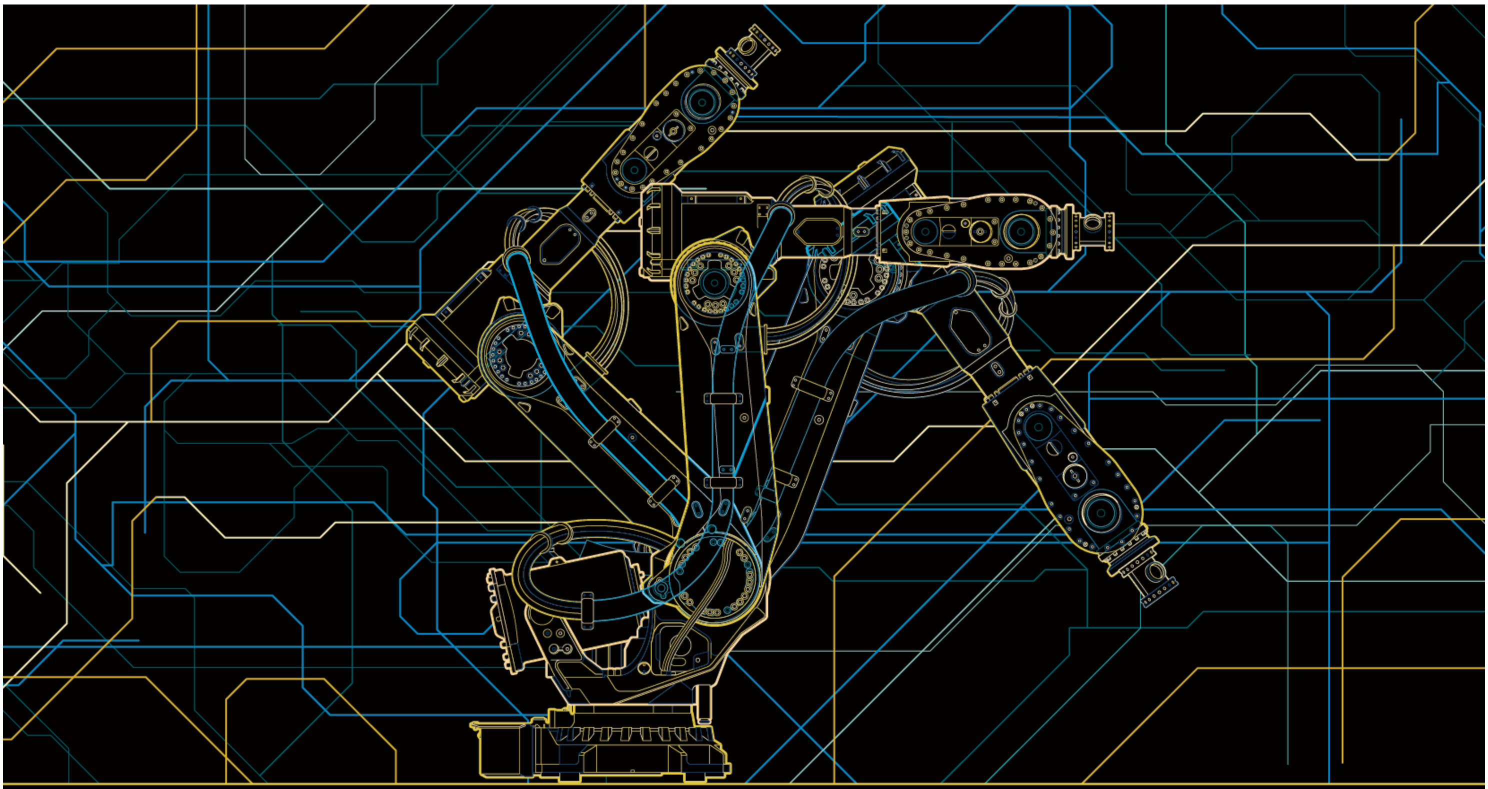


Technical reference manual
System parameters

Power and productivity
for a better world™

Trace back information:
Workspace R15-2 version a14
Checked in 2015-10-07
Skribenta version 4.6.081

Technical reference manual
System parameters

RobotWare 6.02

Document ID: 3HAC050948-001
Revision: C

ABB AB
Robotics Products
Se-721 68 Västerås
Sweden

Table of contents

Overview of this manual

About this manual

This manual describes the IRC5 system parameters by topic and type in an overview. It also covers some basic workflow descriptions on how to add, edit and delete parameters. This can be done via specific software tools, which are not described here, nor how to use them.

The manual covers the most common types and parameters in the topics *Communication*, *Controller*, *I/O System*, *Man-machine communication*, and *Motion*.

Usage

This manual should be used as a reference during configuration of the robot system.

The manual includes parameters for both the basic robot system and selected software and hardware options. The option parameters require that you have the specified option installed in your robot system.

It is recommended that you create a backup or save the configuration files before changing any parameters.

Note

This should only be performed by a trained technician.

Who should read this manual?

This manual is intended for:

- production technicians
- programmers
- service technicians

Prerequisites

The reader should be familiar with:

- industrial robots and terminology.
- the RAPID programming language.
- how to configure system parameters using RobotStudio or FlexPendant.

References

The manual contains references to the following information products:

Reference	Document ID
Operating manual - Getting started, IRC5 and RobotStudio	*3HAC027097-001*
Operating manual - IRC5 with FlexPendant	*3HAC050941-001*
Operating manual - RobotStudio	*3HAC032104-001*
Operating manual - Trouble shooting IRC5	*3HAC020738-001*
Operating manual - Calibration Pendulum	*3HAC16578-1*
Operating manual - Service Information System	*3HAC050944-001*

Continues on next page

Reference	Document ID
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC050917-001*
Technical reference manual - RAPID overview	*3HAC050947-001*
Technical reference manual - RAPID kernel	*3HAC050946-001*
Application manual - Additional axes and stand alone controller	*3HAC051016-001*
Application manual - DeviceNet Master/Slave	*3HAC050992-001*
Application manual - PROFIBUS Controller	*3HAC050966-001*
Application manual - EtherNet/IP Scanner/Adapter	*3HAC050998-001*
Application manual - PROFINET Controller/Device	*3HAC050969-001*
Application manual - Controller software IRC5	*3HAC050798-001*
Application manual - MultiMove	*3HAC050961-001*

Revisions

The following revisions of this manual have been released.

Revision	Description
-	First edition. Released with RobotWare 6.0.

A	Released with RobotWare 6.01.
	• Added a new chapter *Type T10 Function Keys on page 370* under topic *Man-machine Communication*.
	• Added the following new system parameters under topic *Motion*:
	  - Added new system parameters *Type Arm Check Point on page 427* and *Check Point Bound Limit Outside Cube on page 686*.
	  - *Enable high accuracy position synchronization on page 603*.

B	Released with RobotWare 6.01. Changes in *Topic Communication on page 29*: • Added the type *Ethernet Port*. • Added the type *IP Setting*. • Added the type *IP Route*. • Added the type *Static VLAN*. • The type *Physical Channel* is renamed *Serial Port*. • Added parameter *Remote port number* in type *Transmission Protocol*. Changes in *Topic I/O System on page 173*. • System parameter *Connection* removed from *Industrial Network*, since it is only used for some communication protocols. *Connection* is described in the application manuals where it is used. • System parameter *Address* removed from *Industrial Network*, since it is only used for some communication protocols. *Address* is described in the application manuals where it is used.

Continues on next page

Revision	Description
C	Released with RobotWare 6.02. Minor corrections in I/O System section. Added a new type *DNS Client* in the topic *Communication*. See *Type DNS Client on page 50*. Added the following new system parameters under topic *I/O System*: • *Collision Avoidance on page 299*. • *Absolute Accuracy Active on page 316*. • *CPU Fan not Running on page 342*. • *SMB Battery Charge Low on page 346*. • *Temperature Warning on page 345*. Added the following new system parameters under topic *Motion*: • *Global Speed Limit on page 697*. • *Arm Check Point Speed Limit on page 685*. • *Coll-Pred Safety Distance on page 632*. • *Force Detection Min Time on page 482*. • *Setup optimized start from finepoint on page 604*. • *Arm-Angle Reference Direction on page 698*.

Product documentation, IRC5

Categories for user documentation from ABB Robotics

The user documentation from ABB Robotics is divided into a number of categories. This listing is based on the type of information in the documents, regardless of whether the products are standard or optional.

All documents listed can be ordered from ABB on a DVD. The documents listed are valid for IRC5 robot systems.

Product manuals

Manipulators, controllers, DressPack/SpotPack, and most other hardware is delivered with a **Product manual** that generally contains:

- Safety information.
- Installation and commissioning (descriptions of mechanical installation or electrical connections).
- Maintenance (descriptions of all required preventive maintenance procedures including intervals and expected life time of parts).
- Repair (descriptions of all recommended repair procedures including spare parts).
- Calibration.
- Decommissioning.
- Reference information (safety standards, unit conversions, screw joints, lists of tools).
- Spare parts list with exploded views (or references to separate spare parts lists).
- Circuit diagrams (or references to circuit diagrams).

Technical reference manuals

The technical reference manuals describe reference information for robotics products.

- *Technical reference manual - Lubrication in gearboxes*: Description of types and volumes of lubrication for the manipulator gearboxes.
- *Technical reference manual - RAPID overview*: An overview of the RAPID programming language.
- *Technical reference manual - RAPID Instructions, Functions and Data types*: Description and syntax for all RAPID instructions, functions, and data types.
- *Technical reference manual - RAPID kernel*: A formal description of the RAPID programming language.
- *Technical reference manual - System parameters*: Description of system parameters and configuration workflows.

Continues on next page

Application manuals

Specific applications (for example software or hardware options) are described in **Application manuals**. An application manual can describe one or several applications.

An application manual generally contains information about:

- The purpose of the application (what it does and when it is useful).
- What is included (for example cables, I/O boards, RAPID instructions, system parameters, DVD with PC software).
- How to install included or required hardware.
- How to use the application.
- Examples of how to use the application.

Operating manuals

The operating manuals describe hands-on handling of the products. The manuals are aimed at those having first-hand operational contact with the product, that is production cell operators, programmers, and trouble shooters.

The group of manuals includes (among others):

- *Operating manual - Emergency safety information*
- *Operating manual - General safety information*
- *Operating manual - Getting started, IRC5 and RobotStudio*
- *Operating manual - Introduction to RAPID*
- *Operating manual - IRC5 with FlexPendant*
- *Operating manual - RobotStudio*
- *Operating manual - Trouble shooting IRC5*, for the controller and manipulator.

This page is intentionally left blank

1 About system parameters

1.1 About system parameters

Overview

System parameters describe the configuration of the robot system. The parameters are configured according to order on delivery.

By changing the parameters values, the performance of the system can be adjusted. The system parameters usually only need changing if the robot system is modified due to a changed process.

Parameter structure

The parameters are grouped together in a number of different configuration areas, named topics. These topics are divided into different types of parameters.

For each type, a number of objects or instances can be defined, thus having the same type. Each such instance has a number of parameters, which must be given specific values. In some cases these parameters, depending on their values, are further structured in subparameters, also called arguments or action values.

Topic definition

A topic is a configuration area with a specific collection of parameters.

There are six topics in the controller, each describing different areas in the robot system. All parameters are stored in a separate configuration file for each topic. These are known as cfg files (file extension .cfg). See *Configuration files on page 27*.

Type definition and type instances

A type is a section of a topic, which defines parameters of the same type. As indicated above, there can be many instances of the same type. All such instances are referred to with the name of the type. For example, an instance of the type *Signal* is called a Signal instance or just a Signal. Note that each separate signal instance has a unique name, for example digin1.

Some of the instances may be shown in the system configuration for display only purposes and are therefore read-only. They belong to the default configuration of the system and can not be modified. In the RobotStudio editor they are grayed-out and on the FlexPendant they are marked with a separate icon. Read-only instances are never stored in the customer configuration files, when a topic is stored in a cfg file.

System parameters definition

All parameters of an instance are assigned a value to describe the robot system configuration.

The parameter values are normally predefined on delivery. The values are restricted to data type, and sometimes to be within an interval, which is described for each parameter in *Technical reference manual - System parameters*.

Most parameters require a restart of the controller to take effect after being changed.

Continues on next page

Some parameters are visible but not editable since they are a part of the system and should not be changed.

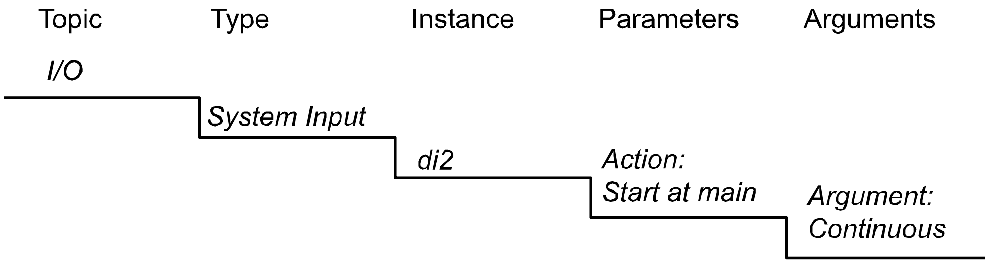
Working with system parameters

System parameters are configured using RobotStudio or the FlexPendant. This is detailed in *Operating manual - RobotStudio* and *Operating manual - IRC5 with FlexPendant*.

Experienced users can also edit the configuration files. In the configuration files all types and parameters have specific names. To help working with such text files these names are indicated in the parameter descriptions under the caption "Cfg name".

Example illustration

This example illustrates the structure from topic, down to arguments (also called action values).



en0800000183

1.2 Configuration files

Configuration files

A configuration file is a text file that lists the values of system parameters. Note that if the parameter is assigned the default value, then it will not be listed in the configuration file.

There are six configuration areas in the controller, saved as configuration files (*.cfg).

Configuration files are by default saved in the folder SYSPAR for the system, for example ..\MySystem\SYSPAR\.

The configuration files are included in backups.

Note

Configuration files shall not be loaded in systems running an older RobotWare version than in which they were created.

Topic:	Configuration area:	Configuration file:
Communication	Serial channels and file transfer protocols	SIO.cfg
Controller	Safety and RAPID specific functions	SYS.cfg
I/O	I/O boards and signals	EIO.cfg
Man-machine communication	Functions to simplify working with the system	MMC.cfg
Motion	The robot and external axes	MOC.cfg
Process	Process specific tools andequipment	PROC.cfg

Example

This is an example from SIO.cfg, topic *Communication*.

```
#
COM_PHY_CHANNEL:
  -Name "COM1" -Connector "COM1"
#
COM_TRP:
  -Name "TCPIP1" -Type "TCP/IP" -PhyChannel "COM1"
```

Explanation

One instances of the type *Serial Port* with:

- *Name* defined as COM1 and *Connector* defined as COM1.

One instance of the type *Transmission Protocol* with:

- *Name* defined as TCPIP1, *Type* defined as TCP/IP, and *Physical Channel* defined as COM1.

1.3 File system

Overview

This section describes how paths on the controller can be defined using enviroment variables.

Examples of paths

Environment variables

Path	Description
BACKUP/my_dir	The backup folder, i.e., */<system_partition>/BACKUP/my_dir*
HOME/my_dir	The home folder in the active system, i.e., */<system_partition>/<system_name>/HOME/my_dir*
SYSTEM/my_dir	The active system folder, i.e., */<system_partition>/<system_name>/my_dir*
SYSTEM_PARTITION/my_dir	The root of the system partition on the controller, i.e., */<system_partition>/my_dir*
REMOVABLEDISK1/my_dir	USB device on the controller.
REMOVABLEDISK2/my_dir	Second USB device on the controller.

The environment variables in the examples exist by default in the system. An environment variable is only detected if it is placed first in a path.

Current directory

Current directory is not defined but varies depending on what happens in the system. Therefore, all references should be defined with complete paths (or using environment variables).

Mounted disks

To be able to use mounted disks in the paths, there must be an FTP or NFS connection to a running FTP/NFS server with read and write access to the directory. In the following example, the mounted disk is named pc:

```
pc:/my_dir
```

Related information

Backup on page 269

Load on page 276

Load and Start on page 278

2 Topic Communication

2.1 The Communication topic

Overview

This chapter describes the types and parameters of the *Communication* topic. Each parameter is described in the section for its type.

Description

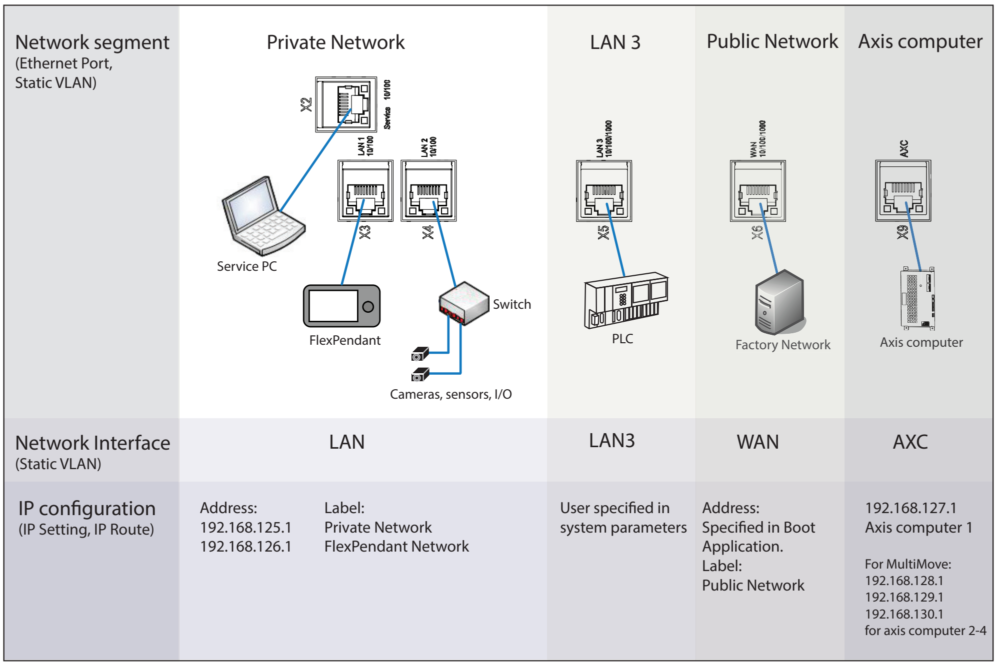
The Communication topic contains parameters for configuring the main computer's connectivity using serial and Ethernet ports.The parameters are organized in the following types:

1. Application protocol
2. Ethernet Port
3. IP Setting
4. Serial Port
5. IP Route
6. Static VLAN
7. Transmission protocol

2.2 The relation between physical Ethernet ports and system parameters

Ethernet ports and system parameters

The Ethernet ports on the main computer belong to network segments according to the following illustration.



xx1500000395

Network segments

Ports X2 (Service), X3 (LAN 1), and X4 (LAN 2) belong to the "Private Network" segment. Depending on the configuration, X5 (LAN 3) can also be part of the Private Network segment, see *How to configure LAN 3 to be part of private network on page 34*. Private Network segments of multiple robot controllers cannot be connected to each other.

By default, the X5 (LAN 3) port is configured as an isolated network. This allows the robot controller to be connected to an external network. Typically a PLC controlling several robot controllers is connected on LAN 3.

X6 (WAN) belongs to the "Public Network" segment. This is for connecting the robot controller to an external network (factory network). Typically the Public Network segment is used for:

- connect a PC running RobotStudio
- using FTP clients
- mounting FTP or NFS disks from the controller
- running Ethernet based fieldbuses

X9 (AXC) is always connected to the axis computer. If MultiMove is used, AXC is connected to a switch that connects to all the axis computers.

Continues on next page

Network interfaces

There is a one-to-one relationship between network segment and *Interface*. The instances of *Interface* are predefined according to the above figure. They are: LAN, WAN, AXC, and LAN3 (unless LAN 3 is configured to be part of the Private Network segment).

IP configuration

IP Setting specifies an IP address for the *Interface*.

One *Interface* can have more than one *IP Setting* for multiple addresses to the same network segment. In that case, a main computer network interface becomes multi-homed on multiple IP subnets running on the same physical network segment.

The LAN *Interface* has two predefined instances of *IP Setting*, Private Network and FlexPendant Network.

LAN 3 does not have any predefined *IP Setting*. Users have to create their own settings for LAN 3.

WAN has a predefined *IP Setting*, Public Network, but its address depends on what is set in the Boot Application.

AXC has a *IP Setting* called Axis computer 1. If the option MultiMove is used, there is one *IP Setting* for each axis computer.

In addition to the existing instances of *IP Setting*, the user can add new ones as desired, except for the Axis computer interface.

IP addresses

Predefined networks

The following addresses are taken by the predefined networks.

IP address range	Network
192.168.125.0 - 255	Private Network
192.168.126.0 - 255	FlexPendant Network (same network segment as Private Network)
192.168.127.0 - 255	Axis computer 1
192.168.128.0 - 255	Axis computer 2 (same network segment as Axis computer 1) Only used if the option MultiMove is used.
192.168.129.0 - 255	Axis computer 3 (same network segment as Axis computer 1) Only used if the option MultiMove is used.
192.168.130.0 - 255	Axis computer 4 (same network segment as Axis computer 1) Only used if the option MultiMove is used.

Available addresses for customer equipment on the Private Network

On the Private Network, some addresses are reserved for ABB equipment. To avoid conflicts, use addresses in the following range for user specific equipment:

- 192.168.125.150 - 199

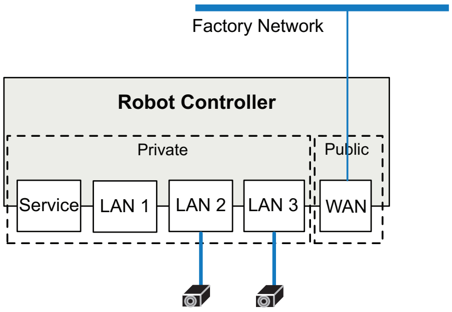
Continues on next page

> **Note**
>
> There is a DHCP server active on the main computer for the Private Network. It gives out IP addresses to any DHCP client that connects to the Private Network, such as a service PC, sensor or camera.

Use cases

Use case 1: LAN 3 as part of the Private network

In this use case the WAN port should be connected to the factory network and both LAN 2 and LAN 3 should connect to equipment that is private to the robot controller.



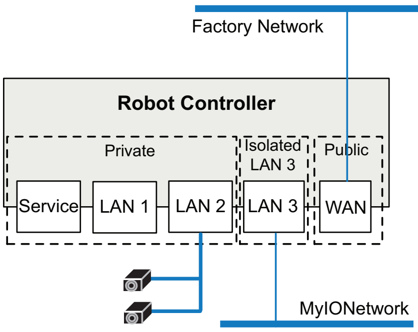
xx1500000530

	Action	Note
1	In the Boot Application, set the IP address, subnet mask, and gateway for the WAN port.	
2	In the system parameters, select topic **Communication** and type **Static VLAN**. Select **X5** and change the parameter **Interface** to **LAN**.	This makes LAN 3 part of the Private Network. It automatically uses the *IP Setting* that applies to the network interface LAN, so there is no need to create any new *IP Setting*.

Continues on next page

Use case 2: isolated LAN 3

In this use case, the WAN port should be connected to the factory network and LAN 3 should be configured with an IP address on an external network (isolated from the Private Network). Remember that the Private Network of two robots cannot be connected. Multiple robot controllers can only appear on LAN3 and Public Network.



xx1500000529

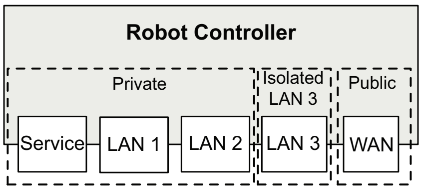
	Action	Note
1	In the Boot Application, set the IP address, subnet mask, and gateway for the WAN port.	
2	In the type **IP Setting**, create a new instance. Set the parameters: • **IP**: IP address, e.g. 192.168.99.1 • **Interface**: LAN 3 • **Subnet mask**: 255.255.255.0 • **Label**: Network name, e.g. MyIONetwork	This assigns IP address 192.168.99.1 to interface LAN3 on this robot controller and makes it visible on the isolated LAN3 network. If there is another robot controller on this network, it could be assigned e.g. address 192.168.99.2, with the same subnet mask.

2.3 Workflows

2.3.1 How to configure LAN 3 to be part of private network

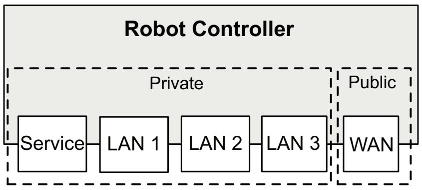
Overview

The default configuration is that LAN 3 is configured as an isolated network. This allows LAN 3 to be connected to an external network, including other robot controllers. The isolated LAN 3 network cannot use any of the address ranges specified in *Predefined networks on page 31*.



xx1500000393

An alternative configuration is that LAN 3 is part of the private network. The ports Service, LAN 1, LAN 2, and LAN 3 then belong to the same network and act just as different ports on the same switch.



xx1500000394

Note

For more information and examples of connecting to different networks, see *Application manual - EtherNet/IP Scanner/Adapter* or *Application manual - PROFINET Controller/Device*.

Configuring LAN 3 as part of private network

1. In the topic **Communication**, choose the type **Static VLAN**.
2. Select X5.
3. Change the parameter **Interface** to **LAN**.
4. Save the changes.

Related information

Interface on page 85

2.4 Type Application Protocol

2.4.1 The Application Protocol type

Overview

This section describes the type *Application Protocol*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_APP

Type description

The type *Application Protocol* is used to configure some of the application level network protocols available the robot controller. It is currently applicable to:

- FTP Client
- NFS Client
- Storing configuration information for cameras used by the option *Integrated vision*

Appropriate RobotWare options need to be installed. For more details, see the section about communication in *Application manual - Controller software IRC5*.

Prerequisites

A transmission protocol must be defined before an application protocol can be defined. For *FTP Client* and *NFS Client*, it is always predefined to TCPIP1.

Relevant options must be installed in the robot system.

Related information

Type Transmission Protocol on page 86

Application manual - Controller software IRC5.

Example: FTP

This is a typical example of a configuration for *FTP Client*.

Parameter:	Value:
Name	MyFTP
Type	FTP
Transmission Protocol	TCPIP1
Server Address	100.100.1.10
Trusted	Yes
Local Path	pc:
Server Path	c:\backup
Username	Operator1
Password	robot

Continues on next page

Parameter:	Value:
Memory Partition Size	500

Example: NFS

This is a typical example of a configuration for *NFS Client*.

Parameter:	Value:
Name	MyNFS
Type	NFS
Transmission Protocol	TCPIP1
Server Address	255.255.100.105
Trusted	Yes
Local Path	pc:
Server Path	c:\backup
User ID	10
Group ID	0

2.4.2 Name

Parent

Name belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Name

Description

The name of the application protocol.

Usage

Used as a protocol label (to tell the application protocols apart).

Allowed values

A string with maximum 40 characters.

2.4.3 Type

Parent

Type belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Type

Description

The type of application protocol.

Usage

Specify the type of application protocol, FTP or NFS.

Allowed values

FTP or NFS

Related information

Application manual - Controller software IRC5

2.4.4 Transmission Protocol

Parent

Transmission Protocol belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Trp

Description

Specifies which transmission protocol is used by the application protocol.

Usage

Transmission Protocol is set to the same value as the parameter *Name*, in the type *Transmission Protocol*, for the transmission protocol you want to use (e.g., TCPIP1).

Allowed values

A string with maximum 40 characters.

Related information

Name on page 87.

Application manual - Controller software IRC5

2.4.5 Server Address

Parent

Server Address belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

ServerAddress

Description

The IP address of the computer that runs the server application that the application protocol communicates with.

Usage

If the application protocol is used for communication with a remote computer, the IP address of that computer is specified in *Server Address*.

Allowed values

Four integers between 0 and 255, separated with dots.

Related information

Application manual - Controller software IRC5

Example

An IP address typically looks like this:

100.100.100.100

2.4.6 Trusted

Parent

Trusted belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Trusted

Description

A flag that specifies if losing the connection should make the program stop.

Usage

An application protocol used for backups or similar can have *Trusted* set to No. If the connection is lost, the program continues and the backup can be made later.

An application protocol that relies on the connection for safety must have *Trusted* set to Yes. If the connection is lost, the program will stop and no hazardous situations can occur because of the lost connection.

Allowed values

Yes or No.

Related information

Application manual - Controller software IRC5

2.4.7 Local Path

Parent

Local Path belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

LocalPath

Description

The controller's reference to the connection.

Usage

When the connection is used from a RAPID program or the FlexPendant, it is referenced with the name defined in *Local Path*.

Defines what the shared unit will be called on the robot. The parameter value must end with a colon (:).

Allowed values

A string with a maximum of 20 characters. The string must end with a colon (:).

Related information

Application manual - Controller software IRC5

Example

The application protocol is used for a connection with unit `C:` on a remote PC. *Local Path* is set to `pc:`. The file `C:\test.prg` can then be accessed from a RAPID program or the FlexPendant as `pc:test.prg`.

2.4.8 Server Path

Parent

Server Path belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

ServerPath

Description

The name of the disk or folder to connect to, on a remote computer.

Usage

Specify the path of the disk or folder that the application protocol should connect to.

> **Note**
>
> The exported path should not be specified if communicating with an FTP server of type Distinct FTP, FileZilla or MS IIS.

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Controller software IRC5

Example

The usage of *Server Path* may depend on which FTP server is being used.

For most FTP servers

If the application protocol should connect to the folder `C:\Robot1\Backup` on a remote computer, *Server Path* is set to `C:\Robot1\Backup`.

For FTP servers Distinct FTP, MS IIS, and FileZilla

If the server exports `C:\Robot1` and the application protocol want to connect to `C:\Robot1\Backup`, *Server Path* is set to `Backup`.

2.4.9 Username

Parent

Username belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

UserName

Description

The user name used by the robot when it logs on to an FTP server on a remote computer.

Usage

Create a user account on the FTP server. The user name of this account is then specified in *Username*, and the password in *Password*.

Limitations

Username is only used with the RobotWare option *FTP Client*.

Allowed values

A string with a maximum of 40 characters.

Related information

Password on page 45.

Application manual - Controller software IRC5.

2.4.10 Password

Parent

Password belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Password

Description

The password used by the robot when it logs on to an FTP server on a remote computer.

Usage

Create a user account on the FTP server. The user name of this account is then specified in *Username*, and the password in *Password*.

Limitations

Password is only used with the RobotWare option *FTP Client*.

Allowed values

A string with a maximum of 40 characters.

Additional information

Note that the password written here will be visible to all who have access to the system parameters.

Related information

Username on page 44.

Application manual - Controller software IRC5.

2.4.11 User ID

Parent

User ID belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

UserID

Description

Used by the NFS protocol as a way of authorizing the user to access a specific server.

Usage

If the NFS server requires a User ID and Group ID for access to the server, these numbers are specified in the parameters *User ID* and *Group ID*.

If this parameter is not used, set it to the default value 0.

Note that *User ID* must be the same for all mountings on one controller.

Limitations

User ID is only used with the RobotWare option *NFS Client*.

Allowed values

An integer between 0 and 2,147,483,647.

Default value is 0.

Related information

Group ID on page 47.

Application manual - Controller software IRC5.

2.4.12 Group ID

Parent

Group ID belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

GroupID

Description

Used by the NFS protocol as a way of authorizing the user to access a specific server.

Usage

If the NFS server requires a User ID and Group ID for access to the server, these numbers are specified in the parameters *User ID* and *Group ID*.

If this parameter is not used, set it to the default value 0.

Note that *Group ID* must be the same for all mountings on one controller.

Limitations

Group ID is only used with the RobotWare option *NFS Client*.

Allowed values

An integer between 0 and 2,147,483,647.

Default value is 0.

Related information

User ID on page 46.

Application manual - Controller software IRC5.

2.4.13 Memory Partition Size

Parent

Memory Partition Size belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

CommPartSize

Description

The parameter *Memory Partition Size* defines the size of the allocated memory partition for the FTP communication.

Usage

By using a separate memory partition for the FTP communication, the risk of disturbing other program execution is avoided.

If no separate memory partition is desired, set the value to 0.

Prerequisites

Memory Partition Size is only used with the RobotWare option *FTP Client*.

Allowed values

Partition size in kB (kilo bytes), between 0 and 2000.

Default value is 300 kB.

Note that values above default value cannot be guaranteed to function. The available memory partition size depends on what other options are installed.

Related information

Application manual - Controller software IRC5

2.4.14 Show Device

Parent

Show Device belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

ShowDevice

Description

Show Device defines if the storage device should be visible in the list of storage devices on the FlexPendant.

Usage

The *Show Device* parameter can be used to restrict access to an FTP or an NFS mounted storage device. If the *ShowDevice* parameter is set to No, it will not be visible in the open/save dialogs on the FlexPendant.

NOTE! If the path of the storage device is known to the user, it is possible to access that storage device by entering the path in the open/save dialogs on the FlexPendant, regardless of the value of the *Show Device* parameter.

Prerequisites

Show Device is used only with the RobotWare options *FTP client* and *NFS client*.

Allowed values

Yes or No.

2.5 Type DNS Client

2.5.1 The DNS Client type

Overview

This section describes the type *DNS Client*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

DNSC

Type description

The type *DNS Client* is used to enable, disable, and change parameters for IRC5 DNS Client.

2.5.2 Name

Parent

Name belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

Name

Description

Must exist and be set to *DNS Client*.

Default value

The default value is *DNSC*.

Allowed values

DNSC

2.5.3 Enabled

Parent

Enabled belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

Enabled

Description

This defines the *DNS Client* is turned on or off.

Default value

The default value is *No*.

Allowed values

Yes or *No*

2.5.4 Domain Name

Parent

Domain Name belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

DomainName

Description

Defines the domain where the host is located. If it is not defined, the DNS users must provide fully qualified domain names in address lookups.

Default value

The default value is an empty string.

Allowed values

A string with maximum 80 characters.

2.5.5 1st Name Server

Parent

1st Name Server belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

PrimaryNameServer

Description

Defines the primary name server. If it is not defined, the *DNS Client* will not perform any lookups.

Default value

The default value is an empty string.

Allowed values

0.0.0.0 - 255.255.255.255

2.5.6 2nd Name Server

Parent

2nd Name Server belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

SecondaryNameServer

Description

Defines the secondary name server.

Default value

The default value is an empty string.

Allowed values

0.0.0.0 - 255.255.255.255

2.5.7 3rd Name Server

Parent

3rd Name Server belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

TertiaryNameServer

Description

Defines the third name server.

Default value

The default value is an empty string.

Allowed values

0.0.0.0 - 255.255.255.255

2.5.8 4th Name Server

Parent

4th Name Server belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

QuarternaryNameServer

Description

Defines the fourth name server.

Default value

The default value is an empty string.

Allowed values

0.0.0.0 - 255.255.255.255

2.5.9 Server Port

Parent

Server Port belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

ServerPort

Description

Defines the port used by the *DNS Client* for DNS queries. This parameter is rarely changed.

Default value

The default value is *53*.

Allowed values

0 - 65535

2.5.10 Retries

Parent

Retries belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

Retries

Description

Defines the number of retries used by the *DNS Client* for DNS queries. This number is carried out for each name server. This parameter is rarely changed.

Default value

The default value is *2*.

Allowed values

0 - 65535

2.5.11 Timeout

Parent

Timeout belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

Timeout

Description

Defines the timeout in seconds used by the *DNS Client* between retries. This parameter is rarely changed.

Default value

The default value is *10*.

Allowed values

0 - 65535

2.5.12 IPv4 Zone Name

Parent

IPv4 Zone Name belongs to the type *DNS Client*, in the topic *Communication*.

Cfg name

Ipv4Zone

Description

Defines the zone used by the *DNS Client* for address-to-name lookups of IPv4 addresses. This parameter is rarely changed.

Default value

in-addr.arpa

Allowed values

A string with maximum 80 characters.

2.6 Type Ethernet Port

2.6.1 The Ethernet Port type

Overview

This section describes the type *Ethernet Port*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ETHERNET_PORT

Type description

The type *Ethernet Port* is used for configuring the Ethernet ports on the main computer:

- X2 (Service)
- X3 (LAN 1)
- X4 (LAN 2)
- X5 (LAN 3)
- X6 (WAN)

2.6.2 Port

Parent

Port belongs to the type *Ethernet Port*, in the topic *Communication*.

Cfg name

Name

Description

The connector ID on the main computer.

Usage

Used as a port descriptor (to tell the ports apart).

Allowed values

X2, X3, X4, X5, X6.

These ports are predefined and cannot be changed, deleted or created.

2.6.3 Port Speed

Parent

Port Speed belongs to the type *Ethernet Port*, in the topic *Communication*.

Cfg name

PortSpeed

Description

The parameter *Port Speed* specifies the transmission speed for the Ethernet connector. The following three values are defined:

- Auto: The Ethernet connector will choose the highest performance transmission speed, the connecting devices support.
- 10 Mbit/s: The transmission speed on the Ethernet connector will be fixed to 10 Mbit/s.
- 100 Mbit/s: The transmission speed on the Ethernet connector will be fixed to 100 Mbit/s.

Note

Only full duplex mode is supported.

Note

If the *Port Speed* is changed, all clients using this connector will be affected.

Default value

Auto

Allowed values

Auto

10 Mbps

100 Mbps

2.7 Type IP Setting

2.7.1 The IP Setting type

Overview

This section describes the type *IP Setting*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

IP_SETTING

Type description

The type *IP Setting* is used to set an address to a network interface of the main computer. If necessary, multiple addresses can be set for the same interface (multi-homing an interface).

Additional information

The following instances of *IP Setting* are locked and cannot be edited or removed by configuring system parameters:

- Axis computer 1
- FlexPendant Network
- Private Network
- Public Network (set up using the Boot Application)

RobAPI clients (for example RobotStudio, FlexPendant, and PC SDK) can access the robot controller via the Private Network, FlexPendant Network or Public Network.

2.7.2 IP

Parent

IP belongs to the type *IP Setting*, in the topic *Communication*.

Cfg name

Address

Description

The parameter *IP* specifies the IP address that is added to the network interface specified in the parameter *Interface*.

Usage

The parameter *IP* is used to set the IP address of the IRC5 controller on the used network interface.

The IP address must belong to another subnet than the IP address of any other port on the IRC5 controller.

Note

The following IRC5 controller subnets are reserved:

- 192.168.125.0/24
- 192.168.126.0/24
- 192.168.127.0/24
- 192.168.128.0/24 (for MultiMove only)
- 192.168.129.0/24 (for MultiMove only)
- 192.168.130.0/24 (for MultiMove only)
- 192.168.136.0/24 (for Paint robots only)

Allowed values

0.0.0.0 - 255.255.255.255

2.7.3 Subnet

Parent

Subnet belongs to the type *IP Setting*, in the topic *Communication*.

Cfg name

SubnetMask

Description

Defines which subnet the IP address belongs to.

Usage

The parameter *Subnet* is used to divide the network into logical subnets.

Allowed values

0.0.0.0 - 255.255.255.255

2.7.4 Interface

Parent

Interface belongs to the type *IP Setting*, in the topic *Communication*.

Cfg name

Interface

Description

Interface specifies the network interface to which the IP address and the subnet mask are applied to.

Default value

LAN

Allowed values

WAN

LAN

LAN3 (when using the default configuration with isolated LAN 3)

2.7.5 Label

Parent

Label belongs to the type *IP Setting*, in the topic *Communication*.

Cfg name

Name

Description

User friendly name of the network that the IP address belongs to.

Allowed values

A string with maximum 80 characters.

2.8 Type Serial Port

2.8.1 The Serial Port type

Overview

This section describes the type *Serial Port*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_PHY_CHANNEL

Type description

The type *Serial Port* is used for configuring the serial channel on the controller.

If the controller has the board DSQC1003 installed, there is one serial channel, which can be used for communication with printers, terminals, computers, and other equipment.

Serial channel:	Description:
COM1	A standard RS232 port

2.8.2 Name

Parent

Name belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

Name

Description

Name specifies the logical connection. It is used when accessing the physical serial channel.

Allowed values

A string with maximum 16 characters.

2.8.3 Connector

Parent

Connector belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

Connector

Description

Connector connects a physical communication port with a specific configuration in the system.

Allowed values

COM1, in a system with the board DSQC1003 installed.

2.8.4 Baudrate

Parent

Baudrate belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

Baudrate

Description

Baudrate defines the baud the controller will operate on for the selected serial port.

Usage

Baud is the signalling rate of the communication, which determines the maximum speed of the data transfer in serial channels. The higher the baud, the faster the communication can be.

Limitations

Both devices, the serial ports in both ends, that communicate on the channel have to use the same baud. The devices have to be defined with the same transmission speed. Therefore, *Baudrate* must be set to the baud of the device that is connected to the controller.

Allowed values

A value between 300-115200, specifying the signal rate.

The default value is 9600.

2.8.5 Parity

Parent

Parity belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

Parity

Description

Parity configures the parity check for the data transfer.

Usage

Parity check is an error detection method to help detect data corruption that might occur during transmission of data. The parity check adds a parity bit to each byte that is transmitted.

Depending on whether the transmitted byte contains an odd or even number of 1-bits, the parity bit will be either 0 or 1. Each time a data byte is received, it is checked that the number of 1-bits matches the parity bit.

Limitations

Both receiver and transmitter of data must agree on the type of parity.

Allowed values

Value	Description
Odd	The number of 1-bits in a transfer byte must be odd. If they are odd, the parity bit is set to 0.
Even	The number of 1-bits in a transfer byte must be odd. If they are even, the parity bit is set to 1.
None	No parity check is performed.

2.8.6 Number of Bits

Parent

Number of Bits belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

NoOfBits

Description

Number of Bits defines the number of data bits in each byte.

Usage

The number of bits depends on the device the controller should communicate with. Both receiver and transmitter must agree on the number of data bits as well as the baudrate. There may either be 7 or 8 data bits depending on the selection made.

Limitations

Both receiver and transmitter of data must agree on the number of bits.

Allowed values

7 or 8, specifying the number of data bits.

Related information

Baudrate on page 73.

2.8.7 Number of Stop Bits

Parent

Number of Stop Bits belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

NoOfStopBits

Description

Number of Stop Bits defines the number of stop bits.

Usage

A stop bit is used to identify the end of a data byte when it is transmitted. A stop bit can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's duration.

Limitations

Both receiver and transmitter of data must agree on the number of bits.

Stop bits are excluded from the parity calculation. For more information about parity, see *Parity on page 74*.

Allowed values

1 or 2, specifying the number of stop bits.

Related information

Parity on page 74.

2.8.8 Duplex

Parent

Duplex belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

Duplex

Description

Duplex defines whether or not the controller shall be able to send and receive data simultaneously on this serial port.

Usage

Duplex is the ability to transport data in both directions.

With full duplex the controller is able to both send and receive data at the same time.

With half duplex the data flow is limited to one direction at a time.

Allowed values

FULL or HALF.

2.8.9 Flow Control

Parent

Flow Control belongs to the type *Serial Port*, in the topic *Communication*.

Cfg name

FlowControl

Description

Flow Control defines which type of data flow control is used between the devices that are communicating on the serial port.

