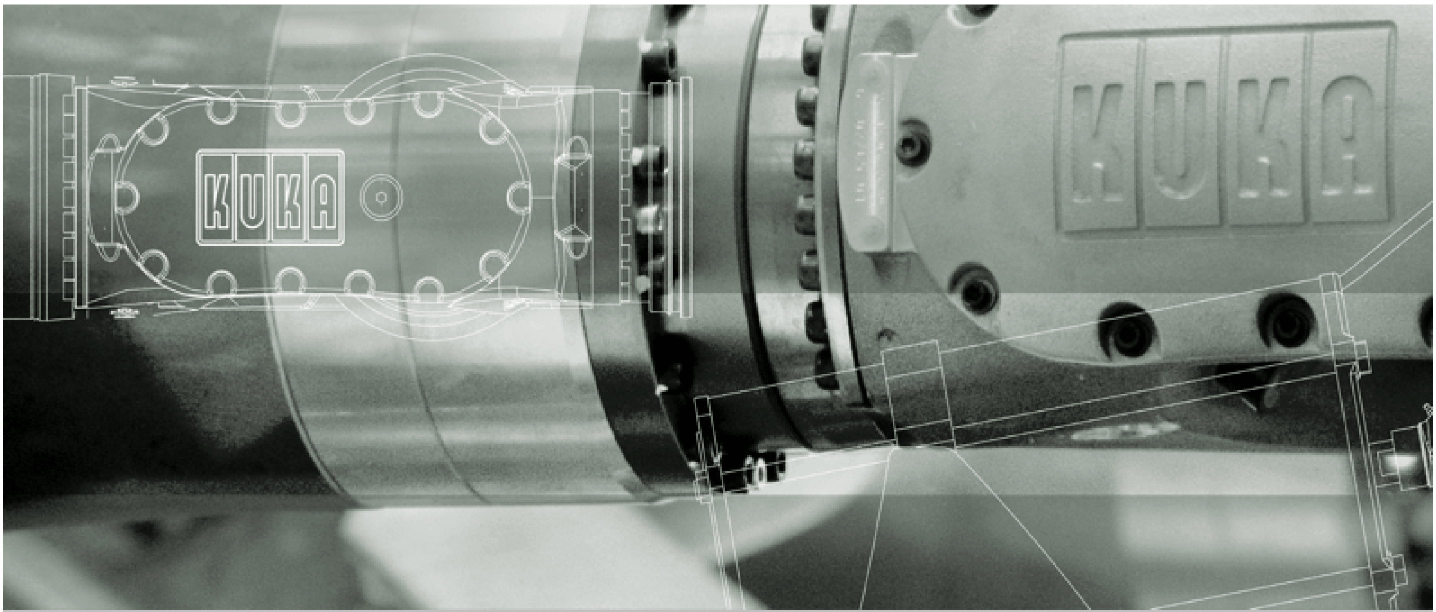


KUKA Robot Group

KUKA System Technology (KST)

KUKA.RobotSensorInterface (RSI) 2.1

For KUKA System Software (KSS) 5.4, 5.5, 7.0

Issued: 08.05.2007 Version: 2.1

KUKA Roboter GmbH
Zugspitzstraße 140
D-86165 Augsburg
Germany

Other functions not described in this documentation may be operable in the controller. The user has no claims to these functions, however, in the case of a replacement or service work.

We have checked the content of this documentation for conformity with the hardware and software described. Nevertheless, discrepancies cannot be precluded, for which reason we are not able to guarantee total conformity. The information in this documentation is checked on a regular basis, however, and necessary corrections will be incorporated in the subsequent edition.

Subject to technical alterations without an effect on the function.

KIM-PS4-DOC

Contents

1 Introduction

1.1 Target group

This documentation is aimed at users with the following knowledge and skills:

- Advanced KRL programming skills
- Advanced knowledge of the robot controller system
- Advanced knowledge of field bus interfaces
- Knowledge of digital technology
- Knowledge of object-oriented programming

For optimal use of our products, we recommend that our customers take part in a course of training at KUKA College. Information about the training program can be found at www.kuka.com or can be obtained directly from our subsidiaries.

1.2 Robot system documentation

The robot system documentation consists of the following parts:

- Operating instructions for the robot
- Operating instructions for the robot controller
- Operating and programming instructions for the KUKA System Software
- Documentation relating to options and accessories

Each of these sets of instructions is a separate document.

1.3 Representation of warnings and notes

Safety

Warnings marked with this pictogram are relevant to safety and **must** be observed.

Danger!
This warning means that death, severe physical injury or substantial material damage **will** occur, if no precautions are taken.

Warning!
This warning means that death, severe physical injury or substantial material damage **may** occur, if no precautions are taken.

Caution!
This warning means that minor physical injuries or minor material damage **may** occur, if no precautions are taken.

Notes

Notes marked with this pictogram contain tips to make your work easier or references to further information.

Tips to make your work easier or references to further information.

1.4 Terms used

Term	Description
Container	Containers are used to group RSI objects together and structure them. All RSI objects in a single container can be deleted, activated or deactivated simultaneously.
Object ID	Each object is assigned a unique identifier by the system when it is created. The object ID can be used to address an RSI object.
Object parameters	The object parameters are used to adapt the function of an RSI object.
Parameter ID	The object parameters of an RSI object are addressed by a consecutive index, commencing with 1.
RSI context	The RSI context is the entire signal processing programmed with KUKA.RobotSensorInterface and consists of RSI objects and links between the RSI objects.
RSI monitor	The RSI monitor can record and visualize up to 24 signals from the RSI context.
RSI object	Each RSI object has a signal functionality and corresponding signal inputs and/or outputs.

1.5 Trademarks

Windows is a trademark of Microsoft Corporation.

2 Product description

2.1 KUKA.RobotSensorInterface overview

KUKA.RobotSensorInterace is an add-on technology package with the following functions:

Functions

- Configuration of deterministic signal processing in the real-time system of the robot controller.
- Influence on the robot motion or program execution by means of the signal processing.
- Visualization of the signals via a monitor.

Characteristics

- Cyclical signal processing and evaluation in the interpolation cycle (12 ms) parallel to program execution.
- Configuration of the signal processing in the KRL program via RSI commands.
- Library with function blocks for signal processing (e.g. filters, logic gates, transformations, controllers, etc.).
- Creation of signal processing with up to 100 active RSI objects.
- Combination of different sensor technologies.

Areas of application

- Basic technology for real-time sensor applications with cyclical signal processing and evaluation.

Communication

The robot controller communicates with the sensor system via a field bus. The required data and signals for the signal processing are supplied by the robot system or read by the field bus.

Further information about KUKA field bus cards can be found in the corresponding KUKA documentation.

2.2 Functional principle

Description

Signal processing is established using RSI objects. An RSI object has a signal functionality and corresponding signal inputs and/or outputs.

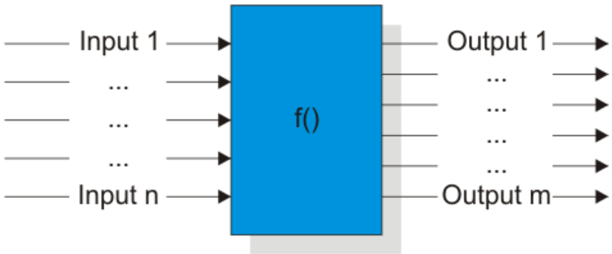


Fig. 2-1: Schematic structure of an RSI object

The following groups of RSI objects exist:

- Data access objects
- Signal processing objects
- Action objects

The RSI context is derived from the links of the signal functionalities of multiple RSI objects. The RSI context is a freely configurable signal flow and is generated from the KRL program.

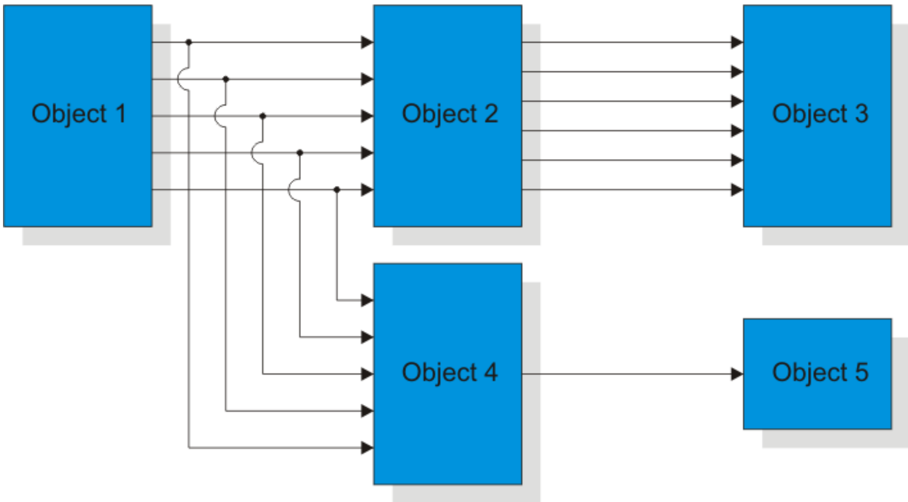


Fig. 2-2: Schematic structure of an RSI context

The KRL program also activates and deactivates calculation of the signal processing parallel to program execution.

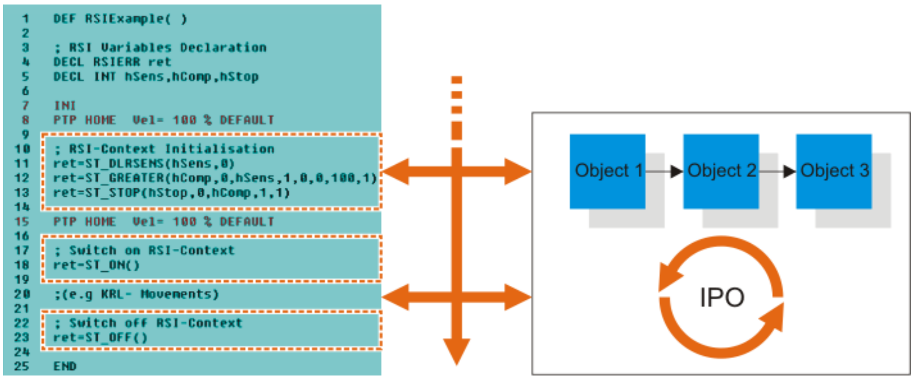


Fig. 2-3: Interaction between KRL program and signal processing

3 Safety

Personnel

- All persons working with the robot system must have read and understood the robot system documentation, including the safety chapter.

Further information is contained in the operating and programming instructions, in the robot operating instructions and in the robot controller operating instructions.

- KUKA.RobotSensorInterface is a software package and contains no hardware components. The system integrator is responsible for correct selection of the necessary components.
- The system integrator can generate complete applications with this technology package. For this, the system integrator must configure the technology package as appropriate for the specific application.

Robot system

- The robot system with KUKA.RobotSensorInterface must be operated in accordance with the applicable national laws, regulations and standards.
- The user must ensure that the system can be operated in complete safety.

