The Drive & Control Company

Rexroth
Bosch Group

IndraWorks 14VRS
IndraLogic 2G

SFC with Operation Modes

Application Description
R911344490

Edition 02

Title
IndraWorks 14VRS
IndraLogic 2G
SFC with Operation Modes

Type of Documentation
Application Description

Document Typecode
DOK-IL*2G*-SFC*OPM*V14-AP02-EN-P

Internal File Reference
RS-60ebeb9937bf81160a6846a50128eb56-2-en-US-8

Change Record

Edition	Release Date	Note
01	2014-07	First edition
02	2015-11	Update of the "Variables supported for the ProVi SFC diagnostics"

Editorial Department
Engineering Automation Systems PLC Programming Interface VG (KaWa/MePe)

Table of Contents

Table of Contents

1 About this documentation

1.1 Validity of the documentation

Target group

In the following illustration, the framed activities, product phases and target groups refer to this documentation.

Example: In the product phase "Engineering", the target group "programmer" can execute the activities "parameterize", "program" and "configure" using this documentation.

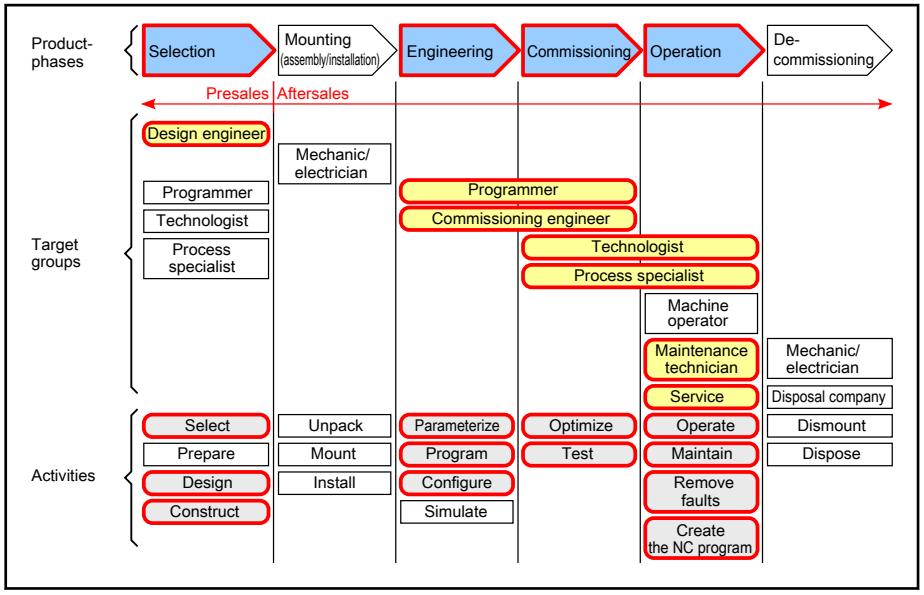


Fig. 1-1: Assigning this documentation to the target groups, product phases and target group activities

1.2 Required and supplementing documentation IndraWorks

1.2.1 Engineering

Documentation titles with type codes and part numbers

Rexroth IndraWorks 14VRS Software Installation DOK-IWORKS-SOFTINS*V14-CORS-EN-P, R911344286 This documentation describes the IndraWorks installation.
Rexroth IndraWorks 14VRS Engineering DOK-IWORKS-ENGINEE*V14-APRS-EN-P, R911343566 This documentation describes the application of IndraWorks in which the Rexroth Engineering tools are integrated. It includes instructions on how to work with IndraWorks and how to operate the oscilloscope function.
Rexroth IndraWorks 14VRS IndraLogic 2G PLC Programming System DOK-IWORKS-IL2GPRO*V14-APRS-EN-P, R911343571 This documentation describes the PLC programming tool IndraLogic 2G and its use. It includes the basic use, first steps, visualization, menu items and editors.
Rexroth IndraWorks 14VRS Basic Libraries IndraLogic 2G DOK-IL*2G*-BASLIB**V14-LIRS-EN-P, R911343920 This documentation describes the system-comprehensive PLC libraries.

About this documentation

Rexroth IndraWorks 14VRS WinStudio
DOK-IWORKS-WINSTUD*V14-APRS-EN-P, R911341585
This "User Manual and Technical Reference Book" facilitates working with the "Rexroth WinStudio"™ software for optimal results. This document provides technical information and step-by-step instructions to create web-enabled HMI/SCADA programs.
Rexroth IndraWorks 13VRS CamBuilder
DOK-IWORKS-CAMBUIL*V13-APRS-EN-P, R911336291
This documentation describes the basic principles and operation of the CamBuilder, the cam editing tool.
Rexroth IndraWorks 14VRS Field Buses
DOK-IWORKS-FB******V14-APRS-EN-P, R911341485
This documentation describes the field bus and local periphery connections supported by the IndraLogic XLC, IndraMotion MLC and IndraMotion MTX systems. The focus of this documentation is on the configuration, parameterization, commissioning and the diagnostics of the different periphery connections.
Rexroth IndraWorks IndraMotion Service Tool
DOK-IWORKS-IMST*******-APRS-DE-P, R911341383
This documentation describes the IndraMotion Service Tool (IMST). IMST is a web-based diagnostic tool used to access a control system via an Ethernet high-speed connection. The IMST allows OEMs, end users and service engineers to access and remotely diagnose a system. The PC has to use at least Internet Explorer 8, Firefox 3.5 or a higher version. The following control variants are supported: • IndraMotion MLC L25/L45/L65 • IndraLogic XLC L25/L45/L65/VEP
Rexroth IndraDrive Service Tools IMST/IDST
DOK-IM*MLD-IMSTIDSTV13-RERS-EN-P, R911342652
This documentation describes the IndraMotion Service Tools (IMST and IDST). It is a web-based diagnostic tool used to access a drive system via a high-speed Ethernet connection. Using these service tools OEMs, end users and service engineers can access and remotely diagnose a system from anywhere.
Rexroth IndraWorks 14VRS SafeLogic First Steps
DOK-IWORKS-SL*STEP*V14-CORS-EN-P, R911341520
This documentation provides information to facilitate an easy start in the safety control engineering SafeLogic. By using project examples, information about the installation, configuration, commissioning, troubleshooting and diagnostics is provided to the user.
Rexroth IndraWorks 12VRS FDT Container
DOK-IWORKS-FDT*CON*V12-APRS-EN-P, R911334398
This documentation describes the IndraWorks FDT Container functionality. It includes the activation of the functionality in the project and working with DTMs.
Rexroth IndraWorks 14VRS Field Buses Libraries
DOK-IWORKS-FB*LIB**V14-LIRS-EN-P, R911343575
This documentation describes field bus libraries for the IndraLogic XLC, IndraMotion MLC and IndraMotion MTX systems.
Rexroth IndraWorks Remote Condition Monitoring RMB_TechRCM 14VRS
DOK-IWORKS-TEC*RCM*V14-LIRS-EN-E, R911343671
This documentation describes the individual functions of the "Remote Condition Monitoring" systems as well as the most important use cases for a quick status assessment of the connected machines/systems.
Rexroth IndraWorks 14VRS SafeLogic Project Configuration
DOK-IWORKS-SL**PRJ*V14-APRS-EN-P, R911341694
This documentation describes the creation and programming of SafeLogic projects in IndraWorks Engineering.

Tab. 1-1: IndraWorks documentation overview - Engineering

1.2.2 Engineering & visualization

Documentation titles with type codes and part numbers

Rexroth IndraWorks 14VRS Energy Efficiency Management
DOK-IWORKS-4EE*****V14-APRS-EN-P, R911339229
This document describes the use of the Energy Efficiency Management function within the MLC, XLC and MTX systems. Using example applications, this documentation provides information about the project planning, configuration and visualization of the system.

Tab. 1-2: IndraWorks documentation overview - Engineering & visualization

1.2.3 Visualization

Documentation titles with type codes and part numbers

Rexroth IndraWorks 14VRS HMI
DOK-IWORKS-HMI*****V14-APRS-EN-P, R911343569
This documentation describes the HMI operating interface IndraWorks Operation, its operation and functions as well as the configuration in IndraWorks Engineering.

Tab. 1-3: IndraWorks documentation overview - Visualization

1.3 Use of the safety instructions

1.3.1 Structure of the safety instructions

The safety instructions are structured as follows:

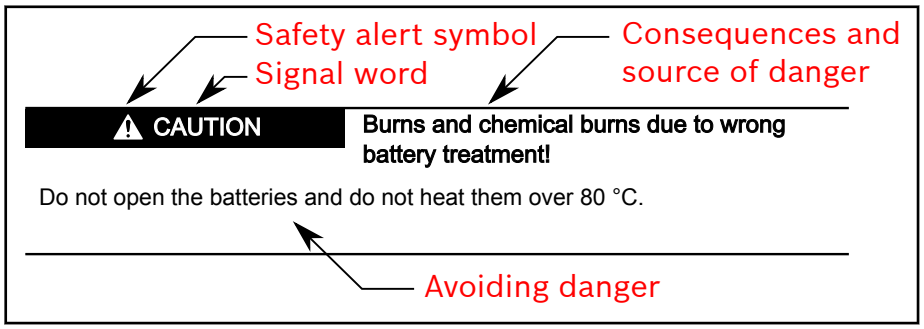


Fig. 1-2: Structure of the safety instructions

1.3.2 Explaining signal words and safety alert symbol

The safety instructions in this documentation contain specific signal words (danger, warning, caution, notice) and, if necessary, a safety alert symbol (according to ANSI Z535.6-2006).

The signal word is used to draw attention to the safety instruction and also provides information on the severity of the hazard.

The safety alert symbol (a triangle with an exclamation point), which precedes the signal words danger,warning and caution is used to alert the reader to personal injury hazards.

⚠ DANGER

In the event of non-compliance with this safety instruction, death or serious injury **will** occur.

About this documentation

⚠ WARNING

In the event of non-compliance with this safety instruction, death or serious injury **will** occur.

⚠ CAUTION

In the event of non-compliance with this safety instruction, minor or moderate injury can occur.

NOTICE

In the event of non-compliance with this safety instruction, material damage can occur.

1.3.3 Symbols used

Hints are represented as follows:

This is an information.

Tips are represented as follows:

This is a tip.

1.3.4 Signal graphic explanation on the device

Prior to the installation and commissioning of the device, refer to the device documentation.

1.4 Names an abbreviations

Term	Explanation
CANopen	Field bus
DeviceNet	Field bus
Ethernet	Communication interface
IWE	IndraWorks Engineering
IWO	IndraWorks Operation
NC	Numerical Control
OEM	Original Equipment Manufacturer
Profibus DP	Field bus
Sercos	Sercos (Serial Realtime Communication System) interface is a world-wide standardized interface for the communication between controls and drives

Tab. 1-4: Names and abbreviations used

1.5 Customer feedback

Customer requests, comments or suggestions for improvement are of great importance to us. Please email your feedback on the documentations to Feedback.Documentation@boschrexroth.de. Directly insert comments in the electronic PDF document and send the PDF file to Bosch Rexroth.

2 SFC with operation modes

2.1 General information

IndraLogic provides IEC 61131 SFCs with automatic operation mode and diagnostics.

Operation modes

The SFC knows and manages operation modes such as automatic mode, manual mode, jog mode, semi-automatic mode and homing.

The operation modes of the SFC can be controlled using input and output signals of the SFC.

The engineer does not have to take care of the programming of the operation modes. He only has to apply the respective signals at the inputs and outputs of the SFC.

Diagnostics

The SFC automatically supports different diagnostics. These include SFC status, time errors, monitor errors, missing starting condition, missing home position as well as a feasibility display.

The engineer only has to assign the relevant conditions.

Different SFC types

If available, different SFC types can be used at the same time. The different SFCs can differ in their response as well as in their input and output signals.

The standard IndraLogic SFC, with or without IEC steps, can still be used.

- All SFCs with diagnostics have to be programmed in one task
- The maximum number of steps per POU (**P**rogram **O**rganization **U**nit) is 65,000 or it is limited by the available memory in the control

2.2 What is an SFC with operation modes?

In case of an SFC with operation modes, proceeding the steps is controlled depending on the operation mode set.

The operation mode of an SFC can be controlled via the SFC input signals:

Automatic	As soon as the transition condition is fulfilled during error-free operation, the SFC switches to the next step, analog to the standard IEC SFC
Manual mode	Individual steps and/or actions can be activated specifically. The transition conditions are analyzed, but there is no block transition
Jog mode	The steps are processed until the transition condition is fulfilled. A new starting signal is required for transition to the next step
Semi-automatic	For normal transitions, the SFC advances to the next step when the transition condition is fulfilled. If the transition is defined as a stop point for the semi-automatic mode, the system does not automatically advance to the next steps. It requires a new starting signal

Tab. 2-1: Operation modes

SFC diagnostics

For an SFC with operation modes, there is a diagnostics automatically deriving from the programmed logic.

The response of the SFC to corresponding states occurs automatically. The messages are displayed in IndraWorks.

SFC with operation modes

Module An SFC can be assigned to a certain module. The messages are set for this module. Up to 99 modules are possible.

The module serves to logically assign an SFC to a specified machine component.

Example: The PLC controls several logically separated machine components. The diagnostic messages of one component should be separated from those of the other component.

Criteria analysis Represents the contacts which have caused the message or the error. This determination is executed automatically, considers only Boolean logic and is generated without further programming effort.

Diagnostic messages *There are eight different types of diagnostic messages in the SFC:*

1. Active steps
2. Home position reached
3. Homing
4. Missing home position with criteria analysis
5. Monitor error with criteria analysis
6. Time error with criteria analysis
7. Missing starting condition with criteria analysis
8. Wait for transition

Diagnostic message	Description
Active steps	The active steps are reported as status message on the IndraWorks diagnostic screen
Home position reached	The SFC reports that it is in the Init step and the machine/SFC is in home position
Homing	The SFC reports that the home position branch is currently being processed
Monitor error	For this diagnostic type, the user has to program. The current result (CR) can be monitored by individual networks. If the CR has an incorrect state, the SFC automatically switches to the error state. **Example:** An error is triggered if the CR is to be monitored for FALSE and if the CR of the network is FALSE during processing of the action. If the CR is used to set a coil with storing properties (set coil) monitoring of the RLO continues after setting. An error is triggered in spite of the set coil if the CR of the network turns to FALSE during processing. The networks to be monitored have to be defined manually. For this diagnostic type, the criteria analysis is also available. The criteria analysis reports which contact in the network triggered the error. The criteria analysis calculates only Boolean variables that are directly related to the CR
Time error	For the diagnostic type "TimeError", the user has to program. The duration of the steps being active are monitored. A maximum time can be set for every step. If the step is active for a longer period, the SFC automatically switches to the error state. Monitoring the maximum step time applies to all operation modes except the manual mode. In the jog mode, monitoring the step time stops when the transition condition is fulfilled. The criteria analysis is available for this diagnostic type. A time error can only have one cause: The transition condition was not achieved, since a required signal has not come in. In this case, the missing transition condition is displayed with the responsible variables

Diagnostic message	Description
Start condition	The diagnostic type "starting condition" requires that the engineer defines the parameters for the starting condition of the SFC. This diagnostic type applies to the operation modes: Automatic, semi-automatic and jog mode. Further analysis is then automatically performed by the SFC. For error-free processing, the starting conditions should to be assigned in the action "aUserIn" in the step "Init". If the starting condition is missing, the SFC stops immediately and switches to the error state. If the SFC is stopped and starting conditions are missing, this is displayed as message of the type "starting condition". If the user now tries to start the SFC, it switches immediately to the error state and the message is output as an error. The criteria analysis is available for this diagnostic type and shows the responsible variables
Setup diagnostics	The setup diagnostics corresponds to the diagnostic type "starting condition" in manual mode. The criteria analysis is available for this diagnostic type and shows the responsible variables
Home position	The diagnostic type "HomePosition" requires that the engineer defines the conditions for the home position of the SFC. Further analysis is then automatically performed by the SFC. For error-free processing, the conditions should be assigned in the action "aUserIn" in the step "Init". If the SFC stopped in the Init step and home position conditions are missing, this is displayed as a message of the type "starting condition". If the user now tries to exit the Init step, the SFC immediately switches to the error state and the message is output as an error. No error is triggered if the Init step is branched to the home position branch. The criteria analysis is available for this diagnostic type and shows the responsible variables
Wait for transition	This diagnostic type is only available for the jog and semi-automatic operation modes. There is no criteria analysis available for this message type The diagnostic message describes the sequence state in which the SFC waits for the respective starting signal (Enable or Start) with the transition condition fulfilled

Tab. 2-2: Diagnostic messages

The starting condition has a higher priority than the home position, i.e. as long as a starting condition message is pending, the home position conditions are not analyzed.

Cascaded machine states

The parameters for the starting condition and home position of several SFCs can be cascaded.

Conditions that apply to all SFCs of the control, only have to be defined once for the control (still without criteria analysis).

Conditions that apply to all SFCs of a module have to be defined only once for this module (still without criteria analysis).

Conditions that only apply to one SFC, are only defined for this SFC (with criteria analysis).

3 Sequence description of the SFC

3.1 General information

IEC 61131

The basis for an SFC with operation modes is the IEC 61131 SFC. However, since the IEC 61131 standard does not know any operation modes and diagnostics, the SFC is extended by further elements.

Basic programming still complies with the IEC 61131 standard.

Steps/transitions

The sequence of an SFC is programmed using steps and transitions. The steps execute the corresponding code and the transitions define the switching point to the next step.

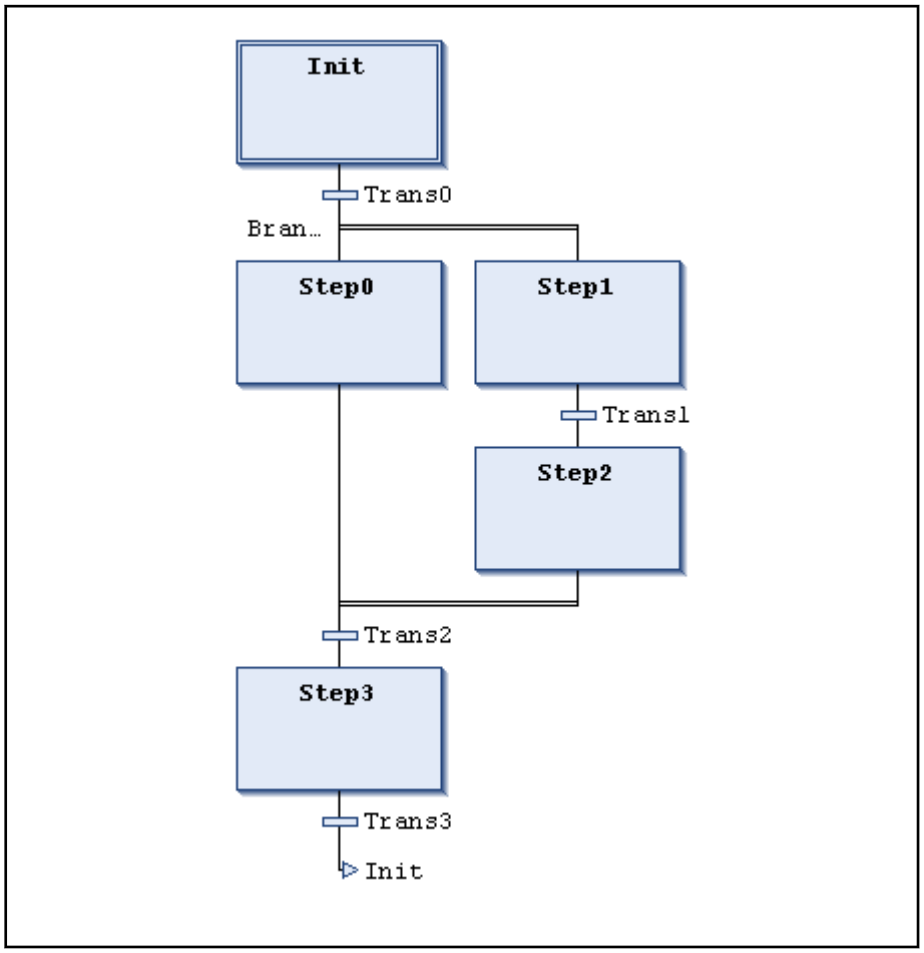


Fig. 3-1: Sequence of an SFC with steps and transitions

Transition condition

A transition switches to the next step if the previous step is active and the condition for the transition is met. For parallel branches, the last previous steps of the relevant branches and their transitions have to be met, see "Parallel branching" on page 15.

The step transition conditions may only be implemented in ST (Structured Text). Using other programming languages such as LD, FUP (FBs) and IL cause errors when creating diagnostic data. The error includes the note that the code for the transition could not be determined.

Sequence description of the SFC

Step – Minimum time

If a step is programmed with a minimum time, the fulfilled transition only switches to the next step after the minimum time for this step has elapsed, i.e. the step is active for at least this period.

The minimum step time can be used to process a step with action/s over a certain period of time even if the transition condition is met.

Example: A predetermined number of products is supplied for processing via a flap mechanism. Even when the flap is open (transition met) there has to be a certain waiting time (minimum time) before the flap closes again. This ensures that no products remain in the flap mechanism.

Transition delay

If the conditions for the advancing a transition with a transition delay are met, the system waits until the programmed delay time has elapsed before advancing to the next step.

When using transition delay, no assignment of "ActionQ" (or an AND operation with "ActionQ") may not be carried out on "ActionTrans".

A fulfilled transition results in post-processing the "ActionQ" = FALSE and thus also the "ActionTrans" = FALSE. The action returns to processing before the transition delay time has elapsed. It is not possible to exit the step anymore.

The transition delay can be used to debounce input signals, e.g. a mechanical limit switch.

Alternative branching

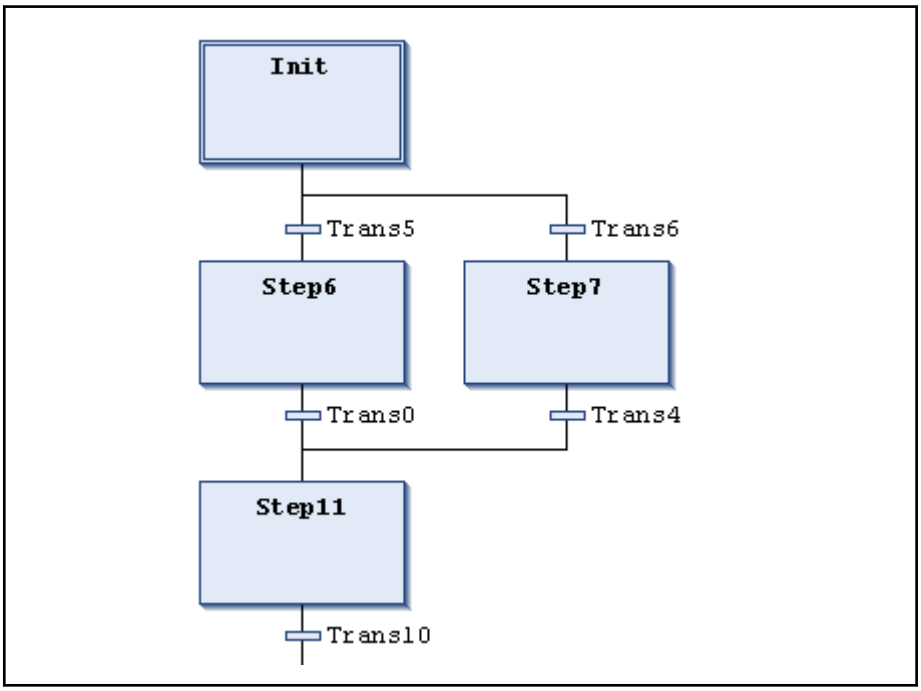


Fig. 3-2: Alternative branching

For alternative branching, it is possible to switch to different steps.

The following applies for this example: If the condition for "Trans5" is met, step "Step6" is activated next. If the condition for "Trans6" is met instead, step "Step7" is activated next.

If both transitions are met, it is switched in the first branch from the left side. In this example "Step6".

Parallel branching

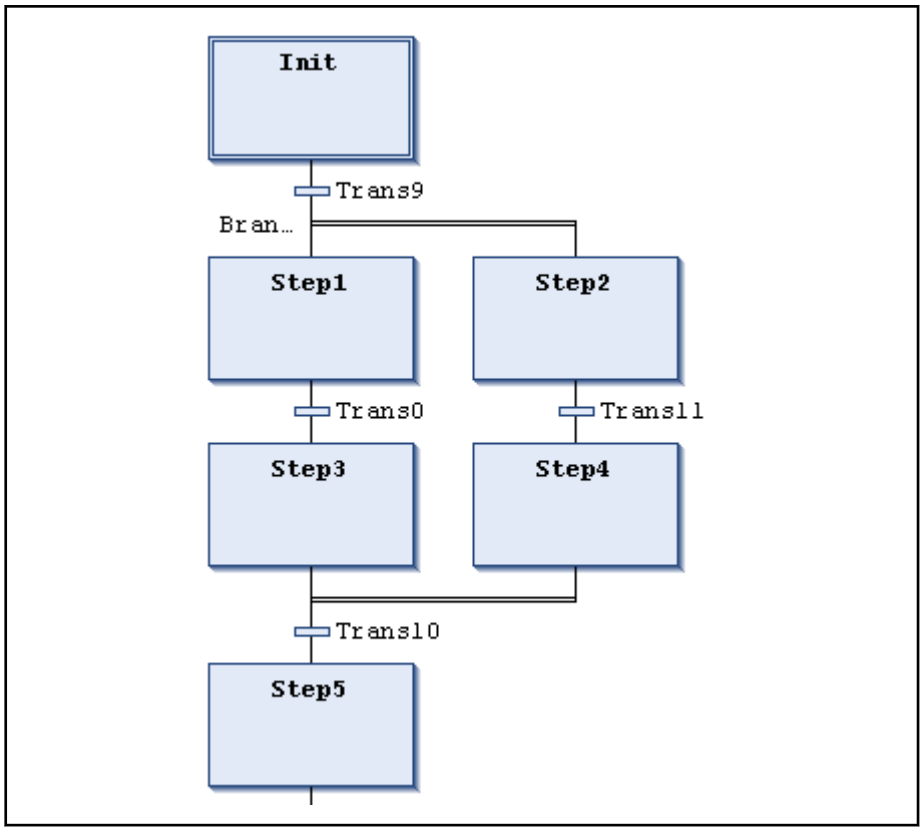


Fig. 3-3: Parallel branching

For a parallel branching, the system is advanced in several steps. This means for this example: If "Trans9" is met, the steps "Step1" and "Step2" are activated.

Transition "Trans10" only advances if the steps "Step3" and "Step4" are active.

Action processing

Actions are attached to the steps. In these actions, the code executed for an active step can be programmed in the programming languages FBS, LD, IL and ST.

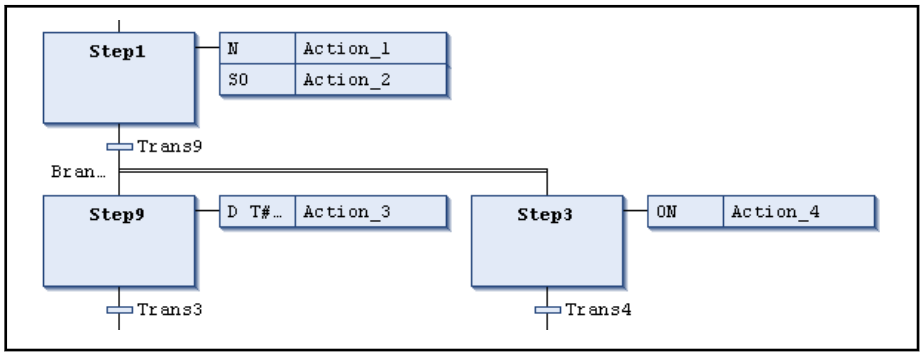


Fig. 3-4: Actions at a step

An action can be used at any number of steps.

If a step becomes active, all actions associated with this step are processed according to the qualifier used.

If several steps are active, all actions are processed at the active steps.

Sequence description of the SFC

- It is not possible to process Boolean flags instead of actions when using SFC with diagnostics!
- If a certain action is processed at several steps which are simultaneously active, the status of the action processing is not clearly defined. Such constellations are to be avoided
- It is not permitted to call an action within a second action for a POU of the type "SFC", e.g. having the call of an action associated to a step and at the same time in the "aUserIn"

Action Sequence

The actions are processed in alphabetical order. In the example in fig. 3-4 "Actions at a step" on page 15, this means that, if "Step9" and "Step3" are enabled, first "Action_3" and then "Action_4" is processed.

aUserIn

The "aUserIn" action is a specific action. The "aUserIn" should also contain the "StartCondition" as well as the "HomePosition" assignments and called in the Init step.

If an action with this name is used, it is always processed cyclically. That means this action is also processed even if the Init step is no longer active. This applies to all operation modes.

Post-processing of actions

If an action is no longer to be processed, it is nevertheless processed for one more cycle. This post-processing disables outputs activated during the action or prepare an function block for a new call. Post-processing can be identified by the flags "ActionA" and "ActionQ".

