fischertechnik

GROLLMUS

KNOWLEDGE | SOFTWARE | AUTOMATION

24V production line

Planning and implementing automatic processes

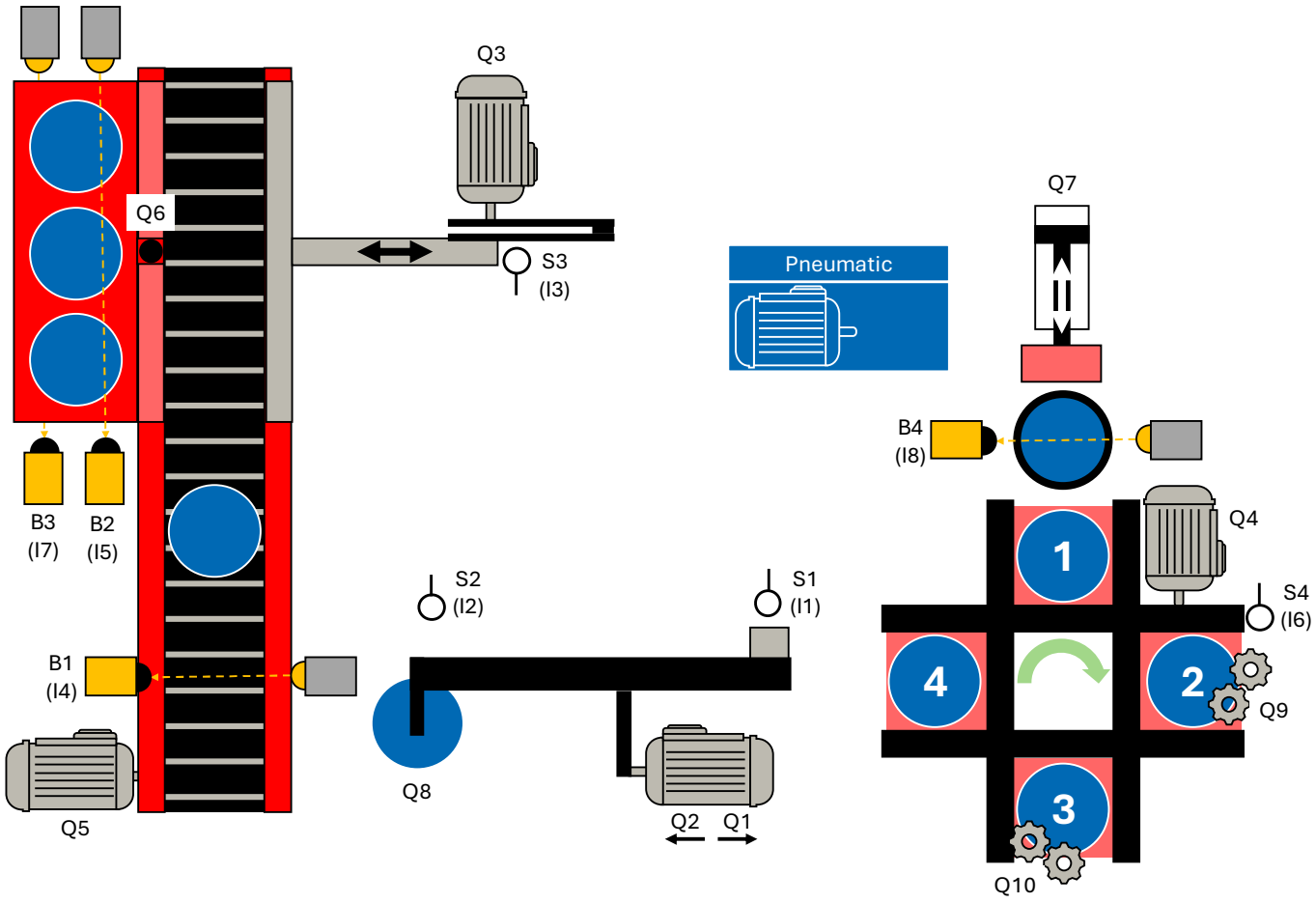


Table of contents

10 Planning and implementing automatic processes

10.1 Exercise: Planning the process chain with GRAFCET - transport route

Target:

I can use the functional description to create a GRAFCET sequence chain for the transport section of the production line.

Task:

Create a sequence chain in GRAFCET for the transport route of the production line so that the function is implemented based on the system description.

Function:

1. Initial step
   No actions are carried out in the initial step.
   The step chain remains in this step until an empty pallet is inserted. The pallet interrupts the light barrier B3. B2 must not be interrupted.

2. Move slider to home position
   After an empty pallet has been inserted, the slider (Q3) can be moved to the home position. The slider may only be activated if it is not in the home position and a pallet is in front of it.
   If the pusher is activated, the counter for the workpieces on the belt must also be reset.
   If the slider is in the home position (S3 actuated), you can switch to the next step.

3. Close divider
   Now that the slider is in the home position, the separator can be closed. The valve (Q6) must be permanently activated to keep the separator closed. 500 milliseconds after the valve has been activated, it can be assumed that the separator is closed. This means you can switch to the next step.

4. Waiting for workpiece
   In this step, the transfer unit remains in place until a new workpiece has been placed on the belt; no actions are performed. A new workpiece is detected by interrupting light barrier B1. Before switching to the next step, the transfer unit must have left the belt again (S1 not actuated).

5. Removing the workpiece
   If a workpiece is on the belt and the transfer unit has left the station, it can be transported away. The belt (Q5) may only be switched on if the slider is in the home position (S3 actuated), the separator is closed (Q6 actuated) and the transfer unit is not in the area of the belt (S1).
   The conveyor belt must run for at least 3 seconds before changing to the next step.

6. Increase counter
   After the workpiece has reached the end of the belt, the corresponding counter must be increased by one.

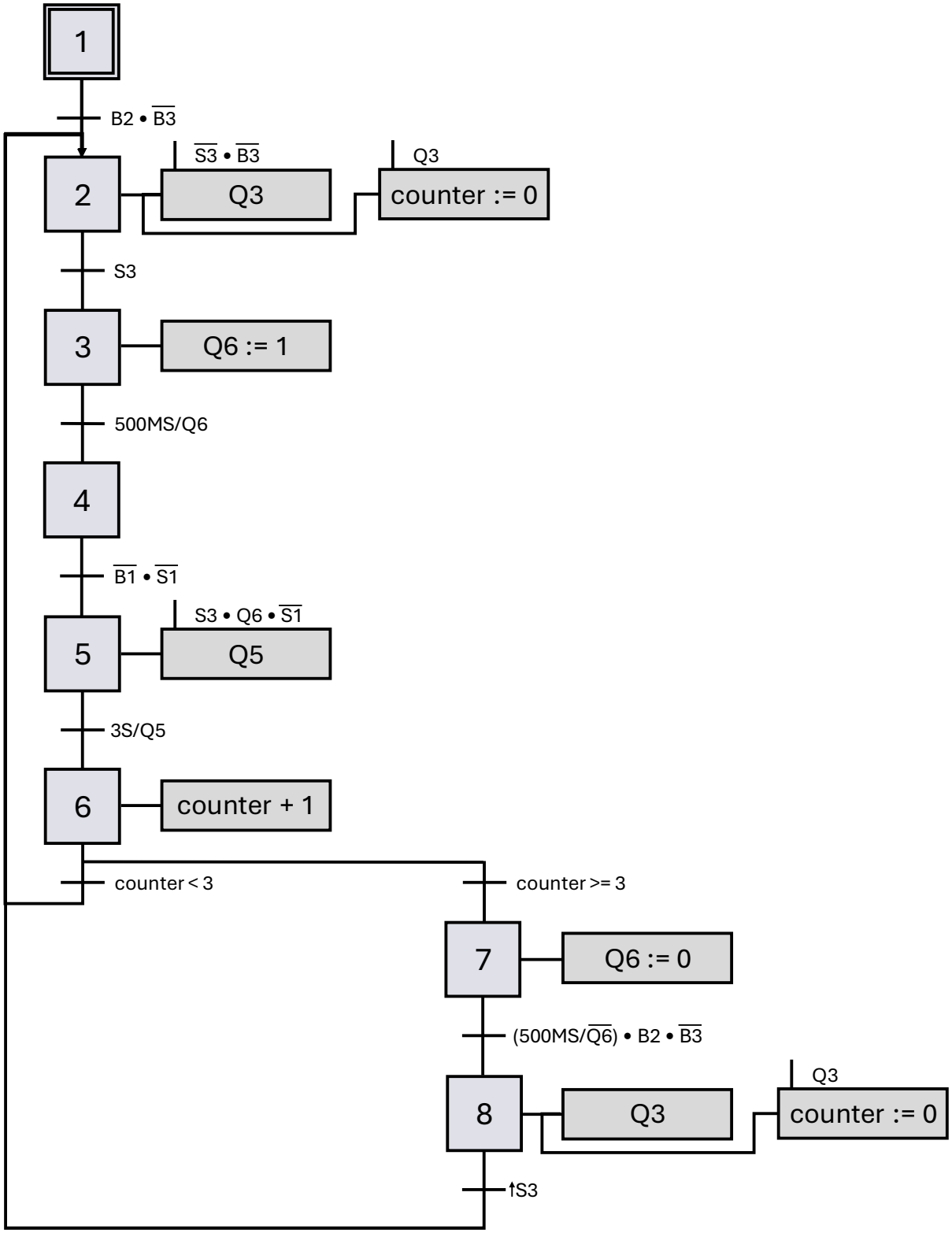
If there are already 3 workpieces on the belt, ejection can begin.
If the number has not yet been reached, you must return to step 2.

7. Open divider
   Once there are 3 workpieces on the conveyor belt, they can be pushed onto a pallet. To do this, first open the divider (Q6).
   500 milliseconds after the valve is no longer actuated, it can be assumed that the separator is fully open. To switch to the next step, an empty pallet must also be available.

8. Push workpiece onto pallet
   If a pallet is ready (B3 interrupted) and the divider is open, the workpieces can be pushed onto it.
   If the pusher is activated, the counter for the workpieces on the belt must also be reset.
   If the slider reaches its home position again (positive edge S3), you can jump back to step 2.

Solution

Solution:


1
B2 • B3
2
S3 • B3
Q3
Q3
counter := 0
S3
3
Q6 := 1
500MS/Q6
4
B1 • S1
5
S3 • Q6 • S1
Q5
3S/Q5
6
counter + 1
counter < 3
counter >= 3
7
Q6 := 0
(500MS/Q6) • B2 • B3
8
Q3
Q3
counter := 0
↑S3

10.2 Exercise: Planning the sequence chain with GRAFCET - converter

Target:

I can use the functional description to create a GRAFCET sequence chain for the production line converter.

Task:

Create a sequence chain in GRAFCET for the converter of the production line so that the function is given based on the system description.

Function:

1. Initial step
   In the initial step, the converter is moved to the home position. The home position is defined in such a way that neither of the two end positions is reached.
   If the transfer unit is in end position S1, it must be moved in the direction of the turntable (Q1) until it has left this position.
   If the transfer unit is in the end position S2, it must be moved in the direction of the conveyor belt (Q2) until it has left this position.
   Once the converter has reached the home position, you can switch to the next step.

2. Waiting for workpiece
   No actions are performed in this step. The system waits until the turntable provides a finished workpiece that can be picked up.
   If the turntable is in position (S4 actuated) and a workpiece is ready to be picked up (wstReady), the system switches to the next step.

3. Move the transfer unit towards the turntable
   The converter is moved in the direction of the turntable by activating Q1. The action is only executed as long as the end position (S2) has not been reached and the turntable is in position (S4).
   If the transfer unit is on the side of the turntable, you can switch to the next step.

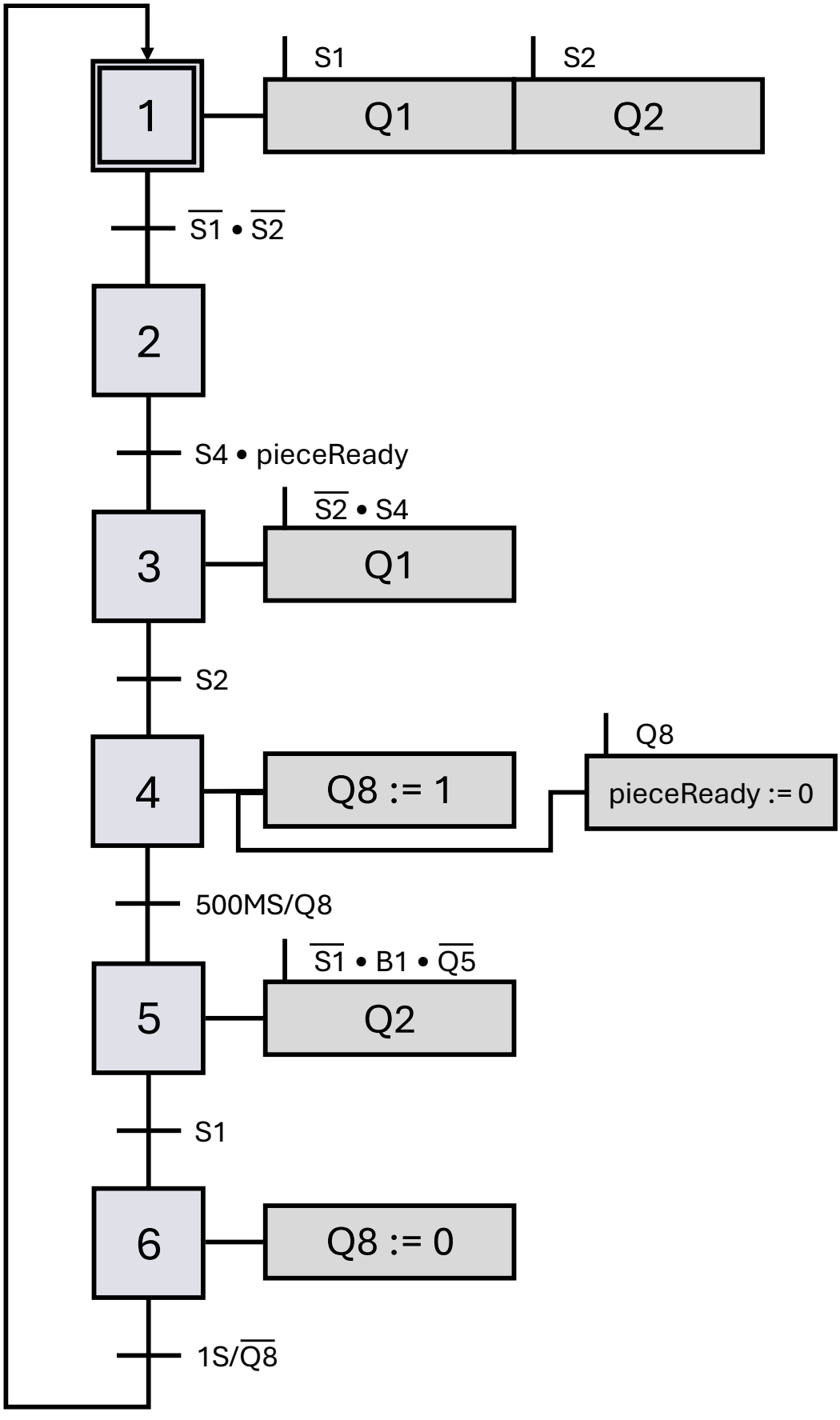
4. Switch on vacuum
   The vacuum is switched on by activating Q8. If the vacuum is switched on, the information that a workpiece is ready on the rotary table must be reset (wstReady := 0).
   500 milliseconds after switching on, it can be assumed that this has built up and can therefore be switched to the next step.

5. Move the converter towards the belt
   The transfer unit is moved in the direction of the turntable by activating Q2. The action is only executed as long as the end position (S1) has not been reached and the deposit position on the conveyor belt is free (B1) and the conveyor belt is not running (Q5).
   Once the converter has reached the end position, you can switch to the next step.

6. Switch off the vacuum
   The vacuum is switched off by resetting Q8. 1 second after switching off, it can be assumed that the vacuum has also dissipated and you can therefore jump back to the initial step.

Solution

Solution:


1
S1
S2
Q1
Q2
S1 • S2
2
S4 • pieceReady
3
S2 • S4
Q1
S2
4
Q8
Q8 := 1
pieceReady := 0
500MS/Q8
5
S1 • B1 • Q5
Q2
S1
6
Q8 := 0
1S/Q8

10.3 Exercise: Planning the process chain with GRAFCET - Magazine

Target:

I can create a GRAFCET sequence chain for the magazine on the rotary table of the production line using the functional description.

Task:

Create a sequence chain in GRAFCET for the magazine of the production line so that the function is given based on the system description.

Function:

1. Initial step
   No actions are carried out in the initial step.
   If the turntable is in position (S4 activated), workpieces are in the magazine (B4 interrupted) and the space (nest) on the turntable is empty ("nestBelegt" = "FALSE"), you can jump to the next step.

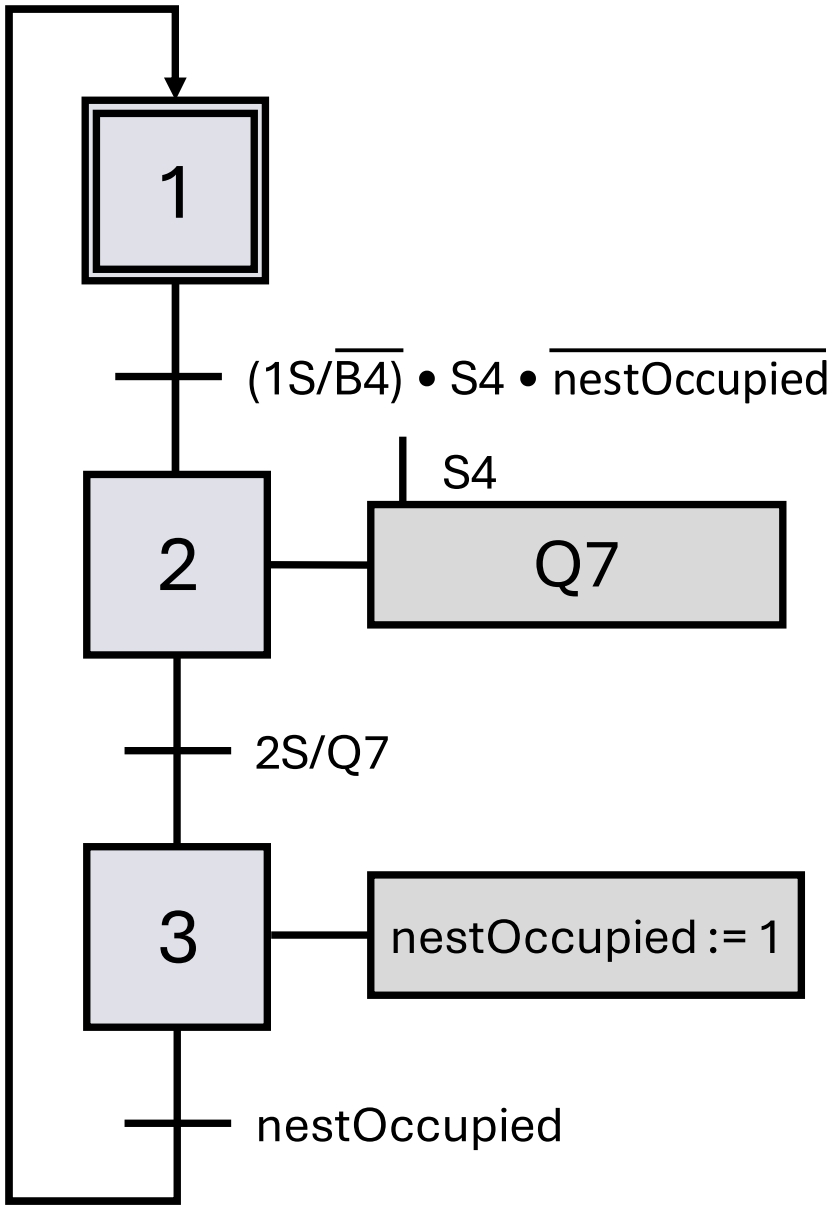
   It is advisable to delay the signal from light barrier B4 by approx. 1 second, as otherwise the pusher could extend as soon as the magazine is filled, which could lead to the workpieces becoming wedged.

2. Extend slider
   By activating Q7, the slider is extended and a workpiece is pushed out of the magazine onto the turntable. The slider may only be extended when the turntable is in position (S6).
   1 second after the valve is activated, it can be assumed that the slider is fully extended. You can jump to the next step.

3. Set nest occupied
   As there is now a workpiece on the turntable, the nest must be written as occupied. To do this, the "nest occupied" bit must be set to "TRUE".
   If the "nestOccupied" bit has the value "TRUE", you can jump to the initial step.

Solution

Solution:


1
(1S/B4) • S4 • nestOccupied
S4
2
Q7
2S/Q7
3
nestOccupied := 1
nestOccupied

10.4 Exercise: Planning the process chain with GRAFCET - Processing station

Target:

I can use the functional description to create a GRAFCET sequence chain for a processing station on the rotary table of the production line.

Task:

Create a sequence chain in GRAFCET for a processing station of the production line so that the function is given based on the system description.
Both the "Drilling" station and the "Welding" station have the same sequence. It is therefore only necessary to create a general sequence chain that is valid for both stations.

Function:

The process is started when a raw part is in the station and runs for a defined period of time. Instead of directly triggering the output for drilling (Q9) or welding (Q10), the term "machining" is generally used. No defined time is assumed as the transition, but "time" is generally assumed. After the work step, the "raw part" variable must be reset and the "finished part" variable set.

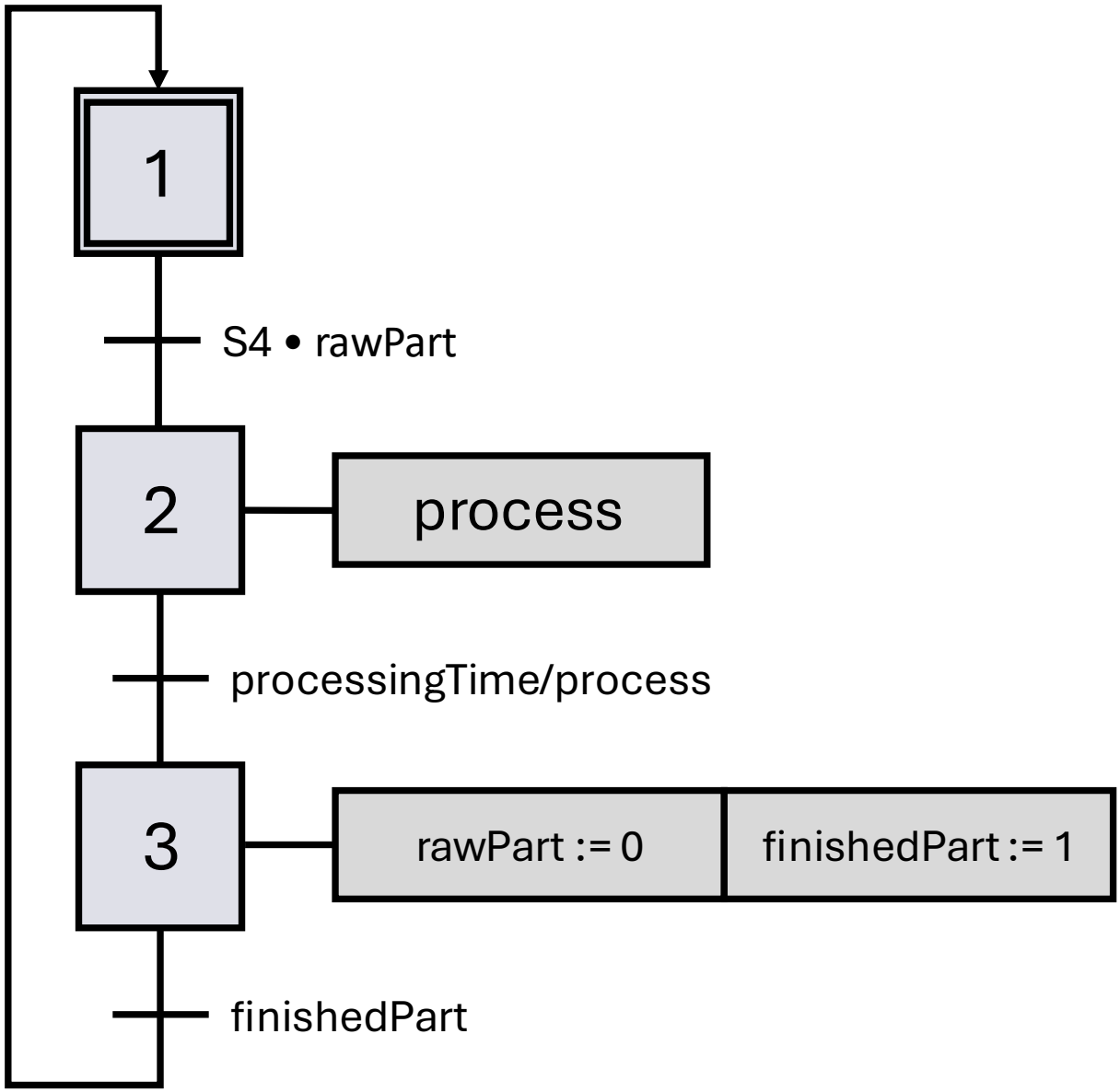
1. Initial step
   No actions are carried out in the initial step.
   If the turntable is in position (S4 activated) and a blank is in the station, you can jump to the next step.

2. Editing
   The workpiece is processed in this step. The "process" output must be activated.
   If the output is active for the defined "time", you can jump to the next step.