Sensor-assisted operation

- If used incorrectly, KUKA.RobotSensorInterface can cause personal injury and material damage.
- In sensor-assisted operation, the robot may move unexpectedly in the following cases:
  - Incorrectly parameterized RSI objects
  - Hardware fault (e.g. incorrect cabling, break in the sensor cable or sensor malfunction)
- Unexpected movements may cause serious injuries and substantial material damage. The user is obliged to minimize the risk of injury to himself/herself and other people, as well as the risk of material damage, by adopting suitable safety measures (e.g. workspace limitation).
- At the start of signal processing with KUKA.RobotSensorInterface, the system generates the following acknowledgement message in T1 or T2 mode:

  !!!Attention –RSI sensor mode active!!!
- New or modified programs must always be tested first in operating mode T1. If the reduced velocity in T1 mode is insufficient for the process, a program can be tested in T2 mode, as long as there is no-one in the danger zone of the robot.
- During programming, the presence of persons within the danger zone of the robot is to be avoided.
- The robot must only be moved at reduced velocity (max. 250 mm/s) in T1 mode during programming in the danger zone of the robot. This is to give the operator enough time to move out of the way of hazardous robot motions or to stop the robot.
- In T1 and T2 modes, an enabling switch and a Start key must be held down in order to move the robot. In the event of unexpected movements, the robot can be stopped immediately by releasing the enabling switch or pressing it down fully (panic position).

Workspace limitation

- The axis ranges of all robot axes are limited by means of adjustable software limit switches. These software limit switches must be set in such a way that the workspace of the robot is limited to the minimum range required for the process.
- The KUKA System Software (KSS) allows the configuration of a maximum of 8 Cartesian and 8 axis-specific workspaces. The user must configure

the workspaces in such a way that they are limited to the minimum range required for the process. This reduces the risk of damage caused by unexpected movements in sensor-assisted operation to a minimum.

Further information about configuring workspaces is contained in the Operating and Programming Instructions for System Integrators.

Path correction limitation

- By default, KUKA.RobotSensorInterface limits the maximum path correction to +/- 5 mm for translational direction corrections and +/- 5° for orientation or axis corrections.
- If the signal processing with KUKA.RobotSensorInterface results in a larger path correction than the limitation will permit, the correction is rejected and the following message is generated:
  - For a Cartesian path correction (ST_PATHCORR): **SEN: ST_PATHCORR – correction out of range xxx**
  - For an axis angle correction (ST_AXISCORR): **SEN: ST_AXISCORR – correction out of range xxx**
- If the preset correction range is not sufficient for the process, it can be adapted. To do so, the RSI command ST_SETPARAM is used to assign correspondingly adapted values to the object parameters of the RSI objects ST_PATHCORR or ST_AXISCORR.

  (>>> 6.11 "Example program for adapting the maximum path correction" page 28)

More detailed information about the RSI objects and commands can be found in the RSI command reference ...\DOC\rsiCommands.chm on the CD-ROM.

4 Installation

4.1 System requirements

Hardware

- Robot controller:
  - KR C2
  - KR C2 ed05
  - KR C3
  - KR C2 sr

Field bus

The following field bus components can be used for communication between the robot controller and connected peripheral devices:

- PCI Interbus card M/S (fiber-optic cable or copper)
- PCI Profibus card M/S
- PCI DeviceNet card M/S
- PCI ControlNet card

Sensor system

- Sensor system components according to the specific application

Software

- KUKA System Software (KSS) 5.4, 5.5, 7.0
- The following KRL resources are assigned during installation of the technology package:

KRL resource	Number
Outputs	$OUT[16]
Interrupts	11
Function generators	1

The KRL resources can be reconfigured (>>> 5.4 "Configuring KRL resources" page 15).

4.2 Installing KUKA.RobotSensorInterface

Precondition

- User group Expert
- Windows interface (CTRL+ESC)

Procedure

1. Start the **Setup** program from the CD-ROM and select a language. The files are copied onto the hard drive.
2. Confirm the reboot prompt with **OK**.
3. Reboot the robot controller. The installation is resumed and completed.

LOG file

A LOG file is created under C:\KRC\ROBOTER\LOG.

4.3 Uninstalling KUKA.RobotSensorInterface

Precondition

- KUKA.RobotSensorInterface is installed.
- User group “Expert”
- Windows interface (CTRL+ESC)

Procedure

1. Start the **UnInstall.exe** program in the directory C:\KRC_OPTION\RSI\UNINST. Uninstallation is prepared.

2. Confirm the reboot prompt with **OK**.
3. Reboot the robot controller.

LOG file

A LOG file is created under C:\KRC\ROBOTER\LOG.

4.4 Reinstalling KUKA.RobotSensorInterface

Precondition

- KUKA.RobotSensorInterface has been uninstalled.
- User group “Expert”
- Windows interface (CTRL+ESC)

Procedure

1. Start the **ReInstall.exe** program in the directory C:\KRC_OPTION\RSI\REINST. Setup is prepared.
2. Confirm the reboot prompt with **OK**.
3. Reboot the robot controller.

LOG file

A LOG file is created under C:\KRC\ROBOTER\LOG.

5 Configuration

5.1 Units for signal processing

Description

For signal processing with KUKA.RobotSensorInterface, the signals must be assigned units. When RSI objects are linked, the assigned units are used to carry out a plausibility check of the signal flow. A check is made to see whether a signal with the permissible unit has been applied to the input of an RSI object.

The plausibility check only recognizes different units, not different unit prefixes (e.g. **k**m, **m**m, etc.).

The units for the signals include the base SI units:

Variable	Unit	RSI constant
Length	Meter [m]	RSIUNIT_m
Mass	Kilogram [kg]	RSIUNIT_kg
Time	Second [s]	RSIUNIT_s
Electric current	Ampere [A]	RSIUNIT_A
Temperature	Kelvin [K]	RSIUNIT_K
Luminous intensity	Candela [cd]	RSIUNIT_Cd
Amount of substance	Mole [mol]	RSIUNIT_mol
No unit	-----	RSIUNIT_No

Additional units can be derived from the base SI units. The following derived units are already included in KUKA.RobotSensorInterface:

Variable	Unit	RSI constant
Force	Newton [N]	RSIUNIT_N
Torque	Newton-meter [Nm]	RSIUNIT_Nm
Electric potential difference	Volt [V]	RSIUNIT_V
Pressure	Pascal [Pa]	RSIUNIT_Pa

The unit of a signal is defined with a 32-bit INTEGER variable. There are 4 bits available for each base SI unit. The 4 bits can be used to raise each base SI unit to the power of -8 to +7.

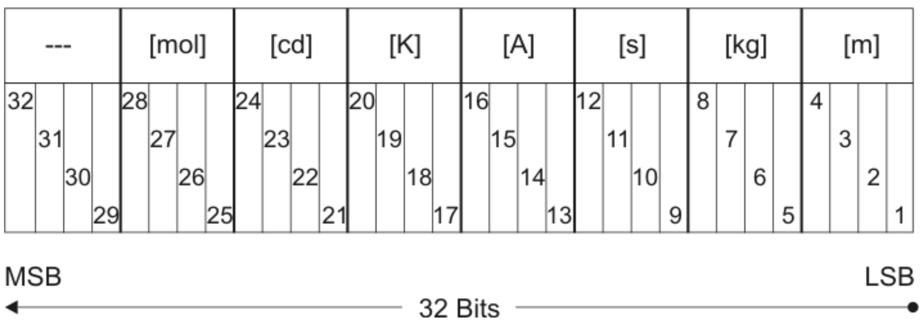


Fig. 5-1: Unit scheme of KUKA.RobotSensorInterface

LSB	Least significant bit
MSB	Most significant bit

5.2 Creating a new unit

Precondition

- User group “Expert”.
- Operating mode T1 or T2.
- No program may be selected.

Procedure

1. Express the new unit is base SI units.

   Example: electric field strength [V/m] in base SI units corresponds to $[kg]*[m]/[A]*[s]^3$
2. Calculate the hexadecimal value of the new unit in accordance with the unit scheme (>>> 5.1 "Units for signal processing" page 13).

   Example: electric field strength $[kg]*[m]/[A]*[s]^3$ corresponds to the hexadecimal value $FD11_{Hex}$
3. Open the file ...\R1\TP\RSI\RSILIB.DAT.
4. Create the new unit as a global constant in the 'Composite Units' section of the file ...\R1\TP\RSI\RSILIB.DAT and assign the calculated hexadecimal value.

```
;Composite Units:
GLOBAL CONST INT RSIUNIT_N='HE11' ;[N]  Newton
GLOBAL CONST INT RSIUNIT_Nm='HE12' ;[Nm] Newtonmeter
GLOBAL CONST INT RSIUNIT_V='HFD12' ;[V]  Volt
GLOBAL CONST INT RSIUNIT_Pa='HE1F' ;[Pa] Pascal
GLOBAL CONST INT RSIUNIT_E='HFD11' ;[V/m] Volt per meter
;End Composite Units
```

5. Close and save the file ...\R1\TP\RSI\RSILIB.DAT. The new RSI unit [V/m] can be used in the signal processing.

Example

This example illustrates how the unit Newton [N] is derived. The formula for the unit Newton [N] in base SI units is $[kg]*[m]/[s]^2$. The unit Newton is not included in the base SI units and must therefore be derived.

Base SI unit	Bit range in the unit scheme	Power	Value
[kg]	$000000X0_{Hex}$	1	00000010_{Hex}
[m]	$0000000X_{Hex}$	1	00000001_{Hex}
$[s]^{-2}$	$00000X00_{Hex}$	-2	$00000E00_{Hex}$
Total			$00000E11_{Hex}$

The base SI units [kg] and [m] appear in their basic form in the formula for the unit Newton [N] and are not raised to a power. The resulting hexadecimal value in the corresponding bit range of these units is thus 1_{Hex} ($=0001_{Bin}$).

The base SI unit [s] appears in the formula for the unit Newton [N] as the denominator and is raised to the power -2_{Dec}. The hexadecimal value for the unit $[s]^{-2}$ is derived as follows:

1. The power, not preceded by a sign, specifies the binary starting value of the base SI unit:

   $2_{Dec} = 0010_{Bin}$
2. For the negative sign of the power, the ones complement of the binary value must be formed:

Ones complement of $0010_{Bin} = 1101_{Bin}$

3. Add a 1_{Bin} to the ones complement and form the twos complement:

   $1101_{Bin} + 0001_{Bin} = 1110_{Bin}$

The calculated binary value 1110_{Bin} for the unit $[s]^{-2}$ corresponds to the hexadecimal value E_{Hex}. The calculated hexadecimal values of the individual units must now be grouped together using the unit scheme.

---	[mol]	[cd]	[K]	[A]	[s]	[kg]	[m]
0	0	0	0	0	E	1	1

For the unit Newton [N], this results in the hexadecimal value $00000E11_{Hex}$.

5.3 Configuring the RSI message display

Description

The display of the RSI-specific messages in the message window can be deactivated if required. In this case, no RSI-specific messages are displayed.

Precondition

- User group “Expert”.
- Operating mode T1 or T2.
- No program may be selected.