ActionA, ActionQ

State	ActionA	ActionQ
Processing	TRUE	TRUE
Post-processing	TRUE	FALSE
No processing	FALSE	FALSE

Tab. 3-1: "ActionA" and "ActionQ" flags

The flags "ActionA" and "ActionQ" are global flags provided by the ILDSFC of the SFC.

The status of these flags is only defined within the action processing.

During action processing, the two flags are provided with a matching state for this action. The state may differ when processing the next action.

Since the flags "ActionA" and "ActionQ" are global flags, they cannot be observed network-/action-dependent while they are logged in. The displayed state of these two global flags is always "TRUE".

The flag "ActionA" seems to have no function, since it is always TRUE while processing the action and it has no defined state beyond processing. However, under the specific circumstances described below, an action is processed cyclically irrespective of the related step. In this case, flag "ActionA" is also FALSE during cyclic action processing.

The action "aUserIn" is processed cyclically.

However, the characteristic feature is the fact that flags "ActionA" and "ActionQ" are always FALSE.

Post-processing examples

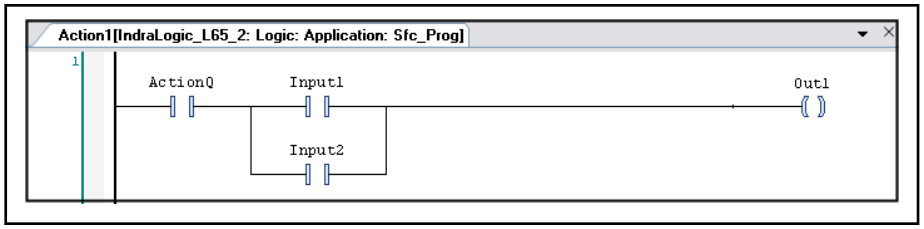


Fig. 3-5: Disabling outputs during post-processing

The output is disabled while processing the action depending on the inputs. If the action without post-processing is now no longer processed, the output is still TRUE. Post-processing, however, resets the output to FALSE due to the AND operation of the "ActionQ" flag. For an example, refer to "Signal-time diagram: Post-processing with N-qualifier" on page 21.

Action qualifier

The action qualifier used determines when an action is processed at an active step. These qualifiers are indicated during use of actions and thus control the enabling of actions. Additional values can be specified for different qualifiers. If an action is disabled, it is still post-processed for one action. unless the action is described differently in the following table.

Qualifier	Description
N	Non-saving. This action is processed as long as the step is active. In the cycle in which the step is disabled, the action is disabled as well
S	Set saving. The action is processed from the point in time when the step becomes active, even if the step is then no longer active. The action can be disabled with the R-qualifier
R	Dominant reset. If an action was activated with the qualifier S, SD, DS, or SL, it is disabled when the step is active
L	Non-saving, time-limited. For this qualifier, a time period is indicated as additional information. The action is processed as long as the step is active. If the step requires longer than the time specified, the action is disabled after the time has elapsed
D	Non-saving, with time delay. For this qualifier, a time period is indicated as additional information, e.g. "D T#2s500ms". Time starts running when the step is activated. After time-out and if the step is still active, the action is processed. If this step is disables, the action is also disabled again
P	Pulse. If the step is active, the action is processed for two cycles and then disabled again. Using the flag "ActionQ" ensures that there is always a positive edge or a pulse to be evaluated (see "Signal-time diagram qualifiers P/P0/P1" on page 26)

Sequence description of the SFC

Qualifier	Description
P0	Post-processing upon disabling the step. If the step is disabled, the action is post-processed for one cycle. A processing does not take place (refer to "Signal-time diagram qualifiers P/P0/P1" on page 26)
P1	Post-processing upon enabling the step. When the step becomes active, the action is post-processed for one cycle. A processing does not take place (refer to "Signal-time diagram qualifiers P/P0/P1" on page 26)
SD	Saving, with time delay. For this qualifier, a time period is indicated as additional information, e.g. "SD T#2s500ms". Time starts running when the step is activated. The action is only processed after time-out irrespective of the step being active or not being active. It is disabled with the R-qualifier
DS	With time delay, saving. For this qualifier, a time period is indicated as additional information, e.g. "DS T#2s500ms". Time starts running when the step is activated. The action is only processed after time-out and if the step is still active. The action is disabled with the R-qualifier
SL	Saving, with time limit. For this qualifier, a time period is indicated as additional information, e.g. "SL T#2s500ms". The action is processed as soon as this step is activated. It is even processed if the step is disabled. The action is disabled again after time-out. Even when the time has elapsed, saving with the R-qualifier has to be reset
ON	Cyclic processing, enabling. The action receives two additional states: "Enable" and "Disable". If the step is enabled, the action is processed in the "Enable" state (refer to "Signal-time diagram ON-/OFF-qualifiers" on page 28 and "ON- and OFF-qualifier" on page 19)
OFF	Cyclic processing, disable. The action receives two additional states: "Enable" and "Disable". If the step is enabled, the action is processed in the "Disable" status (refer to "Signal-time diagram ON-/OFF-qualifiers" on page 28 and "ON- and OFF-qualifier" on page 19)

Qualifier	Description
GO	Cyclic processing, value transfer. For this qualifier, a REAL value is indicated as additional information. This action receives an additional "transfer value" state. If the step is enabled, the action is processed in the "Transmission of value" state (refer to "Signal-time diagramGO-qualifiers" on page 27)
GOdi	Cyclic processing, value transfer. For this qualifier, a DINT value is indicated as additional information. This action receives an additional "transfer value" state. If the step is enabled, the action is processed in the "Transmission of value" state (refer to "Signal-time diagramGO-qualifiers" on page 27)

Tab. 3-2: Describing the action qualifiers

Cyclic action processing

Actions used with the qualifiers ON, OFF, GO and GOdi have a different behavior than other actions.

Actions used with ON, OFF, GO and GOdi cannot be used with any other qualifier.

Since the flags "ActionON" and "ActionOFF" and "Action GO" are global flags, they **cannot** be observed network-/action-dependent while logged in. The state displayed of these global flags is always "TRUE".

These actions are **always** processed cyclically irrespective of the state the SFC or the related steps are in.

Regarding the processing and post-processing, these actions respond like an action with N-qualifier, except that these actions are still processed when an N-action is not processed. "ActionA" and "ActionQ" are then FALSE.

In this case, processing and post-processing are not relevant. The additional states are the decisive factor.

ON- and OFF-qualifier

The "ON"- and "OFF"-qualifiers provide two additional states to the action in addition to the state "cyclic processing": "Enable" and "Disable". These states can be identified by the global flags "ActionON" and "ActionOFF".

As long as the step with the ON-qualifier is active, the action is processed in the "Enable" state. During this processing, the "ActionON" flag is TRUE for at least one PLC cycle. If the action transition "ActionTransON" is met, the flag "ActionON" is FALSE.

As long as the step with the OFF-qualifier is active, the action is processed in the "Disable" status. During this processing, the "ActionOFF" flag is TRUE for at least one PLC cycle. If the action transition "ActionTransOFF" is met, the flag "ActionOFF" is FALSE.

Sequence description of the SFC

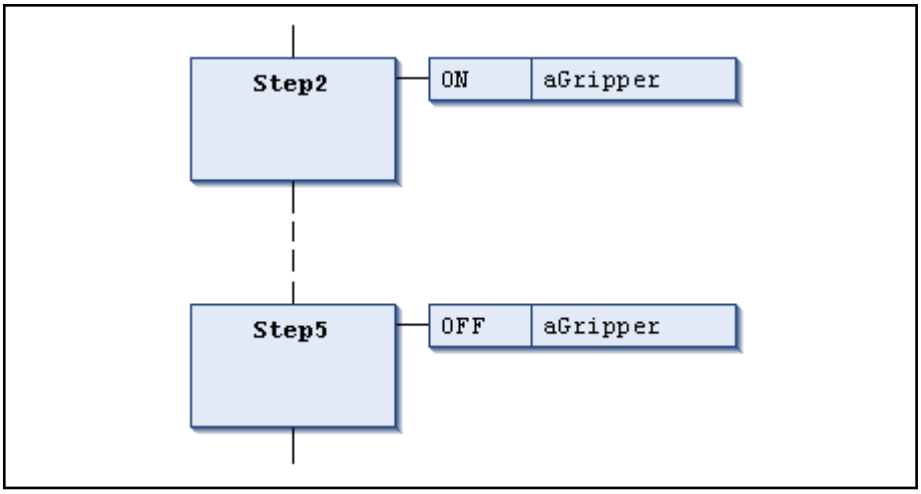


Fig. 3-6: Call of an action with ON- and OFF-qualifier

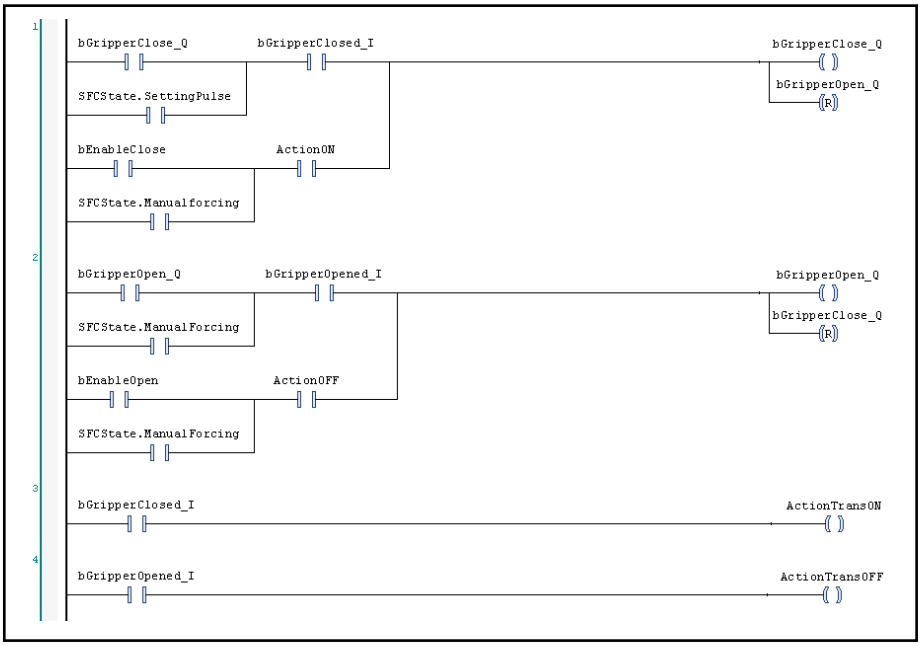


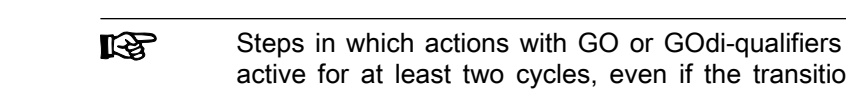
Fig. 3-7: Use of the "ActionON" flag in an action

This qualifier allows for instance to enable or disable hydraulic valves in an SFC or the cyclic processing of the output wiring.

GO- and GOdi-qualifier

"GO" and "GOdi" provide the action with the additional state "transfer value". This state can be identified by the global flag "ActionGO".

As soon as the step with the GO- or GOdi qualifier is enabled, the action is processed in the transfer value state for one cycle (ActionGO becomes TRUE for one PLC cycle). The variable value at the action qualifier is copied to the global variable “ActionValueREAL”, or, in case of GOdi to the “Action-ValueDINT” during the complete active processing of the action (ActionQ = TRUE). The validity of the value is only given within the action.



Steps in which actions with GO or GOdi-qualifiers are used, are active for at least two cycles, even if the transition has already been met. It is thus made sure that there is always a positive edge of the "ActionGO" flag (refer to "Signal-time diagramGO-qualifiers" on page 27).

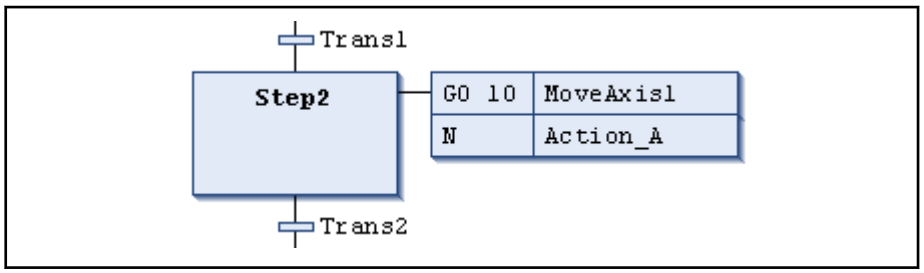


Fig. 3-8: Using the GO-qualifier

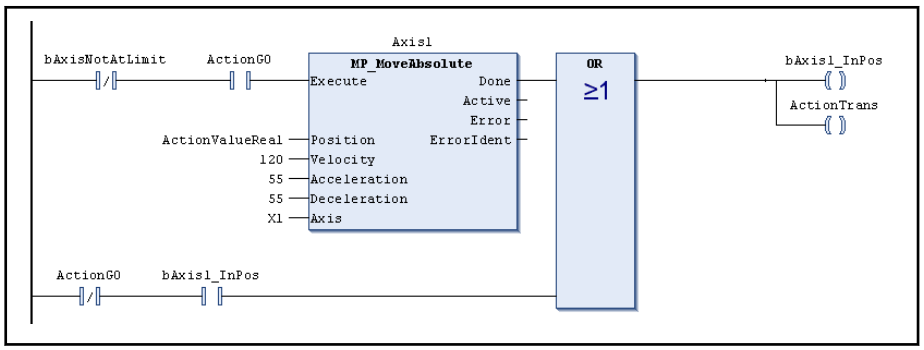


Fig. 3-9: Using the "ActionGO" and "ActionValueREAL" flags in an action

This qualifier allows for instance the transfer of target positions in an SFC to PLCopen function blocks to control field bus drives.

Processing the SFC

The SFC is processed in each cycle.

The individual parts of the SFC are processed in the following order:

- Processing of the "aUserIn" action
- Determining SFC transition conditions and disabling or enabling steps when the transition condition is met
- Determining which actions have to be processed or post-processed in this cycle
- Processing and post-processing of actions (except "aUserIn") in **alphabetical order**: This means that possibly after enabling a subsequent step, an action on the subsequent step is processed **first** and an action on a previous step is post-processed **afterwards**

3.2 Signal-time diagram qualifiers

Signal-time diagram: Post-processing with N-qualifier

The following signal-time diagrams are schematic circuit diagrams for the SFC representation.

Sequence description of the SFC

Signal-time diagram: Post-processing with N-qualifier:

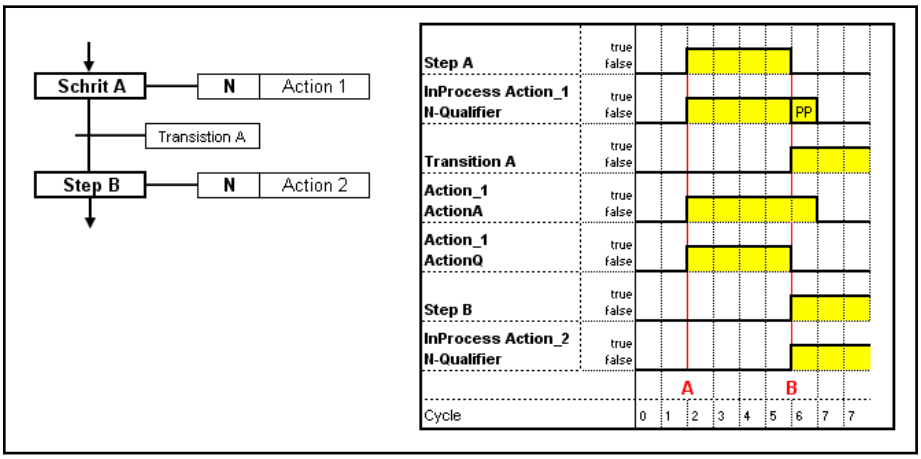


Fig. 3-10: Post-processing with N-qualifier

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Execution Action_1
B	Jog Semi Auto	Step A deselected by meeting transition A	ActionQ	SFC switches from step A to step B, post-processing of Action_1

Tab. 3-3: Table for signal-time diagram: Post-processing with N-qualifier

Signal-time diagram: Calling an action with transition met when enabling step:

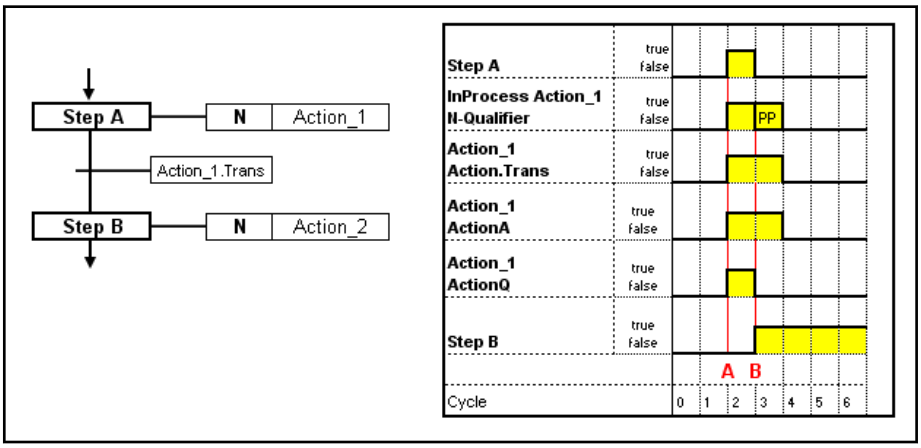


Fig. 3-11: Signal-time diagram: Calling an action with transition met when enabling step

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Execution Action_1 for two cycles
B	Jog Semi Auto	Step A deselected by meeting transition A, continue processing of Action_1	ActionQ	SFC switches from step A to step B, post-processing of Action_1

Tab. 3-4: Table for signal-time diagram: Calling an action with transition met when enabling step

Signal-time diagram: Calling the same action in consecutive steps:

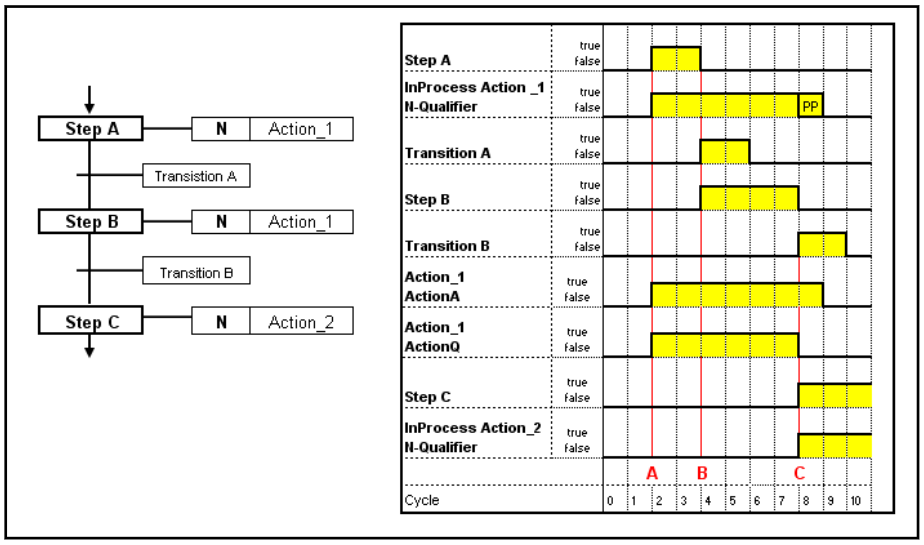


Fig. 3-12: Signal-time diagram: Calling the same action in consecutive steps

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Execution Action_1
B	Jog Semi Auto	Step A deselected by meeting transition A		Continued processing of Action_1
C	Jog Semi Auto	Step B deselected by meeting transition B	ActionQ	SFC switches from step B to step C, post-processing of Action_1

Tab. 3-5: Table for signal-time diagram: Calling the same action in consecutive steps

Sequence description of the SFC

Signal-time diagram: Calling the same action in consecutive steps with the transition already met and the assignment "ActionQ" → "ActionTrans" (assignment of "ActionQ" or linked AND conditions to "ActionTrans"):

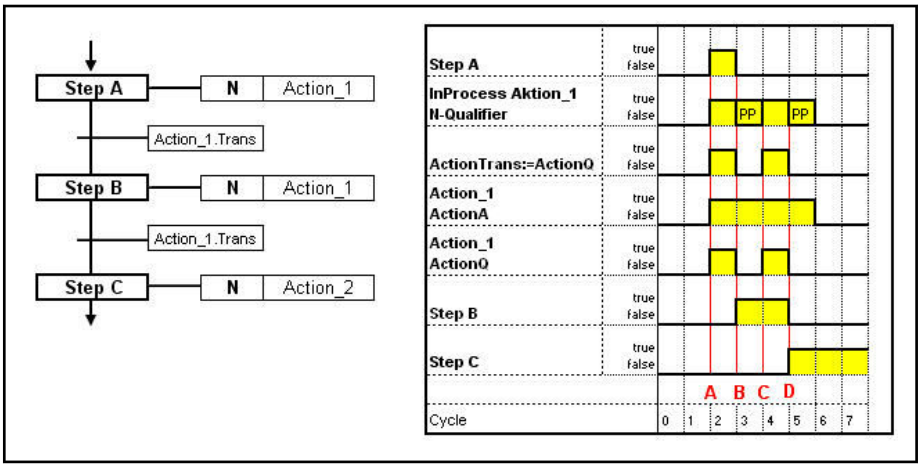


Fig. 3-13: Calling the same action in consecutive steps with the transition already met and the assignment "ActionQ" → "ActionTrans"

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Execution Action_1 "ActionQ" assigns TRUE to "ActionTrans" in "Action_1")
B	Jog Semi Auto	Step A → B by existing transition met upon step activation	ActionQ	Continued processing of Action_1 (in post-processing)
C	Jog Semi Auto	Step B → C by existing transition met upon step activation	ActionQ	Continued processing of Action_1
D	Jog Semi Auto	Step C-->D by existing transition met upon step activation	ActionQ	Continued processing of Action_1 (in post-processing)

Tab. 3-6: Table for signal-time diagram: Calling the same action in consecutive steps with transition already met when enabling the step and assignment "ActionQ" → "ActionTrans"

Signal-time diagram: Calling the same action in consecutive steps with transition already met on step activation without assignment "ActionQ" → "ActionTrans":

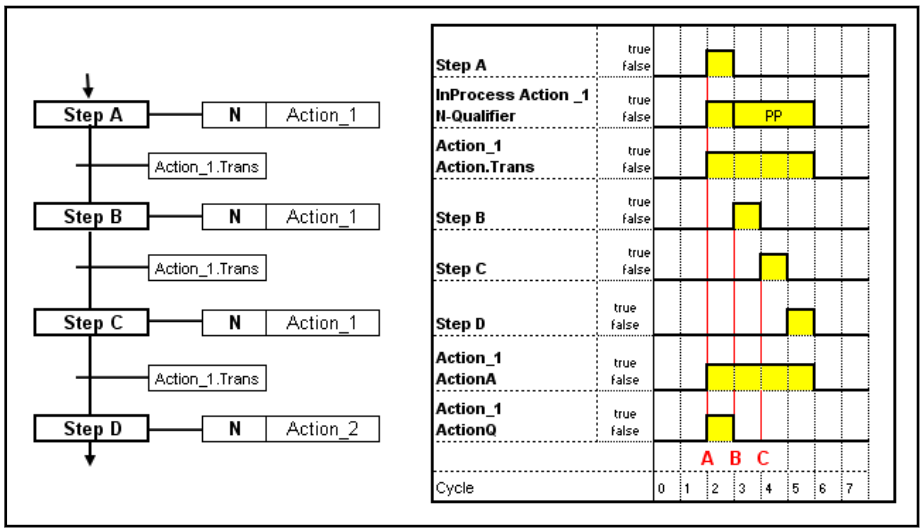


Fig. 3-14: Signal-time diagram: Calling the same action in consecutive steps with transition already met on step activation without assignment "ActionQ" → "ActionTrans"

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Processing Action_1
B	Jog Semi Auto	Step A → B by existing transition met upon step activation	ActionQ	Continued processing of Action_1 (in post-processing)
C	Jog Semi Auto	Step B → C by existing transition met upon step activation		Continued processing of Action_1 (in post-processing)

Tab. 3-7: Table for signal-time diagram: Calling the same action in consecutive steps with transition already met on step activation

Signal-time diagram R-qualifiers

Signal-Time response for the R-qualifiers:

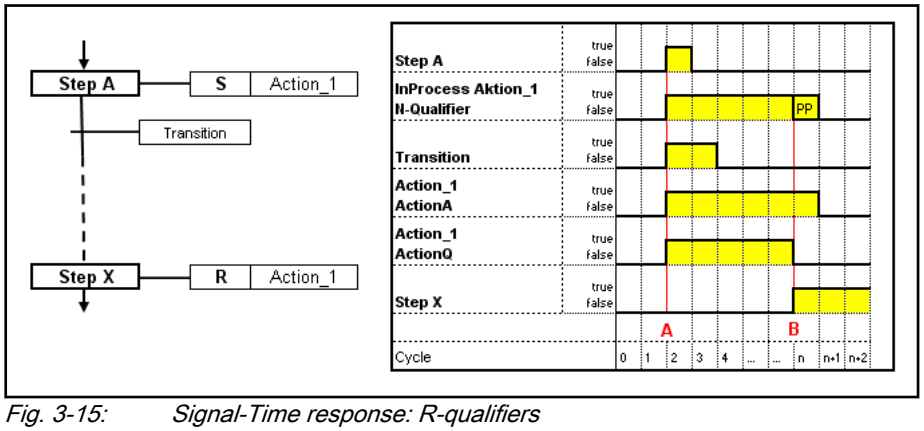


Fig. 3-15: Signal-Time response: R-qualifiers

Sequence description of the SFC

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q	Permanent processing of Action_1 set
B	Jog Semi Auto	Step X enabled	ActionQ	Resetting permanent processing, post-processing of Action_1

Tab. 3-8: Table for signal-time diagram: R-qualifiers

Signal-time diagram qualifiers P/P0/P1

Signal-Time response for the qualifiers P/P0/P1:

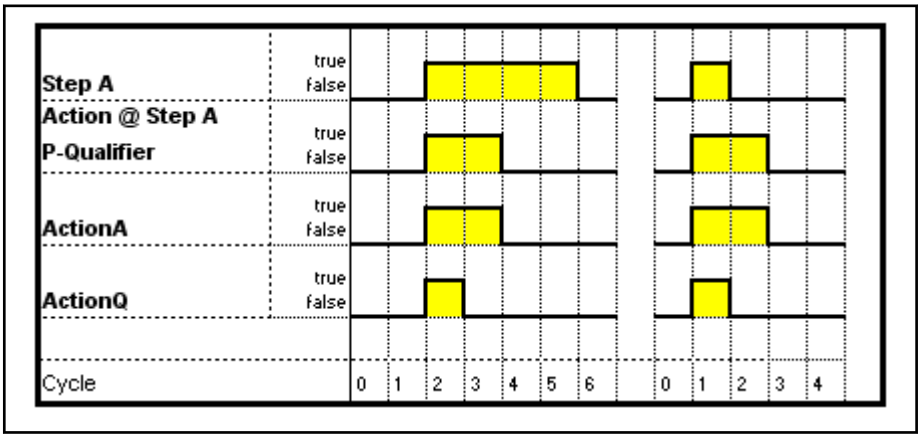


Fig. 3-16: Signal-time diagram: P-qualifiers

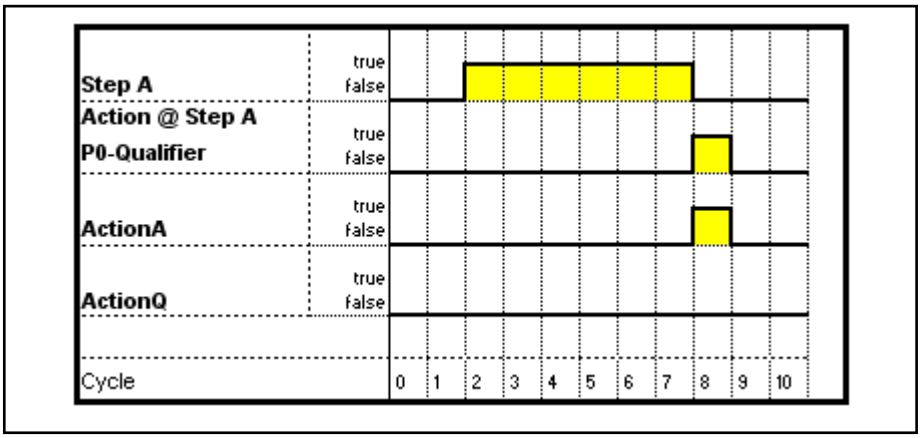


Fig. 3-17: Signal-time diagram: P0-qualifiers

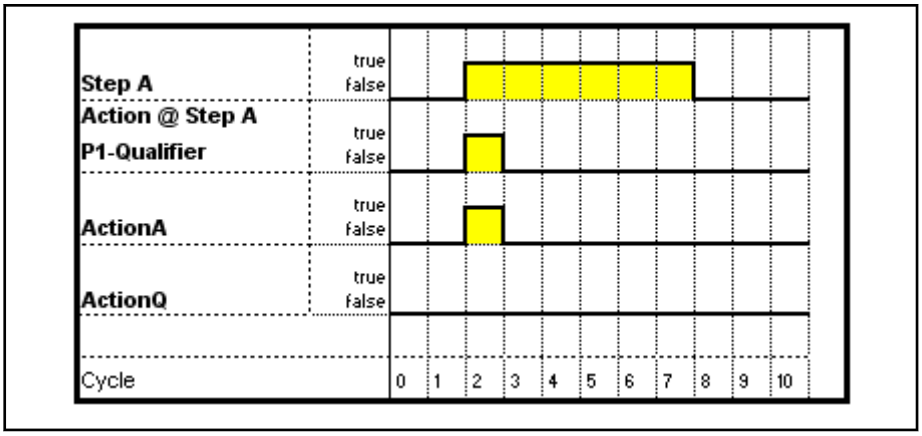


Fig. 3-18: Signal-time diagram: P1-qualifier

Signal-time diagramGO-qualifiers

Signal-time response for the GO- and GOdi-qualifiers:

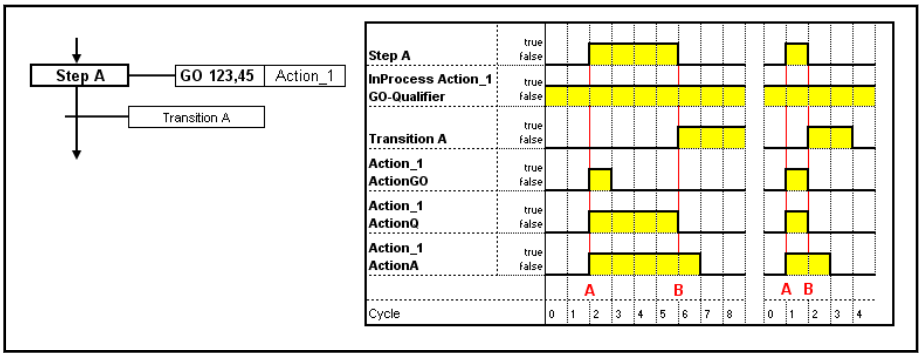


Fig. 3-19: Signal-time diagram: GO-qualifiers

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionA/Q ActionGO	Flag "ActionGO" on TRUE for one cycle, processing of the action with GO- or GOdi qualifier for at least two cycles
B	Jog Semi Auto	Transition condition met	ActionQ	"Post-processing response" for one cycle for the evaluation in connection with ActionQ

Tab. 3-9: Table for signal-time diagram: GO-qualifiers

A positive edge on `iSFC.Enable` always creates a positive edge on the system flag "ActionGO". The same applies when calling an action in "manual mode". The system flag "ActionGO" receives one positive edge per call in consecutive steps with the call of the same GO- or GOdi action.