3. Set precast element
   The "Unmachined part" variable must be reset. The "finished part" variable must be set.
   If "Finished part" is set, you can jump back to the initial step.

Solution

Solution:


1
S4 • rawPart
2
process
processingTime/process
3
rawPart := 0
finishedPart := 1
finishedPart

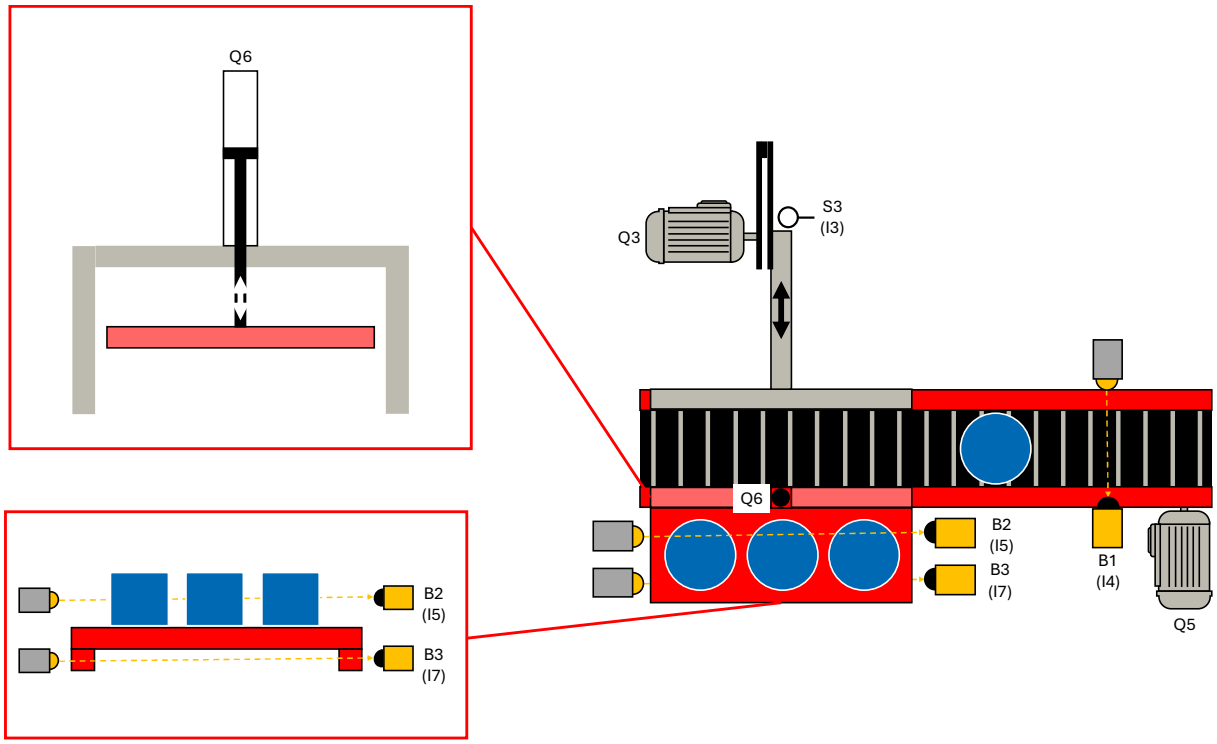
10.5 Exercise: Convert GRAFCET sequence chain into program code [FBD] - Transport route

Target:

I can create the automatic program for the transport section of the production line based on the functional description and the sequence chain created in GRAFCET.

Task:

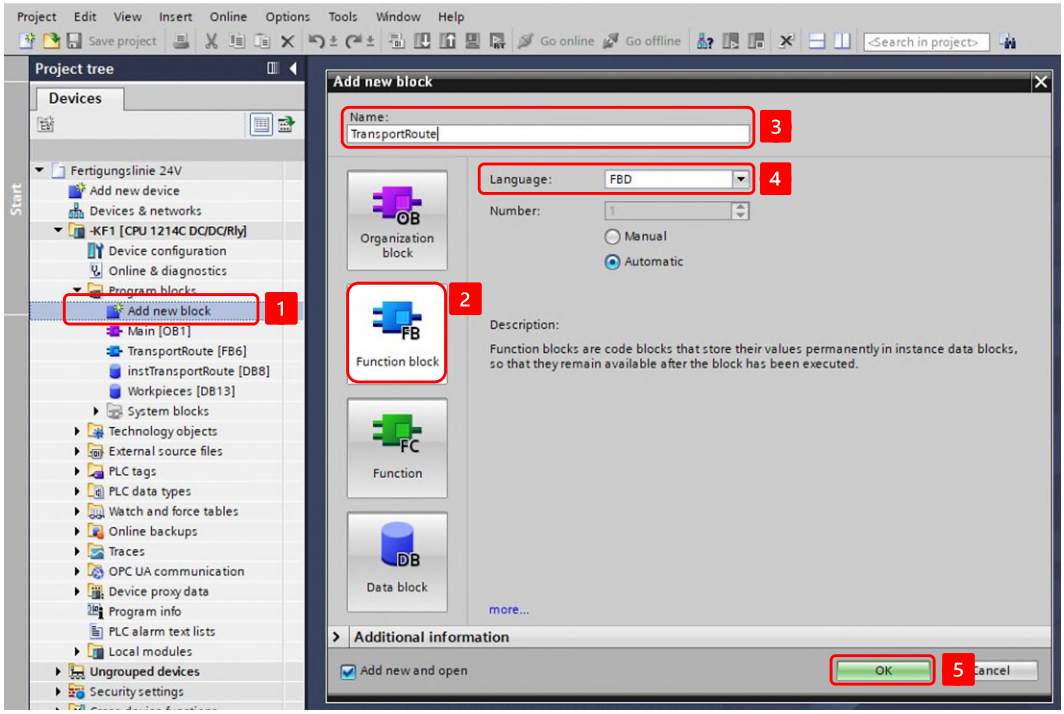
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



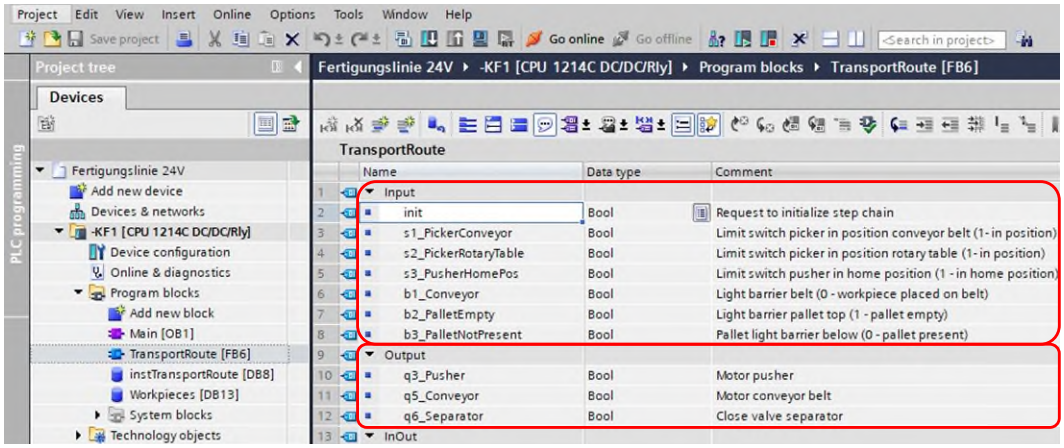
Picture 1 System diagram - transport route

Procedure:

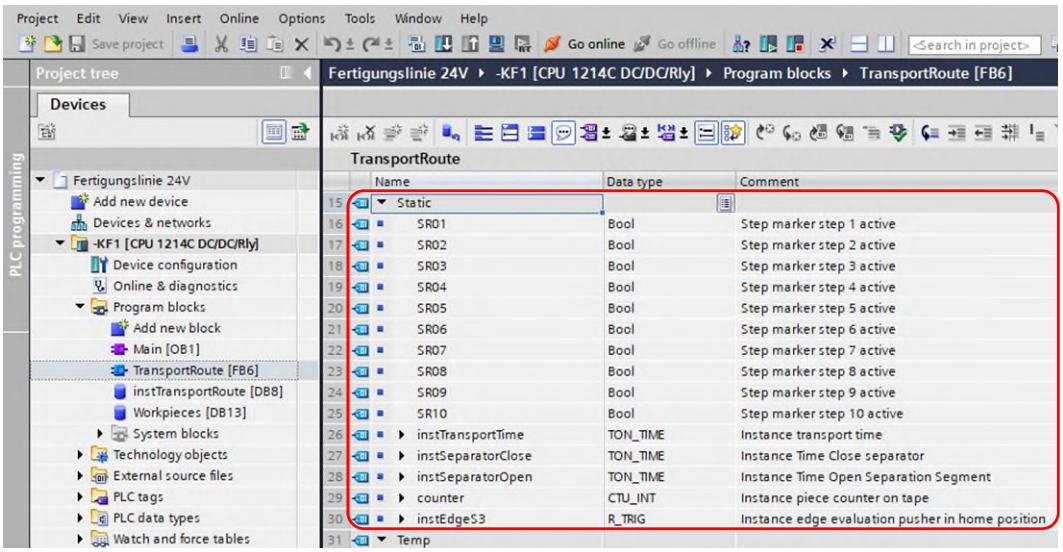
1. Create a new function block, select the desired programming language and assign a meaningful name:



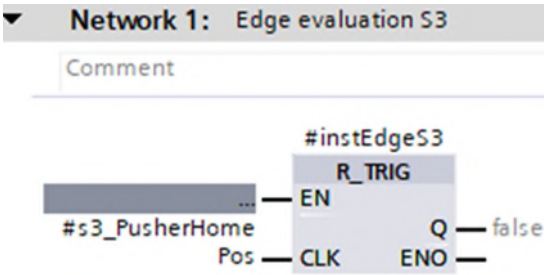
2. Declare variables for the sensors and actuators, as well as a variable for initializing the step chain in the function block interface:



3. Declare the step flags in the static area of the function block interface, an instance for the edge evaluation of the limit position switch S3, instances for the delay times and an instance for the workpiece counter:



4. Program the edge evaluation for the limit position switch S3 in the first network:



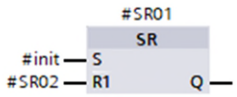
5. In the following networks, implement the individual steps using flip-flops based on the GRAFCET. A new network must be used for each step:

Step 1 is the initial step. In the automatic sequence, the system jumps back from step 6 to step 2 if the count has not yet been reached. If workpieces have to be ejected, the system jumps back to step 2 after step 8.

Step 6 is reset by the following step, or by step 2 in the case of a return.

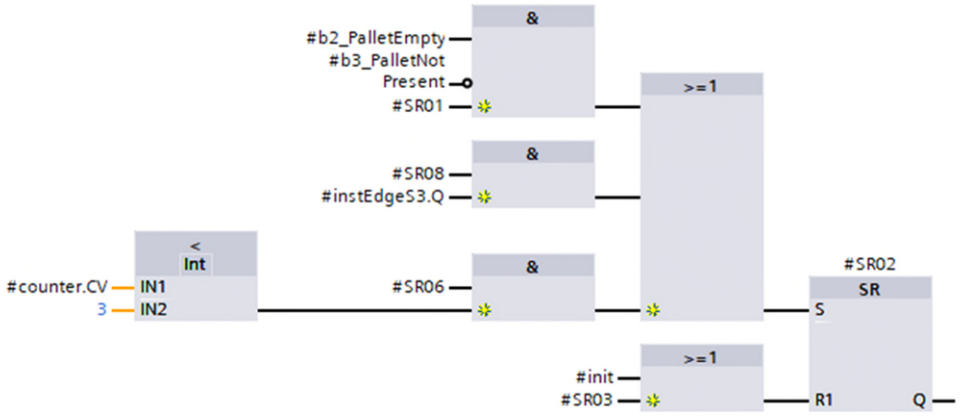
Network 2: Step 1 - Initial step

Comment



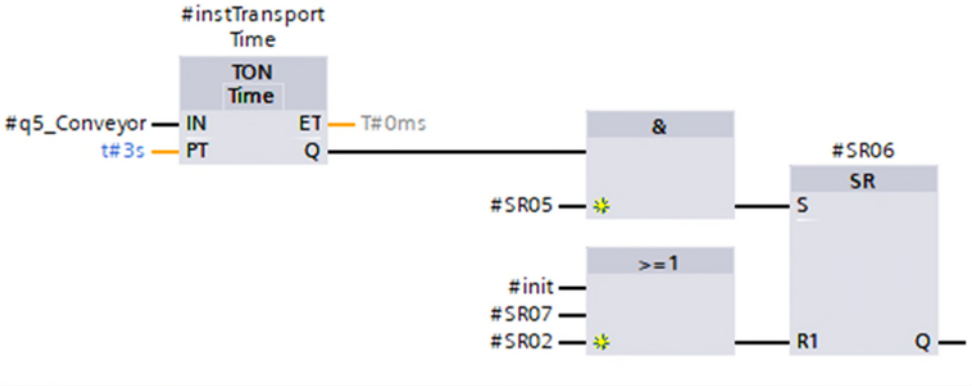
Network 3: Step 2 - Move pusher to home position

Comment



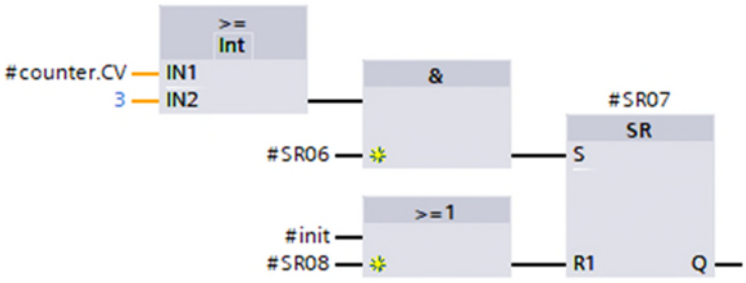
Network 7: Step 6 - Increase counter

Comment

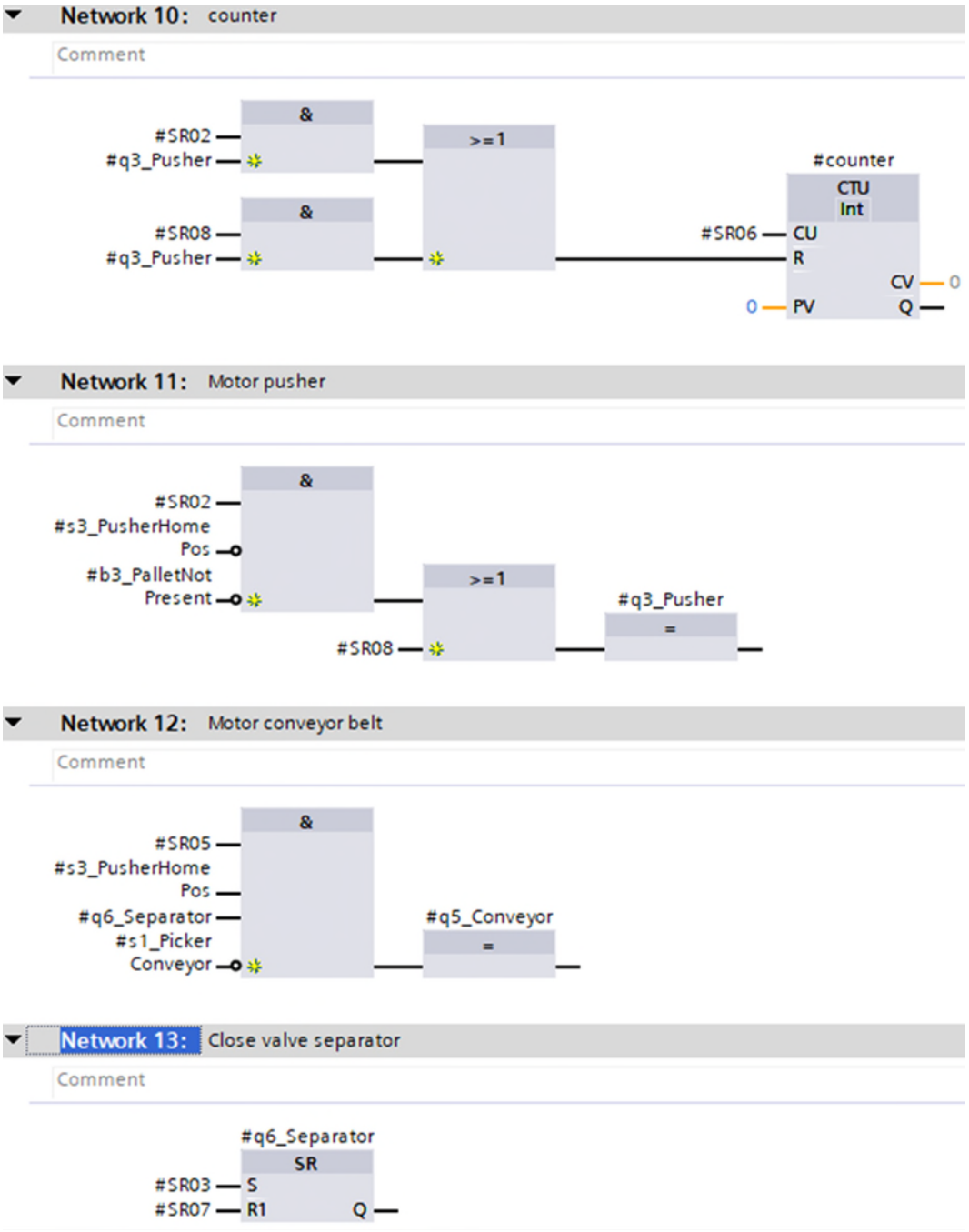


Network 8: Step 7 - Open the separator

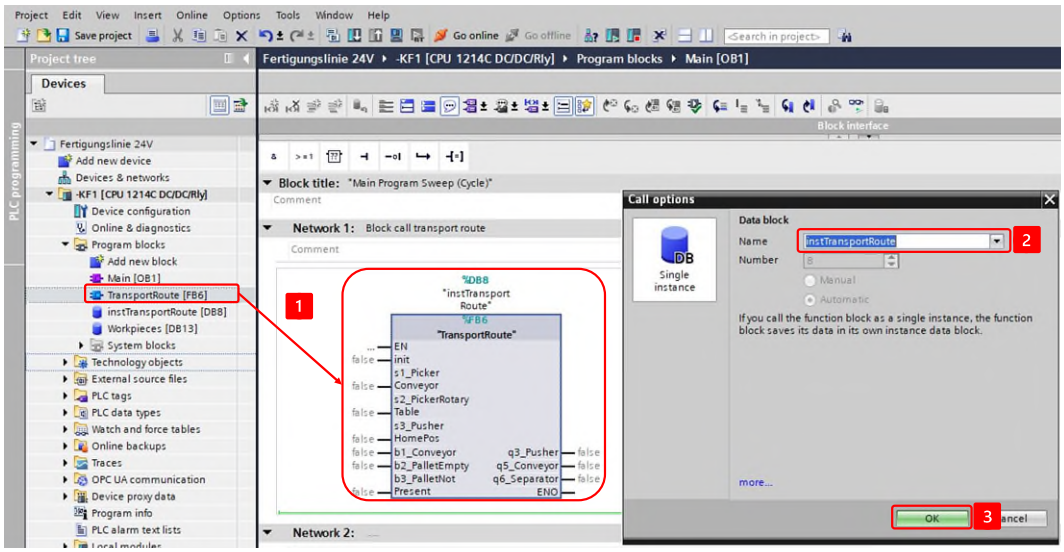
Comment



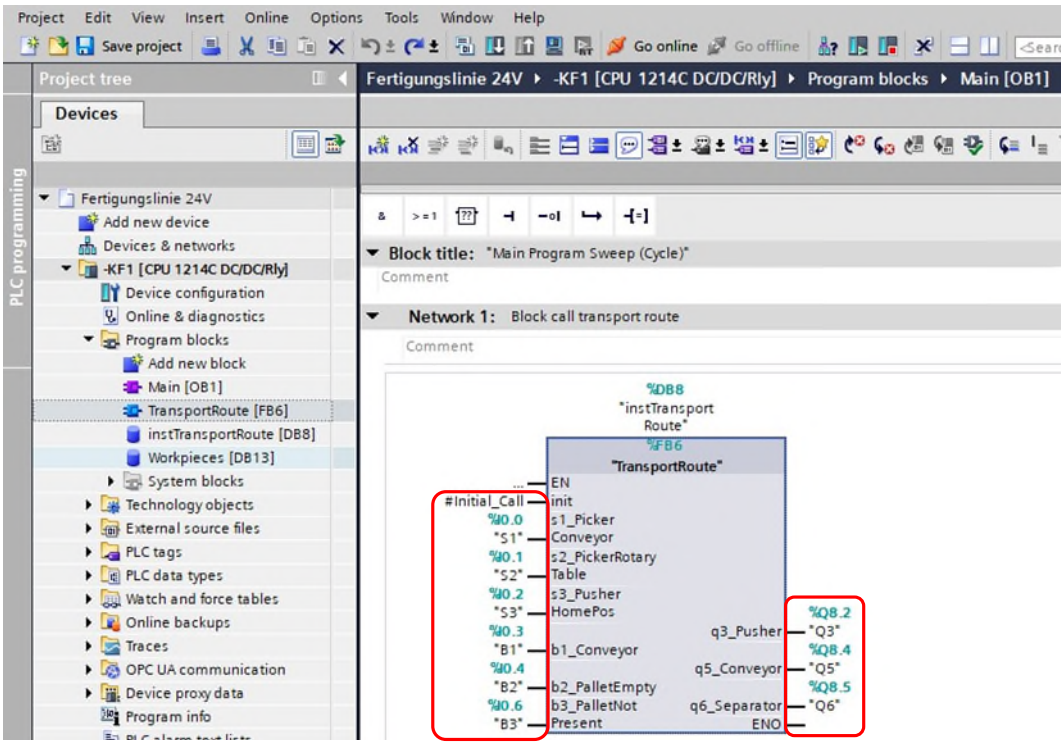
6. Assign the actions below the step chain in the next networks:



7. Call up the function module in "MAIN" and create an instance:



8. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

9. Commission the system in a structured manner with the aid of a commissioning protocol.

Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_02_Transportstrecke_FUP.zap17".

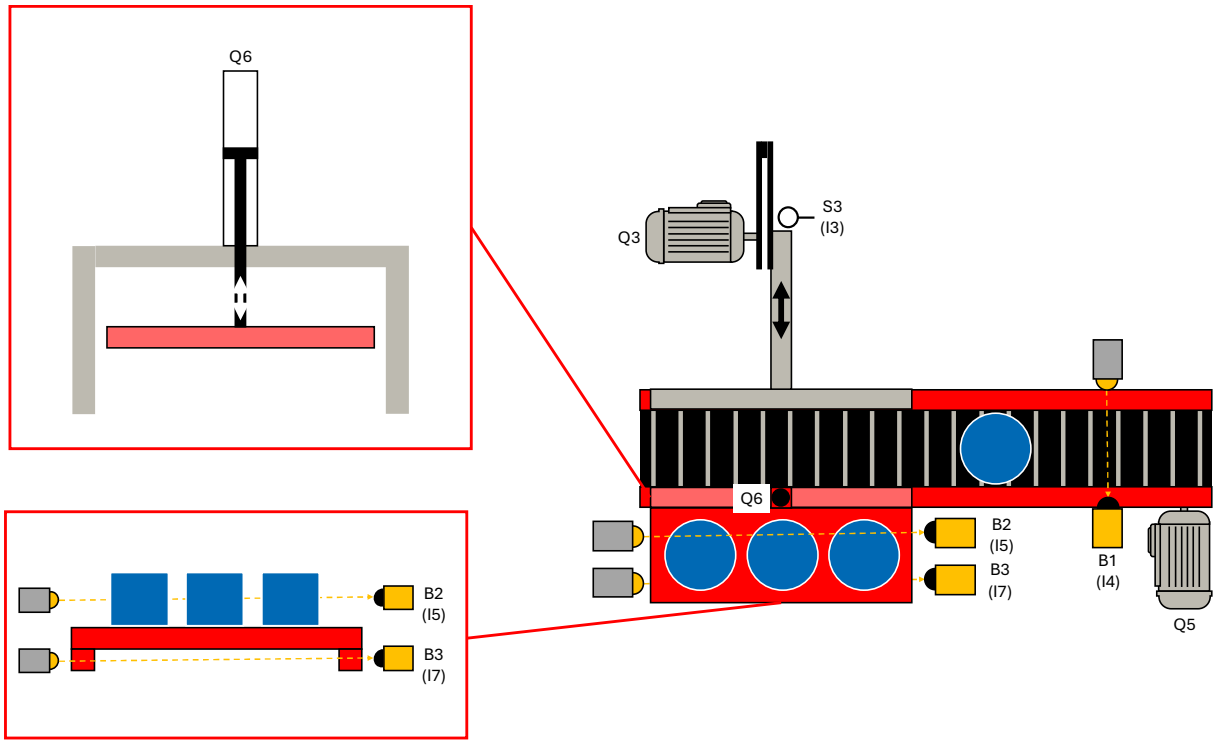
10.6 Exercise: Convert GRAFCET sequence chain into program code [ST / SCL] - Transport route

Target:

I can create the automatic program for the transport section of the production line using the functional description and the sequence chain created in GRAFCET.