Usage

Flow control adjusts the data transfer so that no data is sent before the receiving device can receive it. Flow control is extra important when the sending device can send data at a higher speed than the receiving device is able to receive.

Limitation

Both receiver and transmitter of data must agree on the type of flow control used.

Allowed values

Value	Description
RTS/CTS	Hardware flow control, uses signals on the serial cable to control if sending or receiving is enabled.
XON/XOFF	Software flow control, uses characters in the communication stream to control sending and receiving of data.
NONE	Flow control will not be used.

2.9 Type IP Route

2.9.1 The IP Route type

Overview

This section describes the type *IP Route* which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ROUTE

Type description

IP Route is used to configure the IP routing table of the main computer. If a default gateway is specified in the Boot Application then it is shown as a read-only instance.

2.9.2 Destination

Parent

Destination belongs to the type *IP Route*, in the topic *Communication*.

Cfg name

Destination

Description

Destination is used if a new route should be added to the system routing table.

Usage

Specify a destination if a new route should be added. The default gateway will not be changed.

The address specified is in CIDR format.

Default value

Empty

Example

Gateway "192.168.20.10"

Destination "192.168.20.0/24"

The routing table will be updated with a new route to the 192.168.20.0 network through 192.168.20.10 gateway.

Allowed values

0.0.0.0 - 255.255.255.255.

2.9.3 Gateway

Parent

Gateway belongs to the type *IP Route*, in the topic *Communication*.

Cfg name

Gateway

Description

The parameter *Gateway* specifies the node on the network that serves as an entrance to another network.

Usage

Use this parameter if the traffic needs to be routed to another network. The parameter value is the address to a physical gateway on the network.

Note

A destination address must be specified if the gateway address is specified.

Default value

The default value is set up using the Boot Application.

Allowed values

0.0.0.0 - 255.255.255.255

2.9.4 Label

Parent

Label belongs to the type *IP Route*, in the topic *Communication*.

Cfg name

Name

Description

User friendly name of the routing entry.

Allowed values

A string with maximum 80 characters.

2.10 Type Static VLAN

2.10.1 The Static VLAN type

Overview

This section describes the type *Static VLAN* which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

STATIC_VLAN

Type description

Static VLAN is used to configure grouping of physical Ethernet ports into static VLAN groups. Ports in the same group are also a part of the same network interface in the main computer (see *Ethernet ports and system parameters on page 30*).

Only X5 can be configured. It can belong to either LAN interface or LAN3 interface.

2.10.2 Port

Parent

Port belongs to the type *Static VLAN*, in the topic *Communication*.

Cfg name

Name

Description

Name of the connectors X2 to X6.

2.10.3 Interface

Parent

Interface belongs to the type *Static VLAN*, in the topic *Communication*.

Cfg name

Interface

Description

Name of the network interface and the static VLAN group that the physical port shall be a part of.

Limitations

Only port X5 can be configured. Other ports have predefined group/interface membership.

Allowed values

LAN

LAN3

Related information

How to configure LAN 3 to be part of private network on page 34

2.11 Type Transmission Protocol

2.11.1 The Transmission Protocol type

Overview

This section describes the type *Transmission Protocol* which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_TRP

Type description

The type *Transmission Protocol* is used to configure connections to serial channels and certain network devices.

For network devices, the connection instance is configured by setting the parameter *Type* to TCP/IP and specifying *Remote Address* and *Remote port number*. *Serial Port* is not applicable (N/A).

For serial channel connections, instances are configured by specifying *Type* and *Serial Port*, while *Remote Address* and *Remote port number* are not applicable.

More details and examples are provided in separate manuals for options that are dependent on these system parameters.

2.11.2 Name

Parent

Name belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

Name

Description

Name specifies the name of the transmission protocol.

Allowed values

A string with maximum 16 characters.

2.11.3 Type

Parent

Type belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

Type

Description

Type defines the type of transmission protocol to be used.

Allowed values

Installed Transmission protocol types. Number and names of the installed types depend on the installed system options.

Related information

The Serial Port type on page 70.

Operating manual - RobotStudio.

For configuration of the LAN port, see *Operating manual - IRC5 with FlexPendant*.

2.11.4 Serial Port

Parent

Serial Port belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

PhyChannel

Description

Serial Port connects a transmission protocol with a serial port.

Limitations

It is not possible to connect to the LAN port. For configuration of the LAN port, see *Operating manual - IRC5 with FlexPendant*.

Allowed values

COM1, in a system with the board DSQC1003 installed.

Additional information

For IP based transmission protocols (i.e. *Type* has value TCP/IP, SOCKDEV, LTAPPTCP or UDPUC), *Serial Port* is not used and has the value N/A.

Related information

The Serial Port type on page 70.

Operating manual - IRC5 with FlexPendant.

2.11.5 Remote Address

Parent

Remote Address belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

RemoteAdress

Description

Remote Address specifies the IP address of the sensor.

Limitations

The parameter *Remote Address* can only be used for protocols that communicate over an IP network. The parameter is N/A for communication over a Serial Port.

Allowed values

A string consisting of 4 integer values between 0 and 255, each specifying one of the four parts, separated by dots.

Related information

Type on page 88.

Example

An IP address consists of four parts, each with eight bits, separated by dots: `100.100.100.100` or `138.227.1.45`.

2.11.6 Remote port number

Parent

Remote port number belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

RemotePortNumber

Description

Remote port number specifies port number on the network node identified by *Remote Address*, that connection shall be established to.

Allowed values

Integer value

This page is intentionally left blank

3 Topic Controller

3.1 The Controller topic

Overview

This chapter describes the types and parameters of the *Controller* topic. Each parameter is described in the section for its type.

Description

The *Controller* topic contains parameters for safety and RAPID specific functions.

The parameters are organized in the following types:

1. Auto Condition Reset
2. Automatic Loading of Modules
3. Event Routine
4. Mechanical Unit Group
5. ModPos Settings
6. Operator Safety
7. Path Return Region
8. Run Mode Settings
9. Present Options
10. Safety Run Chain
11. General Rapid
12. Task

3.2 Workflows

3.2.1 How to activate hold-to-run control

Overview

Safety in program execution is essential. The function hold-to-run control is used when extra safety is necessary in the operating mode Manual. The hold-to-run function only allows robot movements when a button is manually actuated and immediately stops these movements when released.

Additional information

The hold-to-run control is always activated in Manual Full Speed mode.

How to activate the hold-to-run control

To activate the hold-to-run control for manual reduced speed mode:

1. In the **Controller** topic, choose the type **Operator Safety.**
2. Edit the parameters for robot movement control and execution. Set the parameter **Active** to True.

   For detailed information about the parameters, see the descriptions in the *Operator Safety* type.
3. Save the changes.

Related information

The Operator Safety type on page 133.

Operating manual - IRC5 with FlexPendant.

3.2.2 How to define path return region

Return movement

A return movement must take place if the current robot path deviates from the programmed path. This happens for example if an uncontrolled stop has occurred or the robot has been jogged away from its path. A return movement begins when program start is ordered and stops before the program continues with the instruction that was interrupted.

Path return region

In a return movement, the path return region specifies the distance from the current robot position to the last executed path. The maximum path return region can be set both for start in manual mode and for start in automatic mode.

How to define path return region

To define the path return region:

1. In the **Controller** topic, choose the type **Path Return Region**.
2. Edit the **Mode** parameter to specify the operating mode.
3. Edit the parameters for movement in the selected mode. For detailed information about each parameter, see the descriptions in the type *Path Return Region*.
4. Save the changes.

Related information

The Path Return Region type on page 139.

3.3 Type Auto Condition Reset

3.3.1 The Auto Condition Reset type

Overview

This section describes the type *Auto Condition Reset*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

AUTO_COND_RESET

Type description

The type *Auto Condition Reset* defines if a number of conditions should be reset when switching to auto mode.

A message box is displayed on the FlexPendant with information about the reset conditions.

Limitations

There can be only one instance of the type *Auto Condition Reset*.

3.3.2 Name

Parent

Name belongs to the type *Auto Condition Reset*, in the topic *Controller*.

Cfg name

name

Allowed values

AllDebugSettings (cannot be changed).

3.3.3 Reset

Parent

Reset belongs to the type *Auto Condition Reset*, in the topic *Controller*.

Cfg name

reset

Description

Reset defines if a number of conditions should be reset when switching to auto mode.

If any of the conditions cannot be executed, then switching to auto will be rejected. The *Reset* setting is also applied when starting the controller in auto mode.

Usage

If *Reset* is set to YES then the following conditions are reset when switching to auto:

- The Program Pointer (PP) is set to Main module for all tasks if callchain does not originate from Main routine.
- All tasks are enabled.
- All stopped background tasks are started.
- Simulation of all simulated I/O signals is removed.
- Speed is set to 100%.
- RAPID Spy is deactivated.

If *Reset* is set to NO, then none of the above conditions are reset automatically.

If a service routine is running and PP was manually moved to another routine before the service routine was called, then the above does not apply. Switching to auto will then be rejected.

Allowed values

YES

NO

Default value is YES.

3.4 Type Automatic Loading of Modules

3.4.1 The Automatic Loading of Modules type

Overview

This section describes the type *Automatic Loading of Modules* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CAB_TASK_MODULES

Type description

RAPID modules can be loaded automatically when the controller is restarted if they are specified in the type *Automatic Loading of Modules*.

Usage

There must be one instance of the type *Automatic Loading of Modules* for each of the module to be loaded.

System restart

All changes in the type *Automatic Loading of Modules* will take effect after a normal restart or using the restart mode **Reset RAPID**.

Additional information

If the configuration module is changed, it may in one case (see below) replace the loaded module after a normal restart. In any other case, you will get a warning. To replace the loaded module regardless of task type, restart using the restart mode **Reset RAPID**.

The configuration module replaces the loaded module if the:

- loaded module is a program module AND
- the task is semistatic.

The program pointer is only lost if a configuration change results in unloading of the module that the program pointer is in. If a shared or installed module is changed from True to False, or is moved to another task, the task will be reinstalled and the program pointer is reset. All previously loaded modules are reloaded and unsaved changes will not be lost.

If a changed and unsaved user-loaded module is unloaded due to configuration changes, it will be saved to a recovery directory and pointed out in an ELOG message.

If a changed and unsaved configuration loaded module is unloaded due to configuration changes, it will be saved from where it was loaded.

All tasks are reinstalled with modules according to the configuration after a restart using the restart mode **Reset RAPID**. Note that after using the restart mode **Reset RAPID**, all user-loaded modules are lost.

Continues on next page

Related information

The Task type on page 159.

Technical reference manual - RAPID overview.

ELOG messages are described in *Operating manual - Trouble shooting IRC5*.

Restarts are described in *Operating manual - IRC5 with FlexPendant*.

3.4.2 File

Parent

File belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

File

Description

The parameter *File* describes a path to the module file.

Usage

The module file shall contain one module to be loaded, installed, or shared.

Allowed values

A path, for example, `HOME:base.sys`

Related information

Technical reference manual - RAPID overview.

3.4.3 Task

Parent

Task belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Task

Description

Task is the symbolic name of the task to which the module will be loaded.

Usage

The task is defined in the type *Task*.

The available task(s) is shown under the type *Task*.

Limitations

Cannot be combined with *All Tasks*, *All Motion Tasks*, or *Shared*.

Allowed values

A task name with maximum 30 characters.

Additional information

All automatically loaded modules need information on which task they will be loaded or installed in, even if only one task is configured in the system.

Related information

The Task type on page 159.

All Tasks on page 105.

Shared on page 104.

Application manual - Controller software IRC5.

3.4.4 Installed

Parent

Installed belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Installed

Description

A module can be installed or loaded. A loaded module is visible in remote clients, for example, RobotStudio and FlexPendant. An installed module is not visible, that is, it does not occur in the list of modules.

Usage

Set *Installed* to Yes to install a module, and to No to load a module.

Limitations

Cannot be combined with *Shared*.

Allowed values

YES or NO.

The default value is No.

Additional information

To remove an installed module, the parameter *Installed* must be set to No and restart the system.

Related information

Shared on page 104.

All Tasks on page 105.

Technical reference manual - RAPID overview.

3.4.5 Shared

Parent

Shared belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Shared

Description

It is possible to install the module (and all its objects) as shared so it is reachable from all the tasks.

Usage

If a module should be reachable from any task, set the parameter *Shared* to YES. This installs the module to the system internal shared task, not visible from any user interface or in the configuration. All data in the module is then shared (that is the same) for all tasks.

Limitations

Cannot be combined with *Task, All Tasks*, *All Motion Tasks*, or *Installed*.

Allowed values

YES or NO.

Default value is No.

Additional information

If Shared:	and if Installed:	Then:
Yes	No	The module is installed shared. Module data is shared between all tasks.
No	Yes	The module is installed and only available from the named task.
No	No	The module is loaded.

Related information

All Tasks on page 105.

Task on page 102.

Installed on page 103.

3.4.6 All Tasks

Parent

All Tasks belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

AllTask

Description

The *All Tasks* module will be loaded or installed in all the tasks available in the system.

Note that there can be more tasks available in the system than can be seen, that is, tasks with *Type* defined as STATIC or SEMISTATIC, or *Hidden* defined as YES.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task, All Motion Tasks,* or *Shared*.

A module with *All Motion Tasks* set to Yes can only contain the code possible to run in any motion task in the system.

Allowed values

YES

NO

Default value is No.

Additional information

If *All Tasks* is set to Yes and *Installed* is set to Yes then the module is installed in each task as a separate module. That is, the module data is not shared between the tasks (as opposed to if the module is installed shared).

Related information

Task on page 102.

Shared on page 104.

The Task type on page 159.

3.4.7 All Motion Tasks

Parent

All Motion Tasks belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

AllMotionTask

Description

The *All Motion Tasks* module will be loaded or installed in all motion tasks available in the system.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task*, *Shared*, or *All Tasks*.

A module with *All Motion Tasks* set to Yes can only contain the code possible to run in any motion task in the system.

Allowed values

YES or NO.

The default value is NO.

Additional information

If *All Motion Tasks* is set to Yes and *Installed* is set to Yes then the module is installed in each motion task as a separate module. That is, module data is not shared between the tasks (as opposed to if the module is installed shared).

Related information

Task on page 102.

Shared on page 104.

The Task type on page 159.

3.4.8 Hidden

Parent

Hidden belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Hidden

Description

RAPID modules, routines and data may be hidden, which may be used to prevent inexperienced end users from tampering (accidentally deleting or changing) with the contents.

Note that the hidden contents is not protected! It can easily be shown again by setting the parameter value to NO.

Note that any hidden contents will still be available when using the `SetDataSearch` instruction to search RAPID data.

Limitations

This parameter affects only modules, routines, and data that are loaded automatically on start, that is no programs etc. that are loaded by the operator once the system has been started.

Changes to the parameter will be effective only after using the restart mode **Reset RAPID**.

Allowed values

YES or NO.

Default value is NO.

3.5 Type Event Routine

3.5.1 The Event Routine type

Overview

This section describes the type *Event Routine* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CAB_EXEC_HOOKS

Type description

The type *Event Routine* contains parameters for event handling. Special system events, such as program stop, can be connected to a RAPID routine. When the event occurs, the connected event routine is executed automatically.

An event routine is made up of one or more instructions. The routine runs in the task specified in parameter *Task* or *All Tasks*.

The tasks available are dependent on the type *Tasks*.

Event routines

The following event routines are available:

- PowerOn
- Start
- Step
- Restart
- Stop
- QStop
- Reset

Event routines can be started for one or many tasks. A normal task execution will not wait for the event routines in other tasks. Tasks that are dependent on the event routines of other tasks shall therefore be synchronized, for example, with *WaitSyncTask* before a normal task execution.

A stopped event routine will continue from where it was stopped when pressing the start button on the FlexPendant or when calling the start command via a system I/O.

Pressing the stop button when the Stop event routine is executing does not generate a new Stop event. However, if a problem has occured in the event routine then pressing the stop button will force the execution to leave the event routine after 10 seconds.

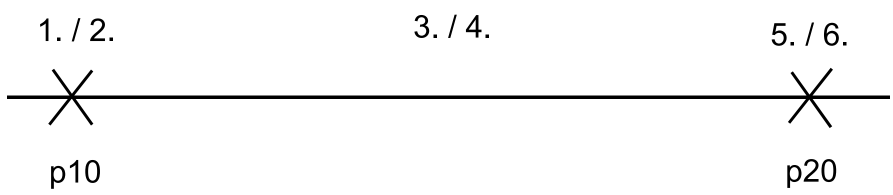
The only way to cancel a stopped event routine from system I/O is to start the program from main.

Continues on next page

A `Stop` instruction (without the optional argument `-All`) or a `Break` instruction in an event routine will stop the program execution. This means that instructions after the `Stop` or `Break` instruction will never be executed. See *Example 1 on page 110*.

Event routine execution examples

The following is an illustration of the sample code that is shown below it. The examples that follow show which event routines are executed for the various buttons pressed on the FlexPendant.



xx1100000050

```
PROC main()
MoveJ p20, v100, fine, tool0;
MoveJ p10, v100, fine, tool0;
ENDPROC
```

Example 1

The following procedure shows that the START, STOP, and RESTART event routines are executed when the Start and Stop buttons are pressed on the FlexPendant.

Step	Action	Executed event routine
1	Tap PP to Main.	-
2	Press the Start button.	START
3	Press the Stop button.	STOP
4	Press the Start button.	RESTART
5	p20 is reached.	-
6	Execution continues.	-

Example 2

The following procedure shows that the START, STOP, and RESTART event routines are executed when the Start, Stop, and Step buttons are pressed on the FlexPendant.

Step	Action	Executed event routine
1	Tap PP to Main.	-
2	Press the Start button.	START
3	Press the Stop button.	STOP

Continues on next page

Step	Action	Executed event routine
4	Press the Step button.	RESTART
5	p20 is reached.	-
6	Execution stops.	STOP

Example 3

The following procedure shows that the START, STOP, and STEP event routines are executed when the Step and Stop buttons are pressed on the FlexPendant.

Step	Action	Executed event routine
1	Tap PP to Main.	-
2	Press the Step button.	START
3	Press the Stop button.	STOP
4	Press the Step button.	STEP
5	p20 is reached.	-
6	Execution stops.	-

System restart

Any changes in configuration of event routines are activated after a normal restart.

Example 1

This example illustrates the consequences after a `Stop` instruction in a routine.

At restart `mydo` will be set to 1. `mydo` will never be set to 0 since the execution stops after the stop instruction.

The instruction `TPWrite` will never be executed because `myexample2` has sequence number (`SeqNo`) 1.

```
MODULE example(SYSMODULE)
  PROC myexample1()
    SetDO mydo, 1;
    Stop;
    SetDO mydo, 0;
  ENDPROC

  PROC myexample2()
    TPWrite "This is an example";
  ENDPROC
ENDMODULE

CAB_EXEC_HOOKS:
  -Routine "myexample1" -Shelf "RESTART"
  -Routine "myexample2" -Shelf "RESTART" -SeqNo 1
```

Example 2

This example illustrates how to use the same routine for both Start and Step events.

```
MODULE example(SYSMODULE)
  PROC myexample2()
    TEST RunMode()
```

Continues on next page

```
    CASE RUN_CONT_CYCLE:
      ! PLAY button pressed
      ...
    CASE RUN_INSTR_FWD:
      ! FORWARD STEP button pressed
      ...
    CASE RUN_INSTR_BWD:
      ! BACKWARD STEP button pressed
      ...
    ENDTEST
  ENDPROC
ENDMODULE

CAB_EXEC_HOOKS:
  -Routine "myexample2" -Shelf "START"
  -Routine "myexample2" -Shelf "STEP"
```

Related information

The Task type on page 159.

Technical reference manual - RAPID overview.

Technical reference manual - RAPID Instructions, Functions and Data types. The function EventType can be useful.

3.5.2 Routine

Parent

Routine belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Routine

Description

Routine specifies which routine that should be run for an event.

Usage

Define the routine to be assigned to a system event.

It is advisable to use a routine in a system module.

Limitations

The specified routine must be a procedure without any parameters.

The event Reset requires a routine in a system module.

Allowed values

A string defining a routine.

3.5.3 Event

Parent

Event belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Shelf

Description

Event specifies which system event in the robot system the routine should run.

Usage

A system event can trigger a corresponding routine to be run, see *Operating manual - IRC5 with FlexPendant*.

It is advisable to keep the routines short and quick.

Limitations

The following limitations should be considered:

- The events are not activated when executing a routine manually, for example, a service routine.
- A maximum of 20 routines may be specified for each system event and each task (multitasking). The same routine can be used in more than one event (e.g. the same routine can be run for both Start and Restart).
- The specified event routine cannot be executed if the task program has semantic errors (reference errors and so on). If this is the case, the system generates an error.
- Only the event routine for Start can have motion instructions. A motion instruction in any other event routine will result in a runtime execution error. The only exception is the motion instruction StepBwdPath, which is allowed in the event routine for Restart.

Allowed values

The following values are allowed.

Value:	Description:
Power On	The specified routine is run when the robot is restarted (restart) from a remote client or by power on.
Start	Execution is started from the beginning of the program. This is when you press the start or step buttons after having: • loaded a new program or a new module • ordered Start from beginning • ordered Debug/Move PP to Main • ordered Debug/Move PP to Routine • moved the program pointer in such a way that the execution order is lost.

Continues on next page

Value:	Description:
Step	The specified routine is run for every forward and backward step. Use the RAPID function `RunMode` to see if it is a forward or backward step. Use the RAPID function `ExecLevel` to see if it is executing on trap or normal level.
Stop	The program was stopped: • with the stop button • with a STOP instruction • stop after current instruction. **Note** A delayed stop after current cycle will not execute the routines connected to this state. The event is not activated at `Exit` instruction or stop due to execution error.
QStop	The robot was quick stopped (emergency stop).
Restart	Execution is started from the position where it was stopped, or from another instruction the program pointer has been moved to, without having lost the execution order. The event is not activated after having executed one instruction in step by step mode (FWD or MStep).
Reset	Close and load a new program using the FlexPendant. The event is not activated after having loaded a system module or a program module.

Additional information

The following event routines are predefined for all tasks in all systems and must not be removed.

Event:	Routine:	Sequence no.
Reset	`SYS_RESET`	0
Start	`SYS_RESET`	0
Power On	`SYS_POWERON`	0

Related information

Operating manual - IRC5 with FlexPendant.

3.5.4 Sequence Number

Parent

Sequence Number belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

SeqNo

Description

Sequence Number specifies in which order the routine should be executed for a specific event.

Usage

Order the event routines in a sequence where the first routine shall have a low value and the routines that shall run last has the highest value.

0 will run first.

Note

If several event routines has the same sequence number, the execution order will be unpredictable.

Allowed values

A value between 0 and 100.

Default value is 0.

3.5.5 Task

Parent

Task belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Task

Description

Task specifies the name of the task that the routine will run in.

Usage

The task is defined in the type *Task*.

Limitations

Cannot be combined with *All Tasks* or *All Motion Tasks*.

Allowed values

Names of configured tasks of the type *Task*.

Additional information

All event routines need information on which task they will run, even though only one task is configured in the system.

Related information

The Task type on page 159.

All Tasks on page 117.

All Motion Tasks on page 118.

3.5.6 All Tasks

Parent

All Tasks belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

AllTasks

Description

All Tasks defines if the routine will run in all configured tasks in the system.

Note that there can be more tasks available in the system than can be seen, that is tasks with *Type* defined as STATIC or SEMISTATIC, or *Hidden* defiined as YES.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task* or *All Motion Tasks*.

A routine with *All Tasks* set to Yes can only contain code possible to run in any task in the system.

Allowed values

YES or NO.

The default value is No.

Additional information

All event routines need information on which task they will run, even if only one task is configured in the system.

Related information

Task on page 116.

The Task type on page 159.

3.5.7 All Motion Tasks

Parent

All Motion Tasks belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

AllMotionTasks

Description

All Motion Tasks defines if the routine will run in all configured motion tasks in the system.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task* or *All Tasks*.

A routine with *All Motion Tasks* set to Yes can only contain the code possible to run in any motion task in the system.

Allowed values

Yes or No.

The default value is No.

Additional information

All event routines need information on which task they will run, even if only one task is configured in the system.

Related information

Task on page 116.

The Task type on page 159.

3.6 Type Mechanical Unit Group

3.6.1 The Mechanical Unit Group type

Overview

This section describes the type *Mechanical Unit Group*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MECHANICAL_UNIT_GROUP

Type description

With the option *MultiMove* comes the possibility to control several robots from one controller. Each task can control one robot and up to six positioners. The mechanical units that will be controlled by one task are grouped in a mechanical unit group.

Related information

Use Mechanical Unit Group on page 166.

Application manual - MultiMove.

3.6.2 Name

Parent

Name belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

Name

Description

The name of the mechanical unit group.

Usage

This is the public identity of the mechanical unit group. It is used by the parameter *Use Mechanical Unit Group* in the type *Tasks*.

Limitations

Mechanical Unit Group is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

Use Mechanical Unit Group on page 166.

3.6.3 Robot

Parent

Robot belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

Robot

Description

Specifies the robot (with TCP), if there is any, in the mechanical unit group.

Usage

Robot is set to the same value as the parameter *Name* for the *Mechanical Unit Group* type that it represents.

Limitations

The parameter *Robot* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

Name on page 120.

3.6.4 Mechanical Unit 1, 2, 3, 4, 5, 6

Parent

Mechanical Unit 1, *Mechanical Unit 2*, *Mechanical Unit 3*, *Mechanical Unit 4*, *Mechanical Unit 5*, and *Mechanical Unit 6* belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

MechanicalUnit_1

MechanicalUnit_2

MechanicalUnit_3

MechanicalUnit_4

MechanicalUnit_5

MechanicalUnit_6

Description

*Mechanical Unit 1*specifies the first mechanical unit without TCP, if there is any, in the mechanical unit group.

Mechanical Unit 2 specifies the second mechanical unit without TCP, if there is more than one, in the mechanical unit group.

Mechanical Unit 3 specifies the third mechanical unit without TCP, if there are more than two, in the mechanical unit group.

Mechanical Unit 4 specifies the fourth mechanical unit without TCP, if there are more than three, in the mechanical unit group.

Mechanical Unit 5 specifies the fifth mechanical unit without TCP, if there are more than four, in the mechanical unit group.

Mechanical Unit 6 specifies the sixth mechanical unit without TCP, if there are more than five, in the mechanical unit group.

Usage

Mechanical Unit is set to the same value as the parameter *Name* for the *Mechanical Unit Group* type that it represents.

Limitations

The parameters *Mechanical Unit* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

Name on page 120.

3.6.5 Use Motion Planner

Parent

Use Motion Planner belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

UseMotionPlanner

Description

Specifies which motion planner shall be used for calculating the movements of the mechanical units in this group.

Usage

Use Motion Planner is set to the same value as the parameter *Name* for the *Motion Planner* type that you want to use.

Limitations

The parameter *Use Motion Planner* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

The Motion Planner type on page 577 in the topic *Motion*.

3.7 Type ModPos Settings

3.7.1 The ModPos Settings type

Overview

This section describes the type *ModPos Settings* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

HOTEDIT_MODPOS

Type description

It is sometimes desirable to limit how much a robtarget position can be moved by a ModPos or HotEdit operation. The limited deviation concerns both the linear distance and the orientation.

Limitations

There can be only one set of parameters of the type *ModPos Settings* in the system.

3.7.2 Name

Parent

Name belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

name

Description

Name defines that the parameter configuration is for ModPos.

Allowed values

modpos

Related information

Operating manual - IRC5 with FlexPendant.

3.7.3 Limited ModPos

Parent

Limited ModPos belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

type

Description

Limited ModPos defines if a ModPos change must be within a limited sphere for the position deviation and within a limited cone for the reorientation.

Usage

Set *Limited ModPos* to False when no limit is required, and to True when limits should apply.

Allowed values

FALSE or TRUE.

Default value is FALSE.

3.7.4 Mode

Parent

Mode belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

mode

Description

Mode defines how the limit is defined; to an absolute point or relative to the current position.

Usage

Setting *Mode* to Absolute means that the limited sphere/cone is around a fixed original point, i.e. position changes are accumulated and the accumulated deviation value is checked against the set max limits each time a change is made.

Setting *Mode* to Relative means that the limited sphere/cone is around the current point and will be moved when you modify the position.

Limitations

Mode is available only if *Limited ModPos* is set to TRUE.

Absolute is effective only on named robtargets, for example, p10, p20. * robtargets are not visible on the tree view.

Allowed values

Absolute or Relative.

Default value is Relative.

Related information

Limited ModPos on page 126.

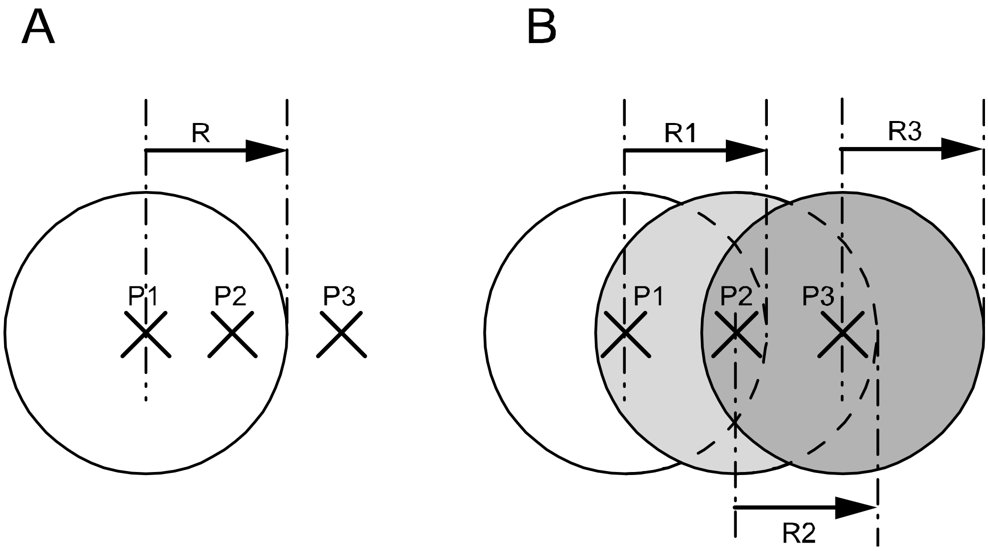
Example

In this example, the original point P1 is moved two times, first to P2 and then to P3. In figure A, *Mode* is set to Absolute, and in figure B, *Mode* is set to Relative.

The allowed move distance, R does not change in figure A. This makes it impossible to move the point to P3, as this is beyond R.

Continues on next page

In figure B however, the allowed move distance follows the last point. So from P1 it is possible to move as far as R1 allows, and from P2, it is allowed to move as far as R2, etc.



en0500001454

3.7.5 Limit Trans

Parent

Limit Trans belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limittrans

Description

Limit Trans defines the maximum allowed deviation in mm from the current or original position.

Usage

If *Limited ModPos* is set to TRUE, then *Limit Trans* is used by both ModPos and HotEdit, otherwise it is only used by HotEdit.

Allowed values

0 - 1000 mm.

Default value is 5.

Related information

Limited ModPos on page 126.

3.7.6 Limit Rot

Parent

Limit Rot belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitrot

Description

Limit Rot defines the maximum allowed reorientation in degrees from the current or original position.

Usage

If *Limited ModPos* is set to TRUE, then *Limit Rot* is used by both ModPos and HotEdit, otherwise it is only used by HotEdit.

Allowed values

0 - 360 degrees (0 - 6.280 radians).

Default value is 10 degrees (0.17 radians).

Additional information

Convert degrees to radians: `radians = (degrees/360)*(2*pi)`

Related information

Limited ModPos on page 126.

3.7.7 Limit External Trans

Parent

Limit External Trans belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitexttrans

Description

Limit External Trans defines the maximum allowed deviation in mm from the current or original position concerning external linear axes.

Usage

If *Limited ModPos* is set to TRUE, then *Limit External Trans* is used by both ModPos and HotEdit, otherwise it is only used by HotEdit.

Allowed values

0 - 1000 mm.

Default value is 50.

Related information

Limited ModPos on page 126.

3.7.8 Limit External Rot

Parent

Limit External Rot belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitextrot

Description

Limit External Rot defines the maximum allowed deviation in degrees from the current or original position concerning external rotational axes.

Usage

If *Limited ModPos* is set to TRUE, then *Limit External Rot* is used by both ModPos and HotEdit, otherwise it is only used by HotEdit.

Allowed values

0 - 360 degrees (0 - 6.280 radians).

Default value is 10 degrees (0.17 radians).

Additional information

Convert degrees to radians: `radians = (degrees/360)*(2*pi)`

Related information

Limited ModPos on page 126.

3.8 Type Operator Safety

3.8.1 The Operator Safety type

Overview

This section describes the type *Operator Safety* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MASTER_BOOL

Type description

The *Operator Safety* type is used to define extra safety for system execution.

Related information

How to activate hold-to-run control on page 94.

Operating manual - IRC5 with FlexPendant, chapter *Safety*.

3.8.2 Function

Parent

Function belongs to the type *Operator Safety*, in the topic *Controller*.

Cfg name

Name

Description

Function defines safety functions for the robot system.

Allowed values

Value	Description
Hold-to-run	Hold-to-run enables a functionality that requires a button to be pressed in to allow execution in Manual Reduce Speed mode. When the button is released the executions are immediately stopped. Hold-to-run is always activated in Manual Full Speed operating mode. Hold-to-run is further described in standard ISO 10218 (EN775).

Related information

How to activate hold-to-run control on page 94.

Operating manual - IRC5 with FlexPendant chapter *Safety*.

3.8.3 Active

Parent

Active belongs to the type *Operator Safety*, in the topic *Controller*.

Cfg name

Select

Description

Active defines whether the value of *Function* is activated.

Allowed values

Value	Description
TRUE	Activated
FALSE	Not activated

Related information

Function on page 134.

3.9 Type Options

3.9.1 The Options type

Overview

This section describes the type *Options*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

PRESENT_OPTIONS

Type description

Options contains read-only names and descriptions of the installed options in the system.

3.9.2 Name

Parent

Name belongs to the type *Options*, in the topic *Controller*.

Cfg name

name

Description

Short unique ID of an option.

Usage

Uniquely identifies an option.

Limitations

Read-only

3.9.3 Description

Parent

Description belongs to the type *Options*, in the topic *Controller*.

Cfg name

desc

Description

Complete name of an option.

Usage

Human friendly identification of an option.

Limitations

Read-only

3.10 Type Path Return Region

3.10.1 The Path Return Region type

Overview

This section describes the type *Path Return Region* which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

CAB_REGAIN_DIST

Type description

In a return movement, the path return region specifies the distance from the current robot position to the last executed path. The maximum path region can be set both for start in manual mode and for start in automatic mode.

There must be two sets of parameters defined for this type; one for automatic mode (AUTO) and one for manual mode (MAN). Both are predefined on delivery.

Return movements

A return movement must take place when the current path of the robot deviates from the programmed path. For example, this is required when an uncontrolled stop has occurred or when the robot has been jogged away from its path.

A return movement begins when program start is ordered and stops before the program continues with the instruction that was interrupted due to a stop request.

Predefined path return regions

AUTO

MAN

3.10.2 Mode

Parent

Mode belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Name

Description

Mode defines in which operating mode a return movement will start.

Usage

Both Auto and Man mode must be defined in the system and are configured on delivery.

Allowed values

AUTO

MAN

3.10.3 TCP Distance

Parent

TCP Distance belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

TCP_Dist

Description

TCP Distance defines the maximum allowed TCP distance from the current robot position to the last executed path.

Usage

TCP Distance is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-2.000 meters, specifying the movement in meters.

The default value is 0.05 meter for manual mode and 0.5 meter for automatic mode.

Related information

Mode on page 140.

Application manual - Controller software IRC5.

3.10.4 TCP Rotation

Parent

TCP Rotation belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

TCP_Rot

Description

TCP Rotation defines the maximum allowed TCP rotation from the current robot position to the last executed path.

Usage

TCP Rotation is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-6.280, specifying the movement in radians.

The default value is 0.2 radians for manual mode and 1.57 radians for automatic mode.

Additional information

To convert degrees to radians, use this formula:

```
radians = 2*pi*degrees/360
```

Related information

Mode on page 140.

Application manual - Controller software IRC5.

3.10.5 External Distance

Parent

External Distance belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Ext_Dist

Description

External Distance defines the maximum allowed external axes distance from the current robot position to the last executed path.

Usage

External Distance is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-2.000, specifying the movement in meters.

The default value is 0.05 meter for manual mode and 0.5 meter for automatic mode.

Related information

Mode on page 140.

Application manual - Controller software IRC5.

3.10.6 External Rotation

Parent

External Rotation belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Ext_rot

Description

External Rotation defines the maximum allowed external axes rotation from the current robot position to the last executed path.

Usage

External Rotation is used to limit the regain movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-6.280, specifying the movement in radians.

The default value is 0.2 radians for manual mode and 1.57 radians for automatic mode.

Additional information

To convert degrees to radians, use this formula:

```
radians = 2*pi*degrees/360
```

Related information

Mode on page 140.

Application manual - Controller software IRC5.

3.11 Type Run Mode Settings

3.11.1 The Run Mode Settings type

Overview

This section describes the type *Run Mode Settings* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

RUN_MODE_SETTINGS

Type description

The type *Run Mode Settings* defines if the run mode should change when changing operating mode.

3.11.2 Name

Parent

Name belongs to the type *Run Mode Settings*, in the topic *Controller*.

Cfg name

name

Description

Name of the operating mode setting.

Usage

There can be only one instance with each allowed value, that is a maximum of two instances in the system.

Allowed values

Value	Description
AutoToManual	Defines settings when switching from automatic to manual operating mode.
ManualToAuto	Defines settings when switching from manual to automatic operating mode.

3.11.3 Switch

Parent

Switch belongs to the type *Run Mode Settings*, in the topic *Controller*.

Cfg name

SwitchTo

Description

Switch defines the run mode when switching operating mode.

Usage

Defines if the run mode should be changed when changing operating mode.

Allowed values

Value	Description
Keep	Keep current run mode.
Single	Set run mode to single cycle.
Continuous	Set run mode to continuous.

3.12 Type Safety Run Chain

3.12.1 The Safety Run Chain type

Overview

This section describes the type *Safety Run Chain* which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

RUNCHN_BOOL

Type description

It is possible to have a delayed stop that gives a smooth stop, this is called a Soft Stop. The type of Soft Stop used in the system is defined in the *Safety Run Chain* type.

There can be more than one set of parameters defined for the type *Safety Run Chain*.

Soft Stop

In the Soft Stop, the robot stops in the same way as a normal program stop with no deviation from the programmed path. After approximately 1 second the power supply to the motors shuts off. The stopping distance can be longer than at a hard stop (e.g. emergency stop) where the power shuts off immediately.

3.12.2 Function

Parent

Function belongs to the type *Safety Run Chain*, in the topic *Controller*.

Cfg name

Name

Description

Function defines if Soft Stop is activated or deactivated.

Usage

A Soft Stop is the same as a safety stop of category 1, which means that the robot will be stopped by the servos.

The robot can easily be started again after a soft stop on path and continue with the activity where it was interrupted.

Allowed values

Value:	Description:
SoftES	Soft emergency stop is activated by pressing the emergency stop push button on the FlexPendant or the control module. SoftES is only used in Auto. In manual mode, SoftES will be a category 0 stop regardless of the value set in the parameter *Active*.
SoftAS	Soft automatic mode stop is intended for automatic mode during normal program execution. This stop is activated by safety devices such as light curtains, light beams, or sensitive mats.
SoftGS	Soft general stop is activated by safety devices such as light curtains, light beams, or sensitive mats.
SoftSS	Soft superior stop has the same function as a general stop but is intended for externally connected safety devices.

Related information

Operating manual - IRC5 with FlexPendant chapter *Safety*.

3.12.3 Active

Parent

Active belongs to the type *Safety Run Chain*, in the topic *Controller*.

Cfg name

Select

Description

Active defines whether the Soft Stop is activated or not.

Usage

If *Active* is set to True, the Soft Stop is activated.

Allowed values

TRUE or FALSE.

The Soft Stops are defined with default values.

Soft Stop:	Default value:	Description:
SoftES	FALSE	Deactivated
SoftAS	TRUE	Activated
SoftGS	TRUE	Activated
SoftSS	TRUE	Activated

3.13 Type General Rapid

3.13.1 The General Rapid type

Overview

This section describes the type *General Rapid*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SYS_MISC

Type description

General Rapid contains parameters that are general for the controller.

3.13.2 Name

Parent

Name belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

name

Description

Name defines either of the two arguments for retry or simulated menus.

Limitations

There can be only one instance with *Name* set to *NoOfRetry*, *SimulateMenu*, *ModalPayLoadMode*, *StationaryPayloadMode*, or *CollisionErrorHandling*.

Allowed values

CollisionErrorHandler

NoOfRetry

SimulateMenu

ModalPayLoadMode

StationaryPayLoadMode

CollisionErrorHandling

Related information

NoOfRetry on page 153.

SimulateMenu on page 154.

ModalPayLoadMode on page 155.

StationaryPayLoadMode on page 156

CollisionErrorHandling on page 157

3.13.3 Action values

3.13.3.1 NoOfRetry

Parent

NoOfRetry is an action value for the parameter *Name* that belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

NoOfRetry

Description

The action value *NoOfRetry* specifies that there is a limit to the number of times the routine with a recoverable error is called before the error is reported as fatal and execution is stopped. The number of times is set by the parameter *Value*.

Usage

Can be useful, for example, if the network is shaky and the first attempt at opening a file does not work.

Limitations

Works only if an ERROR handler that takes care of the error situation is programmed with the RETRY statement.

Additional information

Changes are activated after a normal restart.

Related information

Value on page 158.

Example

This example shows that it can take some time before an I/O unit is enabled. Several attempts are needed before it is possible to set the digital output signal.

```
PROC A()
  ...
  IOEnable "cell_1", 0;
  SetDO cell_1_sig3, 1; !This might not work on the first attempt
  ...
ERROR IF ERRNO = ERR_IOENABLE THEN
    RETRY;
  ENDIF
ENDPROC
```

3.13.3.2 SimulateMenu

Parent

SimulateMenu is an action value for the parameter *Name* that belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

SimulateMenu

Description

The `WaitTime`, `WaitUntil`, `WaitDO`, and `WaitDI` instructions generate an alert box in manual mode to make it possible to simulate the instruction and continue to execute the next instruction. The parameter *Value* defines if *SimulateMenu* is on or off.

Usage

It is useful to switch this parameter off if no alert boxes are desired. Set *Value* to 0 to disable menus.

Limitations

The parameter is only active in manual mode. There are no alert boxes in automatic mode.

Additional information

Changes are activated after a normal restart.

Related information

Value on page 158.

3.13.3.3 ModalPayLoadMode

Parent

ModalPayLoadMode is an action value for the parameter *Name* that belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

ModalPayLoadMode

Description

ModalPayLoadMode defines whether or not *ModalPayLoadMode* shall be used. When *ModalPayLoadMode* is used, any payload is set by the *GripLoad* instruction. When *ModalPayLoadMode* is not used, the optional argument *TLoad* is used for setting payload.