Sequence description of the SFC

Signal-time diagram ON-/OFF-qualifiers

Signal-time response for ON-/OFF-qualifiers:

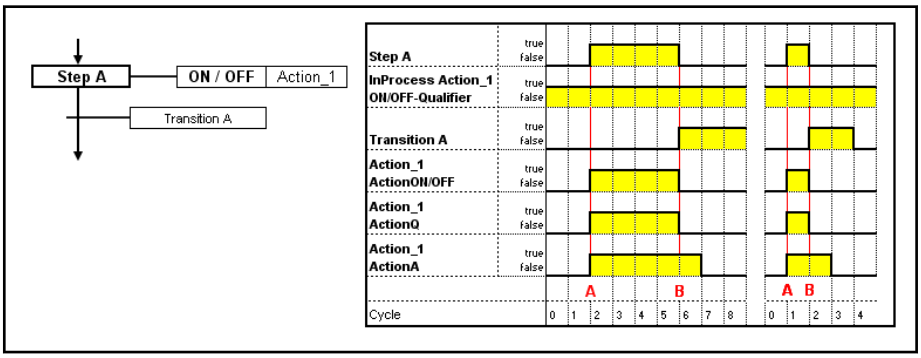


Fig. 3-20: Signal-time diagram: ON-/OFF-qualifiers

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Step A enabled by selection	ActionON/OFF ActionQ ActionA	
B	Jog Semi Auto	Transition condition met	ActionQ	"Post-processing response" for one cycle for the analysis in connection with ActionON/OFF

Tab. 3-10: Table for signal-time diagram: ON-/OFF-qualifiers

4 Programming

4.1 General information

This chapter describes the general SFC programming. The input and output signals as well as the behavior of the SFC are described in chapter 5 "SFC types" on page 51.

4.2 Definition of the supported variables for the ProVi SFC diagnostics

Declarable variable types

The ProVi SFC diagnostics supports the following variable types:

1. Standard types:

BOOL	DINT	LREAL
BYTE	LINT	DT
WORD	USINT	TOD
DWORD	UINT	TIME
LWORD	UDINT	POINTER
SINT	ULINT	STRING
INT	REAL	STRING[8] (with length specification)

To issue messages, only Boolean operations may be used irrespective of the programming code.

E.g. Boolean variables in ST:

```
{ProVi Error, Module: 1, Not: FALSE, Set: FALSE, CA: TRUE, Indicate: FALSE, No: 123}
bMyBool := bMyBool_1 OR bMyBool_2;
```

or different integer variables with the point operator:

```
{ProVi Warning, Module: 1, Not: FALSE, Set: FALSE, CA: TRUE, Indicate: FALSE, No: 456}
iMyInt.5 := dwMyDWord.6 AND byMyByte.7 OR bMyBool;
```

All standard types mentioned above can use variables as placeholders in message texts. However, these placeholders had to be declared in the project.

2. *Arrays:*
   - ARRAY[from .. to] OF XXX

     or ARRAY[from .. to] OF ARRAY[from .. to] OF XXX

     (XXX = standard types, function blocks or structures) "from" "to" - have to be standard types.

☞ *Multi-dimensional arrays are only supported up to the third dimension:*

- `ARRAY[`from.. to, from .. to`] OF` (standard types, function blocks or structures)
- `ARRAY[`from.. to, from .. to, from .. to`] OF` (standard types, function blocks or structures)

The array elements have to be addressed directly or via enum for the feasibility display and the online criteria analysis:

- `sMy_Struct.Array[5, 5].Struct.bMyBool`
- `sMy_Struct.Array[five, 5].Struct.bMyBool`

Indirect addressing is not supported:

- `sMy_Struct.Array[iMyInt].Struct.bMyBool`

Function blocks, in which messages are issued, may be instantiated using arrays. When declaring these arrays, only decimal specifications, local or global project constants from the project are permitted. The used constants have to be initialized with a decimal value.

Example of a supported declaration:

```
aMy_FB_Array: ARRAY[1..17] OF My_FB
aMy_FB_Array: ARRAY[1..Constant_local] OF MY_FB;
aMy_FB_Array:  ARRAY[Glob_ConstMin..Glob_ConstMax]  OF
MY_FB;
```

Example of unsupported declarations:

```
aMy_FB_Array:           ARRAY[OtherProg.ConstMin..Li-
brary.GVL.ConstMax] OF MY_FB;
```

Constants from other POUs or libraries are not supported.

Example of the implementation or calling of such declared function blocks:

`Array_1[Array_2[Int]]` -"Array_2" has to be globally declared

`Array[Standard_Type].Array[Standard_Type]`

3. **Structures:**

   Example of a supported definition:

   ```
   TYPE MyStruct :
   Struct
   ```

   - `Standard types`
   - `Struct`
   - `Array[Standard_Type..Standard_Type,`
     `Standard_Type..Standard_Type] OF ...`

   ```
   End_Struct
   END_TYPE
   ```

   Example of supported declarations:

```
sMy_Struct : MyStruct;
```

Example of the implementation:

```
sMy_Struct.Array[5, five].Struct.Standard_Typ
```

Instantiation

Example:

`Program.InstanceFB.BOOL`

`Program.InstanceFB.InstanceFB_2.BOOL`

`Program.InstanceFB.InstanceFB_2.InstanceFB_3.BOOL`

`GVL.InstanceFB.BOOL`

`Program.InstanceFB[Index,Index2,Index3].bool` - "Index" has to be a number or an enum if this variable is used for the feasibility display or the criteria analysis.

☞ Using the criteria analysis (CA) and the feasibility display, the following applies to all variables involved:

- Only operations of Boolean variables can be analyzed
- The logic to be diagnosed has to be as simple as possible
- The variables have to be declared in the project
- Complex calculations, comprehensive and nested function block calls should be outsourced to the function blocks if possible
- Complex or indexed structure or array elements should be replaced by simple Boolean variables
- The deeper the nesting and the higher the number of elements, the longer the diagnostic data generation. Saving the states in case of a diagnostic issue requires longer accordingly

---

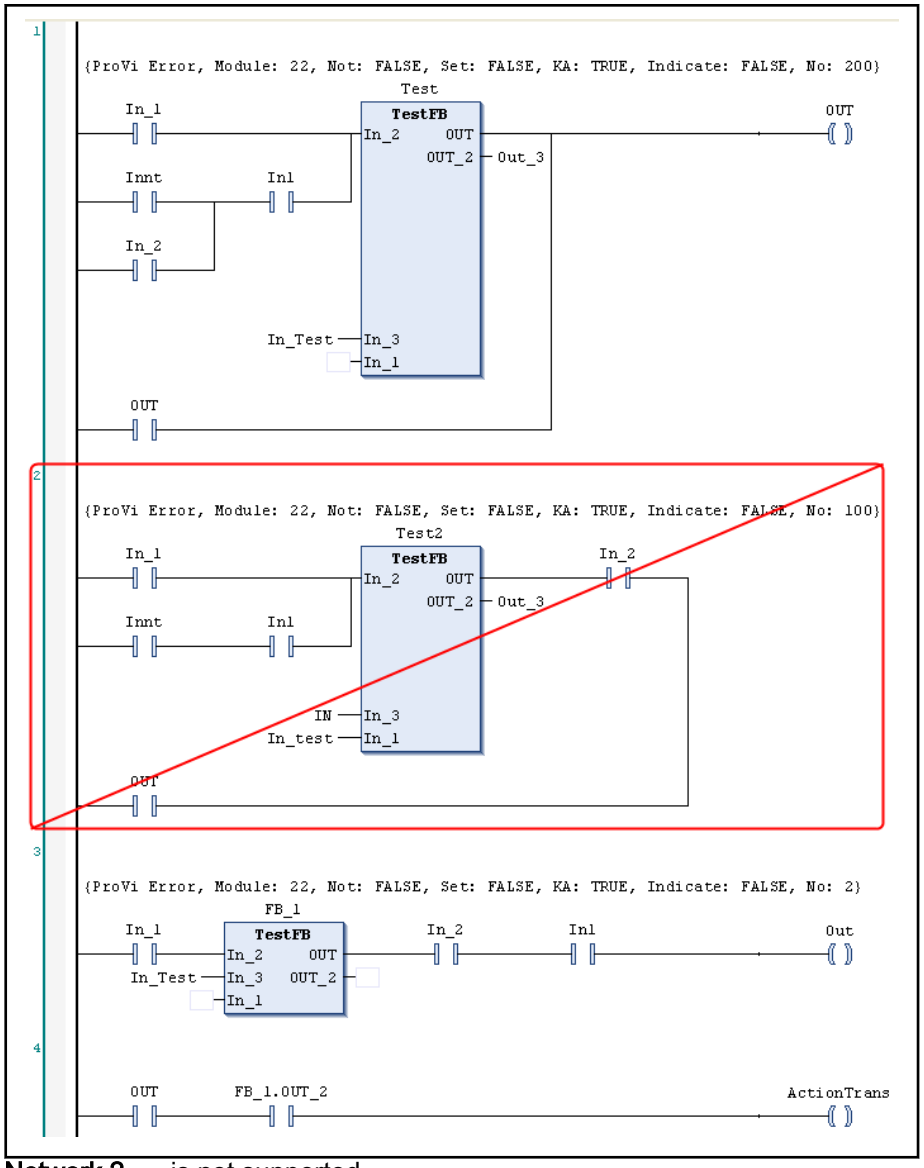
Restrictions

The following restrictions apply to the criteria analysis (CA) and the feasibility display:

- VAR_IN_OUT, VAR_TEMP variables, bit access and array variables with indirect addressing can be used to issue the ProVi or SFC messages. These messages are not supported by the criteria analysis. The frozen variable states are not determined and no criteria analysis is calculated. If the states of these VAR_IN_OUT, VAR_TEMP variables, bit access and array variables with indirect addressing should be used, these variables have to be projected via the intermediate flag.

  When these variables are used, a warning is output during compilation. If this warning is ignored, an error message is displayed in the visualization device under CA. This error message reports that the criteria analysis could not be calculated due to missing variables. In operating screens, a question mark signals this problem in the feasibility display.
- Non-Boolean variable types do not have an effect on the criteria analysis
- Box with "EN" (also with Boolean output) must not be used in diagnostic networks. Optionally, function blocks or functions can be used. The first input and the first output are considered by the diagnostics as if the input and the output would be connected internally

- POUs for the feasibility display created in the programming code "ST" may not contain any multi-line assignments, since the feasibility can otherwise not be calculated
- *Using function blocks, functions and line branching:*
  - The ProVi SFC diagnostics does not support any derived function blocks and may not be programmed as such
  - Only one function block may be used per network
  - Only the first input and the last output are considered (short-circuited). The current result of the input and output has to be assigned to a Boolean variable
  - The logic of the first input may only consist of Boolean variables
  - The first output has to be assigned
  - The function block may be provided with an unlimited number n of inputs/outputs
  - There may be no further logic at the inputs / outputs 2 to n
  - The inputs and outputs 2 to n may also remain unassigned
  - To ensure a unique network limit identification, the inputs/outputs may not be used at the first place in the subsequent network
  - Each function block output can be used to create a transition condition (last network in the screen)



Network 2 is not supported

Fig. 4-1: Example the supported and unsupported function block usage

- When using line branching in diagnostic networks, note that only the uppermost branch is responsible for issuing the message. The criteria analysis is only calculated for this branch. Do not confuse line branching with multiple assignments. The diagnostics supports multiple assignments. All assigned variables are displayed for the criteria analysis

Excluding from ProVi SFC diagnostics

When using assignments not supported by the ProVi SFC diagnostics, these networks or assignments can be completely excluded from the diagnostics. Open this functionality via the menu item "Diagnostics" or via the context menu in the relevant editors. The following pragma `{ProVi SFC diag: Exclude}` is inserted as response and for identification.

Programming

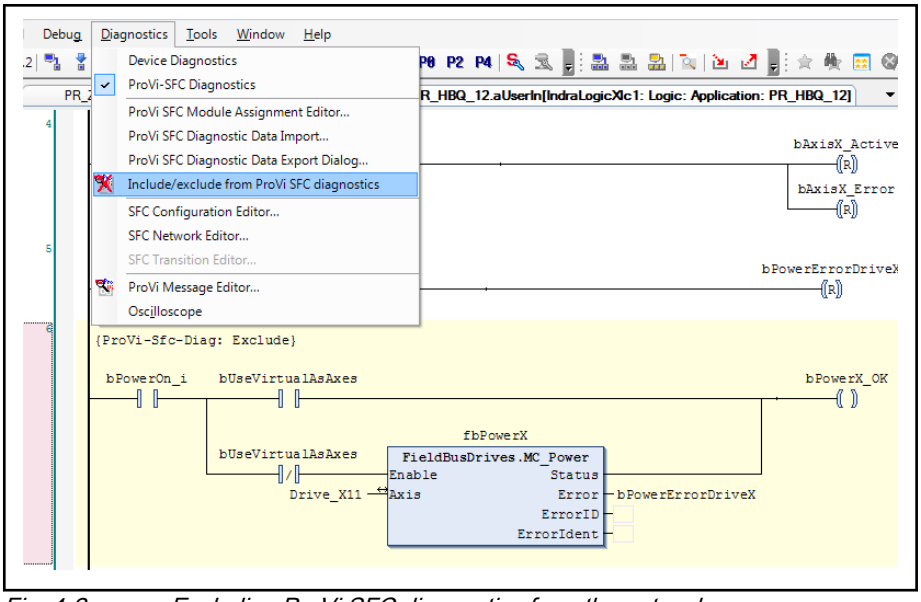


Fig. 4-2: Excluding ProVi SFC diagnostics from the network

If a network or an assignment is excluded from the ProVi SFC diagnostics like this, this exclusion applies to all diagnostic messages.

Diagnostic limit values

Do not exceed the following limits when using the ProVi SFC diagnostics:

- Maximum number of POUs with diagnostics = 2048
- Maximum number of instances that can be created by a diagnostics POU = 512
- Maximum number of messages in one POU = 4096
- Maximum number of SFCs with operation modes including instances = 256
- Maximum number of instances that can be created by an SFC with operation modes = 32
- Maximum number of steps in an SFC with operation modes = 512
- Maximum number of networks with diagnostics in an SFC with operation modes = 131072
- Maximum number of transitions in an SFC with operation modes = there is currently no limit

If the specified limits are exceeded, an error is output.

4.3 Basic settings

Creating the SFC

To program a machine in the sequential function chart with operation modes and diagnostics, a program or a function block is defined in the sequential function chart.

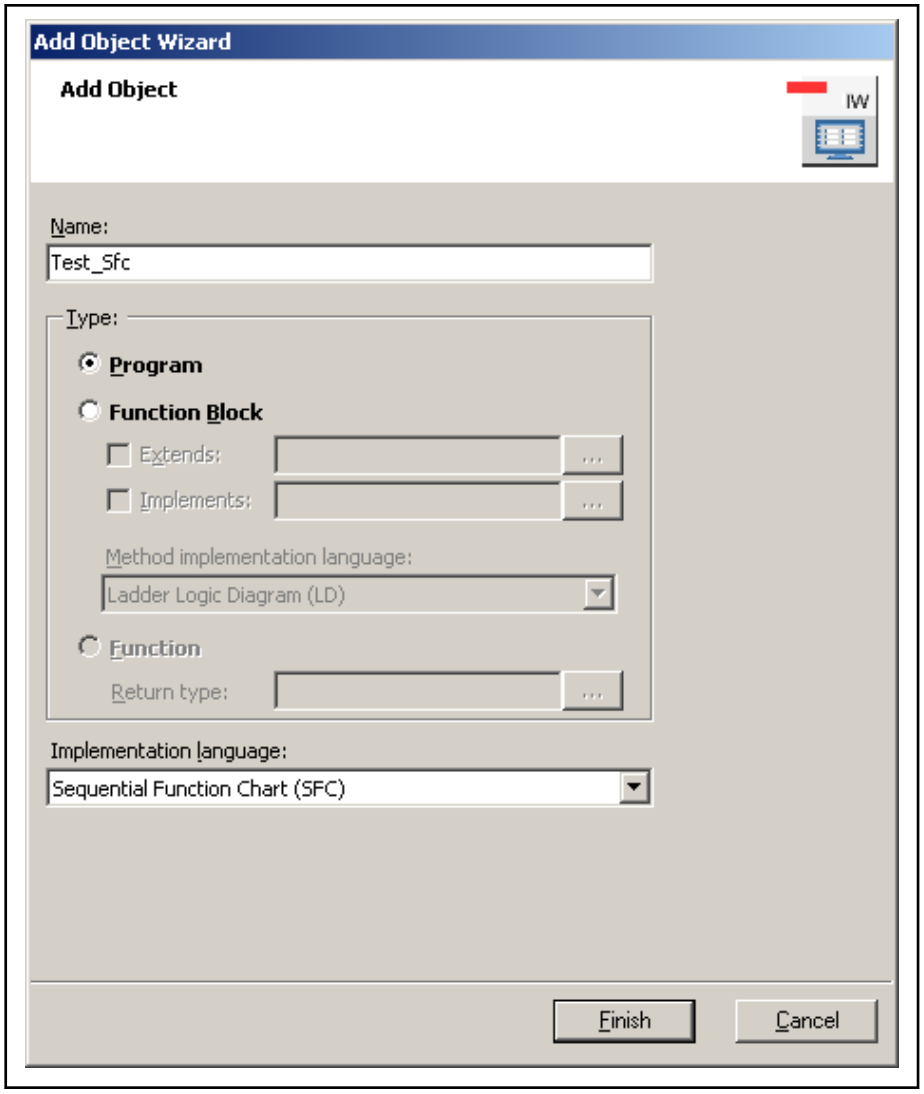


Fig. 4-3: Add object wizard (SFC definition)

After clicking on **Finish**, a basic SFC structure is automatically created.

This basic structure looks as follows:

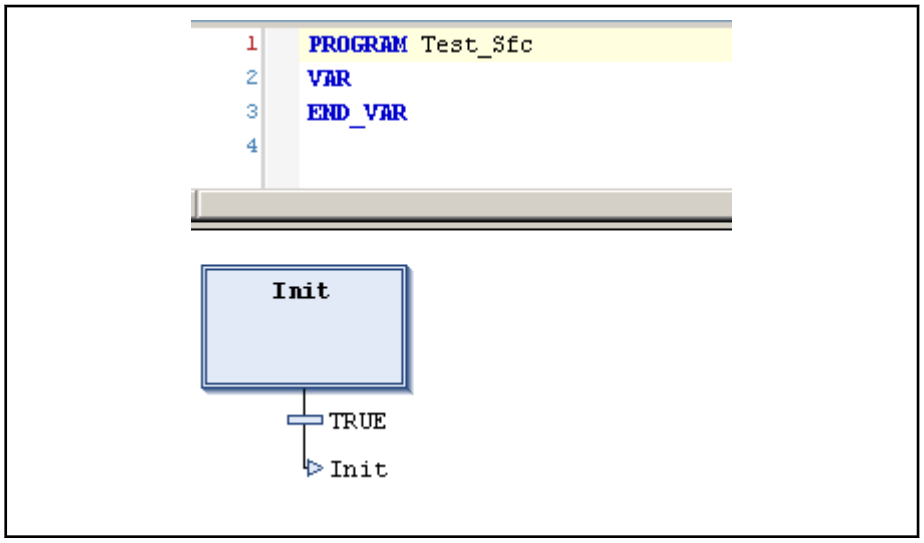


Fig. 4-4: SFC basic structure

Condition for SFC diagnostics

The diagnostics is enabled via the menu items **Diagnostics ▸ ProVi SFC diagnostics**.

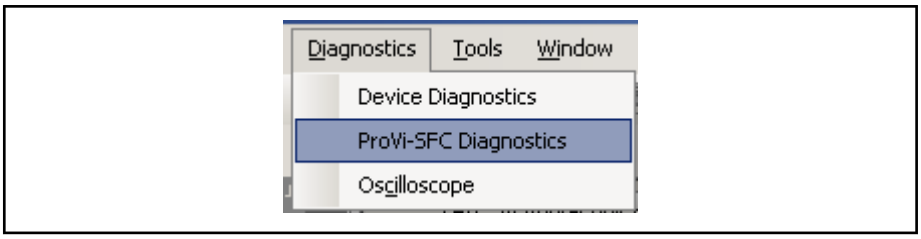


Fig. 4-5: Enabling diagnostics

4.4 Configuring the SFC

After enabling the diagnostics, the SFC can be configured. Under **Diagnostics ▸ SFC configuration editor**, the created SFC can be selected and configured in the SFC configuration editor. In the default configuration, "With operation modes and diagnostics" is not selected.

Selecting this option allows making further settings:

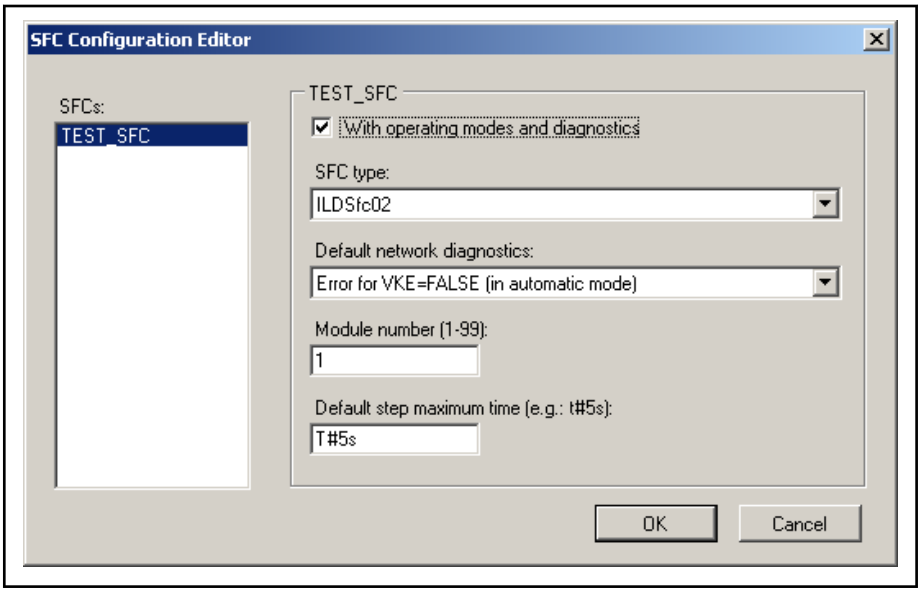


Fig. 4-6: SFC configuration editor

Under "SFC type", only "ILDSfc02" can be selected. The next setting is "Default network diagnostics" (is preset to "VKE=FALSE").

Depending on the diagnostic concept, it may also be changed

Selecting standard network diagnostics for monitor errors (MonError)	Note
No error	No network diagnostics in the SFC
Error in CR=TRUE	Diagnostics to CR=TRUE in all networks
Error at CR=FALSE	Diagnostics to CR=FALSE in all networks
Error at CR=TRUE (ON execution)	Diagnostics at CR=TRUE in all networks of an action which is actively executed with ON-qualifier
Error at CR=FALSE (ON execution)	Diagnostics to CR=FALSE in all networks of an action actively processed with ON-qualifier

Error at CR=TRUE (OFF execution)	Diagnostics at CR=TRUE in all networks of an action actively processed with OFF-qualifier
Error at CR=FALSE (OFF processing)	Diagnostics at CR=FALSE in all networks of an action actively processed with OFF-qualifier
Error at CR=TRUE (in automatic mode)	Diagnostics at CR=TRUE in all networks when the SFC is in automatic mode
Error at CR=FALSE (in automatic mode)	Diagnostics at CR=FALSE in all networks when the SFC is in automatic mode
Error at CR=TRUE (ON processed in automatic mode)	Diagnostics at CR=TRUE in all networks of an action actively processed with ON-qualifier in automatic mode
Error at CR=FALSE (ON processing in automatic mode)	Diagnostics at CR=FALSE in all networks of an action actively processed with ON-qualifier in automatic mode
Error at CR=TRUE (OFF processing in automatic mode)	Diagnostics at CR=TRUE in all networks of an action actively processed with OFF-qualifier in automatic mode
Error at CR=FALSE (OFF processing in automatic mode)	Diagnostics at CR=FALSE in all networks of an action actively processed with OFF-qualifier in automatic mode

Tab. 4-1: Table: Selection of the network diagnostics

In the input field "Module number", the SFC is assigned to a module. Another setting option is the setting of the "Default maximum step time". The time format entry depends on the standard and looks as follows: "**T#5s**". The settings selected in this editor are valid for the entire SFC to the extent that the individual networks and steps do not provide otherwise.

In order to completely exclude an individual step from time monitoring despite having entered a default maximum step time in the SFC configuration editor, enter "**T#0s**" in the step attributes of the relevant step as maximum time (see "Time errors" on page 45).

After confirming the configuration settings in fig. 4-6 "SFC configuration editor" on page 36 with **OK**, the settings in the curly brackets are automatically inserted in the configured SFC, in the first line of the declaration. The variables for the SFC control are also created.