Task:

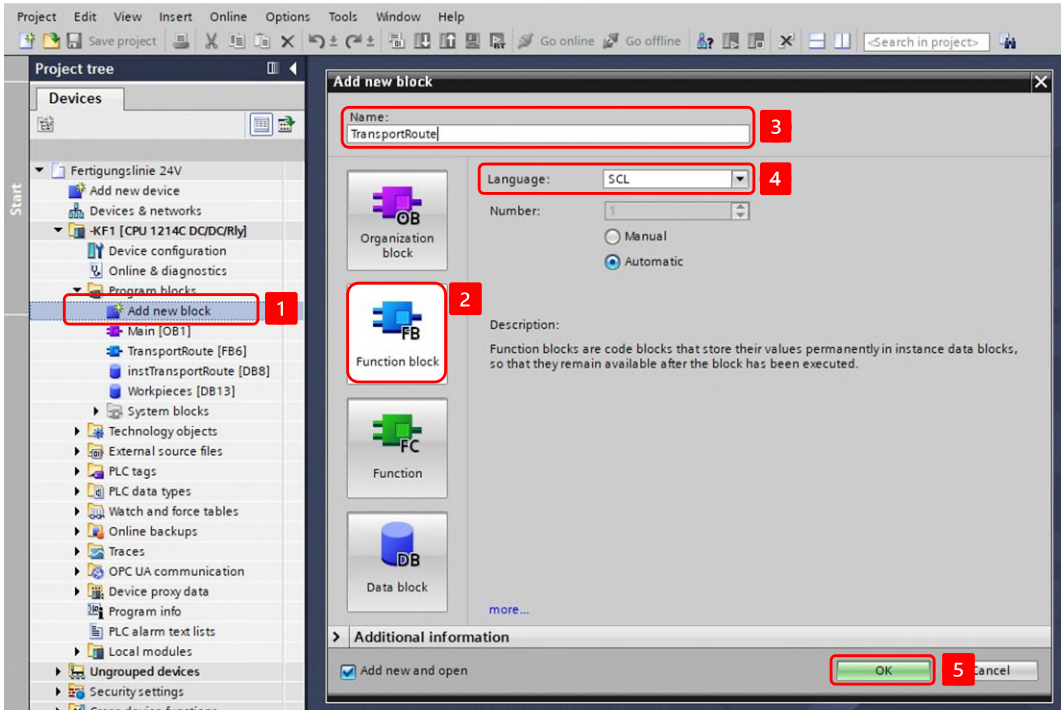
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



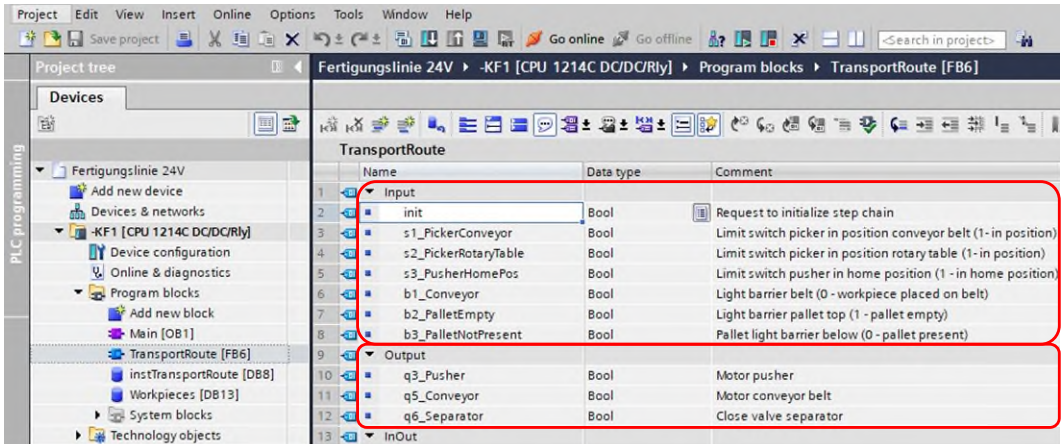
Picture 2 System diagram - transport route

Procedure:

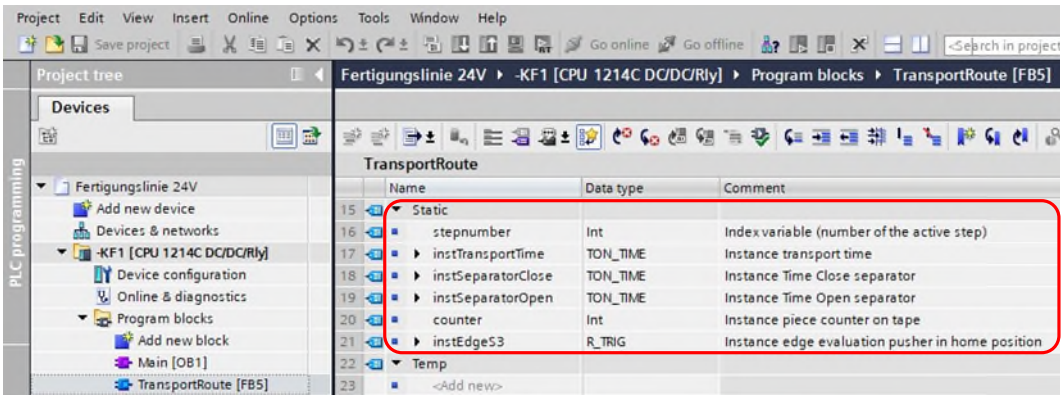
1. Create a new function block, select the desired programming language and assign a meaningful name:



2. Declare variables for the sensors and actuators, as well as a variable for initializing the step chain in the function block interface:



3. Declare the index variable in the static area of the function block interface, an instance for the edge evaluation of the limit position switch S3, instances for the delay times and a variable for the counter of the workpieces:



4. Program the initialization of the chain:

```
// initialize step chain
IF #init THEN
    #stepnumber := 1;   // initialize step chain
END_IF;
```

5. Program the edge evaluation for the limit position switch S3:

```
// edge evaluation pusher in home position
#instEdgeS3(CLK := #s3_PusherHomePos);
```

6. Implement the individual steps from the GRAFCET in the following CASE structure. A new CASE must be created in the structure for each step, which represents the step number:

Step 1 is the initial step. In the automatic sequence, the system jumps back from step 6 to step 2 if the count has not yet been reached. If workpieces have to be ejected, the system jumps back to step 2 after step 8.

All actions are reset in the initial step. This ensures that no actions remain set if an active step chain is aborted by an initialization request.

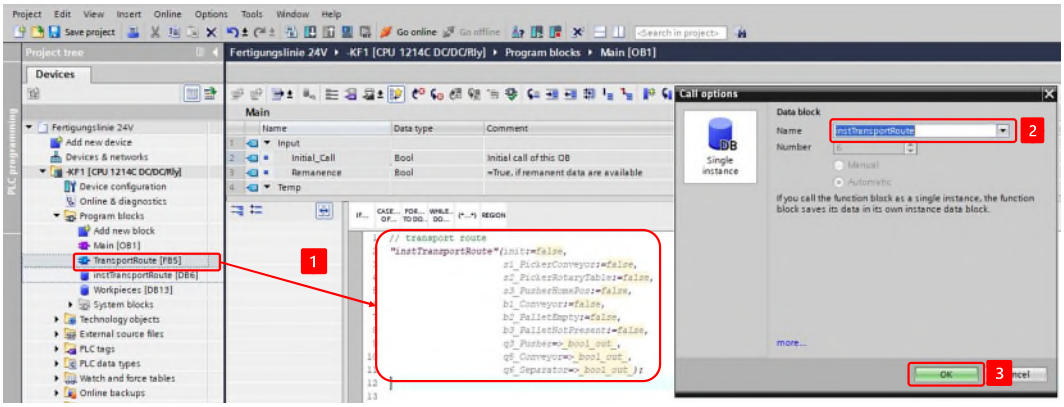
```
// step chain
CASE #stepnumber OF
    1: // step 1 - Initstep
        // Reset all actions
        #q3_Pusher := FALSE;
        #q5_Conveyor := FALSE;
        #q6_Separator := FALSE;

        // transitions
        IF NOT #b3_PalletNotPresent             // pallet inserted
            AND #b2_PalletEmpty                 // pallet empty
        THEN
            #stepnumber := 2;                   // next step
        END_IF;

    2: // step2 - move Pusher to home position
        // actions
        #q3_Pusher := NOT #s3_PusherHomePos     // move pusher to home position
        AND NOT #b3_PalletNotPresent;           // pallet inserted

        IF #q3_Pusher THEN                      // if pusher is activated
            #counter := 0;                      // reset counter
        END_IF;

        // transitions
        IF #s3_PusherHomePos  THEN              // pusher is in home position
            #stepnumber := 3;                   // next step
        END_IF;
```

```
    6: // step 6 - count up
        // actions
        #counter := #counter + 1;               // count up workpiece

        IF #counter < 3 THEN                    // if counter is smaller than 3
            #stepnumber := 2;                   // jump to 2
        ELSE
            #stepnumber := 7;                   // Start discharge
        END_IF;

    7: // step 7 - open separator
        // actions
        #q6_Separator := FALSE;                 // open separator

        // transitions
        IF #instSeparatorOpen.Q                 // separator opened
            AND #b2_PalletEmpty                 // pallet is empty
            AND NOT #b3_PalletNotPresent        // palette is present
        THEN
            #stepnumber := 8;                   // next step
        END_IF;

    8: // step 8 - drive pusher
        // actions
        #q3_Pusher := TRUE;                     // drive pusher

        IF #q3_Pusher THEN                      // if pusher driven
            #counter := 0;                      // reset counter
        END_IF;

        // transitions
        IF #instEdgeS3.Q THEN                   // pusher back in home position arrived
            #stepnumber := 2;                   // jump to 2
        END_IF;
END_CASE;
```

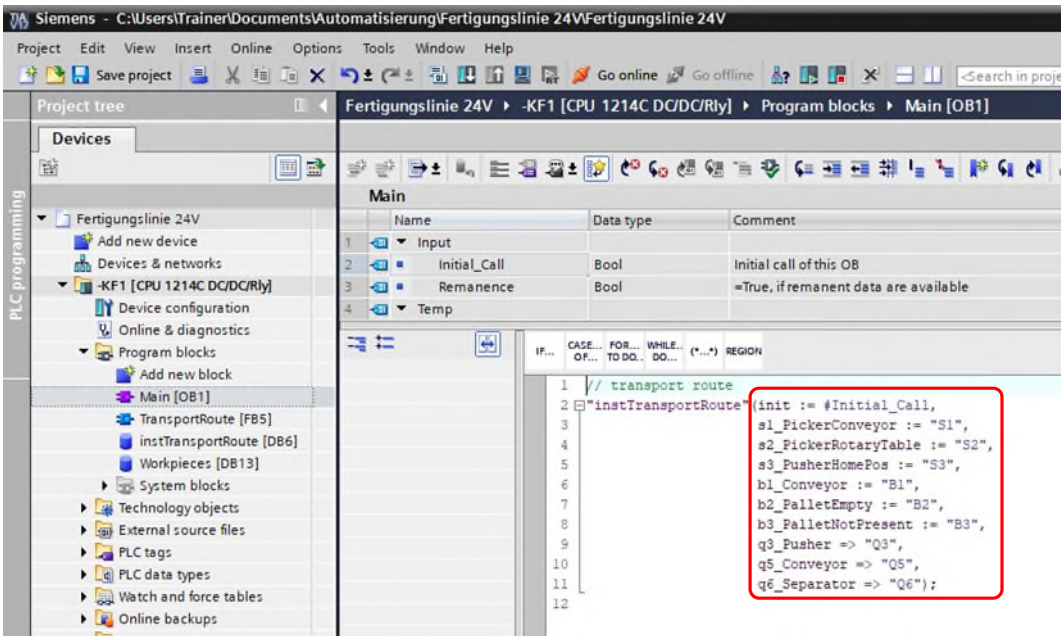
7. Program the time functions according to the CASE structure:

```
// timer
// transport time
#instTransportTime(IN := #q5_Conveyor,
                   PT := t#3s);

// Time Close separator
#instSeparatorClose(IN := #q6_Separator,
                    PT := t#500ms);
// Time Open separator
#instSeparatorOpen(IN := NOT #q6_Separator,
                   PT := t#500ms);
```

8. Call up the function module in "MAIN" and create an instance:



9. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

10. Commission the system in a structured manner with the aid of a commissioning protocol.

Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_02_Transportstrecke_SCL.zap17".

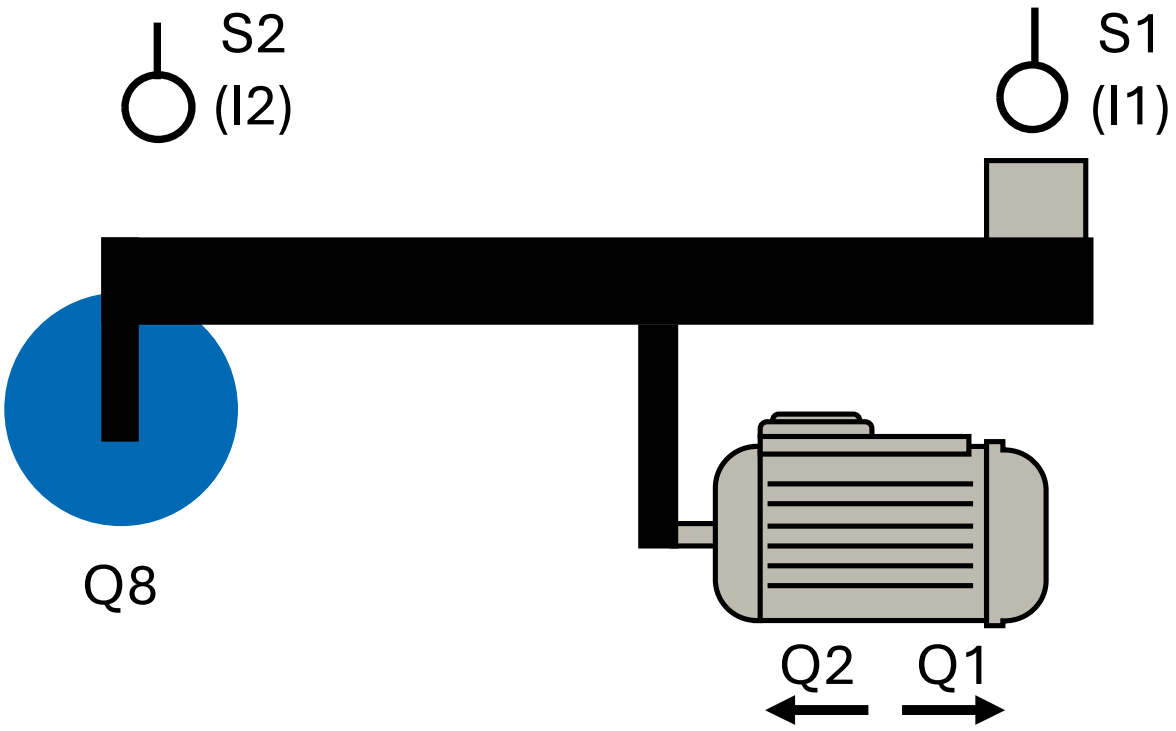
10.7 Exercise: Convert GRAFCET sequence chain into program code [FBD] - Converter

Target:

I can create the automatic program for the converter of the production line using the function description and the sequence chain created in GRAFCET.

Task:

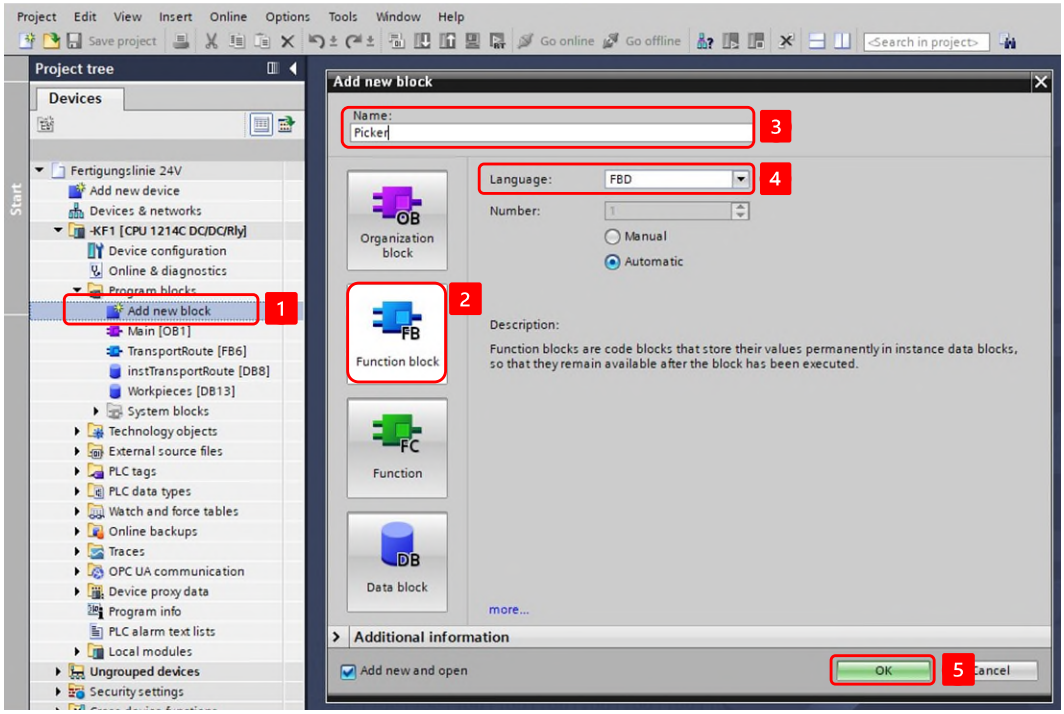
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



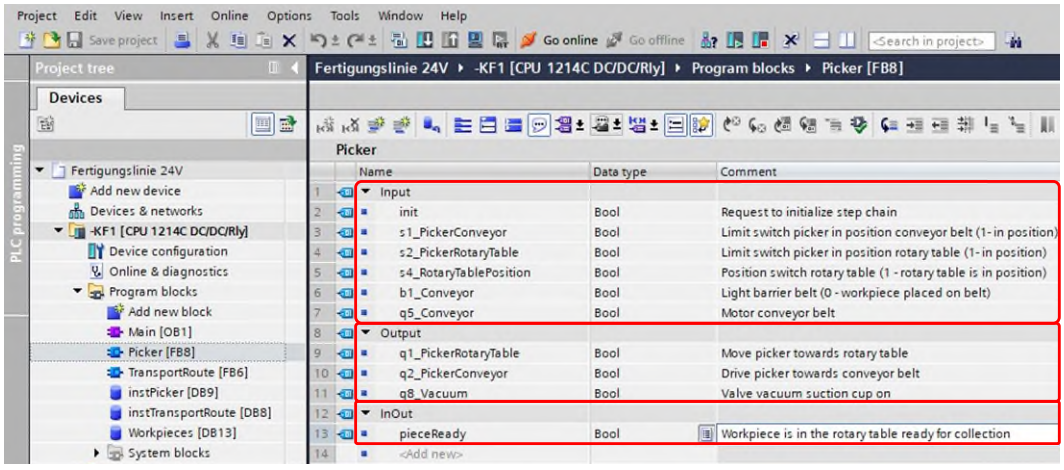
Picture 3 System diagram - converter

Procedure:

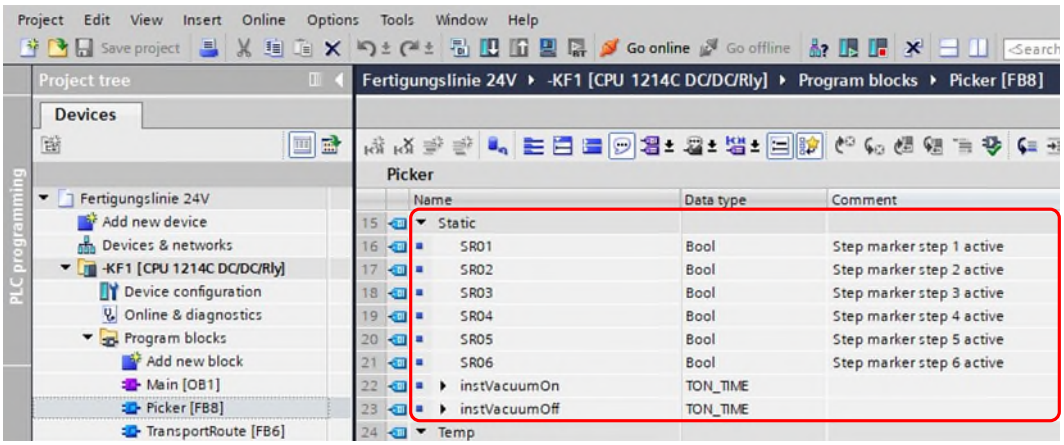
1. Create a new function block, select the desired programming language and assign a meaningful name:



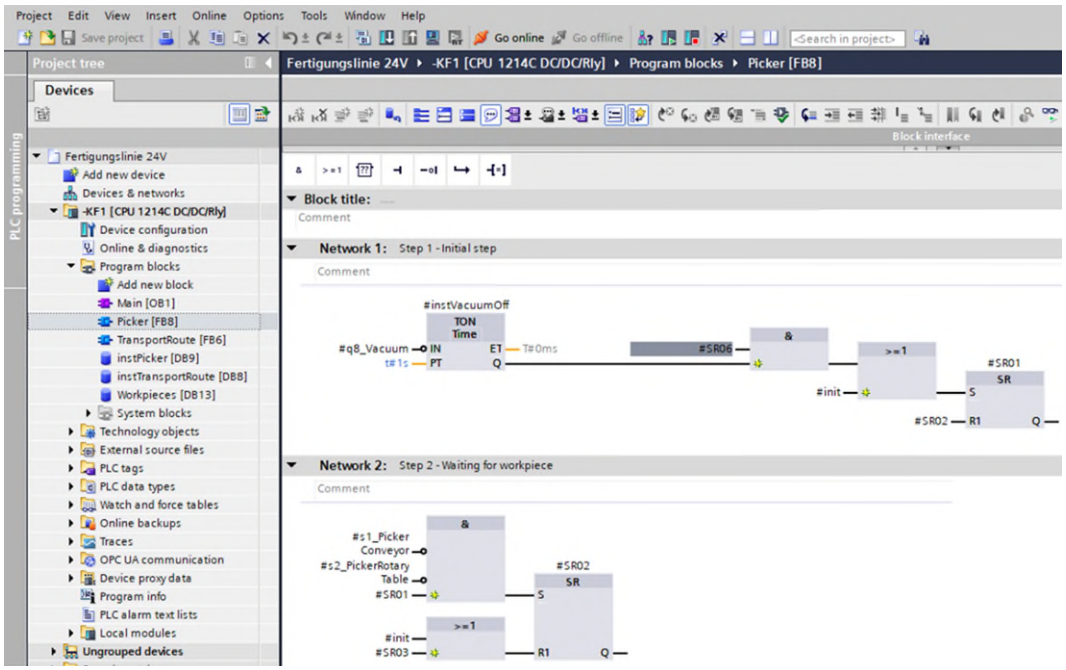
2. Declare variables for the sensors and actuators, a variable for initializing the step chain and a variable for transferring the workpiece information in the function block interface:



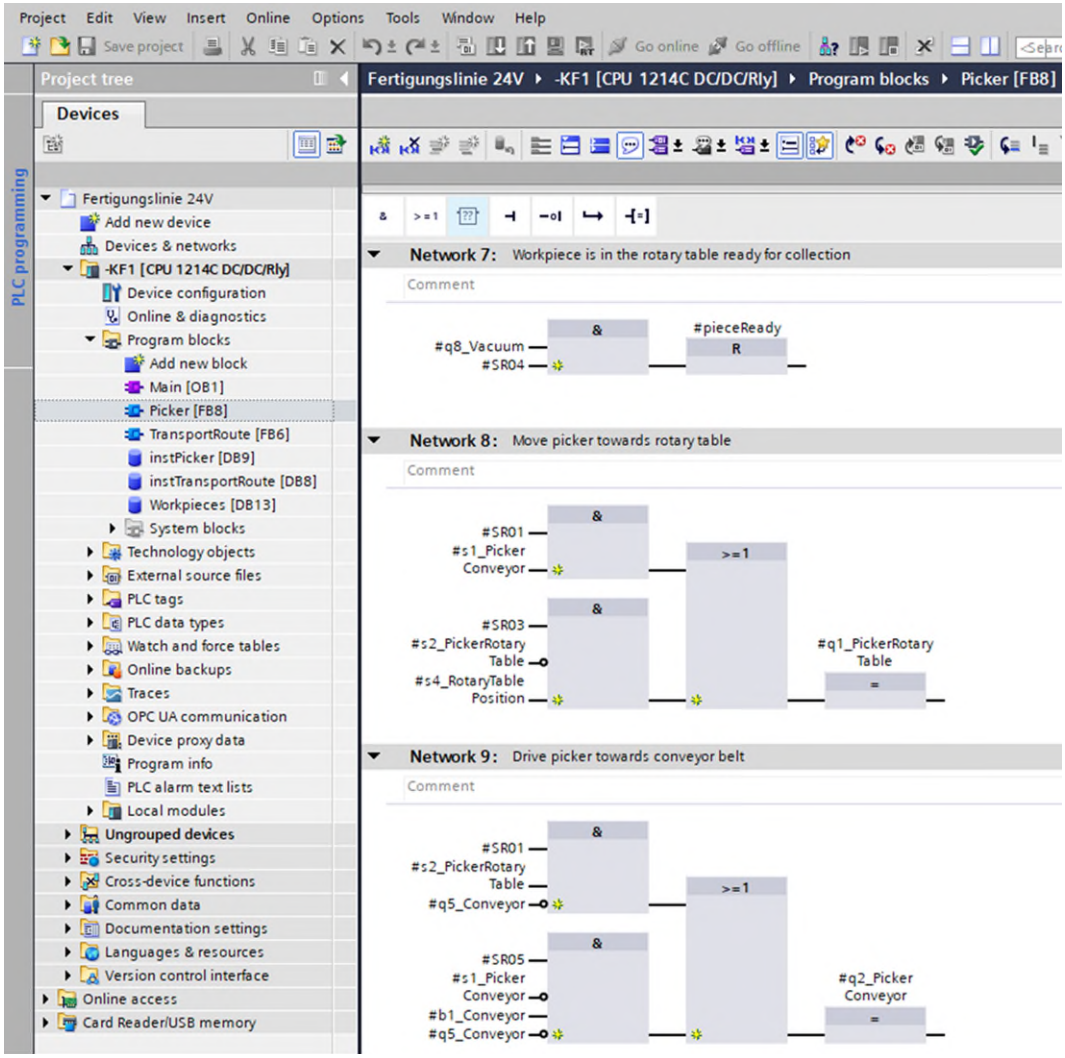
3. Declare the step flags in the static area of the function block interface, as well as two instances for the delay of the vacuum signal:



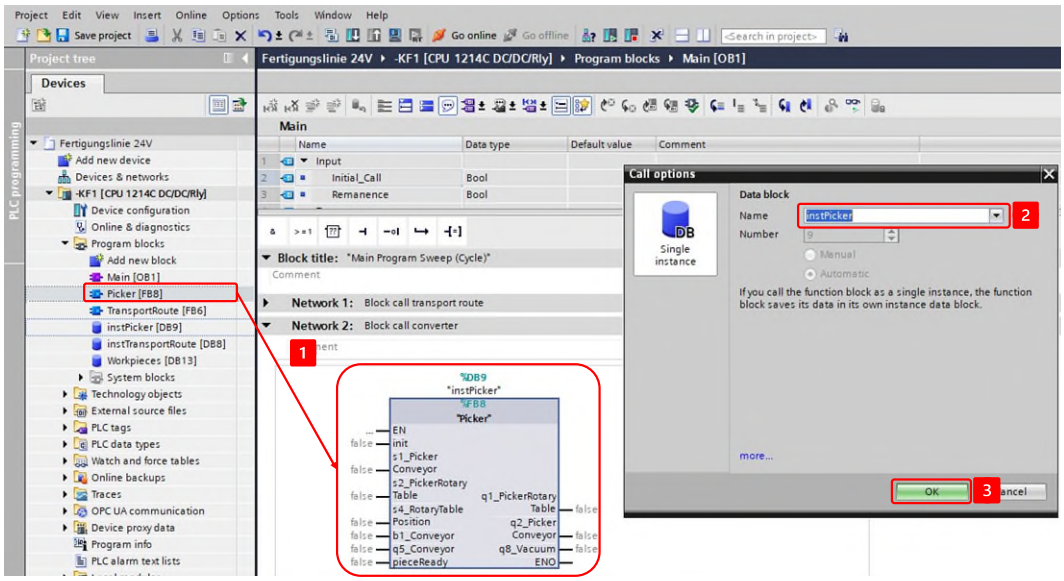
4. Implement the individual steps using flip-flops based on the GRAFCET. A new network must be used for each step:



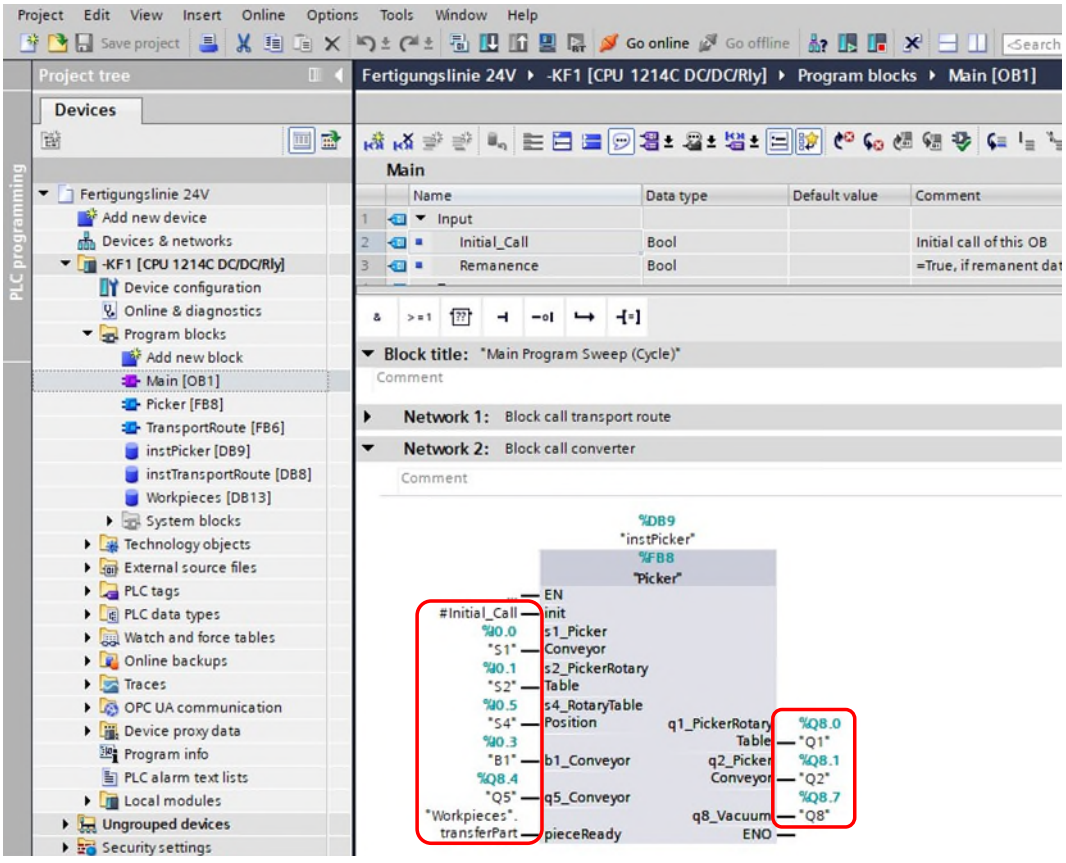
5. Assign the actions below the step chain in the next networks.



6. Call up the function module in "MAIN" and create an instance:



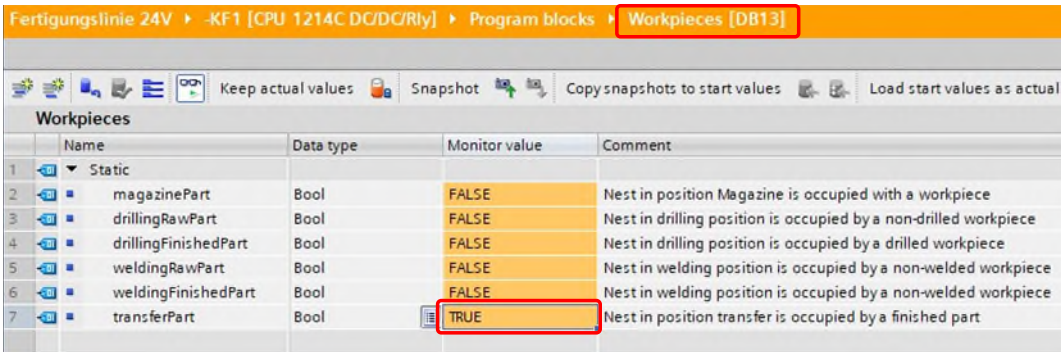
7. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

8. Commission the system in a structured manner with the aid of a commissioning protocol.

In order for the transfer unit to pick up a workpiece from the turntable, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_03_Umsetzer_FUP.zap17".

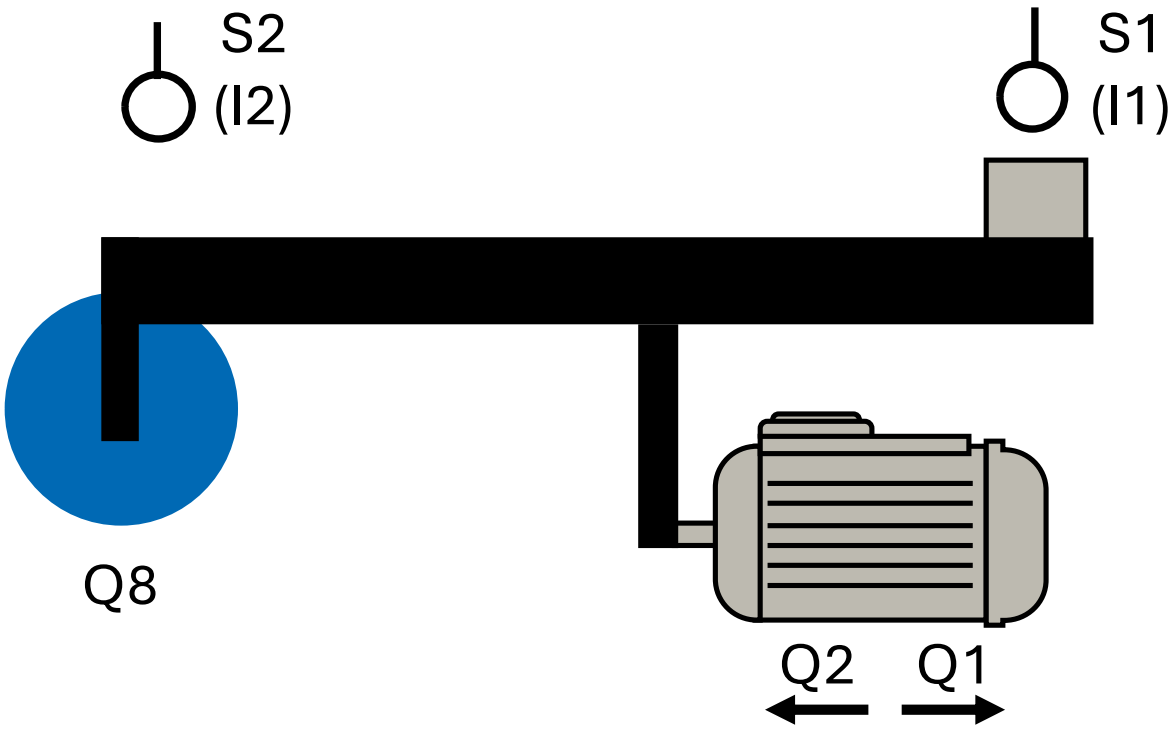
10.8 Exercise: Convert GRAFCET sequence chain into program code [ST / SCL] - Converter

Target:

I can create the automatic program for the converter of the production line based on the function description and the sequence chain created in GRAFCET.

Task:

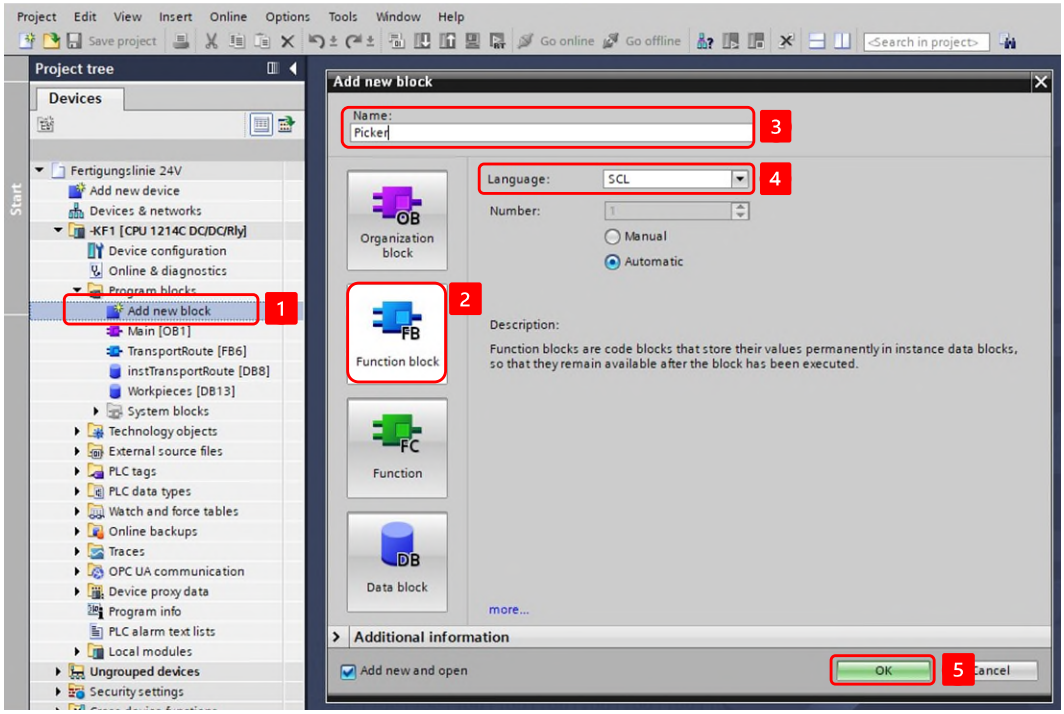
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



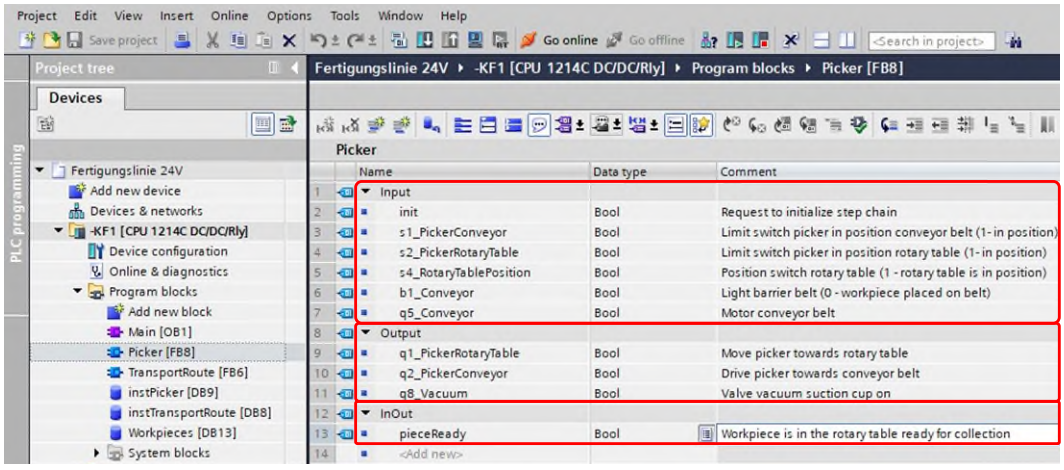
Picture 4 System diagram - converter

Procedure:

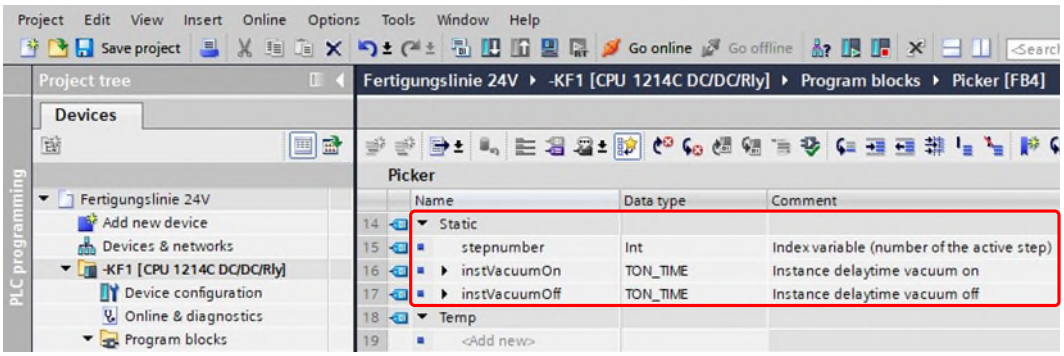
1. Create a new function block, select the desired programming language and assign a meaningful name:



2. Declare variables for the sensors and actuators, a variable for initializing the step chain and a variable for transferring the workpiece information in the function block interface:



3. Declare the index variable in the static area of the function block interface, as well as two instances for the delay of the vacuum signal:



4. Program the initialization of the chain:

```
// initialize step chain
IF #init THEN
    #stepnumber := 1;    // initialize step chain
END_IF;
```

5. Implement the individual steps from the GRAFCET in the following CASE structure. A new CASE must be created in the structure for each step, which represents the step number:

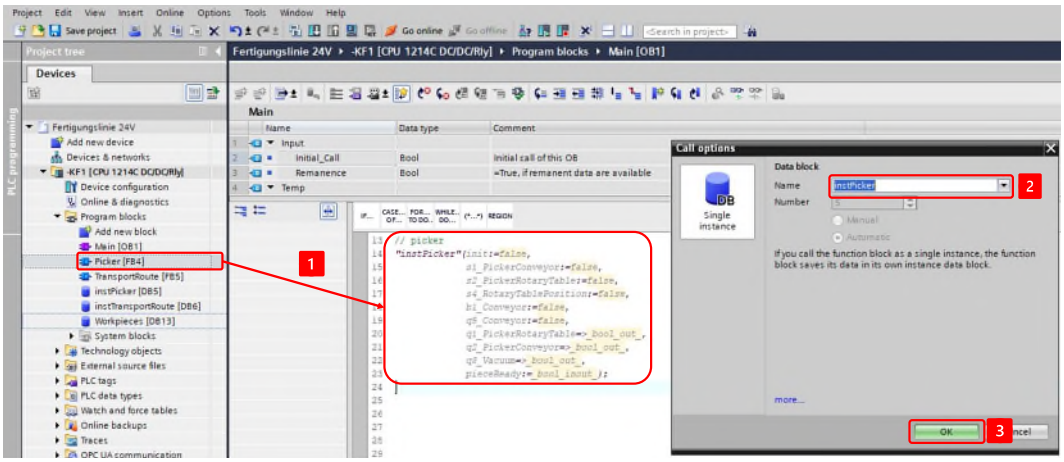
All actions are reset in the initial step. This ensures that no actions remain set if an active step chain is aborted by an initialization request.

```
// step chain
CASE #stepnumber OF
    1: // step 1 - init step
        // actions
        #q1_PickerRotaryTable := #s1_PickerConveyor;
        #q2_PickerConveyor := #s2_PickerRotaryTable;

        #q8_Vacuum := FALSE;    // reset action

        // transitions
        IF NOT #s1_PickerConveyor
            AND NOT #s2_PickerRotaryTable
        THEN
            #stepnumber := 2;
        END_IF;

    2: // step 2 - waiting for workpiece
        // actions

        // transitions
        IF #s4_RotaryTablePosition
            AND #pieceReady
        THEN
            #stepnumber := 3;
        END_IF;
```

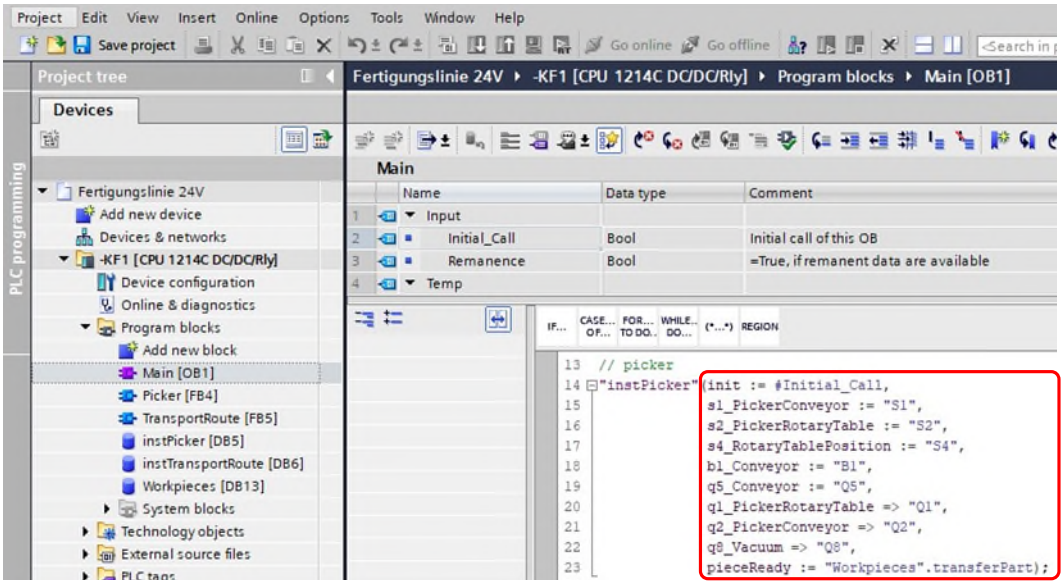
6. Program the time functions according to the CASE structure:

```
78  // delaytimes
79  // delaytime vacuum on
80  #instVacuumOn(IN := #q8_Vacuum,          // vacuum on
81                PT := t#500ms);
82
83  // delaytime vacuum off
84  #instVacuumOff(IN := NOT #q8_Vacuum,     // vacuum off
85                 PT := t#1s);
```

7. Call up the function module in "MAIN" and create an instance:



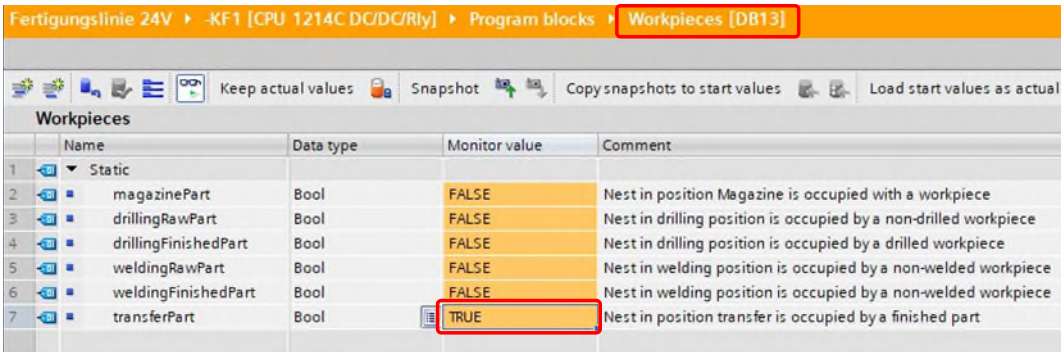
8. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

9. Commission the system in a structured manner with the aid of a commissioning protocol.

In order for the transfer unit to pick up a workpiece from the turntable, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_03_Umsetzer_SCL.zap17".

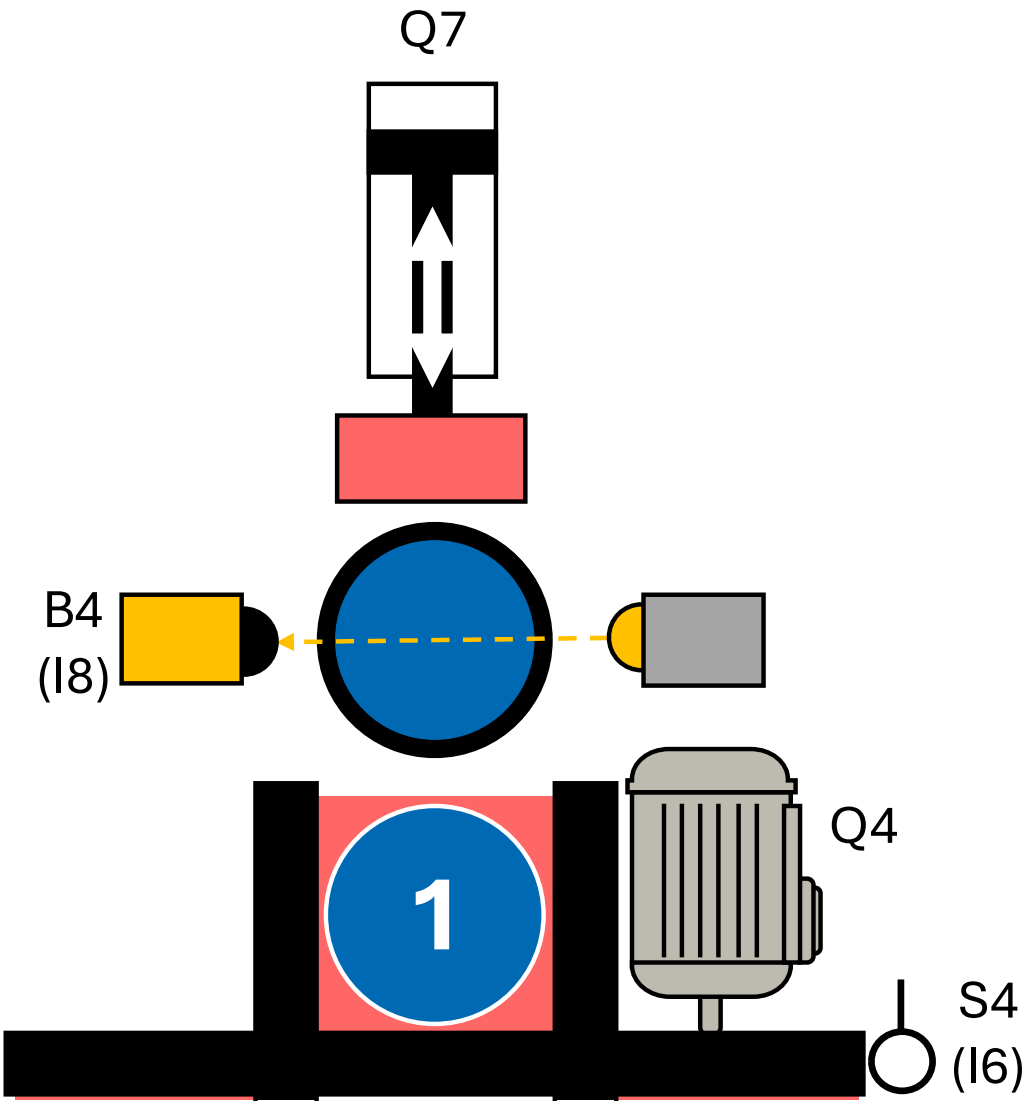
10.9 Exercise: Convert GRAFCET sequence chain into program code [FUP] - Magazine

Target:

I can create the automatic program for the magazine of the production line based on the functional description and the sequence chain created in GRAFCET.