Usage

Can be useful, for example, if the modal instruction GripLoad is not desirable.

Allowed values

Name:	Value:	Description:
ModalPayLoadMode	1	ModalPayLoadMode shall be used. Any payload is set by the GripLoad instruction. This is a default value.
	0	ModalPayLoadMode shall not be used, instead the optional argument TLoad is used. The argument TLoad is available on all motion instructions.

Additional information

Changes are activated after a normal restart.

Related information

For more information about *GripLoad* and *TLoad*, see*Technical reference manual - RAPID Instructions, Functions and Data types*.

3.13.3.4 StationaryPayLoadMode

Parent

StationaryPayLoadMode is an action value for the parameter *Name* that belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

StationaryPayLoadMode

Description

StationaryPayLoadMode defines whether or not *StationaryPayLoadMode* shall be used. The *StationaryPayLoadMode* can only be used if a stationary tool is used. When *StationaryPayLoadMode* is used, any payload is added relative to the wrist coordinate system. When *StationaryPayLoadMode* is not used, any payload is added relative to the work object.

Usage

Can be useful, for example, if several work objects are used for one stationary tool. In this case only one Load Identification is needed instead of one for each work object.

Limitations

The parameter *StationaryPayLoadMode* will only impact if a stationary tool is used.

Allowed values

Name:	Value:	Description:
StationaryPayLoadMode	0	*StationaryPayLoadMode* shall not be used, any payload is added relative to the work object. This is the default value.
	1	*StationaryPayLoadMode* shall be used. Any payload is added relative to the wrist.

Additional information

Changes are activated after a normal restart

Related information

For more information about how loads are added, see*Technical reference manual - RAPID Instructions, Functions and Data types*, section *loaddata – Load data*.

3.13.3.5 CollisionErrorHandling

Parent

CollisionErrorHandling is an action value for the parameter *Name* that belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

CollisionErrorHandling

Description

Defines if the execution shall stop or not when a motion collision occurs. If *CollisionErrorHandling* is set the execution will continue to the Error handler.

Usage

Used if it is possible to execute after some error handling after a collision.

Allowed values

YES or NO

Default value is NO.

Additional information

Changes are activated after a normal restart

Related information

For detailed information about collision detection, see *Collision Detection* in *Application manual - Controller software IRC5*.

Technical reference manual - RAPID kernel

3.13.4 Value

Parent

Value belongs to the type *General Rapid*, in the topic *Controller*.

Cfg name

value

Description

Defines the values for the action values defined in parameter *Name*.

Allowed values

Name:	Value:	Description:
NoOfRetry	1-1000	Defines number of times the number of times a routine with a recoverable error is called before the system is stopped.
SimulateMenu	0 or 1	Defines if instructions should be possible to simulate in manual mode.
ModalPayLoadMode	0 or 1	Defines whether or not *ModalPayLoadMode* shall be used. When *ModalPayLoadMode* is used, any payload is set by the `GripLoad` instruction. When *ModalPayLoadMode* is not used, the optional argument `TLoad` is used for setting payload.
StationaryPayLoad-Mode	0 or 1	Defines whether or not *StationaryPayLoadMode* shall be used. The *StationaryPayLoadMode* can only be used if a stationary tool is used. When *StationaryPayLoadMode* is used, any payload is added relative to the wrist coordinate system. When *StationaryPayLoadMode* is not used, any payload is added relative to the work object.
CollisionErrorHand-ling	YES or NO	Defines whether or not *CollisionErrorHandling* shall be used.

Related information

Name on page 152.

NoOfRetry on page 153.

SimulateMenu on page 154.

ModalPayLoadMode on page 155.

StationaryPayLoadMode on page 156.

CollisionErrorHandling on page 157.

3.14 Type Task

3.14.1 The Task type

Overview

This section describes the type *Task*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

CAB_TASKS

Type description

Each set of parameters of the *Task* type represents a program task on the controller.

If you have the option *Multitasking*, there can be up to 20 tasks. Otherwise there can be only one.

Related information

Application manual - Controller software IRC5 chapter *Multitasking*.

3.14.2 Task

Parent

Task belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Name

Description

The name of the task.

Usage

This is the public identity of the task.

Allowed values

A string with maximum 30 characters. The first character may not be a digit.

Limitations

Editing the task entry in the configuration editor and changing the task name will remove the old task and add a new one. This means that any program or module in the task will disappear after a restart with these kind of changes.

3.14.3 Task in Foreground

Parent

Task in Foreground belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Task_in_foreground

Description

Used to set priorities between tasks.

Task in Foreground contains the name of the task that should run in the foreground of this task. This means that the task for which the parameter is set will only execute if the foreground task is idle.

Usage

The default behavior is that all tasks run at the same priority level. If you want to customize the priorities, the *Task in Foreground* parameter can be set for the tasks that should run in the background.

If *Task in Foreground* is set to empty string or to -1 for a task, it runs at the highest priority, i.e. no other task can suspend its execution.

Limitations

The parameter *Task in Foreground* can only be used if you have the option *Multitasking*.

Allowed values

A string with maximum 30 characters.

3.14.4 Type

Parent

Type belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Type

Description

Controls the start/stop and system restart behavior of a task.

Usage

When creating a new task, use the *Type* parameter to configure how the task should be started.

Limitations

A task that controls a mechanical unit must be of the type NORMAL.

The parameter *Type* can only be used if you have the option *Multitasking*.

Allowed values

Value:	Description:
NORMAL	The task reacts on START/STOP requests given from the FlexPendant or other sources. The task is stopped when an emergency stop occurs.
STATIC	At restart, the task restarts at the current position. The task is not stopped by emergency stops. The task is normally not stopped by the stop button on the FlexPendant. This can be configured on the FlexPendant by the operator.
SEMISTATIC	The task restarts from the beginning at all restarts. Modules will be reloaded if the file with automatic loaded modules is updated. The task is not stopped by emergency stops. The task is normally not stopped by the stop button on the FlexPendant. This can be configured on the FlexPendant by the operator.

Default value is SEMISTATIC.

3.14.5 Check Unresolved References

Parent

Check Unresolved References belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

BindRef

Description

Check Unresolved References determines if the system shall check for unresolved references or ignore them.

Usage

This parameter should be set to "0" if the system is to accept unresolved references in the program while linking a module, or otherwise set to "1".

If set to "1", a runtime error will occur on execution of an unresolved reference.

Limitations

The parameter has no effect when using instructions `Load`, `StartLoad`, `WaitLoad`, or `Erase`. In this case the system will never check for unresolved references.

Allowed values

1 or 0.

Default value is 1.

3.14.6 Main Entry

Parent

Main Entry belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Entry

Description

The name of the start routine for the task.

Usage

The task starts its execution in the routine specified by *Main Entry*. It should be a RAPID routine without any parameters and reachable in this task.

Allowed values

A routine name, with maximum 32 characters.

Default value is main.

3.14.7 Trustlevel

Parent

Trustlevel belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Trustlevel

Description

Trustlevel handles the system behavior when a SEMISTATIC or STATIC task is stopped or not executable.

Usage

If a task that handles safety supervision stops, it might be dangerous to continue running the task that controls the robot motion. Use *TrustLevel* to set the behavior of NORMAL tasks when a SEMISTATIC or STATIC task stops.

Tip

To simplify debugging of background tasks you can make all tasks (including the background tasks) visible in the task panel on the FlexPendant. Then, in manual mode, all tasks that are selected in the task panel (including background tasks) will stop when pressing the stop button.

See **Task Selection Panel Settings** in the **Control Panel** for FlexPendant.

Limitations

The parameter *Trustlevel* can only be used if you have the option *Multitasking*.

Allowed values

Value:	Description:
SysFail	All NORMAL tasks will be stopped. Besides that the system is set to system failure state (SYS_FAIL). All jogging and program start orders will be rejected. Only a new normal restart resets the system. This should be used when the task has some safety supervisions.
SysHalt	All NORMAL tasks will be stopped. The system is forced to Motors off state. Taking up the system to Motors on resets the system.
SysStop	All NORMAL tasks will be stopped but are restartable. Jogging is also possible.
NoSafety	Only the task itself will stop.

The default value is SysFail.

Related information

Operating manual - IRC5 with FlexPendant.

3.14.8 Use Mechanical Unit Group

Parent

Use Mechanical Unit Group belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

UseMechanicalUnitGroup

Description

Defines which mechanical unit group is used for the task.

Usage

A motion task (*MotionTask* set to Yes) controls the mechanical units in the mechanical unit group. A non-motion task (*MotionTask* set to No) will still be able to read values (e.g. the TCP position) for the mechanical units in the mechanical unit group.

Limitations

The parameter *Use Mechanical Unit Group* is only used if you have the option *MultiMove*.

Allowed values

Use Mechanical Unit Group is set to the same value as the parameter *Name* for the type *Mechanical Unit Group*.

A string with maximum 32 characters.

Related information

MotionTask on page 167.

Name on page 120.

Application manual - MultiMove.

3.14.9 MotionTask

Parent

MotionTask belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

MotionTask

Description

Indicates which task is the motion task, e.g. can be able to run RAPID move instructions. *MotionTask* must be used even though only one task is configured in the system.

Usage

Set *MotionTask* to YES for the task that will be used for robot move instructions.

Limitations

Only one task in the system can be a motion task unless you have the option *MultiMove*.

The parameter *MotionTask* is only used if you have the option *Multitasking*.

Allowed values

YES or NO.

The default behavior is NO.

The value must be set to YES for one, and only one, task.

Related information

Application manual - MultiMove.

Application manual - Controller software IRC5.

3.14.10 Hidden

Parent

Hidden belongs to the type *Task* in the topic *Controller*.

Cfg name

Hidden

Description

RAPID tasks may be hidden, which may be used to prevent inexperienced end users from tampering (accidentally deleting or changing) with the contents.

Note that the hidden contents is not protected! It can easily be shown again by setting the parameter value to NO.

Note that any hidden contents will still be available when using the `SetDataSearch` instruction to search RAPID data.

Limitation

This parameter is available when using multitasking systems only, such as *MultiMove*.

Changes to the parameter will become effective only after using the restart mode **Reset RAPID**.

Allowed values

YES or NO.

Default value is NO.

3.14.11 RMQ Type

Parent

RMQ Type belongs to the type *Task*, in the topic *Controller*.

Cfg name

RmqType

Description

Used for the functionality *RAPID Message Queue*. *RMQ Type* defines if the queue of this RAPID task should accept messages from anyone, only other tasks on the same controller, or from no one.

Usage

RMQ Type can be used to turn off all *RAPID Message Queue* communication to a RAPID task. It can also be used to limit the communication so that only other RAPID tasks on the same controller may send messages to this task.

Limitations

The parameter *RMQ Type* is only used if you have the functionality *RAPID Message Queue*.

Allowed values

Value:	Description:
None	Disable the receiving of *RAPID Message Queue* messages in this RAPID task.
Internal	Enable the receiving of *RAPID Message Queue* messages from other tasks on the controller.
Remote	Enable the receiving of *RAPID Message Queue* messages both from other tasks on the controller, from the FlexPendant and from PC applications.

The default value is *None*.

Related information

For more information about *RAPID Message Queue*, see *Application manual - Controller software IRC5*, section *RAPID Message Queue*.

3.14.12 RMQ Max Message Size

Parent

RMQ Max Message Size belongs to the type *Task*, in the topic *Controller*.

Cfg name

RmqMaxMsgSize

Description

The maximum data size, in bytes, for a *RAPID Message Queue* message.

Usage

The default value is 400, and there is normally no reason to change this value. The value cannot be changed in RobotStudio or on the FlexPendant. The only way to change the value is by editing the sys.cfg file.

Limitations

The parameter *RMQ Max Message Size* is only used if you have the functionality *RAPID Message Queue*.

Allowed values

An integer between 400 and 3000.

Default value is 400.

Related information

For more information about *RAPID Message Queue*, see *Application manual - Controller software IRC5*, section *RAPID Message Queue*.

3.14.13 RMQ Max No Of Messages

Parent

RMQ Max No Of Messages belongs to the type *Task*, in the topic *Controller*.

Cfg name

RmqMaxNoOfMsg

Description

Maximum number of *RAPID Message Queue* messages in the queue to this task.

Usage

The default value is 5, and there is normally no reason to change this value. The value cannot be changed in RobotStudio or on the FlexPendant. The only way to change the value is by editing the sys.cfg file.

Limitations

The parameter *RMQ Max No Of Messages* is only used if you have the functionality *RAPID Message Queue*.

Allowed values

An integer between 1 and 10.

Default value is 5.

Related information

For more information about *RAPID Message Queue*, see *Application manual - Controller software IRC5*, section *RAPID Message Queue*.

3.14.14 RMQ Mode

Parent

RMQ Mode belongs to the type *Task*, in the topic *Controller*.

Cfg name

RmqMode

Description

Used for functionality *RAPID Message Queue*. *RMQ Mode* defines which mode the message queue for this task will use.

Usage

RMQ Mode defines the message queue handling should be based on interrupts (data types) or synchronous (all messages are handled).

Limitations

The parameter *RMQ Mode* is only used if you have the functionality *RAPID Message Queue*.

Allowed values

Value:	Description:
Interrupt	A message can only be received by connecting a trap routine to a specified message type. See instruction `IRMQMessage`.
Synchronous	A message can only be received by executing an `RMQReadWait` instruction.

The default value is Interrupt.

Related information

For more information about *RAPID Message Queue*, see *Application manual - Controller software IRC5*, section *RAPID Message Queue*.

RAPID instructions are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

4 Topic I/O System

4.1 The I/O System topic

Overview

This chapter describes the types and parameters of the *I/O System* topic. Each parameter is described in the section for its type.

Description

The *I/O System* topic contains parameters for I/O devices and signals.

The parameters are organized in the following types:

1. Access Level
2. Industrial Network
3. Cross Connection
4. Device Trust Level
5. Device Command
6. Device
7. Internal Device
8. Signal Safe Level
9. Signal
10. System Input
11. System Output
12. Route

Configuration results

The changed I/O System parameters are effective after a restart of the robot controller.

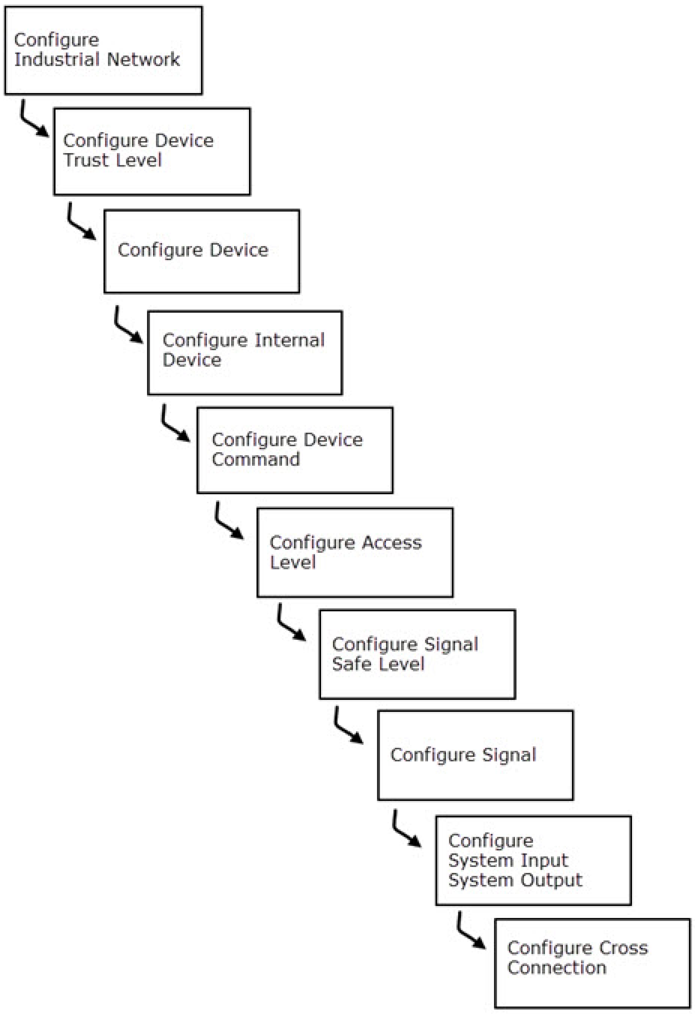
4.2 Workflows

4.2.1 How to configure an industrial network

Overview

There is a systematic way to configure the parameters before actually operating the I/O system. This is an overview of how to configure the industrial networks, I/O devices, and I/O signals in the I/O system. For different industrial network configuration details, refer to the respective application manuals.

The following diagram shows the systematic way of configuring the different parameters to set up the I/O system.



xx1500000354

4.2.2 How to define I/O devices

Overview

I/O device is a logical software representation in I/O system of a physical device that is connected to an industrial network handled by the robot controller. I/O devices allow you to control electronic devices and read sensor data. They are used for controlling I/O signals in the robot system.

Available I/O devices

Several I/O devices can be defined within the robot system. The types of I/O devices available, depend on what type of industrial network is being used.

The following are examples of available I/O devices for DeviceNet:

1. Digital I/O
2. Analog I/O
3. AD Combi I/O
4. Relay I/O
5. Gateways
6. Simulated I/O
7. Encoder interface devices

Prerequisites

Before defining an I/O device, you must:

1. Configure parameters on the *Industrial Network*, if necessary.
2. Make sure the appropriate *Device Trust Level* is available, either by creating it or using a predefined device trust level.

How to define I/O devices

To define an I/O device:

1. In the topic **I/O System**, choose the type **Device**.
2. Select the I/O device to change, delete, or add a new one.
3. Enter, delete, or change the values for the parameters.
4. Save the changes.
5. Restart the controller.

Related information

Type Device on page 213

The Device Trust Level type on page 202

4.2.3 How to define input and output I/O signals

Overview

An I/O signal is the logical software representation of a:

- Inputs or outputs located on an I/O device that is connected to an industrial network within the robot system (real I/O signal).
- An I/O signal without a representation on any I/O device (simulated I/O signal).

Available input and output I/O signals

The I/O signals can be of different types.

The type of I/O signals available depends on the type of I/O device. Typical I/O signal types on an I/O device are:

- Digital inputs and outputs 24 V DC
- Digital inputs and outputs 120 V DC
- Analog inputs and outputs ±10 V
- Analog outputs 0 to +10 V

The I/O signal types possible to configure in the robot system are:

- Digital input, DI
- Digital output, DO
- Analog input, AI
- Analog output, AO
- Group input, GI
- Group output, GO

Limitations

Maximum 12000 user I/O signals can be defined in the robot system. This includes digital, analog, and group I/O signals of both input and output type.

Prerequisites

Before defining an I/O signal, you must:

1. Configure the *Device.*
2. Make sure the appropriate *Access Level* is available, either by creating it or by using a predefined access level.
3. Make sure the appropriate *Safe Level* is available, either by creating it or by using a predefined safe level.

How to define input and output I/O signals

To define I/O signals:

1. In the topic **I/O System**, choose the type **Signal.**
2. Add a new one or select an existing I/O signal to be changed or deleted.
3. Save the changes.
4. Restart the controller.

Continues on next page

Related information

How to define an I/O signal group on page 178.

The Signal type on page 237.

Type Signal Safe Level on page 230.

4.2.4 How to define an I/O signal group

Signal group

Digital inputs or outputs located on an I/O device can be grouped and handled as one I/O signal in the robot system. The value of such an I/O signal will thus be a positive integer that is binary coded using the individual digital inputs or outputs on the I/O device as a basis.

Limitations

When defining I/O signal groups, you have to consider the following limitation in the robot system:

- Maximum 32 inputs and outputs located on an I/O device can be defined in an I/O signal group.

How to define an I/O signal group

To define an I/O signal group:

1. In the **I/O System** topic, choose the type **Signal**.
2. Add a new one or select an existing I/O signal to be changed or deleted.
3. Enter, delete, or change the values for the parameters. Set the parameter *Type of Signal* to value *Group Input* or *Group Output*.

   The required parameters depend on the type of signal. See parameter descriptions and examples of typical configurations in the description of the type *Signal*.
4. Save the changes.
5. Restart the controller.

Related information

How to define input and output I/O signals on page 176.

The Signal type on page 237.

Type Signal Safe Level on page 230.

Example

If an I/O signal group spans over 4 digital inputs on the I/O device, the maximum value is 15 (2^4-1) and the minimum value is 0.

4.2.5 How to define system inputs

Overview

Input I/O signals can be assigned specific system inputs. The input triggers a system action that is handled by the system, without using the FlexPendant or other hardware devices.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Limitations

The following limitations have to be considered:

- Only one system action can be assigned to the input I/O signal. However, several input I/O signals can be assigned the same system action.
- When deleting a system action, the I/O signal itself remains defined. The I/O signal has to be deleted separately.
- System input I/O signals are only valid for the currently executed program in the system, with exceptions on the action value level. These exceptions are described together with the corresponding action value.
- The system must be in automatic mode to react on the system signal.

How to define system inputs

To define a system input:

1. In the topic **I/O System**, choose the type **System Input**.
2. Select the system input to change, delete, or add a new one.
3. Enter, change, or delete the values for the parameters.

   To add or delete the system action values *Interrupt*, *Load and Start*, *Motors On and Start*, *Start*, and *Start at Main*, you must also define the parameter *Argument 1*.

   To add or delete the system action values *Interrupt* and *Load and Start*, you must also define the parameter *Argument 2*.
4. Save the changes.
5. Restart the controller.

Rejected system inputs

If the system is in manual mode or cannot perform the defined system action due to any other unfulfilled requirement, no error message is displayed. When a system action is rejected the error message is stored in the error log (ELOG).

Related information

The System Input type on page 266.

The Signal type on page 237.

4.3 Type Access Level

4.3.1 The Access Level type

Overview

This section describes the *Access Level* type which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_ACCESS

Type description

An *Access Level* type is a configuration that defines the write access to I/O signals for categories of I/O controlling clients connected to the robot controller.

Usage

To limit write access to I/O signals from clients it is necessary to use an access level. The access level settings differentiates local clients (for example, FlexPendant) from remote clients (for example, RobotStudio).

Limitations

It is not possible to configure different write access levels for different remote clients, since the controller does not differentiate, for example, RobotStudio from other remote clients.

Predefined access levels

Access Level:	Description:
ReadOnly	No client has write access, typically used by read only I/O signals. This access level cannot be changed.
Default	Only allowed to write to signals from RAPID instructions and local clients (for example FlexPendant) in manual mode. This access level cannot be changed.
All	All clients, local and remote, have write access. This access level cannot be changed.
Internal	No clients have write access, typically used for system internal I/O signals, for example safety I/O signals. This access level cannot be changed.

Example

In this example, it is possible to modify only I/O signals with this access level with RAPID and local clients in manual mode. Remote clients cannot modify these I/O signals.

Parameter:	Value:
Name	Default

Continues on next page

Parameter:	Value:
Rapid	Write enabled
Local client in manual mode	Write enabled
Local client in auto mode	Read only
Remote client in manual mode	Read only
Remote client in auto mode	Read only

4.3.2 Name

Parent

The parameter *Name* belongs to the type *Access Level*, in the topic *I/O System*.

Cfg name

Name

Description

The parameter *Name* specifies the logical name of the access level.

Usage

The name of the access level is used as a reference to the specific access level when configuring the I/O signals.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named I/O objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

4.3.3 Rapid

Parent

The parameter *Rapid* belongs to the type *Access Level*, in the topic *I/O System*.

Cfg name

Rapid

Description

The parameter *Rapid* specifies the level of access granted to RAPID instructions.

Usage

Specify the level of access that should be granted to RAPID instructions when accessing objects associated with this access level.

Default value

The default value is Read only.

Allowed values

Write enabled

Read only

4.3.4 Local Client in Manual Mode

Parent

The parameter *Local Client in Manual Mode* belongs to the type *Access Level*, in the topic *I/O System*.

Cfg name

LocalManual

Description

The parameter *Local Client in Manual Mode* specifies the level of access granted to local RobAPI clients in manual mode.

A local client is a client using RobAPI and is connected directly to the controller, for example the FlexPendant.

Usage

Specifies the level of access that should be granted to local RobAPI clients in manual mode, when accessing objects associated with this access level.

Default value

The default value is Read only.

Allowed values

Write enabled

Read only

4.3.5 Local Client in Auto Mode

Parent

The parameter *Local Client in Auto Mode* belongs to the type *Access Level*, in the topic *I/O System*.

Cfg name

LocalAuto

Description

The parameter *Local Client in Auto Mode* specifies the level of access granted to local RobAPI clients in automatic mode.

A local client is a client using RobAPI and is connected directly to the controller, for example the FlexPendant.

Usage

Specify the level of access that should be granted to local RobAPI clients in automatic mode when accessing objects associated with this access level.

Default value

The default value is Read only.

Allowed values

Write enabled

Read only

4.3.6 Remote Client in Manual Mode

Parent

The parameter *Remote Client in Manual Mode* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

RemoteManual

Description

The parameter *Remote Client in Manual Mode* specifies the level of access granted to remote RobAPI clients in manual mode.

A remote client is a client or application using RobAPI and not being connected directly to the controller, for example RobotStudio.

Usage

Specify the level of access that should be granted to remote RobAPI clients in manual mode when accessing objects associated with this access level.

Default value

The default value is Read only.

Allowed values

Write enabled

Read only

4.3.7 Remote Client in Auto Mode

Parent

The parameter *Remote Client in Auto Mode* belongs to the type *Access Level*, in the topic *I/O System*.

Cfg name

RemoteAuto

Description

The parameter *Remote Client in Auto Mode* specifies the level of access granted to remote RobAPI clients in automatic mode.

A remote client is a client or application using RobAPI and not being connected directly to the controller, for example RobotStudio.

Usage

Specify the level of access that should be granted to remote RobAPI clients in automatic mode when accessing objects associated with this access level.

Default value

The default value is Read only.

Allowed values

Write enabled

Read only

4.4 Type Industrial Network

4.4.1 The Industrial Network type

Overview

This section describes the type *Industrial Network*, which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

INDUSTRIAL_NETWORK

Type description

An Industrial Network is a logical software representation of a real industrial network within the controller.

Usage

By specifying an Industrial Network, a logical representation of the real industrial network is created. The network configuration defines the specific parameters that will determine the behavior for the industrial network, like communication speed, address, connection, etc.

The Industrial Network is used when defining the I/O devices and other objects in the I/O system.

Prerequisites

Before configuring parameters on the Industrial Network, the industrial network option must be installed.

The industrial network option typically consists of software to configure Industrial Networks of the specific type, and the hardware required to equip the controller with the physical interfaces needed for the specific network.

Limitations

The Industrial Network has the following limitations:

- The maximum number of Industrial Network in the system depends on the installed network options.
- It is only possible to configure Industrial Networks of types for which the respective industrial network option has been installed in the system.

Predefined industrial networks

Industrial Network:	Description:
Local	Local is used for communication with the safety I/O boards. No extra user defined I/O devices can be configured to this Industrial Network.

Depending on the installed industrial network options, there can be other predefined industrial networks not described in this manual.

Continues on next page

Related information

More information about the industrial network configuration can be found in the manual for the respective industrial network option, for example *Application manual - DeviceNet Master/Slave*.

Example DeviceNet

This is an example for a DeviceNet industrial network. For more information about DeviceNet, refer *Application manual - DeviceNet Master/Slave*.

Parameter:	Value:
Name	DeviceNet
Identification Label	DeviceNet Master/Slave
Address	2
DeviceNet Communication Speed	250 kbps

4.4.2 Name

Parent

Name belongs to the type *Industrial Network*, in the topic *I/O System*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the industrial network.

Usage

The name of the *Industrial Network* is used as a reference to the specific network when configuring the I/O devices on the industrial network.

The following names are allowed for the industrial networks:

- DeviceNet
- DeviceNet_Anybus
- PROFIBUS
- PROFIBUS_Anybus
- EtherNetIP
- EtherNetIP_Anybus
- PROFINET
- PROFINET_Anybus
- Local
- ICI

Default value

The default value is specified by the specific industrial network option.

Allowed values

A string of maximum 32 characters. The allowed value(s) is specified by the specific industrial network option.

4.4.3 Identification Label

Parent

Identification Label belongs to the type *Industrial Network*, in the topic *I/O System*.

Cfg name

Label

Description

Identification Label provides a way to identify the industrial network physically.

Usage

Using *Identification Label* is optional. It provides a label to identify the physical industrial network or hardware communication interface (connection port) that this network configuration is representing.

Default value

The default value is an empty string.

Allowed values

A string with maximum 80 characters.

4.4.4 Simulated

Parent

Simulated belongs to the type *Industrial Network*, in the topic *I/O System*.

Cfg name

Simulated

Description

The parameter Simulated specifies that the industrial network and all I/O devices connected to it should be treated as simulated.

Usage

The parameter *Simulated* defines that the entire industrial network is simulated.

Default value

The default value is No.

Allowed values

Yes

No

4.5 Type Cross Connection

4.5.1 The Cross Connection type

Overview

This section describes the type *Cross Connection* which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_CROSS

Type description

A cross connection is a logical connection between I/O signals of type digital (DO, DI) or group (GO, GI), that allow one or several I/O signals to automatically affect the state of other I/O signals.

Usage

Using cross connections is a simple way to interconnect I/O signals and let the robot system handle I/O activity without having to execute any RAPID code.

Cross connecting I/O signals is a good alternative if there is an input I/O signal in the process that, when activated, automatically activates one or several output I/O signals.

It is also possible to construct more complex conditions by combining up to five different actor I/O signals with operators. The actor I/O signals can also be inverted.

Limitations

The maximum number of cross connections handled by the robot system is 300.

Cross connections must not form a chain that is deeper than 20 levels. A chain is formed when cross connections are interlinked so that an I/O signal that is part of a resultant expression in one cross connection is also part of the actor expression of another cross connection, and so on. The depth of such chain is the number of transitions from the first actor I/O signal to the last resultant I/O signal.

Cross connections must not form closed chains since that would cause infinite evaluation and oscillation. A closed chain appears when cross connections are interlinked so that the chain of cross connections forms a circle.

Ambiguous resultant I/O signals are not allowed since the outcome would depend on the order of evaluation (which cannot be controlled). Ambiguous resultant I/O signals occur when the same I/O signal is resultant in several cross connections.

The expressions are evaluated from left to right, that is, the priorities of the logical operator OR and the logical operator AND are the same. For clarity, our advise is to avoid mixing the logical operator OR and the logical operator AND in the same expression.

The resultant I/O signal in a cross connection must not have an overlapping device map with any inverted actor I/O signals defined in the cross connection. Using I/O

Continues on next page

signals with overlapping device map in a cross connection can cause infinity signal setting loops.

The parameters *Default Value* and *Signal Safe Level* do not affect signals that are a resultant in a cross connection. The resultant signal is only affected by the actor signal values in the cross connection.

Related information

For more information about *Logical Cross Connections*, see *Application manual - Controller software IRC5*.

Device Mapping on page 245

Invert Physical Value on page 254

Type Signal Safe Level on page 230

4.5.2 Name

Parent

Name belongs to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Name

Description

Specifies the name of the cross connection.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O System configuration.

Note

Names differing only in upper and lower case are considered to be equal.

4.5.3 Resultant

Parent

Resultant belongs to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Res

Description

The parameter *Resultant* specifies the digital or group I/O signal to which the result of the condition formed by the actor I/O signals will be stored.

Whenever the outcome of the condition formed by the actor I/O signals is altered the *Resultant* I/O signal will take the same value as that outcome.

Usage

Specify the I/O signal that will be effected by the outcome of the condition formed by the actor I/O signals.

Default value

The default value is an empty string.

Allowed values

A string defining a digital I/O signal or group I/O signal that is defined in the robot system.

4.5.4 Actor 1

Parent

Actor 1 belongs to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Act1

Description

The parameter *Actor 1* specifies the first digital or group I/O signal that forms the actor expression of the cross connection.

Whenever the value of the I/O signal referred to by *Actor 1* is altered, the logical condition formed by the cross connection will be evaluated and the value of the I/O signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the first of the digital or group I/O signals that forms the condition that will control the value of the I/O signal referred to by *Resultant*.

With the *Logical Cross Connections*, the *Actor 1* parameter can be part of a more complex statement formed by combining it with other parameters such as *Invert Actor 1*, *Operator 1*, and *Actor 2*.

Default value

The default value is an empty string.

Allowed values

A string defining a digital I/O signal or group I/O signal defined in the robot system.

Related information

Resultant on page 196.

4.5.5 Invert Actor 1, Invert Actor 2, Invert Actor 3, Invert Actor 4, Invert Actor 5

Parent

Invert Actor 1, *Invert Actor 2*, *Invert Actor 3*, *Invert Actor 4*, and *Invert Actor 5* belong to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Act1_invert, Act2_invert, Act3_invert, Act4_invert, Act5_invert

Description

The parameter *Invert Actor 1* specifies whether the inverted value of the I/O signal referred to by parameter *Actor 1* will be used in the evaluation instead of the actual I/O signal value.

The parameter *Invert Actor 2* specifies whether the inverted value of the I/O signal referred to by parameter *Actor 2* will be used in the evaluation instead of the actual I/O signal value.

The parameter *Invert Actor 3* specifies whether the inverted value of the I/O signal referred to by parameter *Actor 3* will be used in the evaluation instead of the actual I/O signal value.

The parameter *Invert Actor 4* specifies whether the inverted value of the I/O signal referred to by parameter *Actor 4* will be used in the evaluation instead of the actual I/O signal value.

The parameter *Invert Actor 5* specifies whether the inverted value of the I/O signal referred to by parameter *Actor 5* will be used in the evaluation instead of the actual I/O signal value.

Usage

The *Invert Actor 1* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 1* should be used.

The *Invert Actor 2* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 2* should be used.

The *Invert Actor 3* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 3* should be used.

The *Invert Actor 4* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 4* should be used.

The *Invert Actor 5* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 5* should be used.

Default value

The default value is No.

Allowed values

Yes

No

Continues on next page

Related information

Actor 1 on page 197.

4.5.6 Operator 1, Operator 2, Operator 3, Operator 4

Parent

Operator 1, *Operator 2*, *Operator 3*, and *Operator 4* belong to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Oper1, Oper2, Oper3, Oper4

Description

The parameter *Operator 1* specifies the logical operation to be performed between the I/O signals referred to by parameter *Actor 1* and *Actor 2*.

The parameter *Operator 2* specifies the logical operation to be performed between the I/O signals referred to by parameter *Actor 2* and *Actor 3*.

The parameter *Operator 3* specifies the logical operation to be performed between the I/O signals referred to by parameter *Actor 3* and *Actor 4*.

The parameter *Operator 4* specifies the logical operation to be performed between the I/O signals referred to by parameter *Actor 4* and *Actor 5*.

Usage

If only one actor I/O signal is used, *Operator 1* is left out.

If no more than two actor I/O signals are used, then *Operator 2* is left out.

If no more than three actor I/O signals are used, then *Operator 3* is left out.

If no more than four actor I/O signals are used, then *Operator 4* is left out.

Prerequisites

By specifying *Operator 1* it is explicitly demanded that the parameter *Actor 2* must also be specified.

By specifying *Operator 2* it is explicitly demanded that the parameter *Actor 3* must also be specified.

By specifying *Operator 3* it is explicitly demanded that the parameter *Actor 4* must also be specified.

By specifying *Operator 4* it is explicitly demanded that the parameter *Actor 5* must also be specified.

Default value

The default value is an empty string.

Allowed values

AND

OR

Related information

Actor 1 on page 197.

Actor 2, Actor 3, Actor 4, Actor 5 on page 201.

4.5.7 Actor 2, Actor 3, Actor 4, Actor 5

Parent

Actor 2, *Actor 3*, *Actor 4*, and *Actor 5* belongs to the type *Cross Connection*, in the topic *I/O System*.

Cfg name

Act2, Act3, Act4, Act5

Description

The parameter *Actor 2* specifies the second digital or group I/O signal that forms the actor expression of the cross connection.

The parameter *Actor 3* specifies the third digital or group I/O signal that forms the actor expression of the cross connection.

The parameter *Actor 4* specifies the fourth digital or group I/O signal that forms the actor expression of the cross connection.

The parameter *Actor 5* specifies the fifth digital or group I/O signal that forms the actor expression of the cross connection.

Whenever the value of the I/O signal referred to by an *Actor* parameter is altered, the logical condition formed by the cross connection will be evaluated and the value of the I/O signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the second of the digital or group I/O signal that forms the condition that will control the value of the I/O signal referred to by *Resultant*. If only one actor signal is used, then *Actor 2*, *Actor 3*, *Actor 4*, and *Actor 5* is left out.

Prerequisites

Actor 2 will be ignored unless the parameter *Operator 1* is specified.

Actor 3 will be ignored unless the parameter *Operator 2* is specified.

Actor 4 will be ignored unless the parameter *Operator 3* is specified.

Actor 5 will be ignored unless the parameter *Operator 4* is specified.

Default value

The default value is an empty string.

Allowed values

A string defining a digital I/O signal or group I/O signal defined in the robot system.

Related information

Resultant on page 196.

Operator 1, Operator 2, Operator 3, Operator 4 on page 200.

4.6 Type Device Trust Level

4.6.1 The Device Trust Level type

Overview

This section describes the type *Device Trust Level*, which belongs to the topic *I/O System*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DEVICE_TRUST_LEVEL

Type description

Device Trust Level defines the behavior for I/O devices at different execution situations in the robot controller.

Usage

Using device trust levels is a simple way to control the behavior of the robot system and event generation for I/O devices.

Limitations

The maximum number of device trust levels handled by the robot system is 10.

Predefined device trust levels

Device Trust Level:	Description:
DefaultTrustLevel	Default for an I/O device. Using this level - • there is no system action performed but an error event is reported, when the I/O device is disconnected. • an information event is reported, when the I/O device is reconnected.
SafetyTrustLevel	Default for a safety I/O device. Using this level - • there is no system action performed and no error event is reported, when the I/O device is disconnected. • there is no event reported, when the I/O device is reconnected.

4.6.2 Name

Parent

Name belongs to the type *Device Trust Level*, in the topic *I/O System*.

Cfg name

Name

Description

Specifies the name of the device trust level.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter Basic elements.

The name must be unique among all named objects in the I/O System configuration.

Note

Names differing only in upper and lower case are considered to be equal.

4.6.3 Deny Deactivate

Parent

Deny Deactivate belongs to the type *Device Trust Level*, in the topic *I/O System*.

Cfg name

DenyDeactivate

Description

Specifies if it is possible to deactivate the I/O device or not.

Default value

Default value is Allow Deactivate.

Allowed values

Deny Deactivate or Allow Deactivate

4.6.4 Action when Disconnected

Parent

Action when Disconnected belongs to the type *Device Trust Level*, in the topic *I/O System*.

Cfg name

ActionWhenLost

Description

Specifies the system action to perform when the communication with an I/O device is lost.

Default value

No action

Allowed values

Value	Description
No action	No action is performed.
Generate "System Fail"	All NORMAL tasks will be stopped. Besides that, the system is set to system failure state (SYS_FAIL). All jogging and program start orders will be rejected. Only a new normal restart resets the system.
Generate "System Halt"	All NORMAL tasks will be stopped. The system is forced to Motors off state. Changing the system to Motors on resets the system.
Generate "System Stop"	All NORMAL tasks will be stopped but can be restarted. Jogging is also possible.

4.6.5 Report when Disconnected

Parent

Report when Disconnected belongs to the type *Device Trust Level*, in the topic *I/O System*.

Cfg name

ReportWhenLost

Description

Specifies the event reporting when the communication with an I/O device is lost.

Default value

Generate error

Allowed values

Value	Description
Generate error	Report of error event.
Generate information (state change)	Report of information event (state change).
Generate warning	Report of warning event
No error reporting	No report of event.

Related information

Operating manual - Trouble shooting IRC5

4.6.6 Report when Reconnected

Parent

Report when Reconnected belongs to the type *Device Trust Level*, in the topic *I/O System*.

Cfg name

ReportWhenReconnected

Description

Specifies the event reporting when the communication with an I/O device is re-established again.

Default value

Generate information (state change)

Allowed values

Value	Description
Generate error	Report of error event.
Generate information (state change)	Report of information event (state change).
Generate warning	Report of warning event
No error reporting	No report of event.

4.7 Type Device Command

4.7.1 The Device Command type

Overview

This section describes the type *Device Command*, which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

<Network>_COMMAND

where <Network> can be:

- DEVICENET
- ETHERNETIP

Type description

The Device commands for an I/O device used on a specific industrial network are defined through an industrial network option. Each industrial network needs to use own configuration type specific for the network. Device commands can be used on the following type of industrial networks:

- DeviceNet
- EtherNet/IP

Usage

The *Device Command* type is used to send device commands to specific I/O devices on the industrial network.

This is done:

- At start.
- When connecting the I/O device after a power fail.
- When activating the I/O device from RobotStudio or the FlexPendant.

Limitations

The *Device Command* has the following limitations:

- Maximum 300 device commands can be defined in the robot system.

Example

Parameter:	Value:
Name	LinkAddr
Device	d350
Download Order	1
Path	6,20 64 24 01 30 01,C6,1
Service	Set Attribute Single
Value	1

4.7.2 Name

Parent

Name belongs to the type *Device Command*, in the topic *I/O System*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the command.

Default value

The default value is an empty string.

Allowed values

A string defining the name with maximum 80 characters.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

Type Device on page 213

4.7.3 Device

Parent

Device belongs to the type *Device Command*, in the topic *I/O System*.

Cfg name

Device

Description

Specifies the name of the I/O device the command is connected to.

Default value

The default value is an empty string.

Allowed values

A string defining the name of the I/O device with maximum 32 characters.

Note

Names that differ only in upper and lower case are considered to be equal.

Related information

Type Device on page 213

4.7.4 Download Order

Parent

Download Order belongs to the type *Device Command*, in the topic *I/O System*.

Cfg name

OrderNr

Description

The parameter *Download Order* specifies the sequence number in which this command shall be downloaded to the I/O device that have several commands assigned to it.

Usage

Use *Download Order* to control the order in which the commands are downloaded (and executed) on an I/O device.

Lower download orders are downloaded **before higher download orders.**

Default value

The default value is 0.

Allowed values

0 - 100.

4.7.5 Value

Parent

Value belongs to the type *Device Command*, in the topic *I/O System*.

Cfg name

Value

Description

The parameter *Value* specifies the value for this command.

Default value

The default value is an empty string.

Allowed values

A string with maximum 200 characters.

4.8 Type Device

4.8.1 The Device type

Overview

This section describes the type *Device*, which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

<Network>_DEVICE

where <Network> can be:

- DEVICENET
- ETHERNETIP
- PROFINET
- PROFIBUS

Type description

A device is a logical software representation of a real I/O device that is connected to an industrial network within the controller. I/O devices allow you to control electronic devices and read sensor data. They are used for controlling I/O signals in the robot system.

For internal slave device options, a predefined device is created at startup. For more information, see *Type Internal Device on page 224*.

Usage

By specifying an I/O device, a logical representation of the real I/O device is created. The I/O device configuration defines the specific parameters that will control the behavior of the I/O device.

The *Device* is used when defining the I/O signals and device commands in the I/O system.

Prerequisites

Defining a new I/O device:

1. Configure the *Industrial Network* and
2. Make sure that the appropriate device trust level is available (either by creating it or using a predefined device trust level).