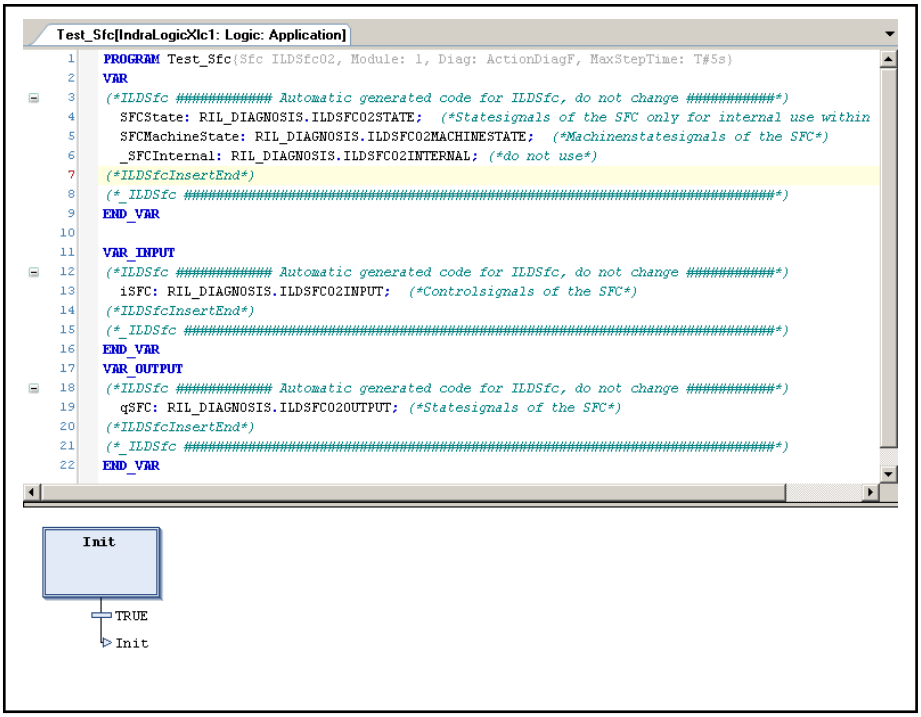


Fig. 4-7: Configuration settings in the SFC

If the SFC is configured, the individual actions can now be created if not yet done. The subsequent sequence as well as the connections are not relevant at this stage. First, the actions are to be determined.

As a rule, the creation of a PLC program in the SFC should begin with programming individual actions. The actions should be complete units that also fulfill the transition condition in chapter 4.5 "Action transition ActionTrans" on page 38. If the actions are created according to this principle, it is guaranteed that they can be used in different places within the SFC later on (in different steps or by copying to different SFCs).

4.5 Action transition "ActionTrans"

The following figure shows a possible structure for a self-contained action:

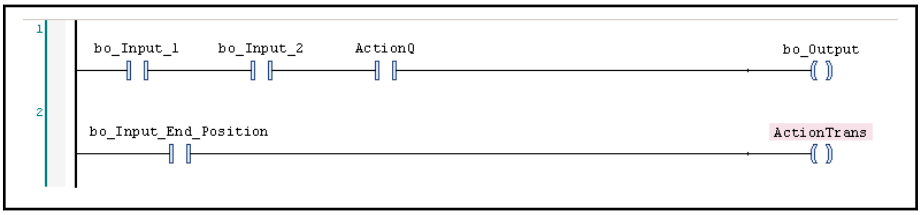


Fig. 4-8: Self-contained action with transition condition

The variables "bo_Input_1", "bo_Input_2" as well as "bo_Input_End_Position" are user bits. The "ActionQ" bit is an action-internal bit which is further described in the table "ActionA, ActionQ" on page 16. The "ActionTrans" flag is another action-internal flag that can be used for the transition. In this transition, this flag or these flags - if several actions are used - can be queried with "ActionName.Trans". The "ActionTrans" flag allows the programming of actions as self-contained units which can be called from several locations in the program with an implemented transition condition. This simplifies "programming" and "readability" of the program. Since the flag "ActionTrans" is defined globally and used depending on the action/network, trailing is not possible with this flag.

The "ActionTrans" flag furthermore provides the possibility to comfortably re-merge two parallel SFC branches.

The following figure shows an example of such a connection:

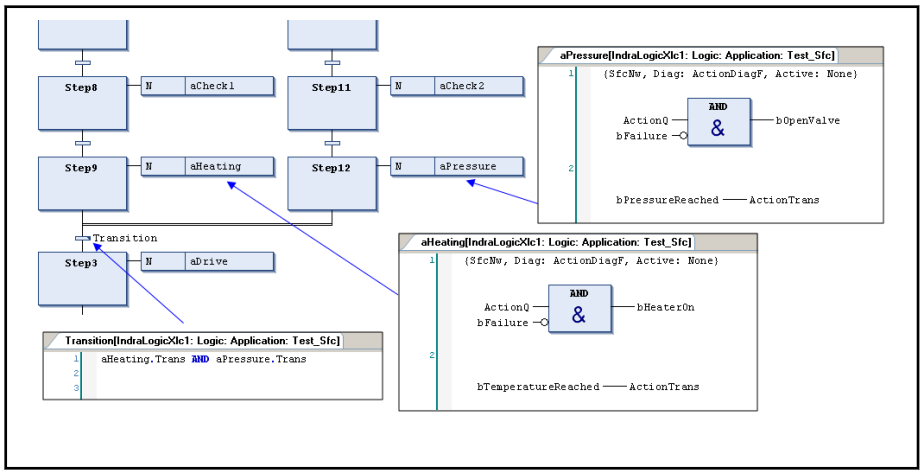


Fig. 4-9: Merging of two parallel SFCs with the "ActionTrans" flag

The actions "aHeating" and "aPressure" contain each one network diagnostics to the CR=FALSE. Switching from Step12/Step9 to Step3 is only carried out when "bPressureReached" **and** "bTemperatureReached" or their ActionTrans assignments are TRUE.

If the transition condition in the action "aHeating" or "aPressure" is met with ActionTrans, the network diagnostics in this action is masked in the networks which are provided with a logic connection to "ActionQ" (in the example always network 1). In case of an invalid transition condition, "ActionQ" is used to select the outputs "bHeaterOn" or "bOpenValve" and to enable the network diagnostics. In the example given above, this might result in mutual oscillation.

- The "ActionTrans" flag acts only locally in the relevant action. If "ActionTrans"=TRUE, the flag "ActionQ" = FALSE. To achieve step advancing, the "Actionname.Trans" **must** be used in the transition and can be linked with other conditions if necessary
- In actions with ON-/OFF-qualifiers, the transition assignment in the action is not carried out on "ActionTrans" but on "ActionTransON"; for OFF-qualifiers on "ActionTransOFF"
- If a condition is linked to an output control and if this condition is at the same time assigned to "ActionTrans" with negation, a monitor error is generated when monitoring is active and the condition is met

Redundant links such as these (see following figure) are not required and to be avoided:

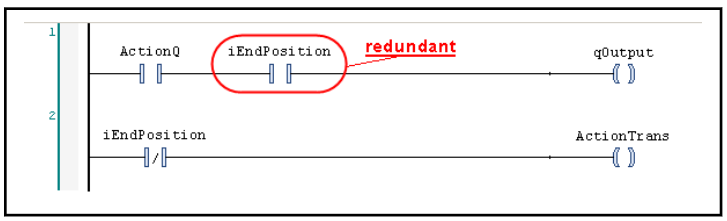


Fig. 4-10: Redundant links

When all inputs and outputs are wired and all required actions are implemented, the actual sequence can be programmed. To reasonably use the SFC with operation modes and diagnostics with all its functions provided, only two branches are to be realized:

The "Automatic" and the "Homing" branch.

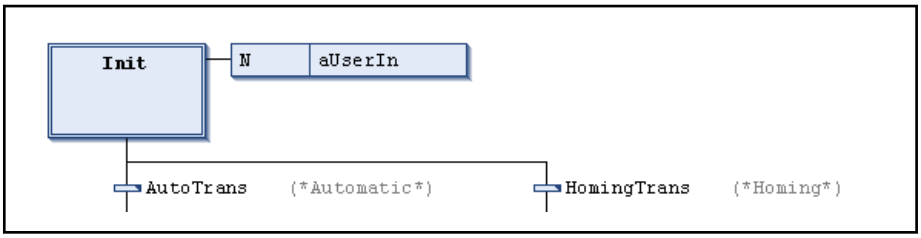


Fig. 4-11: Definition of automatic and homing branch

4.6 Automatic, manual, and homing branch

"Homing" branch

The "Homing" branch describes a defined movement to the "Home position". After reaching the home position, a return to the "Init" step is absolutely imperative in order to finalize the home request. The branch is only executed when the operation mode "Homing" was preselected. In addition, a transition condition has to be specified. The simplest case being "TRUE". It is also possible to create several "homing branches."

"Automatic" branch

The "Automatic" branch defines the sequence for the automatic mode.

Manual mode

There are different methods and possibilities to manually traverse the SFC. Each step and each action is provided with flags which allow the activation of the SFC irrespective of the current position. By assigning "TRUE" to the "ActionName.F" or "StepName.F" flag, the action or step is enabled or executed if the signals "iSfc.Manual" and "iSfc.Enable" are applied to the input of the SFC. The corresponding actions or steps in the SFC are enabled and processed.

- When controlling steps in manual mode, the transition conditions are taken into account and when they are met, the step is disabled (even if the signal from flag "StepName.F" is still active). A positive edge at the "StepName.F" flag enables the step again
- Even when the step transition is already fulfilled, the linked actions with N-qualifier are processed for at least one cycle and post-processed (pulse for one cycle on ActionQ)
- If a minimum step time was programmed, the step can be enabled for at least this period of time irrespective of the step transition state. Upon activation, the linked actions are processed for **at least** the duration of the minimum step time

For manual operations, it is also useful to create a "manual" branch in addition to the "automatic" and the "homing" branch. In this branch, the corresponding steps or actions can be enabled using the flags "ActionName.F" or "StepName.F" or the M-panel keys:

- that are not used in the automatic branch
- Steps linked to actions in the automatic branch, which are partially not processed in manual mode
- Steps not linked to required actions

The assignment of "TRUE" to flags "ActionName.F" or "StepName.F" can for instance be carried out in the cyclically called action "aUserIn". If an action is to be called with On or OFF processing, the assignment is made to "ActionName.FON" or "ActionName.FOFF".

4.7 HomePosition & StartCondition

As already mentioned in "Diagnostic messages" on page 10, errors such as "HomePosition", "StartCondition" and "TimeOut" are automatically generated and displayed in IndraWorks Operation without the user configuring them separately. The errors "HomePosition" and "StartCondition" are graded, i.e. the local variables "SfcMaschineState.HomePosition" and "SfcMaschineState.StartCondition" are available for every SFC. These local variables are automatically created in the corresponding SFC when the diagnostic data is generated (initial value on TRUE) and can be used directly in the SFC (e.g. by calling from the action "aUserIn").

"StartCondition" and "HomePosition" can also be defined for modules and/or for the entire machine. When "StartCondition" and "HomePosition" are defined for a module **and** for the entire machine, then both conditions (module **and** machine) have to be fulfilled to meet the SFC HomePosition/StartCondition.

The variables have to be declared globally according to the following specification:

- For the entire machine:

  `MachineStateControl : SFCMachineState;`

- For the modules:

  `MachineStateModule` + INDEX `: SFCMachineState;`

  A value between 1 and 99 is to be selected for "INDEX"

  For example: MachineStateModule13

The criteria analysis of "HomePosition" or "StartCondition" is only available for the local variables of "SFCMachineState".

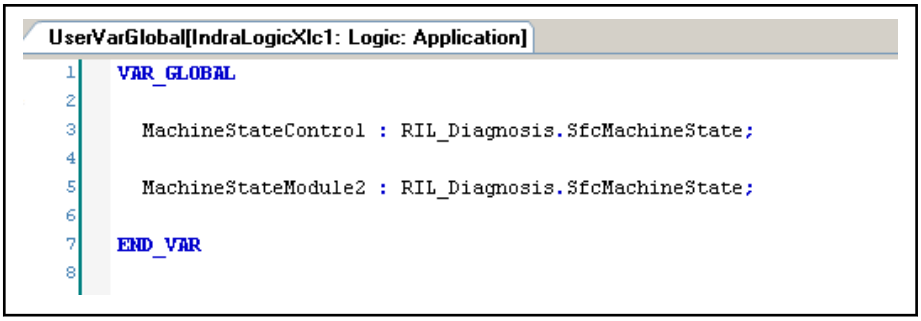


Fig. 4-12: Global declaration of machine or module state variables

The following figure shows an example of such assignments:

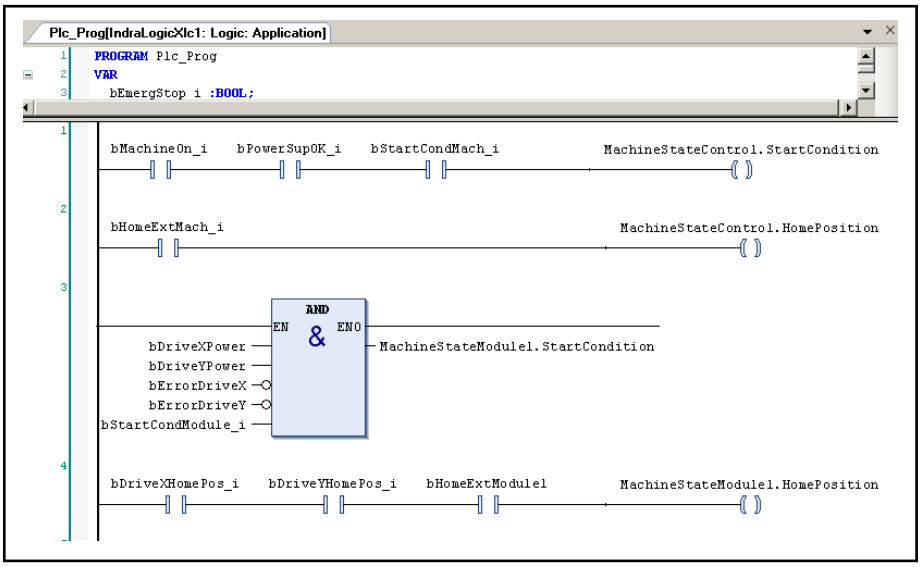


Fig. 4-13: Evaluation and assignment of state signals

If a machine or module state error is present for which several SFCs are used, the error is indicated for every SFC. This means that if three SFCs are used in module 1 and a "HomePosition" error is present, that is "MachineStateModule1.HomePosition"=FALSE, three error messages are displayed on the HMI device.

4.8 Transition properties

Transition for semi-automatic

The breakpoints for semi-automatic can be defined via **Diagnostics** ▸ **SFC Transition Editor** or via the context menu in the transition.

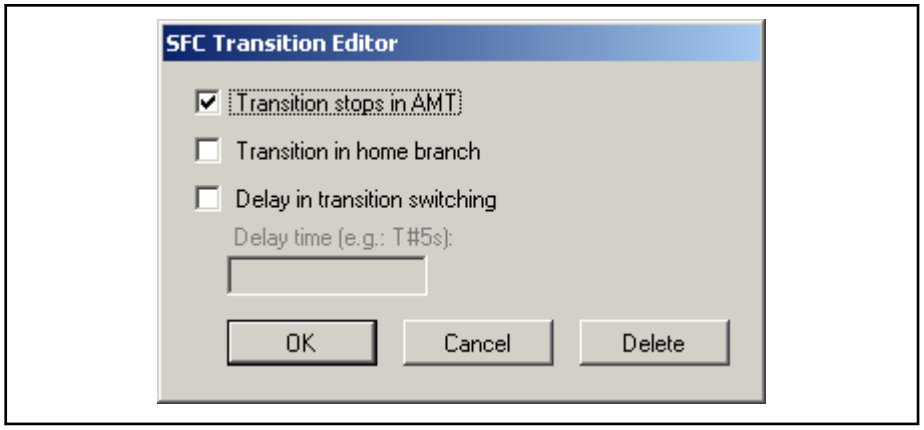


Fig. 4-14: SFC transition editor (transition stops in semi-automatic)

After selecting "Transition stops in semi-automatic" and subsequently exiting the editor with **OK**, the selection is placed in the volatile memory and can be added to the desired transition with <Ctrl> + <V>.

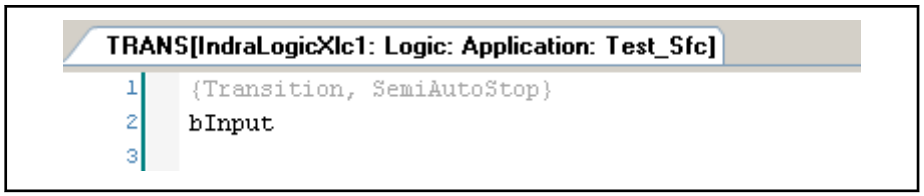


Fig. 4-15: Adding the selection to the transition

Transition for home position branch

The homing branch can be defined via **Diagnostics ▸ SFC Transition Editor** or via the context menu in the transition.

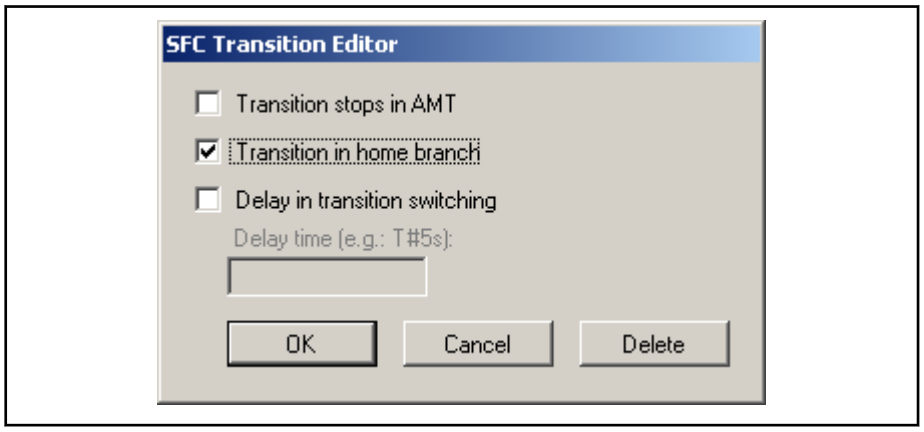


Fig. 4-16: SFC transition editor (transition to home position branch)

After selecting "Transition stops in home position branch" and subsequently exiting the editor with **OK**, the selection is placed in the volatile memory and can be added to the desired transition with <Ctrl> + <V>.

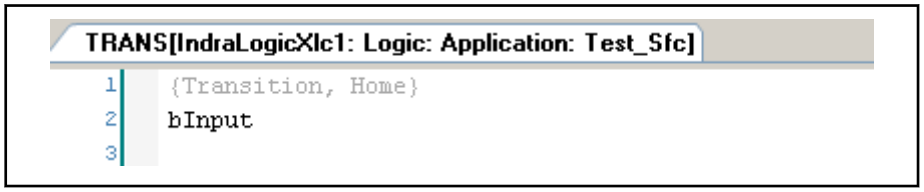


Fig. 4-17: Adding the selection to the transition

☞ Several home position branches can exist. The transition conditions given in each case determine which branch it is jumped to.

Programming

Configuring transition delay

The configuration of the transition delay can be defined via **Diagnostics ▸ SFC transition editor** or via the context menu in the transition.

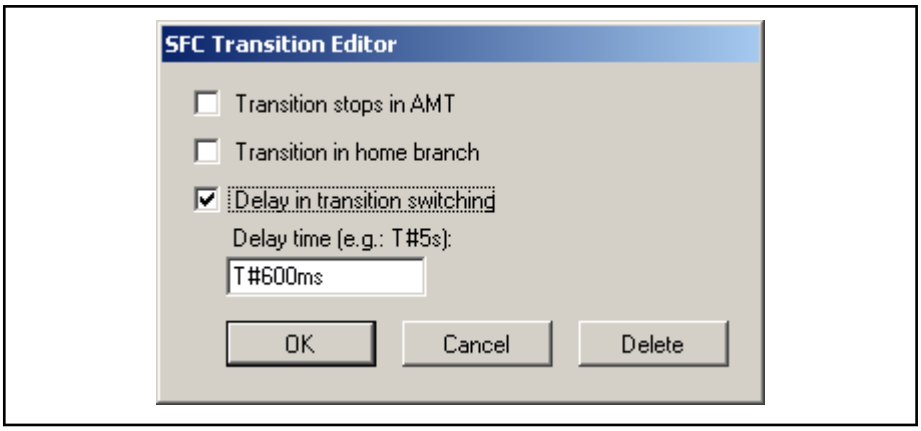


Fig. 4-18: SFC transition editor (transition switches with time delay)

After selecting "Transition switches with delay" and entering the desired time and subsequently exiting the editor with **OK**, the selection is placed in the volatile memory and can be added to the desired transition with <Ctrl> + <V>.

Fig. 4-19: Adding the selection to the transition

Signal-time diagram: Transition delay

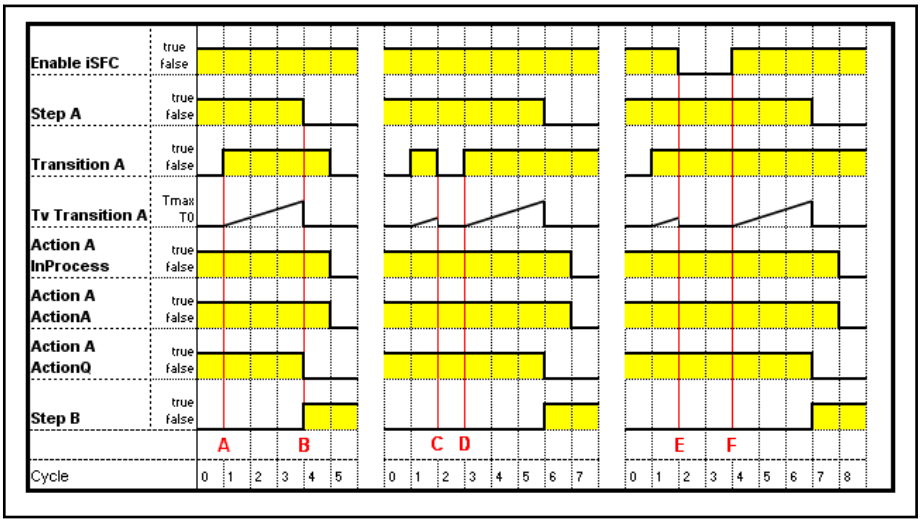


Fig. 4-20: Signal-time diagram transition delay

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Transition condition met: 0-1		Transition delay time started
B	Jog Semi Auto	Transition delay time elapsed	ActionQ	Action A moves to post-processing
C	Jog Semi Auto	Transition condition no longer met: 1-0		The transition delay time is reset
D	Jog Semi Auto	Transition condition met: 0-1		Transition delay time starts again
E	Jog Semi Auto	iSFC.Enable: 1-0, Transition condition met		The transition delay time is reset
F	Jog Semi Auto	iSFC.Enable: 0-1, Transition condition met		Transition delay time starts again

Tab. 4-2: Table transition delay

☞ The transition delay runs independent of the "iSFC.Stop" input.

4.9 Diagnostics

Time errors

The maximum step time set in the SFC configuration editor is valid for all steps in the respective SFC, unless otherwise defined in the individual steps. In the individual steps, the maximum step time and the minimum time step can be set. Click on the respective text with the left mouse button. On the right under "Properties" tab, the times can also be entered in the known format "**T#5s**" next to the comment.

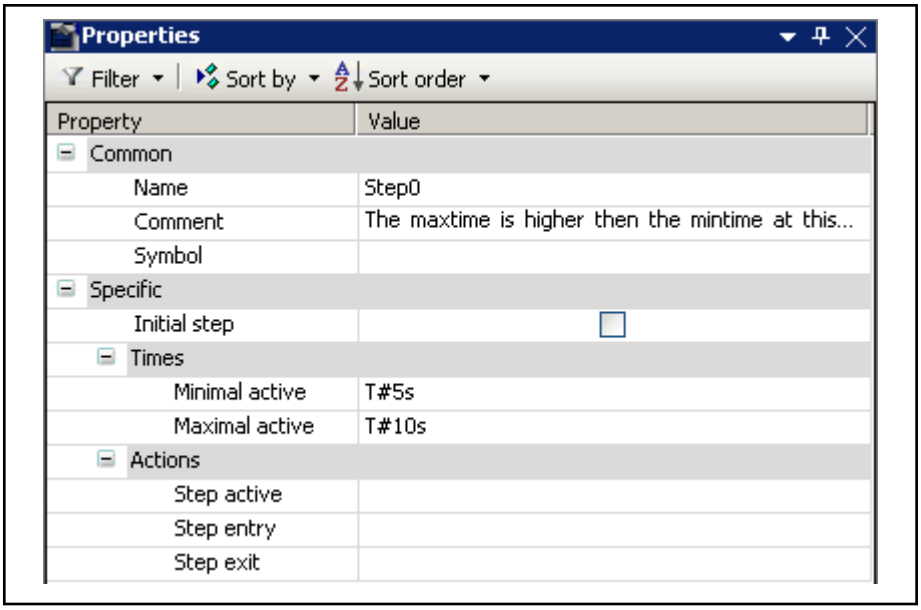


Fig. 4-21: Step attributes (editor)

Please enter comments to step names in this field only (for this, see also chapter "Incorrectly configured M-key" on page 73).

☞ If a minimum step time is entered in a step, processing this step has to take least this long irrespective of the subsequent transition. If, however, the default maximum step time is smaller than the minimum step time and if the maximum step time was not entered in the step attribute editor, a TimeOut error is always triggered after the minimum step time has elapsed. The indicated step name appears in the diagnostic message.

If the step transition is met but the minimum time has not yet elapsed, the actions with the linked step are processed. In this case, e.g. "ActionQ" and "ActionA" of an action with N-qualifier remain on "TRUE". If the transition assignment in an action with N-qualifier is met with "ActionTrans", "ActionQ" = FALSE and "ActionA" = TRUE for the processing in the minimum time.

The step processing time can be read out in the PLC program using "StepName.T".

The times set for the individual steps can be checked via the context menu (right mouse button) in the "Time Schedule". These values overwrite the default maximum step time.

☞ If the maximum standard step period was entered in the SFC configuration editor and if a step is to be completely excluded from time monitoring, this can be done by entering zero seconds in the step attribute editor in the format "**T#0s**" for the maximum time step. The minimum step time, however, can be maintained.

Monitor errors

The monitor error is processed similar to the time error. The condition set under **Diagnostics ▸ SFC configuration editor** applies to the entire SFC, i.e. for all networks in the SFC, unless otherwise defined in the individual networks. The individual networks can be defined via **Diagnostics ▸ SFC network editor** or via the context menu or the respective network.

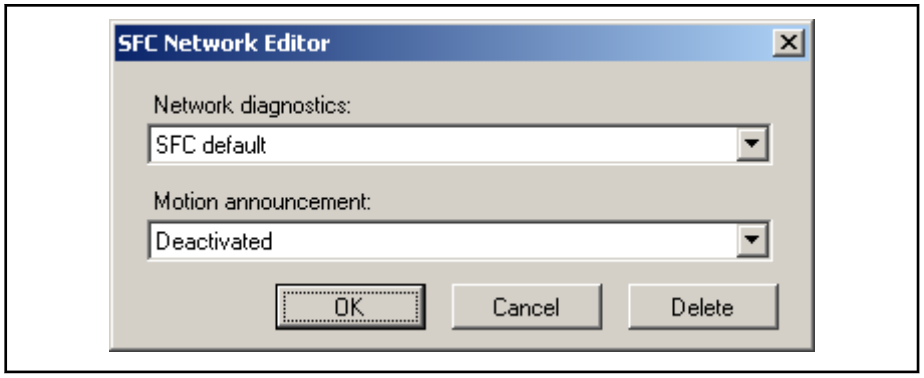


Fig. 4-22: SFC network editor (editing properties)

In addition to the configuration of monitor errors, this dialog also allows the entry of the relevance of this network for the feasibility display. Please note that all networks are not relevant to the feasibility by default. To enable a network for the feasibility, select this explicitly via this dialog.

There are four options:

- "Deactivated" - For explicit deactivation
- "Activated" - For operation mode-independent activation
- "Activated for: ON processing " - Activate for ON processing
- "Activated for: OFF processing " - Activate for OFF processing

After selecting the desired network diagnostics and exiting the editor with **OK**, the selection is saved to the volatile memory and can be added to the desired networks using <CTRL> + <V>. If a network is to be excluded from the diagnostics, the property "No Error" can be selected in the editor.

The following figure shows a possible structure of an action after editing the different network settings:

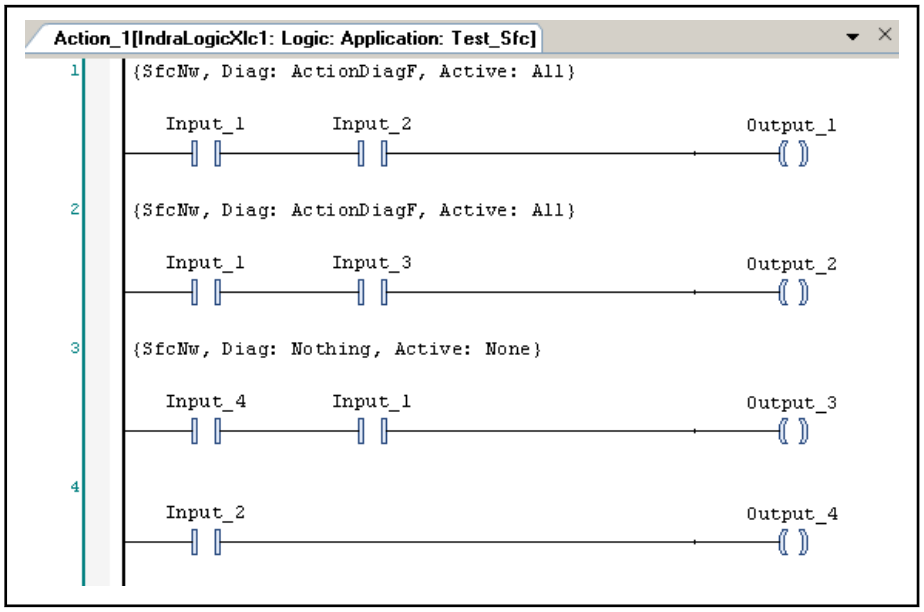


Fig. 4-23: Manual editing of the network diagnostics

If no explicit diagnostic properties has been edited for a network or the SFC default (see fig. 4-22 "SFC network editor (editing properties)" on page 47) is retained, the diagnostic type selected in the SFC configuration editor is retained.