Procedure

1. Open the file ...\R1\TP\RSI\RSILIB.DAT.
2. Assign the required value to the global variable RSIERRMSG in the 'RSI global Variables' section of the file ...\R1\TP\RSI\RSILIB.DAT.

Value of the variable RSIERRMSG	Description
TRUE	The RSI-specific messages are displayed in the message window. Default setting
FALSE	The RSI-specific messages are deactivated and are **not** displayed in the message window.

```
;RSI global Variables:
GLOBAL INT OV_RSI=30 ; Override for ST_SKIP/RET... movement after
interrupt
GLOBAL INT RSIBREAK=16 ;Index of break motion condition
GLOBAL BOOL RSIERRMSG=TRUE ;Flag for enabling BOF RSI error mes-
sages
GLOBAL INT RSITECHIDX=1 ; Tech Channel used for RSI
;End RSI global Variables
```

3. Close and save the file ...\R1\TP\RSI\RSILIB.DAT.

5.4 Configuring KRL resources

Description

The following KRL resources from KUKA.RobotSensorInterface can be configured:

- Digital output $OUT[16]
- Function generator 1

Interrupt 11 which is used must not be modified.

Precondition

- User group “Expert”.
- Operating mode T1 or T2.
- No program may be selected.

Procedure

1. Open the file ...\R1\TP\RSI\RSILIB.DAT.
2. To modify digital output $OUT[16], assign the number of the digital output to the global variable RSIBREAK in the 'RSI global Variables' section.

```
;RSI global Variables:
GLOBAL INT OV_RSI=30 ; Override for ST_SKIP/RET... movement after
interrupt
GLOBAL INT RSIBREAK=16 ;Index of break motion condition
GLOBAL BOOL RSIERRMSG=TRUE ;Flag for enabling BOF RSI error mes-
sages
GLOBAL INT RSITECHIDX=1 ; Tech Channel used for RSI
;End RSI global Variables
```

3. To modify the function generator, assign the number of the function generator to the global variable RSITECHIDX in the 'RSI global Variables' section.

```
;RSI global Variables:
GLOBAL INT OV_RSI=30 ; Override for ST_SKIP/RET... movement after
interrupt
GLOBAL INT RSIBREAK=16 ;Index of break motion condition
GLOBAL BOOL RSIERRMSG=TRUE ;Flag for enabling BOF RSI error mes-
sages
GLOBAL INT RSITECHIDX=1 ; Tech Channel used for RSI
;End RSI global Variables
```

4. Close and save the file ...\R1\TP\RSI\RSILIB.DAT.

6 Programming

6.1 Programming overview

Overview

Step	Description
1	Declaring variables. (>>> 6.2 "Declaring variables" page 18)
2	Creating RSI objects or containers. (>>> 6.3 "Creating RSI objects and containers" page 20)
3	Linking signals (optional). (>>> 6.4 "Linking signals" page 21)
4	Reading / setting object parameters (optional). (>>> 6.5 "Reading / setting object parameters" page 22)
5	Activating / deactivating RSI objects or containers (optional). (>>> 6.6 "Activating / deactivating RSI objects or containers" page 22)
6	Deleting RSI objects or containers (optional). (>>> 6.7 "Deleting RSI objects or containers" page 23)
7	Programming motions. (>>> 6.9 "Programming RSI motions" page 25)
8	Activating / deactivating signal processing. (>>> 6.12 "Activating/deactivating signal processing" page 29)

Description

The following elements are required for signal processing:

- RSI objects
- Links
- Containers
- RSI commands (optional)

More detailed information about the RSI objects and commands can be found in the RSI command reference ...\DOC\rsiCommands.chm on the CD-ROM.

The RSI objects must be created in the KRL program. Each RSI object has a signal functionality and corresponding signal inputs and/or outputs. An RSI object can also have object parameters to adapt the function of the RSI object.

The signal inputs and outputs of the RSI objects must be linked to one another for a signal flow. Vital inputs must be linked when the RSI object is created (e.g. the first 2 inputs of an AND operation). Additional optional inputs can be linked subsequently.

Containers are used to group RSI objects together and structure them. All RSI objects in a single container can be deleted, activated or deactivated simultaneously. The RSI objects for the signal processing must be located in a container. Global container 0 is automatically created and activated when the system is booted. All RSI objects created in container 0 are calculated when signal processing is activated. Other containers with other RSI objects can be created, nested inside container 0 one level deeper. A container can contain a maximum of 254 RSI objects.

If other containers are created in global container 0, these other containers and all RSI objects inside them are initially deactivated.

The RSI monitor can record and visualize up to 24 signals from the RSI context (>>> 8.2.1 "RSI monitor" page 41).

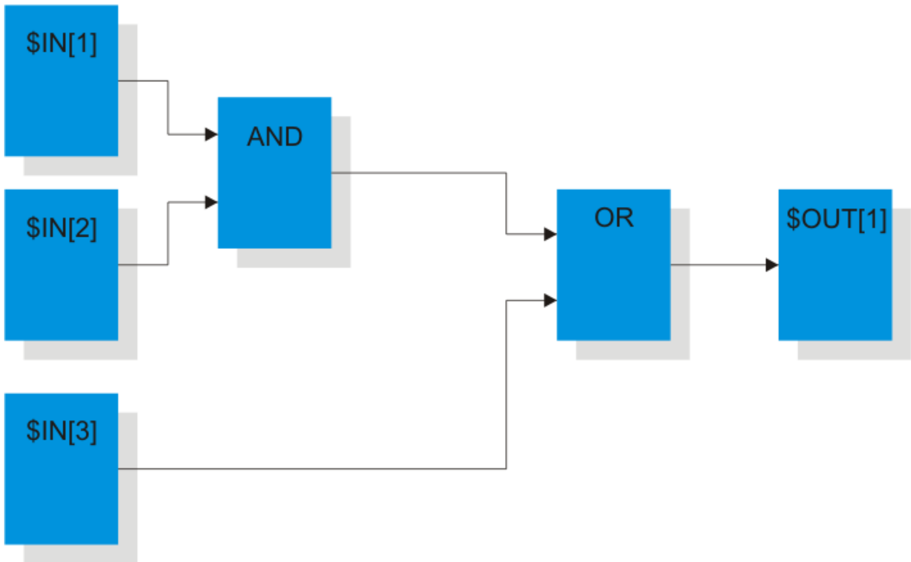


Fig. 6-1: Signal processing example

$IN[1]...$IN[3]	RSI objects for reading the states of digital inputs $IN[1]...$IN[3].
$OUT[1]	RSI object for setting digital output $OUT[1].
AND	Logic AND gate
OR	Logic OR gate

6.2 Declaring variables

Description

Variables must be declared in different places, depending on the sensor application and the RSI context. The following variables are required:

Variable	Data type
Variable for the return values of the RSI commands	RSIERR
Variables for the IDs	INTEGER
Variables for the object parameters	Dependent on the RSI object used
If RSI commands are used, variables for the RSI commands	Dependent on the RSI command used

The required variables can be declared in the following places:

Structure of the application	Variable for return values	Variables for IDs	Variables for object parameters	Variables for RSI commands
Without subprograms	Declare in the data list of the program.	Declare in the declaration section of the SRC file of the program.		
With subprograms	Declare as global variable in the global data list of the main program.	Declare in the declaration section of the SRC file of the program in which the RSI objects were created.		Declare as global variable in the global data list of the main program.

The required variables for the RSI context can also be declared in other places, depending on the application.

Precondition

- Program must be created.
- User group “Expert”.

Procedure

1. Open the SRC file of the program.
2. Select the menu sequence **Configure** > **Tools** > **Editor** > **Def-line** and display the DEF line of the SRC file.
3. Declare the required variables in the declaration section of the program.
4. Close and save the SRC file.
5. Open the data list of the main program and declare the variable for the return values of the RSI commands.

In order to enable the return values to be read with the variable display, the variable for the return values must be declared in the data list of the main program.

6. Close and save data list.

Example

```
1   DEF Program( )
2   INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID
3
4   INI
5
6   PTP HOME  Vel= 100 % DEFAULT
7
8   PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
9   LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
10  CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
11  LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
12  PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]
13
14  PTP HOME  Vel= 100 % DEFAULT
15
16  END
```

Line	Description
2	INTEGER variables for the object IDs of the required RSI objects
6...14	Programmed motions

6.3 Creating RSI objects and containers

Description

An RSI object is created by means of a command line in the statement section of the SRC file. The schematic structure of this command line is identical for all RSI objects.

Schematic structure of the command line:

<Return_value>=ST_<Object_name>(Object_ID,Container_ID,[{Signal_input 1...n: Object_X_ID, Output_index-X}],[Parameter]

Part of command line	Description
<Return_value>	The return value contains the error code after an RSI command has been executed.
ST_<Object_name>	Type of RSI object to be created (e.g. ST_ANAIN, ST_AND, ST_MAP2DIGOUT, etc.).
Object_ID	INTEGER variable for the object ID in order to access the RSI object. The value is automatically assigned by the robot system when the RSI object is created.
Container_ID	Number of the container in which the RSI object is to be created.
Signal input 1...n	For each signal input of the RSI object, the signal source (Object_X_ID) and the output index (Output_index_X) of the signal source must be specified.
Object_X_ID	Object ID of the RSI object that serves as the signal source.
Output_index_X	Signal output of the RSI object at which the signal is tapped for further processing.
Parameter	Object parameter for adapting the function of the object.

Precondition

- User group “Expert”.
- Variables for the RSI objects, containers and return values must be declared.
- If sensor data are read by the field bus: field bus connection must be established and configured.

Procedure

1. Open the SRC file of the program.
2. Select the menu sequence **Configure** > **Tools** > **Editor** > **Def-line** and display the DEF line of the SRC file.
3. Create the RSI objects or containers required for the signal processing in the statement section of the program.
4. Close and save the SRC file.

Example

```
1   DEF Program( )
2   INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID
3
4   INI
5
6   RET=ST_DIGIN(OBJECT1_ID,0,1,0,RSIUNIT_No)
7   RET=ST_DIGIN(OBJECT2_ID,0,2,0,RSIUNIT_No)
8   RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
9   RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
10
11  PTP HOME  Vel= 100 % DEFAULT
12
13  PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
14  LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
15  CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
16  LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
17  PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]
18
19  PTP HOME  Vel= 100 % DEFAULT
20
21  END
```

Line	Description
6, 7	RSI object ST_DIGIN for reading a digital input.
8	RSI object ST_OR for linking the signal outputs of the ST_DIGIN RSI objects with a logic OR gate.
9	RSI object ST_MAP2DIGOUT for sending the result of the OR operation to a digital output.
11...19	Programmed motions

6.4 Linking signals

Description

The optional signal inputs of an RSI object can be linked subsequently. The following RSI commands are available:

RSI command	Description
ST_NEWLINK	Links the signal output of an RSI object to an optional signal input of a different RSI object.
ST_DELLINK	Deletes the link between a signal output of an RSI object and the optional signal input of a different RSI object.
ST_CHANGELINK	Changes the signal source of an existing link. The command may only be used in the following cases: ■ Signal processing is deactivated. ■ Modification of a link to an optional signal input.

