti z aktivacijo S1. Po preteku 3 sekund se prične premikati.

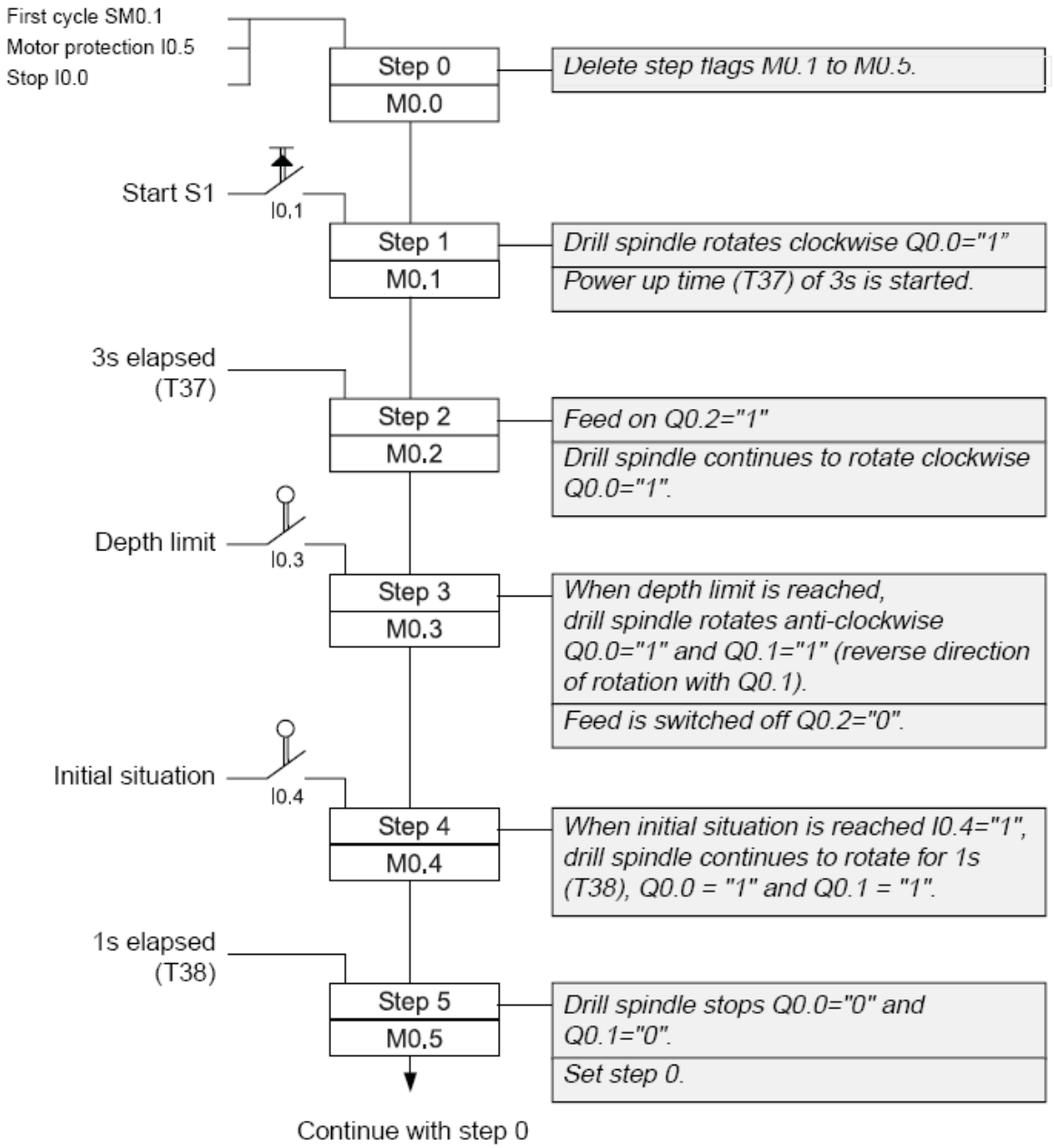
Ko doseže ustrezno globino (senzor I0.3), se aktuator premika izklopi. Sveder se prične vrteti v nasprotni smeri, s čimer se vrača v izhodiščni položaj. Smer vrtenja upravljamo s Q0.1.

KO sveder doseže izhodiščni položaj (senzor I0.4), se sveder še vrti 1 sekundo.