After the SFC has been completed and configured, the diagnostic data can be created. Diagnostic data is created automatically. This creates diagnostic data for the entire project.

The library for the diagnostics "RIL_Diagnosis" is opened and updated automatically.

4.10 Module assignment editor

☞ Consider the following section only if you have divided your PLC project into different modules or if you use several instances of a function block with ProVi or SFC diagnostics.

If a function block with diagnostics is declared several times, this leads to the question in which module the diagnostics of the individual instances is to be shown. A unique module number can be assigned for each use of a diagnostics in the module assignment editor. Thus, the messages programmed for a specific module can also be assigned with a different module number later on.

Opening module assignment editor

Open the module assignment editor via the node **Diagnostic text data ▸ Module assignments** in the Project Explorer.

Enter the module assignments into the table considering the syntax.

Module assignment syntax

POU[.Instance][.Diagnostictype][.Messagetype][.OrgModulNo=NewModuleNo

The specifications in square brackets are optional (from left to right). Example: If no "Diagnostic type" is selected, no selection can be made for the parameters "Message type" and "Module number" located next to the right (for further examples see tab. 4-3 "Examples of the module number assignment" on page 49).

For the syntax of module assignments with the diagnostic type SFC, note that the parameters "Message type" and "Module number" located on the right, cannot be selected.

It is mandatory to enter the new module number to provide a purpose for the module assignment. If the column of the new module number remains empty, the respective line is lost when closing. Valid entries of the module numbers are between 0 and 99!

Entering the instance

POU can be entered either manually or via drag&drop using the process variable browser.

- Manually: The complete instance name has to be specified for the POU, e.g. "Station_01.Drilling_Module1"
- Process variable browser: Drag the instance from the “Process Variables” window to the previously selected cell of the “POU/Instance” column

☞ Please ensure a correct spelling of the instance names. Instance names are not checked automatically.

Example:

ProVi messages

There is a function block (FB_Drilling) that controls a drill completely and also contains the diagnostic messages of the drill. The control should contain two modules, each controlling one drill. For both drills, instances of the same function block are used.

The program, in which both drills are declared, is called Station_01.

```
4   VAR
5     Drilling_Module1: FB_Drilling;
6     Drilling_Module2: FB_Drilling;
7
```

Fig. 4-24: Declaring two FB_Drilling instances

The ProVi messages (error and information) are programmed in the function block.

When programming, module number 1 was specified for these diagnostics (see "Module assignment syntax" on page 48). However, the diagnostics for the first drill is to be displayed in module 1 and the diagnostics for the second drill is to be displayed in module 2.

Module assignment editor [IndraLogicXlc1: Logic: Application]

	POU/Instance	Diagnostic Type	Message Type	Module number	New module number
▶	Station_01.Drillin_Module1	PROVI	Error	1	2
	Station_01.Drillin_Module1	PROVI	Note	1	2
*		All	All		

Fig. 4-25: Diagnostic module assignment

In the module assignment editor, module number 2 is assigned to all diagnostics from Drilling_Modul2; in this case, the two ProVi messages “Error” and “Note”.

If no entry is made in this editor, the original module number 1 is used.

Examples of module assignments:

Character string	Description
Station_01.Drilling_Module2.PROVI.Error.1=3	The ProVi error with the module number 1 of this instance is displayed in module 3
Station_01.Drilling_Module2.PROVI.Error=3	All ProVi errors in this instance are displayed in module 3
Station_01.Drilling_Module2.PROVI=3	All ProVi messages in this instance are displayed in module 3
Station_01.Drilling_Module2=5	All diagnostics in this POU instance are displayed in module 5. This also applies to all instances of the POUs declared in this POU
Station_01=5	All diagnostics in this program are displayed in module 5. This also applies to all instances of the POUs declared in this POU (in this example e.g. Drilling_Module1 and Drilling_Module2)

Tab. 4-3: Examples of the module number assignment

The last, i.e. the most accurate module assignment is always applied to an instance path.

The following assignments are assumed:

- Station_01.Drilling_Module2.PROVI.Error.1 =4
- Station_01.Drilling_Module2=3
- Station_01=2

The exemplary diagnostics is displayed in the following modules:

- Instance Station_01.Drilling_Module1

  ProVi note 1 in module 2

  ProVi note 1 in module 2

Programming

- Instance Station_01.Drilling_Module2
  ProVi note 1 in module 4
  ProVi note 1 in module 3

5 SFC types

5.1 General information

This chapter describes the input and output signals as well as the behavior of different SFC types.

Apart from the standard IEC SFC without operation modes and diagnostics currently SFC types "ILDSfc01" and "ILDSfc02" with operation modes and diagnostics are available.

5.2 SFC types ILDSfc01

SFC ILDSfc01

The SFC of type "ILDSfc01" is not supported anymore by the diagnostics under IndraLogic 2G.

5.3 SFC types ILDSfc02

SFC ILDSfc02

The SFC of the "ILDSfc02" type has the input and output signals listed in tab. 5-1 "Input control signals of the ILDSfc02 SFC" on page 51.

The group of input signals in this SFC can be accessed via the input variable "iSFC". It is an automatically declared entity with the data structure "ILDSfc02Input".

In contrast to the SFC type "ILDSfc01", the input "Enable" of the SFC type "ILDSfc02" is a static input signal for process enabling of the SFC (refer to the description of the “Enable” input in the following table).

Input	Type	Initial value	Description
Enable	BOOL	FALSE	SFC is actively executed when "Enable" is pending. If "Enable" is missing, the SFC is in stopped state
Start	BOOL	FALSE	A positive edge at "Start" starts the SFC processing, allows step switching at waiting transition in jog and semi-automatic mode or enables an operation mode change
Halt	BOOL	FALSE	All times except the transition delay are stopped. The actions continue to be processed, see "Signal-time diagram SFC type ILDSfc02" on page 54
Reset	BOOL	FALSE	Reset SFC and set initialization step, see "Signal-time diagram SFC type ILDSfc02" on page 54
Manual	BOOL	FALSE	SFC operation mode manual (automatic mode = no manual mode, no semi-automatic mode and no jog mode), see "Signal-time diagram SFC type ILDSfc02" on page 54
SemiAuto	BOOL	FALSE	SFC operation mode semi-automatic (automatic mode = no manual mode, no semi-automatic mode and no jog mode), see "Signal-time diagram SFC type ILDSfc02" on page 54
Jog	BOOL	FALSE	SFC operation mode jog (automatic mode = no manual mode, no semi-automatic and no jog mode), see "Signal-time diagram SFC type ILDSfc02" on page 54
Synchronize	BOOL	FALSE	Synchronizing SFC. A step is enabled depending on a fulfilled transition. The input has no effect on the manual mode, see "iSFC.Synchronize ILDSfc02" on page 62

SFC types

Input	Type	Initial value	Description
Home	BOOL	FALSE	Set SFC to the Init step and move to the branch marked as "Home" branch if a positive edge is pending at "Enable", see "Signal-time diagram SFC type ILDSfc02" on page 54
ClearError	BOOL	FALSE	SFC deletes pending errors, see "Signal-time diagram SFC type ILDSfc02" on page 54
SettingPulse	BOOL	FALSE	SFC generates an pulse to initialize independent of the operation mode, see "Signal-time diagram SFC type ILDSfc02" on page 54
ManualForcing	BOOL	FALSE	SFC sets the flag "ManualForcing" to TRUE when the manual mode is selected (e.g. used as acknowledgement button in manual mode), see "Signal-time diagram SFC type ILDSfc02" on page 54
StopOnError	BOOL	TRUE	SFC stops in case of time errors, FALSE = no stopping in case of time errors; the pending time error is deleted upon exiting the step, see "Signal-time diagram SFC type ILDSfc02" on page 54. Does not affect monitor errors or starting condition. These errors still stop the SFC
DisableHomeError	BOOL	FALSE	SFC does not require home position to be reached (SFCState.Home) in order to move to the "Auto" branch
EnableExecutingDiag	BOOL	TRUE	SFC reports actively processed steps to display in IndraWorks Operation
EnableWaitOnTransitionDiag	BOOL	TRUE	SFC reports in IndraWorks Operation when the SFC waits for advancing by the starting signal in jog or semi-automatic mode at fulfilled transition condition

Tab. 5-1: Input control signals of the ILDSfc02 SFC

The group of output signals in this SFC can be reached via the output variable `qSFC`. It is an automatically declared instance of the data structure "ILDSfc02Output".

Output	Type	Description
InOperation	BOOL	SFC is enabled (not in the Init step; manual mode "iSFC.Manual" not selected)
Run	BOOL	SFC is processing (not in the Init step) and "Enable" is present. Actions are actively processed
Halt	BOOL	SFC has stopped
Reset	BOOL	iSFC.Reset is controlled and SFC is in the Init step
Manual	BOOL	SFC manual mode is active
SemiAuto	BOOL	SFC is in semi-automatic mode
Jog	BOOL	SFC is in jog mode
Home	BOOL	SFC homing branch is processed
HomeOk	BOOL	SFC is in home position ("ILDSfc02MachineState.HomePosition"=TRUE)
ManualForced	BOOL	SFC was actively executed in manual mode, i.e. steps or actions were operated manually
StartConditionOk	BOOL	SFC starting conditions are met ("ILDSfc02MachineState.StartCondition"=TRUE)
WaitOnTransition	BOOL	SFC waits at fulfilled transition for edge at "Enable" in jog and semi-automatic mode
Error	BOOL	SFC reports error at the "Message" output
Info	BOOL	SFC reports information at the "Message" output

Output	Type	Description
Startup	BOOL	SFC reports missing starting condition at the "Message" output
Setup	BOOL	SFC reports missing starting conditions at the "Message" output. The output is TRUE • if the SFC is in stop or in the INIT step ("qSFC.RUN"=FALSE) and the SFCMachineState "starting condition" is set to FALSE and/or • if the SFC in the INIT step and the "HomePosition" is set to FALSE The evaluation is carried out irrespective of the "iSFC.Enable" state
Message	STRING [255]	SFC message text

Tab. 5-2: Output status signals of the ILDSfc02 SFC

Further status signals of the SFC for internal use (read-only) are "SFCState" and "SFCMaschineState". These status signals are only valid for the respective SFC.

Variables of SFCState	Type	Description
Enable	BOOL	SFC - "Enable" is present
Started	BOOL	SFC - Rising edge at "iSFC.Start" is present
Halt	BOOL	SFC - Stop is present
Home	BOOL	SFC - Homing request is present
SettingPulse	BOOL	SFC - Initializing pulse: This pulse can be used to trigger a trailing edge in the SFC again for example
Synchronize	BOOL	SFC - Synchronization request was executed
ManualForcing	BOOL	SFC - Manual forcing present in manual mode

Tab. 5-3: Internal SFCState variables of the ILDSfc02 SFC

The signals and variables listed in the following table are only valid for the respective SFC and have to be defined in the cyclically called action "aUserIn" for the criteria analysis of "StartCondition" and "Homing". The status signals for the module and the machine as a whole have to be declared and assigned globally (see chapter 4.7 "HomePosition & StartCondition" on page 41).

Variables of the SFCMachineState	Type	Initial value	Description
HomePosition	BOOL	TRUE	SFC - HomePosition reached. The signals required have to be defined by the user
StartCondition	BOOL	TRUE	SFC StartCondition is met. The signals required have to be defined by the user
EnMotionSignal	BOOL	TRUE	SFC - Enabling of motion signal

Tab. 5-4: Internal SFCMachineState variables of the ILDSfc02 SFC

The general behavior and programming of SFCs with operation modes are described in chapter 4 "Programming" on page 29.

SFC types

Signal-time diagram SFC type ILDSfc02

Signal-time diagram:

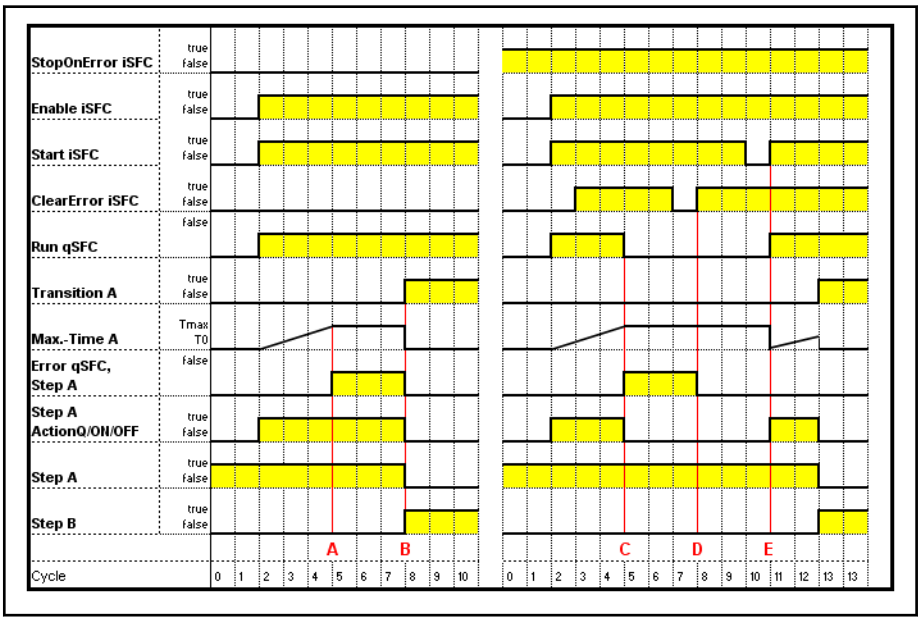


Fig. 5-1: Error case in SFC

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	Error, e.g. missing transition condition when exceeding the maximum step time	qSFC.Error	SFC remains in active processing
B	Jog Semi Auto	Transition condition met: 0-1	qSFC.Error	SFC switches from step A to step B
C	Jog Semi Auto	Error, e.g. missing transition condition when exceeding the maximum step time	qSFC.Error qSFC.Run ActionQ/ON/OFF	SFC - processing set to inactive; post-processing of the action carried out upon stop
D	Jog Semi Auto	Positive edge iSFC.ClearError: 0-1	qSFC.Error	Acknowledge with positive edge
E	Jog Semi Auto	iSFC.Start: 0-1	qSFC.Run ActionQ/ON/OFF	There has to be one PLC cycle between "iSFC.ClearError" and "iSFC.Start"

Tab. 5-5: Table for signal-time diagram: Error case in SFC

Signal-time diagram:

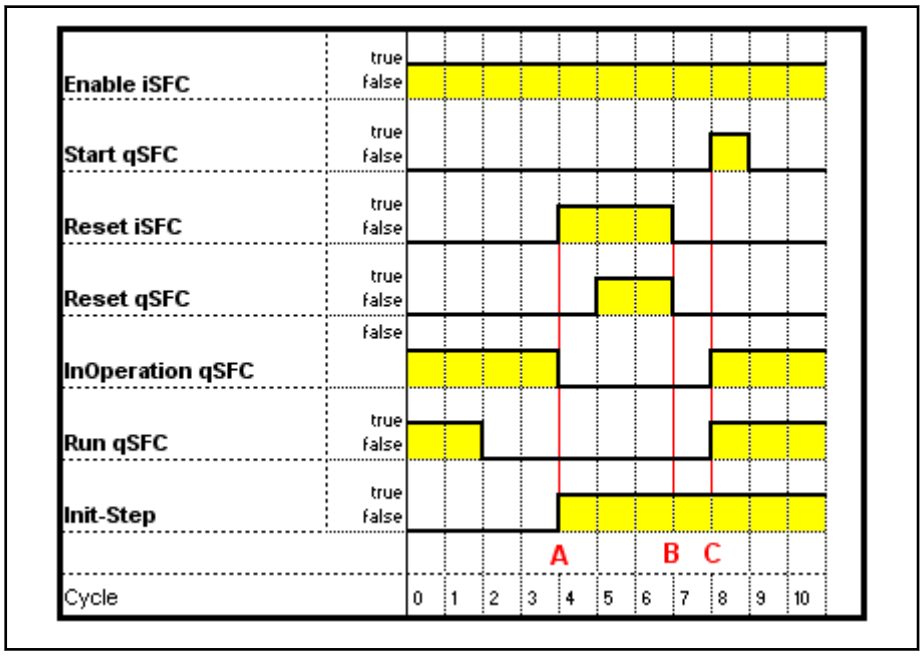


Fig. 5-2: Reset of SFC

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	iSFC.Reset: 0-1	qSFC.InOperation	There has to be at least one PLC cycle between "iSFC.Reset" 0-1 and "iSFC.Start" 1-0
B	Jog Semi Auto	iSFC.Reset: 1-0		
C	Jog Semi Auto	Positive edge iSFC.Start: 0-1	qSFC.Run qSFC.InOperation	There has to be at least one PLC cycle between "iSFC.Reset" 1-0 and "iSFC.Start" 0-1

Tab. 5-6: Table for signal-time diagram: Reset of SFC

SFC types

Signal-time diagram:

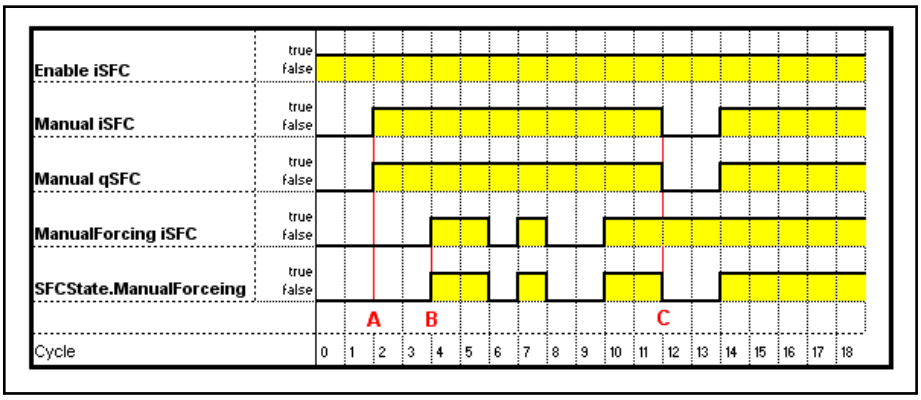


Fig. 5-3: ManualForcing of SFC

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	iSFC.Manual: 0-1	qSFC.Manual	SFC in manual mode
B	Jog Semi Auto	iSFC.ManualForcing: 0-1	SFCState.ManualForcing	The signal change "iSFC.ManualForcing" 1-0 can be carried out in the same cycle as the signal change "iSFC.Manual" 1-0
C	Jog Semi Auto	iSFC.Manual: 1-0	SFCState.ManualForcing	

Tab. 5-7: Table for signal-time diagram: ManualForcing of SFC

Signal-time diagram:

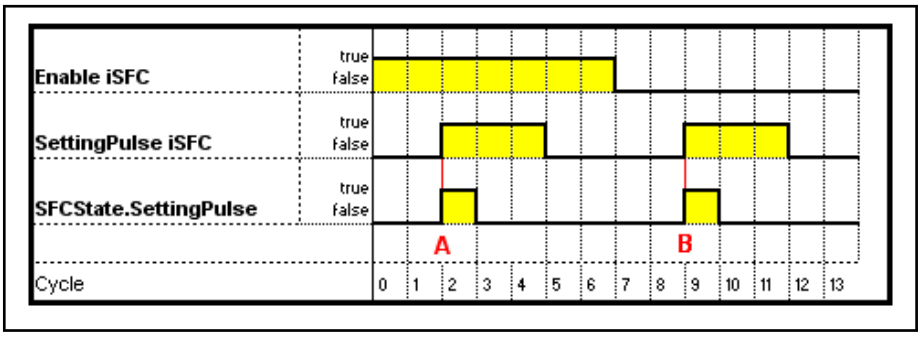


Fig. 5-4: SettingPulse of SFC

Event	Mode	Cause	Effect on	Note
A	Jog Semi Auto	iSFC.SettingPulse: 0-1	SFCState.SettingPulse	Pulse for one cycle
B	Jog Semi Auto	iSFC.SettingPulse: 0-1	SFCState.SettingPulse	Pulse for one cycle

Tab. 5-8: Table for signal-time diagram: SettingPulse of SFC

Signal-time diagram:

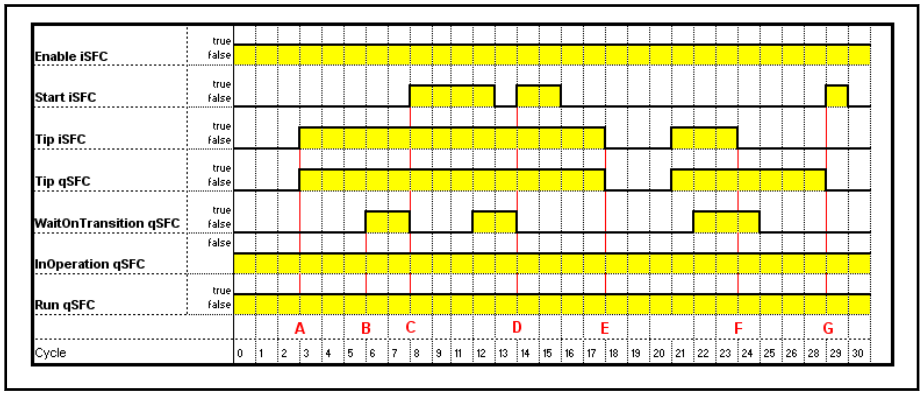


Fig. 5-5: Operation mode change Auto-Jog of the SFC

Event	Mode	Cause	Effect on	Note
A	Auto to jog	iSFC.Tip: 0-1	qSFC.Tip	SFC remains in active processing
B	Jog	Transition condition met: 0-1	qSFC.WaitOnTransition	SFC waiting for positive edge "iSFC.Start"
C+D	Jog	Pos. edge iSFC.Start	qSFC.WaitOnTransition	
E	Jog to auto	iSFC.Tip: 1-0	qSFC .Tip	Operation mode change to AUTO

SFC types

Event	Mode	Cause	Effect on	Note
F	Jog to auto	iSFC.Tip: 1-0		When the condition "qSFC.WaitOnTransition" is present, direct switching to AUTO not possible: "qSFC.Tip"=1
G	Auto	iSFC.Start: 0-1		To change the operation mode, a positive edge is required at "iSFC.Start". The signal "iSFC.Jog" has to change to FALSE at the start of the same cycle or before

Tab. 5-9: Table for signal-time diagram: Operation mode change Auto-Jog of the SFC

Signal-time diagram:

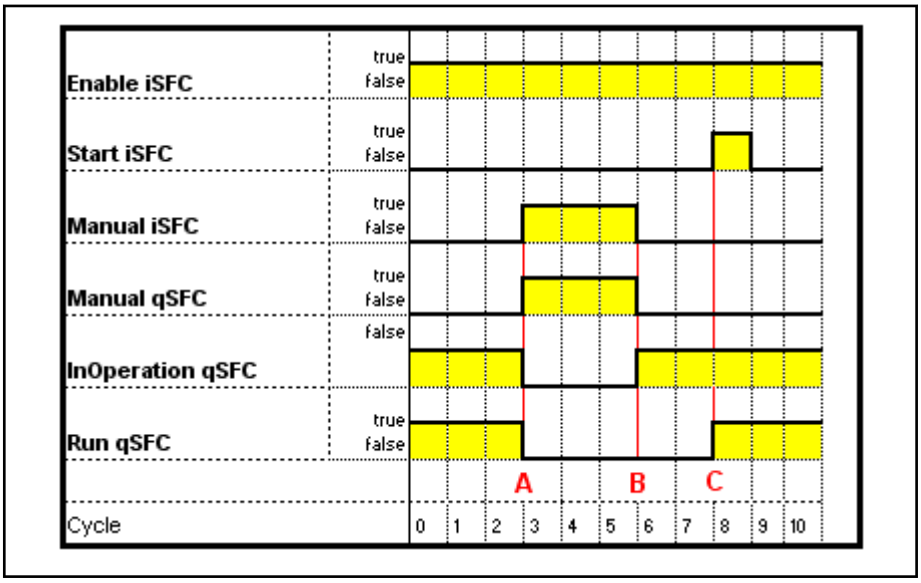


Fig. 5-6: Operation mode change Auto-Manual of SFC

During active SFC processing in manual mode and assignment of "TRUE" to the flag "ActionName.F" or "StepName.F", the SFC outputs "qSFC.InOperation" and "qSFC.Run" change to TRUE.