Limitations

The I/O device has the following limitations:

- Maximum number of user I/O devices in the robot system are 50.
- Maximum number of I/O devices on one industrial network is 20 (except for the *PROFINET Master/Slave* option which allows 50 I/O devices).

Continues on next page

Predefined units

The following I/O units are predefined and located on the Local Industrial Network:

- PANEL
- DRV_1

Depending on installed options, there can be other predefined I/O devices not described in this manual.

Related information

Connected to Industrial Network on page 216.

Type Device Command on page 208.

Type Device Trust Level on page 202.

Type Internal Device on page 224

For more information on safety signals, see *Operating manual - IRC5 with FlexPendant*.

Example

Parameter:	Value:
Name	board10
Connected to Industrial Network	DeviceNet
State at System Restart	Activated
Trust Level	DefaultTrustLevel
Simulated	No
Recovery Time	5000
Identification Label	U137, placed in process cabinet C5
Address	63
Vendor ID	0
Product Code	0
Device Type	
Production Inhibit Time	10
Connection Type	Polled
Poll Rate	1000
Connection Output Size	0
Connection Input Size	0
Quick Connect	Deactivated

4.8.2 Name

Parent

Name belongs to the type *Device*, in the topic *I/O System*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the I/O device.

Usage

The name of the I/O device is used as a reference to the specific I/O device when configuring the I/O signals and device commands.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules, as described in *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

4.8.3 Connected to Industrial Network

Parent

Connected to Industrial Network belongs to the type *Device*, in the topic *I/O System*.

Cfg name

Network

Description

The parameter *Connected to Industrial Network* specifies which industrial network this I/O device is physically connected to.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules, as described in *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

The Industrial Network type on page 188.

4.8.4 Identification Label

Parent

Identification Label belongs to the type *Device*, in the topic *I/O System*.

Cfg name

Label

Description

The parameter *Identification Label* provides a way to label the real I/O device.

Usage

The parameter *Identification Label* is an optional way to provide a label that will help the operator to identify the I/O device physically.

Default value

The default value is an empty string.

Allowed values

A string with maximum 80 characters.

4.8.5 Vendor Name

Parent

Vendor Name belongs to the type *Device*, in the topic *I/O System*.

Cfg name

VendorName

Description

The parameter *Vendor Name* specifies the name of the I/O device vendor.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4.8.6 Product Name

Parent

Product Name belongs to the type *Device*, in the topic *I/O System*.

Cfg name

ProductName

Description

The parameter *Product Name* specifies the product name for this I/O device according to industrial network type standard.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4.8.7 Trust Level

Parent

Trust Level belongs to the type *Device*, in the topic *I/O System*.

Cfg name

TrustLevel

Description

The parameter *Trust Level* defines the behavior for I/O devices at different execution situations in the robot controller.

The *Trust Level* only affects physical devices controlled by an industrial network master in the robot controller. An internal slave device is not controlled by an industrial network master in the robot controller and is therefore not affected by the *Trust Level* setting.

Usage

This parameter is used to specify the I/O device behavior as per the user requirements at different error situations in the robot controller.

Default value

The default value is *DefaultTrustLevel*.

Allowed values

A string corresponding to the name of a defined *Device Trust Level* type. A string with maximum 32 characters. A string following the RAPID rules, as described in *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

Type Device Trust Level on page 202.

4.8.8 State when System Startup

Parent

State when System Startup belongs to the type *Device*, in the topic *I/O System*.

Cfg name

StateWhenStartup

Description

The parameter *State when System Startup* defines which logical state the I/O device shall have after startup of the robot system..

Usage

The parameter *State when System Startup* value defines the logical state that the robot system shall try to set for the I/O device when system startup. The available options are:

- Establish communication (*Activated*)
- Don't establish communication (*Deactivated*)
- Restore the previously stored logical state for the I/O device at system shutdown (*Last State*)

Default value

The default value is Activated

Allowed values

Activated

Deactivated

Last State

4.8.9 Simulated

Parent

Simulated belongs to the type *Device*, in the topic *I/O System*.

Cfg name

Simulated

Description

The parameter *Simulated* specifies that the I/O device should be treated as simulated.

Usage

The parameter *Simulated* defines that the I/O device is simulated on the industrial network it is connected to.

Default value

The default value is No.

Allowed values

Yes

No

4.8.10 Recovery Time

Parent

Recovery Time belongs to the type *Device*, in the topic *I/O System*.

Cfg name

RecoveryTime

Description

The parameter *Recovery Time* defines how often the recovery of a lost I/O device shall be performed on a specific Industrial Network.

The recovery is performed regularly by the robot controller, to regain contact with lost I/O devices (an I/O device in disconnected or error state).

Default value

The default value is 5000 ms.

Allowed values

An integer value defining the time, in ms, between two recoveries for the specific I/O device. The value must be a multiple of 5000 ms. Minimum value is 5000 ms and maximum limit is 2.147484E+09.

Related information

Technical reference manual - RAPID overview.

4.9 Type Internal Device

4.9.1 The Internal Device type

Overview

This section describes the type *Internal Device*, which belongs to the topic *I/O System*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

<Network>_INTERNAL_DEVICE

where <Network> can be:

- DEVICENET
- ETHERNETIP
- PROFINET

<Network>_INTERNAL_ANYBUS_DEVICE

where <Network> can be:

- DEVICENET
- ETHERNETIP
- PROFINET
- PROFIBUS

Type description

For the internal slave device and the anybus industrial network options, a predefined *Internal Device* is created at system startup.

Example

This is an example for a DeviceNet internal slave device. For more information about DeviceNet, refer *Application manual - DeviceNet Master/Slave*.

Parameter:	Value:
Name	DN_Internal_Device
Connected to Industrial Network	DeviceNet
Simulated	No
Vendor Name	ABB Robotics
Product Name	DeviceNet Internal Slave Device
Identification Label	
Connection Type	Polled
Poll Rate	1000
Connection Output Size	8
Connection Input Size	8

Continues on next page

Related information

More information about the internal slave device and the anybus device can be found in the Application manual for the respective industrial network option, for example *Application manual - DeviceNet Master/Slave*.

Type Device on page 213.

4.9.2 Vendor Name

Parent

Vendor Name belongs to the type *Internal Device*, in the topic *I/O System*.

Cfg name

VendorName

Description

The parameter *Vendor Name* specifies the name of the I/O device vendor.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4.9.3 Product Name

Parent

Product Name belongs to the type *Internal Device*, in the topic *I/O System*.

Cfg name

ProductName

Description

The parameter *Product Name* specifies the product name for this I/O device according to industrial network type standard.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4.9.4 Identification Label

Parent

Identification Label belongs to the type *Internal Device*, in the topic *I/O System*.

Cfg name

Label

Description

The parameter *Identification Label* provides a way to label the real I/O device.

Usage

The parameter *Identification Label* is an optional way to provide a label that will help the operator to identify the I/O device physically.

Default value

The default value is an empty string.

Allowed values

A string with maximum 80 characters.

4.9.5 Simulated

Parent

Simulated belongs to the type *Internal Device*, in the topic *I/O System*.

Cfg name

Simulated

Description

The parameter *Simulated* specifies that the I/O device should be treated as simulated.

Usage

The parameter *Simulated* defines that the I/O device is simulated on the industrial network it is connected to.

Default value

The default value is No

Allowed values

Yes

No

4.10 Type Signal Safe Level

4.10.1 The Signal Safe Level type

Overview

This section describes the type *Signal Safe Level*, which belongs to the topic *I/O System*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

EIO_SIGNAL_SAFELEVEL

Type description

The parameter *Signal Safe Level* defines the behavior of logical output, digital, group and analog signals at different execution situations in the robot system.

Usage

Signal Safe Level is used to define the behavior of the logical output signals in different execution situations in the robot system like system startup, when signal becomes accessible, signal is not accessible and system is shutdown. It is user defined and makes the signal behavior more flexible, user friendly at different situation.

Limitations

The maximum number of signal safe levels handled by the robot system is 10.

Predefined signal safe levels

Signal Safe Level:	Description:
DefaultSafeLevel	This is the default signal safe level. Using this signal safe level - • the signal is using its default value, when system startup and when the signal becomes not accessible. • when the signal becomes accessible and the system is shut-down, the signal takes the last written value. This signal safe level cannot be changed.
SafetySafeLevel	This is the safety signal safe level. It is used by safety signals in the robot system. Using this signal safe level - • the signal is using its default value when system startup and when the signal becomes accessible or not accessible. • when the system is shutdown, the signal safe level takes the last written value. This signal safe level cannot be changed.

Continues on next page

Example

This is an example of a signal safe level.

Parameter:	Value:
Name	MySafeLevel
Action when System Startup	Set default value
Action when Signal Accessible	Set last value
Action when Signal Not Accessible	Set default value
Action when System Shutdown	Set last value

Related information

Safe Level on page 251.

Default Value on page 250

Operating manual - IRC5 with FlexPendant

4.10.2 Name

Parent

Name belongs to the type *Signal Safe Level*, in the topic *I/O System*.

Cfg name

Name

Description

Specifies the name of the signal safe level.

Usage

The name of the signal safe level is used as a reference to the specific signal behavior when configuring the logical output signals.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter*Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

4.10.3 Action When Startup

Parent

Action When Startup belongs to the type *Signal Safe Level*, in the topic *I/O System*.

Cfg name

ActionWhenStartup

Description

Specifies the value for a logical output signal after startup of the robot system.

Default value

Set default value

Allowed values

Set default value

Set last value

Set zero value

Additional information

For logical ouput signals mapped against the same bits in the I/O memory map, there are certain limitations. For example, logical output signals of type Digital Output mapped on Group Output. To prevent unpredictable signal values for these logical output signals at system startup, the conditions are:

- The logical output signals must have the same value for the parameter *Action When Startup*.
- If the parameter *Action When Startup* use the value Set default value, the defined default value must match for each logical output signal on a bitwise level.

Related information

Default Value on page 250.

Device Mapping on page 245.

4.10.4 Action when Signal Accessible

Parent

Action when Signal Accessible belongs to the type *Signal safe Level*, in the topic *I/O System*.

Cfg name

ActionWhenAccessible

Description

Specifies the value for a logical output signal when its physical state becomes accessible.

Default value

Set last value

Allowed values

Set default value

Set last value

Set zero value

Related information

Default Value on page 250

4.10.5 Action when Signal Not Accessible

Parent

Action when Signal Not Accessible belongs to the type *Signal Safe Level*, in the topic *I/O System*.

Cfg name

ActionWhenNotAccessible

Description

Specifies the value for a logical output signal when its physical state becomes not accessible.

Default value

Set default value

Allowed values

Set default value

Set last value

Set zero value

Related information

Default Value on page 250

4.10.6 Action when System Shutdown

Parent

Action when System Shutdown belongs to the type *Signal SafeLevel*, in the topic *I/O System*.

Cfg name

ActionWhenShutdown

Description

Specifies the value for a logical output signal when the robot system is shutdown.

Default value

Set last value

Allowed values

Set default value

Set last value

Set zero value

Related information

Default Value on page 250

4.11 Type Signal

4.11.1 The Signal type

Overview

This section describes the type *Signal*, which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_SIGNAL

Type description

An I/O signal is the logical software representation of:

- Inputs or outputs located on an I/O device that is connected to a Industrial Network within the robot system (real I/O signal).
- An I/O signal without a representation on any I/O device (simulated I/O signal).

Usage

By specifying an I/O signal, a logical representation of the real or simulated I/O signal is created. The I/O signal configuration defines the specific system parameters for the I/O signal that will control the behavior of the I/O signal.

Many of the parameters depend on the type of the I/O signal, therefore it is recommended that the parameter *Type of Signal* is assigned first.

Prerequisites

Before defining a new I/O signal, make sure that the appropriate *Signal Safe Level* and *Access Level* are available (either by creating them or using a predefined *Signal Safe Level* and *Access Level* respectively).

Limitations

A maximum of 12000 user I/O signals can be defined in the robot system.

Predefined signals

There are a number of predefined I/O signals in the robot controller. Depending on installed options there can also be other predefined I/O signals.

Example digital input

The following is a typical example of a digital input I/O signal (DI).

Parameter	Value
Name	ObjectAtPlace
Type of Signal	Digital Input
Assigned to device	board10
Signal Identification Label	X4:4

Continues on next page

Parameter	Value
Device Mapping	11
Category	
Access Level	Default
Default Value	0
Filter Time Passive	0
Filter Time Active	0
Invert Physical Value	No
Safe Level	DefaultSafeLevel

Example analog output

The following is a typical example of an analog output I/O signal (AO).

Parameter	Value
Name	Speed
Type of Signal	Analog Output
Assigned to Device	board10
Signal Identification Label	X6:4
Device Mapping	16-31
Category	
Access Level	Default
Default Value	0
Analog Encoding Type	Two complement
Maximum Logical Value	21474.8
Maximum Physical Value	10
Maximum Physical Value Limit	10
Maximum Bit Value	32767
Minimum Logical Value	-21474.8
Minimum Physical Value	-10
Minimum Physical Value Limit	-10
Minimum Bit Value	-32767
Safe Level	DefaultSafeLevel

Example group input

The following is a typical example of a group input I/O signal (GI).

Parameter	Value
Name	StatusGroup
Type of Signal	Group Input
Assigned to Device	board10
Signal Identification Label	X2:1-X2:8

Continues on next page

Parameter	Value
Device Mapping	0-7
Category	
Access Level	Default
Default Value	0
Filter Time Passive	0
Filter Time Active	0
Invert Physical Value	No
Safe Level	DefaultSafeLevel

Example simulated digital input

The following is a typical example of a simulated digital input I/O signal (DI).

Parameter	Value
Name	StatusDigital
Type of Signal	Digital Input
Assigned to Device	
Signal Identification Label	
Device Mapping	
Category	
Access Level	Default
Default Value	0
Filter Time Passive	0
Filter Time Active	0
Invert Physical Value	No
Safe Level	DefaultSafeLevel

Example simulated analog output

The following is a typical example of an simulated analog output I/O signal (AO).

Parameter	Value
Name	StatusAnalog
Type of Signal	Analog Output
Assigned to Device	
Signal Identification Label	
Category	
Access Level	Default
Default Value	0
Analog Encoding Type	Twos complement
Maximum Logical Value	10
Maximum Physical Value	10

Continues on next page

Parameter	Value
Maximum Physical Value Limit	10
Maximum Bit Value	0
Minimum Logical Value	-10
Minimum Physical Value	-10
Minimum Physical Value Limit	-10
Minimum Bit Value	0
Safe Level	DefaultSafeLevel

Example simulated group input

The following is a typical example of a simulated group input I/O signal (GI).

Parameter	Value
Name	StatusGroup
Type of Signal	Group Input
Assigned to Device	
Signal Identification Label	
Device Mapping	
Category	
Access Level	Default
Default Value	0
Filter Time Passive	0
Filter Time Active	0
Invert Physical Value	No
Safe Level	DefaultSafeLevel

Related information

The Device type on page 213.

The Access Level type on page 180.

Type Signal Safe Level on page 230

Operating manual - IRC5 with FlexPendant.

4.11.2 Name

Parent

Name belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the logical I/O signal.

Usage

The name of the I/O signal is used as a reference to the specific I/O signal when:

- Accessing the I/O signal (that is reading or writing its value) in RAPID.
- Configuring cross connections.
- Configuring system inputs and system outputs.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

The Cross Connection type on page 193.

The System Input type on page 266.

The System Output type on page 312.

4.11.3 Type of Signal

Parent

Type of Signal belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

SignalType

Description

Type of Signal specifies the signal's representation, behavior, and direction.

Usage

Each I/O signal must be classified as one of the predefined types. The type of I/O signal will determine the behavior of the I/O signal as well as how it will be represented and interpreted.

As the behavior of the I/O signal depends upon its type, the settings of other parameters will vary, therefore it is recommended that the *Type of Signal* parameter is assigned before any other parameter for the I/O signal.

Default value

The default value is an empty string.

Allowed values

Digital Input

Digital Output

Analog Input

Analog Output

Group Input

Group Output

4.11.4 Assigned to Device

Parent

Assigned to Device belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Device

Description

The parameter *Assigned to Device* specifies which I/O device the I/O signal is connected to (if any).

Limitations

An I/O signal that is not mapped against an I/O device (that is *Assigned to Device* is not defined) will be considered as simulated.

Default value

The default value is an empty string.

Allowed values

A string, either:

- Empty (unspecified), that is a simulated I/O signal, or
- Defining the name of a defined I/O device.

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

The Device type on page 213.

4.11.5 Signal Identification Label

Parent

Signal Identification Label belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

SignalLabel

Description

The parameter *Signal Identification Label* provides a free-text label to an I/O signal.

Usage

Signal Identification Label is optional for use in providing a label for the physical contact or cable that this I/O signal configuration represents.

Assign an easy-to-understand name (free text) to the I/O signal to make it easy to physically identify. For example, map the I/O signal to a physical identification such as a cable marking or an outlet label.

Default value

The default value is an empty string.

Allowed values

A string of maximum 80 characters.

Example

Conn. X4, Pin 1

4.11.6 Device Mapping

Parent

Device Mapping belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

DeviceMap

Description

The parameter *Device Mapping* specifies which bit(s) in the I/O memory map of the assigned I/O device, the I/O signal is mapped to.

Usage

All I/O signals except simulated I/O signals must be mapped.

Limitations

An I/O signal must be completely mapped to bits on the same I/O device. For example, it is not possible to map a group signal to bits on different I/O devices.

Default value

The default value is an empty string.

Allowed values

A string with maximum 80 characters.

The string should contain the mapping order of the individual bits of the I/O signal, using the following syntax:

- Refer to a bit in the I/O memory map by the index of the bit, the bits are indexed from 0 (zero) and upwards.
- If the I/O signal is mapped to several continuous bits, these can be given as a range: <first bit in range> - <last bit in range>
- If the I/O signal is mapped to several discontinuous bits and/or ranges, these should be separated by commas: <bit/range>, <bit/range>, <bit/range>

Additional information

Overlapping of device maps is not recommended. That is, the *Device Mapping* must not refer to the same bit more than once. A lot of unwanted scenarios can appear when different logical signals are mapping on the same physical bit.

One example is if two overlapping group signals are used in one cross connection where one is actor and inverted and one is resultant. This scenario will cause an endless loop.

Restrictions for overlapping signals is necessary because of the importance to have predictability in the system.

Allowed with restrictions

The following rules are present for overlapping signals of type:

- Group Output/Digital Output

Continues on next page

- Group Input/Digital Input
- Group Output/Group Output
- Group Input/Group Input

The overlapping signals are allowed with the following restrictions:

- Overlapping signals must have the same parameter value for *Signal Safe Level ActionWhenStartup*.
- The *Signal Safe Level* parameter *ActionWhenStartup* (*Default*) must be consistent on the overlapping bit(s) level.
- It is not allowed to have two overlapping signals where one signal is actor and one signal is resultant in a cross connection.

Allowed with event log warning

The following rules are present for overlapping signals but with an event log warning.

- Group Output/Analog Output
- Group Input/Analog Input
- Digital Output/Analog Output
- Digital Input/Analog Input
- Analog Output/Analog Output
- Analog Input/Analog Input

Note

Overlapping of analog signals with digital or group signals is not recommended due to the complexity in comparing a scalable value with a bit value.

Not allowed

The following overlapping signals are not allowed:

- Digital Input/Digital Input
- Digital Output/Digital Output

Allowed size of the signal

The size of the I/O signal (that is, the number of bits in *Device Mapping*) is restricted. The restriction depends on the type of I/O signal. Following are the restrictions:

- Digital signals must be mapped to exactly one bit.
- Analog signals must be mapped between 2 and 32 bits[I].
- Group signals must be mapped between 1 and 32 bits[II].

I A simulated analog I/O signal is by default mapped to 23 bits but the number of bits can be defined by the I/O signal configuration parameter *Number Of Bits*.

II A simulated group I/O signal is by default mapped to 23 bits but the number of bits can be defined by the I/O signal configuration parameter *Number Of Bits*.

Example

Examples of valid mapping of a digital signal (1 bit):

- 0
- 13

Continues on next page

Examples of valid mapping of an analog or group signal (2-32 bits):

- 4, 6-7
- 16-31
- 8-15, 0-7

Example of *invalid* mapping (bit 7 is overlapped):

- 0-7, 15-7

Related information

Number Of Bits on page 265

4.11.7 Category

Parent

Category belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Category

Description

The parameter *Category* provides a free-text categorization to an I/O signal.

Usage

Category is optional to use for categorizing the I/O signals so that tools (for example software tools) can filter and sort signals based on these categories.

Limitations

I/O signals defined as Safety or Internal are hidden for the user in RobotStudio and FlexPendant.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Additional information

The category of all safety-related I/O signals (internally loaded by the system) are set to Safety.

4.11.8 Access Level

Parent

Access Level belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Access

Description

The parameter *Access Level* specifies which clients have write access to the I/O signal.

Usage

Access Level defines the write access of the I/O signal for different categories of I/O controlling applications, such as RobotStudio and RAPID programs.

Default value

The default value is Default.

Allowed values

A string corresponding to the name of a defined *Access Level* type.

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

The Access Level type on page 180.

4.11.9 Default Value

Parent

The parameter *Default Value* belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Default

Description

The parameter *Default Value* specifies the I/O signal default value.

Usage

The default value:

- is used for initializing the I/O signal at different execution situation in the robot system, see *Type Signal Safe Level on page 230*.
- is used for the evaluation of cross connections whenever the I/O signal is not accessible, that is for example when the I/O device to which the I/O signal is assigned is disconnected.

Allowed values

Depending on the type of I/O signal, the following values are allowed:

Type of I/O signal	Allowed value
Digital	0 or 1
Analog	Any value in the range *Minimum Logical Value* to *Maximum Logical Value*.
Group	Any value in the range 0 to 2^{size}-1 (size = number of bits in the *Device Mapping* parameter or in the *Number Of Bits parameter* for simulated group signals).

Default value

The default value is 0.

Additional information

For I/O signals mapped against the same bits in the I/O memory map, there are certain limitations. For more information, refer to Additional information in *Device Mapping on page 245*.

Related information

Minimum Logical Value on page 261.

Maximum Logical Value on page 256.

Device Mapping on page 245.

Type Signal Safe Level on page 230

4.11.10 Safe Level

Parent

Safe Level belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

SafeLevel

Description

Safe Level specifies the behavior of logical output I/O signals at different execution situations in the robot system.

Usage

This parameter is used to specify the logical output signal behavior as per the user requirements at different execution situation like system startup, when signal becomes accessible, signal is not accessible and system shutdown.

Default value

The default value is DefaultSafeLevel.

Allowed values

A string with maximum 32 characters. A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration.

Note

Names differing only in upper and lower case are considered to be equal.

Related information

Type Signal Safe Level on page 230

4.11.11 Filter Time Passive

Parent

Filter Time Passive belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

FiltPas

Description

The parameter *Filter Time Passive* specifies the filter time for detection of negative flanks (that is I/O signal physical value goes from active to passive).

Usage

The passive filter time filters I/O signals from noise that could otherwise be interpreted as a pulse of the I/O signal.

The passive filter time specifies the period in ms (milliseconds) that the physical value of the I/O signal must remain passive before the I/O signal will be considered passive and the logical I/O signal is changed to passive, that is if the time period that the physical value is passive is shorter than *Filter Time Passive*, the logical signal is not changed.

Prerequisites

This parameter is applicable on digital input and group input I/O signals only, that is *Type of Signal* must be set to one of these types or this parameter will be ignored.

Default value

The default value is 0.

Allowed values

Value:	Description:
0	No filter
10-32000	Filter time in ms

Additional information

Note that many I/O devices have built-in hardware for filtering I/O signals. This filter time is then added to the value of *Filter Time Passive*.

Related information

Type of Signal on page 242.

Filter Time Active on page 253.

4.11.12 Filter Time Active

Parent

Filter Time Active belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

FiltAct

Description

The parameter *Filter Time Active* specifies the filter time for detection of positive flanks (that is I/O signal physical value goes from passive to active).

Usage

The active filter time filters I/O signals from noise that could otherwise be interpreted as a pulse of the I/O signal.

The active filter time specifies the period in ms (milliseconds) that the physical value of the I/O signal must remain active before the I/O signal will be considered active and the logical I/O signal is changed to active, that is if the time period that the physical value is active is shorter than *Filter Time Active*, the logical I/O signal is not changed.

Prerequisites

This parameter is applicable on digital input and group input I/O signals only, that is *Type of Signal* must be set to one of these types or this parameter will be ignored.

Default value

The default value is 0.

Allowed values

Value:	Description:
0	No filter
10 - 32000	Filter time in ms

Additional information

Note that many devices have built-in hardware for filtering I/O signals. This filter time is then added to the value of *Filter Time Active*.

Related information

Type of Signal on page 242.

Filter Time Passive on page 252.

4.11.13 Invert Physical Value

Parent

Invert Physical Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Invert

Description

The parameter *Invert Physical Value* specifies whether the physical representation should be the inverted of the logical representation.

Usage

Use this parameter to apply an inversion between the physical value of the I/O signal and its logical representation in the system.

How to invert the I/O signal depends on the direction of the I/O signal (see *Type of Signal*):

- The logical value of an **input** I/O signal will be the inversion of its physical value
- The physical value of an **output** I/O signal will be the inversion of its logical value.

Inverting a group I/O signal will make each individual bit in the group inverted.

Prerequisites

This parameter is only applicable on digital or group I/O signals, that is *Type of Signal* must be set to one of these types or this parameter will be ignored.

Default value

The default value is No.

Allowed values

Yes

No

Related information

Type of Signal on page 242.

4.11.14 Analog Encoding Type

Parent

Analog Encoding Type belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

EncType

Description

The parameter *Analog Encoding Type* specifies how the value of an analog I/O signal is interpreted.

Usage

Use this parameter to specify if the physical representation of an analog I/O signal should be interpreted as a signed (twos complement) or unsigned value.

Prerequisites

This parameter is only applicable on analog I/O signals, that is *Type of Signal* must be set to an analog signal type or this parameter will be ignored.

Default value

The default value is Two complement.

Allowed values

Value:	Description:
Two complement	If the physical analog range for a specific I/O signal is symmetric around 0, for example -32768 to +32767, the I/O signal is most likely coded as Two complement.
Unsigned	Unsigned is used for I/O signals ranging from 0 and upwards.

Related information

Type of Signal on page 242.

4.11.15 Maximum Logical Value

Parent

Maximum Logical Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MaxLog

Description

The parameter *Maximum Logical Value* specifies the logical value that will correspond to the *Maximum Physical Value*.

Usage

The logical values offer a way to access the I/O signals (for example through RAPID programs) by using logical quantities rather than physical.

By setting up the extremes (minimum and maximum values) of the logical and physical values the system will be able to calculate scale and offset factors for transforming I/O signal values between the different quantities.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog signal type or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Logical Value*.

Default value

The default value is 0.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Logical Value* and *Maximum Logical Value* are set to zero (0), the logical values will be directly mapped against the physical values:

- minimum logical value = minimum physical value
- maximum logical value = maximum physical value

Hence there is no scaling or offset factor between the logical and physical representation of the value of an I/O signal.

Additional information

The logical value is a representation of a signal that makes it possible to handle the signal in quantities known from the real world feature it corresponds to rather than the physical value used to control it. For example it would be more natural to set the speed of a moving axis in mm/s (the logical value) rather than the amount of voltage needed to attain that speed (the physical value).

Continues on next page

Related information

Minimum Logical Value on page 261.

Maximum Physical Value on page 258.

Minimum Physical Value on page 262.

Type of Signal on page 242.

4.11.16 Maximum Physical Value

Parent

Maximum Physical Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MaxPhys

Description

The parameter *Maximum Physical Value* specifies the physical value that will correspond to the *Maximum Bit Value*.

Usage

The physical value directly corresponds to the value of the I/O signal that this system parameter corresponds to, for example the amount of voltage given by a sensor or the current feed into a manipulator.

By setting up the extremes (minimum and maximum values) of the bit and physical values the system will be able to calculate scale and offset factors for transforming signal values between the bit and physical quantities.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Physical Value*.

Default value

The default value is 0.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Physical Value* and *Maximum Physical Value* are set to zero (0), the physical values will be directly mapped against the bit values:

- minimum physical value = minimum bit value
- maximum physical value = maximum bit value

Hence there is no scaling or offset factor between the physical and bit representation of the value of an I/O signal.

Related information

Minimum Physical Value on page 262.

Maximum Logical Value on page 256.

Maximum Bit Value on page 260.

Minimum Bit Value on page 264.

Type of Signal on page 242.

4.11.17 Maximum Physical Value Limit

Parent

Maximum Physical Value Limit belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MaxPhysLimit

Description

The parameter *Maximum Physical Value Limit* specifies the maximum allowed physical value, acting as a working range limiter.

Usage

The *Maximum Physical Value Limit* limits the allowed maximum physical value, for example if a bit or logical value is given that would exceed this limit, the physical value is automatically adjusted to *Maximum Physical Value Limit*.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog signal type or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Physical Value Limit*.

Default value

The default value is 0.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Physical Value Limit* and *Maximum Physical Value Limit* are set to zero (0), the physical value limits will be directly mapped against the physical values:

- minimum physical value limit = minimum physical value
- maximum physical value limit = maximum physical value

Related information

Minimum Physical Value on page 262.

Maximum Physical Value on page 258.

Type of Signal on page 242.

4.11.18 Maximum Bit Value

Parent

Maximum Bit Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MaxBitVal

Description

The parameter *Maximum Bit Value* specifies the bit value that will correspond to the *Maximum Logical Value*.

Usage

The bit value is the I/O signal's representation when transmitted on the network.

The bit value is used when calculating the physical and logical values.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog signal type or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Bit Value*.

Default value

The default value is 0.

Allowed values

-2,147,483,648 to 2,147,483,647

If both *Minimum Bit Value* and *Maximum Bit Value* are set to zero (0) then the bit values will be calculated based on the selected *Analog Encoding Type*.

If *Analog Encoding Type* is set to Twos complement:

- maximum bit value = $2^{(\text{no of bits in Device Mapping})-1}-1$
- minimum bit value = $2^{(\text{no of bits in Device Mapping})-1}$

If *Analog Encoding Type* is set to Unsigned:

- maximum bit value = $2^{(\text{no of bits in Device Mapping})}-1$
- minimum bit value = 0

Related information

Minimum Bit Value on page 264.

Maximum Logical Value on page 256.

Maximum Physical Value on page 258.

Analog Encoding Type on page 255.

Type of Signal on page 242.

4.11.19 Minimum Logical Value

Parent

Minimum Logical Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MinLog

Description

The parameter *Minimum Logical Value* specifies the logical value that will correspond to the *Minimum Physical Value*.

Usage

See *Maximum Logical Value*.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog I/O signal type or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Logical Value*.

Default value

The default value is 0.

Allowed values

See *Maximum Logical Value*.

Related information

Maximum Logical Value on page 256.
Minimum Physical Value on page 262.
Type of Signal on page 242.

4.11.20 Minimum Physical Value

Parent

Minimum Physical Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MinPhys

Description

The parameter *Minimum Physical Value* specifies the physical value that will correspond to the *Minimum Logical Value*.

Usage

See *Maximum Physical Value*.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to one of the analog I/O signal types or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Physical Value*.

Default value

The default value is 0.

Allowed values

See *Maximum Physical Value*.

Related information

Maximum Physical Value on page 258.

Type of Signal on page 242.

4.11.21 Minimum Physical Value Limit

Parent

Minimum Physical Value Limit belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MinPhysLimit

Description

The parameter *Minimum Physical Value Limit* specifies the minimum allowed physical value, hence it acts as a working range limiter.

Usage

See *Maximum Physical Value Limit*.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog I/O signal type or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Physical Value Limit*.

Default value

The default value is 0.

Allowed values

See *Maximum Physical Value Limit*.

Related information

Maximum Physical Value Limit on page 259.

Type of Signal on page 242.

4.11.22 Minimum Bit Value

Parent

Minimum Bit Value belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

MinBitVal

Description

The parameter *Minimum Bit Value* specifies the bit value that will correspond to the *Minimum Logical Value*.

Usage

See *Maximum Bit Value*.

Prerequisites

This parameter is only applicable to analog I/O signals, that is *Type of Signal* must be set to an analog I/O signal type or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Bit Value*.

Default value

The default value is 0.

Allowed values

See *Maximum Bit Value*.

Related information

Maximum Bit Value on page 260.

Type of Signal on page 242.

4.11.23 Number Of Bits

Parent

Number Of Bits belongs to the type *Signal*, in the topic *I/O System*.

Cfg name

Size

Description

The parameter *Number Of Bits* specifies the number of bits used for simulated group I/O signals.

Usage

Can be used to specify the number of bits to be used for simulated group I/O signals.

Prerequisites

This parameter is only applicable to group I/O signals not assigned to any I/O device, simulated I/O signals.

Default value

The default value is 23.

Allowed values

1 to 32.

Related information

Device Mapping on page 245.

4.12 Type System Input

4.12.1 The System Input type

Overview

This section describes the type *System Input* which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SYSSIG_IN

Type description

Input I/O signals can be assigned specific system inputs, for example Start or Motors on. The input triggers a system action that is handled by the system, without using the FlexPendant or other hardware devices.

It is possible to use a PLC to trigger the system inputs.

Rejected system inputs

If the system is in manual mode or cannot perform the action due to any other unfulfilled requirement, no error messages are displayed. When a system action is rejected the error messages are stored in the error log.

Limitations

The following limitations have to be considered:

- Only one system action can be assigned to the input I/O signal. However, several input I/O signals can be assigned the same system action.
- When deleting a system action the input I/O signal itself remains defined. The I/O signal has to be deleted separately.
- System input I/O signals are only valid for the Main task, with exceptions on the action value level. These are described together with the corresponding action value.

Additional information

Most system inputs are 0 to 1 level sensitive. The pulse length has to exceed 50 ms or according to the configured filter settings for I/O signals. The *System Input* signal *SimMode* is both 0 to 1 and 1 to 0 level sensitive.

Related information

How to define system inputs on page 179.

Filter Time Passive on page 252.

Filter Time Active on page 253.

4.12.2 Signal Name

Parent

Signal Name belongs to the type *System Inputs* in the topic *I/O System*.

Cfg name

Signal

Description

Signal Name is the name of the configured digital input I/O signal to use. It connects the system input with a configured digital input I/O signal.

Prerequisite

A digital input I/O signal with a defined name has to be configured in the system.

Allowed values

Available configured digital input I/O signal names.

Related information

The Signal type on page 237.

How to define system inputs on page 179.

4.12.3 Action

Parent

Action belongs to the type *System Inputs*, in the topic *I/O System*.

Cfg name

Action

Description

Input signals can be assigned to specific system status. *Status* defines the system action to be triggered by the signal. The system status is handled by the system without an input from the user.

Allowed values

The following values are allowed and described on the following pages:

- *Backup on page 269*.
- *Disable Backup on page 271*.
- *Interrupt on page 272*.
- *Limit Speed on page 274*.
- *Load on page 276*.
- *Load and Start on page 278*.
- *Motors Off on page 280*.
- *Motors On on page 281*.
- *Motors On and Start on page 283*.
- *Reset Emergency Stop on page 285*.
- *Reset Execution Error Signal on page 287*.
- *Start on page 288*.
- *Start at Main on page 290*.
- *Stop on page 292*.
- *Quick Stop on page 293*.
- *Soft Stop on page 294*.
- *Stop at End of Cycle on page 295*.
- *Stop at End of Instruction on page 296*.
- *System Restart on page 297*.
- *SimMode on page 298*.
- *Enable Energy Saving on page 300*.
- *Write Access on page 302*.

Related information

How to define system inputs on page 179.

4.12.4 Action values

4.12.4.1 Backup

Parent

Backup is an action value for the parameter *Status* that belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Backup

Description

The action value *Backup* starts a backup and saves the backup according to the parameter arguments.

Arguments

When the parameter *Action* is set to *Backup*, the arguments *Argument 1*, *Argument 3*, *Argument 4*, and *Argument 5* must also be used.

Parameter	Allowed value
Argument 1	Specify a name for the backup. If the string "SYSTEM:" is specified, the name is set to be the system name.
Argument 3	Specify a path for the backup. Always define the entire path, for example, BACKUP/sysinBackup.
Argument 4	UniqueName means that the backup gets a unique name. If the name already exists, a higher number is added at the end of the name. Overwrite means that a backup with the same name is overwritten.
Argument 5	AddDate means that the backup gets the date in the name automatically. The date is in YYYYMMDD format and is put at the end of the name but before any sequence number.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The parameter *Argument 1* has to be defined with the backup name or the environment variable "SYSTEM:".
- The parameter *Argument 3* has to be defined with the backup path.
- The parameter *Argument 4* has to indicate if the backup shall have a unique name or if an existing backup shall first be deleted.
- The parameter *Argument 5* has to indicate if the backup shall have the date in the name.

Limitations

The backup order is ignored with a warning if a backup is already in progress.

Continues on next page

Additional information

The system output *Backup Error* tells if the backup was successful or not.

The system output *Backup in progress* tells if the backup process is active or not.

The ordered Backup will take the program control during the operation.

Related information

Action on page 268.

Argument 1 on page 303.

Argument 3 on page 306.

Argument 4 on page 307.

Argument 5 on page 308

Backup Error on page 318.

Backup in progress on page 319.

4.12.4.2 Disable Backup

Parent

Disable Backup is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

DisableBackup

Description

The action value *Disable Backup* will prevent starting a backup as long as the signal is set.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Limitations

If a backup is prevented, it will not be started when the signal gets low.

If a backup is ongoing when the signal is set, the backup will continue until it has finished.

4.12.4.3 Interrupt

Parent

Interrupt is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

Interrupt

Description

The action value *Interrupt* executes a routine and after running the routine the execution will resume to the same instruction as before. If neccessary, a regain movement is always performed before the interrupt routine executes.

Interrupt can be used by a PLC to let the robot go to a service position.

Arguments

When the parameter *Action* is set to *Interrupt*, the parameters *Argument1* and *Argument2* must also be used.

Parameter:	Allowed value:
Argument1	The name of the routine to be executed.
Argument2	The task in which the routine defined in *Argument1* should be executed.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The program execution has to be stopped.
- The parameter *Argument 1* has to be defined with the name of the routine to be executed, for example routine 1.
- If the option *MultiMove* is installed, the parameter *Argument 2* has to be defined with a task in which the routine should execute.

Limitations

The parameter has the following limitations:

- The system has to be in automatic mode.
- You cannot use this action value if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- The *Interrupt* action is not valid during program execution.

Continues on next page

Additional information

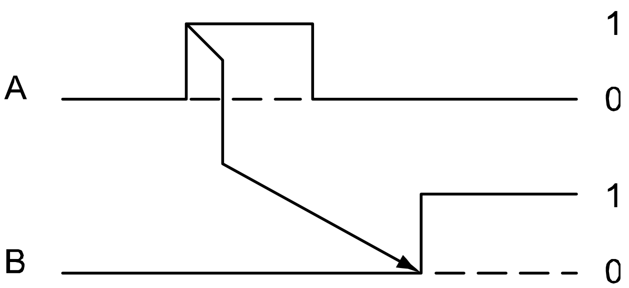
When the execution is stopped, the robot still remembers the point to which it is supposed to go. To prevent the robot going to this position when the Interrupt routine starts and delay it until after the Interrupt, the following RAPID sequence can be used in the Interrupt routine:

```
PROC A()
  StopMove\Quick; !Prevent current move instruction to continue
  StorePath; !For later use
  currpos:=CRobT(); !Save current position
  -----
  ----- ! Place the code for the routine to run here.
  -----
  MoveJ currpos,v600,fine,toolx; !Move back to programmed position
  RestoPath; !Restore StorePath
  StartMove; !Restore StopMove
ENDPROC
```

After the `StartMove` instruction, the stopped movement will continue to move to its fine point. When the routine A has been executed, the normal program can be restarted.

Signal sequence

The signal sequence for *Interrupt* is:



xx0400000949

A: Interrupt (IN)

B: Cycle On (OUT)

Related information

Action on page 268.

Argument 1 on page 303.

Argument 2 on page 305.

4.12.4.4 Limit Speed

Parent

Limit Speed is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

LimitSpeed

Description

The action value *LimitSpeed* shall be set when the speed of one or all motion task is to be reduced. The reduction of the speed is considered to be completed when the *System Output Signal LimitSpeed* is set to 1.

The speed limitation is set up with RAPID instructions *SpeedLimAxis* and *SpeedLimCheckPoint* (see RAPID reference manual for further details) or the manual mode default values will be used.

Arguments

When the parameter *Action* is set to *LimitSpeed*, the parameter *Argument 6* must be used to specify a motion task.

Note

The drop-down list in the FlexPendant or RobotStudio configuration tool shows only TCP robots. Use ABC... to add any other mechanical unit.

WARNING

Connecting more than one signal to the system input signal *LimitSpeed* (connected to same robot) can cause unpredictable behavior during power failure restart.

Prerequisites

A digital input I/O signal with a defined signal name has to be available, not used by any other resource.

Program execution

When the system input signal `LimitSpeed` is set to 1, the speed is ramped down to the reduced speed.

When the system input signal `LimitSpeed` is set to 0, the speed is ramped up to the programmed speed used in the current movement instruction.

The system output signal `LimitSpeed` is set to 1, when the reduced speed is reached.

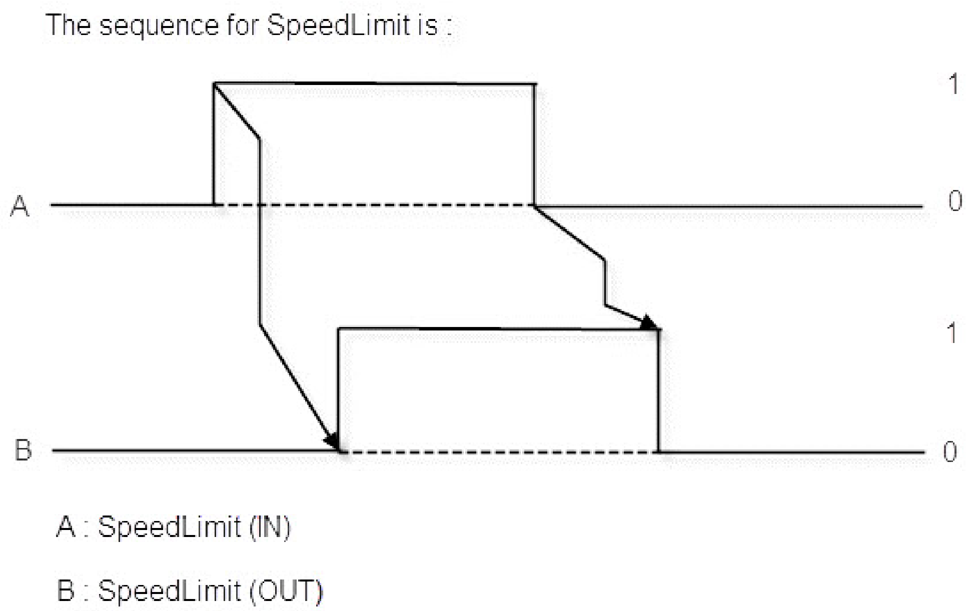
The system output signal `LimitSpeed` is set to 0, when the speed starts to ramp up.