Precondition

- User group “Expert”.
- RSI objects must be created.

Procedure

1. Open the SRC file of the program.
2. Program the command line for the new link after the command lines for creating the RSI objects.

> The command cannot be executed until the relevant RSI objects have been created.

3. Close and save the SRC file.

6.5 Reading / setting object parameters

Description

The object parameters are used to adapt the function of an RSI object. The object parameters are transferred or assigned default values when the RSI object is created. The object parameters of an RSI object are addressed by an index, ranging from 1 to the number of object parameters of the RSI object.

The following RSI commands are available:

RSI command	Description
ST_GETPARAM	Reads the REAL value of an object parameter.
ST_GETPARAMINT	Reads the INT value of an object parameter.
ST_SETPARAM	Sets the object parameter of an RSI object to a user-defined value.

Precondition

- User group “Expert”.
- RSI objects must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Program the command line for the reading or setting of object parameters after the command lines for creating the RSI objects.

> The command cannot be executed until the relevant RSI objects have been created.

3. Close and save the SRC file.

6.6 Activating / deactivating RSI objects or containers

Description

Individual RSI objects or entire containers can be activated or deactivated in the signal flow. An RSI object can only be activated or deactivated if the following conditions are met:

- An RSI object can be activated if the preceding RSI objects in the signal flow are active.
- An RSI object can be deactivated if the following RSI objects in the signal flow are deactivated.

The following RSI commands are available:

RSI command	Description
ST_ENABLE	Activates an RSI object or a container in order to integrate the RSI objects into the signal processing.
ST_DISABLE	Deactivates an RSI object or a container in order to exclude the RSI objects from the signal processing.

Precondition

- User group “Expert”.
- RSI objects must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Program the command line for activating / deactivating RSI objects or containers after the command lines for creating the RSI objects.

The command cannot be executed until the relevant RSI objects have been created.

3. Close and save the SRC file.

6.7 Deleting RSI objects or containers

Description

Individual RSI objects or entire containers can be deleted. An RSI object can only be deleted if the following conditions are met:

- An RSI object can be deleted if it is not followed in the signal flow by other RSI objects.

The following RSI commands are available:

RSI command	Description
ST_DELOBJ	Deletes an RSI object or a container.
ST_RESET	Deletes the entire RSI context in the program.

Precondition

- User group “Expert”.
- RSI objects must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Program the command line for deleting RSI objects or containers after the command lines for creating the RSI objects.

The command cannot be executed until the relevant RSI objects have been created.

3. Close and save the SRC file.

6.8 Overview of RSI motions

Description

KUKA.RobotSensorInterface can be used to influence the motion of the robot by means of sensor data. The motion of the robot can be influenced in the following ways:

Type of motion	Description
Cancelable motion to the next point but one	When the sensor event occurs, the robot stops the current motion and moves directly to the next point but one (>>> Fig. 6-2).
Cancelable motion back to the start point	When the sensor event occurs, the robot stops the current motion and moves directly back to the start point of the current motion (>>> Fig. 6-3).
Cancelable relative motion, relative to the actual position	The robot follows the coordinates of the relative motion block relative to the actual position (>>> Fig. 6-4). If a motion is canceled, the robot moves to the next point but one.

Type of motion	Description
Cancelable sensor-guided motion	The robot moves solely in accordance with the sensor data provided and does **not** move to a defined end point. If KUKA.RobotSensorInterface calculates a path correction, the robot moves. If there is no path correction, the robot remains stationary (>>> Fig. 6-5).
Path correction	If the robot is moving along its programmed path and KUKA.RobotSensorInterface calculates a path correction, the robot corrects its path. The path correction can be absolute, to the programmed path, or relative to the correction of the previous interpolation cycle (>>> Fig. 6-6).

If an RSI motion is to be canceled on the grounds of a sensor event, the RSI object ST_BREAKMOVE must be used.

Paths

Cancelable motion to the next point but one:

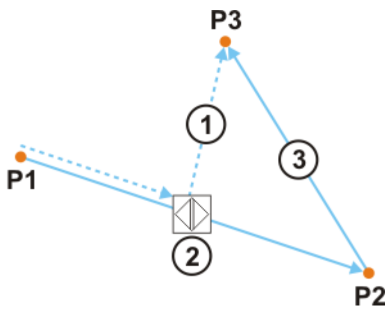


Fig. 6-2: Path of a cancelable motion to the next point but one

1 Canceled motion to the next point but one
2 Sensor event
3 Programmed path

Cancelable motion back to the start point:

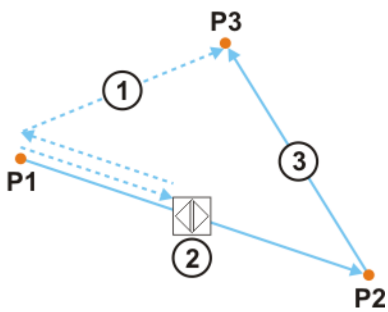


Fig. 6-3: Path of a cancelable motion back to the start point

1 Canceled motion back to the start point
2 Sensor event
3 Programmed path

Cancelable relative motion, relative to the actual position:

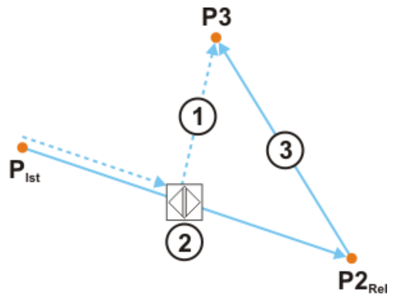


Fig. 6-4: Path of a cancelable relative motion, relative to the actual position

1	Canceled motion back to the next point but one
2	Sensor event
3	Programmed path
P_{Act}	Actual position
$P2_{Rel}$	Cartesian coordinates of the next point to which the robot is to move, relative to the actual position

Cancelable sensor-guided motion:

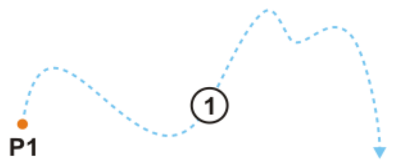


Fig. 6-5: Path of a cancelable sensor-guided motion

1	Sensor-guided robot motion
P1	Start point of the sensor-guided motion

Path correction:

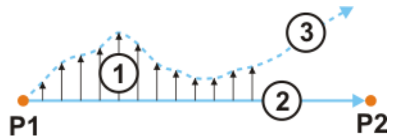


Fig. 6-6: Path of a path correction

1	Path corrections calculated with KUKA.RobotSensorInterface
2	Programmed path
3	Path corrected on the basis of the sensor data

6.9 Programming RSI motions

Precondition

- User group “Expert”.
- RSI objects must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Move the robot to the required position and teach the point.

> Note the name of the taught point.

3. Delete the point in the SRC file.

The coordinates of the taught point are retained in the data list of the program.

4. Close and save the SRC file.
5. Open the data list of the program and search for the taught point.

The variable name for the taught point is preceded in the data list by a capital X.
Example: Point P1 in the SRC file corresponds to **X**P1 in the data list.

6. Note the variable name for the coordinates of the taught point or rename it if required.
7. Close and save data list.
8. Open the SRC file and create the necessary RSI objects, depending on the RSI motion.
9. Program the required RSI motions in the statement section of the SRC file (>>> 6.8 "Overview of RSI motions" page 23).

The point name in the RSI motion command must be identical to the variable name for the coordinates of the taught point in the data list.

10. Close and save the SRC file.

6.10 Example programs for RSI motions

Example

The example program shows the programming for the following RSI motions:

- Cancelable motion to the next point but one
- Cancelable motion back to the start point
- Cancelable relative motion, relative to the actual position

```
DEF Program( )
DECL RSIERR RET
INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID,OBJECT5_ID

INI

RET=ST_DIGIN(OBJECT1_ID,0,1,0,RSIUNIT_No)
RET=ST_DIGIN(OBJECT2_ID,0,2,0,RSIUNIT_No)
RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
RET=ST_BREAKMOVE(OBJECT5_ID,0,OBJECT3_ID,1)

PTP HOME  Vel= 100 % DEFAULT

PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]

RET=ST_ON()

ST_SKIPLIN(XP8)
;ST_RETLIN(XP8)
;ST_LINREL {X 100}

RET=ST_OFF()

PTP HOME  Vel= 100 % DEFAULT

END
```

Line	Description
3	Required object IDs
11	RSI object ST_BREAKMOVE for canceling the motion if the RSI object ST_OR outputs a HIGH level
23	Cancelable motion to the next point but one
24	Cancelable motion back to the start point
25	Cancelable relative motion in the X direction, relative to the actual position

Example

The example program shows the programming for a cancelable sensor-guided motion.

```
1   DEF Program( )
2   DECL RSIERR RET
3   DECL INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID, _
             OBJECT5_ID,OBJECT6_ID,OBJECT7_ID
4
5   INI
6
7   RET=ST_DIGIN(OBJECT1_ID,0,1,0,RSIUNIT_No)
8   RET=ST_DIGIN(OBJECT2_ID,0,2,0,RSIUNIT_No)
9   RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
10  RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
11  RET=ST_BREAKMOVE(OBJECT5_ID,0,OBJECT3_ID,1)
12
13  RET=ST_DIGIN(OBJECT6_ID,0,2,1,RSIUNIT_m)
14  RET=ST_PATHCORR(OBJECT7_ID,0)
15  RET=ST_NEWLINK(OBJECT6_ID,1,OBJECT7_ID,1)
16
17  PTP HOME  Vel= 100 % DEFAULT
18
19  PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
20  LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
21  CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
22  LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
23  PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]
24
25  RET=ST_ON1(#TOOL,1)
26
27  ST_MOVESENS(1)
28
29  RET=ST_OFF()
30
31  PTP HOME  Vel= 100 % DEFAULT
32
33  END
```

Line	Description
3	Required object IDs
11	RSI object ST_BREAKMOVE for canceling the motion if the RSI object ST_OR outputs a HIGH level
13	RSI object ST_DIGIN for recording the sensor data via digital inputs $IN[9]...$IN[16] with a width of one byte (1 byte = 8 bits)
14	RSI object ST_PATHCORR for generating the path correction from the recorded sensor data
15	RSI command ST_NEWLINK for creating a link between the RSI objects ST_DIGIN and ST_PATHCORR
25	RSI command ST_ON1 for activating the signal processing and making the path correction relative to the TOOL coordinate system
27	Cancelable sensor-guided motion

Example

The example program shows the programming for a path correction.

```
1   DEF Program( )
2   DECL RSIERR RET
3   DECL INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID, _
             OBJECT5_ID,OBJECT6_ID
4
5   INI
6
7   RET=ST_DIGIN(OBJECT1_ID,0,1,0,RSIUNIT_No)
8   RET=ST_DIGIN(OBJECT2_ID,0,2,0,RSIUNIT_No)
9   RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
10  RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
11
12  RET=ST_DIGIN(OBJECT5_ID,0,2,1,RSIUNIT_m)
13  RET=ST_PATHCORR(OBJECT6_ID,0)
14  RET=ST_NEWLINK(OBJECT5_ID,1,OBJECT6_ID,1)
15
16  PTP HOME  Vel= 100 % DEFAULT
17
18  PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
19  LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
20  CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
21  LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
22  PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]
23
24  RET=ST_ON1(#TTS,1)
25
26  LIN XP8
27
28  RET=ST_OFF()
29
30  PTP HOME  Vel= 100 % DEFAULT
31
32  END
```