Event	Mode	Cause	Effect on	Note
A	Auto to Manual	iSFC.Manual: 0-1	qSFC.Manual qSFC.InOperation qSFC.Run	Selecting manual mode
B	Manual	iSFC.Manual: 1-0	qSFC.InOperation qSFC.Manual	Deselecting manual mode
C	Manual to Auto	Pos. edge iSFC.Start	qSFC.Run	

Tab. 5-10: Table for signal-time diagram: Operation mode change Auto-Manual of SFC

Signal-time diagram:

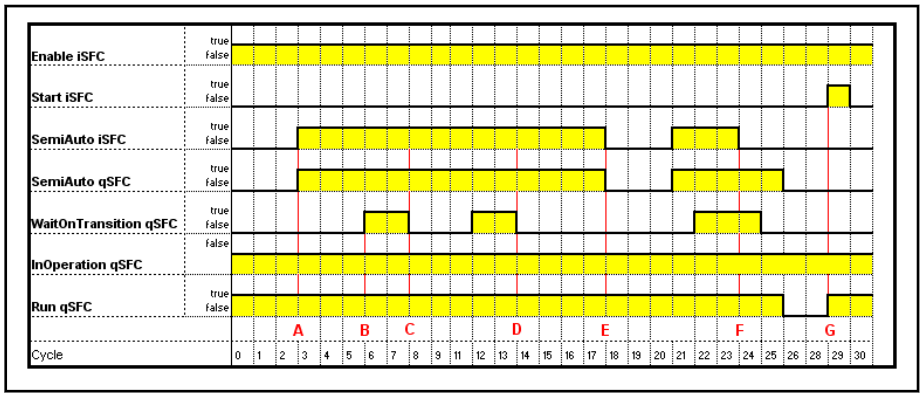


Fig. 5-7: Operation mode change Auto-SemiAuto of SFC

Event	Mode	Cause	Effect on	Note
A	Auto to Semi-Auto	iSFC.SemiAuto: 0-1	qSFC.SemiAuto	SFC remains in active processing
B	Semi-Auto	Transition condition met: 0-1	qSFC.WaitOn-Transition	SFC waiting for positive edge "iSFC.Start"
C+D	Semi-Auto	Pos. edge iSFC.Start	qSFC.WaitOn-Transition	
E	Semi-Auto to Auto	iSFC.SemiAuto: 1-0	qSFC.SemiAuto	Operation mode change to AUTO

SFC types

Event	Mode	Cause	Effect on	Note
F	Semi-Auto to Auto	iSFC.SemiAuto: 1-0		When the condition "qSFC.WaitOnTransition" is present, direct switching to Auto not possible: "qSFC.SemiAuto": 1-0
G	Auto	iSFC.Start: 0-1	qSFC.Run	To change to the operation mode AUTO, a positive edge is required at "iSFC.Start"

Tab. 5-11: Table for signal-time diagram: Operation mode change Auto-SemiAuto of SFC

Signal-time diagram:

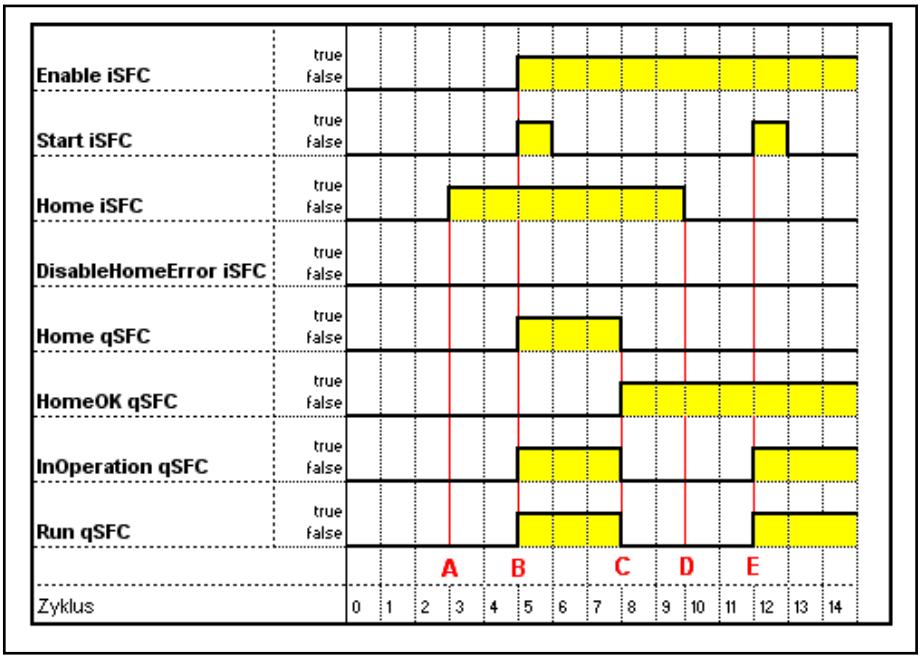


Fig. 5-8: Homing of SFC

Event	Mode	Cause	Effect on	Note
A		iSFC.Home: 0-1		Request homing
B	Homing	Transition condition for homing branch met, iSFC.Enable: 0-1	qSFC.Run qSFC.InOperation qSFC.Home	SFC jumps to homing branch
C	Homing	HomeCondition met	qSFC.Run qSFC.InOperation qSFC.HomeOK	Jump back to Init step

Event	Mode	Cause	Effect on	Note
D		iSFC.Home: 1-0		Deselect homing request
E	to Auto	iSFC.Start: 1-0	qSFC.Run qSFC.InOperation	

Tab. 5-12: Table for signal-time diagram: Homing of SFC

Signal-time diagram:

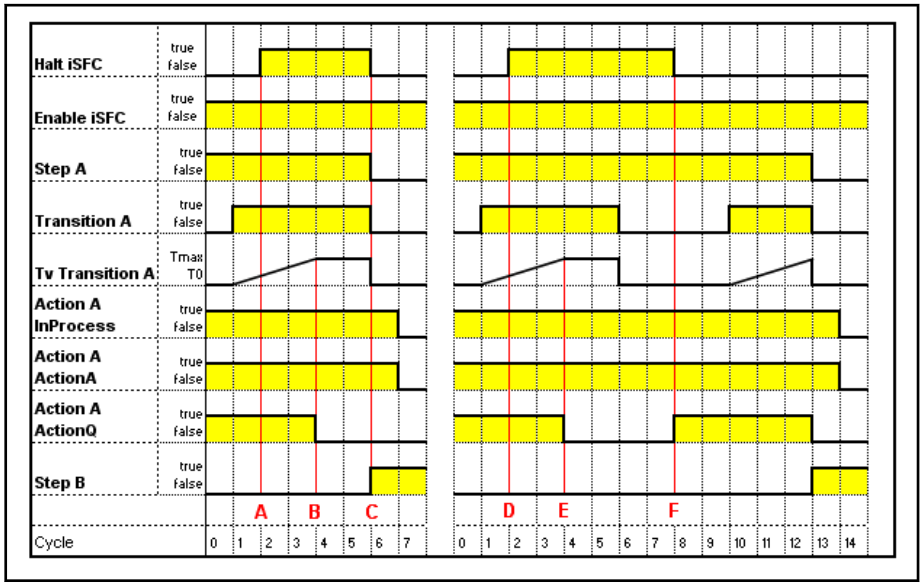


Fig. 5-9: Stop with transition delay of SFC

Event	Mode	Cause	Effect on	Note
A+D	Jog Semi Auto	iSFC.Halt: 0-1		SFC stopped, actions continue to be processed
B+E	Jog Semi Auto	Transition condition met, delay time elapsed	ActionQ	SFC stopped, actions continue to be processed
C	Jog Semi Auto	iSFC.Halt: 1-0		Advancing to step B
F	Jog Semi Auto	iSFC.Halt: 1-0	ActionQ	"ActionQ" is selected again only after "iSFC.Halt" has been deselected when the transition had already been met once at "iSFC.Halt"=TRUE

Tab. 5-13: Stop with transition delay of SFC

SFC types

iSFC.Synchronize ILDSfc02

This input of the SFC with diagnostics sets a step depending on **one** fulfilled transition.

Operation mode Automatic: With "iSFC.Synchronize"=TRUE and "iSFC.Enable"=TRUE, a positive edge at "iSFC.Start" causes the step to be set **after the fulfilled transition** and starts the execution of the SFC when "iSFC.Enable"=TRUE is present.

Operation mode Jog: With "iSFC.Synchronize"=TRUE and "iSFC.Enable"=TRUE, a positive edge at "iSFC.Enable" causes the step to be set **before the fulfilled transition**. The actions are processed for one cycle and then post-processed. The output "qSFC.WaitOnTransition" is set.

Operation mode SemiAuto: With "iSFC.Synchronize"=TRUE and "iSFC.Enable"=TRUE, a positive edge at "iSFC.Start" causes the step to be set **before the fulfilled transition** when the transition has the additional property "Stop in Semiautomatic". The actions are processed for one cycle and then post-processed. The output "qSFC.WaitOnTransition" is set. If the fulfilled transition is not provided with the additional property "stop in semi-automatic", the behavior is identical to the one in automatic mode.

Successful synchronization causes the variable "SFCState.Synchronize" to be "TRUE" for one cycle.

- The fulfilled transition conditions "NOT FALSE" and "TRUE" are not considered here
- When synchronizing, **only one** transition condition of the SFC can be TRUE. In case of multiple valid transition conditions, it is synchronized on an undefined transition
- Transitions that were formed in actions with "ActionTrans" cannot be synchronized. The qualifiers ON/OFF/GO are an exception, their actions are executed cyclically and consequently an assignment to the action transition is processed
- If no transition is met the "Init" step is set

6 SFC diagnostics

6.1 General information

SFCs of the type ILDSfc support automatically generated types of diagnostics. Additional message types can be triggered directly in the SFC program.

6.2 Diagnostic overview in IndraWorks Operation

Diagnostic overview

The diagnostic overview in IndraWorks Operation lists the pending messages as shown in the following example:

- Number: ProVi error number
- CA: Criteria analysis
- Date: Date when the message occurred
- Time: Time when the message occurred
- Description: Describes the message

The message highlighted in blue was selected using the mouse or the cursor keys. For this message, additional information is displayed in the lower section of the window.

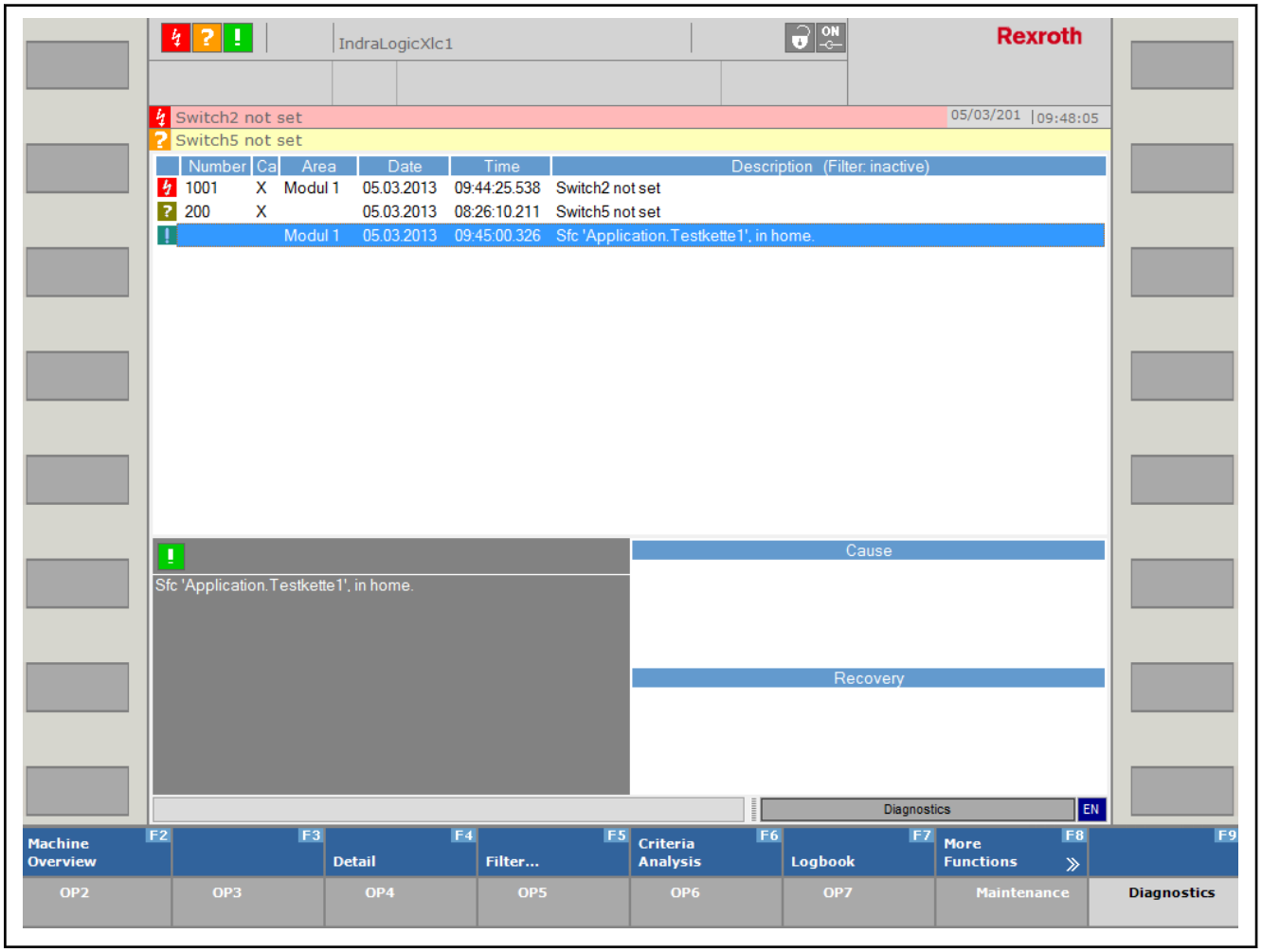


Fig. 6-1: Diagnostic overview

Detailed view

Press <F4> "Details" for further information on this message.

SFC diagnostics

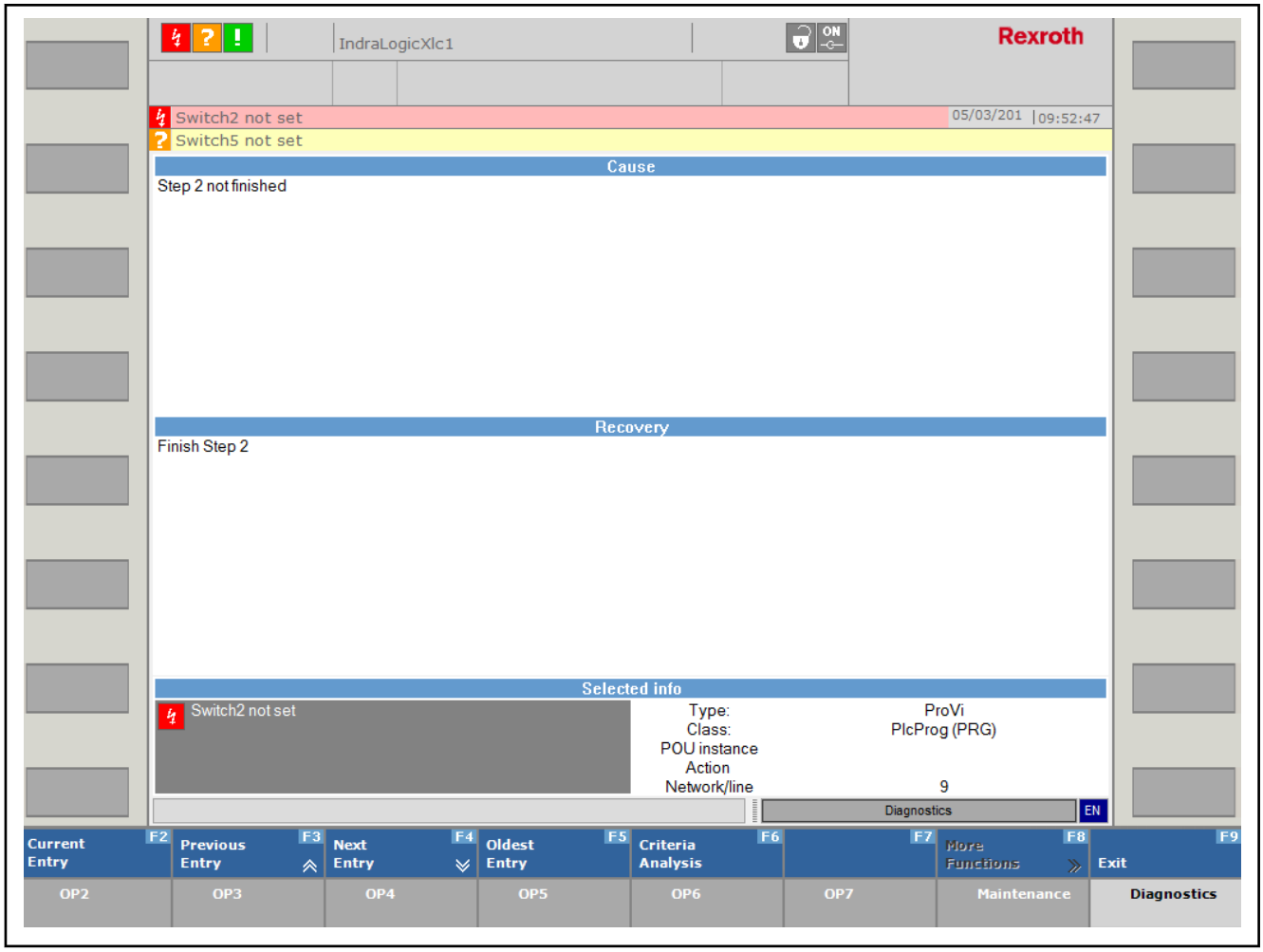


Fig. 6-2: Detailed view

The following additional information is shown in the "Selected info" window:

- Type: There are four different types of diagnostics
  1. MTX
  2. ProVi
  3. SFC messages:
     - Active steps
     - Home position reached
     - Homing
     - Missing home position
     - Monitor error
     - Time error
     - Missing starting condition
     - Wait for transition
  4. MLC
- Class: Name of the **P**rogram **O**rganization **U**nit in which the message was programmed. The type of the POU is given behind it in brackets (PRG = program, FM = function module)

- POU instance: Instance name of the POU in which the message was programmed
- Action: In case of time errors, the name of the step in which the error occurs is indicated. The identifier changes from "Action" to "Step". For all other message types, the action causing the message is indicated
- Network/line: In case of monitor errors, the line in the action causing the error is indicated. Otherwise, this line contains the network number

Criteria analysis

In detailed view, press <F6> "Criteria analysis" to switch to the criteria analysis.

The criteria causing the error are identified for the selected error. These are displayed in IL view in a new window. A reduced view is selected when opening the criteria analysis window. Reduced means that only those criteria that caused the error are displayed.

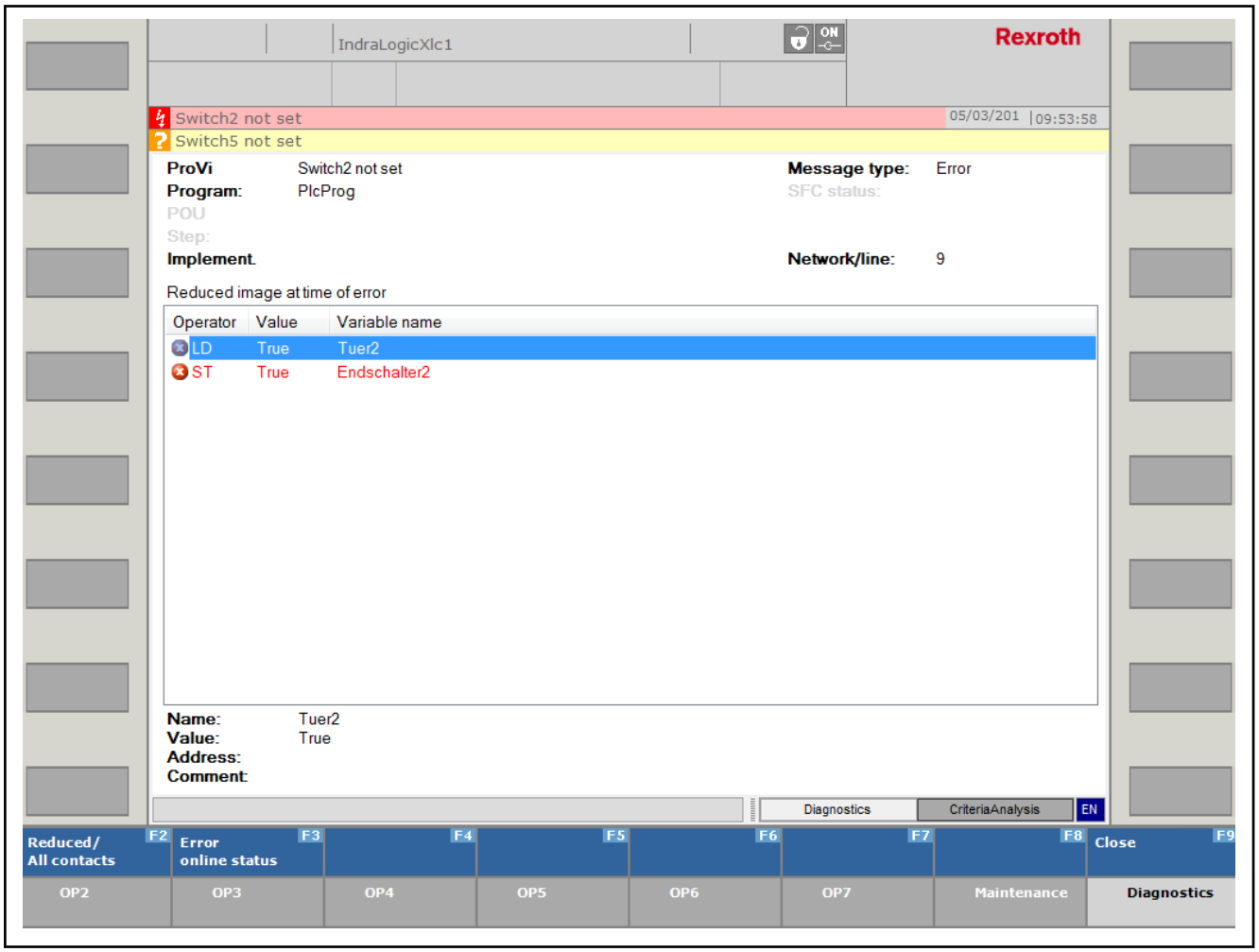


Fig. 6-3: Criteria analysis window - Reduced view

The criteria analysis window is divided into three sections:

- The upper part displays basic information on the message already available in the detailed view. This facilitates keeping an overview on the message the criteria analysis belongs to
- The centered section displays the relevant criteria (variables) of the pending message.

SFC diagnostics

The operator is displayed in IL syntax. The state of the variables can be TRUE or FALSE. The state when the message occurs is shown (snapshot).

Press <F2> "Reduced/All contacts" to switch between the reduced view (displays only relevant network criteria) and full view (displays all network criteria).

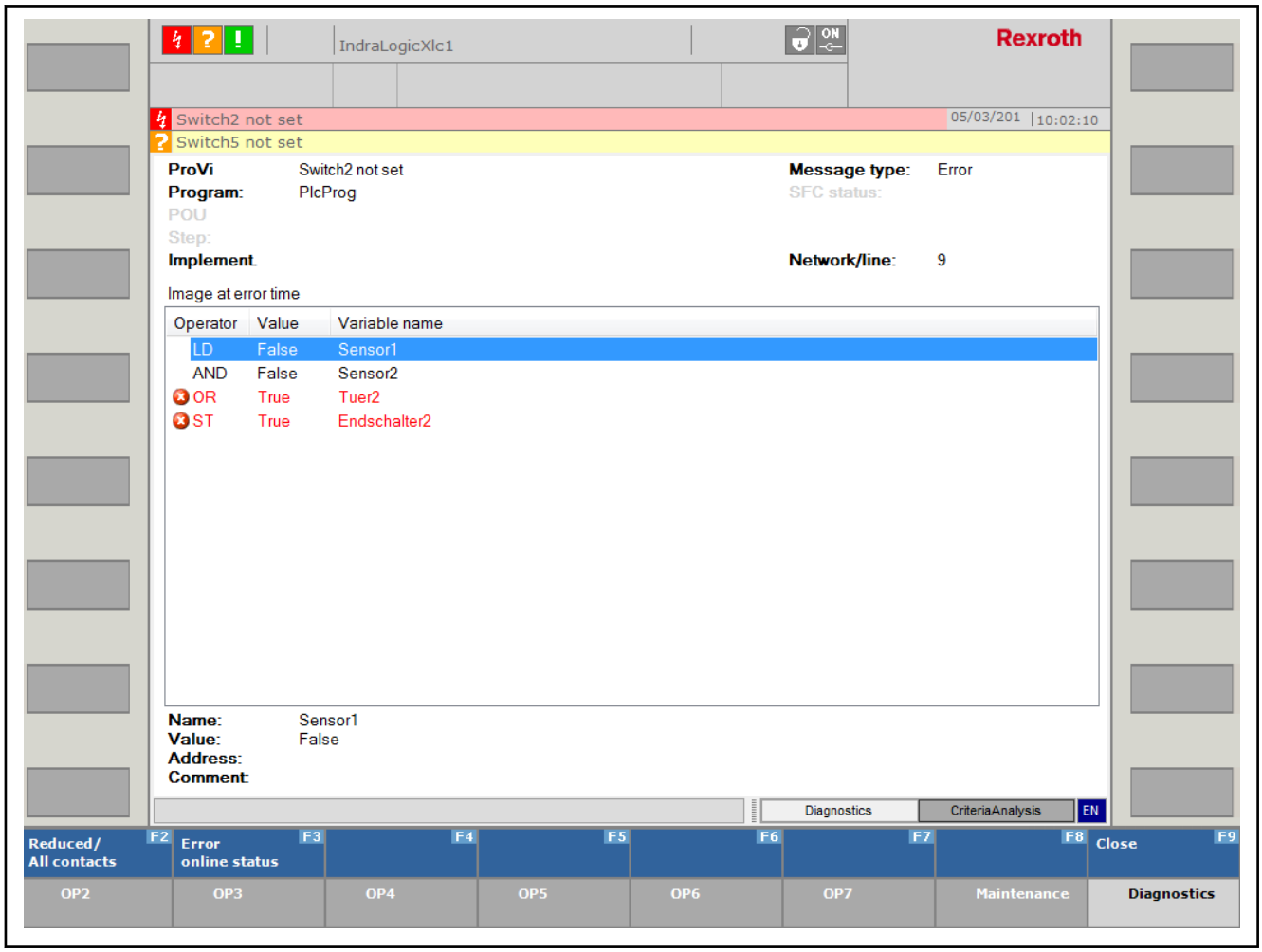


Fig. 6-4: Criteria analysis window - Full view

- The lower section displays additional information on the variables used. This additional information refers to the highlighted line in the centered section of the criteria analysis window and is obtained from the variable declaration.

  In this view, relevant criteria are highlighted in red and provided with an icon. The icon emphasizes the relevant lines

Exit the criteria analysis window with <F9> "Close".

Additional information on the message types "Home position" and "Starting condition":

For criteria analysis of the home position and starting condition messages, the online status is always displayed. That means that the variable state is updated on the display when it changes.

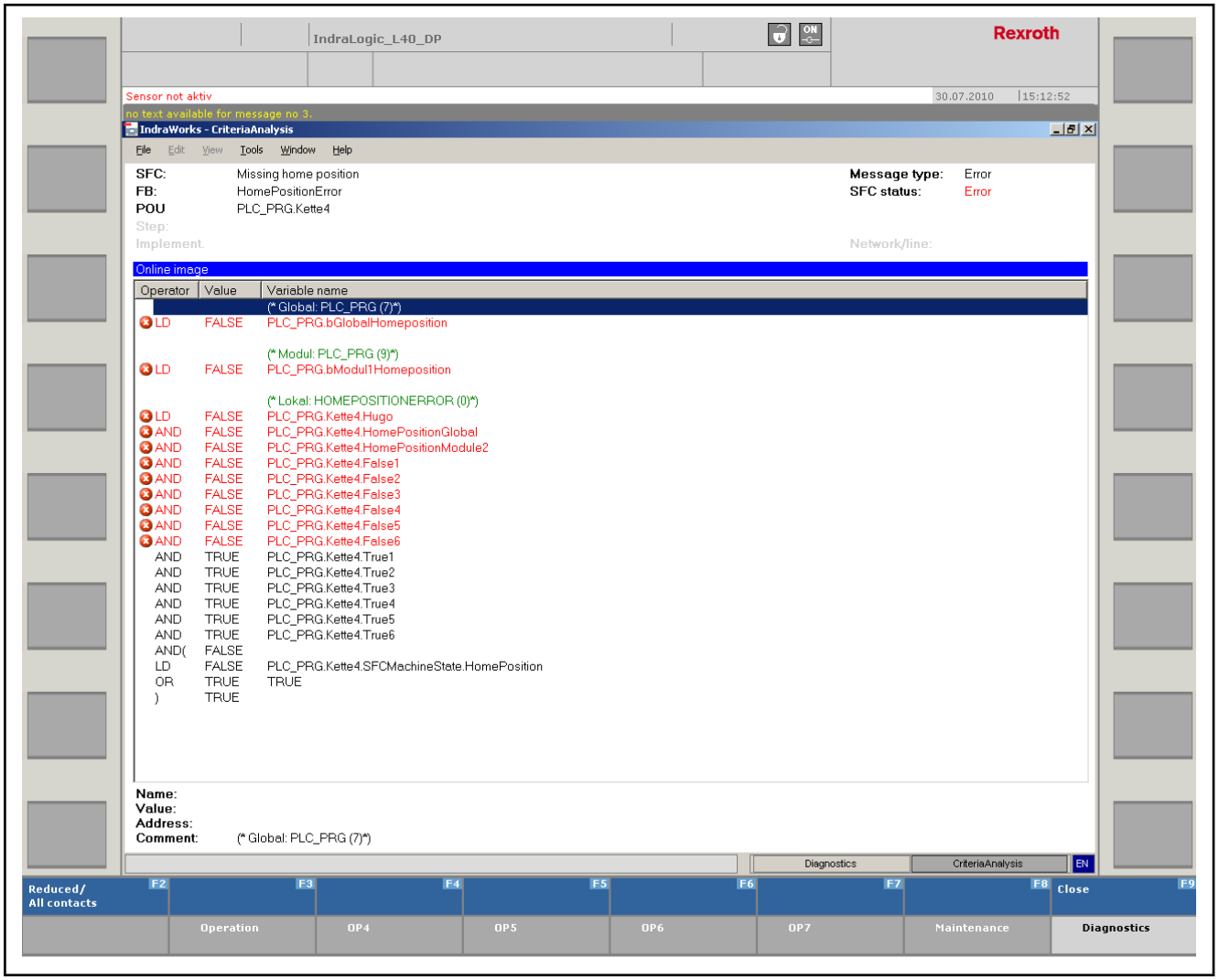


Fig. 6-5: Criteria analysis window - Display of causes

For easier identification purposes, a blue bar is displayed above the criteria analysis window during online status.

Home position and start condition errors can be defined locally, globally or related to a module. The cause for a home position/start condition error can be local and/or module-related and/or global.

The criteria analysis displays all causes. It displays the global cause first, then the modular and finally the local cause. These causes are displayed one below the other. In full view, every cause is preceded by a header with the following structure:

Type: POU name (line number)

Type = Global, module, local

POU name = Name of the POU in which the message was triggered

(line number) = Line number in which the message was triggered

Replacing the scratch flag

Scratch flags are partial results in logic operations that are used again in subsequent networks. Thus, the PLC program is more clearly structured and easier to maintain. Often, many partial results are merged again in a separate network at a later point.

The criteria analysis calculation has previously not considered the scratch flags. Only the flags directly in the network line were taken into consideration for the analysis.

Now, also those flags composed of the scratch flags are considered during the analysis.

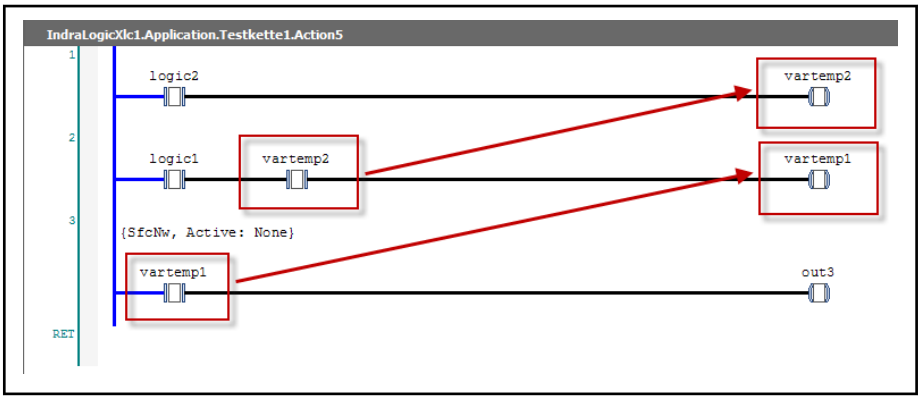


Fig. 6-6: Scratch flags in the network

Network2 causes a monitor error if the state of the assignment is "False".