Continues on next page

The default values for speed limitation are automatically set

- when using the restart mode **Reset RAPID**.
- when a new program is loaded.
- when starting program execution from the beginning.

Signal sequence



en1200000680

Additional information

A system output signal (called EnergySavingBlocked) can be configured to reflect if Energy Saving is blocked or not. It is not only the System Input Signal *EnableEnergySaving* that can cause the Energy Saving functionality to be blocked. That is, the System Output Signal *EnergySavingBlocked* can be set even if the System Input action *EnableEnergySaving* is set.

4.12.4.5 Load

Parent

Load is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

Load

Description

The action value *Load* loads a RAPID program (files of type .mod, .prg, and .pgf) from a mass storage device. The program starts from the beginning.

Note! The previously loaded files (of type .prg or .pgf) will be unloaded.

Load can be used by a PLC to load a program, instead of using the FlexPendant.

The program pointer is set to the main entry routine after the module has been loaded. Program pointers in other tasks are not affected.

Arguments

When the parameter *Action* is set to *Load*, the parameters *Argument1* and *Argument2* must also be used.

Parameter:	Allowed value:
Argument 1	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, e.g. HOME:ModuleA.mod
Argument 2	The task in which the program defined in *Argument 1* should be loaded.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The program control has to be available, that is not used by any other resource.
- The parameter *Argument 1* has to be defined with the program file name.
- If the option *MultiMove* is installed, the parameter *Argument 2* must be defined with a task for which the program or module should be loaded.

Limitations

This action value has the following limitations:

- The controller has to be in automatic mode.
- *Load* is not valid during program execution.
- If the current program has been changed, the changes will not be saved before the load.

Continues on next page

Additional information

If the System Input should be used to load modules in many tasks, it is necessary to use a mechanism so that all modules are not loaded at once. The reset routine needs to be used, see section *The Event Routine type on page 108*. This routine is called once the module is loaded. This routine can then trigger the call for loading the next module by setting an I/O signal (for example `SetDO \SDelay := 0.2, do_module_loaded, 1;`). By doing this a chain of calls is made to load all wanted modules.

Related information

Action on page 268.

Argument 1 on page 303.

Argument 2 on page 305.

4.12.4.6 Load and Start

Parent

Load and Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

LoadStart

Description

The action value *Load and Start* loads a RAPID program (files of type .mod, .prg, and .pgf) from a mass storage device. The program starts from the beginning.

Note

The previously loaded files (of type .prg or .pgf) will be unloaded.

Load and Start can be used by a PLC to load and start a program, instead of using the FlexPendant.

The Program Pointer is set to the main entry routine after the module has been loaded. Program pointers in other tasks are not affected.

Arguments

When the parameter *Action* is set to *Load and Start*, the parameters *Argument1* and *Argument2* must also be used.

Parameter:	Allowed value:
Argument1	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, for example `HOME:ModuleA.mod`
Argument2	The task in which the program defined in *Argument 1* should be loaded.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, that is not used by any other resource.
- The parameter *Argument 1* has to be defined with an existing program file name.
- If the option *MultiMove* is installed, the parameter *Argument 2* must be defined with a task for which the program or module should be loaded.

Limitations

This action value has the following limitations:

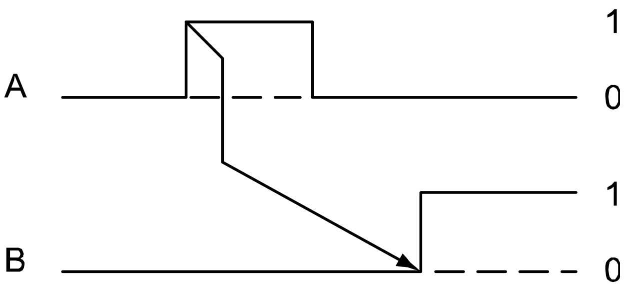
- The controller has to be in automatic mode.

Continues on next page

- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- *Load and Start* action is not valid during program execution.
- The run mode will always be set to Cyclic.
- If the controller is in Motors Off state, only the load is performed.
- If the current program has been changed, the changes will not be saved before the load.

Additional information

The signal sequence for *Load Start* is:



xx0400000949

A: Load and Start (IN)

B: Cycle On (OUT)

Related information

Action on page 268.

Argument 1 on page 303.

Argument 2 on page 305.

4.12.4.7 Motors Off

Parent

Motors Off is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

MotorOff

Description

The action value *Motors Off* sets the controller in the Motors Off state. If a program is executing, it is stopped before changing state.

Motors Off can be used by a PLC to set the controller in Motors Off state.

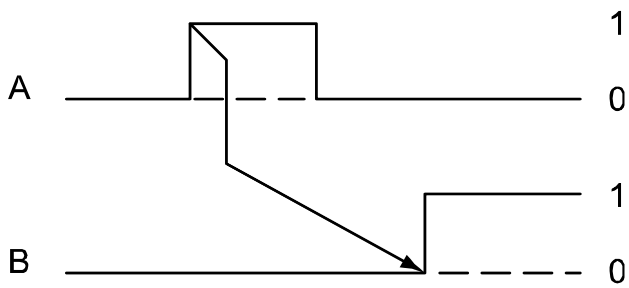
Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- We recommend stopping the program execution before using the action *Motors Off* to secure a controlled stop.

Additional information

The signal sequence for *Motors Off* is:



xx0400000949

A: Motors Off (IN)

B: Motors Off (OUT)

Related information

Operating manual - IRC5 with FlexPendant.

4.12.4.8 Motors On

Parent

Motors On is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

MotorOn

Description

The action value *Motors On* sets the controller in the Motors On state.

This action can be used by a PLC to set the controller in Motors On state.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The safety chain has to be closed. To check if the safety chain is closed, use the parameter *Run Chain OK* of the type *System Output*.

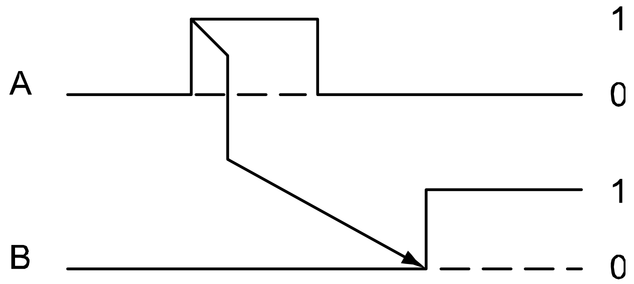
Limitations

The action value has the following limitations:

- The controller has to be in automatic mode.
- The controller cannot be in the Motors On state if the system input I/O signal action *Motors Off* is high.
- The *Motors On* action is not valid during program execution.

Additional information

The signal sequences for *Motors On* is:



xx0400000949

A: Motors On (IN)

B: Motors On (OUT)

Related information

Motors Off on page 280.

Run Chain OK on page 336.

Continues on next page

Operating manual - IRC5 with FlexPendant, chapter *Handling inputs and outputs, I/O*.

4.12.4.9 Motors On and Start

Parent

Motors On and Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

MotOnStart

Description

The action value *Motors On and Start* sets the controller in the Motors On state and starts the RAPID program from the current instruction, continuous or cycle execution.

Motor On and Start can be used by a PLC to set Motors On in one single step and start a RAPID program, instead of using the FlexPendant and the control panel.

The Program Pointer needs to be set in all tasks before starting the program. The action will be rejected if the program pointer is missing in any task.

Arguments

When the parameter *Action* is set to *Motors On and Start*, the parameter *Argument1* must also be used, specifying continuous or cycle. The default value is continuous.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The parameter *Argument 1* has to be defined with the run mode.

Limitations

The action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action value if the *Stop*, *Stop at end of Cycle*, *Stop at end of Instruction*, or *Motors Off* actions are set.
- The *Motors On and Start* action is not valid during program execution.

Related information

Argument 1 on page 303.

Action on page 268.

Operating manual - IRC5 with FlexPendant.

4.12.4.10 PP to Main

Parent

PP to Main is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

Pp to Main

Description

The action value *PP to Main* sets the program pointer to the configured production entry that is the main routine.

Usage

When the parameter *Action* is set to *PP to Main*, the parameter *Task Name* can be used to set PP to Main in a specific task. If the *Task Name* is left empty, all tasks will be affected.

Limitations

PP to Main can only be used with Normal tasks.

Related information

Action on page 268.

Task Name on page 310

4.12.4.11 Reset Emergency Stop

Parent

Reset Emergency Stop is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

ResetEStop

Description

The action value *Reset Emergency Stop* confirms the reset of an emergency stop. When an emergency stop has occurred, it must first be restored mechanically and the reset has to be confirmed. The controller can then be set to the Motors On state.

It is possible to use a PLC to confirm the reset of the emergency stop instead of using the Motors On button.

Prerequisites

The following prerequisites have to be considered:

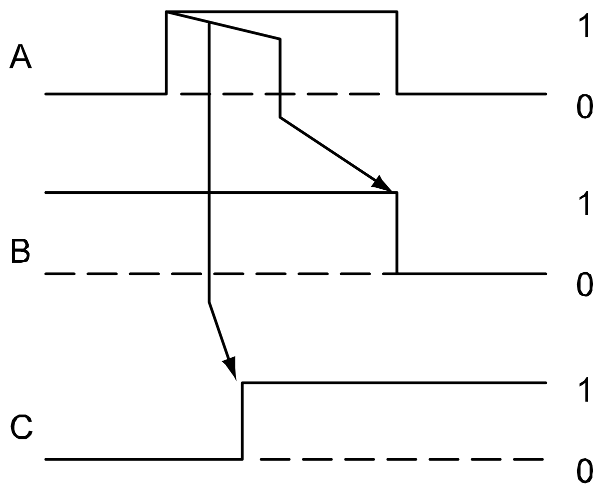
- A digital input I/O signal with a defined signal name has to be configured in the system.
- The safety chain must be closed by restoring the emergency stop mechanically.

Limitations

The controller has to be in automatic mode.

Additional information

To reset an emergency stop, set the signal sequences according to the image.



xx0400000948

A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

Continues on next page

C: Run Chain OK (OUT), Response

4.12.4.12 Reset Execution Error Signal

Parent

Reset Execution Error Signal is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

ResetError

Description

The action value *Reset Execution Error Signal* resets the system output signal action *Execution Error*.

This action can be used by a PLC to reset the error signal.

Prerequisite

A digital input I/O signal with a defined signal name has to be configured in the system.

Related information

Execution Error on page 322.

4.12.4.13 Start

Parent

Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

Start

Description

The action value *Start* starts a RAPID program from the current instruction, continuous or cycle run mode.

Start can be used by a PLC to start the program execution.

The Program Pointer needs to be set in all tasks before starting the program. The action will be rejected if the program pointer is missing in any task.

Arguments

When the parameter *Action* is set to *Start*, the parameter *Argument1* must also be used, specifying continuous or cycle. The default value is continuous.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, that is not used by any other resource.
- The parameter *Argument 1* has to be defined with the run mode.

Limitations

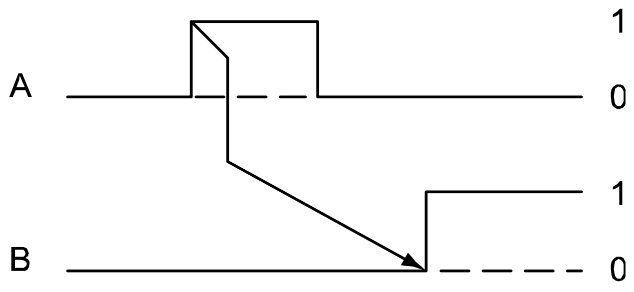
This action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- The *Start* action is not valid during program execution.

Continues on next page

Additional information

The signal sequence for *Start* is:



xx0400000949

A: Start (IN)

B: Cycle On (OUT)

Related information

Argument 1 on page 303.

Action on page 268.

4.12.4.14 Start at Main

Parent

Start at Main is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

StartMain

Description

The action value *Start at Main* starts a RAPID program from the beginning, continuous or cycle run.

Start at Main can be used by a PLC to start the program execution from the beginning.

Arguments

When the parameter *Action* is set to *Start at Main*, the parameter *Argument1* must also be used, specifying continuous or cycle. The default value is continuous.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, that is not used by any other resource.
- The parameter *Argument 1* has to be defined with the run mode.

Limitations

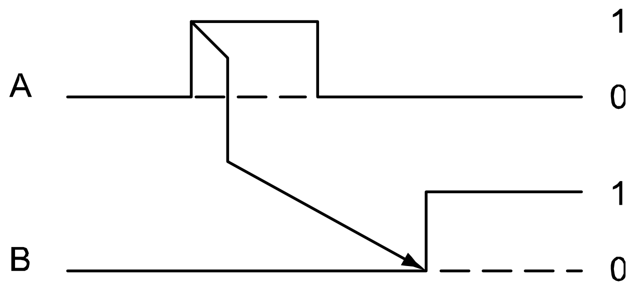
This action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- *Start at Main* action is not valid during program execution.

Continues on next page

Additional information

The signal sequence for *Start at Main* is:



xx0400000949

A: Start at Main (IN)

B: Cycle On (OUT)

Related information

Argument 1 on page 303.

Action on page 268.

4.12.4.15 Stop

Parent

Stop is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

Stop

Description

The action value *Stop* stops the RAPID program execution. All robot movements will be stopped on the path with no deviation. This stop is the slowest stop and will take a couple of hundred milliseconds extra since the demand is to stop exactly on the programmed path. The extra delay is due to a deceleration ramp that needs to be recalculated to be able to stop on the path.

A program cannot be started when this signal is high. This stop is similar to a normal program stop using the stop button on the FlexPendant.

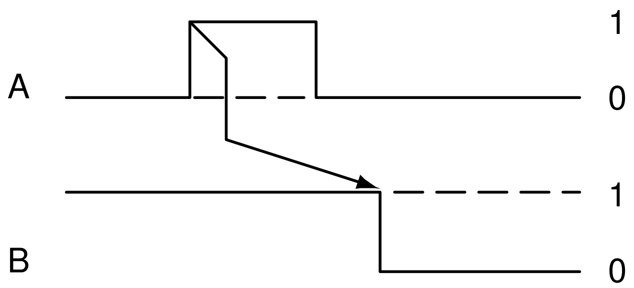
Stop can be used by a PLC to stop the program execution.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Additional information

The signal sequence for *Stop* is:



xx0400000950

A: Stop (IN)

B: Cycle On (OUT)

4.12.4.16 Quick Stop

Parent

Quick Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

QuickStop

Description

The action value *Quick Stop* stops the RAPID program execution quickly, like a category 1 emergency/safety stop. This stop is performed by ramping down motion as fast as possible using optimum motor performance. The different axes are still coordinated to try to keep the robot on path even if the robot may slide off with some millimeter.

This system output should not be used for safety functions since it is not a safety I/O signal according to ISO 10218-1 and ISO 13849-1:1999. For safety functions the options *Electronic Position Switches* or *SafeMove* can be used.

Note

This stop should not be used for normal program stops as this causes extra, unnecessary wear on the robot.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

4.12.4.17 Soft Stop

Parent

Soft Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

SoftStop

Description

The action value *Soft Stop* will stop the RAPID program execution much like an ordinary program stop, but slightly faster. The stop is performed by ramping down motion in a controlled and coordinated way, to keep the robot on the programmed path with minor deviation.

This stop has the same braking performance as stopping on path to a fine point.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

4.12.4.18 Stop at End of Cycle

Parent

Stop at End of Cycle is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

StopCycle

Description

The action value *Stop at End of Cycle* stops the RAPID program when the complete program is executed, i.e. when the last instruction in the main routine has been completed. A program cannot be started when this signal is high.

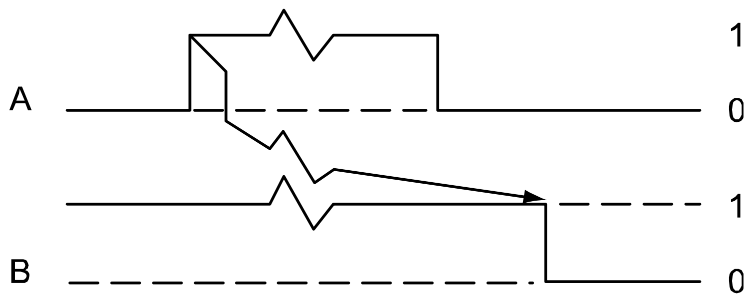
Stop at End of Cycle can be used by a PLC to stop the program execution when the complete program has been executed.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Additional information

The signal sequence for *Stop at End of Cycle* is:



xx0400000951

A: Stop at end of Cycle (IN)

B: Cycle On (OUT)

4.12.4.19 Stop at End of Instruction

Parent

Stop at End of Instruction is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

StopInstr

Description

The action value *Stop at End of Instruction* stops program execution after the current instruction is completed. A program cannot start when this signal is high.

Stop at end of Instruction can be used by a PLC to stop the program execution when the current instruction is completed.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Additional information

If using *Stop at End of Instruction* in combination with an instruction that is waiting for an I/O signal or an instruction, for example `WaitSyncTask`, `WaitDI`, or `SyncMoveOn`, then the waiting instruction may not be finished. We recommend using system input *Stop* together with *Stop at End of Instruction* to prevent the program from hanging.

Related information

Stop on page 292.

Example

If a `WaitTime` instruction is executed, it can take a while before the execution is stopped.

4.12.4.20 System Restart

Parent

System Restart is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

SysReset

Description

The action value *System Restart* performs a controller restart, similar to power off/on.

This action can be used by a PLC to restart the controller.

Prerequisites

The following prerequisites have to be considered:

- A digital input I/O signal with a defined signal name has to be configured in the system.
- We recommend stopping all RAPID programs before using the action.

4.12.4.21 SimMode

Parent

SimMode is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

SimMode

Description

The action value *SimMode* shall be set when the simulation mode shall be entered.

Arguments

When the parameter *Action* is set to *SimMode*, the *Argument1* must also be used.

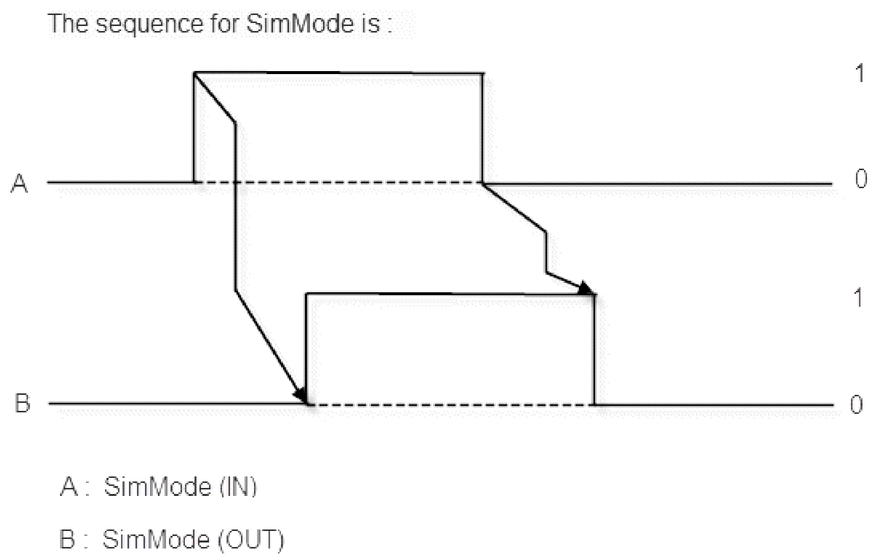
Parameter	Allowed value
Argument1	LOAD

When *Argument1* is set to LOAD, the robot shall run without payload when the signal is set.

Prerequisites

A digital input I/O signal with a defined signal name has to be available, that is, not used by any other resource.

Signal sequence



en1100000964

Additional information

A system output signal (also called *SimMode*) can be configured that reflects the status of the system state *SimMode*.

4.12.4.22 Collision Avoidance

Parent

Collision Avoidance is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

CollAvoidance

Description

The action value *Collision Avoidance* shall be set when the functionality for Collision Avoidance is activated. If no value is defined, then the functionality is not activated.

Collision Avoidance monitors a detailed geometric model of the robot. If two bodies of the model come too close to each other, the controller warns about a predicted collision and stops the robot. The system parameter *Coll-Pred Safety Distance* (*coll_pred_default_safety_distance*) determines at what distance the two objects are considered to be in collision. The default value for this parameter is 0.01 meter, but it can be set to any value between 0.001 and 1 meters.

Prerequisites

A digital input I/O signal with a defined signal name has to be configured in the system.

Limitations

This parameter is currently applicable only to IRB 14000 (YuMi robot).

4.12.4.23 Enable Energy Saving

Parent

Enable Energy Saving is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

EnableEnergySaving

Description

Setting the action value *Enable Energy Saving* enables the controller to enter an energy saving state. Resetting the signal while in an energy saving state will cause the controller to resume.

Prerequisites

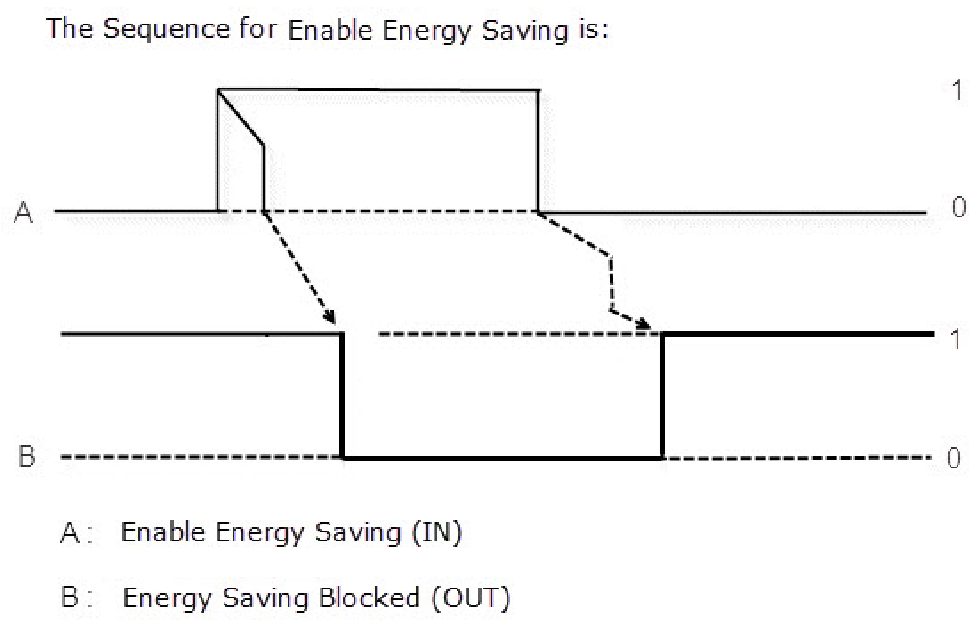
A digital input I/O signal with a defined signal name has to be configured in the system.

Limitations

The energy saving functionality is available only with *PROFIenergy*.

The action value *EnableEnergySaving* is therefore idle for the systems lacking the *PROFIenergy* option.

Signal sequence



xx1500000337

Additional information

A system output signal (called *EnergySavingBlocked*) can be configured to reflect if Energy Saving is blocked or not.

Continues on next page

It is not only the System Input Signal *EnableEnergySaving* that can cause the Energy Saving functionality to be blocked. That is, the System Output Signal *EnergySavingBlocked* can be set even if the System Input action *EnableEnergySaving* is set.

4.12.4.24 Write Access

Parent

Write Access is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O System*.

Cfg name

WriteAccess

Description

The Action value *Write Access* can be used by an I/O client to request write access the same way as can be done from RobotStudio.

The write access is granted if not already held by any other client and will prevent other clients from requesting write access, until the signal is reset.

Prerequisites

The signal can only be used in automatic mode. The write access is released when entering manual mode.

Additional information

A System Output Signal *Write Access* can be configured to reflect if the I/O client has write access or not.

4.12.5 Argument 1

Parent

Argument 1 belongs to the type *System Inputs*, in the topic *I/O System*.

Cfg name

Arg1

Description

Argument 1 is an argument required to perform the system actions *Interrupt*, *Load and Start*, *Motors On and Start*, *Start*, *Start at Main*, *Load*, or *Backup*. If the parameter *Action* has one of the action values listed above then *Argument 1* must also be set.

Limitations

The only environment variable allowed for *Argument 1* is "SYSTEM:".

Allowed values

The following values are allowed:

System action:	Allowed value:	Cfg value:
Interrupt	The name of the routine to be executed.	
Load and Start	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, for example `HOME:ModuleA.mod`	
Load	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, for example `HOME:ModuleA.mod`	
Motors On and Start	Run mode continuous or cycle, default value is continuous.	`CONT` (continuous) or `CYCLE` (cycle)
Start	Run mode continuous or cycle, default value is continuous.	`CONT` (continuous) or `CYCLE` (cycle)
Start at Main	Run mode continuous or cycle, default value is continuous.	`CONT` (continuous) or `CYCLE` (cycle)
Backup	The name of the backup. The environment variable "SYSTEM:" will indicate that the system name shall be used.	

Related information

Action on page 268.

Interrupt on page 272.

Load and Start on page 278.

Motors On and Start on page 283.

Start on page 288.

Start at Main on page 290.

Continues on next page

Load on page 276.

Backup on page 269.

Argument 2 on page 305.

Argument 3 on page 306.

Argument 4 on page 307.

4.12.6 Argument 2

Parent

Argument 2 belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg2

Description

Argument 2 is an argument required to perform the system actions *Load and Start*, *Interrupt* and *Load*, that is when the parameter *Action* is set to *Load and Start*, *Interrupt* or *Load*, *Argument 2* must be set too.

Usage

Argument 2 is used to define a task.

Prerequisites

Action must be set to *Load and Start*, *Interrupt*, or *Load*.

Limitations

Argument 2 is only used with the option *MultiMove*.

Allowed values

System action	Allowed value
Load and Start	The task in which the program defined in *Argument 1* should be loaded.
Interrupt	The task in which the module or program defined in *Argument 1* should be loaded.
Load	The task in which the program defined in *Argument 1* should be loaded.

If *MultiMove* is not installed, then *Argument 2* must be set to T_ROB1.

Related information

Action on page 268.

Load and Start on page 278.

Interrupt on page 272.

Load on page 276.

Argument 1 on page 303.

4.12.7 Argument 3

Parent

Argument 3 belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg3

Description

Argument 3 is an argument required to perform the system action *Backup*, that is, when the parameter *Action* is configured to *Backup*.

Usage

Argument 3 is used to define the path for a backup.

Prerequisites

Action must be set to *Backup*.

Allowed values

System action	Allowed value
Backup	The path of the backup.

Related information

Action on page 268.

Backup on page 269.

4.12.8 Argument 4

Parent

Argument 4 belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg4

Description

Argument 4 is an argument required to perform the system action *Backup*, that is, when the parameter *Action* is configured to *Backup*.

Usage

Argument 4 is used to define if the backup shall have a uniqe name or if a backup with the same name shall be deleted first.

Prerequisites

Action must be set to *Backup.*

Allowed values

System action	Allowed value
Backup	Unique Name or Overwritten. Default value is Unique Name.

Related information

Action on page 268.

Backup on page 269.

4.12.9 Argument 5

Parent

Argument 5 belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg5

Description

Argument 5 is a required argument for the system action *Backup*, that is, when the parameter *Action* is configured to *Backup*.

Usage

Argument 5 is used to define if the backup shall have date added in the name of the backup. If UniqueName is set as *Argument 4*, the sequence number will come after the date.

Prerequisites

Action must be set to *Backup*.

Allowed values

System action	Allowed value
Backup	AddDate or NoDate. Default value is NoDate.

Related information

Action on page 268.

Backup on page 269.

4.12.10 Argument 6

Parent

Argument 6 belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg6

Description

Argument6 is an argument required to perform the system action *LimitSpeed*, that is, when the parameter *Action* is configured to *Limit Speed*.

Usage

Argument6 is used to define if the speed reduction shall be for a specified motion task or all motion tasks.

Prerequisites

Action must be set to *Limit Speed*.

Allowed values

System action	Allowed value
Limit Speed	A robot from the type *Robot* in the topic *Motion*.

4.12.11 Task Name

Parent

Task Name belongs to the type *System Input*, in the topic *I/O System*.

Cfg name

Arg7

Description

Task Name is an argument required to perform the system input action PP to Main. It is available when the parameter *Action* is set to *PP to Main.*

Usage

Task Name can be used to specify a RAPID task or can be left blank for all normal tasks.

Allowed values

System action	Allowed value
PP to Main	A task from the type CAB_TASKS in the topic Controller.

Related information

PP to Main on page 284.

4.12.12 Overview of the values for Action

Overview

Overview showing all values for *Action* in *System Input* and how they are allowed to be used in different type of system modes and states.

	Manual full speed mode motors on program execution	Manual reduced speed mode motors on program execution	Auto mode motors off	Auto mode motors on	Auto mode motors on program execution	The controller system is in system failure state[i]	An external client has write access (e.g. RobotStudio)	During a backup operation
Backup		X	X	X	X	X	X	
DisableBackup		X	X	X	X	X	X	X[ii]
Interrupt				X				
LimitSpeed	X	X	X	X	X	X	X	X
Load			X	X				
LoadStart			See note[iii]	X				
MotOnStart			X	X			See note[iv]	Refer to note iv
MotorOff	X	X		X	X		X	X
MotorOn			X				X	X
QuickStop	X	X			X		X	X
ResetError		See note[v]	X	X	Refer to note v		X	X
ResetEstop			X	X	X		X	X
SimMode		X	X	X	X		X	X
SoftStop	X	X			X		X	X
Start				X				
StartMain				X				
Stop	X	X			X		X	X
StopCycle	X	X			X		X	X
StopInstr	X	X			X		X	X
SysReset		X	X	X	X	X	X	X[vi]

i The cause of the System Failure can have impact on the function for the given System Input Actions
ii Do not affect the ongoing Backup
iii Only load of the program module is performed
iv Motor On only
v Execution error triggered during program execution
vi Ongoing Backup will be deleted

4.13 Type System Output

4.13.1 The System Output type

Overview

This section describes the type *System Output* which belongs to the topic *I/O System*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SYSSIG_OUT

Type description

Output I/O signals can be assigned for a specific system action. These I/O signals are set automatically by the system without user input when the system action occurs.

The system output I/O signals can be both digital and analog.

Prerequisites

An I/O signal must be configured in the system. The signal name must be a string of maximum 32 characters.

Limitations

The following limitations have to be considered:

- Several output I/O signals can be assigned the same system action, but several system actions may not be assigned to the same I/O signal.
- When deleting a system action the I/O signal itself remains defined. The I/O signal must be deleted separately.
- The predefined system output for the Motors On lamp cannot be edited.

Predefined system outputs

Motors On is predefined in the robot system. This output is linked to the Motors On lamp on the controller.

Additional information

The actions are valid for both manual and automatic mode unless stated otherwise in the value descriptions.

Related information

The Signal type on page 237.

4.13.2 Status

Parent

Status belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

Status

Description

Output signals can be assigned to specific system actions. *Status* defines the system status that triggered the signal. The system actions are handled by the system without an input from the user.

Allowed values

The following values are allowed and are described on the following pages:

- *Absolute Accuracy Active on page 316*.
- *Auto On on page 317*.
- *Backup Error on page 318*.
- *Backup in progress on page 319*.
- *CPU Fan not Running on page 342*.
- *Cycle On on page 320*.
- *Emergency Stop on page 321*.
- *Execution Error on page 322*.
- *Limit Speed on page 323*.
- *Mechanical Unit Active on page 324*.
- *Mechanical Unit Not Moving on page 325*.
- *Motors Off on page 327*.
- *Motors On on page 328*.
- *Motors Off State on page 329*.
- *Motors On State on page 330*.
- *Motion Supervision On on page 331*.
- *Motion Supervision Triggered on page 332*.
- *Path Return Region Error on page 333*.
- *Power Fail Error on page 334*.
- *Production Execution Error on page 335*.
- *Run Chain OK on page 336*.
- *Simulated I/O on page 337*.
- *TaskExecuting on page 338*.
- *TCP Speed on page 339*.
- *TCP Speed Reference on page 340*.
- *Temperature Warning on page 345*.

Continues on next page

- *SimMode on page 341*.
- *SMB Battery Charge Low on page 346*
- *Energy Saving Blocked on page 343*.
- *Write Access on page 344*.

4.13.3 Signal Name

Parent

Signal Name belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

Signal

Description

Signal Name is the name of the configured digital output I/O signal to use. It connects the system output with a configured digital output I/O signal.

Prerequisites

A digital output I/O signal with a defined name has to be configured in the system.

Allowed values

Available configured digital output I/O signal names.

Related information

The Signal type on page 237.

4.13.4 Status values

4.13.4.1 Absolute Accuracy Active

Parent

Absolute Accuracy Active is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

AbsAccActivated

Description

If *Status* has the value *Absolute Accuracy Active*, the I/O signal is set when the absolute accuracy is activated. The signal is cleared when the absolute accuracy is not activated.

1: absolute accuracy is activated

0: absolute accuracy is not activated

Prerequisites

The RobotWare option *603-1 Absolute Accuracy* is necessary to configure this output signal on a system.

4.13.4.2 Auto On

Parent

Auto On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

AutoOn

Description

If *Status* has the value *Auto On*, the I/O signal is set when the controller is in automatic mode.

Related information

Operating manual - IRC5 with FlexPendant.

4.13.4.3 Backup Error

Parent

Backup Error is a value for the parameter *Status* and belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

BackupError

Description

If *Status* has the value *Backup Error*, the signal is set when the system detects the backup failure. The failure can be detected during the backup or after a power failure if the backup has been interrupted by this. The signal is cleared when a new backup is started.

Additional information

The output signal reflects the overall system backup error state independent of the application starting the backup, that is, RobotStudio, FlexPendant, and system input signal *Backup*.

Related information

Action on page 268.

Backup on page 269.

4.13.4.4 Backup in progress

Parent

Backup in progress is a value for the parameter *Status* and belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

BackupInProgress

Description

If *Status* has the value *Backup in progress*, the signal is set when a backup is started and cleared when the backup is complete with or without errors.

Additional information

This output signal reflects the overall system backup state independent of the application starting the backup, that is, RobotStudio, FlexPendant, and system input signal *Backup*.

Related information

Action on page 268.

Backup on page 269.

4.13.4.5 Cycle On

Parent

Cycle On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

CycleOn

Description

If *Status* has the value *Cycle On*, the I/O signal is set when the robot program is executing.

Additional information

Cycle On is also active for Service and Event Routine execution (Start, Restart, and Stop).

During path recovery operations, the I/O signal is set.

4.13.4.6 Emergency Stop

Parent

Emergency Stop is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

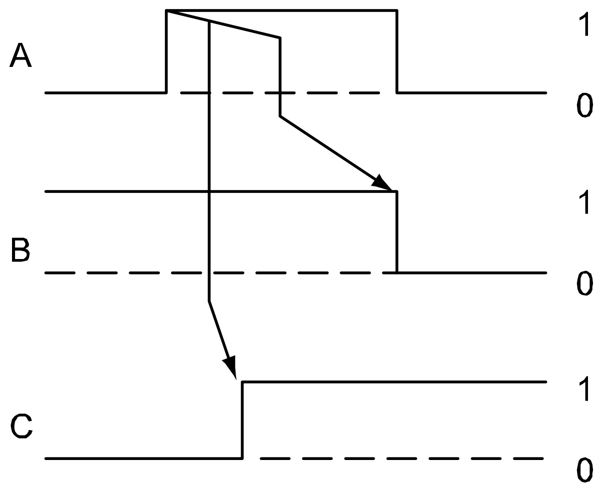
EmStop

Description

If *Status* has the value *Emergency Stop*, the I/O signal is set when the controller is in the Emergency Stop state.

Additional information

The signal sequence for *Emergency Stop* is:



xx0400000948

A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

C: Run Chain OK (OUT), Response

4.13.4.7 Execution Error

Parent

Execution Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

Error

Description

If *Status* has the value *Execution Error*, the I/O signal is set high because the robot program execution has been stopped due to a program error during execution. The execution error state occurs when there is no error recovery, that is if there is no error handler that takes care of the current error.

If *Argument 2* is specified with a task name, the I/O signal will only react on execution errors for that task.

The I/O signal stays set high until any of the following events occur for the task:

- Program start.
- Program restart.
- Reset of program pointer.
- System signal *Reset Execution Error* set high (resets all tasks).

If *Argument 2* is not specified with a task name, the I/O signal will react on execution errors in any task. In this case, the I/O signal stays high until any of the events listed above occur for any of the tasks.

The signal state is not kept after power fail (Restart of controller).

Related information

Reset Execution Error Signal on page 287.

Argument 2 on page 348.

4.13.4.8 Limit Speed

Parent

Limit Speed is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

LimitSpeed

Description

If *Status* has the value *LimitSpeed*, the I/O signal is set when the specified motion task is running with reduced speed triggered by the System Input Signal *LimitSpeed*.

Arguments

When the parameter *Status* is set to *LimitSpeed*, the parameters *Argument* and *Argument4* must also be used.

Parameter:	Allowed value:
Argument	The motion task which speed shall be monitored.
Argument4	Specifies a delay at setting of the output to minimize the risk of faulty triggering by SafeMove when the output is used to start the supervision. The default delay is 250 ms.

Prerequisites

A digital output I/O signal with a defined signal name has to be available, not used by any other resource.

Additional information

If the specified motion task is running with reduced speed, the system output will be set.

4.13.4.9 Mechanical Unit Active

Parent

Mechanical Unit Active is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MechUnit Active

Description

If *Status* has the value *Mechanical Unit Active*, the I/O signal is set when the configured mechanical unit is active.

Arguments

When the parameter *Status* is set to *Mechanical Unit Active*, the parameter *Argument* must also be used, specifying which mechanical unit the I/O signal is reflecting. The default value is ROB_1.

Note

The drop-down list in the FlexPendant or RobotStudio configuration tool shows only TCP robots. Use ABC... to add any other mechanical unit.

Additional information

If the configured mechanical unit is active, the system output will be set.

If the mechanical unit is configured to be active, the system output will already be set at start.

It is possible to deactivate a mechanical unit on the FlexPendant or via RAPID.

Related information

Argument on page 347.

4.13.4.10 Mechanical Unit Not Moving

Parent

Mechanical Unit Not Moving is a value for the parameter *Status* that belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

MechUnitNotMoving

Description

If *Status* has the value *MechUnitNotMoving*, the I/O signal is set high when the configured mechanical unit is not moving. The I/O signal is only triggered by state changes, that is auto and manual mode. Using the parameter *Mech.Unit Not Moving Detection Level* will also set the output when all axes of the Mechanical Units with a defined *Level* running in the same motion group are moving slower than its *Level*.

Arguments

When the parameter *Status* is set to *Mechanical Unit Not Moving*, the parameter *Argument* defines which mechanical unit the I/O signal is reflecting. The argument defines the name of a mechanical unit.

If *Argument* is not defined (no value) then the I/O signal will reflect the state of the system. The I/O signal will be set low when the first mechanical unit starts to move and will be set high when the last mechanical units stops to move.

The default value is empty.

Note

The drop-down list in the FlexPendant or RobotStudio configuration tool shows only TCP robots. Use ABC... to add any other mechanical unit.

Additional information

In situations where units (for example, a TCP robot and an additional axis) are synchronized in the same movement instruction or by move instructions with same ID in different tasks, the I/O signals will for all units have the same value, that is the I/O signals will not be set until all synchronized units are stopped.

The state of the I/O signal is changed during regain movement. This can make the I/O signal toggle for example when stepping over logical instructions.

This system output should not be used for safety functions since it is not a safety I/O signal according to ISO 10218-1 and ISO 13849-1:1999. For safety functions the options *Electronic Position Switches* or *SafeMove* can be used.

Related information

Argument on page 347.

The Mechanical Unit type on page 564, in topic *Motion*.

Mech.Unit Not Moving Detection Level on page 694, in the topic *Motion*, type *Robot*.

Continues on next page

Mech.Unit Not Moving Detection Level on page 731, in the topic *Motion*, type *Single*.

4.13.4.11 Motors Off

Parent

Motors Off is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotorOff

Description

If *Status* has the value *Motors Off*, the I/O signal is set when the controller is in the Motors Off state.

Additional information

When the controller is in Motors Off state and a safety chain is not closed, the output I/O signal pulses.

If only Motors Off state is requested, the action value *Motors Off State* is preferred.

Related information

Motors Off State on page 329.

Run Chain OK on page 336.

4.13.4.12 Motors On

Parent

Motors On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotorOn

Description

If *Status* has the value *Motors On*, the I/O signal is set when the controller is in the Motors On state.

Additional information

If the controller is in guard stop, the output starts pulsing with a frequency of 1 sec. If the controller is not calibrated or the revolution counter is not updated, the output will pulsate even faster in manual mode.

Motors On can be used to detect if the controller is in Motors On and whether the controller is synchronized or not.

Related information

Motors On State on page 330.

4.13.4.13 Motors Off State

Parent

Motors Off State is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotOffState

Description

If *Status* has the value *Motors Off State*, the I/O signal is set when the controller is in the Motors Off state.

4.13.4.14 Motors On State

Parent

Motors On State is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotOnState

Description

If *Status* has the value *Motors On State*, the I/O signal is set when the controller is in the Motors On state.

4.13.4.15 Motion Supervision On

Parent

Motion Supervision On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotSupOn

Description

If *Status* has the value *Motion Supervision On*, the I/O signal is set when the motion supervision function is active.

Prerequisites

When the parameter *Status* is set to *Motion Supervision On*, the parameter *Argument* must also be used, specifying which robot the supervision is used for. The default value is ROB_1.

Additional information

Motion Supervision On is only valid when the robot is in status Motors On.

Related information

Application manual - Controller software IRC5

4.13.4.16 Motion Supervision Triggered

Parent

Motion Supervision Triggered is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

MotSupTrigg

Description

If *Status* has the value *Motion Supervision Triggered*, the I/O signal is set when the motion supervision function has been triggered.

The signal is set when *Manipulator Supervision* (IRB 360 only) is triggered as well.

Prerequisites

If the parameter *Argument* specifies a robot, the I/O signal will only show if collision detection has been triggered for that robot. If the parameter *Argument* is not used, the I/O signal will show if collision detection has been triggered for any robot.

The signal is set when *Manipulator Supervision* (IRB 360 only) is triggered as well.

Additional information

The I/O signal is reset by one of the following actions:

- The program is restarted.
- The program pointer is manually moved to Main.
- The error message is acknowledged.
- The collision has been handled in an error handler and resumed to normal execution. The signal will then be set only for a short while during execution in the error handler.

Related information

Application manual - Controller software IRC5

CollisionErrorHandling on page 157

4.13.4.17 Path Return Region Error

Parent

Path Return Region Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

RegainDistError

Description

If *Status* has the value *Path Return Region Error*, the I/O signal is set when an attempt to start the robot program has been made but failed since the robot was too far from the programmed path.

Prerequisites

If the parameter *Argument* specifies a robot, the I/O signal will only show if that robot is too far from the programmed path. If the parameter *Argument* is not used, the I/O signal will show if any robot is too far from the programmed path.

Additional information

The value *Path Return Region Error* is set if the current movement is interrupted and then:

- The robot is jogged too far from the programed path and then restarted.
- An emergency stop has occurred and the robot has slid too far away from its programmed path and then restarted.