Sveder lahko kadarkoli ustavimo s tipko Stop (I0.0).



Sekvencer

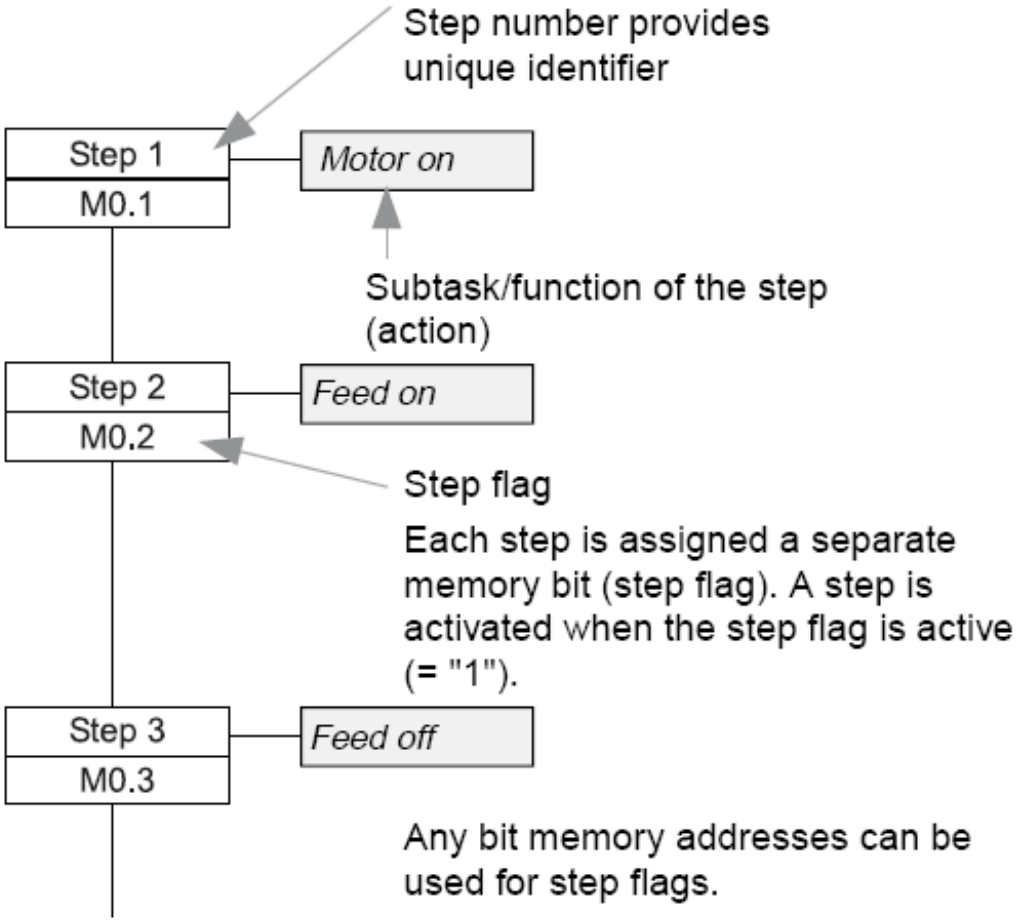

First cycle SM0.1
Motor protection I0.5
Stop I0.0
Step 0
M0.0
Delete step flags M0.1 to M0.5.
Start S1
I0.1
Step 1
M0.1
Drill spindle rotates clockwise Q0.0="1"
Power up time (T37) of 3s is started.
3s elapsed
(T37)
Step 2
M0.2
Feed on Q0.2="1"
Drill spindle continues to rotate clockwise Q0.0="1".
Depth limit
I0.3
Step 3
M0.3
When depth limit is reached, drill spindle rotates anti-clockwise Q0.0="1" and Q0.1="1" (reverse direction of rotation with Q0.1).
Feed is switched off Q0.2="0".
Initial situation
I0.4
Step 4
M0.4
When initial situation is reached I0.4="1", drill spindle continues to rotate for 1s (T38), Q0.0 = "1" and Q0.1 = "1".
1s elapsed
(T38)
Step 5
M0.5
Drill spindle stops Q0.0="0" and Q0.1="0".
Set step 0.
Continue with step 0

Sekvencer

What is a sequencer control?