Task:

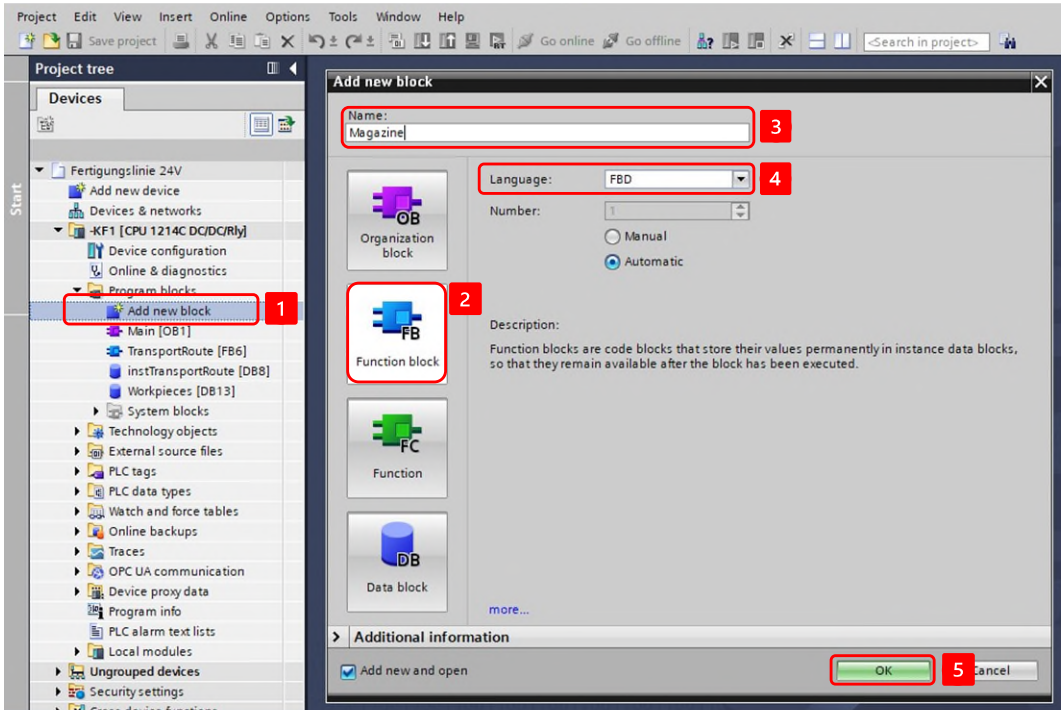
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



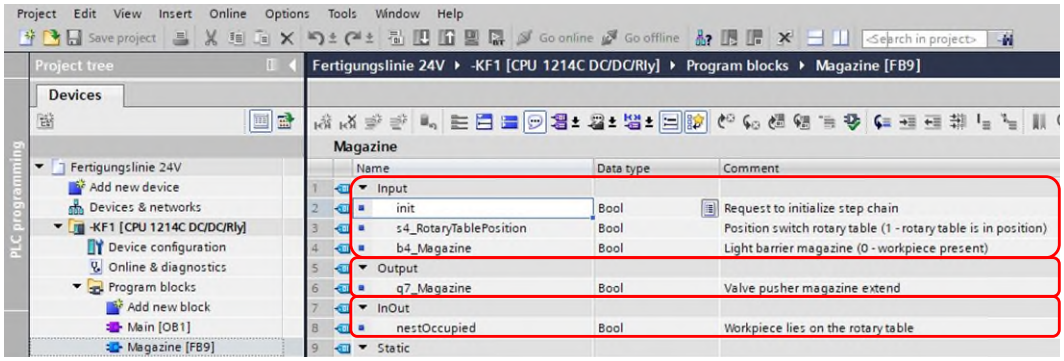
Picture 5 System diagram - Magazine

Procedure:

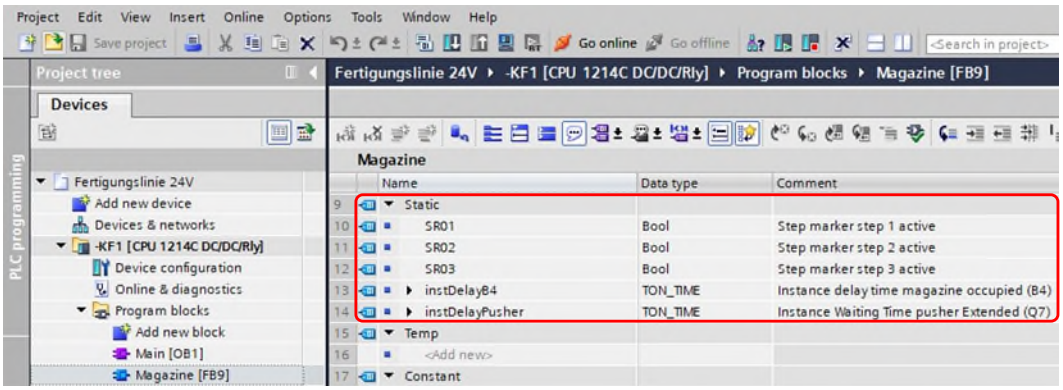
1. Create a new function block, select the desired programming language and assign a meaningful name:



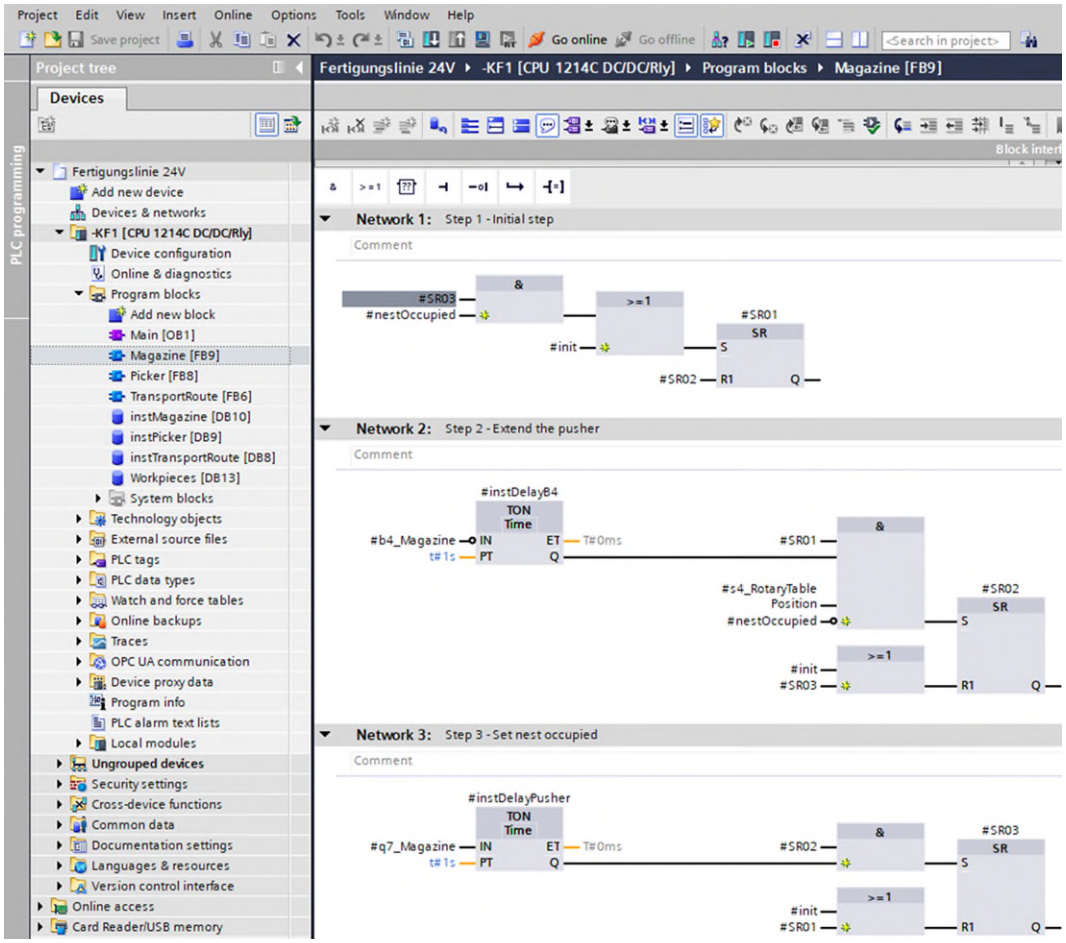
2. Declare variables for the sensors and actuators, a variable for initializing the step chain and a variable for transferring the workpiece information in the function block interface:



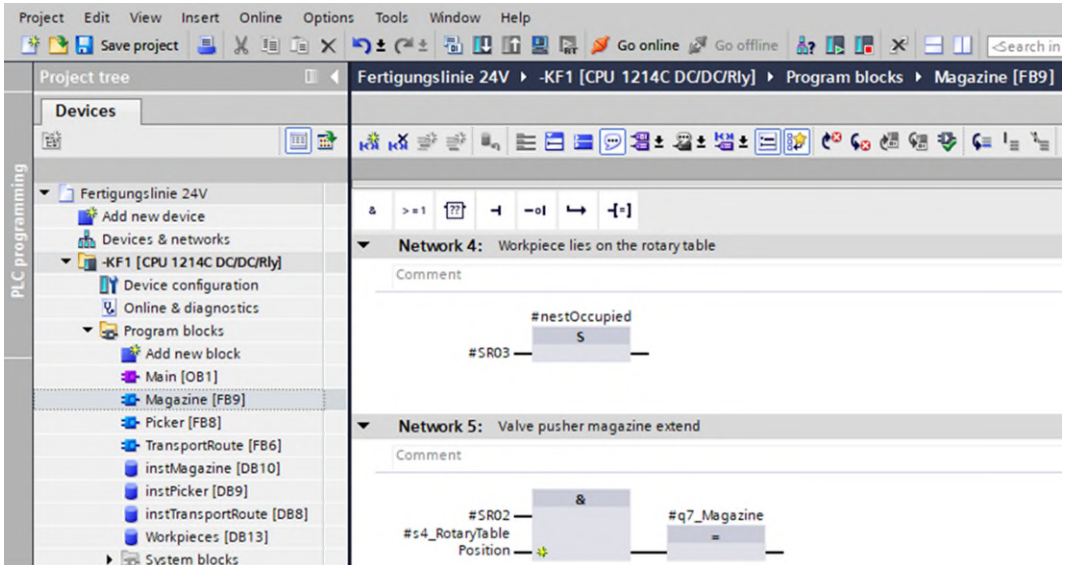
3. Declare the step flags in the static area of the function block interface, as well as two instances for the wait times:



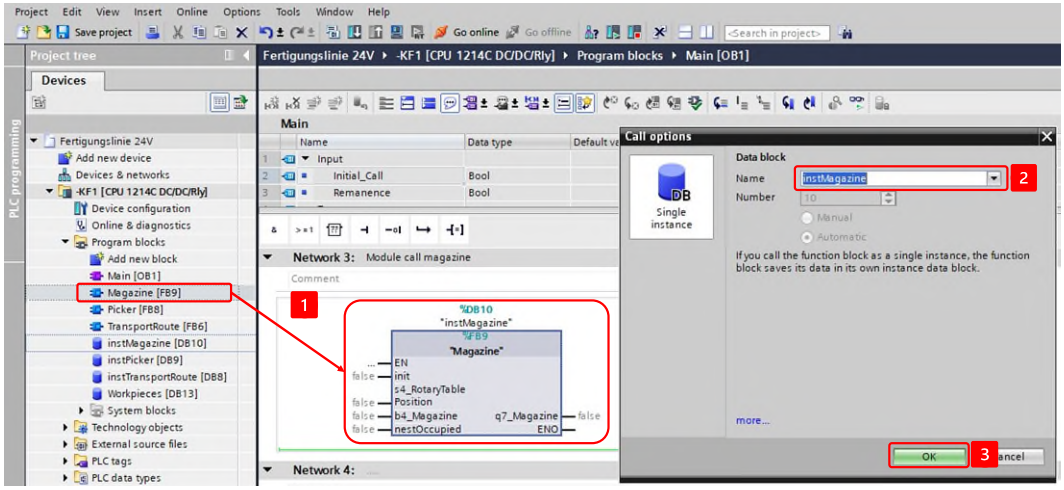
4. Implement the individual steps using flip-flops based on the GRAFCET. A new network must be used for each step:



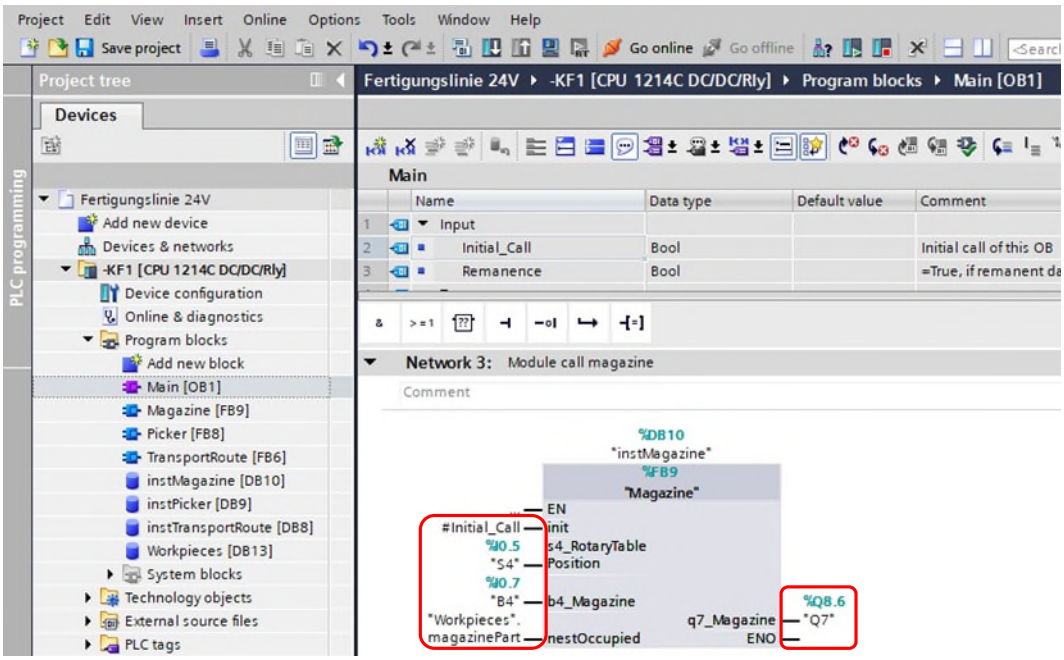
5. Assign the actions below the step chain in the next networks:



6. Call up the function module in "MAIN" and create an instance:



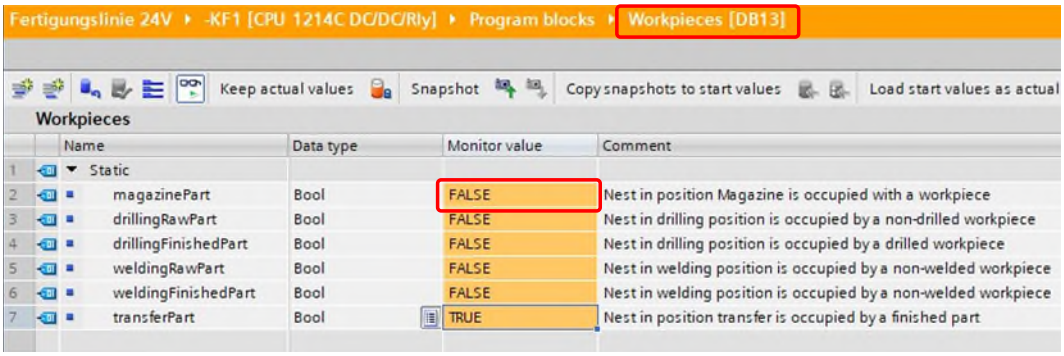
7. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

8. Commission the system in a structured manner with the aid of a commissioning protocol.

To place another workpiece on the turntable, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_04_Magazin_FUP.zap17".

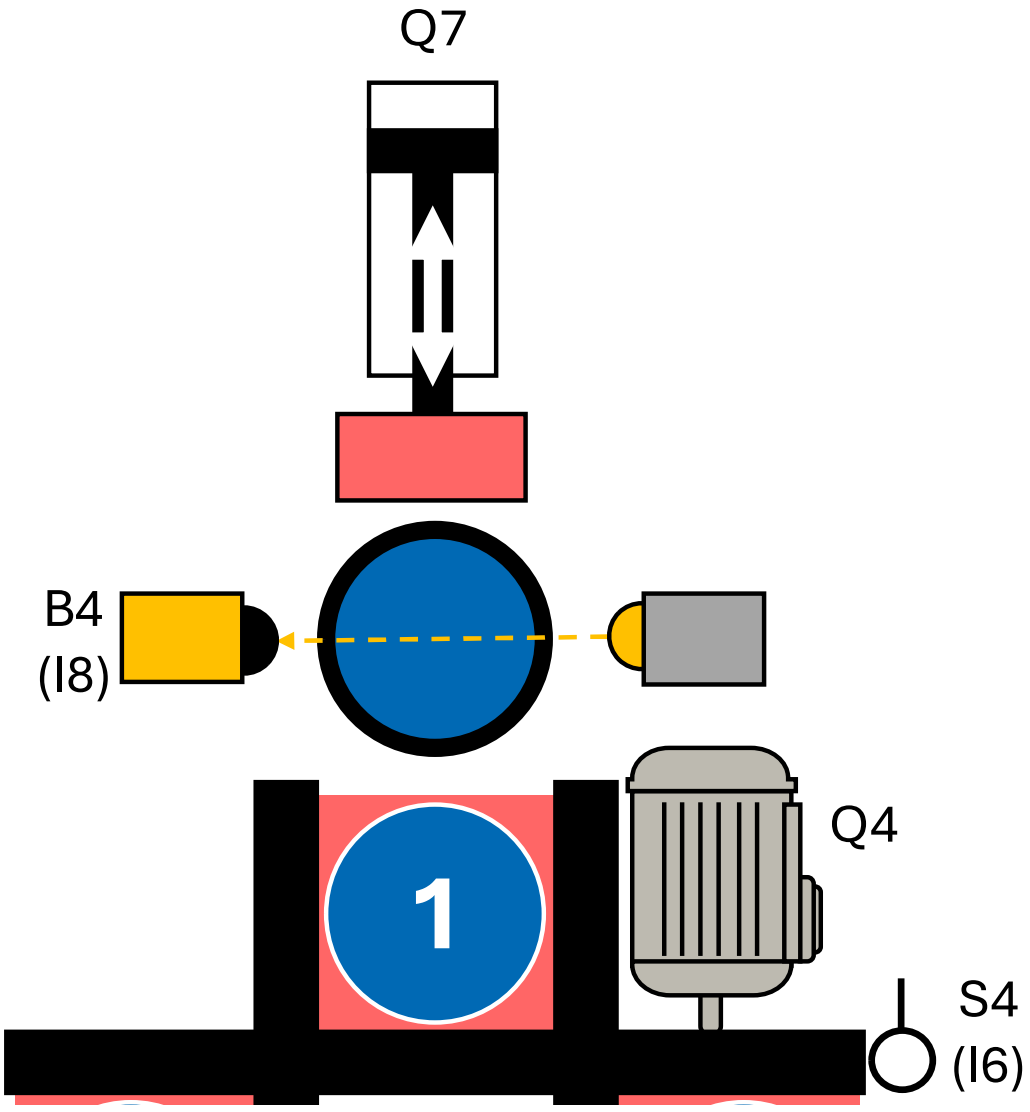
10.10 Exercise: Convert GRAFCET sequence chain into program code [ST / SCL] - Magazine

Target:

I can create the automatic program for the magazine of the production line based on the functional description and the sequence chain created in GRAFCET.

Task:

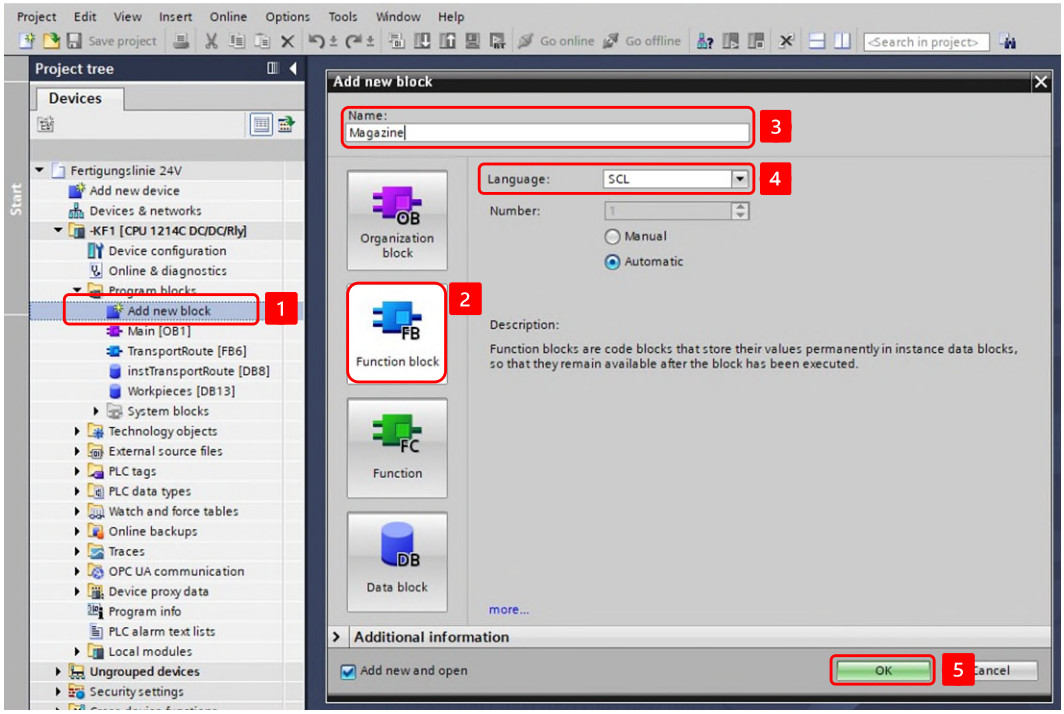
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.