The I/O signal is reset by one of the following actions:

- The program is restarted after the robot has been jogged into the regain zone.
- The program pointer is manually moved to Main.
- The program pointer is manually moved and the program is restarted.

The distances of the zones can be configured in the type *Return Region* in the topic *Controller*.

Related information

The Path Return Region type on page 139 in the topic *Controller*.

4.13.4.18 Power Fail Error

Parent

Power Fail Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

PFError

Description

If *Status* has the value *Power Fail Error*, the I/O signal is set when a program cannot continue from its current position after a power failure.

Additional information

The program will not restart after the value *Power Fail Error* is set. Usually, the program can be started, but it will always start from the beginning.

4.13.4.19 Production Execution Error

Parent

Production Execution Error is a value for the parameter *Status* that belongs to the type *System Output*, in the topic *I/O System*.

Cfg name

ProdExecError

Description

If *Status* has the value *Production Execution Error*, the I/O signal is set high if the system is in automatic mode and when at least one normal task is running and one of the following occurs:

- A program execution error in any normal task.
- A collision*
- A system error: SysFail, SysHalt, or SysStop.

The I/O signal is reset by:

- Program start.
- Program restart.

The I/O signal value is not kept after a restart.

*) **Note!** This is *not* a replacement for *Motion Supervision Triggered*.

Additional information

Using *Production Execution Error* does not effect the functionality in the option *Collision Detection*, nor can it replace the option *Collision Detection*.

Related information

Execution Error on page 322.

Motion Supervision Triggered on page 332.

System errors are described in parameter *Trustlevel on page 165*.

The instruction *SystemStopAction*, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

4.13.4.20 Run Chain OK

Parent

Run Chain OK is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

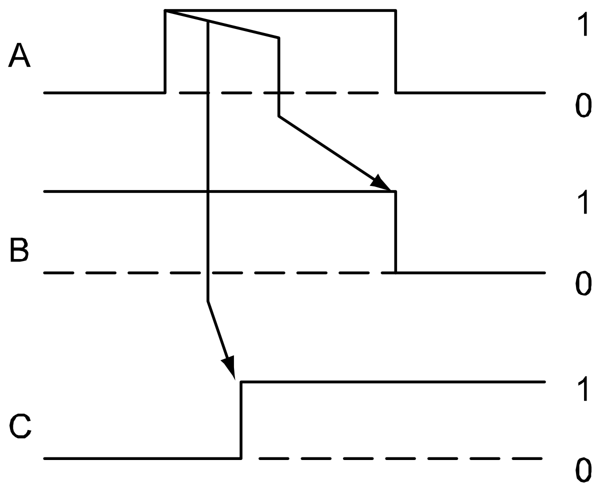
RunchOk

Description

If *Status* has the value *Run Chain OK*, the I/O signal is set when the safety chain is closed. The safety chain must be closed to be able to go to Motors On.

Additional information

Signal sequence:



xx0400000948

A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

C: Run Chain OK (OUT), Response

Example

In Manual mode the safety chain is opened and *Run Chain OK* is not set.

4.13.4.21 Simulated I/O

Parent

Simulated I/O is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

Blocked I/O

Description

If *Status* has the value *Simulated I/O*, the I/O signal is set when at least one I/O signal at any I/O device is in simulated mode.

Additional information

I/O signals can be set to simulated mode during testing, using the FlexPendant.

Related information

Operating manual - IRC5 with FlexPendant.

4.13.4.22 TaskExecuting

Parent

TaskExecuting is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

TaskExecuting

Description

If *Status* has the value *TaskExecuting*, the I/O signal is set when the configured task is executing.

During path recovery operations, the I/O signal is not set.

Prerequisites

The parameter *Argument 2* has to be defined with a task name.

Limitations

The parameter *Argument 2* can be configured only with the name of a NORMAL task.

4.13.4.23 TCP Speed

Parent

TCP Speed is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

TCPSpeed

Description

If *Status* has the value *TCP Speed*, the I/O signal is an analog signal that reflects the speed of the robot's TCP.

Prerequisites

When the parameter *Status* is set to *TCP Speed*, the parameter *Argument* must also be used, specifying which robot the speed refers to. The default value is ROB_1.

Additional information

The logical value of the I/O signal is specified in m/s, for example a speed of 2000 mm/s corresponds to the logical value 2 m/s. The scaling factor for the physical value is specified in the parameters of the corresponding I/O signal.

The analog output is set approximately 40 ms before the actual TCP speed occurs. This prediction time is constant during acceleration and deceleration.

NOTE! The *EvenPreset Time* parameter affects the time interval between the setting up of the analog output and the occurance of the TCP speed. For example, if *Event Preset Time* is set to 0.2 (200 ms), the analog output is set 240 ms before the occurance of the TCP speed.

Related information

Maximum Logical Value on page 256.

Maximum Physical Value on page 258.

4.13.4.24 TCP Speed Reference

Parent

TCP Speed Reference is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

TCPSpeedRef

Description

If *Status* has the value *TCP Speed Reference*, the I/O signal is an analog signal describing the programmed speed of the robot's TCP.

Prerequisites

When the parameter *Status* is set to *TCP Speed Reference*, the parameter *Argument* must also be used, specifying which robot the programmed speed refers to. The default value is ROB_1.

Additional information

TCP Speed Reference works in the same way as *TCP Speed* but uses the programmed speed.

Note: *TCP Speed* can differ from *TCP Speed Reference*, for example at acceleration or if the override speed has been changed.

Related information

TCP Speed on page 339.

4.13.4.25 SimMode

Parent

SimMode is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

SimMode

Description

If status has the value *SimMode*, the I/O signal is set when the state *SimMode* is set. The signal is cleared when the state *SimMode* is cleared.

Arguments/Prerequisites

When the parameter *Status* is set to *SimMode*, the parameter *Argument 3* must also be used, specifying the type of *SimMode*. Currently only *Load* is available as *SimMode*.

Additional information

After a restart, the system output signal *SimMode* will also reflect the state *SimMode*.

Related information

SimMode on page 298.

4.13.4.26 CPU Fan not Running

Parent

CPU Fan not Running is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

CpuFanNotRunning

Description

If *Status* has the value *CPU Fan not Running*, the I/O signal is set when there is CPU fan spinning slowly in the main computer unit. The signal is cleared when the CPU fan is spinning in the main computer unit.

0: The fan spins

1: The fan spins too slowly

Additional information

The CPU fan spins when the computer component heats up and provides cooling. Hence, the CPU fan may not spin during normal conditions and the CPU fan is not supervised on low CPU temperatures, that is below 39 degrees Celsius.

4.13.4.27 Energy Saving Blocked

Parent

Energy Saving Blocked is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

EnergySavingBlocked

Description

If *Status* has the value *EnergySavingBlocked*, the I/O signal is set when the Energy Saving functionality is blocked (disabled).

Prerequisites

A digital output I/O signal with a defined signal name has to be available, not used by any other resource.

Limitations

The energy saving functionality is available only with *PROFIenergy*.

The status value *EnergySavingBlocked* is therefore idle for the systems lacking the *PROFIenergy* option.

Additional information

It is not only the System Input Signal *EnableEnergySaving* that can cause the Energy Saving functionality to be blocked. That is, the System Output Signal *EnergySavingBlocked* can be set even if the System Input action *EnableEnergySaving* is set.

4.13.4.28 Write Access

Parent

Write Access is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

WriteAccess

Description

The status value *Write Access* can be used to reflect if the I/O client has write access or not.

Write access can be requested through the System Inut Signal *Write Access*.

4.13.4.29 Temperature Warning

Parent

Temperature Warning is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

TemperatureWarning

Description

If *Status* has the value *Temperature Warning*, the I/O signal is set when there is over-temperature in the main computer CPU. The signal is cleared when temperature in the main computer CPU is okay.

0: Main computer temperature is okay.

1: Overtemperature in main computer.

Additional information

CPU temperature is cyclically supervised in every 5 second. The overtemperature limit is 95 degrees Celsius.

4.13.4.30 SMB Battery Charge Low

Parent

SMB Battery Charge Low is a value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O System*.

Cfg name

SmbBatteryChargeLow

Description

If *Status* has the value *SMB Battery Charge Low*, the I/O signal is set when the SMB battery is depleted and to be replaced soon. The signal is cleared when the SMB battery charge is okay. The signal describes the state of Serial Measurement Board (SMB) batteries.

There is only one battery in a normal single robot system. However, there are upto 16 SMB batteries in a running MultiMove or/and external axes system. The output is activated if any of the batteries are depleted.

Elog message (8213 SYS_ERR_HW_SMB_WARNING_BATTERY_LOW) is also activated and gives information about which battery should be replaced.

0: SMB battery charge is okay.

1: SMB battery will be soon depleted. Replace battery at a suitable opportunity.

Additional information

SMB batteries are cyclically supervised every 10th hour. After changing the depleted battery on a running system, it can take up to 10 hours for signal to reset. A restart sets the system output direct.

4.13.5 Argument

Parent

Argument belongs to the type *System Outputs*, in the topic *I/O System*.

Cfg name

Arg1

Description

Argument is an argument required to perform the system actions *TCP Speed*, *TCP Speed Reference*, or *Motion Supervision On*, that is when the parameter *Action* has one of the action values listed above, *Argument* must be set too.

Allowed values

If the parameter *Status* has the value *TCP Speed*, *TCP Speed Reference*, or *Motion Supervision On*, the allowed value for *Argument* is a robot from the type *Robot* in the topic *Motion*. Default value is ROB_1.

If the parameter Status has the value *Path Return Region Error* or *Motion Supervision Triggered*, the allowed value for *Argument* is a robot from the type *Robot* in the topic *Motion*. If no robot is specified, the I/O signal reacts on any robot.

If the parameter *Status* has the value *Mechanical Unit Active*, the allowed value for *Argument* is a mechanical unit of the type *Mechanical Unit* in the topic *Motion*. Default value is ROB_1.

If the parameter *Status* has the value *Mechanical Unit Not Moving*, the allowed value for *Argument* is a mechanical unit of the type *Mechanical Unit* in the topic *Motion* or empty. Default value is empty.

Related information

Action value *TCP Speed on page 339*.

Action value *TCP Speed Reference on page 340*.

Action value *Motion Supervision On on page 331*.

Action value *Mechanical Unit Active on page 324*.

Action value *Mechanical Unit Not Moving on page 325*.

The Robot type on page 668 in the topic *Motion*.

The Mechanical Unit type on page 564 in the topic *Motion*.

4.13.6 Argument 2

Parent

Argument 2 belongs to the type *System Outputs*, in the topic *I/O System*.

Cfg name

Arg2

Description

Argument 2 is an argument required to perform system action *TaskExecuting* or *Execution Error*, that is when the parameter *Status* has the value *TaskExecuting* or *Execution Error*, *Argument 2* must be used to specify the task name.

Allowed values

If the parameter *Status* has the value *TaskExecuting* or *Execution Error*, the allowed value is a task name from the type *Task* in the topic *Controller*.

Related information

TaskExecuting on page 338.

Execution Error on page 322.

5 Topic Man-machine communication

5.1 The Man-machine communication topic

Overview

This chapter describes the types and parameters of the *Man-machine communication* topic.

Description

The *Man-machine communication* topic contains parameters for, among other things, creating customized lists for instructions and I/O signals, simplifying everyday work.

The parameters are organized in the following types:

1. Automatically Switch Jog Unit
2. Backup Settings
3. Most Common Instruction - List 1
4. Most Common Instruction - List 2
5. Most Common Instruction - List 3
6. Most Common I/O Signal
7. Production permission
8. Warning at Start

The types for *Most Common Instructions* are identical and therefore only described in one section, but valid for all three types.

5.2 Type Automatically Switch Jog Unit

5.2.1 The Automatically Switch Jog Unit type

Overview

This section describes the type *Automatically Switch Jog Unit* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

AUTO_SWITCH_OF_JOG_UNIT

Type description

The type *Automatically Switch Jog Unit* is used to automatically activate a mechanical unit when switching to a program editor on the FlexPendant, that uses the mechanical unit.

The default setting is that a mechanical unit is not activated automatically when switching to a program editor using an deactivated mechanical unit.

Limitations

There can be only one set of parameters of the type *Automatically Switch Jog Unit* in the system.

5.2.2 Enable switch jog unit

Parent

Enable switch jog unit belongs to the type *Automatically Switch Jog Unit*, in the topic *Man-machine communication*.

Cfg name

enabled

Description

Enable switch jog unit defines if a mechanical unit should be activated automatically when switching program editor.

Usage

Set *Enable switch jog unit* to *Yes* to automatically activate the mechanical unit when switching to a program editor that uses the mechanical unit.

Allowed values

Yes or *No*. Default value is *No*.

5.3 Type Backup Settings

5.3.1 The Backup Settings type

Overview

This section describes the type *Backup Settings* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

BACKUP

Type description

The *Backup Settings* shall be configured when the FlexPendant backup application shall suggest a specific name or path for the backup, or when the user shall be prevented from changing these settings in the FlexPendant backup application.

Limitations

Only one set of parameters of the type *Backup Settings* can be configured in the system.

5.3.2 Name

Parent

Name belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

Cfg name

Backup_name

Description

Name defines the suggested name for the backups created from the FlexPendant.

Usage

The name of the backup.

Allowed values

A string defining the name.

Additional information

The suggested name is not defined only by this parameter. If *Unique Name* is set to *Yes* and if a backup already exists with the same name, an increasing number is added to the end of the name.

If the *Name* parameter is undefined, the default backup name SystemName_Backup_Date (for example, SystemX_Backup_20100101) is suggested.

Related information

Unique name on page 355.

5.3.3 Path

Parent

Path belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

Cfg name

Backup_path

Description

Path defines the suggested path for the backups created from the FlexPendant.

Usage

The path for the backup.

Allowed values

A string defining the path.

Additional information

If the *Path* parameter is undefined, the default backup path BACKUP is suggested.

Example 1

The environment variable BACKUP can be used.

BACKUP/SysInBackup

5.3.4 Unique name

Parent

Unique name belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

Cfg name

Unique_name

Description

Unique name defines if the backup shall be overwritten or get a unique name if it already exists a backup with name *Name*.

Usage

A unique name is suggested if the value of *Unique name* is set to Yes. An increasing number is added at the end of the name if a backup with the same name already exists. The user will get the option to overwrite the old backup if the value of *Unique name* is set to No and if a backup with the same name already exists.

Allowed values

Yes or No.

5.3.5 Disable name change

Parent

Disable name change belongs to the type *Backup Settings*, in the topic *Man-machine communication.*

Cfg name

Disable_name_change

Description

Disable name change prevents the users from changing the name and the path from the FlexPendant backup application.

Usage

Setting the value of the *Disable name change* parameter to Yes prevents the users from changing the suggested name and path in the FlexPendant backup application.

Allowed values

Yes or No.

The default value is No.

5.4 Type Most Common Instruction

5.4.1 The Most Common Instruction types

Overview

This section describes the types *Most Common Instruction - List 1*,*Most Common Instruction - List 2*, and *Most Common Instruction - List 3* which belongs to topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg names

MMC_MC1

MMC_MC2

MMC_MC3

Type description

The system contains lists of instructions to use when programming the robot. There are also three lists available to adapt to personal requirements. These are called *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3*.

The three lists are set up of a number of parameters equal between the lists. Therefore the parameters are described together in this manual.

Required parameters

Only the system parameter *Name* requires a value.

Related information

Instructions and their optional arguments and syntax are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Example: Instruction without argument

To create a `MoveJ` instruction without arguments, only the parameter *Name* is required if *Name* is set to `MoveJ`, exactly as spelled in RAPID.

Parameter:	Value:
Name	MoveJ
Parameter Number	
Alternative Number	
Instruction Name	
Only for Motion Task	

Continues on next page

Example: Instruction with argument

To create a `MoveL` instruction with the option Time set to the alternative T for motion tasks, use the following values.

Parameter:	Value:
Name	MoveL /T
Parameter Number	5
Alternative Number	2
Instruction Name	MoveL
Only for Motion Task	Yes

By setting *Name* to MoveL/T, the button label in the picklist will clearly state to the user that this is a `MoveL` instruction, using the Time option. The parameter number we use is 5, see table below, and we use alternative 2 for [\T]. Since *Name* is not set to only `MoveL`, we must use *Instruction Name* to specify to the system that it is a `MoveL` instruction. *Only for Motion Task* states that it will only be available for motion tasks.

The syntax for the `MoveL` instruction is:

Parameter Number:	Value:
<instr>	MoveL
1	[\Conc]
2	ToPoint
3	[\ID]
4	Speed
5	[\V] or [\T]
6	Zone
7	[\Z]
8	[\Inpos]
9	Tool
10	[\WObj]
11	[\Corr]

5.4.2 Name

Parent

Name belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine communication*.

Cfg name

name

Description

Name defines the name to be visible on the button in the picklist.

Usage

If *Name* is set to an instruction or procedure spelled exactly as in RAPID, no other parameters require a value. But, if *Name* contains more information, as recommended when using instructions with arguments, then the parameter *Instruction Name* specifies the actual instruction syntax.

Allowed values

The instruction name, a string with maximum 32 characters, e.g. "MoveJ".

Note

Do not use a backslash (\) in the name! Names using a backslash will cause errors, unlike when programming in RAPID.

If an additional switch or argument is used, it is recommended to include this in the name for clarity and append the name with a slash (/) and the argument, e.g. "ArcL/On". Furthermore if an optional argument is included in the name then the parameter *Instruction Name* must be set to the instruction.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

Instruction Name on page 362.

Examples

Value:	Description:
MoveJ	The instruction `MoveJ`.
ArcL/On	The instruction `ArcL` with the argument `On`.

5.4.3 Parameter Number

Parent

Parameter Number belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine communication*.

Cfg name

param_nr

Description

Parameter Number specifies which argument should be used for instructions with optional arguments.

Usage

If an instruction with optional arguments is used, then *Parameter Number* specifies which of the arguments should be used. The instructions with parameter numbers are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

If left blank, no optional argument is used.

Allowed values

A positive integer value, starting from 0.

Additional information

If *Parameter Number* is used, then *Alternative Number* must also be used.

Related information

Instruction Name on page 362.

Alternative Number on page 361.

Technical reference manual - RAPID Instructions, Functions and Data types.

5.4.4 Alternative Number

Parent

Alternative Number belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine communication*.

Cfg name

alt_nr

Description

Alternative Number defines which of the optional argument's alternatives to be used for the instruction.

Usage

If the instruction has optional arguments, then *Alternative Number* specifies which of the alternatives to use. The *Parameter Number* specifies which argument to be used.

Prerequisites

The parameter *Parameter Number* must be used.

Allowed values

The following values are allowed (depending on the number of alternatives available for the instruction):

Value:	Description:
0	no alternative is used
1	the first alternative is used
n...	the n^{th} alternative is used

Related information

Instruction Name on page 362.

Parameter Number on page 360.

Technical reference manual - RAPID Instructions, Functions and Data types.

5.4.5 Instruction Name

Parent

Instruction Name belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine communication*.

Cfg name

instr_name

Description

Instruction Name defines which instruction to use if the parameter *Name* contains more information than only the instruction.

Usage

If the instruction contains optional arguments, it is recommended to mark this in the parameter Name. Then *Instruction Name* is used to specify the instruction, as spelled in RAPID.

Allowed values

The instruction name, a string with maximum 32 characters, as spelled in RAPID.

Related information

Name on page 359.

Parameter Number on page 360.

Alternative Number on page 361.

Technical reference manual - RAPID Instructions, Functions and Data types.

5.4.6 Only for Motion Task

Parent

Only for Motion Task belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine communication*.

Cfg name

only_mec_task

Description

Only for Motion Task defines if the instruction only should be visible in Motion Tasks, i.e. should control the robot movement, e.g. `MoveJ`.

Usage

Set *Only for Motion Task* to True if the instruction only should be visible to Motion Tasks.

Allowed values

True or False.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

5.5 Type Most Common I/O Signal

5.5.1 The Most Common I/O Signal type

Overview

This section describes the type *Most Common I/O Signal* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

IO_MOST_COMMON

Type description

It is possible to have hundreds of I/O signals in the system. To simplify working with them it is possible to group them to a list of the mostly used signals. This list is defined by the type *Most Common I/O Signal*.

Prerequisites

A signal must be configured in the system for the signal name.

Example

This is a typical example of an often used I/O to be included in the list.

Parameter:	Value:
Signal Name	MySignalDI1
Signal Type	DI

5.5.2 Signal Name

Parent

Signal Name belongs to the type *Most Common I/O Signal*, in the topic *Man-machine communication*.

Cfg name

name

Description

The *Signal Name* is the I/O signal to be part of the Most Common List.

Prerequisites

A signal must be configured in the system.

Allowed values

A signal configured in the system, a name with a maximum of 32 characters.

Related information

The Signal type on page 237.

5.5.3 Signal Type

Parent

Signal Type belongs to the type *Most Common I/O Signal*, in the topic *Man-machine communication*.

Cfg name

type

Description

Signal Type defines the type of signal to be used in the common list.

Allowed values

The following values are allowed.

Value:	Description:
DI	Digital Input
DO	Digital Output
AI	Analog Input
AO	Analog Output
GI	Group Input
GO	Group Output

5.6 Type Production Permission

5.6.1 The Production Permission type

Overview

This section describes the type *Production Permission* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

PROD_PERMISSION

Type description

Different types of operating restrictions and other features may be connected to specific operating modes. Such connections are specified in the *Production Permission* type.

5.6.2 Name

Parent

Name belongs to the type *Production Permission* in the topic *Man-machine communication*.

Cfg name

name

Description

The parameter *Name* specifies the name of the permission.

Usage

The name of the permission is used as a reference to a specific permission when configuring the system.

Allowed values

RUN Mode.

5.6.3 Permission

Parent

Permission belongs to the type *Production Permission* in the topic *Man-machine communication*.

Cfg name

permission

Description

The parameter *Permission* specifies whether switching to Cycle_mode while running in the Auto mode should be allowed or not.

While running in the Auto Mode, it is normally possible to choose between Cycle_mode and Continuous_mode. In certain circumstances, this is not desired: always when running in the Auto Mode, the Continuous_mode must be active.

The parameter type restricts or permits switching to Cycle_mode while in the Auto mode.

If the name is set to RUN Mode, the permission may be set to Restricted in Auto, and it will not be possible to switch from Continuous_mode to Cycle_mode while in the Auto Mode.

Allowed values

Value	Description
Changeable in Auto	This setting enables the system to be switched to Cycle_mode or Continuous_mode while running in the Auto Mode.
Restricted in Auto	This setting prohibits the system to be switched to Cycle_mode while running in the Auto Mode. Only Continuous_mode is possible.

Default value is Changeable in Auto.

5.7 Type T10 Function Keys

5.7.1 The T10 Function Keys type

Overview

This section describes the type *T10 Function Keys* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MMC_T10_KEYS

Type description

This type is used for configuring the behavior of the function keys (F1 - F4) of the T10 jogging device.

Prerequisites

The RobotWare option *976-1 T10 Support* is necessary to run the T10 with the IRC5 robot controller.

5.7.2 Function Key

Parent

Function Key belongs to the type *T10 Function Keys*, in the topic *Man-machine communication*.

Cfg name

name

Description

Function Key defines the different function keys available to perform the jogging.

Usage

Function keys F1 - F4 are used to jog in the T10 jogging device.

Allowed values

- F1
- F2
- F3
- F4

5.7.3 Action

Parent

Action belongs to the type *T10 Function Keys*, in the topic *Man-machine communication*.

Cfg name

action

Description

Action is the resultant action that happens when different function keys are selected.

Usage

One action can be set that is associated to each function key.

Allowed values

Action:	Description:
Acknowledge Auto Change	Acknowledges an auto change
PP to Main	Moves the program pointer of all tasks to their respective main routine.
Start RAPID Execution	Starts execution of the currently selected tasks in the task panel
Stop RAPID Execution	Stops all tasks
None	No action will be performed (default)

Default value

None

5.7.4 Argument

Parent

Argument belongs to the type *T10 Function Keys*, in the topic *Man-machine communication*.

Cfg name

argument

Description

Argument can be set for a specific action. Currently, it is not used.

5.7.5 Permitted in Auto

Parent

Permitted in Auto belongs to the type *T10 Function Keys*, in the topic *Man-machine communication*.

Cfg name

allow_in_auto

Description

Permitted in Auto defines that the action is permitted in automatic mode. However, it is by default not permitted in automatic mode.

Usage

If *Permitted in Auto* option is:

- *Yes*, then the action is allowed to run in both automatic mode and manual mode.
- *No*, then the action is allowed to run in manual mode only.

Allowed values

Yes or No.

Default value

No

5.8 Type Warning at Start

5.8.1 The Warning at Start type

Overview

This section describes the type *Warning at Start* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

WARN_AT_START

Type description

If *Warning at Start* is used, then if the program pointer (PP) and the cursor are not on the same instruction when starting a program then a dialog box is displayed. The program pointer must be moved to the cursor, or the cursor moved to the program pointer, before the program can be started.

The default setting is that a warning is not displayed. Then the cursor is automatically set to the program pointer and the program is started.

The system must be restarted for changes to take effect.

Limitations

There can be only one instance of the type *Warning at Start* in the system. The name of the instance must not be changed.

The type *Warning at Start* can only be changed via configuration files.

5.8.2 Cursor PP Diff Warning

Parent

Cursor PP Diff Warning belongs to the type *Warning at Start*, in the topic *Man-machine communication*.

Cfg name

Warn

Description

Cursor PP Diff Warning defines if a warning should be displayed if the user tries to start a program when program pointer and cursor are not on the same row.

Usage

Set *Cursor PP Diff Warning* to 1 if the warning should be displayed.

If the operator taps *Cursor PP Diff Warning*then the cursor is moved to the row where the program pointer is and the program can be started.

Allowed values

0 or 1. Default value is 0.

5.8.3 Show PP to Cursor Button

Parent

Show PP to Cursor Button belongs to the type *Warning at Start*, in the topic *Man-machine communication*.

Cfg name

Visible

Description

Show PP to Cursor Button defines if the button labelled *Cursor* should be visible in the warning displayed if the user tries to start a program when program pointer and cursor are not on the same row.

Usage

Set *Show PP to Cursor Button* to 1 if the button should be visible.

If the operator taps *Cursor* then the program pointer is moved to the row where the cursor is and the program can be started.

Prerequisites

The cursor button will only available if the operator has UAS grant UAS_RAPID_DEBUG..

Allowed values

0 or 1. Default value is 0.

This page is intentionally left blank

6 Topic Motion

6.1 The Motion topic

Overview

This chapter describes the types and parameters of the *Motion* topic. Each parameter is described in the section for its type.

The topic *Motion* is extensive, with some 40 types. This manual revision covers the most commonly used parameters and types.

Description

Motion contains parameters associated with motion control in the robot and external equipment. The topic includes configuring the calibration offset and the working space limits.

The described parameters are organized in the following types:

1. Acceleration Data
2. Arm
3. Arm Check Point
4. Arm Load
5. Brake
6. Control Parameters
7. Drive Module
8. Drive System
9. Drive Unit
10. External Motion Interface Data
11. Force Master
12. Force Master Control
13. Friction Compensation
14. Jog Parameters
15. Joint
16. Lag Control Master 0
17. Linked M Process
18. Mains
19. Measurement Channel
20. Mechanical Unit
21. Motion Planner
22. Motion Process Mode
23. Motion Supervision
24. Motion System
25. Motor
26. Motor Calibration

Continues on next page

27 Motor Type
28 Path Sensor Synchronization
29 Process
30 Relay
31 Robot
32 Robot Serial Number
33 SG Process
34 Single
35 Single Type
36 Stress Duty Cycle
37 Supervision
38 Supervision Type
39 Transmission
40 Uncalibrated Control Master 0

Configuration results

Changed motion parameters requires a restart of the controller. Otherwise the changes will not have any effect on the system.

An exception to the rule is the motion supervision parameters which do not require a restart. See the type *Motion Supervision* section for more information.

6.2 Workflows

6.2.1 How to define base frame

The robot and the base frame

Normally, the base frame of the robot coincides with the world frame. However, the base frame can be moved relative to the world frame.

CAUTION

The programmed positions are always related to the world frame. Therefore, all positions are also moved, as seen from the robot.

How to define the base frame

To define the base frame:

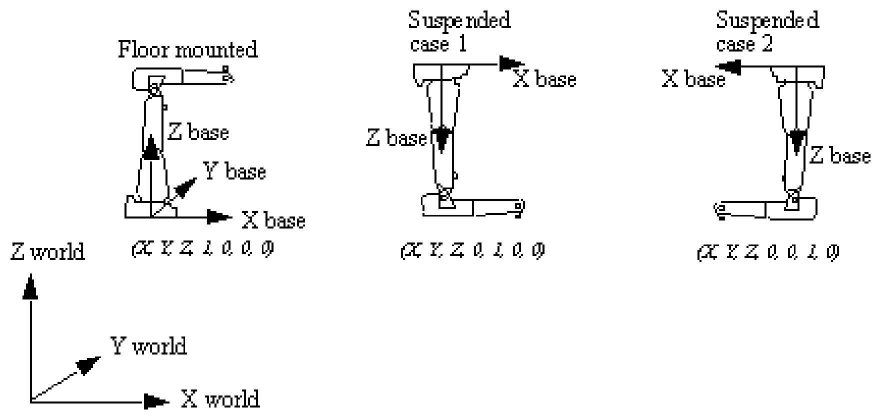
1. In the **Motion** topic, choose the type **Robot**.
2. Select the robot to define the base frame for.
3. Edit the parameters defining the base frame:
   - *Base Frame x*
   - *Base Frame y*
   - *Base Frame z*
   - *Base Frame q1*
   - *Base Frame q2*
   - *Base Frame q3*
   - *Base Frame q4*
   - *Base Frame Moved by*

   For detailed information about each parameter, see the descriptions in the *Robot* type section.
4. Save the changes.

Continues on next page

Additional information

The illustration shows some examples of frame definitions.



en0300000423

Related information

The Robot type on page 668.

6.2.2 How to define gravity

The robot and the gravity

Normally, the gravity does not need to be defined when the robot is mounted on the floor or parallel to the floor. However, the robot can be mounted, for example, on a wall or upside down. In these cases, the robot orientation relative to the gravity needs to be defined.

How to define the gravity

To define the gravity:

1. In the **Motion** topic, choose the type **Robot**.
2. Select the robot to define the gravity for.
3. Edit the parameters defining the gravity:
   - *Gravity Alpha*
   - *Gravity Beta*

   If both angles are needed to describe the robot orientation then the orientation is described by first rotating the robot around X in the base coordinate system with the *Gravity Alpha* parameter and then around Y in the rotated coordinate system with *Gravity Beta* parameter.

   For detailed information about each parameter, see the descriptions in the *Robot* type section.
4. Save the changes and restart controller.

Related information

Gravity Alpha on page 677
Gravity Beta on page 680

6.2.3 How to restrict the work area for articulated robots

Robot work area

The work area for an articulated robot is restricted by limiting the working range for the axes. The work area can also be restricted using hardware stops.

To restrict the robot work area for articulated robots:

1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to edit.
3. Edit the parameters *Upper Joint Bound* and *Lower Joint Bound* to set the respective limit of the work area for this joint in radians.
4. Save the changes.

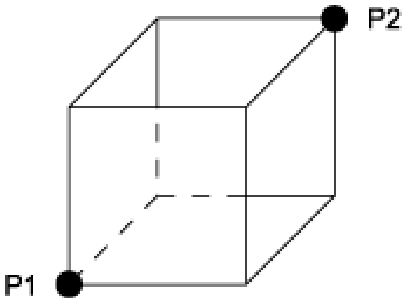
Related information

Upper Joint Bound on page 407.
Lower Joint Bound on page 408.
How to restrict the work area for parallel arm robots on page 385.

6.2.4 How to restrict the work area for parallel arm robots

Robot work area

The work area for a parallel arm robot is restricted by defining a cube in which the TCP0 is allowed to move.



en0500001489

P1	Lower work area x, y, z
P2	Upper work area x, y, z

The coordinates are defined in the base coordinate system and the work area is checked with respect to the predefined tool, tool0. It is not possible to check the position with respect to another tool.

To restrict the robot work area for parallel arm robots:

1. In the **Motion** topic, choose the type **Robot**.
2. Edit the parameters *Upper Work Area* and *Lower Work Area* for the coordinates x, y, and z.
3. Save the changes.

Note

The system parameters that define the work area for parallel robot are valid only for IRB 340 and IRB 360 robots.

Related information

Upper Work Area x, y, z on page 683.

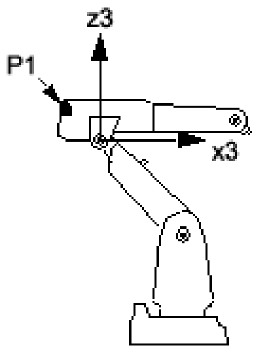
Lower Work Area x, y, z on page 684.

How to restrict the work area for articulated robots on page 384.

6.2.5 How to define arm check point

Arm check point

If an extra load, such as a transformer or a welding-bar roller, is attached to arm 3, a point on this equipment can be defined as a check point. The robot will then monitor the speed of this point so that it does not exceed 250 mm/s in manual reduced speed mode.



en0300000425

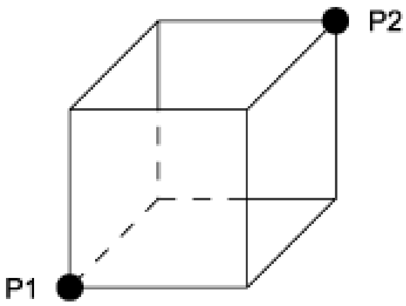
P1	Arm check point
z3	z-axis for arm 3
x3	x-axis for arm 3

Limitations

The value for the *Use Check Point* parameter must be identical to the name used for the arm check point.

Bound check point

The check point can also be restricted to stay outside a defined cube, when the robot is moving. The cube is defined by six coordinates, three upper and three lower, see illustration, all being related to the robot base coordinate system. Thus the defined cube will work as a stationary world zone, where the inside of the cube is the forbidden area for the arm check point.



en0500001489

P1	Lower check point bound x, y, z
P2	Upper check point bound x, y, z

Continues on next page

How to define arm check point

To define the arm check point:

1. In the **Motion** topic, choose the type **Arm Check Point**.
2. Edit the parameters for the check point.

   For detailed information, see the descriptions in the *Arm Check Point type* section.
3. Make a note of the *Name* parameter value to use later.
4. Save the changes.
5. In the topic **Motion**, choose the type **Arm**.
6. First select arm 3 to connect the check point to the arm. Then edit the parameter *Use Check Point*. The value has to be identical to the name used for the arm check point (step 2-3 above).

   For detailed information about the parameters, see sections *Arm* type and *Arm Check Point* type.
7. Save the changes.
8. To restrict the check point, choose the type **Robot** in the topic **Motion**.
9. Edit the parameters *Upper Check Point Bound* and *Lower Check Point Bound* for the six coordinates.

   For detailed information about the parameters, see section *Robot* type.
10. Save the changes.

Related information

The Arm type on page 404.

The Arm Check Point type on page 427.

Upper Check Point Bound x, y, z on page 687.

Lower Check Point Bound x, y, z on page 688.

The Product manual for the robot.

6.2.6 How to define arm loads

Arm load

The arm load is used for defining loads from equipment mounted on robot arms. If the arm load is not defined when equipment is mounted on the robot arms, the performance of the robot is negatively affected.

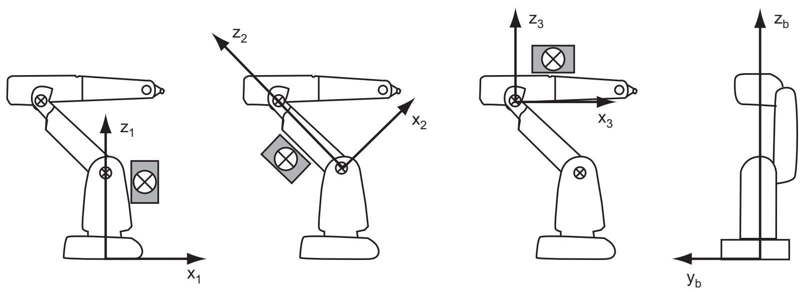
For more information about arm loads, see the type *Arm Load*.

Prerequisites

The mass, the mass center, and the moment of inertia of the load have to be measured or calculated before defining the arm load.

Arms for relating arm load to

The arm loads can be related to all arms of the robot. For the arms 1, 2, and 3, see the following illustration. Generally all loads are defined according to its joint intersection. The y coordinate is relative to the center of the robot base. The load for arm 4 is an exception and is defined according to the joint intersection for axis 3 in the synchronization position. The load for track motion is defined according to the robot base frame.



en0300000424

z_1, x_1	Arm 1
z_2, x_2	Arm 2
z_3, x_3	Arm 3
y, z	View from back, y_b z_b for the robot base

If more than one load is mounted on the same arm, the total weight and the center of gravity for the loads have to be calculated.

How to define an arm load

To define an arm load:

1. In the topic **Motion**, choose the type **Arm Load**.
2. Select the arm load to define, or create a new.
3. Enter or change the parameters of the arm load and save your changes. It is not necessary to restart the system at this point.

   For detailed information about each parameter, see the descriptions in the type *Arm Load*, *The Arm Load type on page 430*.

Continues on next page

4. In the topic **Motion**, choose the type **Arm** and select the arm that the load is mounted on.
5. For the selected arm, choose the *Use Arm Load* parameter and select the name of the arm load in the list of defined loads.
6. Save the changes and restart the system.

Related information

The Arm Load type on page 430.

The Arm type on page 404.

The service routine *LoadIdentify* is described in *Operating manual - IRC5 with FlexPendant*.

6.2.7 How to optimize drive system parameters

The drive system parameters

The drive system can be configured so that it corresponds to the robot's installation. The parameters related to the drive system are organized in two types.

To optimize the...	... use the parameters of the type
tolerance for the mains power supply	*Mains*
cable type and length	*Cable*

Default and optimal values

All drive system parameters have nominal values after installation. For improving the robot's performance, these parameters can be adjusted according to the robot's actual installation.

CAUTION

Parameter settings outside the range of the robot's installation may negatively affect the robot's performance.

How to optimize the mains tolerance

To optimize the tolerance for the mains power supply:

1. In the topic **Motion**, choose the type **Mains.**
2. Edit the *Mains Tolerance Min* parameter according to the robot's installation.

   For detailed information about each parameter, see the descriptions in the type *Mains*.
3. Save the changes.

Example to show how the mains tolerance can affect the robot performance

The systems with 220-230V single phase mains can be optimized using the mains tolerance. For example, for the IRB140T 6kg robot with the default settings 220V mains and mains tolerance min -0.15, the max speed for the corresponding joints become as shown in the following table.

Joint	Max speed Default settings	Max speed mains tolerance min = 0.0
1	229 deg/s	250 deg/s
2	228 deg/s	250 deg/s
3	245 deg/s	260 deg/s
4	348 deg/s	360 deg/s
5	360 deg/s	360 deg/s
6	450 deg/s	450 deg/s

Continues on next page

Setting the mains tolerance min to 0.0 means to have a mains of 220V single phase. At 230V this is equivalent to 230V -4.3%. For more detailed performance data, see the respective robot product specification.

CAUTION

Changing the mains tolerance min can create a situation where the system stops due to a too low DC-bus voltage, rectifier saturation, or some other error code. In this case the tolerance must be increased.

Related information

The Mains type on page 555.

6.2.8 How to tune motion supervision

Motion supervision

Motion supervision is functionality for collision detection with the option *Collision detection*.

How to tune the motion supervision

To tune the motion supervision:

1. In the **Motion** topic, choose the type **Motion Supervision**.
2. Decide which robot to tune the supervision for.
3. Edit the parameters for motion supervision. For detailed information about each parameter, see the descriptions in the type *Motion Supervision*.
4. Save the changes.

Related information

The Motion Supervision type on page 617.

Application manual - Controller software IRC5.

6.2.9 How to define transmission gear ratio for independent joints

Transmission gear ratio

An independent joint can rotate in one direction for a long time, resetting the measurement system regularly. A small round-off in the transmission gear ratio can build up to large errors over time. The transmission gear ratio must therefore be given as an exact fraction (e.g. 10/3 instead of 3.3333).

Define the transmission gear ratio by setting *Transmission Gear High* to the numerator and *Transmission Gear Low* to the denominator.

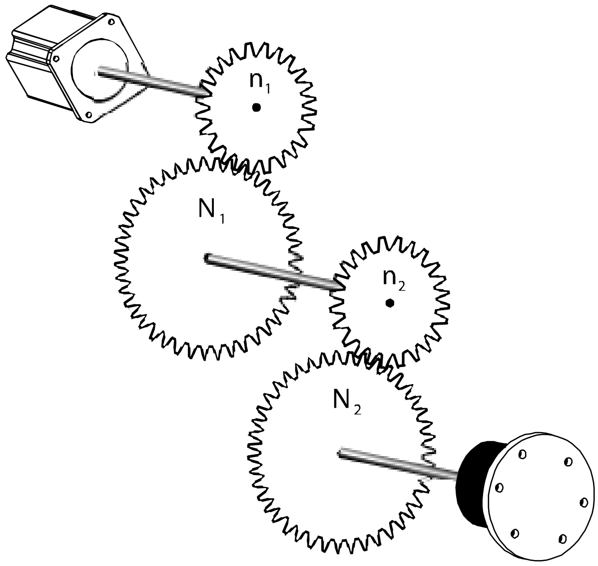
Limitations

The parameters *Transmission Gear High* and *Transmission Gear Low* are only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

How to calculate transmission gear ratio

If the proportions for the transmission gear ratio are complex, count the cogs to get the exact ratio.



xx0300000285

In the illustration, the total transmission gear ratio is:

$$\frac{N_1 \times N_2}{n_1 \times n_2}$$

xx0300000272

N_1, N_2, n_1 and n_2 represent the number of cogs on each gearwheel.

Continues on next page

To get an exact representation of the transmission gear ratio:

1. In the **Motion** topic, choose the type **Transmission**.
2. Decide which for joint to define the transmission gear ratio.
3. Set the parameter *Transmission Gear High* to the value N_1 x N_2.
4. Set the parameter *Transmission Gear Low* to the value n_1 x n_2.

Related information

The Transmission type on page 768.

Application manual - Controller software IRC5.

6.2.10 How to define external torque

External torque

When external equipment, for example a cable or a coiled hose, affects any joint significantly, the external torque should be defined using the following formula:

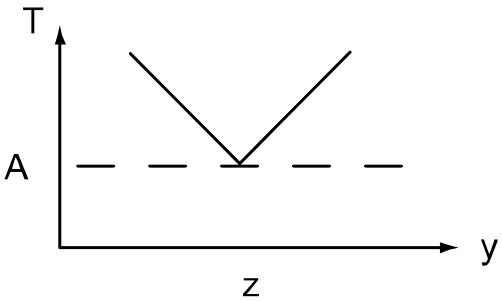
$T = A + |k \times (0 - \theta_0)|$

T = external torque [Nm]

A = constant torque [Nm]

k = scale factor for position dependent torque [Nm]

θ_0= joint position when position dependent torque is zero [rad]



xx0800000265

z	zero angle
y	joint position

If the estimated value of a significant external torque is too low, there can be unnecessary path deviations and the manipulator might be damaged. If the estimated value is too high, the performance of the manipulator is reduced due to restrictive acceleration limits.