- A control method in which a task is broken down into very small, usually sequential, subtasks
  (e.g. Motor on, feed on, 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,
  e.g. M0.1 = "1".

Sekvencer

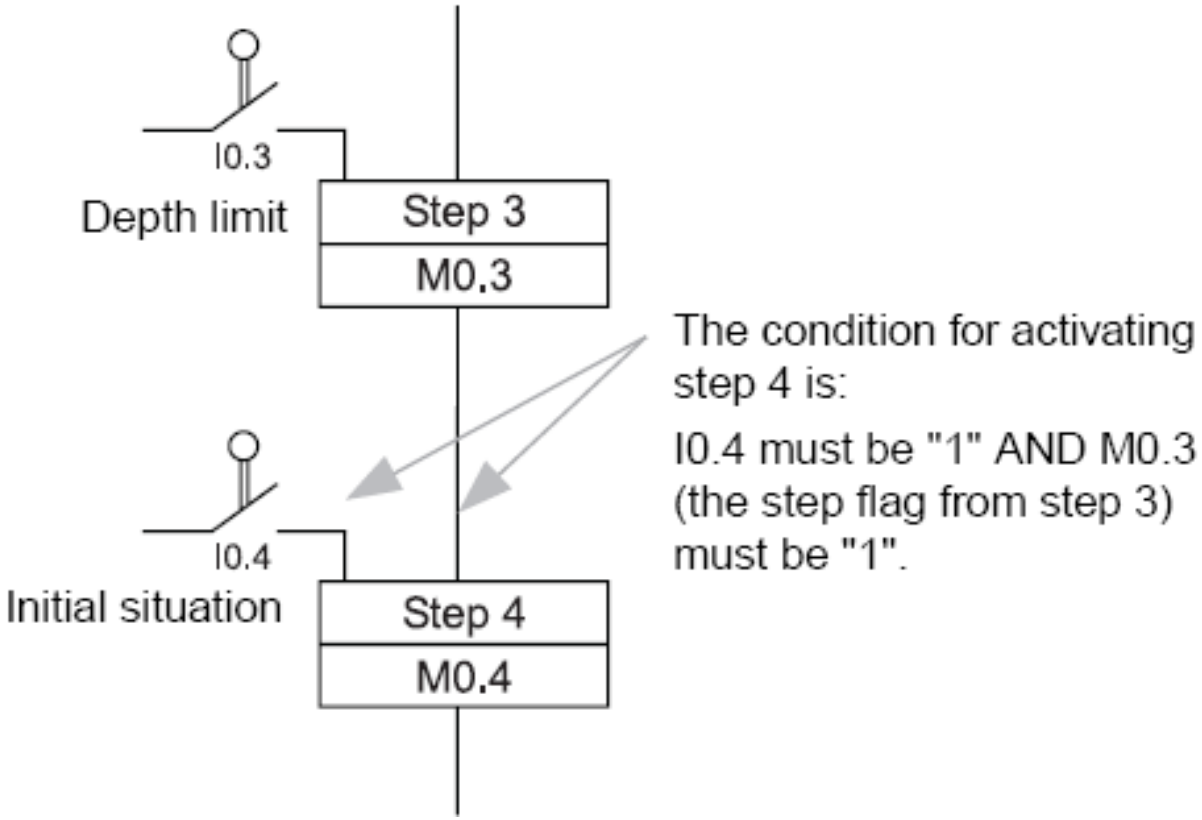

Step number provides unique identifier
Step 1
M0.1
Motor on
Subtask/function of the step (action)
Step 2
M0.2
Feed on
Step flag
Each step is assigned a separate memory bit (step flag). A step is activated when the step flag is active (= "1").
Step 3
M0.3
Feed off
Any bit memory addresses can be used for step flags.

Sekvencer

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.

Sekvencer


I0.3
Depth limit
Step 3
M0.3
I0.4
Initial situation
Step 4
M0.4
The condition for activating step 4 is:
I0.4 must be "1" AND M0.3 (the step flag from step 3) must be "1".