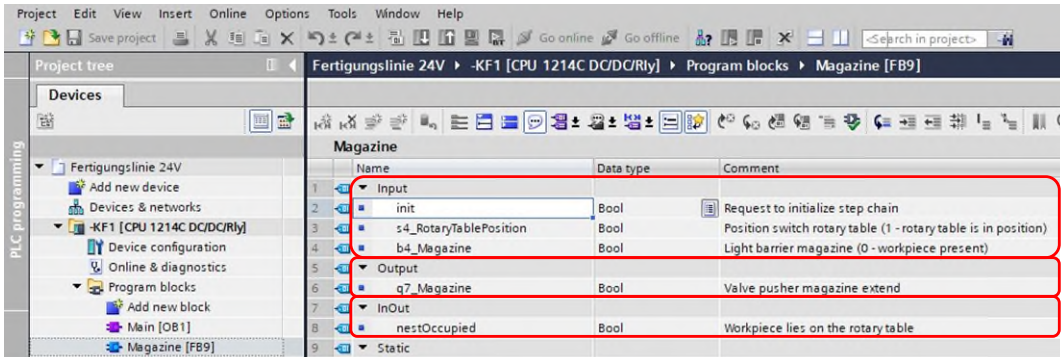
Picture 6 System diagram - Magazine

Procedure:

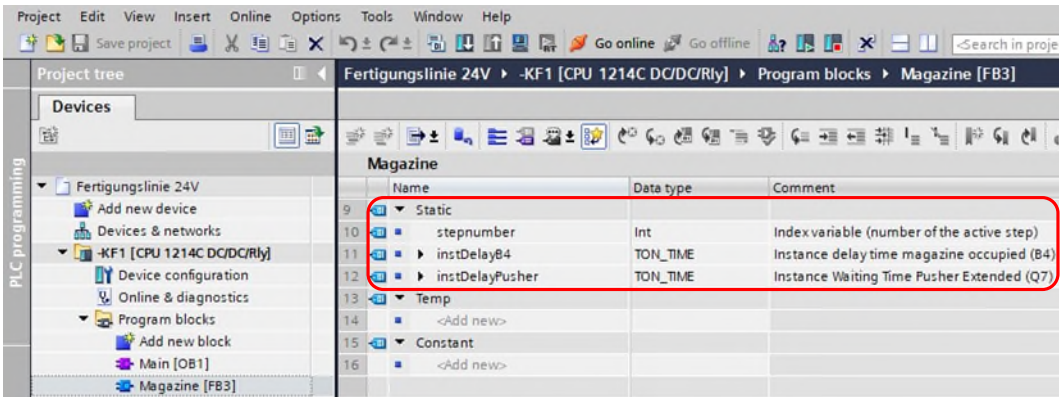
1. Create a new function block, select the desired programming language and assign a meaningful name:



2. Declare variables for the sensors and actuators, a variable for initializing the step chain and a variable for transferring the workpiece information in the function block interface:



3. Declare the index variable in the static area of the function block interface, as well as two instances for the waiting times:



4. Program the initialization of the chain:

```
// initialize step chain
IF #init THEN
    #stepnumber := 1;   // initialize step chain
END_IF;
```

5. Implement the individual steps from the GRAFCET in the following CASE structure. A new CASE must be created in the structure for each step, which represents the step number:

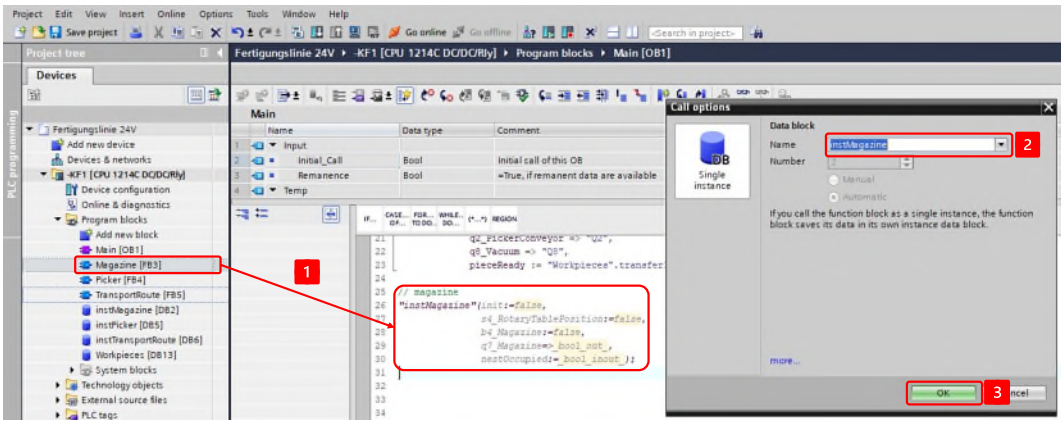
All actions are reset in the initial step. This ensures that no actions remain set if an active step chain is aborted by an initialization request.

```
// step chain
CASE #stepnumber OF
    1: // step 1 - initstep
        // reset all actions
        #q7_Magazine := FALSE;

        // transition
        IF #instDelayB4.Q                       // magazine not empty
            AND #s4_RotaryTablePosition         // rotary table is in position
            AND NOT #nestOccupied               // rotary table nest is empty
        THEN
            #stepnumber := 2;                   // next step
        END_IF;

    2: // step 2 - extend pusher
        // actions
        #q7_Magazine := #s4_RotaryTablePosition;    // extend pusher aufahren, when table in position

        // transitions
        IF #instDelayPusher.Q THEN              // pusher extended long time enough
            #q7_Magazine := FALSE;              // reset action
            #stepnumber := 3;                   // next step
        END_IF;

    3: // step 3 - Nest belegt setzen
        // actions
        #nestOccupied := TRUE;                  // set nest is occupied

        // transitions
        IF #nestOccupied THEN                   //  rotary table nest is occupied
            #stepnumber := 1;                   // jump to init step
        END_IF;
END_CASE;
```

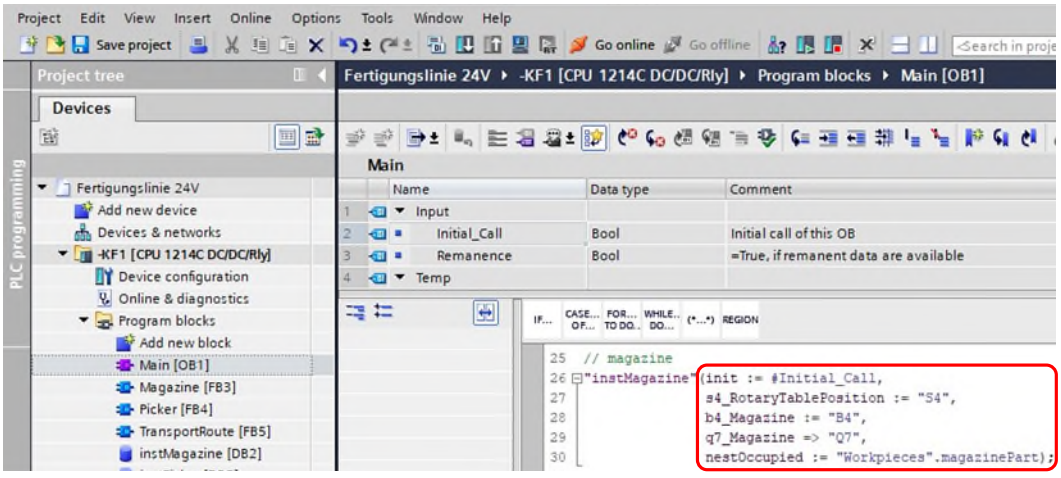
6. Program the time functions according to the CASE structure:

```
// delaytimes
// Light barrier interrupted B4
#instDelayB4(IN := NOT #b4_Magazine,      // Light barrier interrupted
             PT := t#1s);

// pusher activated Q7
#instDelayPusher(IN := #q7_Magazine,     // pusher activated
                 PT := t#2s);
```

7. Call up the function module in "MAIN" and create an instance:



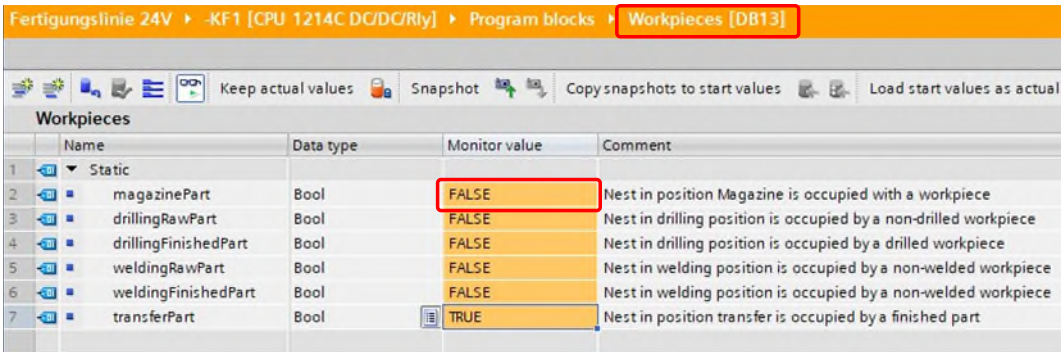
8. Connect the function block interface with the input and output variables from your variable table:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

9. Commission the system in a structured manner with the aid of a commissioning protocol.

To place another workpiece on the turntable, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_04_Magazin_SCL.zap17".

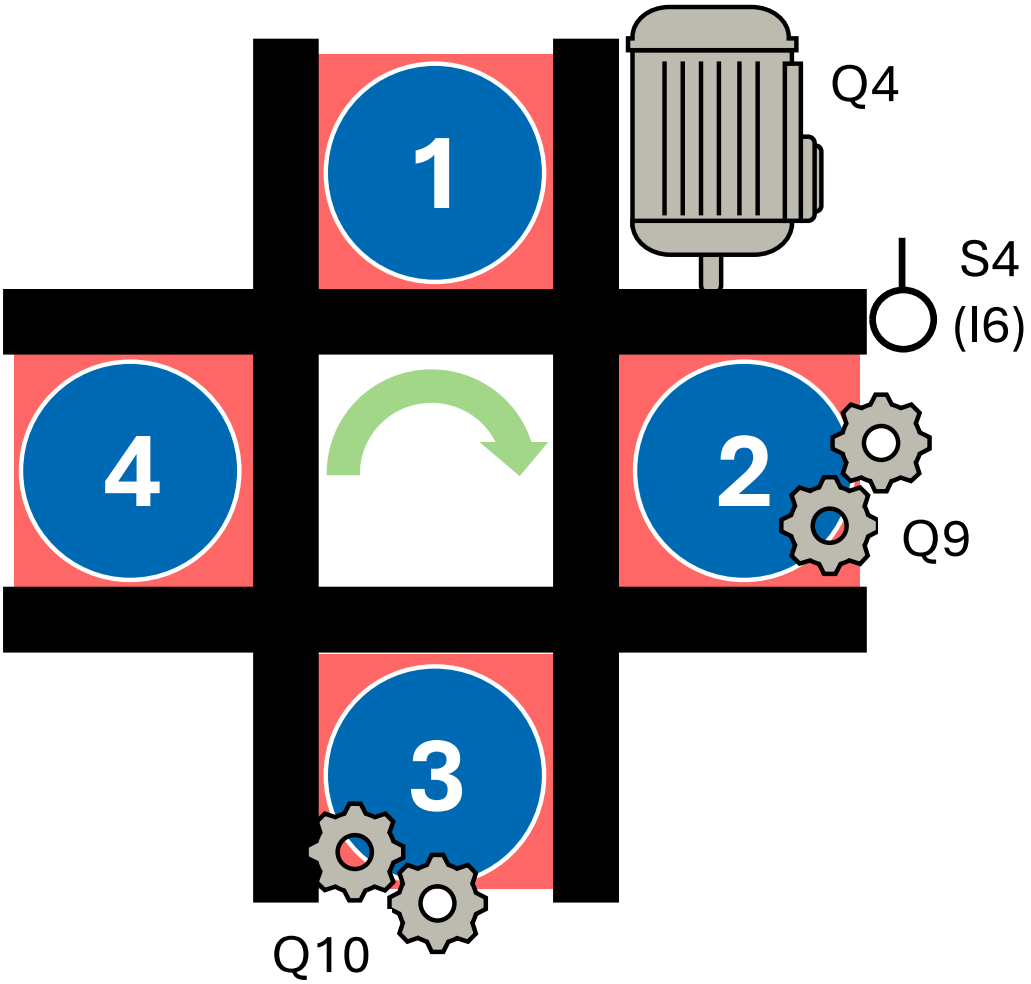
10.11 Exercise: Convert GRAFCET sequence chain into program code [FBD] - Processing station

Target:

I can create the automatic program for a processing station on the production line using the functional description and the sequence chain created in GRAFCET.

Task:

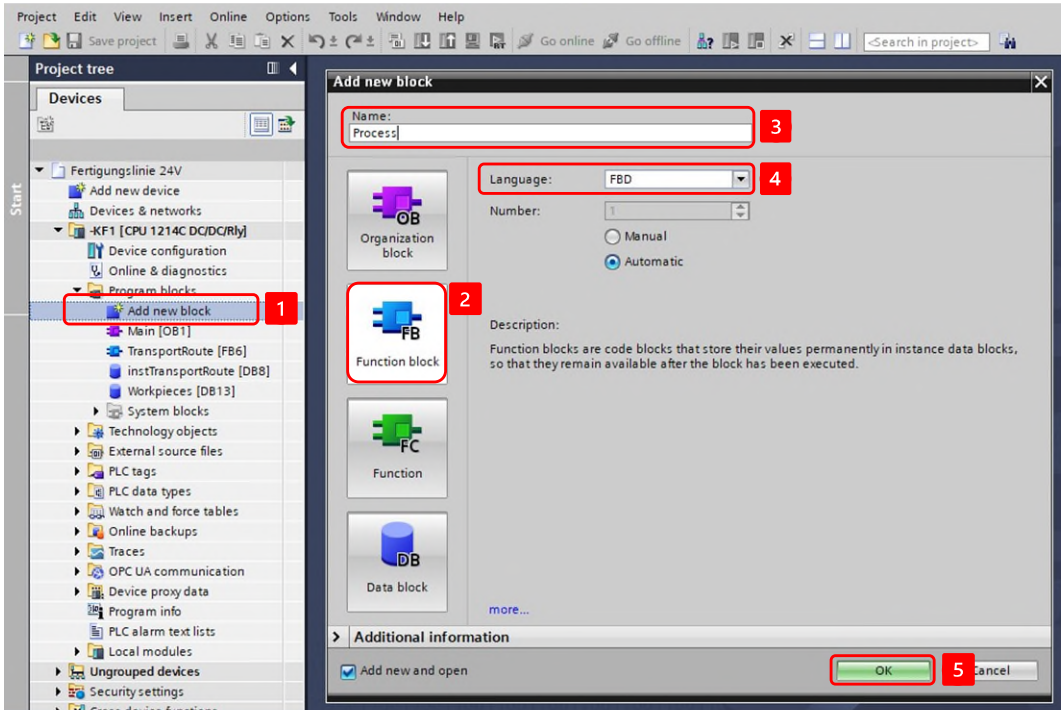
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.
The function block must be called twice, one instance for the drilling station (Q9) and another instance for welding (Q10).



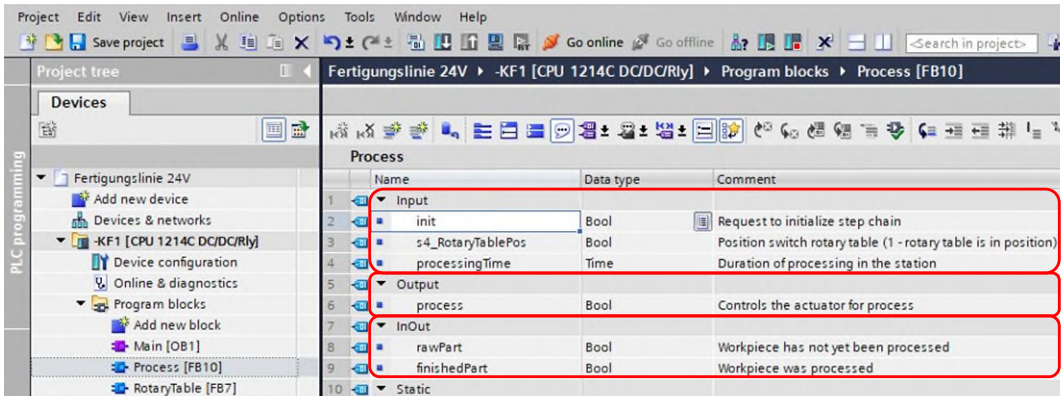
Picture 7 System diagram - processing station

Procedure:

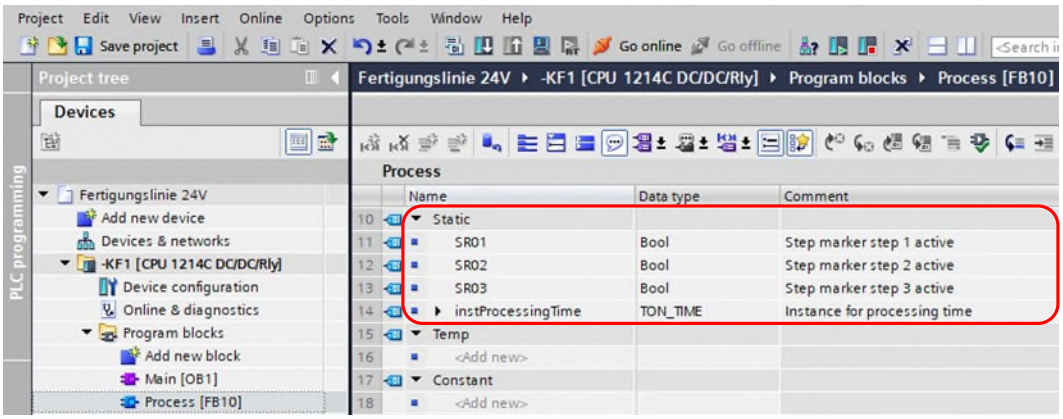
1. Create a new function block, select the desired programming language and assign a meaningful name:



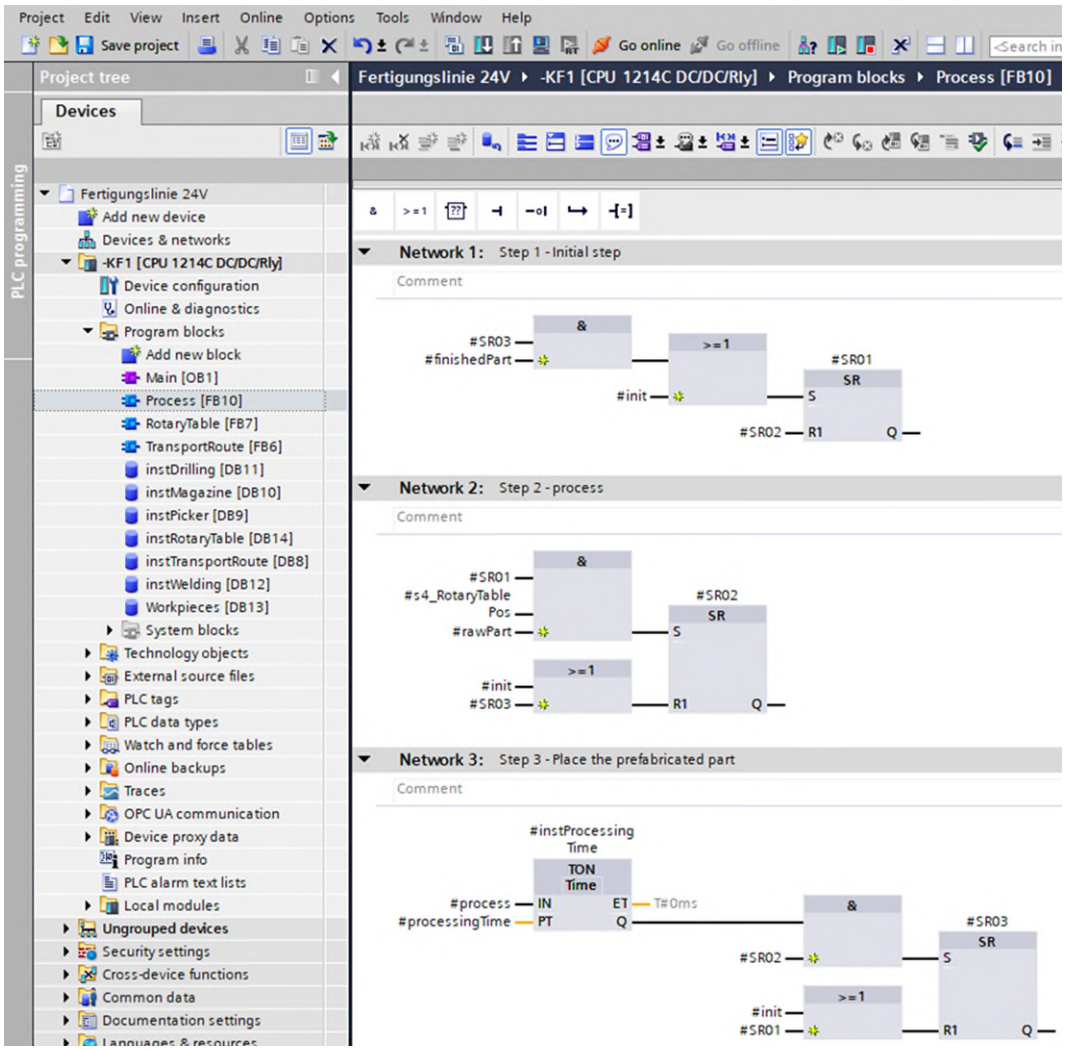
2. Declare variables for the sensors and actuators, a variable for initializing the step chain, for the processing time and two variables for transferring the workpiece information in the function block interface:



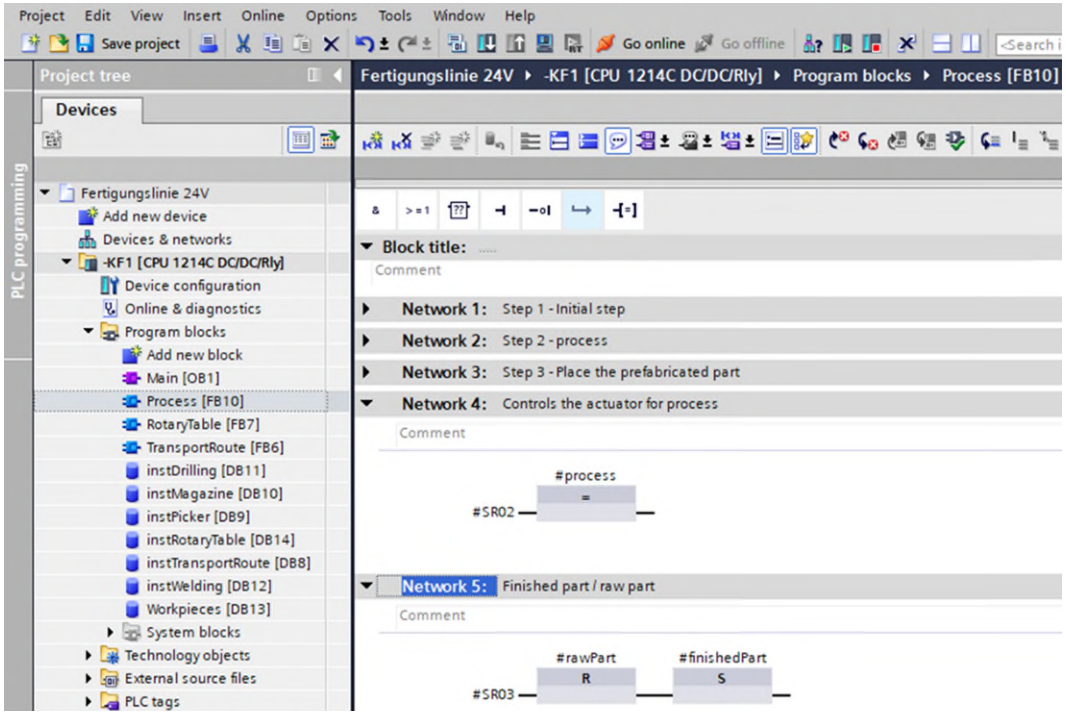
3. Declare the step flags in the static area of the block interface, as well as an instance for the processing time:



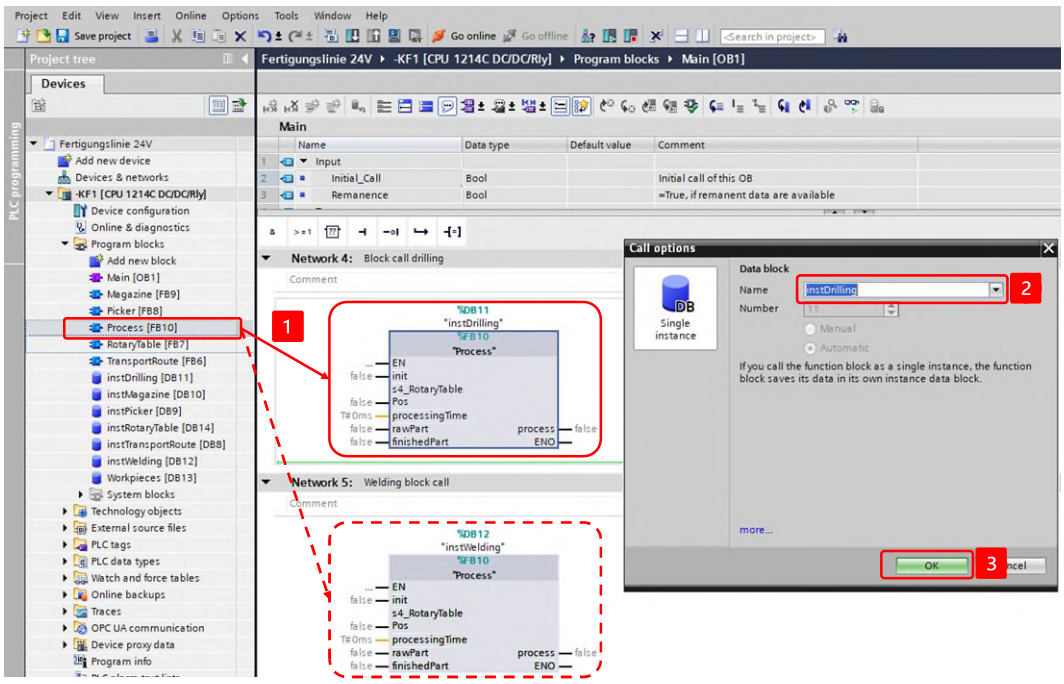
4. Implement the individual steps using flip-flops based on the GRAFCET. A new network must be used for each step:



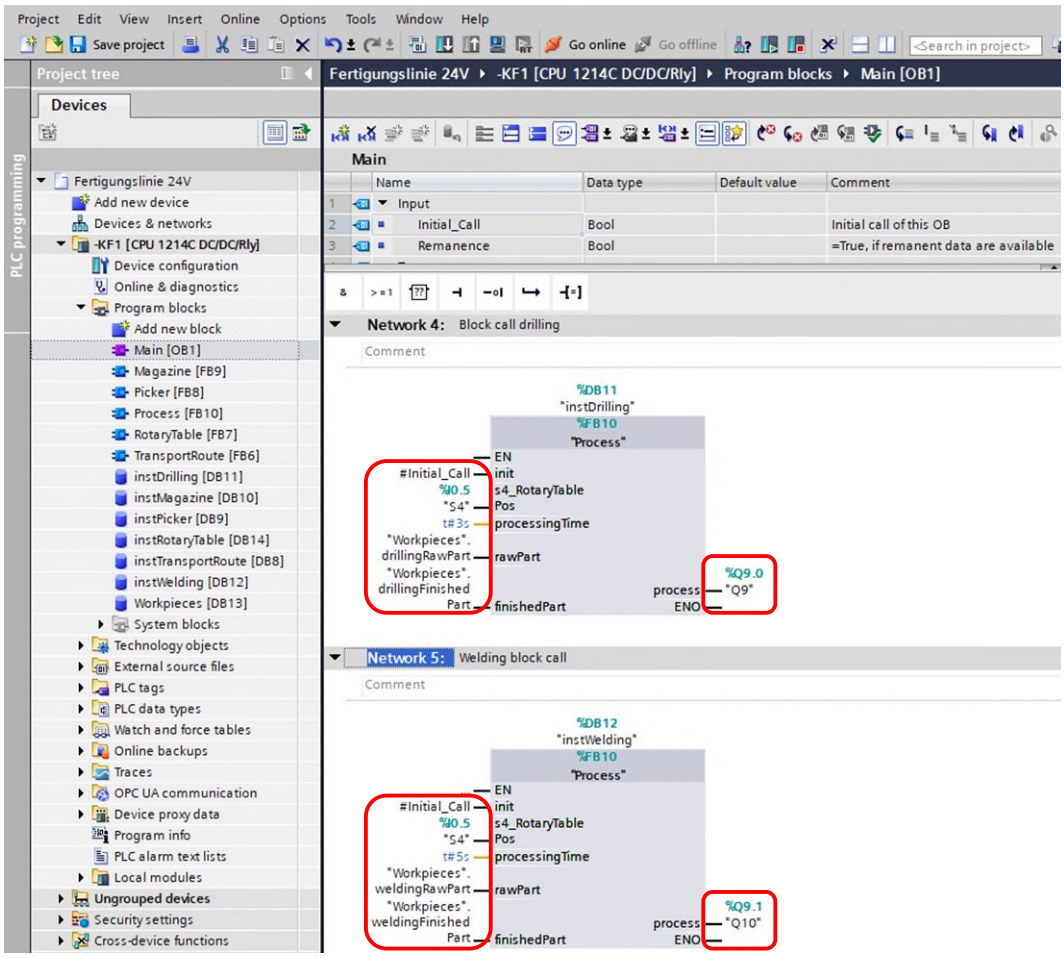
5. Assign the actions below the step chain in the next networks:



6. Call the function module twice in "MAIN" and create a separate instance for each time:



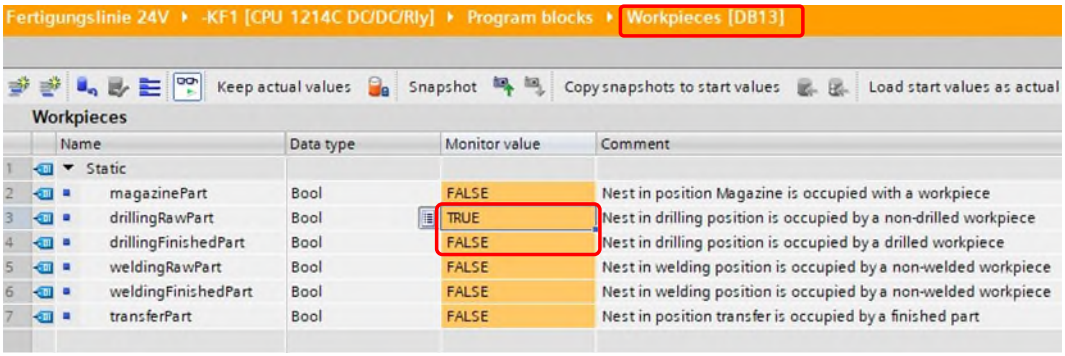
7. Connect the function block interface with the input and output variables from your variable table, as well as the processing time:



The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

8. Commission the system in a structured manner with the aid of a commissioning protocol.

In order for another workpiece to be processed in the station, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_05_Bearbeiten_FUP.zap17".