How to define external torque

To define external torque:

1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to edit.
3. Set the desired values for the parameters *External Const Torque*, *External Proportional Torque*, and *External Torque Zero Angle*.
4. Save the changes.

Related information

The Arm type on page 404.

External Const Torque on page 417.

External Proportional Torque on page 420.

External Torque Zero Angle on page 421.

Continues on next page

Example

A coiled hose is mounted and affects joint 6 as follows:

0 Nm at 0 degrees.

5 Nm at 200 degrees.

This external torque can be defined using the following formula: $A = 0$, $\theta_0 = 0$, $k = 5 / (200 \times (pi / 180))$

6.2.11 How to define supervision level

Supervision level

It is possible to change the default supervision levels if a system needs to be more or less tolerant to external disturbances. A higher tune factor than 1.0 gives a more tolerant robot system, and vice versa. E.g. increasing the tune factor from 1.0 to 2.0, doubles the allowed supervision levels, which makes the robot system more tolerant to external disturbances.

Note

Increasing the tune factors can reduce the lifetime of the robot.

How to define the supervision level

To define the supervision level:

1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to change.
3. For the selected arm, set the desired values of the parameters *Jam Supervision Trim Factor*, *Load Supervision Trim Factor*, *Speed Supervision Trim Factor*, and *Position Supervision Trim Factor*.
4. Save the changes.

Related information

The Arm type on page 404.

Jam Supervision Trim Factor on page 413.

Load Supervision Trim Factor on page 414.

Speed Supervision Trim Factor on page 415.

Position Supervision Trim Factor on page 416.

6.3 Type Acceleration Data

6.3.1 The Acceleration Data type

Overview

This section describes the type *Acceleration Data*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

ACC_DATA

Type description

The type *Acceleration Data* is used to specify some acceleration characteristics for axes without any dynamic model. This is the case for certain additional axes.

For axes that have a dynamic model, *Acceleration Data* must still be specified even if a more complex model is normally used for the acceleration characteristics.

6.3.2 Name

Parent

Name belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

name

Description

The name of the set of *Acceleration Data*.

Usage

Name is used to reference a set of *Acceleration Data* from the parameter *Use Acceleration Data* in the type *Arm*.

Allowed values

A string with maximum 32 characters.

6.3.3 Nominal Acceleration

Parent

Nominal Acceleration belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_acc

Description

Worst case motor acceleration.

Usage

Set *Nominal Acceleration* to a value of the acceleration the axis can always perform (even when gravity and friction are unfavorable).

Nominal Acceleration is always used by axes without any dynamic model. For axes with dynamic model, it is only used in independent mode.

Allowed values

A numeric value between 0 and 1000, in rad/s^2 (or m/s^2) on the arm side.

6.3.4 Nominal Deceleration

Parent

Nominal Deceleration belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_dec

Description

Worst case motor deceleration.

Usage

Set *Nominal Deceleration* to a value of the deceleration the axis can always perform (even when gravity and friction are unfavorable).

Nominal Deceleration is always used by axes without any dynamic model. For axes with dynamic model, it is only used in independent mode.

Allowed values

A numeric value between 0 and 1000, in rad/s^2 (or m/s^2) on the arm side.

6.3.5 Acceleration Derivate Ratio

Parent

Acceleration Derivate Ratio belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_dacc_ratio

Description

Acceleration Derivate Ratio defines how fast the acceleration can build up, i.e. an indication of the derivative of the acceleration.

Usage

If the derivative of the acceleration is not limiting the acceleration, set *Acceleration Derivate Ratio* to 1. If the acceleration must be increased at a slower rate, set *Acceleration Derivate Ratio* to a ratio of the maximum acceleration derivative (e.g. 0.5 to increase the acceleration half as fast as possible).

Limitations

Acceleration Derivate Ratio is not used during independent joint motion.

Allowed values

A numeric value between 0.1 and 1. The value has no unit, but is a ratio of the maximum acceleration derivative.

The default value is 1.

6.3.6 Deceleration Derivate Ratio

Parent

Deceleration Derivate Ratio belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_ddec_ratio

Description

Deceleration Derivate Ratio defines how fast the deceleration can build up, i.e. an indication of the derivative of the deceleration.

Usage

If the derivative of the deceleration is not limiting the deceleration, set *Deceleration Derivate Ratio* to 1. If the deceleration must be increased at a slower rate, set *Deceleration Derivate Ratio* to a ratio of the maximum deceleration derivative (e.g. 0.5 to increase the deceleration half as fast as possible).

Limitations

Deceleration Derivate Ratio is not used during independent joint motion.

Allowed values

A numeric value between 0.1 and 1. The value has no unit, but is a ratio of the maximum deceleration derivative.

The default value is 1.

6.4 Type Arm

6.4.1 The Arm type

Overview

This section describes the type *Arm*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ARM

Type description

The *Arm* type contains a number of parameters that defines the characteristics for an arm. There is one set of parameters of the type *Arm* for each joint.

Related information

How to define supervision level on page 397.

How to define external torque on page 395.

6.4.2 Name

Parent

Name belongs to the type *Arm*, in the topic *Motion*.

Cfg name

Name

Description

Name defines the name of the set of parameters for type *Arm*.

Allowed values

A string with maximum 32 characters.

6.4.3 Independent Joint

Parent

Independent Joint belongs to the type *Arm*, in the topic *Motion*.

Cfg name

independent_joint_on

Description

Independent Joint is a flag for each axis that indicates whether the axis can be changed to independent mode.

Usage

Normally, all external axes and robot axis 6 allow independent mode. To prevent one of these axes moving independently, set *Independent Joint* to Off for that axis.

Limitations

Independent Joint is only useful if you have the RobotWare option *Independent Axes*.

Allowed values

On or Off.

Related information

Application manual - Controller software IRC5

6.4.4 Upper Joint Bound

Parent

Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

upper_joint_bound

Description

Upper Joint Bound defines the upper limit of the working area for this joint.

Usage

Upper Joint Bound can be used to limit the working area (in radians) of the joint.

Note that it is not possible to use a value that is larger than the maximal allowed limit for the specific joint. Trying this will cause the system to use the maximal allowed value instead.

Limitations

This parameter is valid only for articulated robots.

Allowed values

A value between -1,256,637 and 1,256,637 radians.

Related information

Lower Joint Bound on page 408.

How to restrict the work area for articulated robots on page 384.

6.4.5 Lower Joint Bound

Parent

Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

lower_joint_bound

Description

Lower Joint Bound defines the lower limit of the working area for this joint.

Usage

Lower Joint Bound can be used to limit the working area (in radians) of the joint.

Note that it is not possible to use a value that is smaller than the minimal allowed limit for the specific joint. Trying this will cause the system to use the minimal allowed value instead.

Limitations

This parameter is valid only for articulated robots.

Allowed values

A value between -1,256,637 and 1,256,637 radians.

Related information

Upper Joint Bound on page 407.

How to restrict the work area for articulated robots on page 384.

6.4.6 Independent Upper Joint Bound

Parent

Independent Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ind_upper_joint_bound

Description

Defines the upper limit of the working area for the joint when operating in independent mode.

Usage

Independent Upper Joint Bound is used together with *Independent Lower Joint Bound* to limit the work area for a joint that is in independent mode.

Limitations

Independent Upper Joint Bound is only useful if you have the option *Independent Axes*.

Allowed values

Any number (in radians).

Related information

Application manual - Controller software IRC5

6.4.7 Independent Lower Joint Bound

Parent

Independent Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ind_lower_joint_bound

Description

Defines the lower limit of the working area for the joint when operating in independent mode.

Usage

Independent Lower Joint Bound is used together with *Independent Upper Joint Bound* to limit the work area for a joint that is in independent mode.

Limitations

Independent Lower Joint Bound is only useful if you have the option *Independent Axes*.

Allowed values

Any number (in radians).

Related information

Application manual - Controller software IRC5

6.4.8 Calibration Position

Parent

Calibration Position belongs to the type *Arm*, in the topic *Motion*.

Cfg name

cal_position

Description

Calibration Position defines the position of the axis when it was fine calibrated.

Usage

If this value is to be updated, i.e. a fine calibration is to be performed, use *Calibration pendulum* to achieve the correct kinematic position of the axis and then fine calibrate the axis. It is then necessary to subsequently fine calibrate axes of higher order.

Allowed values

A value between -1000 and 1000, specifying the position in radians.

Related information

Product Manual for the manipulator.

Operating manual - Calibration Pendulum.

6.4.9 Performance Quota

Parent

Performance Quota belongs to the type *Arm*, in the topic *Motion*.

Cfg name

performance_quota

Description

Performance Quota can be used to reduce the acceleration for the joint.

Usage

Setting *Performance Quota* value to 1.0 gives normal performance, but if less acceleration is desired, a lower value can be entered.

Allowed values

A number between 0.15 and 1.0.

6.4.10 Jam Supervision Trim Factor

Parent

Jam Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_jam_time_factor

Description

Jam Supervision Trim Factor defines the tune factor for jam supervision.

Usage

The tune factor influences the maximum time allowed at zero speed with maximum torque.

Allowed values

A number between 0.1 and 10.0.

Related information

How to define supervision level on page 397

6.4.11 Load Supervision Trim Factor

Parent

Load Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_load_factor

Description

Load Supervision Trim Factor defines the tune factor for load supervision.

Usage

The factor influences the maximum time allowed at non-zero speed with maximum torque.

Allowed values

A number between 0.1 and 10.0.

Related information

How to define supervision level on page 397.

6.4.12 Speed Supervision Trim Factor

Parent

Speed Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_speed_factor

Description

Speed Supervision Trim Factor defines the tune factor for speed supervision.

Usage

The factor influences the maximum allowed speed error.

Allowed values

A number between 0.05 and 10.0.

Related information

How to define supervision level on page 397.

6.4.13 Position Supervision Trim Factor

Parent

Position Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_pos_factor

Description

Position Supervision Trim Factor defines the tune factor for position supervision.

Usage

The factor influences the maximum allowed position error.

Allowed values

A number between 0.1 and 10.0.

Related information

How to define supervision level on page 397.

6.4.14 External Const Torque

Parent

External Const Torque belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_const_torque

Description

External Const Torque defines the external constant torque.

Usage

The value of *External Const Torque* is used in the formula for calculation of external torque.

Allowed values

A value between 0 and 100,000, specifying the constant torque in Nm.

Related information

How to define external torque on page 395.

6.4.15 Use Arm Load

Parent

Use Arm Load belongs to the type *Arm*, in the topic *Motion*.

Cfg name

use_customer_arm_load

Description

Use Arm Load defines the name of the arm load that is used for this arm.

Usage

The arm load is set in the type *Arm Load*.

Allowed values

A string with maximum 32 characters, defining an *Arm Load* type.

Related information

The Arm Load type on page 430.

6.4.16 Use Check Point

Parent

Use Check Point belongs to the type *Arm*, in the topic *Motion*.

Cfg name

use_check_point

Description

Use Check Point determines which *Arm Check Point* that should be used.

Usage

Use Check Point is a reference to the parameter *Name* in the type *Arm Check Point*.

Prerequisites

An *Arm Check Point* must be configured before *Use Check Point* can refer to it.

Limitations

Use Check Point can only be used for articulated robots.

Allowed values

A string with maximum 32 characters.

Related information

Type Arm Check Point on page 427.

6.4.17 External Proportional Torque

Parent

External Proportional Torque belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_prop_torque

Description

External Proportional Torque defines the scale factor for position-dependent torque.

Usage

The value of *External Proportional Torque* is used in the formula for calculation of external torque.

Allowed values

A value between -100,000 and 100,000, specifying the scale factor in Nm/rad.

Related information

How to define external torque on page 395.

6.4.18 External Torque Zero Angle

Parent

External Torque Zero Angle belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_prop_zero_angle

Description

External Torque Zero Angle defines the joint position when position-dependent torque is zero.

Usage

The value of *External Torque Zero Angle* is used in the formula for calculation of external torque.

Allowed values

A value between -100,000 and 100,000, specifying the position in radians.

Related information

How to define external torque on page 395.

6.4.19 Load Id Acceleration Ratio

Parent

Load Id Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

Cfg name

load_id_acc_ratio

Description

Load Id Acceleration Ratio can be used to reduce the acceleration of the joint during load identification.

Usage

Reducing the acceleration of the joint during load identification can be useful if the torque supervision is triggered when identifying payloads with large inertia. If this happens, try to reduce the value of *Load Id Acceleration Ratio* until the problem disappears.

Allowed values

A number between 0.2 and 1.0.

6.4.20 Angle Acceleration Ratio

Parent

Angle Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

Cfg name

angle_acc_ratio

Description

Angle Acceleration Ratio defines the maximum angle acceleration ratio for the motor sensor.

Usage

This parameter should only be changed by ABB.

Allowed values

A value between 0.02 and 1.0. Default value is 1.0.

6.4.21 Deactivate Cyclic Brake Check for axis

Parent

Deactivate Cyclic Brake Check for axis belongs to the type *Arm*, in the topic *Motion*.

Cfg name

deactivate_cyclic_brake_check_arm

Description

Deactivate Cyclic Brake Check for axis defines if the arm should be excluded from the SafeMove function Cyclic Brake Check.

Usage

If an axis should be excluded from Cyclic Brake Check, set the parameter *Deactivate Cyclic Brake Check for axis* to On.

The axis must also be deactivated in the configuration of Cyclic Brake Check. See *Application manual - SafeMove*.

Allowed values

On or Off

On means that the Cyclic Brake Check is deactivated for the axis.

Default value is Off.

Related information

Application manual - SafeMove.

6.4.22 Change to Logical Axis

Parent

Change to Logical Axis belongs to the type *Arm*, in the topic *Motion*.

Cfg name

change_to_logical_axis

Description

The parameter *Change to Logical Axis* can be used to change the Logical Axis in the type Joint if it is read only. This is normally the case for ABB Positioners (IRBP) and the ABB Tracks (IRBT). If the value is zero, then no change will happen and the value in the Joint will be used as normal.

Usage

The value of Logical Axis is used by RAPID programs to identify individual axes in mechanical units.

Two mechanical units can have the same value set for Logical Axis, but then they cannot be activated at the same time by a RAPID program.

Robots from ABB normally use the values from 1 to 6, while additional axes use from 7 to 12.

Limitations

This parameter cannot be used for robots from ABB.

Allowed values

A value from 0 to 12.

Default value is 0.

Related information

Application manual - Additional axes and stand alone controller.

Logical Axis on page 518.

6.4.23 Thermal Supervision Sensitivity Ratio

Parent

Thermal Supervision Sensitivity Ratio belongs to the type *Arm*, in the topic *Motion*.

Cfg name

thermal_supervision_sensitivity_ratio

Description

The parameter *Thermal Supervision Sensitivity Ratio* can be used for installation adjustment parameter (0.5 = approximate disconnected supervision)

Usage

If the error occurs, in spite of cold motor due to extra cooling or low ambient temperature, the sensitivity of the thermal supervision can be reduced. Decrease the system parameter *Thermal Supervision Sensitivity Ratio* in steps of 0.1. Check the motor temperature during and after tuning.

Allowed values

A value from 0.5 to 2.0.

Default value is 1.0.

Note

With too low value the supervision is deactivated and the motor can be overheated and damaged.

6.5 Type Arm Check Point

6.5.1 The Arm Check Point type

Overview

This section describes the type *Arm Check Point*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic.

Cfg name

ARM_CHECK_POINT

Type description

If an extra load, such as a transformer or a welding-bar roller, is attached to arm 3, a point on this equipment can be defined as a check point. The robot will then monitor the speed of this point so that it does not exceed 250 mm/s in manual reduced speed mode.

Related information

How to define arm check point on page 386.

Check Point Bound Limit Outside Cube on page 686

6.5.2 Name

Parent

Name belongs to the type *Arm Check Point*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the arm check point. A check point can be used to let the robot monitor the speed of that specified point

Allowed values

A string with maximum 24 characters.

Related information

How to define arm check point on page 386.

6.5.3 Position x, y, z

Parent

Position x, *Position y*, and *Position z* belongs to the type *Arm Check Point*, in the topic *Motion*.

Cfg names

position_x

position_y

position_z

Description

Position x defines the x-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Position y defines the y-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Position z defines the z-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Allowed values

A value between -3 to 3, specifying the position in meters.

Related information

How to define arm check point on page 386.

6.6 Type Arm Load

6.6.1 The Arm Load type

Overview

This section describes the type *Arm Load*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

ARM_LOAD

Type description

Arm Load is used for defining loads from equipment mounted on robot arms. If the arm load is not defined when equipment is mounted on the robot arm, the performance of the robot is negatively affected.

The *Arm* configuration defines which *Arm Load* to use for the arm.

Predefined arm loads

There are four predefined arm loads in the robot controller. They are r1_load_1, r1_load_2, r1_load_3, and r1_load_4. For track motion, the predefined arm load in the robot controller is t1_load_1. The predefined arm loads must be adjusted to match the load and selected for the arm that it belongs to before use.

Related information

How to define arm loads on page 388.

6.6.2 Name

Parent

Name belongs to the type *Arm Load*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the arm load setting it belongs to.

Allowed values

A string with maximum 32 characters, specifying the name.

Related information

How to define arm loads on page 388.

6.6.3 Mass

Parent

Mass belongs to the type *Arm Load*, in the topic *Motion*.

Cfg name

mass

Description

Mass specifies the mass of the equipment mounted on a robot arm.

Allowed values

A value between 0 and 500, specifying the weight in kg.

Related information

How to define arm loads on page 388.

6.6.4 Mass Center x, y, z

Parent

Mass Center x, *Mass Center y*, and *Mass Center z* belongs to the type *Arm Load*, in the topic *Motion*.

Cfg names

mass_centre_x

mass_centre_y

mass_centre_z

Description

Mass Center x specifies the x-coordinate of the mass center for an arm load in the arm frame.

Mass Center y specifies the y-coordinate of the mass center for an arm load in the arm frame.

Mass Center z specifies the z-coordinate of the mass center for an arm load in the arm frame.

Allowed values

A value between -3 and + 3, specifying the coordinate in meters.

Related information

How to define arm loads on page 388.

6.6.5 Inertia x, y, z

Parent

Inertia x, *Inertia y*, and *Inertia z* belongs to the type *Arm Load*, in the topic *Motion*.

Cfg names

inertia_x

inertia_y

inertia_z

Description

Inertia x defines the x-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Inertia y defines the y-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Inertia z defines the z-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Allowed values

A value between 0 and 100, specifying the moment of inertia in kgm^2.

Related information

How to define arm loads on page 388.

6.7 Type Brake

6.7.1 The Brake type

Overview

This section describes the type *Brake* which belongs to the topic *Motion*.

Cfg name

BRAKE

Type description

The type *Brake* is used to specify brake parameters for a specific joint.

Related information

The Joint type on page 516.

6.7.2 Name

Parent

Name belongs to the type *Brake*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the brake.

Allowed values

A string with maximum 32 characters.

6.7.3 Control Off Speed Limit

Parent

Control Off Speed Limit belongs to the type *Brake*, in the topic *Motion*.

Cfg name

control_off_speed_limit

Description

Control Off Speed Limit defines the speed for selection of delay time.

Usage

The value for *Control Off Speed Limit* should not be modified.

Allowed values

A value between 0 and 1.

Default value is 0.02.

6.7.4 Control Off Delay

Parent

Control Off Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

control_off_delay_time

Description

Control Off Delay specifies the time of normal control before the motor torque is set to zero.

Usage

Control Off Delay is used when the joint is at zero speed when the brake algorithm is activated. The controller must be active to avoid the joint to fall by gravity before the mechanical brake is engaged.

Time must be longer than the time for mechanical brake to engage.

Allowed values

A value between 0 and 30 seconds.

Default value is 0.010 seconds.

6.7.5 Brake Control On Delay

Parent

Brake Control On Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_control_on_delay_time

Description

Brake Control On Delay specifies the time of normal control before the motor torque is set to zero.

Usage

Brake Control On Delay is used if the joint is moving when the brake algorithm is activated. The controller must be active to avoid oscillations when the mechanical brake is engaged.

The time must be longer than the time for mechanical brake to engage. Normally set to same value as parameter *Control Off Delay*.

Allowed values

A value between 0 and 30 seconds.

Default value is 0.

Related information

Control Off Delay on page 438.

6.7.6 Brake Control Min Delay

Parent

Brake Control Min Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_control_on_min_delay_time

Description

Brake Control Min Delay defines the minimum delay time.

Usage

Brake Control Min Delay should not be changed.

Allowed values

A value between 0 and 5 seconds.

Default value is 0.010.

6.7.7 Absolute Brake Torque

Parent

Absolute Brake Torque belongs to the type *Brake*, in the topic *Motion*.

Cfg name

absolute_brake_torque

Description

Absolute Brake Torque defines the brake torque to be used for a simulated electrical brake.

Usage

Absolute Brake Torque should not be changed.

Allowed values

A value between 0 and 100,000 Nm.

Default value is 0.

6.7.8 Brake Ramp Speed Limit

Parent

Brake Ramp Speed Limit belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_ramp_speed_limit

Description

Brake Ramp Speed Limit is the point of torque reduction for simulated electrical brake.

Usage

Brake Ramp Speed Limit should not be changed.

Allowed values

A value between 0 and 1.

Default value is 1 (equal to 100%).

6.7.9 Max Brake Time

Parent

Max Brake Time belongs to the type *Brake*, in the topic *Motion*.

Cfg name

max_brake_time

Description

A time-out occurs if a large additional axis use the motor to brake during emergency stop and the stop time exceeds the default value of 5 seconds. The time-out results in stopping all the drive units and the brake torque from the motors are set to zero torque. A warning message is generated. By increasing the *Max Brake Time*, the servo motors help the axes to retardate down to zero speed during the whole brake sequence.

Usage

Measure or calculate the maximum brake time for the axis (including safety margin). If the default value of 5 seconds is exceeded, change the parameter to appropriate value.

Allowed values

Min 1 s

Max 60 s

Default value

The default value is 5 s.

6.8 Type Control Parameters

6.8.1 The Control Parameters type

Overview

This section describes the type *Control Parameters*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CONTROL_PARAMETERS

Type description

Each set of parameters of the type *Control Parameters* belongs to a joint (robot joint or additional axis).

The parameters in *Control Parameters* define what compensations should be made for the friction in the joint.

Limitation

Changing the parameter values in *Control Parameters* is only useful if you have the RobotWare option *Advanced Shape Tuning*.

The type *Control Parameters* is only used by robot models IRB 1400 and IRB 1410. All other robot models use the type *Friction Compensation* instead. The parameters are the same however.

Related information

Application manual - Controller software IRC5, chapter *Advanced Shape Tuning*.

6.8.2 Name

Parent

Name belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name to use for the control parameters.

Limitations

Name is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A string with maximum 32 characters.

6.8.3 Friction FFW On

Parent

Friction FFW On belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_on

Description

Friction FFW On determines if the RobotWare option *Advanced Shape Tuning* is active or not.

Usage

Set *Friction FFW On* to Yes if you want to use *Advanced Shape Tuning*.

Limitations

Friction FFW On is useful only if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

Yes or No.

Related information

Application manual - Controller software IRC5

6.8.4 Friction FFW Level

Parent

Friction FFW Level belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_level

Description

Friction FFW Level is set to the level of friction in the robot axis. By setting a value that closely corresponds to the real friction, and using the RobotWare option *Advanced Shape Tuning*, the friction effects can be compensated.

Usage

Friction effects can cause path deviations when performing advanced shapes. By compensating for the friction with the correct friction level value, these effects can be minimized.

Permanent adjustments of the friction level can be made with *Friction FFW Level*. The friction level can also be temporarily tuned with RAPID commands.

Limitations

Friction FFW Level is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A decimal number between 0 and 15 (in Nm).

Related information

Application manual - Controller software IRC5

6.8.5 Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_ramp

Description

Friction FFW Ramp is set to the speed of the robot axis when the friction has reached the constant friction level defined in *Friction FFW Level*. See illustration below.

Usage

Friction effects can cause path deviations when performing advanced shapes. *Friction FFW Ramp* is used when compensating for these friction effects.

Permanent adjustments of the friction ramp can be made with *Friction FFW Ramp*. The friction ramp can also be temporarily tuned with RAPID commands.

Limitations

Friction FFW Ramp is only useful if you have the RobotWare option *Advanced Shape Tuning*.

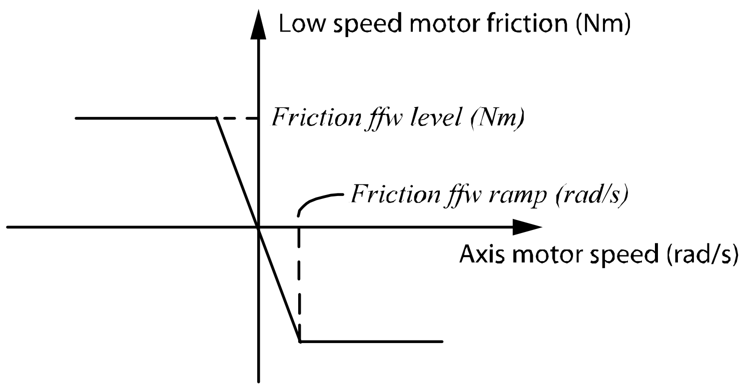
Allowed values

A number between 0.001 and 10 (in radians/second).

Related information

Application manual - Controller software IRC5

Illustration



en0300000278

6.9 Type Drive Module

6.9.1 The Drive Module type

Overview

This section describes the type *Drive Module*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DRIVE_MODULE

Type description

The type *Drive Module* is used to identify and specify each drive module used in the robot system. There is one set of parameters of the type *Drive Module* for each drive module in the robot system.

Limitations

If the robot system does not use *MultiMove*, there is only one drive module, and therefore only set of parameters of the type *Drive Module*.

6.9.2 Name

Parent

Name belongs to the type *Drive Module*, in the topic *Motion*.

Cfg name

name

Description

Defines the unique name of the drive module.

Allowed values

A string with maximum 32 characters.

6.9.3 Number

Parent

Number belongs to the type *Drive Module*, in the topic *Motion*.

Cfg name

number

Description

Defines the identifying number of the drive module.

Usage

The drive module number is used to identify the drive module by other system parameters.

Allowed values

An integer between 1 and 4.

6.10 Type Drive System

6.10.1 The Drive System type

Overview

This section describes the type *Drive System*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DRIVE_SYSTEM

Type description

The type *Drive System* is used to identify and specify each drive system used in the robot system.

6.10.2 Name

Parent

Name belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

name

Description

Defines the name for the drive system.

Allowed values

A string with maximum 32 characters.

6.10.3 Use DC-Link

Parent

Use DC-Link belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

use_dc_link

Description

Use DC-Link determines which dc-link (rectifier) unit should be used.

Allowed values

A string with maximum 32 characters.

6.10.4 Use Drive Unit

Parent

Use Drive Unit belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

use_drive_unit

Description

Use Drive Unit determines which drive unit should be used.

Allowed values

A string with maximum 32 characters.

Related information

The Drive Unit type on page 457.

Application manual - Additional axes and stand alone controller.

6.10.5 Current Vector On

Parent

Current Vector On belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

current_vector_on

Description

Current Vector On defines if the vector control is active.

Usage

Current Vector On controls an activation switch. It is used to prevent that an axis with uncommutated motor runs away at start.

The parameter is reset by the service routine COMMUTATION, or manually via RobotStudio or FlexPendant.

Allowed values

Yes

No

Default value is No.

Related information

Application manual - Additional axes and stand alone controller, section *Tuning*.

6.11 Type Drive Unit

6.11.1 The Drive Unit type

Overview

This section describes the type *Drive Unit*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DRIVE_UNIT

Type description

The type *Drive Unit* is used to identify and specify each drive unit used in the robot system.

Additional information

The Drive System type on page 452.

6.11.2 Name

Parent

Name belongs to the type *Drive Unit*, in the topic *Motion*.

Cfg name

name

Description

Defines the name for the drive unit.

Allowed values

A string with maximum 32 characters.

6.11.3 Drive Unit Position

Parent

Drive Unit Position belongs to the type *Drive Unit*, in the topic *Motion*.

Cfg name

unit_position

Description

Drive Unit Position defines the logical position on the Drive Unit network, starting with 1, then 2, 3, and so on.

Allowed values

A value between 1 and 5.

6.12 Type External Motion Interface Data

6.12.1 The External Motion Interface Data type

Overview

This section describes the type *External Motion Interface Data*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

EXT_MOTION_DATA

Type description

The *External Motion Interface Data* type contains a number of parameters that defines the characteristics for an *External Motion Interface Data*.

6.12.2 Name

Parent

Name belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

Name

Description

The name of the *External Motion Interface Data*.

Usage

This is the public identity of the *External Motion Interface Data*.

The parameter does not require a restart of the controller when modified.

Allowed values

A string with maximum 32 characters.

6.12.3 Level

Parent

Level belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

ext_motion_level

Description

External Motion Interface Level determines the system level at which the corrections are applied.

Usage

Level can have the following values:

Value:	Name	Description:
0	Raw	Corresponds to raw corrections, added just before the servo controllers
1	Filtering	Applies extra filtering on the correction, but also introduces some extra delays and latency
2	Path	Applies path corrections.

The parameter does not require a restart of the controller when modified.

Limitation

When using *Level 0*, low-pass filtering is necessary to avoid vibrations in the robot.

Allowed values

Allowed values are level *0*, *1* or *2*

The default value is 1.

6.12.4 Do Not Restart After Motors Off

Parent

Do Not Restart After Motors Off belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

do_not_restart_after_motors_off

Description

Do Not Restart After Motors Off determines if the *External Motion Interface* execution should automatically restart after the controller has been in the Motors Off state, for instance after emergency stop.

Usage

If *False* (default), execution of the corrections will continue in the same state as when the system entered the Motors Off state.

If *True*, execution will continue with all corrections in the STANDBY state.

Allowed values

True or False.

6.12.5 Return to Programmed Position when Stopped

Parent

Return to Programmed Position when Stopped belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

return_to_prog_pos_at_program_stop

Description

Return to Programmed Position when Stopped determines if axes currently running *External Motion Interface* should return to the programmed position, when program execution is stopped.

Usage

If *False* (default), axes will stop in their current position.

If *True*, axes will move to the programmed start position.

Limitation

The motion returning the axes to the programmed position will be defined in joint space. If the axes are far from the programmed position when *Return to Programmed Position when Stopped* is defined as *False*, unexpected trajectories may result. Therefore, it is recommended only to set this value to *False*, if the distance from the programmed position to the corrected position is known to be small.

Allowed values

True or False.

6.12.6 Default Ramp Time

Parent

Default Ramp Time belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

ramp_time

Description

Default Ramp Time defines the default total time for stopping *External Motion Interface* movements when *External Motion Interface* execution is stopped.

Usage

The value will be used to determine how fast the speed contribution from *External Motion Interface* should be ramped to zero when program execution is stopped, and how fast axes return to the programmed position if the *Return to Programmed Position when Stopped* is *True*.

Limitation

The value only affects the part of the motion that is generated by the *External Motion Interface* execution. It does not affect any simultaneous movements that have, for instance, been programmed in RAPID.

Allowed values

A value between 0.0 s and 10.0 seconds

The default value is 1.0 seconds.

6.12.7 Default Proportional Position Gain

Parent

Default Proportional Position Gain belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

ext_motion_Kp

Description

Default Proportional Position Gain defines the default proportional gain of the *External Motion Interface* position feedback control.

Allowed values

A value between 0.0 and 20.0.

The default value is 5.0.

6.12.8 Default Low Pass Filter Bandwidth

Parent

Default Low Pass Filter Bandwidth belongs to the type *External Motion Interface Data*, in the topic *Motion*.

Cfg name

ext_motion_filter_bandwidth

Description

Default Low Pass Filter Bandwidth Time defines the default bandwidth of the low-pass filter used to filter the speed contribution from the *External Motion Interface* execution.

Allowed values

A value between 0.0 and 100.0 Hz.

The default value is 20.0 Hz.

6.13 Type Force Master

6.13.1 The Force Master type

Overview

This section describes the type *Force Master*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

FORCE_MASTER

Type description

Force Master is used to define how a servo gun behaves during the two faces of the gun closing:

- when approaching the point where position regulation is replaced by force control
- during force control.

Values for position, torque, force, etc. are specified for calibration and gun closing.

Limitations

Force Master can only be used for servo tools.

Non-editable parameters

The following parameters are visible but not editable in the software configuration tools:

- *Force Detection Speed*
- *Max Pos Err Closing*

As a consequence, the above parameters are not described in the manual.

Related information

Application manual - Controller software IRC5

6.13.2 Name

Parent

Name belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Force Master*.

Usage

Name is used to reference a *Force Master* from the parameter *Use Force Master* in the type *SG Process*.

Allowed values

A string with maximum 32 characters.

6.13.3 Use Force Master Control

Parent

Use Force Master Control belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

use_force_master_control

Description

Use Force Master Control determines which *Force Master Control* should be used.

Usage

Use Force Master Control is a reference to the parameter *Name* in the type *Force Master Control*.

Prerequisites

A *Force Master Control* must be configured before *Use Force Master Control* can refer to it.

Limitations

Use Force Master Control can only be used for servo tools.

Allowed values

A string with maximum 32 characters.

Related information

The Force Master Control type on page 483.

6.13.4 References Bandwidth

Parent

References Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

bandwidth_ramping

Description

The frequency limit for the low pass filter for reference values. During position regulation, when approaching the plate thickness, position and speed values will be filtered in this low pass filter to avoid sharp step functions.

Usage

A high value on *References Bandwidth* will make little use of the low pass filter.

If the servo tool is vibrating due to irregular movements, *References Bandwidth* can be set to a lower value. A low value will make the servo tool movements slower.

Limitations

References Bandwidth can only be used for servo tools.

Allowed values

A numeric value between 1 and 124 (Hz).

The default value is 25 Hz.

6.13.5 Use Ramp Time

Parent

Use Ramp Time belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_time_switch

Description

Determines if the ramping of the tip force should use a constant time or a constant gradient.

Usage

If the tip force should be ramped up to its ordered value during the time specified in *Ramp Time*, set *Use Ramp Time* to Yes. The ramp rate will then vary to make the ramp time constant.

If the tip force should be increased at a constant rate, specified in *Ramp when Increasing Force*, set *Use Ramp Time* to No. The ramp time will then vary to make the ramp rate constant.

Limitations

Use Ramp Time can only be used for servo tools.

Allowed values

Yes or No.

6.13.6 Ramp when Increasing Force

Parent

Ramp when Increasing Force belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_torque_ref_closing

Description

Ramp when Increasing Force decides how fast the torque is ramped up to the ordered torque after contact position is reached at a close gun command.

Usage

A higher value of *Ramp when Increasing Force* will make the tip force build up faster.

Prerequisites

Ramp when Increasing Force is only used if *Use Ramp Time* is set to No.

Limitations

Ramp when Increasing Force can only be used for servo tools.

Allowed values

A value between 1 and 10000, specifying the torque increase in Nm/s.

The default value is 100 Nm/s.

6.13.7 Ramp Time

Parent

Ramp Time belongs to the type *Force Control*, in the topic *Motion*.

Cfg name

ramp_time

Description

Ramp Time decides how fast the torque is ramped up to the ordered torque after contact position is reached at a close gun command.

Usage

A lower value of *Ramp Time* will make the tip force build up faster.

Prerequisites

Ramp Time is only used if *Use Ramp Time* is set to Yes.

Limitations

Ramp Time can only be used for servo tools.

Allowed values

A numeric value between 0.001 and 1 (seconds).

The default value is 0.07 s.

6.13.8 Collision LP Bandwidth

Parent

Collision LP Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

bandwidth_lp

Description

Frequency limit for the low pass filter used for tip wear calibration. Position and speed reference values will be filtered in this low pass filter to avoid sharp step functions.

Usage

The only reason for changing *Collision LP Bandwidth* is if repetitive tip wear calibrations give varying results. A lower value for the low pass filter can stabilize the servo tool during the calibration.

Limitations

Collision LP Bandwidth can only be used for servo tools.

Allowed values

A numeric value between 0 and 124 (Hz).

The default value is 25 Hz.

6.13.9 Collision Alarm Torque

Parent

*Collision Alarm Torque*belongs to the type*Force Master*, in the topic *Motion*.

Cfg name

alarm_torque

Description

Collision Alarm Torque determines how hard the tool tips will be pressed together during the first gun closing of new tips calibrations and tool change calibrations.

Usage

Collision Alarm Torque is used for the first gun closing of new tips calibrations and tool change calibrations. This affects the position calibration.

The best way to determine the collision position (where the tool tips meet) is to keep closing the gun until the motor torque reaches the value specified in *Collision Alarm Torque*. The distance the gun then has moved beyond the collision position is defined by the parameter *Collision Delta Position*.

Limitations

Collision Alarm Torque can only be used for servo tools.

Allowed values

A value between 0 and 50 (Nm).

The default value is 1.5 Nm.

6.13.10 Collision Speed

Parent

Collision Speed belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

col_speed

Description

Collision Speed determines the servo gun speed during the first gun closing of new tips calibrations and tool change calibrations. These calibrations affect the position calibration.

Usage

The only reason for changing *Collision Speed* is if repetitive tip wear calibrations give varying results. A lower speed can improve the repeatability.

Limitations

Collision Speed can only be used for servo tools.

Allowed values

A value between 0 and 5 (m/s).

The default value is 0.02 m/s.

6.13.11 Collision Delta Position

Parent

Collision Delta Position belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

distance_to_contact_position

Description

Collision Delta Position defines the distance the servo tool has gone beyond the contact position when the motor torque has reached the value specified in *Collision Alarm Torque*.

Usage

Collision Delta Position is used for the first gun closing of new tips calibrations and tool change calibrations. This affects the position calibration.

The best way to determine the collision position (where the tool tips meet) is to keep closing the gun until the motor torque reach the value specified in *Collision Alarm Torque*. The distance the gun then has moved beyond the collision position is defined in *Collision Delta Position*.

Changing the value of *Collision Delta Position* can remove a constant calibration error, but does not affect if repetitive tip wear calibrations give varying results.

Limitations

Collision Delta Position can only be used for servo tools.

Allowed values

A value between 0 and 1 (m).

The default value is 0.0019 m.

6.13.12 Force Detection Bandwidth

Parent

Force Detection Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

force_ready_detection_bandwidth

Description

Defines the bandwidth for the force detection filter.

Usage

The force detection filter is used to filter the speed of the servo tool. The filtered speed is used to detect if the ordered force has been reached.

Limitations

Force Detection Bandwidth can only be used for servo tools.

Allowed values

A value between 1 and 124 Hz.

6.13.13 Delay Ramp

Parent

Delay Ramp belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

delay_ramp

Description

Delays the starting of torque ramp when force control is started.

Usage

Delay Ramp can be used to give the servo gun some time to stabilize before the force control starts. A higher value of *Delay Ramp* can result in better accuracy of the squeeze force but will increase the cycle time.

Limitations

Delay Ramp can only be used for servo tools.

Allowed values

A numeric value between 0 and 1 (seconds).

6.13.14 Ramp to Real Contact

Parent

Ramp to Real Contact belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_to_real_contact

Description

Determines if the feedback position should be used instead of reference position when deciding the contact position.

Usage

Setting *Ramp to Real Contact* to Yes will make the detection of the contact position (where the force control starts) more exact and improve the accuracy of the squeeze force, but increase the cycle time.

Limitations

Ramp to Real Contact can only be used for servo tools.

Allowed values

Yes or No.

6.13.15 Force Detection Min Time

Parent

Force Detection Min Time belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

force_ready_detection_min_time

Description

Defines the time in the start before the condition of force ready will be evaluated.

Usage

Filtered speed is used to detect if the ordered force has been reached. If the gun seems to weld before force is built up, likely due to high friction, it can be a false trigger of low speed in the initial ramp.

This value can in those cases be increased.

Limitations

Force Detection Min Time is only used for servo tools.

Allowed values

An value between 0 and 1 second.

Default value is 0.060 seconds.

Related information

Application manual - Servo gun tuning

6.14 Type Force Master Control

6.14.1 The Force Master Control type

Overview

This section describes the type *Force Master Control*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

FORCE_MASTER_CONTROL

Type description

Force Master Control is used to prevent a servo tool from closing with too high a speed.

If a servo tool is not completely closed when the force control starts, it can gain too much speed, which can cause damages when contact is reached. This can happen if the programmed thickness is too high, or if the servo tool tips are not properly calibrated.

If the tool is ordered to close with a higher force, it might tolerate a higher speed at impact. The speed limit can be defined as a function of the closing torque, which is a function of the ordered tip force. The loop gain used for regulating the speed when it exceeds the limit is also specified.

Up to 6 points can be defined for speed limit and speed loop gain.

Ordered closing torque:	Speed limit:	Speed loop gain:
torque 1	Speed Limit 1	Kv 1
torque 2	Speed Limit 2	Kv 2
torque 3	Speed Limit 3	Kv 3
torque 4	Speed Limit 4	Kv 4
torque 5	Speed Limit 5	Kv 5
torque 6	Speed Limit 6	Kv 6

Speed limit 1 and *Kv 1* are valid for all torque values lower than *torque 1*. The highest defined speed limit and loop gain are valid for all torque values higher than the highest defined torque. For torque values between defined points, linear interpolation is used.

If only one point is defined, that speed limit and speed loop gain is valid for all torque values.

Limitations

Force Master Control can only be used if you have servo tools.

Related information

Application manual - Controller software IRC5

Continues on next page

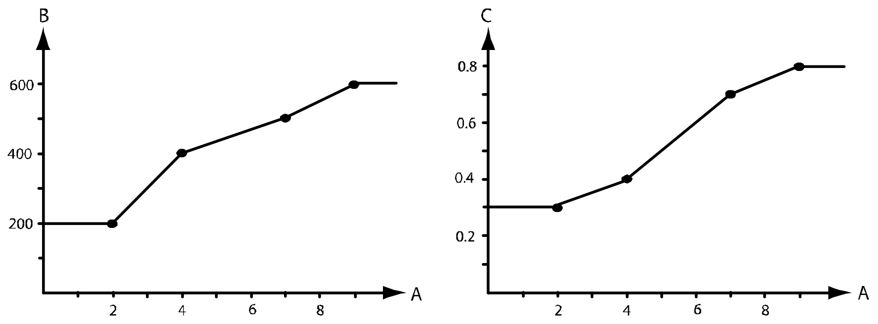
Example

In this example, four points are used to define the speed limit and speed loop gain. Any values given for point 5 and 6 are ignored.

The parameters in the type *Force Master Control* are set to the following values:

Parameter:	Value:
No. of speed limits	4
torque 1	2
torque 2	4
torque 3	7
torque 4	9
Speed Limit 1	200
Speed Limit 2	400
Speed Limit 3	500
Speed Limit 4	600
Kv 1	0.3
Kv 2	0.4
Kv 3	0.7
Kv 4	0.8

The results of this configuration are the following graphs for speed limit and speed loop gain:



xx0400000882

A	Torque (Nm)
B	Speed limit (rad/s on motor)
C	Speed loop gain (Nms/rad)

6.14.2 Name

Parent

Name belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Force Master Control*.