Line	Description
3	Required object IDs
12	RSI object ST_DIGIN for recording the sensor data via digital inputs $IN[9]...$IN[16] with a width of one byte (1 byte = 8 bits)
13	RSI object ST_PATHCORR for generating the path correction from the recorded sensor data
14	RSI command ST_NEWLINK for creating a link between the RSI objects ST_DIGIN and ST_PATHCORR
24	RSI command ST_ON1 for activating the signal processing and making the path correction relative to the tool-based moving frame
26	LIN motion with path correction

6.11 Example program for adapting the maximum path correction

Example

The example program illustrates the programming of an adaptation of the maximum permissible path correction in the X direction.

```
1   DEF Program( )
2   DECL RSIERR RET
3   INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID
4
5   INI
6   PTP HOME
7
8   RET=ST_DIGIN(OBJECT1_ID,0,1,1,RSIUNIT_m)
9   RET=ST_PATHCORR(OBJECT2_ID,0)
10  RET=ST_NEWLINK(OBJECT1_ID,1,OBJECT2_ID,1)
11
12  RET=ST_SETPARAM(OBJECT2_ID,1,-25) ; Min corr. in x: -25 mmm
13  RET=ST_SETPARAM(OBJECT2_ID,7,25)  ; Max corr. in x: 25 mmm
14
15  LIN P1
16  RET=ST_ON()
17  ST_MOVESENS(1)
18
19  END
```

Line	Description
3	Required object IDs
9	RSI object ST_PATHCORR for generating the path correction from the recorded sensor data
12	RSI command ST_SETPARAM to assign the value -25 to parameter 1 of the RSI object ST_PATHCORR; parameter 1 defines the maximum permissible lower limit for a correction in the X direction.
13	RSI command ST_SETPARAM to assign the value +25 to parameter 7 of the RSI object ST_PATHCORR; parameter 7 defines the maximum permissible upper limit for a correction in the X direction.

6.12 Activating/deactivating signal processing

Description

Once the signal processing has been completely created, it must be activated in the KRL program.

The following RSI commands are available:

RSI command	Description
ST_ON	Activates the signal processing. The signals are processed and evaluated in the interpolation cycle. Path corrections with the RSI object ST_PATHCORR **always** refer to the BASE coordinate system.
ST_ON1	Activates the signal processing. The signals are processed and evaluated in the interpolation cycle. Path corrections with the RSI object ST_PATHCORR can refer to the following coordinate systems. ▪ BASE coordinate system ▪ TOOL coordinate system ▪ Tool-based technological system (TTS) ▪ WORLD coordinate system The path corrections can be absolute with respect to the programmed path or relative to the corrected path.
ST_OFF	Stops the signal processing.

Precondition

- User group “Expert”.
- RSI objects and containers must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Depending on the sensor application, program the ST_ON or ST_ON1 command before the RSI motions.
3. Program the ST_OFF command at the required point in the SRC file.

> Signal processing with KUKA.RobotSensorInterface can be activated and deactivated more than once in the same SRC file using the ST_ON, ST_ON1 and ST_OFF commands.

4. Close and save the SRC file.

Example

```
DEF Program( )
DECL RSIERR RET
INT OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID,OBJECT5_ID

INI

RET=ST_DIGOUT(OBJECT1_ID,0,1,0,RSIUNIT_No)
RET=ST_DIGOUT(OBJECT2_ID,0,2,0,RSIUNIT_No)
RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
RET=ST_BREAKMOVE(OBJECT5_ID,0,OBJECT3_ID,1)

PTP HOME  Vel= 100 % DEFAULT

PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
CIRC P4 P5  Vel= 0.5 m/s CPDAT3 Tool[1] Base[1]
LIN P6 CONT Vel= 0.5 m/s CPDAT4 Tool[1] Base[1]
PTP P7 CONT Vel= 100 % PDAT2 Tool[1] Base[1]

RET=ST_ON()

ST_SKIPLIN(XP8)

RET=ST_OFF()

PTP HOME  Vel= 100 % DEFAULT

END
```

6.13 Example of a sensor application

Description

The programming of a sensor application is explained below using the example of force control with a force sensor. A precondition for this example is that the field bus interfacing of the force sensor and tool to the robot system has been carried out correctly.

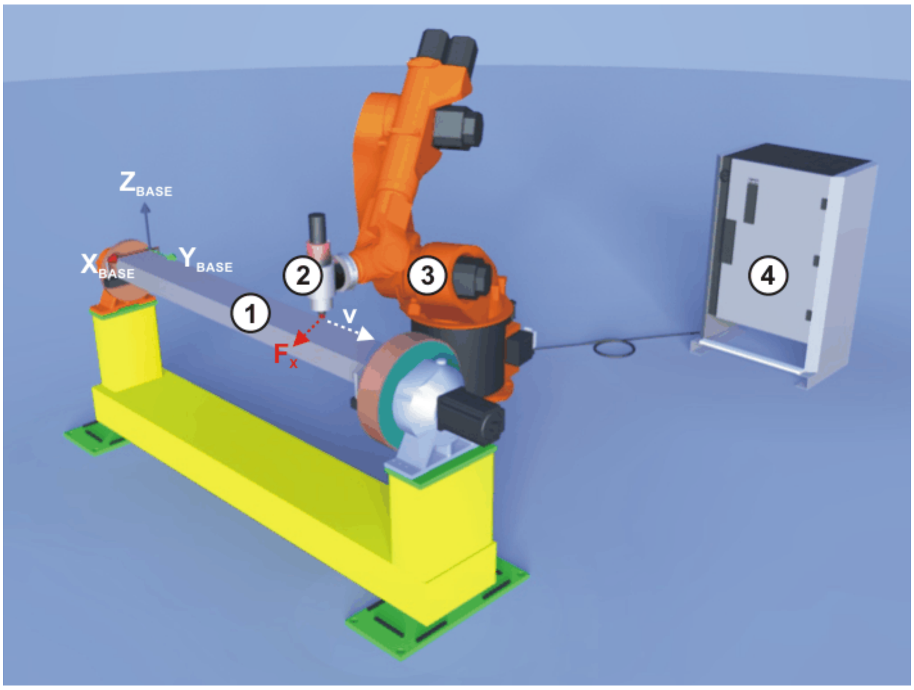


Fig. 6-7: Example of a sensor application

1	Workpiece that is to be deburred along the edge under force control
2	Tool with force sensor
3	Robot
4	Robot controller
F_X	Measured force in the X direction of the BASE coordinate system, perpendicular to the programmed path
v	Direction of motion

A force sensor, which supplies the sensor data for the force control, is mounted between the mounting flange of the robot and the tool. The robot moves to the workpiece and then moves in the Y direction of the BASE coordinate system. At the same time, the robot moves in the X direction of the BASE coordinate system because of the force control. Once component contact is made, the robot continues to move in the X direction of the BASE coordinate system under force control and maintains the force setpoint value throughout the motion. The maximum deviation from the programmed path is limited. If the robot exceeds the maximum permissible deviation, an output is set.

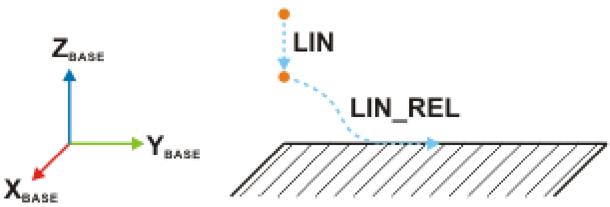


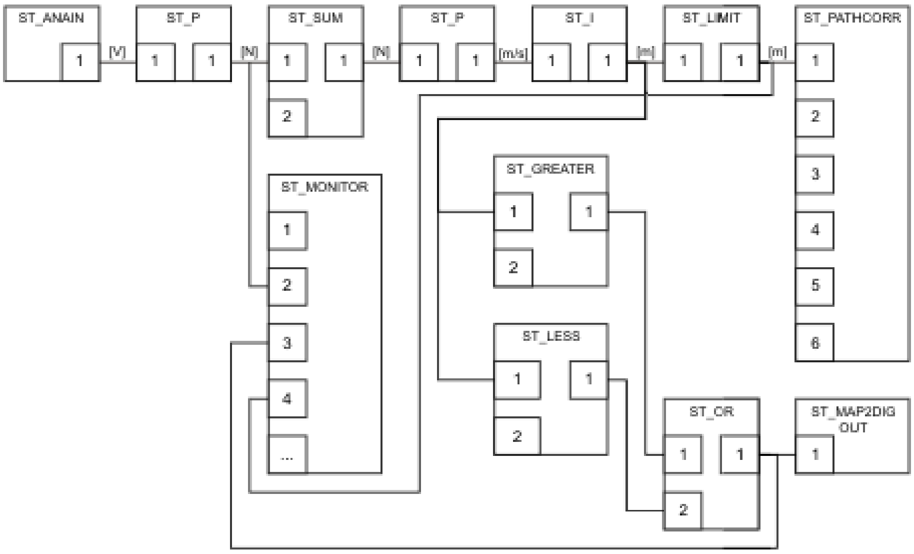
Fig. 6-8: Path of the example sensor application

The following signals are displayed on the RSI monitor:

- Measured value of the force sensor
- Path correction of the robot
- Output for monitoring the maximum permissible deviation from the programmed path

For a real sensor application with force control, the KUKA.ForceTorqueControl technology package, which is optimized for this purpose, is recommended.

Block diagram



RSI object	Description
ST_ANAIN	The RSI object ST_ANAIN records the ±10 V analog voltage supplied by the force sensor and assigns the signal the unit [V].
ST_P	The RSI object ST_P converts the recorded analog voltage to a force value in Newtons and changes the unit of the signal to [N].
ST_SUM	The RSI object ST_SUM calculates the control difference from the deviation between the actual force value and the force setpoint.
ST_P	The RSI object ST_P calculates the correction velocity on the basis of the control difference. The RSI object ST_P also converts the unit of the signal to [m/s].
ST_I	The RSI object ST_I calculates the correction distance by integrating the correction velocity. The RSI object ST_I also adds the calculated correction of the current cycle to the correction of the previous cycle.
ST_LIMIT	The RSI object ST_LIMIT limits the correction distance with a maximum and minimum value.
ST_PATHCORR	The RSI object ST_PATHCORR superposes the correction distance onto the programmed path. The direction in which the path is corrected depends on which signal input of the RSI object ST_PATHCORR is linked.
ST_GREATER ST_LESS	The RSI objects ST_GREATER and ST_LESS compare the correction distance with a constant. The output is set if the correction distance is greater or less than the constant.
ST_OR	The RSI object ST_OR establishes an OR operation between the signal outputs of the RSI objects ST_GREATER and ST_LESS.