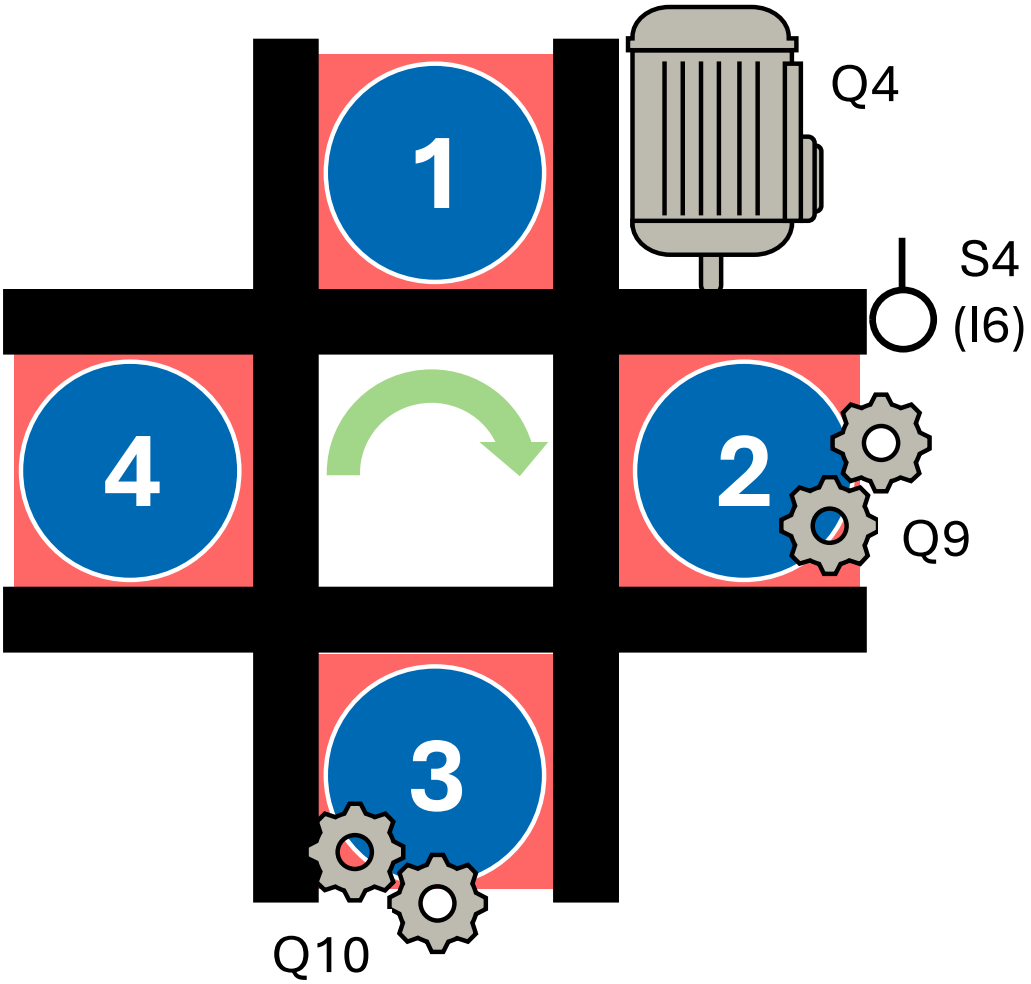
10.12 Exercise: Convert GRAFCET sequence chain into program code [ST / SCL] - Processing station

Target:

I can use the functional description and the sequence chain created in GRAFCET to create the automatic program for a processing station on the production line.

Task:

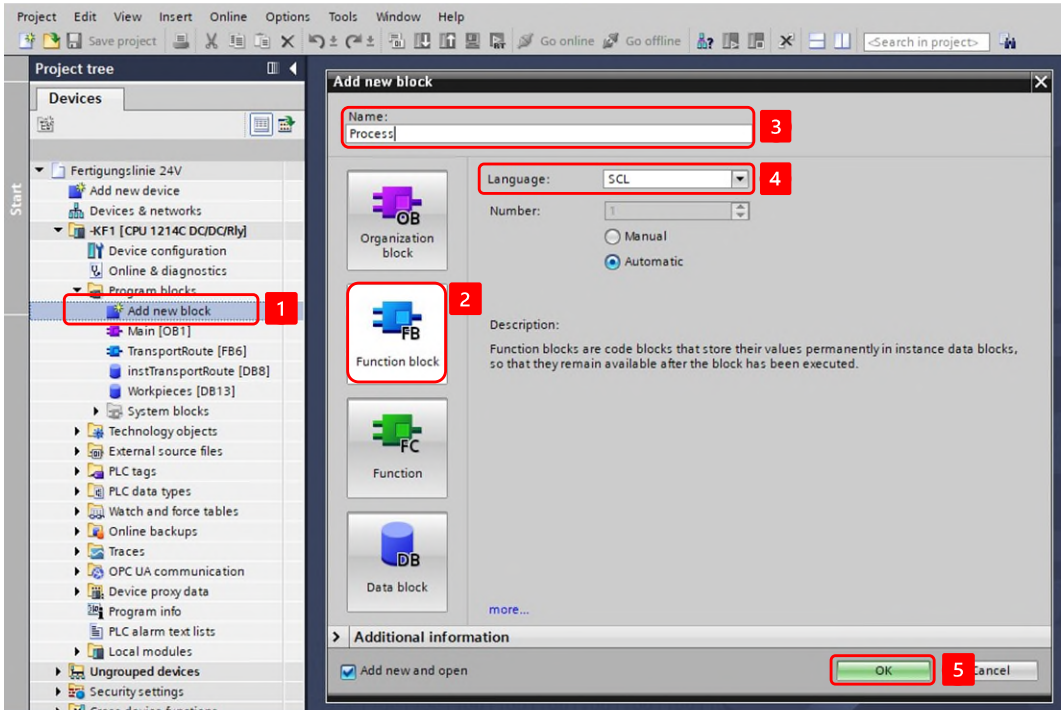
Create the PLC program for the automatic sequence so that the function is implemented based on the system description.
The chain is initialized when the control unit is switched on.
The function block must be called twice, one instance for the drilling station (Q9) and another instance for welding (Q10).



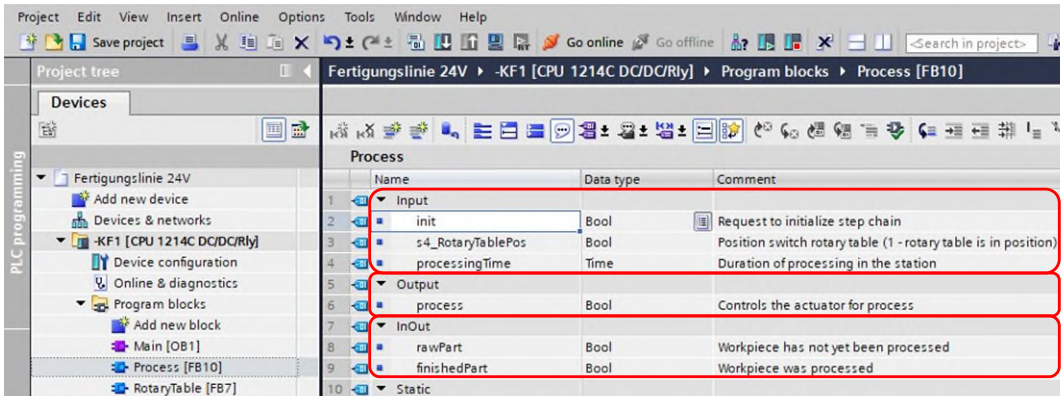
Picture 8 System diagram - processing station

Procedure:

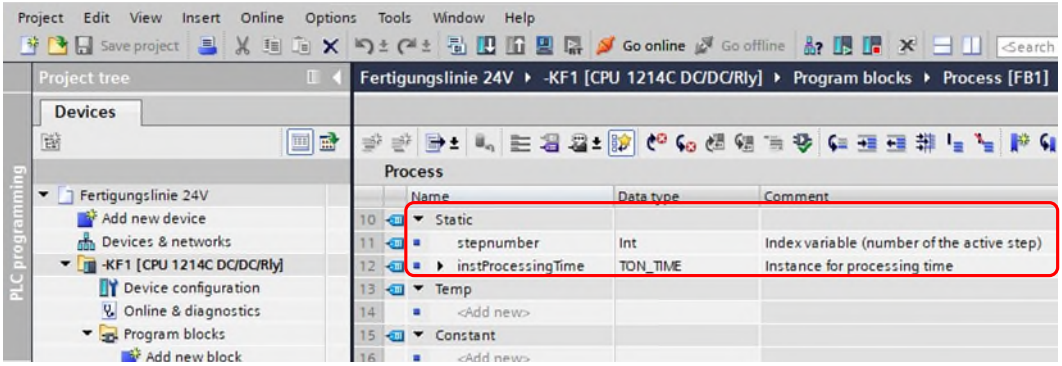
1. Create a new function block, select the desired programming language and assign a meaningful name:



2. Declare variables for the sensors and actuators, a variable for initializing the step chain, for the processing time and two variables for transferring the workpiece information in the function block interface:



3. Declare the index variable in the static area of the block interface, as well as an instance for the processing time:



4. Program the initialization of the chain:

```
// initialize step chain
IF #init THEN
    #stepnumber := 1;    // initialize step chain
END_IF;
```

5. Implement the individual steps from the GRAFCET in the following CASE structure. A new CASE must be created in the structure for each step, which represents the step number:

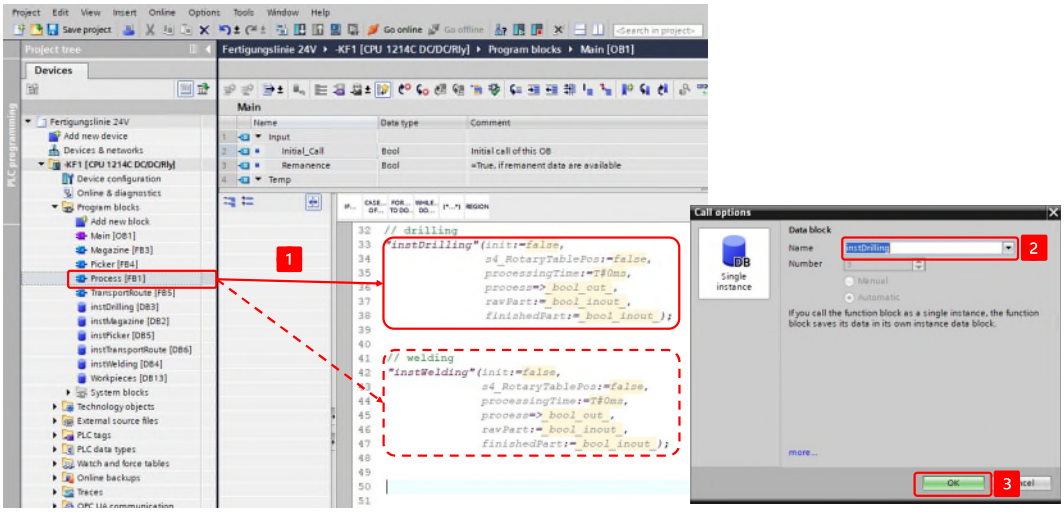
All actions are reset in the initial step. This ensures that no actions remain set if an active step chain is aborted by an initialization request.

```
// step chain
CASE #stepnumber OF
    1: // step 1 - initstep
        // reset all actions
        #process := FALSE;

        // transitions
        IF #s4_RotaryTablePos           // rotary table is in position
            AND #rawPart                // raw part in station
        THEN
            #stepnumber := 2;           // next step
        END_IF;
    2: // step 2 - process
        // actions
        #process := TRUE;               // set process

        // transitions
        IF #instProcessingTime.Q THEN   // Processing time expired
            #process := FALSE;          // reset action
            #stepnumber := 3;           // next step
        END_IF;

    3: // step 3 - set finished part
        // actions
        #finishedPart := TRUE;          // set finished part in station
        #rawPart := FALSE;              // reset raw part in station

        // transitions
        #stepnumber := 1;               // jump to init step
END_CASE;
```

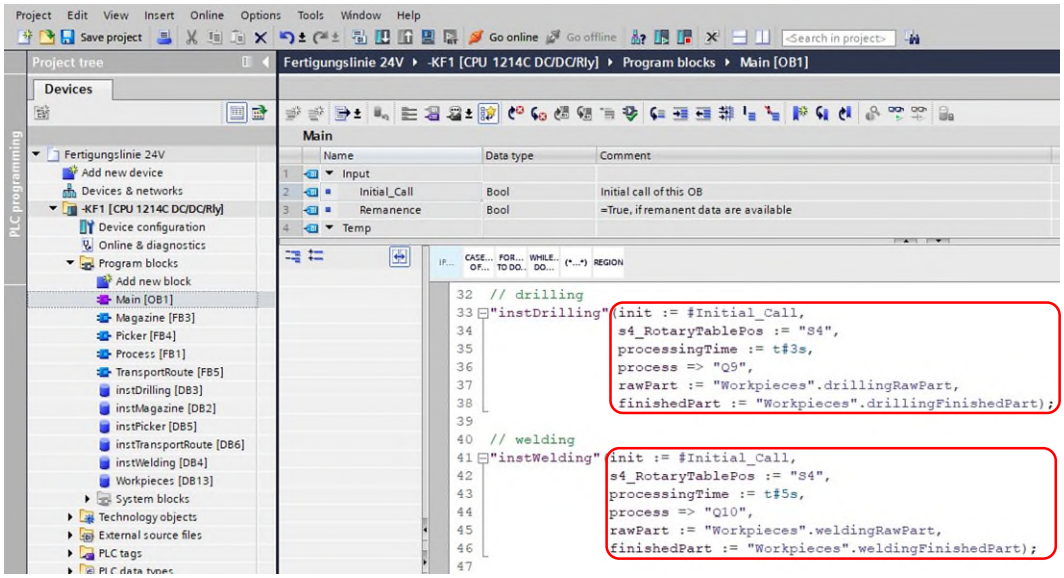
6. Program the time functions according to the CASE structure:

```
// delay Processing time
#instProcessingTime(IN := #process,          // start timer
                    PT := #processingTime); // duration
```

7. Call the function module twice in "MAIN" and create a separate instance for each time:



8. Connect the function block interface with the input and output variables from your variable table, as well as the processing time:

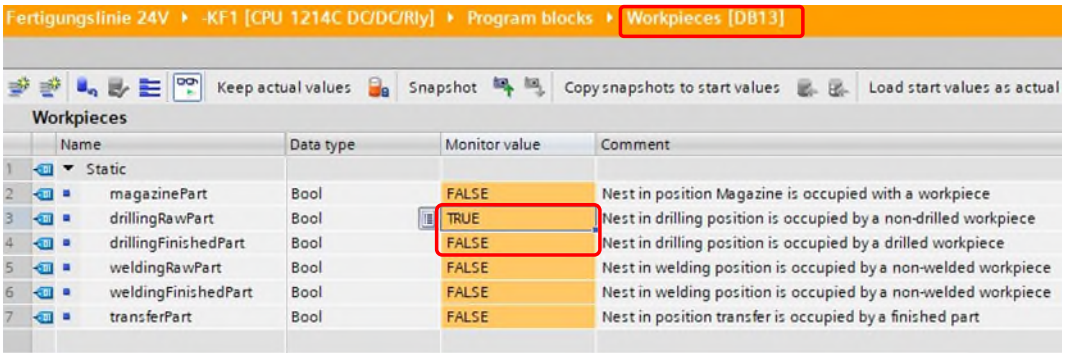


```
// drilling
"instDrilling"(init := #Initial_Call,
               s4_RotaryTablePos := "S4",
               processingTime := t#3s,
               process => "Q5",
               rawPart := "Workpieces".drillingRawPart,
               finishedPart := "Workpieces".drillingFinishedPart);

// welding
"instWelding"(init := #Initial_Call,
              s4_RotaryTablePos := "S4",
              processingTime := t#5s,
              process => "Q10",
              rawPart := "Workpieces".weldingRawPart,
              finishedPart := "Workpieces".weldingFinishedPart);
```

The "Initial_Call" system bit provided by Siemens is used as the initialization request. This is "TRUE" when the MAIN is run through for the first time.

Commission the system in a structured manner with the aid of a commissioning protocol.

In order for another workpiece to be processed in the station, it can be manually set to the corresponding status in the workpiece management.



Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_05_Bearbeiten_SCL.zap17".

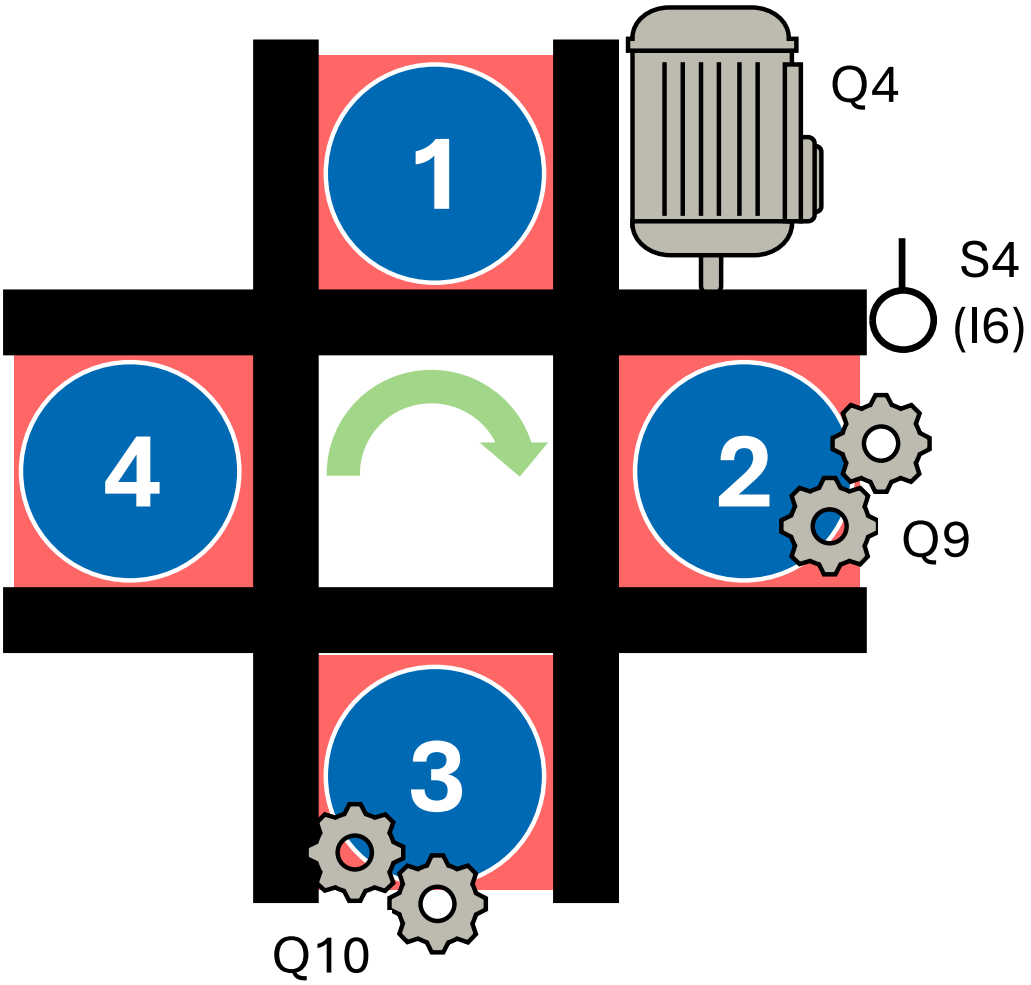
10.13 Exercise: Create function block for rotary table [FBD]

Target:

I can create the automatic program for the rotary table of the production line using the function description.

Task:

Create the PLC program for the automatic sequence so that the function is implemented based on the system description.



Picture 9 System diagram - turntable

Function:

To control the turntable, the following enable conditions must be met at all times:

- Slider Q7 not activated
- Converter (S2) not in the end position on the turntable

The table must be moved until it is back in position (S4).

The table can be moved if,

- a workpiece is in the nest of the magazine (1).
- a finished part is located in one of the two processing stations (2/3).

The table may not be moved as long as

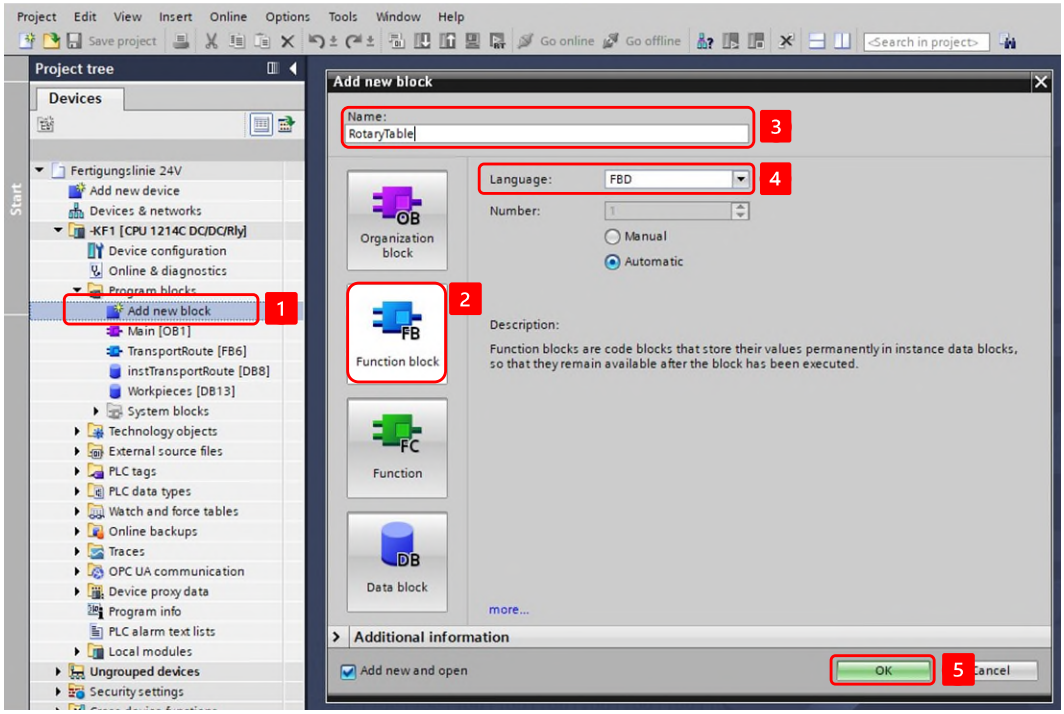
- a workpiece is in the nest of the transfer station (4).
- a raw part is located in one of the two processing stations (2/3).