Sekvencer

The two program sections of a sequencer control:

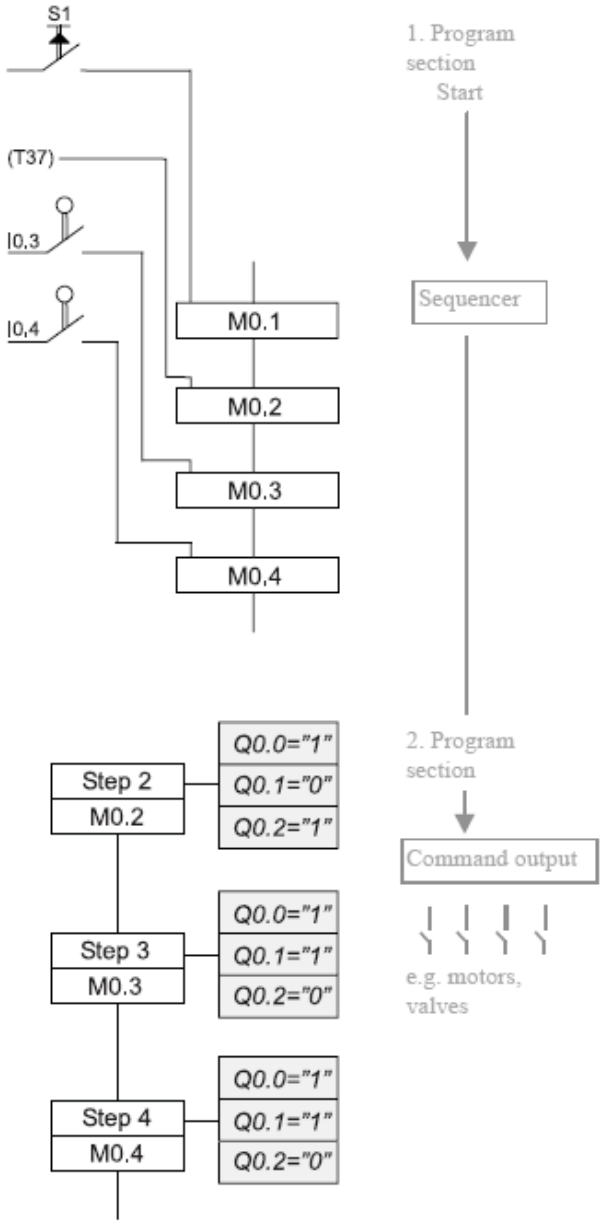
1) The conditions for activating the individual steps (subtasks) are logically combined with the individual step flags.

Start S1 I0.1, 3s delay, depth limit I0.3, initial situation I0.4, preceding step in each case.

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

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

This defines the overall sequence of the task.



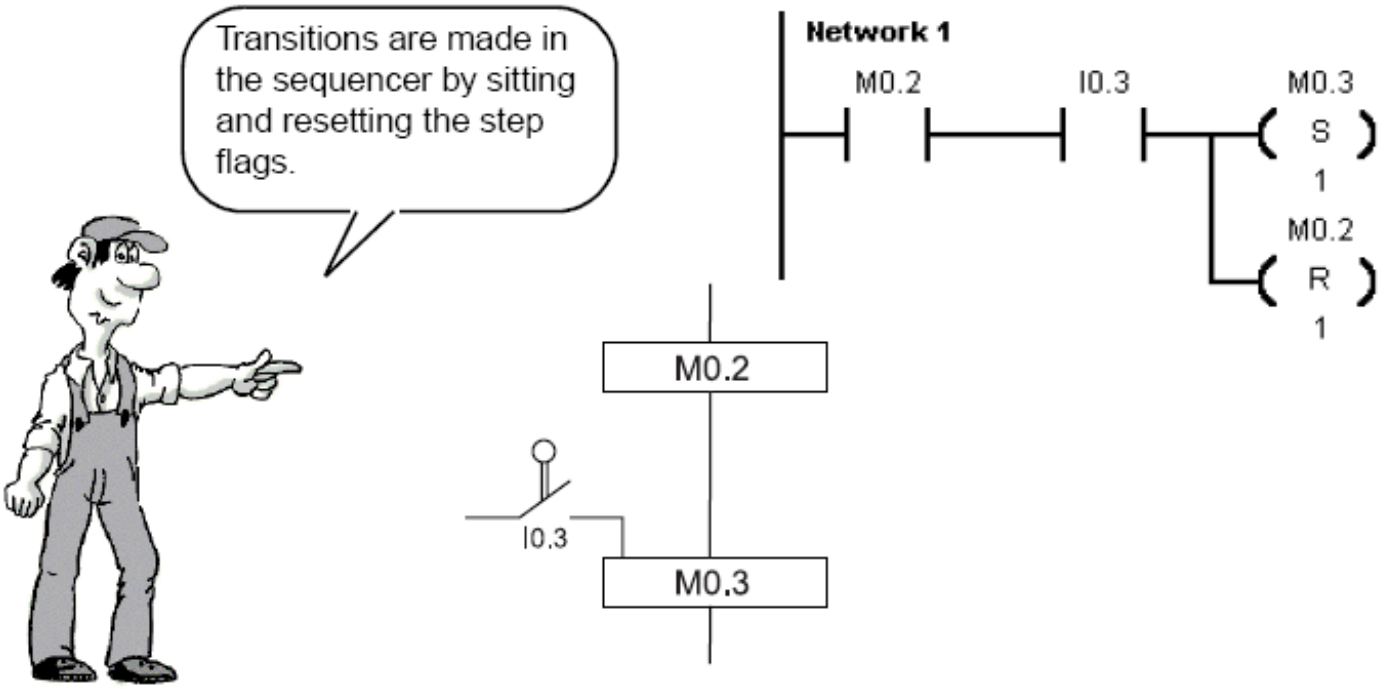
2) The active memory bits are assigned to the outputs of the PLC which then control contactors or valves, for example.