RSI object	Description
ST_MAP2DIGOUT	The RSI object ST_MAP2DIGOUT sets a digital output.
ST_MONITOR	The RSI object ST_MONITOR records the signals that are to be displayed with the RSI monitor.

Program code

The program code for the example sensor application can be implemented in several ways; in this example, the program code is implemented as follows:

- The sensor application does not contain subprograms.
- All RSI objects are situated in global container 0.
- The variable for the return values is declared in the declaration section of the SRC file and cannot be read in the variable display.

```
1   DEF RSI_Example( )
2
3   ; Declaration of KRL variables
4   DECL RSIERR RET; Return Value of RSI Commands
5   DECL INT HAI; ObjectID for ST_ANAIN
6   DECL INT HP1; ObjectID for ST_P
7   DECL INT HSUM; ObjectID for ST_SUM
8   DECL INT HP2; ObjectID for ST_P
9   DECL INT HI; ObjectID for ST_I
10  DECL INT HLIM; ObjectID for ST_LIMIT
11  DECL INT HPC; ObjectID for ST_PATHCORR
12  DECL INT HMON; ObjectID for ST_MONITOR
13  DECL INT HGR; ObjectID for ST_GREATER
14  DECL INT HLE; ObjectID for ST_LESS
15  DECL INT HOR; ObjectID for ST_OR
16  DECL INT HM2DO; ObjectID for ST_MAP2DIGOUT
17  DECL CHAR IP[16]; IP- Address for RSIMonitor
18
19  INI
20
21  PTP HOME  Vel= 100 % DEFAULT
22
23  ; Create RSI Context
24  RET=ST_ANAIN(HAI,0,1,RSIUNIT_V)
25  RET=ST_P(HP1,0,HAI,1,1,RSIUNIT_N)
26  RET=ST_SUM(HSUM,0,HP1,1,0,0,50)
27  RET=ST_P(HP2,0,HSUM,1,1,'HF01') ; Unit m/s
28  RET=ST_I(HI,0,HP2,1,0.012,1)
29  RET=ST_LIMIT(HLIM,0,HI,1,-2,2)
30  RET=ST_PATHCORR(HPC,0)
31  RET=ST_NEWLINK(HLIM,1,HPC,1)
32  RET=ST_GREATER(HGR,0,HLIM,1,0,0,1,0.1)
33  RET=ST_LESS(HLE,0,HLIM,1,0,0,-1,0.1)
34  RET=ST_OR(HOR,0,HGR,1,HLE,1)
35  RET=ST_MAP2DIGOUT(HM2DO,0,HOR,1,1,0)
36  IP[]="192.0.1.2"
37  RET=ST_MONITOR(HMON,0,IP[],6000,1)
38  RET=ST_SETPARAM(HMON,1,1)
39  RET=ST_NEWLINK(HLIM,1,HMON,2)
40  RET=ST_NEWLINK(HP2,1,HMON,3)
41  RET=ST_NEWLINK(HOR,1,HMON,4)
42
43  PTP P1 CONT Vel= 100 % PDAT1 Tool[1] Base[1]
44  LIN P2  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
45
46  ; Start RSI execution
47  RET=ST_ON()
48  LIN_REL{Y 1500}
49  RET=ST_OFF()
50
51  LIN P3  Vel= 0.5 m/s CPDAT2 Tool[1] Base[1]
52  PTP HOME  Vel= 100 % DEFAULT
53
51  END
```

Line	Description	Parameter
4	RSIERR variable for the return values of the RSI commands	-----
5...16	INTEGER variables for the IDs of the required RSI objects	-----
17	CHARACTER variable for the IP address of the Windows operating system to enable creation of the RSI monitor	-----

Line	Description	Parameter
24	Creation of the RSI object ST_ANAIN	HAI: object ID
		0: RSI object in container 0
		1: number of the analog input to be read.
		RSIUNIT_V: unit of the output signal
25	Creation of the RSI object ST_P	HP1: object ID
		0: RSI object in container 0
		HAI: ID of the signal source for the signal input
		1: output index of the signal source
		1: scaling factor for the conversion of voltage to force
		RSIUNIT_N: unit of the output signal
26	Creation of the RSI object ST_SUM	HSUM: object ID
		0: RSI object in container 0
		HP1: ID of the signal source for the signal input
		1: output index of the signal source
		0: the second signal input of the RSI object is not assigned
		0: the second signal input of the RSI object is not assigned
		50: constant that is added to the input signal
27	Creation of the RSI object ST_P	HP2: object ID
		0: RSI object in container 0
		HSUM: ID of the signal source for the signal input
		1: output index of the signal source
		0.001: scaling factor for the conversion of force to velocity
		'HF01': coding of the unit m/s according to the unit scheme
28	Creation of the RSI object ST_I	HI: object ID
		0: RSI object in container 0
		HP2: ID of the signal source for the signal input
		1: output index of the signal source
		0.012: integration time in [ms], in which the correction velocity is calculated and added
		1: integration type, to ensure that the integration is only carried out during a CP motion

Line	Description	Parameter
29	Creation of the RSI object ST_LIMIT	HLIM: object ID
		0: RSI object in container 0
		HI: ID of the signal source for the signal input
		1: output index of the signal source
		-2: lower limit of the signal
		2: upper limit of the signal
30	Creation of the RSI object ST_PATHCORR	HPC: object ID
		0: RSI object in container 0
31	Linking of signal input 1 of the RSI object ST_PATHCORR	HLIM: ID of the signal source
		1: output index of the signal source
		HPC: ID of the target object
		3: index of the signal input of the target object
32	Creation of the RSI object ST_GREATER	HGR: object ID
		0: RSI object in container 0
		HLIM: ID of the signal source for the signal input
		1: output index of the signal source
		0: the second signal input of the RSI object is not assigned
		0: the second signal input of the RSI object is not assigned
		1: default value if the second signal input is not assigned
		0.1: hysteresis for the comparison
33	Creation of the RSI object ST_LESS	HLE: object ID
		0: RSI object in container 0
		HLIM: ID of the signal source for the signal input
		1: output index of the signal source
		0: the second signal input of the RSI object is not assigned
		0: the second signal input of the RSI object is not assigned
		-1: default value if the second signal input is not assigned
		0.1: hysteresis for the comparison

Line	Description	Parameter
34	Creation of the RSI object ST_OR	HOR: object ID
		0: RSI object in container 0
		HGR: ID of the signal source for the signal input 1
		1: output index of the signal source
		HLE: ID of the signal source for the signal input 2
		1: output index of the signal source
35	Creation of the RSI object ST_MAP2DIGOUT	HM2DO: object ID
		0: RSI object in container 0
		HOR: ID of the signal source for the signal input
		1: output index of the signal source
		1: number of the digital output that is set
		0: the RSI object ST_MAP2DIGOUT sends a bit to its signal output
36	IP address of the Windows operating system	-----
37	Creation of the RSI object ST_MONITOR	HMON: object ID
		0: RSI object in container 0
		IP[]: IP address of the Win-dows operating system
		6000: port to which the RSI object ST_MONITOR sends the data packets
		1: the signals displayed in the RSI monitor are refreshed every 12 ms
38	Activation of data transfer of the RSI object ST_MONITOR to the RSI monitor	HMON: ID of the RSI object whose object parameters are to be set
		1: number of the object parameter to be set
		1: value to be assigned to the object parameter
39	Linking of signal input 2 of the RSI object ST_MONITOR	HLIM: ID of the signal source for signal input 2
		1: output index of the signal source
		HMON: ID of the target object
		2: index of the signal input of the target object

Line	Description	Parameter
40	Linking of signal input 3 of the RSI object ST_MONITOR	HP2: ID of the signal source for signal input 3
		1: output index of the signal source
		HMON: ID of the target object
		3: index of the signal input of the target object
41	Linking of signal input 4 of the RSI object ST_MONITOR	HGR: ID of the signal source for the signal input 4
		1: output index of the signal source
		HMON: ID of the target object
		4: index of the signal input of the target object
43, 44	Motions to workpiece	-----
47	Activation of signal processing with the ST_ON() command	-----
48	Relative LIN motion	-----
49	Deactivation of signal processing with the ST_OFF() command	-----
51, 52	Motions away from the work-piece	-----

7 Messages

Description

- If an error occurs when an RSI command is executed, the system generates the following message:

  RSI: Error in Function <X>

  The X in the message stands for the name of the RSI command that was being executed when the error occurred.
- If the signal processing results in a larger path correction than the limitation defined by the RSI objects ST_PATHCORR and ST_AXISCORR will permit, the correction is rejected and the following message is generated:
  - For a Cartesian path correction (ST_PATHCORR): **SEN: ST_PATHCORR – correction out of range xxx**
  - For an axis angle correction (ST_AXISCORR): **SEN: ST_AXISCORR – correction out of range xxx**

  If the set correction range is not sufficient for the process, it can be adapted. To do so, the RSI command ST_SETPARAM is used to assign correspondingly adapted values to the object parameters of the RSI objects ST_PATHCORR or ST_AXISCORR.

  (>>> 6.11 "Example program for adapting the maximum path correction" page 28)

LOG file

A LOG file is created under C:\KRC\ROBOTER\LOG.

8 Diagnosis

8.1 Overview of diagnosis

Description

The following options are available for diagnosis of an error within the RSI context:

- Signal visualization with the RSI monitor (>>> 8.2 "Overview of signal visualization" page 41).
- Displaying information about the RSI commands on Telnet and/or in LOG files (>>> 8.3 "Displaying RSI information on Telnet and/or in LOG files, overview" page 47).

8.2 Overview of signal visualization

Overview

Step	Visualization of signals on the robot controller	Visualization of signals on an external PC
1	Configure the RSI monitor. (>>> 8.2.4 "Configuring RSI monitor" page 43)	Install KUKA.Router on the robot controller. (>>> 8.2.2 "Installing KUKA.Router" page 42)
2	Visualize signals on the robot controller. (>>> 8.2.5 "Visualizing signals on the robot controller" page 44)	Configure KUKA.Router. (>>> 8.2.3 "Configuring KUKA.Router" page 42)
3	-----	Copy the program ...\KRC\TP\RSI\UTIL\RSI-Monitor.exe (RSI monitor) from the robot controller to the external PC.
4	-----	Configure the RSI monitor. (>>> 8.2.4 "Configuring RSI monitor" page 43)
5	-----	Visualize signals on external PC. (>>> 8.2.6 "Visualizing signals on an external PC" page 45)

8.2.1 RSI monitor

Description

The RSI monitor can record and visualize up to 24 signals from the RSI context. The signals can be recorded directly to the robot controller by the RSI monitor **or** via a network to an external PC.

If the signals are recorded by the RSI monitor on an external PC via a network, the KUKA.Router program is required.

If the signals are recorded directly on the robot controller, they cannot be viewed until the program has been terminated. If an external PC is used, the signals can be viewed while the program is being executed.