If the table has been turned to the next position (positive edge S4), the workpiece data must also be rotated by one station:

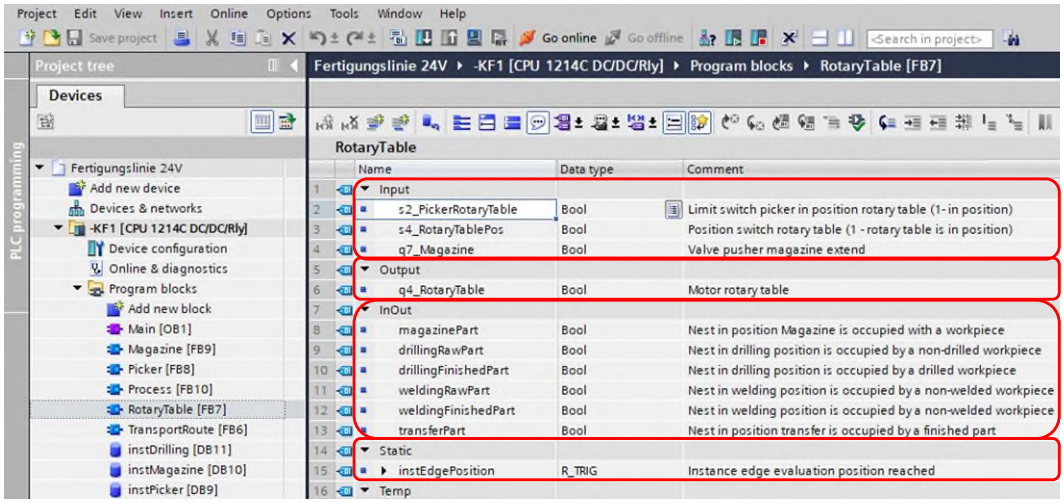
- If a finished part is in the "Welding" station, it must be reset and set as a workpiece in the transfer station.
- If a finished part is in the "Drilling" station, this must be reset and set as a raw part in the "Welding" station.
- If a workpiece is in the "Magazine" station, it must be reset and set as a raw part in the "Drilling" station.

Procedure:

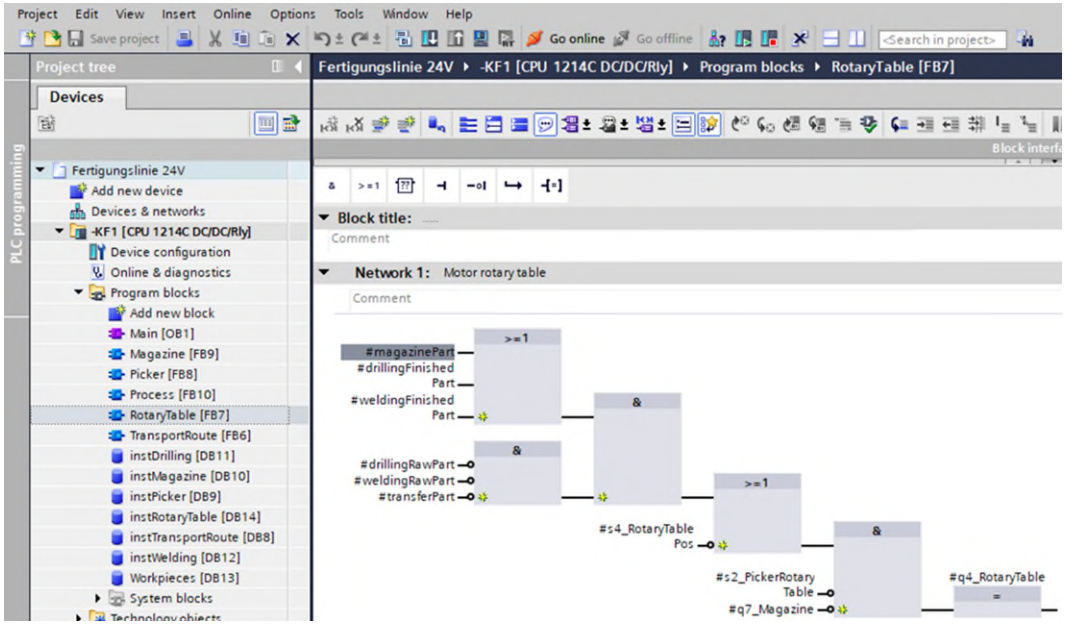
1. Create a new function block, select the desired programming language and assign a meaningful name:



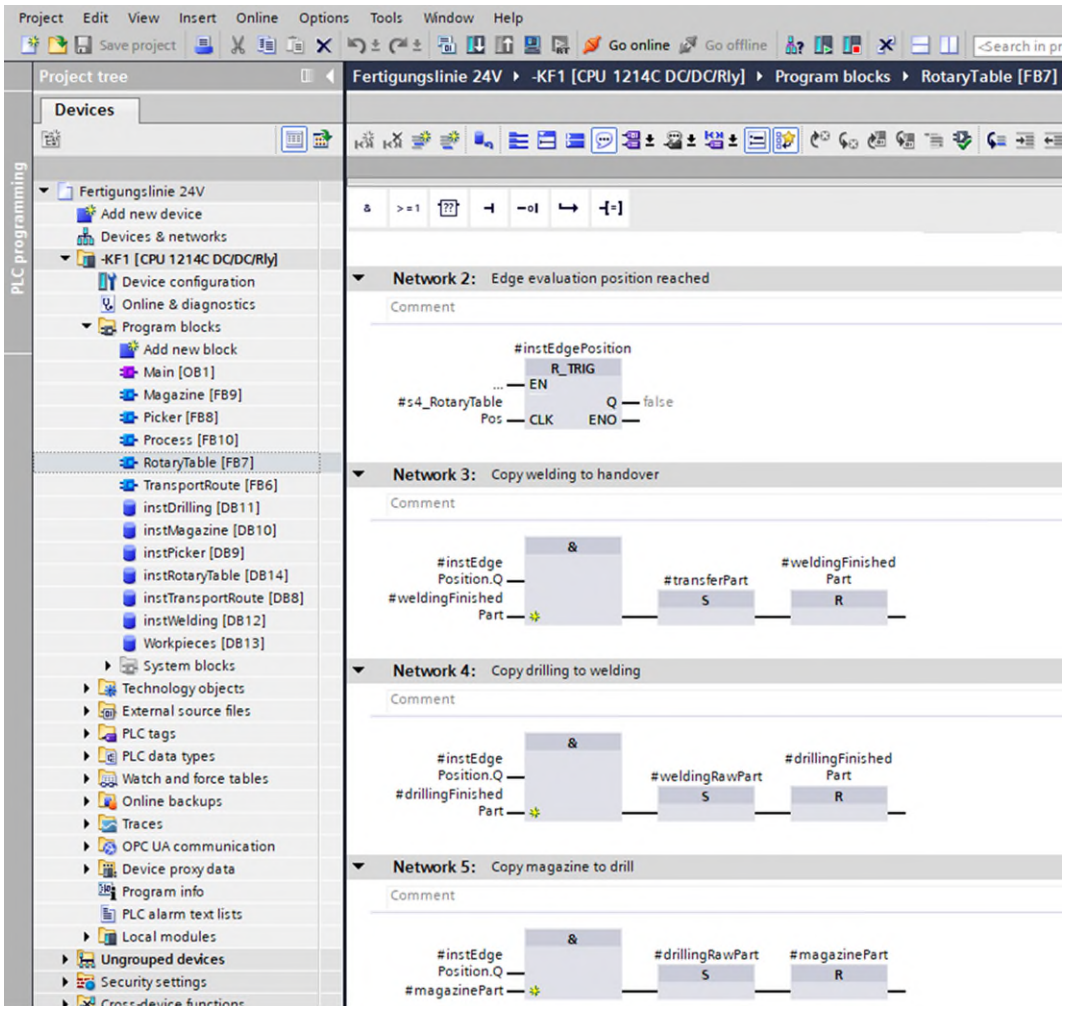
2. Declare variables for the sensors and actuators, the variables for transferring the workpiece information and the instance for edge evaluation in the function block interface:



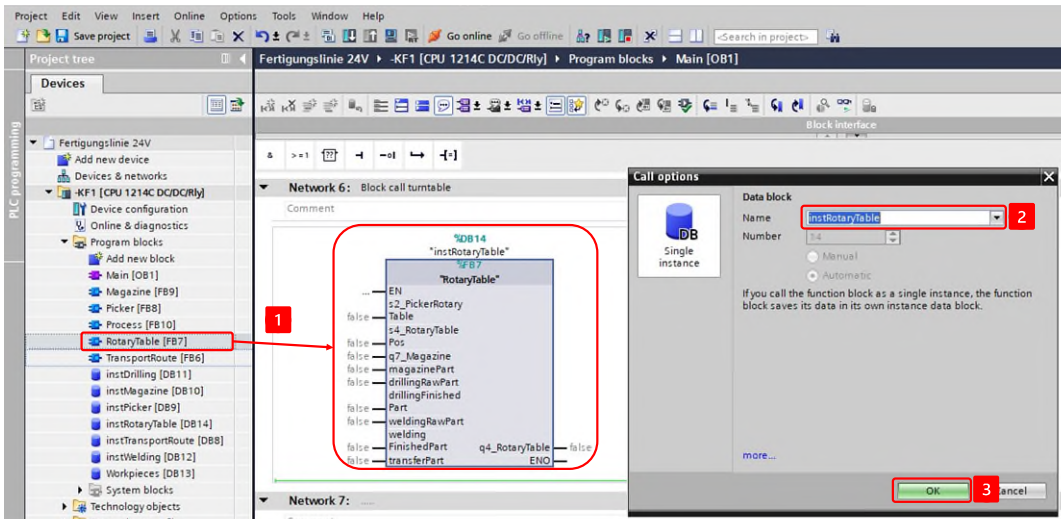
3. Program the control of the turntable (Q4) in the first network:



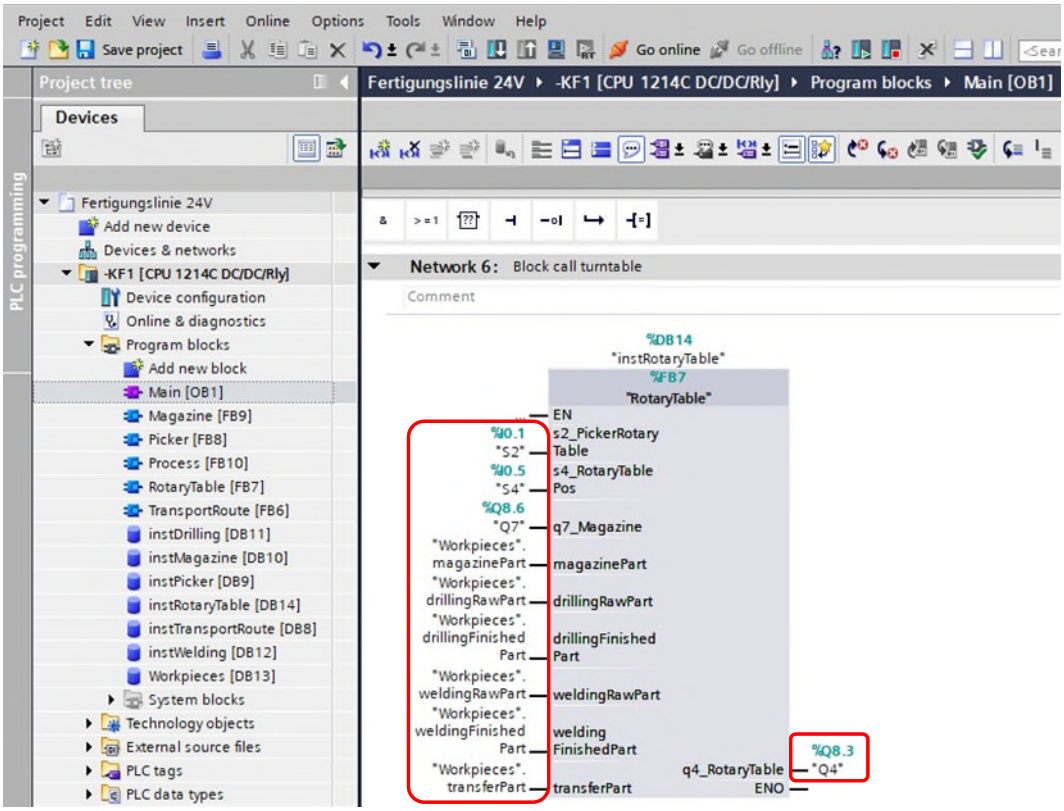
4. Program the edge evaluation (S4) and the rotation of the workpiece data in the following networks:



5. Call up the function module in "MAIN" and create an instance:



6. Connect the module interface:



7. Commission the system in a structured manner with the aid of a commissioning protocol.

Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_06_Drehtisch_FUP.zap17".

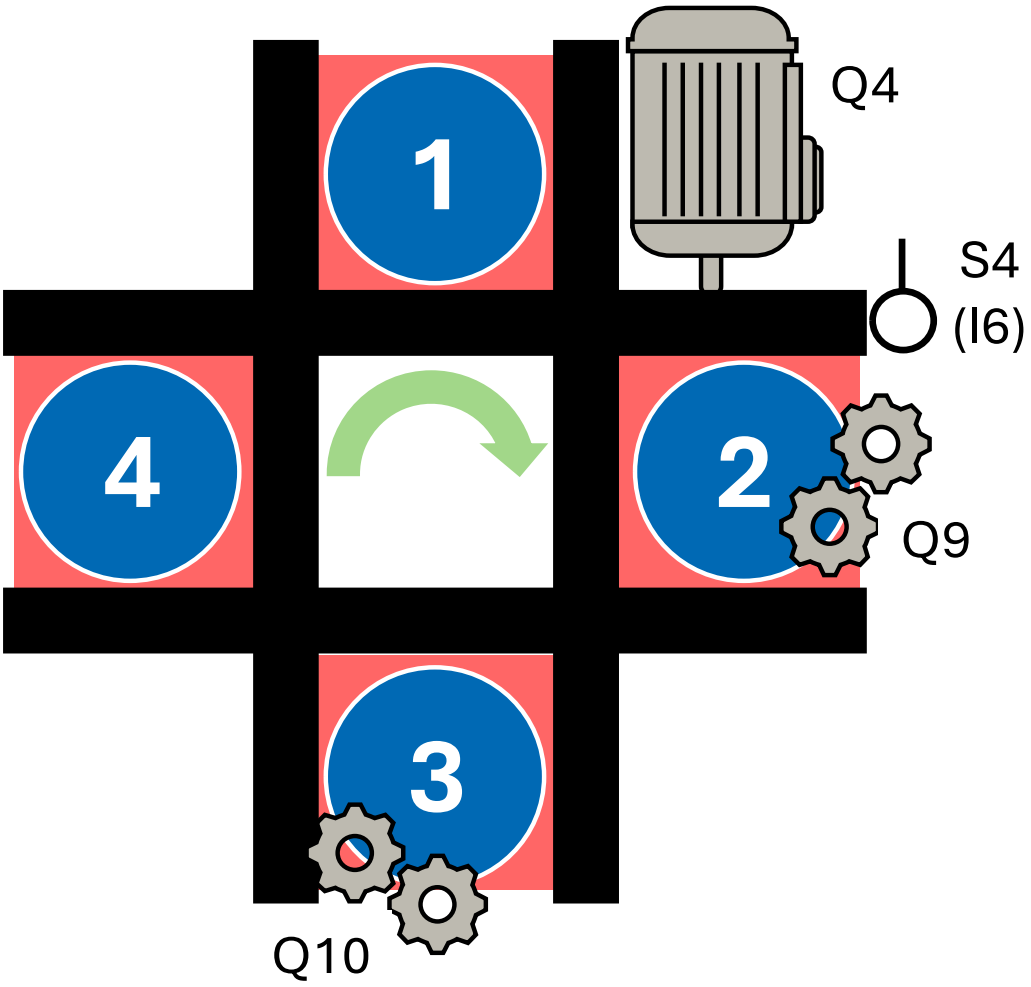
10.14 Exercise: Create function block for rotary table [ST / SCL]

Target:

I can create the automatic program for the rotary table of the production line using the function description.

Task:

Create the PLC program for the automatic sequence so that the function is implemented based on the system description.



Picture 10 System diagram - turntable

Function:

To control the turntable, the following enable conditions must be met at all times:

- Slider Q7 not activated
- Converter (S2) not in the end position on the turntable

The table must be moved until it is back in position (S4).

The table can be moved if,

- a workpiece is in the nest of the magazine (1).
- a finished part is located in one of the two processing stations (2/3).

The table may not be moved as long as

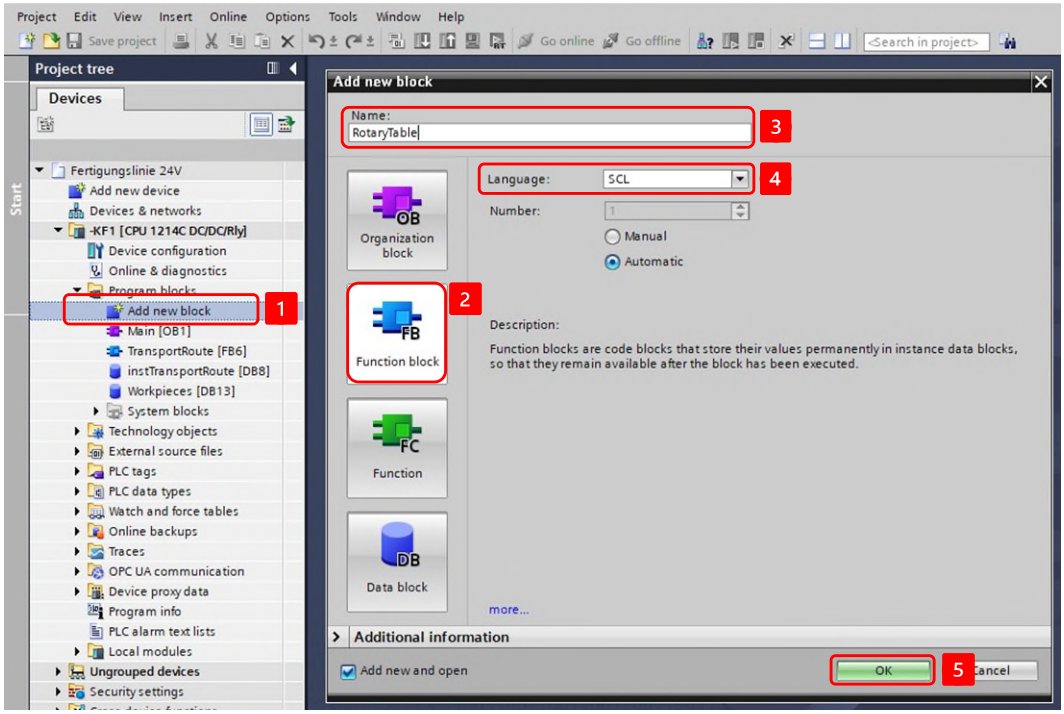
- a workpiece is in the nest of the transfer station (4).
- a raw part is located in one of the two processing stations (2/3).

If the table has been turned to the next position (positive edge S4), the workpiece data must also be rotated by one station:

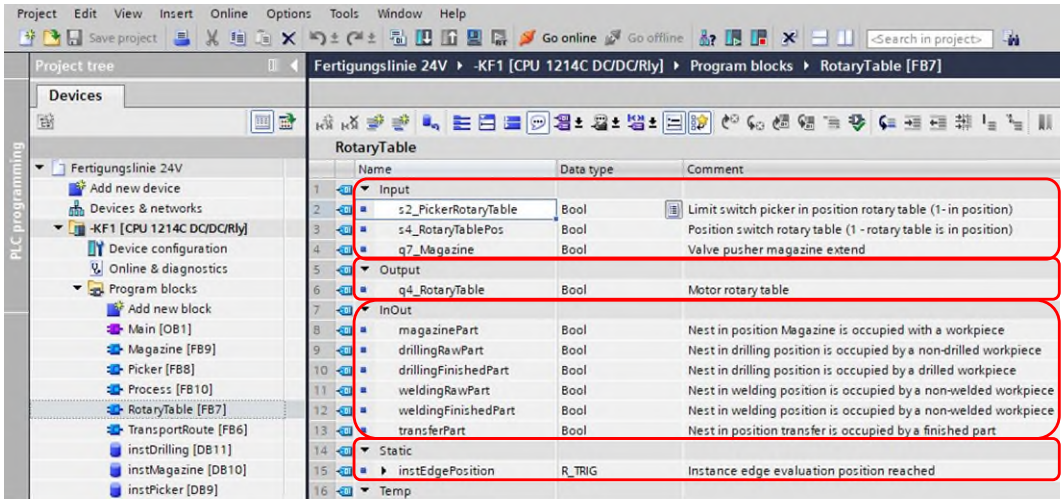
- If a finished part is in the "Welding" station, it must be reset and set as a workpiece in the transfer station.
- If a finished part is in the "Drilling" station, it must be reset and set as a raw part in the "Welding" station.
- If a workpiece is in the "Magazine" station, it must be reset and set as a raw part in the "Drilling" station.

Procedure:

1. Create a new function block, select the desired programming language and assign a meaningful name:



2. Declare variables for the sensors and actuators, the variables for transferring the workpiece information and the instance for edge evaluation in the function block interface:



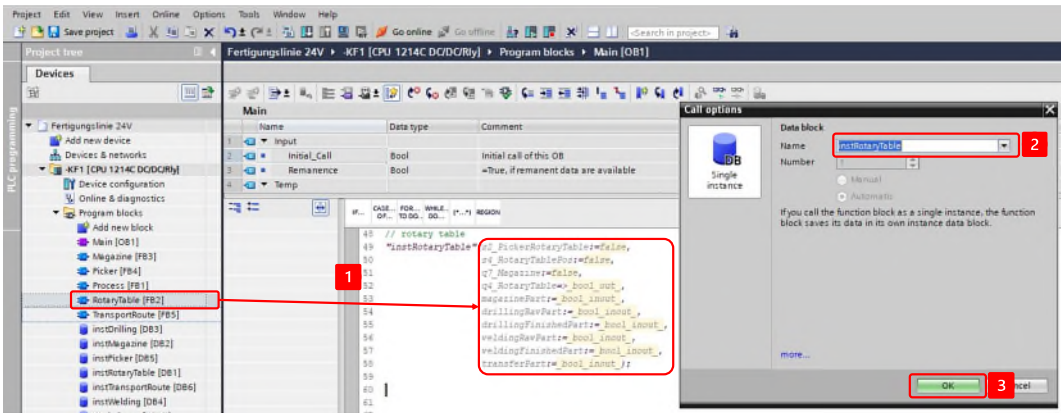
3. Program the control of the turntable (Q4):

```
// control the rotary table motor
#q4_RotaryTable :=
NOT #q7_Magazine AND NOT #s2_PickerRotaryTable                          // permanent interlocks
AND (NOT #s4_RotaryTablePos                                             // rotate until back into position
OR ((NOT #drillingRawPart AND NOT #weldingRawPart AND NOT #transferPart)  // workpiece status locks rotate
AND (#magazinePart OR #drillingFinishedPart OR #weldingFinishedPart)));   // workpiece status requires turning
```

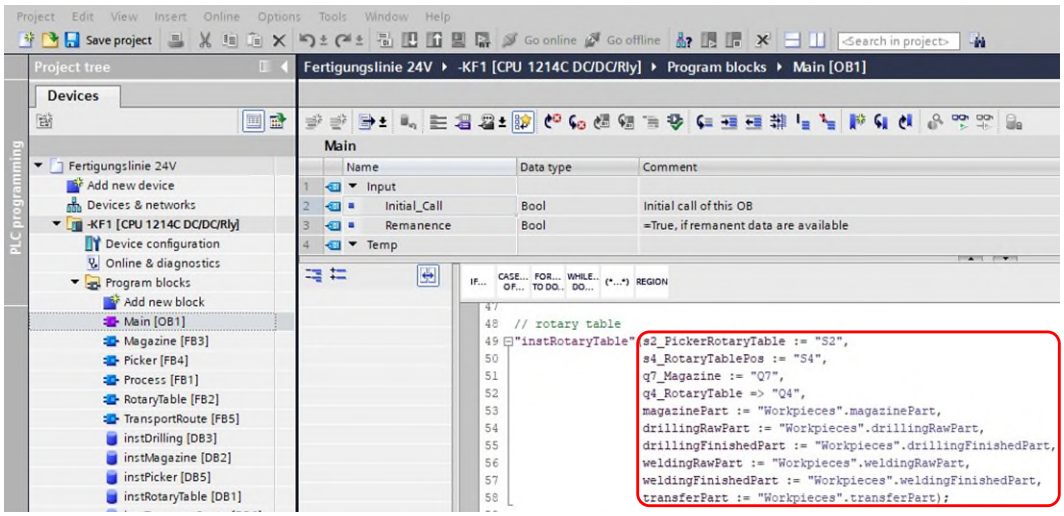
4. Program the edge evaluation (S4) and the rotation of the workpiece data:

```
// edge evaluation position reached
#instEdgePosition(CLK := #s4_RotaryTablePos);


// rotate workpiece data
IF #instEdgePosition.Q THEN
    #drillingRawPart := #magazinePart;
    #weldingRawPart := #drillingFinishedPart;
    #transferPart := #weldingFinishedPart;

    #magazinePart := #drillingFinishedPart := #weldingFinishedPart := FALSE;
END_IF;
```

5. Call up the function module in "MAIN" and create an instance:



6. Connect the module interface:



7. Commission the system in a structured manner with the aid of a commissioning protocol.

Solution

Solution:

The solution can be found in the TIA Portal project "Fertigungslinie_06_Drehtisch_SCL.zap17".