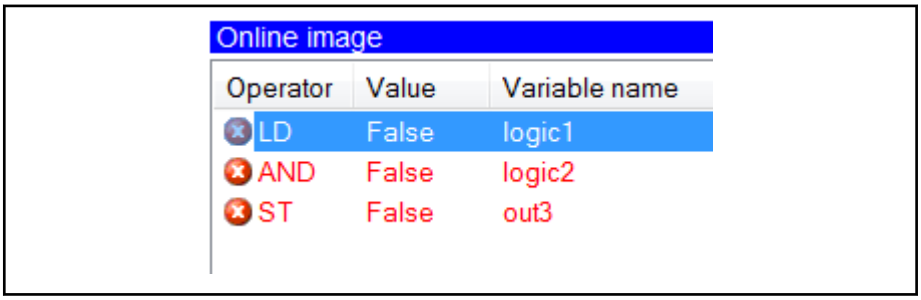


Fig. 6-7: Criteria analysis scratch flag

The scratch flag causing the monitor error was replaced by the logic generating the scratch flag. The states of this logic caused the monitor error. The nesting depth of the scratch flag is not limited. The scratch flags are only determined within the POU, but not beyond the POU limits.

The scratch flags are always automatically replaced. If no scratch flags are available, the criteria analysis corresponds to the previous analysis. Replacing these scratch flags is identically calculated as for the ProVi messages and for the diagnostics of non-executable operating screens

The scratch flags are replaced in frozen state as well as during the online criteria analysis.

6.3 Diagnostic overview in IndraWorks Engineering

Diagnostic overview

SFC messages can be displayed in IndraWorks Engineering. This facilitates commissioning and optimization on the PLC program without using an HMI.

The conditions are:

1. A message is output by the SFC
2. The control is switched to "Online" in IndraWorks Engineering
3. The device diagnostics is switched on in IndraWorks Engineering under **Diagnostics ▸ Device diagnostics**

The diagnostic overview in IndraWorks Engineering lists the pending messages as shown in the following example:

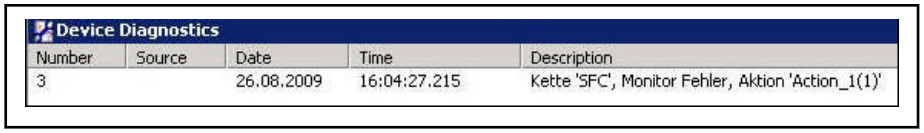


Fig. 6-8: List of pending messages

7 Operating screens

Operating screens can be freely configured. They are used to manually operate machines. Machine motions can be carried out via the machine function keys - short: M-keys - on the left and right side on the screen of the visualization device. The current position of the moving axis as well as the states of freely configurable PLC variables can be displayed. The meaning of individual states is determined while configuring the operating screen.

The executability of a machine function - also referred to as motion signal - can be displayed with the corresponding status display.

Configuring operating screens

The operating screens are configured in IndraWorks Engineering. New operating screens are created in the node of the selected visualization device. These operating screens contain one or more operating lines. Two motions can be configured for each operating line and then activated using the corresponding right or left M-key. Behind every motion there is an action or a step with several actions in which the motion is programmed.

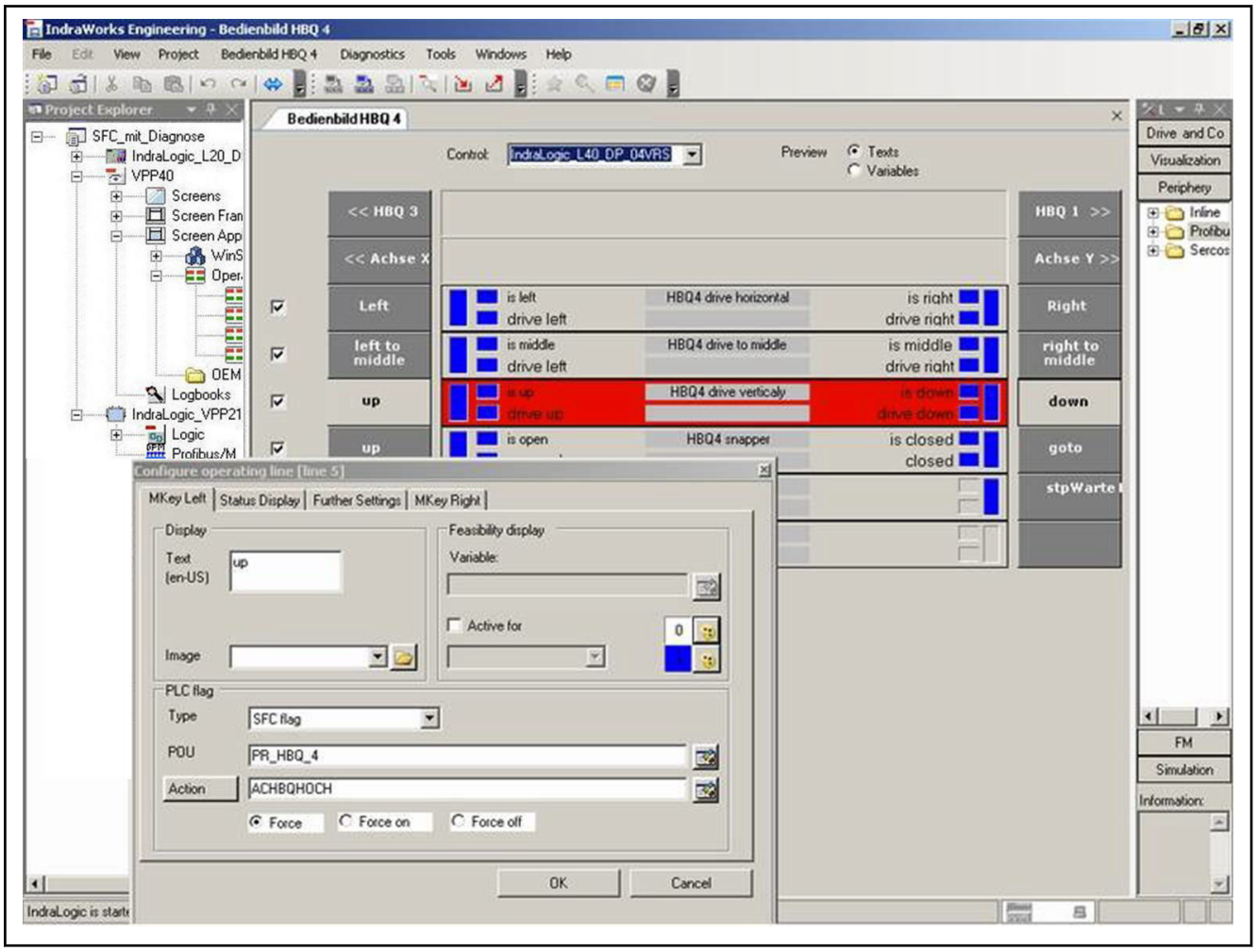


Fig. 7-1: Configuration of a line from the motion screen

The previous example shows the configuration of a line from the operating screen. The action "AXISBQUP" was assigned to the M-key <UP>. At runtime, this action state is analyzed and it is calculated whether this action is "executable". "Executable" means that at least one network whose logic is fulfilled exists within the networks of this action.

Operating screens

The executability display can be set for the PLC flag type "SFC flags" as well as for the type "free flags". For free flags, a variable to display the executability can be specified.

Pressing the **Action** key switches to the option "Step".

When executing an action, it has to be selected whether this action is called by default ("Force") as processing with "ON-qualifier property" ("Force on") or "OFF-qualifier property" ("Force off").

When using "ON"- and "OFF-" qualifier properties, the "Action" option has to be selected. With the "Step" option, the calculation of the motion signal and the corresponding diagnostics are not guaranteed.

No optional variable can be assigned for SFC flags. The diagnostic magnifier can be activated for "All modes", "Manual mode", "SFC" and "None" (see "EnMotionSignal" on page 75).

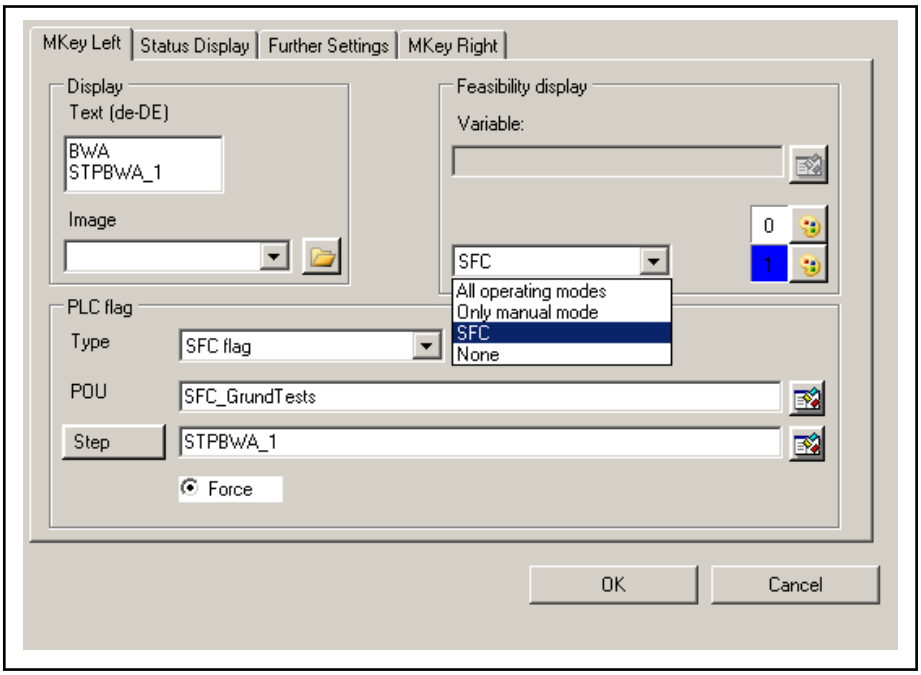


Fig. 7-2: Setting executability display for SFC flags

In the operating screen line, the executability is displayed as a vertical bar in color. The default colors are "blue" (executable) and "white" (non-executable) and can be configured by the user.

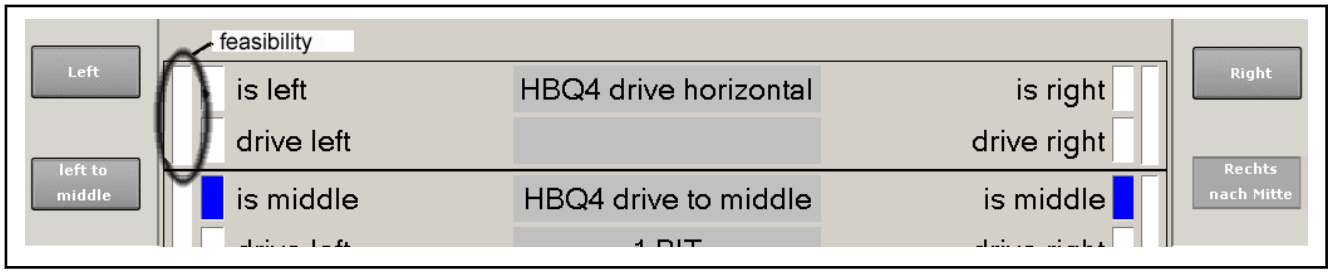


Fig. 7-3: Illustration of executability

The previous example shows an operating screen line with executability display. In this example, the executability is specified. Pressing the M-key <Left> executes the action or step linked with this step.

- Even when executability is **not** given, this still has to be locked in the implementation, as otherwise undesired machine motion is possible
- **Non-calculable executability is always displayed as TRUE** irrespective of its actual executability. This can be caused by e.g. deactivating the motion signal or implementation of non-supported variables (see chapter 4.2 "Definition of the supported variables for the ProVi SFC diagnostics" on page 29)

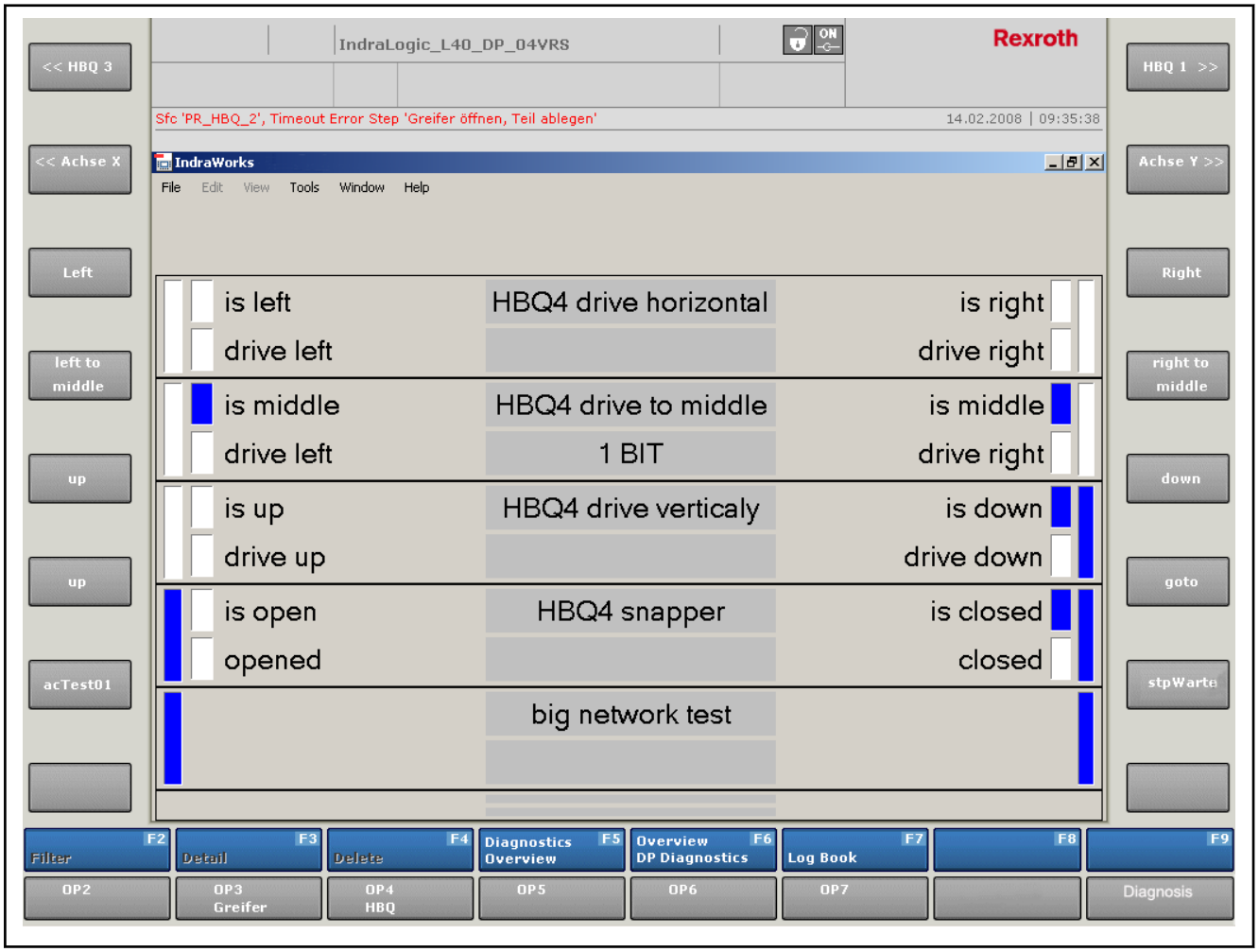


Fig. 7-4: Operating screen in the IndraWorks Operation Desktop

The previous example shows several operating screen lines. Apart from the centered operating line "Move HBQ4 vertically" (left M-key <Up>), all motions are executable.

Incorrectly configured M-key

Incorrectly configured M-keys are directly displayed on the button with the following symbol.

Fig. 7-5: Incorrectly configured M-key

This error type is for example caused by a step/action that does not exist anymore. In order to avoid incorrectly configured M-keys, use the "Validate" function in the context menu of the visualization device.

Operating screens

Diagnostic magnifier

If motions are not executable or if the user releases the corresponding M-key after pressing it, the "diagnostic magnifier" opens. This window contains detailed information for the user about the non-executable action (or step).

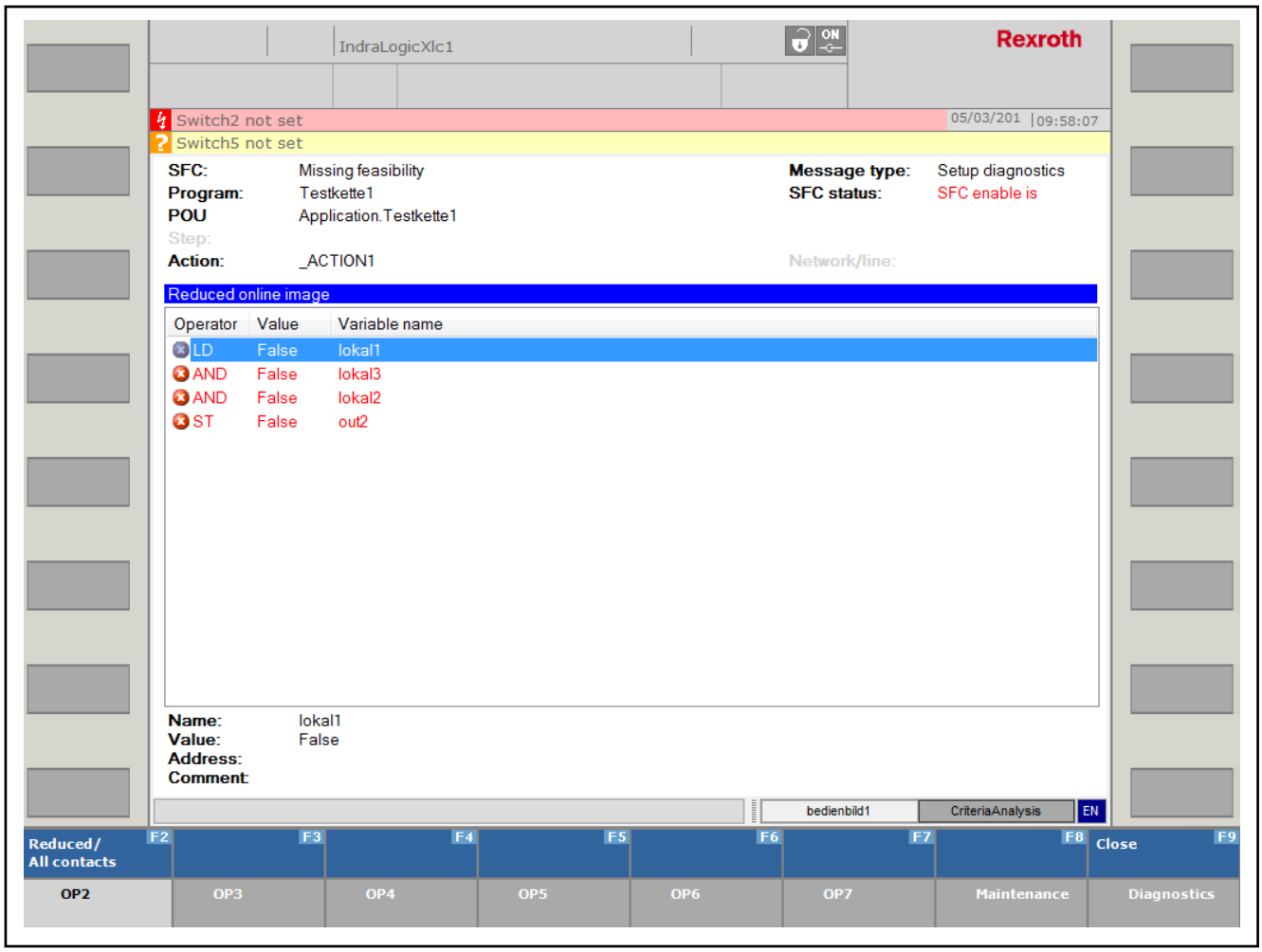


Fig. 7-6: Display of the diagnostic magnifier from an operating screen

The three-section structure of the diagnostic magnifier corresponds to the diagnostic display for the criteria analysis:

- The upper section shows basic information on the action for which the analysis is shown
- The centered section shows the relevant criteria (variables) of the action which prevent execution of the action.

  The operator is displayed in IL syntax. The state of the variables can be TRUE or FALSE. The current state is shown.

  Pressing <F2> "Reduced/All contacts" switches between reduced and full display. The full view shows all lines which were not excluded from the analysis by the corresponding pragma (see "Monitor errors" on page 46). The relevant lines are marked by a preceding icon.

  Note: By default, all networks are excluded from the executability display. To activate the relevant networks for executability, they have to be explicitly configured in IndraLogic using the dialog Diagnostics -> SFC - Network Editor

- The lower section displays additional information on the variables used. This additional information refers to the highlighted line in the centered section of the criteria analysis window.

  Exit the criteria analysis window with <F9> "Close"

EnMotionSignal

The state of "EnMotionSignal" is only relevant in the "SFC" mode. In "SFC" mode, the signal "EnMotionSignal=TRUE/FALSE" enables/disables the diagnostic magnifier:

Operation mode	EnMotionSignal	Diagnostic magnifier
SFC	TRUE(Default)	Appears
SFC	FALSE	Does not appear
All operation modes	Any	Appears
Manual mode	Any	Appears only in manual mode
None	Any	Does not appear

Tab. 7-1: Overview on EnMotionSignal

8 Application example

8.1 General information

This chapter describes a use case for an SFC with diagnostics and operation modes. In this example, a workpiece is to be moved on a rectangular path (from bottom to right, from right to top, from top to left and from left to bottom). Proceed as follows:

Creating a new project

Create a new project in IndraWorks and the desired control.

Enabling diagnostics

Enabling the ProVi SFC diagnostics in IndraWorks in the menu under **Diagnostics ▸ ProVi SFC diagnostics**.

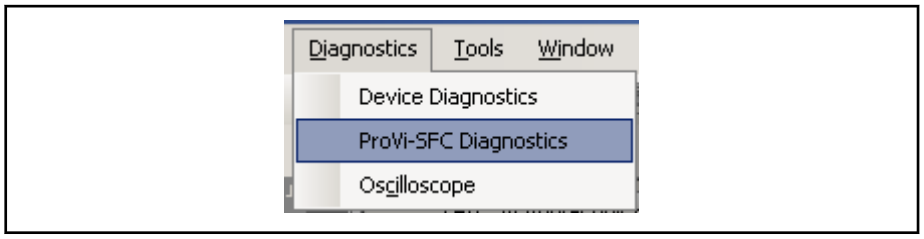


Fig. 8-1: Enabling diagnostic messages

Creating an SFC function block

Select a new function block under **Application ▸ Add ▸ POU...** and configure it in sequential function chart.

Application example

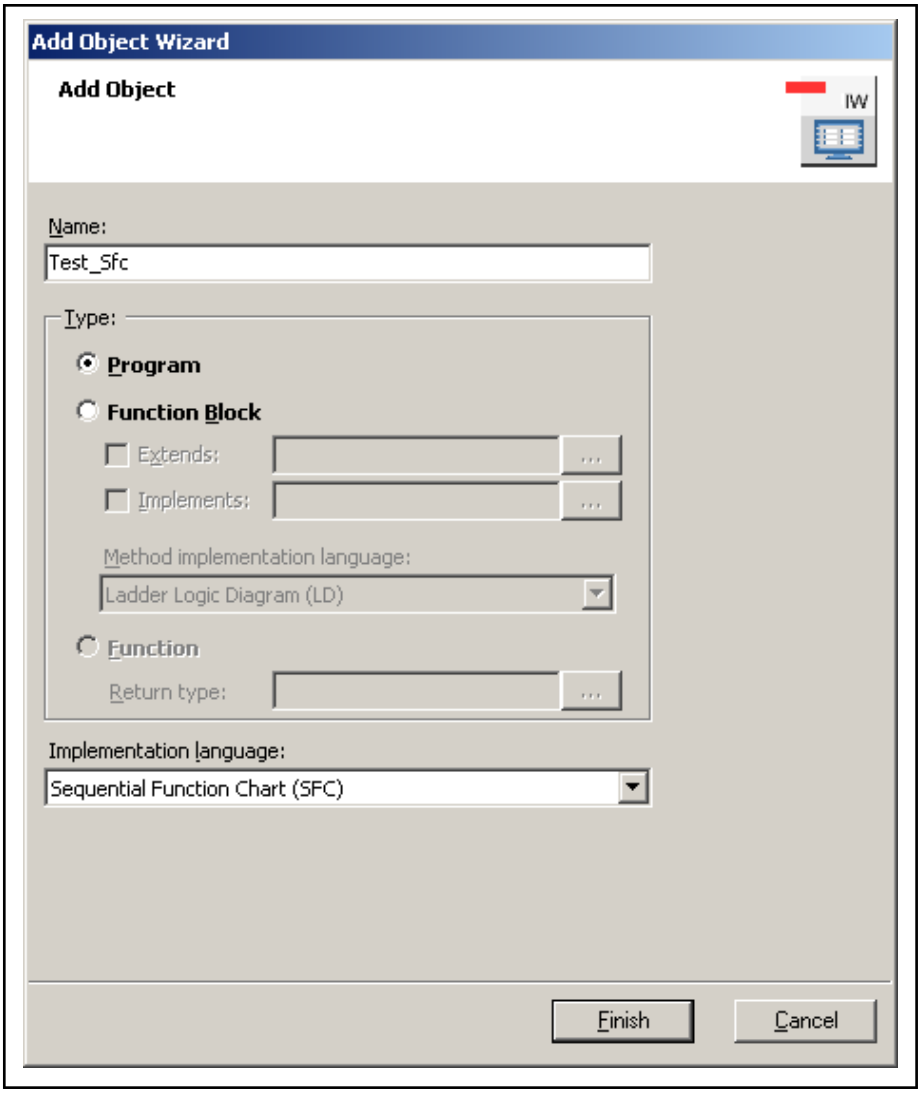


Fig. 8-2: New function block (set function block type and language)

Calling an SFC function block

Call the SFC function block from the main program "PlcProg".

Providing a function block with diagnostics

Provide a function block with diagnostics and operation modes via **Diagnostics ▸ SFC configuration editor...** or via the context menu of the function block **SFC configuration editor...**. Internal variables are automatically created in the declaration.

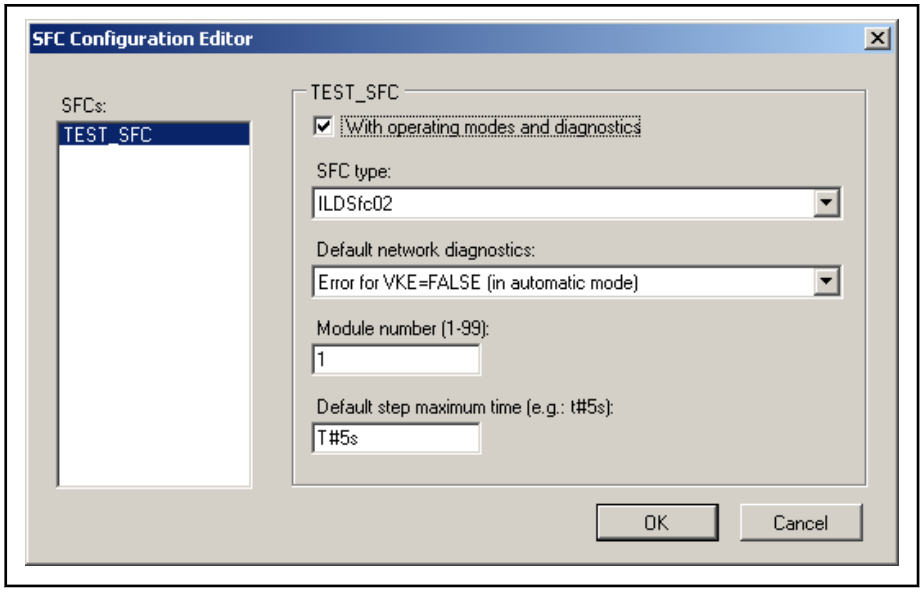


Fig. 8-3: SFC Configuration Editor

After the conversion, the program looks as follows:

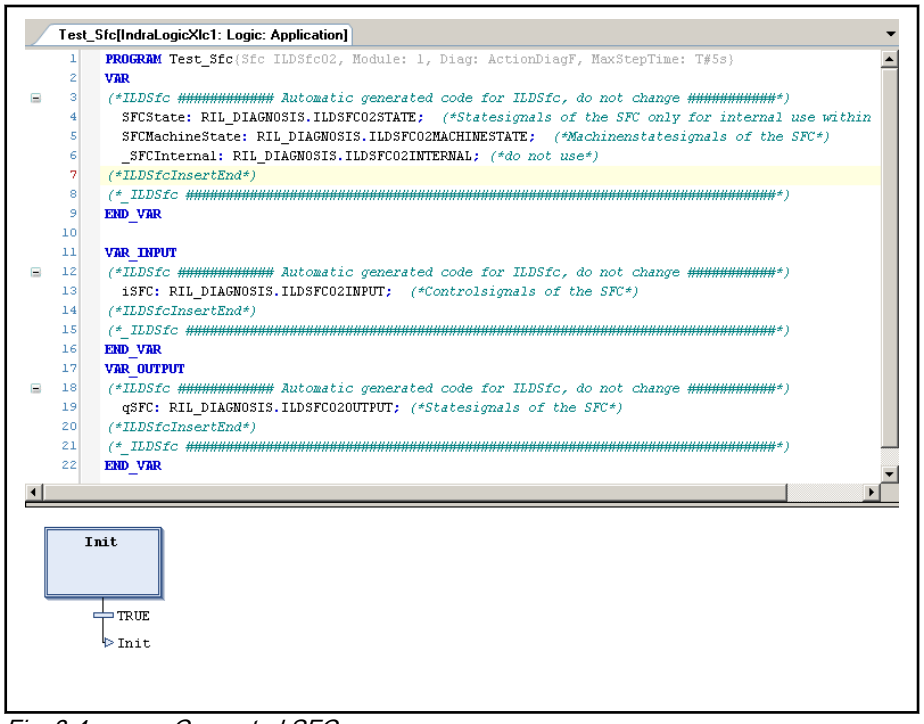


Fig. 8-4: Converted SFC

Programming the "aUserIn" action

Create a new action in the SFC function block with the name "aUserIn".