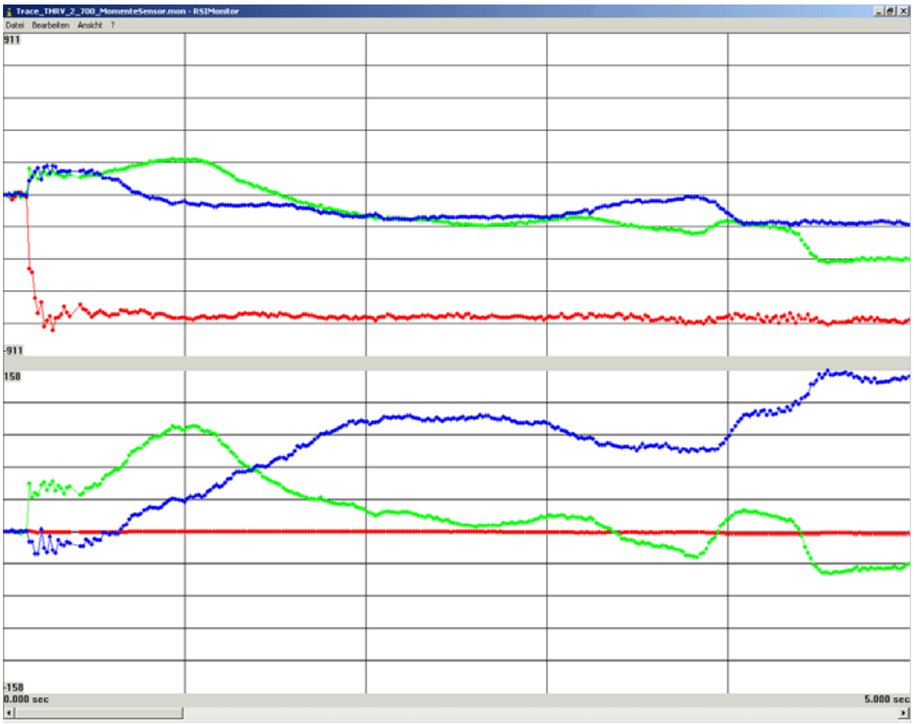


Fig. 8-1: Example of a recording with the RSI monitor

The data required for the recording are acquired using the RSI object ST_MONITOR and sent to the RSI monitor. The signals to be visualized are applied to signal inputs 2...25. The recorded signals can be distributed over up to 5 diagrams.

The object parameter ID 1 of the RSI object ST_MONITOR activates the data transfer. Signal input 1 of the RSI object ST_MONITOR can be used additionally to control the activated data transfer by means of a Boolean signal.

8.2.2 Installing KUKA.Router

Precondition

- Network connection in Windows between robot controller and external PC

> Further information about KUKA.Router can be found in the KUKA_Router directory on the CD-ROM.

Procedure

1. Start the program **Setup.exe** in the KUKA_Router directory on the CD-ROM.

   The installation procedure then runs automatically.
2. Reboot the robot controller.

8.2.3 Configuring KUKA.Router

Precondition

- User group “Expert”
- Windows interface (CTRL+ESC)

Procedure

1. On the Windows interface, open the program KUKA.Router by selecting the menu sequence **Start** > **Programs** > **Startup** > **Router**.
2. Add a new route using the menu sequence **File** > **New**.

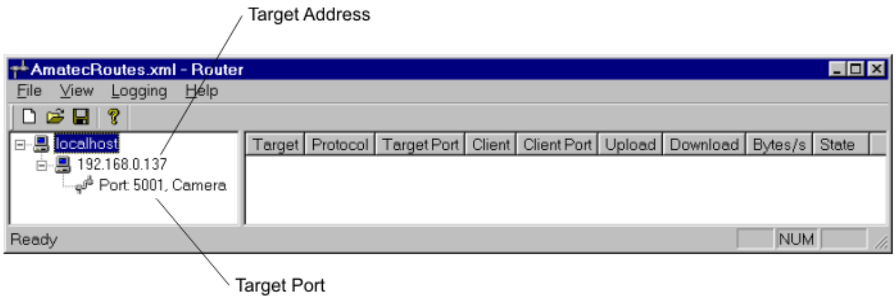


Fig. 8-2: KUKA.Router program

3. Use the arrow keys to move the focus to the IP address, and press the Enter key.

   The **Configure target** window is opened.
4. In the **Target host** box, enter the target address and press OK.
5. Use the arrow keys to move the focus to the port, and press the Enter key.

   The **Configure route** window is opened.
6. Enter the source port in the **Source Port** box.
7. Enter the target port in the **Target Port** box, and press OK.
8. Minimize the **KUKA.Router** window.

Do not close the program, as otherwise no data will be sent.

8.2.4 Configuring RSI monitor

Precondition

- If the RSI monitor is configured on the robot controller: user group “Expert”.

Procedure

1. Select the menu sequence **Monitor** > **RSIMonitor**. The RSI monitor is opened.
2. In the RSI monitor, select the menu sequence **Edit** > **Settings**.
3. In the **Settings** window, configure the RSI monitor.

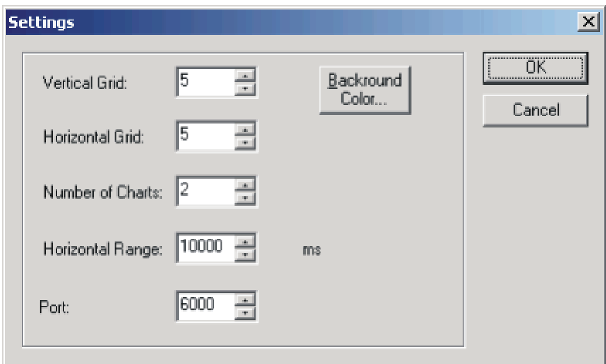


Fig. 8-3: “Settings” window of the RSI monitor

Parameter	Description
Vertical Grid	Number of vertical lines per displayed diagram in the RSI monitor
Horizontal Grid	Number of horizontal lines per displayed diagram in the RSI monitor
Number of Charts	Number of diagrams displayed in the RSI monitor

Parameter	Description
Horizontal Range	Value for the time axis of the RSI monitor in ms
Port	Number of the port at which the RSI monitor is expecting data packets The value of the port must correspond to the port set in KUKA.Router.

4. Accept the settings by pressing **OK**.
5. To modify the properties of the lines for the signals, select the menu sequence **Edit** > **Line Properties...** in the RSI monitor.
6. In the **Line Properties** window, modify the lines for the signals as required.

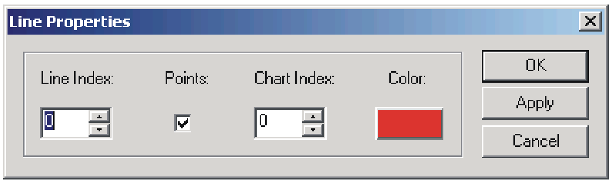


Fig. 8-4: “Line Properties” window of the RSI monitor

Parameter	Description
Line Index	Number of the signal diagram for which the properties are to be changed. 'Line Index 0' corresponds to signal diagram for signal input 2. 'Line Index 23' corresponds to signal diagram for signal input 25.
Points	Check box activated: a point is set in the selected signal diagram in the RSI monitor after every measurement cycle. Check box not activated: the selected signal diagram is displayed as a line in the RSI monitor.
Chart Index	Number of the chart in which the signal diagram is displayed. The 'Chart Index' automatically changes if the 'Line Index' is changed. By default, 3 signal diagrams are displayed per chart.
Color	Color of the selected signal diagram.

7. Accept the settings by pressing **OK**.

8.2.5 Visualizing signals on the robot controller

Precondition

- User group “Expert”.
- RSI objects and containers must be created.
- All vital inputs must be linked.

Procedure

1. Open the SRC file of the program.
2. Create the RSI object ST_MONITOR in the SRC file and apply the required signals to the signal inputs of the RSI object ST_MONITOR (>>> 8.2.7 "Example of signal visualization" page 46).
3. Close and save the SRC file.

4. Select the menu sequence **Monitor** > **RSIMonitor**. The RSI monitor is opened.
5. Set the following elements in the user interface of the System Software:
   - Operating mode
   - Program run mode
   - Program override
6. Select a program and execute it to the end of the program.

   The recording of the RSI monitor starts and finishes automatically with the selected program.

If the program is reset, the recording of the RSI monitor is also deleted.

7. To save the recording as a MON file, select the menu sequence **File** > **Save as...** in the RSI monitor and specify a target folder and file name.
8. To save the recording as a DAT or TXT file, select the menu sequence **File** > **Export** in the RSI monitor and specify a target folder and file name.
9. To open a recording, select the menu sequence **File** > **Open** and the desired MON file.

Only MON files can be opened in the RSI monitor.

8.2.6 Visualizing signals on an external PC

Precondition

- Program must be created.
- User group “Expert”.
- All vital inputs must be linked.
- Signal processing must be fully created.
- Windows network between robot controller and external PC must be established and configured.

Procedure

1. Open the SRC file of the program.
2. Create the RSI object ST_MONITOR in the SRC file and apply the required signals to the signal inputs of the RSI object ST_MONITOR.

   (>>> 6.1 "Programming overview" page 17)

   (>>> 8.2.7 "Example of signal visualization" page 46)
3. Close and save the SRC file.
4. Start the RSIMonitor.exe program on the external PC.
5. Set the following elements in the user interface of the System Software:
   - Operating mode
   - Program run mode
   - Program override
6. Select a program and execute it to the end of the program.

   The recording of the RSI monitor starts and finishes automatically with the selected program.

If the program is reset, the recording of the RSI monitor is also deleted.

7. To save the recording as a MON file, select the menu sequence **File** > **Save as...** in the RSI monitor and specify a target folder and file name.

8. To save the recording as a DAT or TXT file, select the menu sequence **File** > **Export** in the RSI monitor and specify a target folder and file name.
9. To open a recording, select the menu sequence **File** > **Open** and the desired MON file.

> Only MON files can be opened in the RSI monitor.