Q0.1, Q0.2, Q0.0

This is the interface to the plant /machine.

Sekvencer

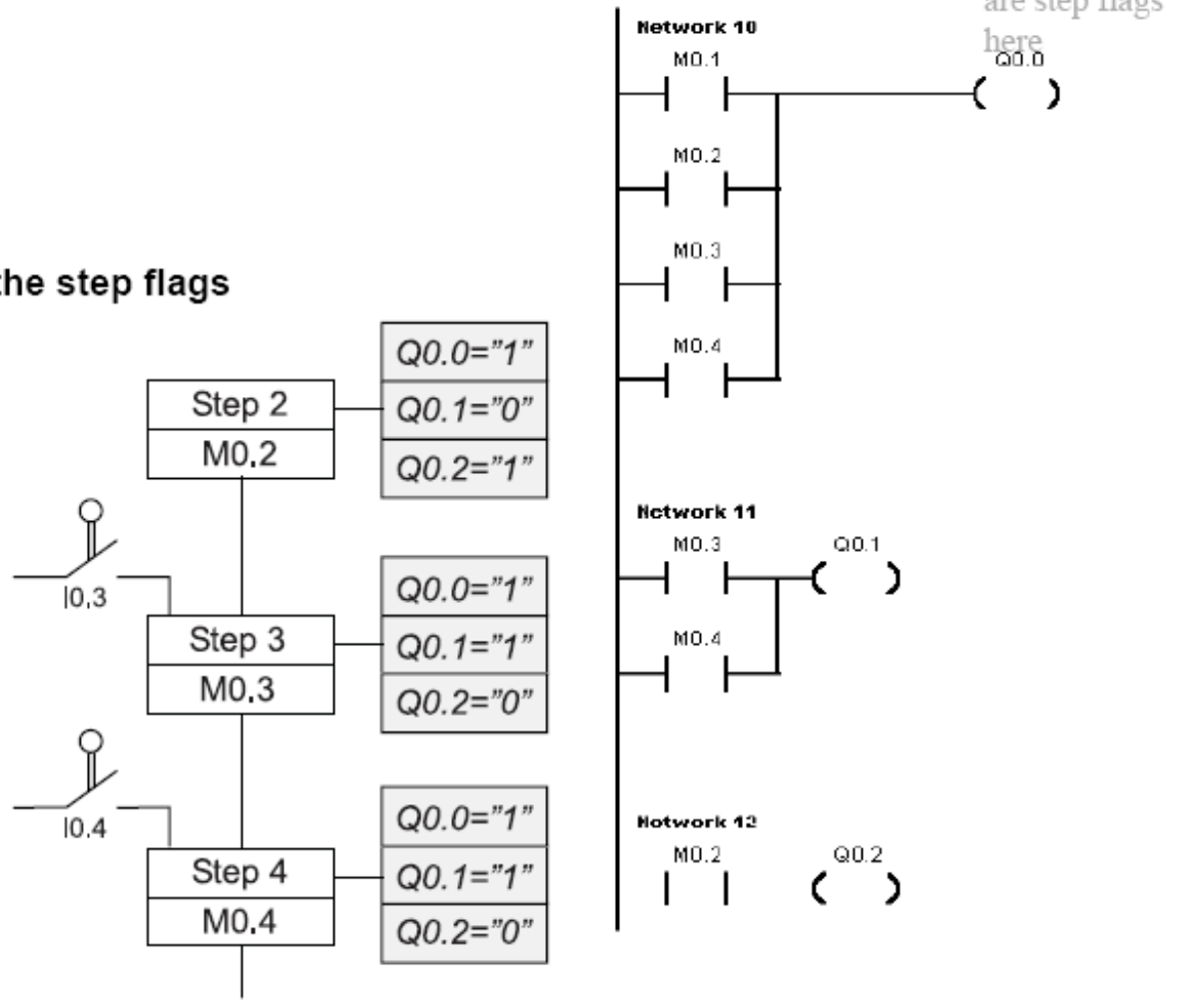
1) Controlling the sequencer/making transitions in the sequencer