Program the "aUserIn" action now. The starting condition is a high signal at the E-STOP switch. The home position is defined as "Stop Position Left" and "Stop Position Bottom".

Application example

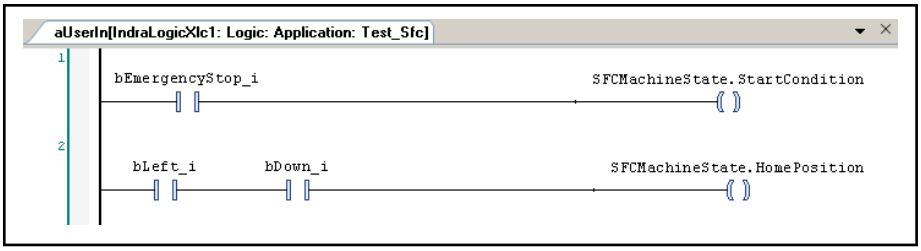


Fig. 8-5: Programming the "aUserIn" action

Creating actions

In addition, four additional actions are required to move the workpiece along the rectangular path:

- "aRight" moves the workpiece to the right
- "aLeft" moves the workpiece to the left
- "aUp" moves the workpiece upwards
- "aDown" moves the workpiece downwards

All actions are programmed according to the same scheme, variables with the suffix "_i" being Boolean inputs (limit switches) and variables with the suffix "_q" representing actuators.

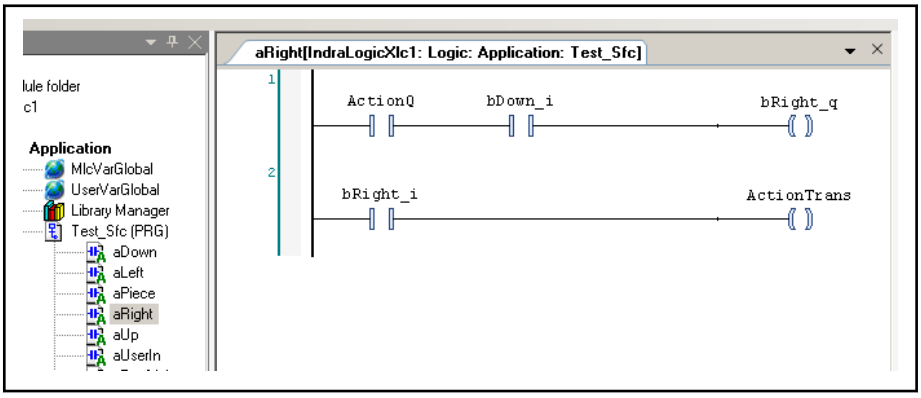


Fig. 8-6: Creating actions

Editing an SFC function block

Now, all required actions have been created. The SFC is at first programmed for automatic mode. The flag "bStart_i" is used to start the SFC.

The comment in the step attributes should be used, since it is displayed in the visualization and thus increases the information contained in the messages.

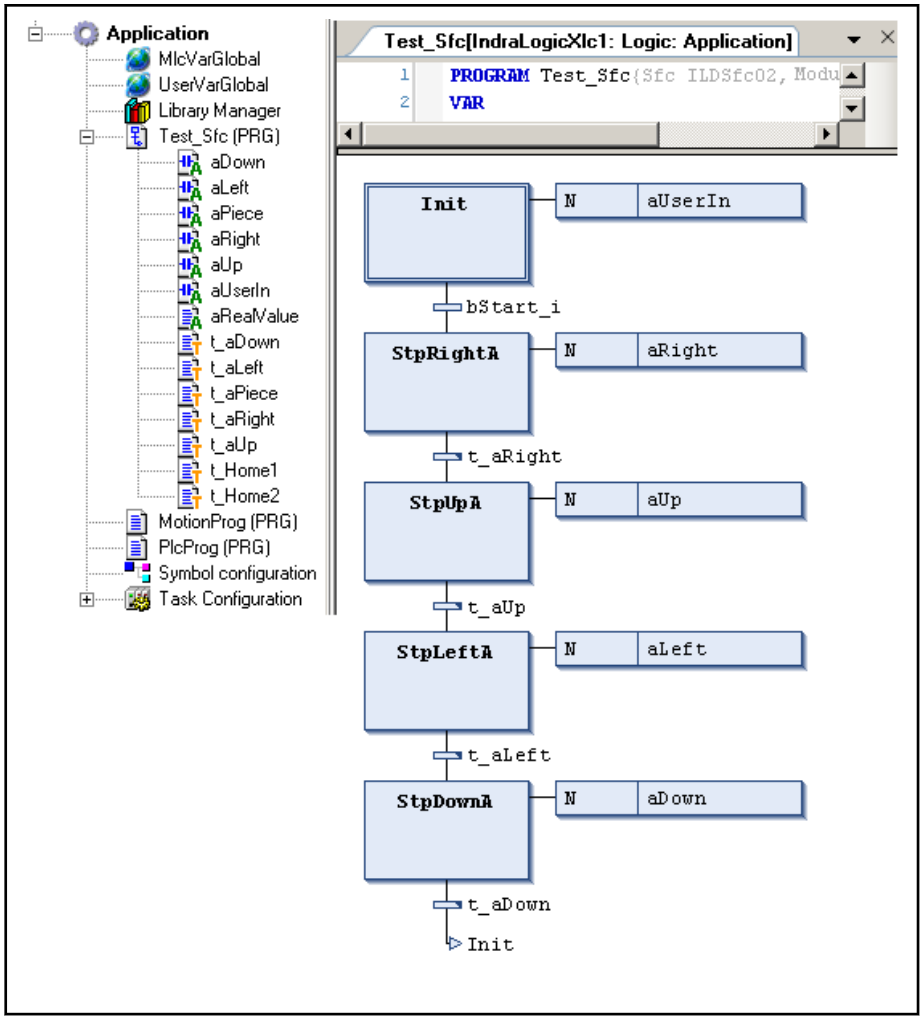


Fig. 8-7: Editing an SFC function block

Editing transitions

In this example, the transitions were edited in the programming language "Structured Text".

The transition conditions can be used either directly on the transition node (single variable) or as individual transition in ST.

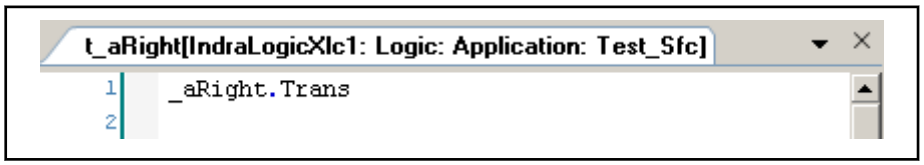


Fig. 8-8: Editing a transition

Editing an SFC function block in homing

The automatic branch is now complete. The homing branch is still missing. The workpiece is to be moved to the starting position "Bottom left" on the shortest way. This results in a homing branch for which the workpiece is first moved to the bottom and then to the left to the home position as well as a second branch at which the workpiece is first moved to the left and then to the bottom.

Application example

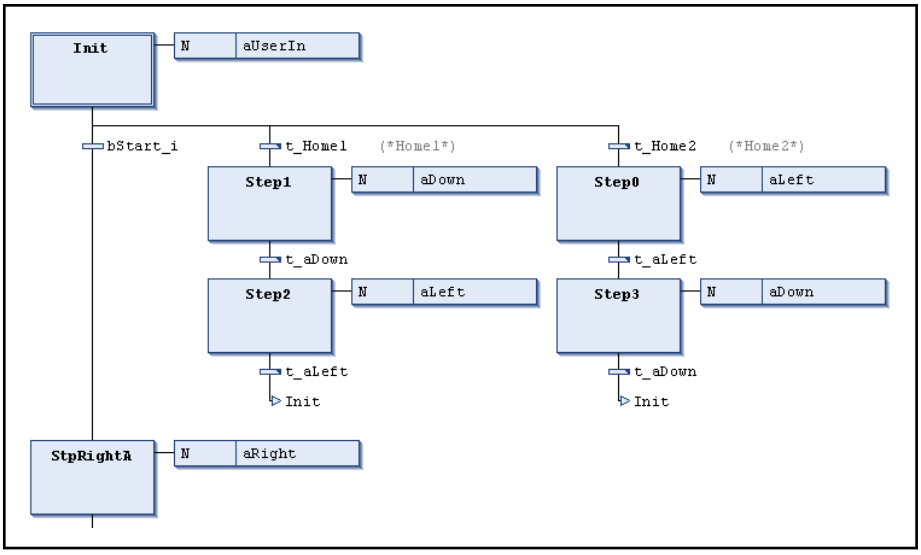


Fig. 8-9: Editing homing

Editing transitions in homing

The transition conditions within the homing chain and the automatic chain are identical. However, the first transition leading into the homing chain has to be differentiated.

First, the transition properties have to be set. For these properties, go to **Diagnostics ▸ SFC transition editor...** or via context menu in the transition **SFC transition editor...**:

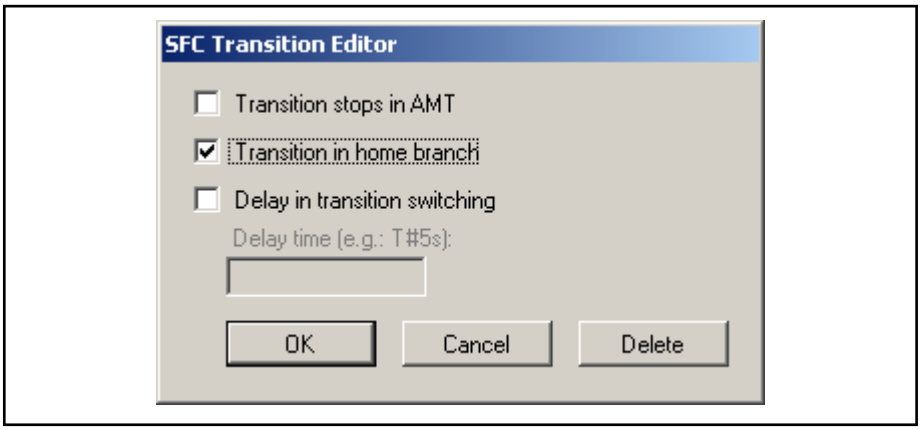


Fig. 8-10: SFC Transition Editor (Homing)

Close the dialog with **OK**. The transition string is automatically inserted into the first line.

The first homing branch is used as long as the workpiece is at the bottom or still on the right side. This leads to the following transition for the first homing branch:

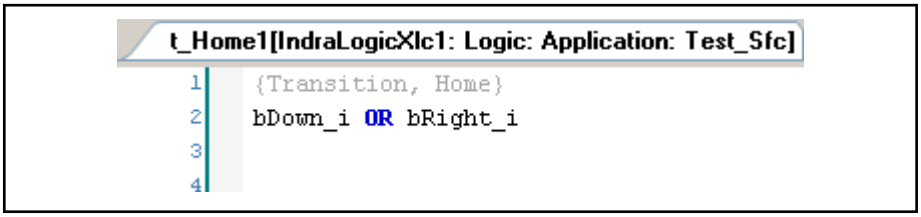


Fig. 8-11: Transition conditions for the first homing branch

The declaration of the transition condition for the second homing branch is created the same way:

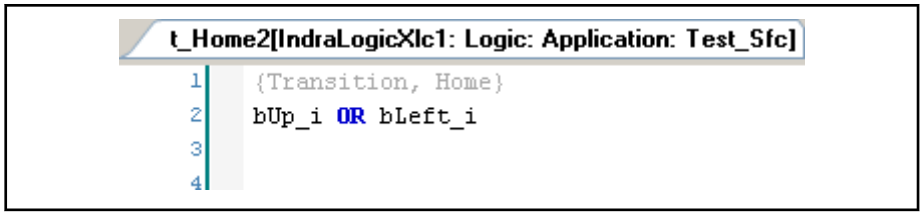


Fig. 8-12: Transition conditions for the second homing branch

Creating a tool for tests

To facilitate testing the SFC, a small program is created in the "PlcProg" function module simulating the procedure.

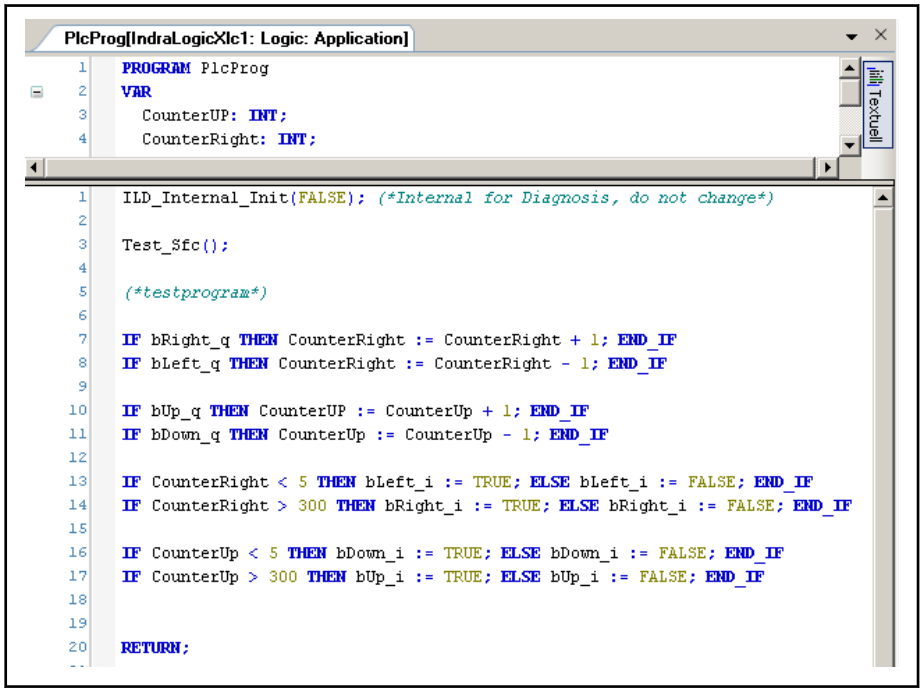


Fig. 8-13: Test program for the SFC

Compiling and loading a project

The project can be compiled with **Create ▸ Build**. The compilation should not output any error messages. If the project has not already been saved, save it now. Load the project to the control and start the control.

Starting an SFC

The forcing of inputs can now be used to control the SFC.

Set the following inputs successively to "TRUE":

- `bEmergencyStop_i`
- `bStart_i`
- `Test_Sfc.iSFC.Enable`
- `Test_Sfc.iSFC.Start`

The SFC runs exactly one cycle and then returns to the Init step. The SFC can be stopped and restarted at any given point in time via the `Test_Sfc.iSFC.Start` input.

Testing an SFC

If an error occurs (e.g. input `bEmergencyStop_i` drops to "Low"), a message appears in the variable `Test_Sfc.qSFC.Message`. Reset the malfunction via the `Test_Sfc.iSFC.Reset` input.

Testing manual mode

Manual mode is now tested. Switch the SFC to manual mode via the `Test_Sfc.iSFC.Manual` input. It is now possible to force either steps or actions. Step and action are only edited until the transition switches. This means that no further programming is required for manual mode. All con-

straints entered while creating the actions remain valid. The easiest way to force is via the visualization of IndraLogic. However, it is also implemented in the M-keys of the operator devices.

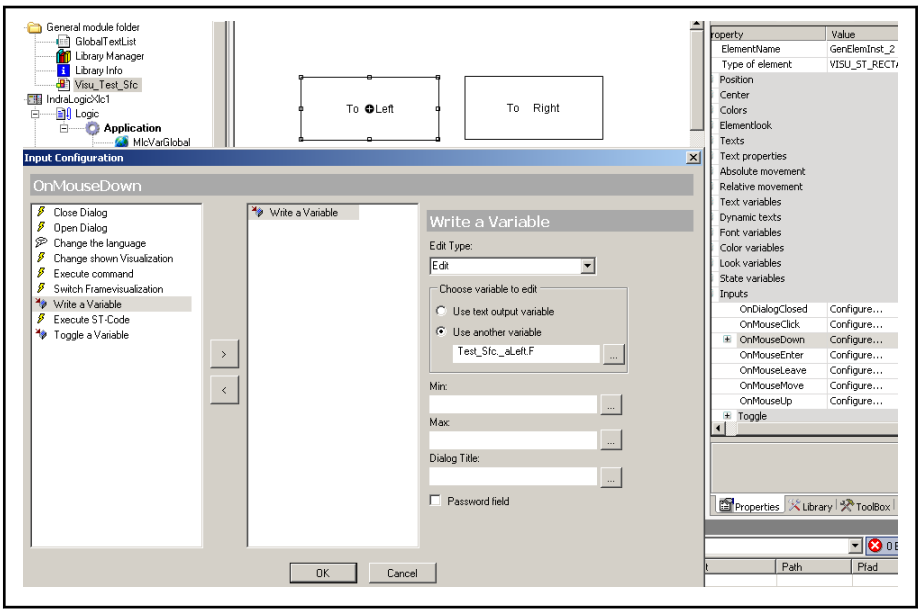


Fig. 8-14: Testing manual mode

Testing homing mode

Afterwards, the "Home position" has to be reached via "Homing". To do this, set the `Test_Sfc.iSFC.Home` input to "TRUE" and the `Test_Sfc.iSFC.Start` input back to "HIGH". Then, the SFC returns to the "home position".

8.2 Additional examples

ON and OFF qualifier

If the ON or OFF qualifier is used, certain functions and network properties are required within the action.

Example:

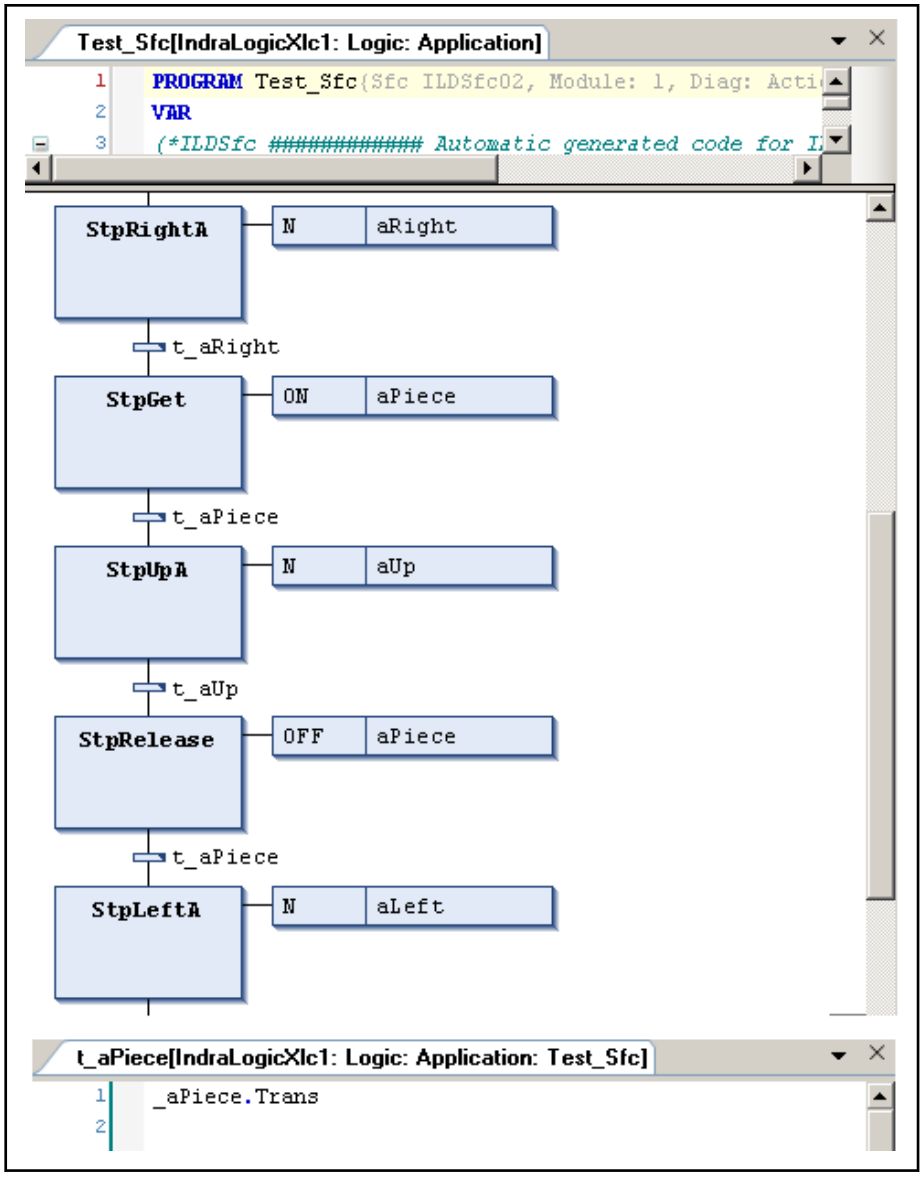


Fig. 8-15: Example: On/Off qualifier in the SFC

Application example

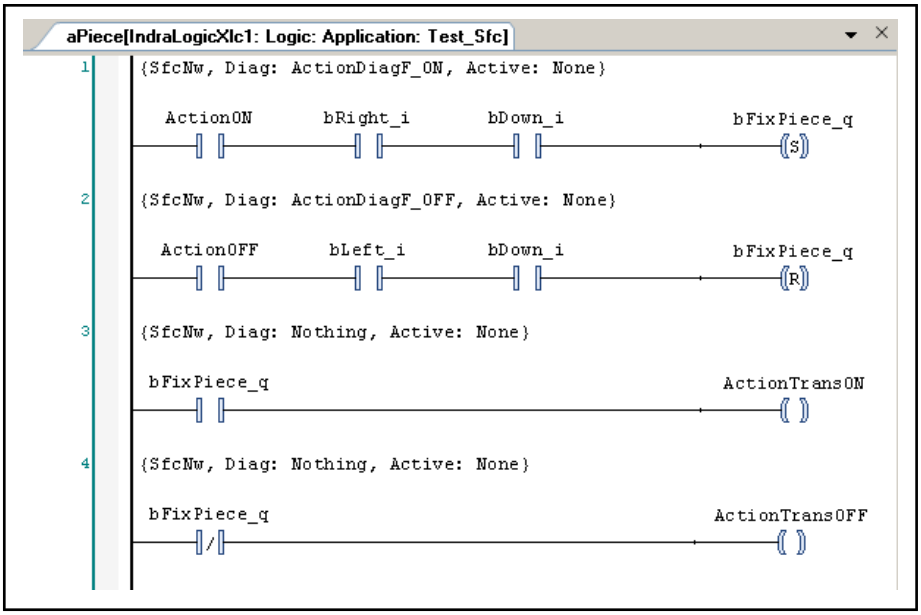


Fig. 8-16: Example: On/Off qualifier in the action

GO-qualifier With the GO-qualifier, REAL values can be transferred to an action.

Example:

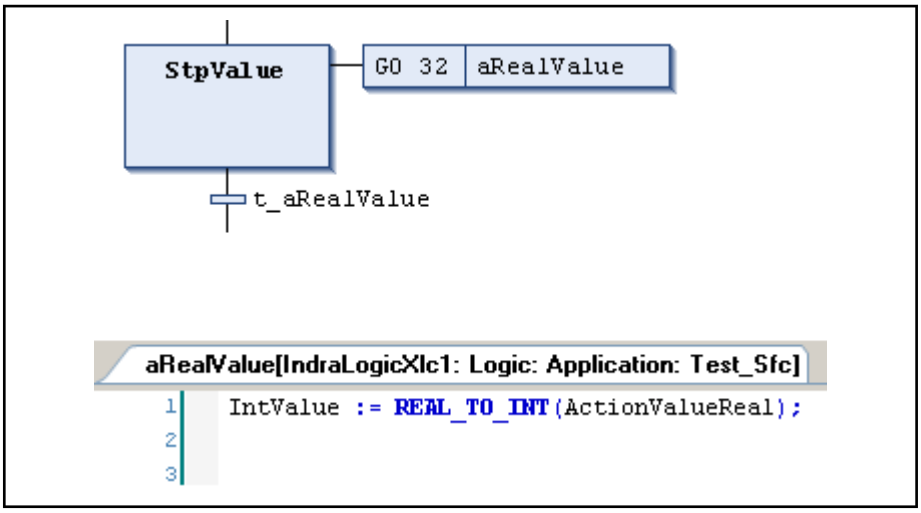


Fig. 8-17: Example: GO-qualifier

8.3 System variables of the SFC

There are internal SFC variables which can be used within the chain.

Variable	Description
ActionQ	Active processing of this action without post-processing
ActionA	Active processing of this action including post-processing
ActionON	Enabling processing of this action
ActionOFF	Disabling processing of this action
ActionGO	Activating this action (only for action qualifier GO)
ActionValueReal	Transfer value (REAL) of this action (only for GO)

Variable	Description
ActionTrans	Transition condition of this action
ActionTransON	Enabling transition condition of this action
ActionTransOFF	Disabling transition condition of this action
ManualForcing	Bypassing conditions in manual mode
SFCStateManual	SFC is in "manual" mode
SFCStateAuto	SFC is in "automatic" mode

Tab. 8-1: System variables of the SFC

☞ The states are always only valid within the currently active action. The states are invalid in other actions or outside the SFC. IndraLogic always displays the variable state of the end of the PLC cycle. The online status display in an action can therefore be different from the actual state.

Each step contains a data structure. The most important elements of the structure are listed in the following:

Structure element	Type	Property	Description
StepName.**X**	BOOL	Read-only	Indicates whether the step is active (TRUE)
StepName.**T**	TIME	Read-only	Returns the time since activation of the step
StepName.**F**	BOOL	Read and write	In manual mode, the step can be activated with TRUE

Tab. 8-2: SFC step data

Each action contains a data structure. The most important elements of the structure are listed in the following:

Structure element	Type	Property	Description
ActionName.**F**	BOOL	Read and write	In manual mode, the step can be activated with TRUE
ActionName.**FON**	TIME	Read and write	In manual mode ,the action can be activated for ON processing with TRUE
ActionName.**FOFF**	BOOL	Read and write	In manual mode, the action can be activated for OFF processing with TRUE
ActionName.**Trans**	BOOL	Read-only	Result of the action transition (ActionTrans) used as step transition

Tab. 8-3: SFC action data

9 Service and support

Our worldwide service network provides an optimized and efficient support. Our experts offer you advice and assistance should you have any queries. You can contact us **24/7**.

Service Germany

Our technology-oriented Competence Center in Lohr, Germany, is responsible for all your service-related queries for electric drive and controls.

Contact the **Service Hotline** and **Service Helpdesk** under:

Phone:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
E-mail:	service.svc@boschrexroth.de
Internet:	http://www.boschrexroth.com/

Additional information on service, repair (e.g. delivery addresses) and training can be found on our internet sites.

Service worldwide

Outside Germany, please contact your local service office first. For hotline numbers, refer to the sales office addresses on the internet.

Preparing information

To be able to help you more quickly and efficiently, please have the following information ready:

- Detailed description of malfunction and circumstances
- Type plate specifications of the affected products, in particular type codes and serial numbers
- Your contact data (phone and fax number as well as your e-mail address)

Index

Index

T

W

Notes

The Drive & Control Company

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Phone +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics

R911344490

DOK-IL*2G*-SFC*OPM*V14-AP02-EN-P