8.2.7 Example of signal visualization

Example

```
1   DEF Program( )
2   DECL RSIERR RET
3   DECL INT
OBJECT1_ID,OBJECT2_ID,OBJECT3_ID,OBJECT4_ID,OBJECT5_ID,PORT,CYCLE
4   DECL CHAR IP[16]
5
6   INI
7
8   PTP HOME  Vel= 100 % DEFAULT
9
10  RET=ST_DIGOUT(OBJECT1_ID,0,1,0,RSIUNIT_No)
11  RET=ST_DIGOUT(OBJECT2_ID,0,2,0,RSIUNIT_No)
12  RET=ST_OR(OBJECT3_ID,0,OBJECT1_ID,1,OBJECT2_ID,1)
13  RET=ST_MAP2DIGOUT(OBJECT4_ID,0,OBJECT3_ID,1,3,0)
14
15  IP[]="192.0.1.2"
16  PORT=6000
17  CYCLE=1
18
19  RET=ST_MONITOR(OBJECT5_ID,0,IP[],PORT,CYCLE)
20  RET=ST_NEWLINK(OBJECT1_ID,1,OBJECT5_ID,2)
21  RET=ST_SETPARAM(OBJECT5_ID,1,1)
22
23  RET=ST_ON()
24
25  Program2()
26
27  RET=ST_SETPARAM(OBJECT5_ID,1,0)
28
29  RET=ST_OFF()
30
31  PTP HOME  Vel= 100 % DEFAULT
```

Line	Description
3	Required INTEGER variables for the RSI object ST_MONITOR
4	CHARACTER variable for the IP address of the Windows operating system
15	IP address of the Windows operating system
16	Port to which the RSI object ST_MONITOR sends the data packets
17	Factor for the cycle time in which the RSI object ST_MONITOR sends the data packages (CYCLE=1 corresponds to 12 ms, CYCLE=2 corresponds to 24 ms, etc.)
19	Creation of the RSI object ST_MONITOR
20	Link to signal input 2 of the RSI object ST_MONITOR
21	The ST_SETPARAM command is used to set the object parameter of the RSI object ST_MONITOR and to activate transmission of the data packets
27	The ST_SETPARAM command is used to set the object parameter of the RSI object ST_MONITOR and to deactivate transmission of the data packets

8.3 Displaying RSI information on Telnet and/or in LOG files, overview

Description

Information about the RSI commands can be displayed on Telnet and/or in RSI-specific LOG files. The RSI-specific error messages are always displayed on Telnet and in the LOG file rsiErrors.log. The following RSI-specific LOG files are located in the folder ...\KRC\ROBOTER\LOG:

LOG file	Description
rsiErrors.log	Only contains the RSI-specific error messages, irrespective of the filter settings.
rsiAll.log	Contains information about each RSI command in accordance with the filter settings.

The entries in the LOG files **cannot** be deleted. When the LOG file reaches a file size of 200 kB, a backup file is created in the folder ...\KRC\ROBOTER\LOG.
Example: backup file of rsiAll.log is renamed rsiAll.bkp.

Overview

Step	Display information on Telnet	Write information to the LOG files	Display information on Telnet and in LOG files
1	Set filter for displaying the information. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)	Set filter for displaying the information. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)	Set filter for displaying the information. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)
2	Open Telnet. (>>> 8.3.2 "Opening Telnet" page 48)	Select and execute program.	Open Telnet. (>>> 8.3.2 "Opening Telnet" page 48)
3	Select and execute program.	Reset program in order to be able to write information to the LOG files.	Select and execute program.
4	Reset filter for displaying the information to the default setting. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)	Reset filter for displaying the information to the default setting. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)	Reset program in order to be able to write information to the LOG files.
5	-----	-----	Reset filter for displaying the information to the default setting. (>>> 8.3.1 "Configuring the filter for displaying information" page 48)

If this diagnostic function is no longer required, the filter must be reset to the default setting. If the filter is not reset, this can have an adverse effect on system performance during operation.

8.3.1 Configuring the filter for displaying information

Precondition

- User group “Expert”.
- Operating mode T1 or T2.
- No program may be selected.

Procedure

1. Open the file ...\KRC\ROBOTER\INIT\amSysObj.ini.
2. In the [ALog] section, assign the Filter parameter the value for the required information display type.

```
[ALog]
MQueueSize=1000; Message queue size
Filter=0x0      ; 0x73F for all types & shell, file1, file2
```

Value for Filter	Description
0x0	**Only** the RSI-specific error messages are displayed on Telnet and in the LOG file rsiErrors.log. Default setting
0x33F	The information about **every** RSI command is displayed on Telnet.
0x63F	The information about **every** RSI command is displayed in the LOG file rsiAll.log.
0x73F	The information about **every** RSI command is displayed on Telnet and in the LOG file rsiAll.log.

3. Close and save file.
4. Select the menu sequence **Configure** > **I/O Driver** > **Reconfigure I/O Driver**.

8.3.2 Opening Telnet

Procedure

1. Click on the Windows **Start** button.
2. Select the menu option **Run...**.
3. In the **Open** box, enter.
   - Windows 95: **Telnet 192.0.1.1**
   - Windows XP Security Patch 2 or higher: **Telnetk 192.0.1.1**
4. Click on **OK**.
   The Telnet window is opened.

In all Telnet entries: observe upper/lower case!

9 KUKA Service

9.1 Requesting support

Introduction

The KUKA Robot Group documentation offers information on operation and provides assistance with troubleshooting. For further assistance, please contact your local KUKA subsidiary.

Faults leading to production downtime are to be reported to the local KUKA subsidiary within one hour of their occurrence.

Information

The following information is required for processing a support request:

- Model and serial number of the robot
- Model and serial number of the controller
- Model and serial number of the linear unit (if applicable)
- Version of the KUKA System Software
- Optional software or modifications
- Archive of the software
- Application used
- Any external axes used
- Description of the problem, duration and frequency of the fault

9.2 KUKA Customer Support

Availability

KUKA Customer Support is available in many countries. Please do not hesitate to contact us if you have any questions.

Argentina

Ruben Costantini S.A. (Agency)
Luis Angel Huergo 13 20
Parque Industrial
2400 San Francisco (CBA)
Argentina
Tel. +54 3564 421033
Fax +54 3564 428877
ventas@costantini-sa.com

Australia

Marand Precision Engineering Pty. Ltd. (Agency)
153 Keys Road
Moorabbin
Victoria 31 89
Australia
Tel. +61 3 8552-0600
Fax +61 3 8552-0605
robotics@marand.com.au

Austria	KUKA Roboter GmbH Vertriebsbüro Österreich Regensburger Strasse 9/1 4020 Linz Austria Tel. +43 732 784752 Fax +43 732 793880 office@kuka-roboter.at www.kuka-roboter.at
Belgium	KUKA Automatisering + Robots N.V. Centrum Zuid 1031 3530 Houthalen Belgium Tel. +32 11 516160 Fax +32 11 526794 info@kuka.be www.kuka.be
Brazil	KUKA Roboter do Brasil Ltda. Avenida Franz Liszt, 80 Parque Novo Mundo Jd. Guançã CEP 02151 900 São Paulo SP Brazil Tel. +55 11 69844900 Fax +55 11 62017883 info@kuka-roboter.com.br
Chile	Robotec S.A. (Agency) Santiago de Chile Chile Tel. +56 2 331-5951 Fax +56 2 331-5952 robotec@robotec.cl www.robotec.cl
China	KUKA Flexible Manufacturing Equipment (Shanghai) Co., Ltd. Shanghai Qingpu Industrial Zone No. 502 Tianying Rd. 201712 Shanghai P.R. China Tel. +86 21 5922-8652 Fax +86 21 5922-8538 Franz.Poeckl@kuka-sha.com.cn www.kuka.cn

France	KUKA Automatisme + Robotique SAS Techvallée 6 Avenue du Parc 91140 Villebon s/Yvette France Tel. +33 1 6931-6600 Fax +33 1 6931-6601 commercial@kuka.fr www.kuka.fr
Germany	KUKA Roboter GmbH Blücherstr. 144 86165 Augsburg Germany Tel. +49 821 797-4000 Fax +49 821 797-1616 info@kuka-roboter.de www.kuka-roboter.de
Hungary	KUKA Robotics Hungaria Kft. Fö út 140 2335 Taksony Hungary Tel. +36 24 501609 Fax +36 24 477031 info@kuka-robotics.hu
India	KUKA Robotics, Private Limited 621 Galleria Towers DLF Phase IV 122 002 Gurgaon Haryana India Tel. +91 124 4148574 info@kuka.in www.kuka.in
Italy	KUKA Roboter Italia S.p.A. Via Pavia 9/a - int.6 10098 Rivoli (TO) Italy Tel. +39 011 959-5013 Fax +39 011 959-5141 kuka@kuka.it www.kuka.it

Korea	KUKA Robot Automation Korea Co. Ltd. 4 Ba 806 Sihwa Ind. Complex Sung-Gok Dong, Ansan City Kyunggi Do 425-110 Korea Tel. +82 31 496-9937 or -9938 Fax +82 31 496-9939 info@kukakorea.com
Malaysia	KUKA Robot Automation Sdn Bhd South East Asia Regional Office No. 24, Jalan TPP 1/10 Taman Industri Puchong 47100 Puchong Selangor Malaysia Tel. +60 3 8061-0613 or -0614 Fax +60 3 8061-7386 info@kuka.com.my
Mexico	KUKA de Mexico S. de R.L. de C.V. Rio San Joaquin #339, Local 5 Colonia Pensil Sur C.P. 11490 Mexico D.F. Mexico Tel. +52 55 5203-8407 Fax +52 55 5203-8148 info@kuka.com.mx
Norway	KUKA Sveiseanlegg + Roboter Bryggeveien 9 2821 Gjövik Norway Tel. +47 61 133422 Fax +47 61 186200 geir.ulsrud@kuka.no
Portugal	KUKA Sistemas de Automatización S.A. Rua do Alto da Guerra n° 50 Armazém 04 2910 011 Setúbal Portugal Tel. +351 265 729780 Fax +351 265 729782 kuka@mail.telepac.pt

Russia	KUKA-VAZ Engineering Jushnoje Chaussee, 36 VAZ, PTO 445633 Togliatti Russia Tel. +7 8482 391249 or 370564 Fax +7 8482 736730 Y.Klychkov@VAZ.RU
South Africa	Jendamark Automation LTD (Agency) 76a York Road North End 6000 Port Elizabeth South Africa Tel. +27 41 391 4700 Fax +27 41 373 3869 www.jendamark.co.za
Spain	KUKA Sistemas de Automatización S.A. Pol. Industrial Torrent de la Pastera Carrer del Bages s/n 08800 Vilanova i la Geltrú (Barcelona) Spain Tel. +34 93 814-2353 Fax +34 93 814-2950 Comercial@kuka-e.com www.kuka-e.com
Sweden	KUKA Svetsanläggningar + Robotar AB A. Odhners gata 15 421 30 Västra Frölunda Sweden Tel. +46 31 7266-200 Fax +46 31 7266-201 info@kuka.se
Switzerland	KUKA Roboter Schweiz AG Riedstr. 7 8953 Dietikon Switzerland Tel. +41 44 74490-90 Fax +41 44 74490-91 info@kuka-roboter.ch www.kuka-roboter.ch

Taiwan
KUKA Robot Automation Taiwan Co. Ltd.
136, Section 2, Huanjung E. Road
Jungli City, Taoyuan
Taiwan 320
Tel. +886 3 4371902
Fax +886 3 2830023
info@kuka.com.tw
www.kuka.com.tw

Thailand
KUKA Robot Automation (M)SdnBhd
Thailand Office
c/o Maccall System Co. Ltd.
49/9-10 Soi Kingkaew 30 Kingkaew Road
Tt. Rachatheva, A. Bangpli
Samutprakarn
10540 Thailand
Tel. +66 2 7502737
Fax +66 2 6612355
atika@ji-net.com
www.kuka-roboter.de

UK
KUKA Automation + Robotics
Hereward Rise
Halesowen
B62 8AN
UK
Tel. +44 121 585-0800
Fax +44 121 585-0900
sales@kuka.co.uk

USA
KUKA Robotics Corp.
22500 Key Drive
Clinton Township
48036 Michigan
USA
Tel. +1 866 8735852
Fax +1 586 5692087
info@kukarobotics.com
www.kukarobotics.com

Index