Sekvencer

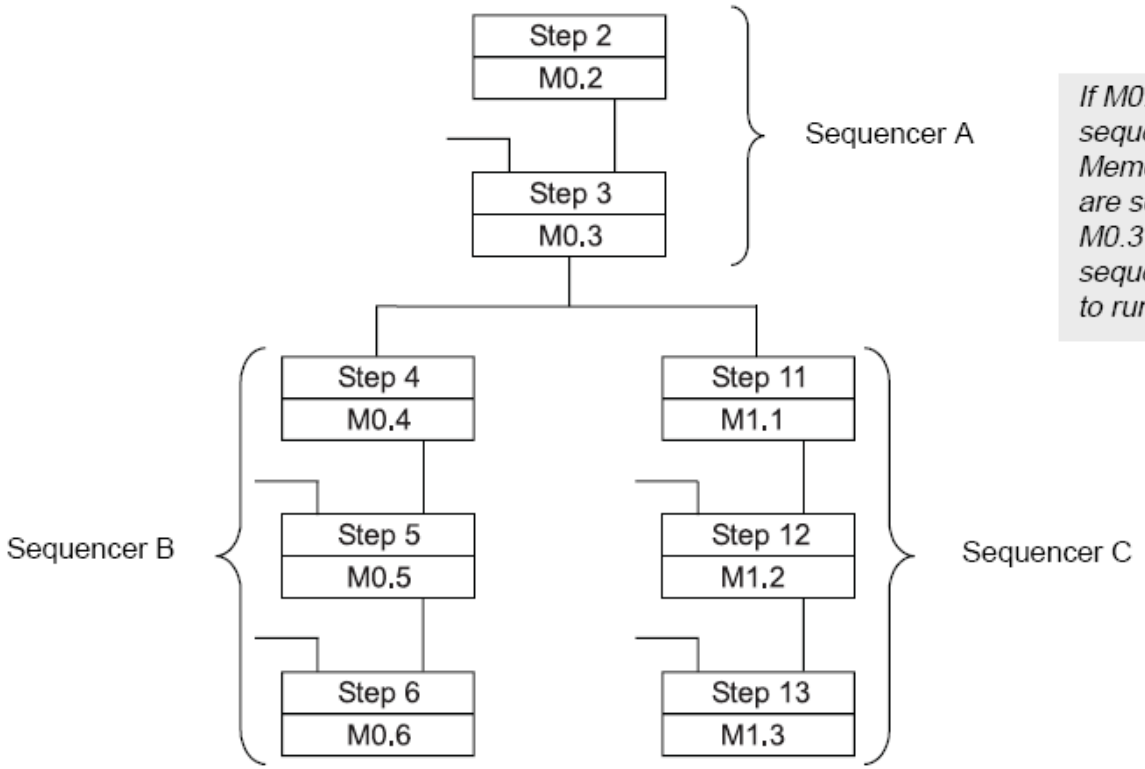
M0.2 and M0.3 are step flags here

2) Setting the outputs via the step flags



Sekvencer

- 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 a further sequencer.
- If two or more processes must be controlled simultaneously and independently, separate sequencers are used. This is shown in the diagram below.



If M0.3 ="1", the two sequencers B and C start. Memory bits M0.4 and M1.1 are set by M0.3. M0.3 is then reset and sequencers B and C continue to run.