Usage

Name is used to reference a *Force Master Control* from the parameter *Use Force Master* in the type *Force Master*.

Allowed values

A string with maximum 32 characters.

6.14.3 No. of Speed Limits

Parent

No. of Speed Limits belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

no_of_posts

Description

No. of Speed Limits defines the number of torque values you want to define for speed limit and speed loop gain, i.e. the number of points in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for a number of torque values. Set *No. of Speed Limits* to the number of torque values you want to specify.

Limitations

No. of Speed Limits can only be used if you have servo tools.

Allowed values

An integer between 1 and 6.

The default value is 1.

Related information

Application manual - Controller software IRC5

6.14.4 Torque 1

Parent

Torque 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_1

Description

Torque 1 defines the ordered closing torque for the first point in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 1* to the torque value of the first point you want to specify.

Limitations

Torque 1 is used for servo tools and can only be used if you have the option *Servo Tool Control*.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 1 Nm.

Related information

Application manual - Controller software IRC5

6.14.5 Torque 2

Parent

Torque 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_2

Description

Torque 2 defines the ordered closing torque for the second point (if more than one) in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 2* to the torque value of the second point you want to specify.

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Torque 2* is not used.

Limitations

Torque 2 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 2 Nm.

Related information

No. of Speed Limits on page 486.

Application manual - Controller software IRC5

6.14.6 Torque 3

Parent

Torque 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_3

Description

Torque 3 defines the ordered closing torque for the third point (if more than two) in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 3* to the torque value of the third point you want to specify.

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *Torque 3* is not used.

Limitations

Torque 3 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 3 Nm.

Related information

No. of Speed Limits on page 486.

Application manual - Controller software IRC5

6.14.7 Torque 4

Parent

Torque 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_4

Description

Torque 4 defines the ordered closing torque for the fourth point (if more than three) in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 4* to the torque value of the fourth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *Torque 4* is not used.

Limitations

Torque 4 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 4 Nm.

Related information

No. of Speed Limits on page 486.

Application manual - Controller software IRC5

6.14.8 Torque 5

Parent

Torque 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_5

Description

Torque 5 defines the ordered closing torque for the fifth point (if more than four) in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 5* to the torque value of the fifth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 5 or higher, otherwise the value of *Torque 5* is not used.

Limitations

Torque 5 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 5 Nm.

Related information

No. of Speed Limits on page 486.

Application manual - Controller software IRC5

6.14.9 Torque 6

Parent

Torque 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_6

Description

Torque 6 defines the ordered closing torque for the sixth point (if all six points are used) in the speed limit graph (see *Example on page 484*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *Torque 6* to the torque value of the sixth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *Torque 6* is not used.

Limitations

Torque 6 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 6 Nm.

Related information

No. of Speed Limits on page 486.

Application manual - Controller software IRC5

6.14.10 Speed Limit 1

Parent

Speed Limit 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_1

Description

Speed Limit 1 defines the maximum allowed speed for the torque specified in *torque 1*.

Usage

Set *Speed Limit 1* to the speed limit for the first point you want to specify in the speed limit graph (see *Example on page 484*).

Limitations

Speed Limit 1 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 1 on page 487.

Application manual - Controller software IRC5.

6.14.11 Speed Limit 2

Parent

Speed Limit 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_2

Description

Speed Limit 2 defines the maximum allowed speed for the torque specified in *torque 2*.

Usage

Set *Speed Limit 2* to the speed limit for the second point (if more than one) you want to specify in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Speed Limit 2* is not used.

Limitations

Speed Limit 2 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 2 on page 488.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.12 Speed Limit 3

Parent

Speed Limit 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_3

Description

Speed Limit 3 defines the maximum allowed speed for the torque specified in *torque 3*.

Usage

Set *Speed Limit 3* to the speed limit for the third point (if more than two) you want to specify in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *Speed Limit 3* is not used.

Limitations

Speed Limit 3 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 3 on page 489.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.13 Speed Limit 4

Parent

Speed Limit 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_4

Description

Speed Limit 4 defines the maximum allowed speed for the torque specified in *torque 4*.

Usage

Set *Speed Limit 4* to the speed limit for the fourth point (if more than three) you want to specify in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *Speed Limit 4* is not used.

Limitations

Speed Limit 4 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 4 on page 490.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.14 Speed Limit 5

Parent

Speed Limit 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_5

Description

Speed Limit 5 defines the maximum allowed speed for the torque specified in *torque 5*.

Usage

Set *Speed Limit 5* to the speed limit for the fifth point (if more than four) you want to specify in the speed limit graph (see *Example on page 484*).

Limitations

Speed Limit 5 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 5 on page 491.

No. of Speed Limits on page 486.

6.14.15 Speed Limit 6

Parent

Speed Limit 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_6

Description

Speed Limit 6 defines the maximum allowed speed for the torque specified in *torque 6*.

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *Speed Limit 6* is not used.

Usage

Set *Speed Limit 6* to the speed limit for the sixth point (if all six points are used) you want to specify in the speed limit graph (see *Example on page 484*).

Limitations

Speed Limit 6 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

Torque 6 on page 492.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.16 Kv 1

Parent

Kv 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_1

Description

Kv 1 defines the proportional gain in the speed loop for the torque specified in *torque 1*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 1* to the proportional gain you want for the first point in the speed limit graph (see *Example on page 484*).

Limitations

Kv 1 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 1 on page 487.

Application manual - Controller software IRC5.

6.14.17 Kv 2

Parent

Kv 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_2

Description

Kv 2 defines the proportional gain in the speed loop for the torque specified in *torque 2*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 2* to the proportional gain you want for the second point (if more than one) in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Kv 2* is not used.

Limitations

Kv 2 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 2 on page 488.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.18 Kv 3

Parent

Kv 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_3

Description

Kv 3 defines the proportional gain in the speed loop for the torque specified in *torque 3*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 3* to the proportional gain you want for the third point (if more than two) in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *Kv 3* is not used.

Limitations

Kv 3 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 3 on page 489.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.19 Kv 4

Parent

Kv 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_4

Description

Kv 4 defines the proportional gain in the speed loop for the torque specified in *torque 4*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 4* to the proportional gain you want for the fourth point (if more than three) in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *Kv 4* is not used.

Limitations

Kv 4 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 4 on page 490.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.20 Kv 5

Parent

Kv 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_5

Description

Kv 5 defines the proportional gain in the speed loop for the torque specified in *torque 5*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 5* to the proportional gain you want for the fifth point (if more than four) in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 5 or higher, otherwise the value of *Kv 5* is not used.

Limitations

Kv 5 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 5 on page 491.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.14.21 Kv 6

Parent

Kv 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_6

Description

Kv 6 defines the proportional gain in the speed loop for the torque specified in *torque 6*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 6* to the proportional gain you want for the sixth point (if all six points are used) in the speed limit graph (see *Example on page 484*).

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *Kv 6* is not used.

Limitations

Kv 6 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

Torque 6 on page 492.

No. of Speed Limits on page 486.

Application manual - Controller software IRC5.

6.15 Type Friction Compensation

6.15.1 The Friction Compensation type

Overview

This section describes the type *Friction Compensation*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CFRIC_BLOCK

Type description

Each set of parameters of the type *Friction Compensation* belongs to a joint (robot joint or additional axis).

The parameters in *Friction Compensation* define what compensations should be made for the friction in the joint.

Limitation

Changing the parameter values in *Friction Compensation* is only useful if you have the RobotWare option *Advanced Shape Tuning*.

The type *Friction Compensation* equivalent to the type *Control Parameters*. The type *Control Parameters* is used by robot models IRB 1400 and IRB 1410, all other robot models use the type *Friction Compensation*. The parameters are the same however.

Related information

Application manual - Controller software IRC5, chapter *Advanced Shape Tuning*.

6.15.2 Name

Parent

Name belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the friction compensation.

Limitations

Name is only useful if you have the RobotWare option *Advanced Shape Tuning.*

Allowed values

A string with maximum 32 characters.

6.15.3 Friction FFW On

Parent

Friction FFW On belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

friction_ff_on

Description

Friction FFW On determines if the RobotWare option *Advanced Shape Tuning* is active or not.

Usage

Set *Friction FFW On* to Yes if you want to use *Advanced Shape Tuning*.

Limitations

Friction FFW On is useful only if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

Yes or No.

Related information

Application manual - Controller software IRC5

6.15.4 Friction FFW Level

Parent

Friction FFW Level belongs to the type *Friction Compensation* , in the topic *Motion*.

Cfg name

friction_ffw_level

Description

Friction FFW Level is set to the level of friction in the robot axis. By setting a value that closely corresponds to the real friction, and using the RobotWare option *Advanced Shape Tuning*, the friction effects can be compensated.

Usage

Friction effects can cause path deviations when performing advanced shapes. By compensating for the friction with the correct friction level value, these effects can be minimized.

Permanent adjustments to the friction level can be made with *Friction FFW Level*. The friction level can also be temporarily tuned with RAPID commands. For more information, see *Application manual - Controller software IRC5*.

Limitations

Friction FFW Level is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A decimal number between 0 and 15 (in Nm).

Related information

Application manual - Controller software IRC5

6.15.5 Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

friction_ffw_ramp

Description

Friction FFW Ramp is set to the speed of the robot axis when the friction has reached the constant friction level defined in *Friction ffw level*. See illustration below.

Usage

Friction effects can cause path deviations when performing advanced shapes. *Friction FFW Ramp* is used when compensating for these friction effects.

Permanent adjustments to the friction ramp can be made with *Friction FFW Ramp*. The friction ramp can also be temporarily tuned with RAPID commands. For more information, see *Application manual - Controller software IRC5*.

Limitations

Friction FFW Ramp is only useful if you have the RobotWare option *Advanced Shape Tuning*.

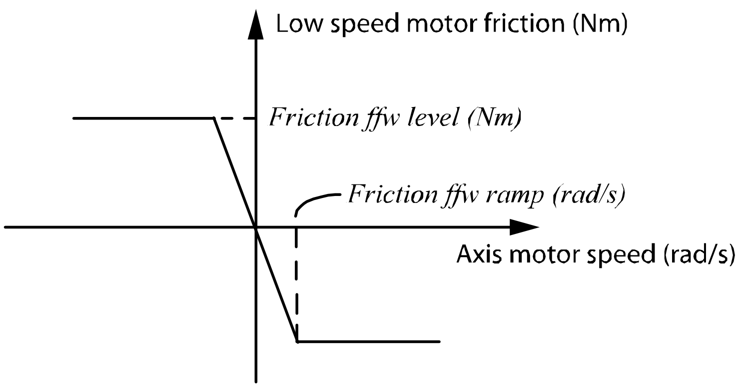
Allowed values

A number between 0.001 and 10 (in radians/second).

Related information

Application manual - Controller software IRC5

Illustration



en0300000278

6.16 Type Jog Parameters

6.16.1 The Jog Parameters type

Overview

This section describes the type *Jog Parameters*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic.

Cfg name

JOG_PARAMETERS

Type description

The *Jog Parameters* type contains parameters that define the step size in the different jogging modes when using incremental jogging with user-defined step.

Incremental movement

Incremental movement is used to adjust the position of the robot exactly. Each time the joystick is moved, the robot moves one step (one increment).

6.16.2 Name

Parent

Name belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

Name

Description

Name defines the name of the Jog parameters data.

Allowed values

A string with maximum 32 characters.

6.16.3 Configurable Linear Step Size

Parent

Configurable Linear Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

linear_step_size

Description

Configurable Linear Step Size defines the step size for user-defined incremental linear jogging.

Usage

Linear jogging step size is set in meters.

Allowed values

0 - 0.005 meters.

6.16.4 Configurable Reorient Step Size

Parent

Configurable Reorient Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

reorient_step_size

Description

Configurable Reorient Step Size defines the step size for user-defined incremental reorient jogging.

Usage

Reorient jogging step size is set in radians.

Convert degrees to radians: `radians = (degrees/360)*(2*pi)`

Allowed values

0 - 0.009 radians.

6.16.5 Configurable Joint Step Size

Parent

Configurable Joint Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

joint_step_size

Description

Configurable Joint Step Size defines the step size for user-defined incremental axes jogging.

Usage

Axes jogging step size is set in radians.

Convert degrees to radians: `radians = (degrees/360)*(2*pi)`

Allowed values

0 - 0.0025 radians.

6.16.6 Jog Mode

Parent

Jog Mode belongs to the type *Jogging Parameters*, in the topic *Motion*.

Cfg name

jog_mode

Description

Jog Mode is used to decide the active jogging mode. When the *Jog Mode* is Responsive the jogging is more responsive than the standard jogging.

Usage

When set to Responsive, the responsive jogging is enabled. For example, the *Jog Mode* should be set to Standard when World Zones is active.

Default value

Default value is Standard

Allowed values

Standard

Responsive

6.17 Type Joint

6.17.1 The Joint type

Overview

This section describes the type *Joint* which belongs to the topic *Motion*. Each parameter is described in a separate information topic in this section.

Cfg name

JOINT

Type description

The *Joint* type contains parameters that define a joint.

Related information

The Arm type on page 404.

The Measurement Channel type on page 559.

6.17.2 Name

Parent

Name belongs to the type *Joint*, in the topic *Motion*.

Cfg name

name

Description

Name defines the unique name to use for this joint.

Allowed values

A string with maximum 32 characters.

6.17.3 Logical Axis

Parent

Logical Axis belongs to the type *Joint*, in the topic *Motion*.

Cfg name

logical_axis

Description

Logical Axis defines the axis number as seen by a RAPID program.

Usage

The value of *Logical Axis* is used by RAPID programs to identify individual axes in mechanical units.

Two mechanical units can have the same value set for *Logical Axis*, but then they cannot be activated at the same time by a RAPID program.

Robots from ABB normally use the values 1-6, while additional axes use 7-12.

Allowed values

A value between 1 and 12.

Related information

Application manual - Additional axes and stand alone controller.

6.17.4 Use Drive System

Parent

Use Drive System belongs to the type *Joint*, in the topic *Motion*.

Cfg name

use_drive_system

Description

Use Drive System determines which drive system should be used.

Allowed values

A string with maximum 32 characters.

Related information

The Drive System type on page 452.

6.17.5 Use Process

Parent

Use Process belongs to the type *Joint*, in the topic *Motion*.

Cfg name

use_process

Description

Use Process defines which process to use for this joint.

Usage

Use Process points to a process ID defined by the parameter *Name* in the type *Process*.

The process can be used to define the joints behavior for either *Electronically Linked Motors* or *Spot Servo*.

Prerequisites

The additional axes must be configured before setting *Use Process*.

Limitations

Use Process is only used for additional axes.

Use Process is only useful if you have either of the RobotWare base functionality *Electronically Linked Motors* or option *Spot Servo*.

Allowed values

A string.

Related information

Name on page 661.

Application manual - Controller software IRC5

6.17.6 Lock Joint in Ipol

Parent

Lock Joint in Ipol belongs to the type *Joint*, in the topic *Motion*.

Cfg name

lock_joint_in_ipol

Description

A flag that locks the axis so it is not used in the path interpolation.

Usage

When setting *Lock Joint in Ipol* to Yes, this axis will not be used for path interpolation.

When using *Electronically Linked Motors*, this parameter must be set to Yes for the follower axis.

Prerequisites

The additional axes must be configured before setting *Lock Joint in Ipol*.

Limitations

Lock Joint in Ipol is only used for additional axes.

Allowed values

Yes or No.

Related information

Application manual - Controller software IRC5

6.17.7 Follower to Joint

Parent

Follower to Joint belongs to the type *Joint*, in the topic *Motion*.

Cfg name

follower_to_joint

Description

When using *Electronically Linked Motors*, *Follower to Joint* defines which master axis this axis should follow.

Usage

When using *Electronically Linked Motors*, the follower axis has the *Follower to Joint* set to the name of the master axis.

Prerequisites

The additional axes must be configured before setting *Follower to Joint*.

Limitations

Follower to Joint is only used for external axes.

Allowed values

A string.

Related information

Application manual - Controller software IRC5

6.17.8 Drive Module Number

Parent

Drive Module Number belongs to the type *Joint*, in the topic *Motion*.

Cfg name

drive_module

Description

Drive Module Number defines the drive module number that should be used.

Usage

Drive Module Number points to the number in the drive module defined by the parameter *Name* in the *Type Drive Module*.

Limitations

The Drive Module Number has to be equal to the number in the parameter *Use Drive Module* in the *Type Joint*.

Allowed values

A value between 1 and 4.

The default value is 1.

Related information

Use Drive Module on page 524

6.17.9 Use Drive Module

Parent

Use Drive Module belongs to the type *Joint*, in the topic *Motion*.

Cfg name

use_drive_module

Description

Use Drive Module determines which drive module should be used.

Usage

Use Drive Module points to a drive module ID defined by the parameter *Name* in the *Type Drive Module*.

Limitations

The number in this name has to be equal to the Drive Module Number in the parameter *Drive Module Number* in the *Type Joint*.

Allowed values

A string with maximum 32 characters.

The default value is drive_module_1.

Related information

Type Drive Module on page 449.

Drive Module Number on page 523.

6.18 Type Lag Control Master 0

6.18.1 The Lag Control Master 0 type

Overview

This section describes the type *Lag Control Master 0*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

LCM0

Type description

The type *Lag Control Master 0* is normally used for control of axes without any dynamic model. This is the case for some additional axes.

For axes that have a dynamic model, *Lag Control Master 0* is only used in exceptional cases.

6.18.2 Name

Parent

Name belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Lag Control Master 0*.

Usage

Name is used to reference a *Lag Control Master 0* from the parameter *Normal Control Master* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6.18.3 Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Kp

Description

Proportional gain in the position control loop.

Usage

The higher the value of *Kp, Gain Position Loop*, the better tracking and disturbance rejection.

If the position control overshoots, decrease *Kp, Gain Position Loop*.

Allowed values

A numeric value between 0 and 1000 (1/s).

6.18.4 Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Lag Control Master*, in the topic *Motion*.

Cfg name

Kv

Description

Proportional gain in the speed regulation loop.

Usage

The higher the value of *Kv, Gain Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, decrease *Kv, Gain Speed Loop*.

Allowed values

A numeric value between 0 and 100 (Nms/rad).

6.18.5 Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Ti

Description

Integration time in the speed regulation loop.

Usage

The lower the value of *Ti Integration Time Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, increase *Ti Integration Time Speed Loop*.

Allowed values

A numeric value between 0 and 10 (seconds).

The default value is 10 seconds.

6.18.6 Forced Control Active

Parent

Forced Control Active belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

use_inpos_forced_control

Description

Determines whether forced control is active for this joint.

Usage

The Forced Control Active parameter can be used if the last part of the movement before a fine point is too slow. The function changes the parameters *Forced Factor for Kp* and *Forced Factor for Ki* in the last part of the movement.

Note! Wrongly used Forced Control Active (too high force factors) might impair the movement with oscillations.

If *Forced Control Active* is set to Yes, *Affects forced ctrl* in type *Supervision* should normally also be set to Yes for this joint.

Allowed values

Yes or No.

Related information

Forced Factor for Kp on page 531.

Forced Factor for Ki on page 532.

Affects Forced Control on page 751, in the type *Supervision*.

Application manual - Additional axes and stand alone controller.

6.18.7 Forced Factor for Kp

Parent

Forced Factor for Kp belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Kp_forced_factor

Description

The forced factor for *Kp*, if forced gain control is active.

Usage

Forced Factor for Kp defines the gain increase factor.

A typical value is 2.

Allowed values

A numeric value between 1 and 4.

6.18.8 Forced Factor for Ki

Parent

Forced Factor for Ki belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Ki_forced_factor

Description

The forced factor for *Ki*, if forced gain control is active.

Usage

Forced Factor for Ki defines the gain increase factor.

Ki equals Kv/Ti, integral gain.

A typical value is 2.

Allowed values

A numeric value between 1 and 4.

6.18.9 Raise Time for Kp

Parent

Raise Time for Kp belongs to the type *Lag Control Master*, in the topic *Motion*.

Cfg name

Kp_raise_time

Description

Defines the raise time for forced *Kp*.

Usage

To avoid transient effects, Kp must be increased slowly over a period of time. This period is defined by *Raise Time for Kp*.

A typical value is 0.2.

Allowed values

A numeric value between 0.002 and 0.5 seconds.

6.18.10 FFW Mode

Parent

FFW Mode belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

ffw_mode

Description

FFW Mode defines the control type to use, i.e. if feed forward should be used.

Usage

To regulate the position, you can:

- use only the desired position as reference.
- in addition to the position, use feed forward of the current speed value.
- in addition to the position, use feed forward of the current speed and torque values.

Allowed values

FFW Mode can have the following values:

Value:	Name:	Description:
0	No	The controller is driven by the position error (lag). Because a relatively large lag is needed to move the axis, the position error can be large.
1	Spd	The controller receives information about the desired speed of the axis. As a result, the position lag is greatly reduced compared to the No configuration. For this reason, Spd is the recommended configuration.
2	Trq	The controller uses the desired speed and acceleration of the axis to calculate the desired motor torque. This requires knowledge of the mass moment of inertia of the axis, which must be supplied by the user. For this reason this configuration is more difficult to tune. It is only recommended for experienced users.

The default value is 0. Recommended value is 1.

Related information

Application manual - Additional axes and stand alone controller.

6.18.11 Bandwidth

Parent

Bandwidth belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

bandwidth

Description

Defines the controller bandwidth when *FFW Mode* is set to 1 or 2.

Usage

A high bandwidth value gives faster control but increases risk of vibrations and overshoot.

The default value is recommended, but can be reduced if undesired vibrations occur.

Allowed values

A value between 0.5 and 75. The default value is 25.

Related information

FFW Mode on page 534.

6.18.12 Df

Parent

Df belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

resonance_frequency

Description

Reduces oscillations.

Usage

Df can be used to damp oscillations of the axis due to mechanical resonance.

Initially *Df* should be left at its default value. It can be adjusted once the other controller parameters have been fixed (*Kv Gain Speed Loop*, *Kp Gain Position Speed Loop*, *Ti Integration Time Speed Loop*, and *Inertia*).

Df is only used when *FFW Mode* is set to 2.

Allowed values

A value between 2 and 100. Default value is 100.

Related information

FFW Mode on page 534.

Kp, Gain Position Loop on page 527.

Kv, Gain Speed Loop on page 528.

Ti Integration Time Speed Loop on page 529.

Inertia on page 539.

6.18.13 Dw

Parent

Dw belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

resonance_damping

Description

Can reduce oscillations further when *Df* is set.

Usage

The default value of *Dw* is recommended.

Allowed values

A value between 0.002 to 1. Default value is 0.01.

Related information

Df on page 536.

6.18.14 Delay

Parent

Delay belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

delay_time

Description

Reduces overshoot.

Usage

Delay can be used when *Df* is set, to reduce overshoot but it impairs the axis coordination when increased.

The default value of *Delay* should normally not be changed.

Allowed values

A value between 0.0 and 0.02. Default value is 0.004.

Related information

Df on page 536.

Dw on page 537.

6.18.15 Inertia

Parent

Inertia belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

inertia

Description

Defines the additional axis' inertia (if rotation) or mass (if translation).

Usage

Inertia is used for calculating the torque when *FFW Mode* is set to 2.

Allowed values

A value between 0.0 and 10,000.

Related information

FFW Mode on page 534.

6.18.16 K Soft Max Factor

Parent

K Soft Max Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_K_max_factor

Description

Determines the value of the product *Kp Gain Position Loop* * *Kv Gain Speed Loop* when the soft servo is used with softness 0%.

Usage

K Soft Max Factor should be in the range 0.1 - 2.0 (default 1.0). When the soft servo is activated with 0% softness, the control parameters *Kp Gain Position Loop (Kp)* and *Kv Gain Speed Loop (Kv)* will be tuned such that *Kp*Kv* = (*Kp*Kv*)normal**K Soft Max Factor*, where (*Kp*Kv*)normal is the product of *Kp* and *Kv* during normal operation.

Allowed values

A value between 0.1 and 2.0. Default value is 1.0.

Related information

Kp, Gain Position Loop on page 527.
Kv, Gain Speed Loop on page 528.

6.18.17 K Soft Min Factor

Parent

K Soft Min Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_K_min_factor

Description

Determines the value of the product *Kp Gain Position Loop * Kv Gain Speed Loop* if the soft servo is used with softness 100%.

Usage

K Soft Min Factor should be in the range 0.001 - 0.1 (default 0.01). When the soft servo is activated with 100% softness, the control parameters *Kp Gain Position Loop (Kp)* and *Kv Gain Speed Loop (Kv)* are tuned such that *Kp*Kv* = (*Kp*Kv*)normal**K Soft Min Factor*.

Allowed values

A value between 0.001 and 0.1. Default value is 0.01.

Related information

Kp, Gain Position Loop on page 527.

Kv, Gain Speed Loop on page 528.

6.18.18 Kp/Kv Ratio Factor

Parent

Kp/Kv Ratio Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_Kp_Kv_ratio_factor

Description

Defines the factor used to tune the *Kp Gain Position Loop/Kv Gain Speed Loop* ratio.

Usage

Kp/Kv Ratio Factor is used to alter the *Kp Gain Position Loop/Kv Gain Speed Loop* ratio during soft servo. *Kp/Kv Ratio Factor* should be in the range 0.1 - 1.0 (default 1.0). In soft servo mode, *Kp* and *Kv* are tuned such that *Kp/Kv* = (*Kp/Kv*)normal * *Kp/Kv Ratio Factor*.

Allowed values

A value between 0.1 and 1.0.

Related information

Kp, Gain Position Loop on page 527.

Kv, Gain Speed Loop on page 528.

6.18.19 Ramp Time

Parent

Ramp Time belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_t_ramp

Description

Defines the default Soft Servo ramp time.

Usage

Ramp Time is used to define the default time for activation of the soft servo.

Allowed values

A value between 0.01 and 0.5. Default value is 0.05.

6.19 Type Linked M Process

6.19.1 The Linked M Process type

Overview

This section describes the type *Linked M Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

LINKED_M_PROCESS

Type description

A *Linked M Process* contains information about alignments between the master axis and the follower axis for *Electronically Linked Motors*.

Related information

Application manual - Controller software IRC5, chapter *Electronically Linked Motors*.

6.19.2 Name

Parent

Name belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

name

Description

Name defines the identity of the linked motor process.

Usage

The *Name* is used when referencing the linked motor process.

The linked motor process defines the behavior of a joint for *Electronically Linked Motors*.

Allowed values

A string.

6.19.3 Offset Adjust. Delay Time

Parent

Offset Adjust. Delay Time belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

offset_adj_delay_time

Description

Offset Adjust. Delay Time defines the time delay from control on until the follower axis starts to follow its master axis.

Usage

When using *Electronically Linked Motors*, you might want to give the master axis some time to stabilize before the follower axis starts following.

Allowed values

A value between 0 and 2, specifying the delay in seconds.

Default value: 0.2

6.19.4 Max Follower Offset

Parent

Max Follower Offset belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

max_offset

Description

Max Follower Offset defines the maximum allowed difference in position between the master and the follower axis.

Usage

If the follower offset exceeds the *Max Follower Offset*, emergency stop is activated and automatic offset adjustment is prohibited.

Allowed values

A value between 0 and 5, specifying the maximum offset in radians (for rotational axes) or meters (for linear axes) on the arm side.

Default value: 0.05.

6.19.5 Max Offset Speed

Parent

Max Offset Speed belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

max_offset_speed

Description

Max Offset Speed defines the maximum allowed difference in speed between the master and the follower axis.

Usage

If the speed difference exceeds the *Max Offset Speed*, emergency stop is activated and automatic offset adjustment is prohibited.

Allowed values

A value between 0 and 1000, specifying the maximum difference in rad/s (for rotational axes) or m/s (for linear axes) on the arm side.

Default value: 0.05.

6.19.6 Offset Speed Ratio

Parent

Offset Speed Ratio belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

offset_speed_ratio

Description

Offset Speed Ratio defines how large a part of the *Max Offset Speed* can be used to compensate for position error.

Usage

Offset Speed Ratio multiplied by *Max Offset Speed* is the highest speed by which the position offset is reduced.

Allowed values

A value between 0 and 1. The value has no unit since it is a multiplication factor.

Default value: 0.33.

Related information

Max Offset Speed on page 548.

6.19.7 Ramp Time

Parent

Ramp Time belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

ramp_time

Description

Ramp Time defines the acceleration up to *Max Offset Speed*.

Usage

The proportion constant for position regulation is ramped from zero up to its final value (*Master Follower kp*) during *Ramp Time*.

Allowed values

A value between 0.01 and 10, specifying the time in seconds.

Default value: 0.05

Related information

Master Follower Kp on page 551.

Max Offset Speed on page 548.

6.19.8 Master Follower Kp

Parent

Master Follower Kp belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

kp_offset

Description

Master Follower Kp is the proportion constant for position regulation.

Usage

Master Follower Kp determines how fast the position error is compensated. If the value is too low, the compensation will be slow. If the value is to large, the compensation will be unstable.

Allowed values

A value between 0 and 5 (unit is 1/s).

Default value: 0.05.

6.19.9 Torque follower

Parent

Torque follower belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

torque_follower

Description

Torque follower specifies whether the follower should share torque with master axis rather than regulating to the exact corresponding position.

Usage

Torque follower turns on or off the torque follower functionality. If the value is Yes the follower axis will share torque with master axis.

Allowed values

Yes or No.

Default value is No.

6.19.10 Torque distribution

Parent

Torque distribution belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

torque_distribution

Description

Torque distribution is a quota defining how much of the total torque should be applied by the follower axis.

Usage

Torque distribution can be used to distribute torque between master and follower axis. Normally when running equal motors and drives the value should be 0.5 corresponding to share torque equal between master and follower.

This parameter will have no effect if *Torque follower* is set to No.

Allowed values

A value between 0 and 1.

Default value is 0.5.

Example

If *Torque distribution* is set to 0.3, the torque is distributed with 30% on the follower and 70% on the master.

6.19.11 Follower axis pos. acc. reduction

Parent

Follower axis pos. acc. reduction belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

follower_axis_pos_accuracy_reduction

Description

Follower axis pos. acc. reduction can be used to reduce torque on master and follower axis if the torque is from position error between the axes.

Usage

Follower axis pos. acc. reduction can be used if mechanical structure is extremely stiff or if to large position error between the axes causes to high torques. By setting this parameter to a higher value, the position accuracy of the follower axis will be reduced and that will lower the part of the total torque which comes from position error.

A too high value of this can cause instability.

Normal value is 10-30.

This parameter will have no effect if *Torque follower* is set to No.

Allowed values

A value between 0 and 100.

Default value: 0.

6.20 Type Mains

6.20.1 The Mains type

Overview

This section describes the type *Mains*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MAINS

Type description

The type *Mains* defines the drive system's mains power tolerance. The parameters of the Mains type have nominal values.

The parameters of the type *Mains* can be used to improve the robot's performance by adjusting them according to the robot's actual installation.

CAUTION

Parameter settings outside the range of the robot's installation may negatively affect the robot's performance.

Related information

How to optimize drive system parameters on page 390.

Mains Tolerance Min on page 557

Mains Tolerance Max on page 558.

6.20.2 Name

Parent

Name belongs to the type *Mains*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the mains tolerance setting it belongs to.

Allowed values

A string with maximum 32 characters, specifying the name.

Related information

How to optimize drive system parameters on page 390.

6.20.3 Mains Tolerance Min

Parent

Mains Tolerance Min belongs to the type *Mains*, in the topic *Motion*.

Cfg name

u_tolerance_min

Description

Mains Tolerance Min specifies the minimum value of the mains tolerance as a percentage. The value is set to -15% on delivery. If the minimum tolerance is less than 15%, the cycle time can be improved by changing the parameter.

For more information, see *How to optimize drive system parameters on page 390*.

Allowed values

A value between -1 and +1 (equals -100% and 100%).

The default value is -0.15 (equals -15%).

For single phase 220V systems the default value is specified as 220V -15%. If 230V mains is used and the tolerance is 230V -15% then set the parameter manually to -0.11 (220V -11% is approximately 230V -15%).

Related information

How to optimize drive system parameters on page 390.

6.20.4 Mains Tolerance Max

Parent

Mains Tolerance Max belongs to the type *Mains*, in the topic *Motion*.

Cfg name

u_tolerance_max

Description

Mains Tolerance Max specifies the maximum value of the mains tolerance. Its default value is 0.1 (10%). This value normally should not be increased since the equipment is rated for this maximum mains tolerance and might be damaged if the voltage is increased.

For 220V single phase systems the default value is 0.10 (10%). If 230 V mains is used and the tolerance should be 230 V + 10% then set the parameter manually to 0.15 (220 V + 15% is the same as 230 V + 10%).

Allowed values

The default value is 0.1.

Related information

How to optimize drive system parameters on page 390.

6.21 Type Measurement Channel

6.21.1 The Measurement Channel type

Overview

This section describes the type *Measurement Channel* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MEASUREMENT_CHANNEL

Type description

The type *Measurement Channel* describes which channel is used to send measurement data from the axis computer to the controller.

Non-editable parameters

The following parameters are visible but not editable in the software configuration tools:

- *Max Normalized Input Level*
- *Min Normalized Input Level*

As a consequence, the above parameters are not described in the manual.

6.21.2 Name

Parent

Name belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

name

Description

Name defines the axis computer's channel name.

Allowed values

A string with maximum 32 characters.

6.21.3 Disconnect at Deactivate

Parent

Disconnect at Deactivate belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

disconnect_at_deactivate

Description

Disconnect at Deactivate defines if the channel should be deactivated when the mechanical unit is deactivated.

Usage

Set *Disconnect at Deactivate* to Yes to avoid error reports when the resolver is disconnected, for instance when switching between tools.

Allowed values

Yes or No.

Default value is No

6.21.4 Measurement Link

Parent

Measurement Link belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

measurement_link

Description

An axis resolver is connected to a Serial Measurement Board (SMB). The SMB communicates with the axis computer via a serial measurement link.

Measurement Link defines the number of the measurement link.

Usage

There are two contacts on the axis computer marked Measurement link 1 and Measurement link 2.

An ABB robot is normally connected to link 1.

Allowed values

1 or 2.

Default value is 1.

6.21.5 Board Position

Parent

Board Position belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

board_position

Description

Board Position defines the position number of the board used for the measurement system.

Usage

The value of *Board Position* defines the physical position of the board on the measurement link. Board position one is closest to the axis computer.

Allowed values

An integer value between 1 and 2.

Default value is 1.

6.22 Type Mechanical Unit

6.22.1 The Mechanical Unit type

Overview

This section describes the type *Mechanical Unit* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MECHANICAL_UNIT

Type description

The *Mechanical Unit* type describes the common parameters for a mechanical unit. There is one set of parameters for each mechanical unit.

This type is only possible to edit for additional axes, not for robots delivered from ABB.

Non-editable parameters

The following parameter is visible but not editable in the software configuration tools:

- *Use Run Enable*

As a consequence, the above parameter is not described in the manual.

Related information

Application manual - Additional axes and stand alone controller.

6.22.2 Name

Parent

Name belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name for the mechanical unit.

Allowed values

A string with maximum 32 characters.

6.22.3 Use Activation Relay

Parent

Use Activation Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_activation_relay

Description

Use Activation Relay defines the Id name for the activation relay.

Usage

Use Activation Relay points out a relay that will be activated or deactivated when the mechanical unit is activated or deactivated.

More information can be found in *Technical reference manual - RAPID Instructions, Functions and Data types* under the instructions `ActUnit/DeactUnit`.

Allowed values

A string with maximum 32 characters.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

6.22.4 Use Brake Relay

Parent

Use Brake Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_brake_relay

Description

Use Brake Relay defines the Id name for the brake relay.

Usage

Use Brake Relay points out what brake relay will be activated or deactivated when the mechanical unit goes to state control on or control off.

Allowed values

A string with maximum 32 characters.

6.22.5 Use Connection Relay

Parent

Use Connection Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_connection_relay

Description

Use Connection Relay defines the Id name for the connection relay.

Usage

Use Connection Relay points out a relay that must be activated when the mechanical unit is activated.

Allowed values

A string with maximum 32 characters.

6.22.6 Use Robot

Parent

Use Robot belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_robot

Description

Use Robot defines which robot is part of the mechanical unit.

Usage

The robot is defined in the type *Robot*.

Allowed values

A string with maximum 32 characters.

Related information

Name on page 669, of the type *Robot*.

6.22.7 Use Single 1, 2, 3, 4, 5, 6

Parent

Use Single 1, *Use Single 2*, *Use Single 3*, *Use Single 4*, *Use Single 5*, and *Use Single 6* belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg names

use_single_0

use_single_1

use_single_2

use_single_3

use_single_4

use_single_5

Description

Use Single defines which singles are part of the mechanical unit.

Usage

The mechanical unit can have six singles, *Use Single 1*, *Use Single 2*, *Use Single 3*, *Use Single 4*, *Use Single 5*, and *Use Single 6*. The singles are defined in the type *Single*.

Allowed values

Each single value is a string with maximum 32 characters.

Related information

Name on page 725, in the type *Single*.

6.22.8 Allow Move of User Frame

Parent

Allow Move of User Frame belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

allow_move_of_user_frame

Description

Allow Move of User Frame defines if a robot or single is allowed to move a user frame.

Usage

A user frame can be moved by a robot or a single that is part of the mechanical unit. Set *Allow Move of User Frame* to Yes to allow a robot or single to move a user frame.

Note that the definition of the work object must allow it to be moved, see `wobjdata` (`ufprog` and `ufmec`) in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Allowed values

Yes or No.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.22.9 Activate at Start Up

Parent

Activate at Start Up belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

activate_at_start_up

Description

Activate at Start Up defines if the mechanical unit should be activated at start.

Usage

Set the value to No to activate the mechanical unit at start.

Allowed values

Yes

No

6.22.10 Deactivation Forbidden

Parent

Deactivation Forbidden belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

deactivation_forbidden

Description

Deactivation Forbidden defines if the mechanical unit is allowed to be deactivated.

Usage

Set *Deactivation Forbidden* to No if the mechanical unit should be allowed to be deactivated.

Robots from ABB always has the value set to Yes. They should not be deactivated. The value No is used only for additional axes that should be possible to deactivate.

Allowed values

Yes or No.

6.22.11 Deactivate PTC superv. at disconnect

Parent

Deactivate PTC superv. at disconnect belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

deactivate_ptc_at_disconnect

Description

When set to Yes, the PTC supervision is disabled when the mechanical unit is disconnected and enabled again when it is activated.

Usage

The PTC supervision is used to detect high motor temperatures for mechanical units. If a unit is physically disconnected while the PTC supervision is active, an error will occur.

When using Servo Tool Change, it must be possible to disconnect the servo tool. By setting *Deactivate PTC superv. at disconnect* to Yes, the servo tool can be deactivated and removed without an error. When the new tool is connected and activated, PTC supervision is activated again.

Prerequisites

Setting *Deactivate PTC superv. at disconnect* to Yes is only useful if an additional axis is disconnected without turning off the robot system. This can only be done if you have the options Servo Tool Control and Servo Tool Change.

Limitations

If *Deactivate PTC superv. at disconnect* is set to Yes and the mechanical unit is deactivated, the PTC supervision is disabled for all additional axes in the system (but not for the robot).

Allowed values

Yes or No.

6.22.12 Activate from any motion task

Parent

Activate from any motion task belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

allow_activation_from_any_motion_task

Description

If *Activate from any motion task* is set to Yes, the mechanical unit can be deactivated by one task and then activated by another motion task. The mechanical unit is then controlled by the task that has activated it.

In other words, if the *Activate from any motion task* parameter is active, a mechanical unit can be moved between different motion tasks. Both the motion control and the RAPID execution for this unit will be moved to the other task.

Usage

If *Activate from any motion task* is set to Yes, a mechanical unit, for example a servo gun, can be used by two robots in a MultiMove system.

Example

A servo gun is held by robot 1 and controlled by the task T_ROB1. It is deactivated and disconnected from robot 1. The servo gun is then connected to robot 2 and activated by the task T_ROB2.

Limitations

The parameter *Deactivation Forbidden* must be set to No for this mechanical unit. *Activate from any motion task* can only be used for a mechanical unit that can be deactivated, that is not for a robot.

Activate from any motion task is only useful for a MultiMove system.

It is only supported to deactivate a mechanical unit from the same motion task as it was activated. This task controls the mechanical unit and can secure that it is standing still before deactivating it. When the mechanical unit has been deactivated, it can be activated in another motion task. The new task will then control the unit. It is important to remember that the two mechanical units with a common logical axis number cannot be active at the same time in a Rapid task, for more information see *Logical Axis on page 518*.

The mechanical unit must still belong to a mechanical unit group, see *Type Mechanical Unit Group on page 119*. This configuration determines which task that will control the mechanical unit at start.

Default value

The default value is No.

Continues on next page

Allowed values

Yes

No

Additional information

If the program pointer is moved to main, the mechanical unit regains its configuration from the system parameters, that is it is activated by its original task. Make sure the program is not restarted from main with the mechanical unit mounted on another robot than configured in the system parameters.

6.23 Type Motion Planner

6.23.1 The Motion Planner type

Overview

This section describes the type *Motion Planner*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Note

When several task programs are run in synchronized mode, the movements of all their mechanical unit groups are calculated by the same motion planner. It is then the first set of parameters of the type *Motion Planner* that is used.

Cfg name

MOTION_PLANNER

Type description

A motion planner is a process on the controller that calculates how mechanical units shall move. A controller that handles more than one robot also has more than one motion planner. Each mechanical unit group has its own motion planner.

Limitations

Unless the option *MultiMove* is installed, there can only be one motion planner configuration.

Related information

Application manual - MultiMove.

6.23.2 Name

Parent

Name belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

name

Description

The name of the motion planner.

Usage

This is the public identity of the motion planner. It is used by the parameter *Use Motion Planner* in the type *Mechanical Unit Group*.

Allowed values

A string with maximum 32 characters. The name must not be changed!

Related information

The Mechanical Unit Group type on page 119 in the topic *Controller*.

6.23.3 TCP Linear Max Speed (m/s)

Parent

TCP Linear Max Speed (m/s) belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

linear_max_speed

Description

It defines the maximum linear speed (m/s) in RAPID speeddata vmax and that is possible to set in RAPID speeddata for a TCP-robot.

Usage

The parameter is used to define v_tcp in RAPID speeddata vmax. If a higher value than this is used in a user defined speeddata it will be limited to this value.

Allowed values

A value between 0.01 and 339.

The default value is 7.

Related information

Technical Reference Manual – *RAPID Instructions, Functions and Data Types - VelSet*

Technical Reference Manual – *RAPID Instructions, Functions and Data Types - motset*

6.23.4 Brake on Time

Parent

Brake on Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

brake_on_timeout

Description

Brake on Time is used to delay the use of brakes when the robot is waiting to move. It defines the time from when the robot stops to when the mechanical brakes are activated.

Note

The brake on time value should be kept high to maintain the reliability of the servo at high level.

Limitations

Brake on Time needs to be set on all motion planners to have effect. It is necessary that all Mechanical Units in the system has a *Use Brake Relay* defined, else this parameter will have no effect. The highest value of all motion planners will be the one used (even if only one of the six motion planners is used).

Allowed values

A value between 0.3 to 3,600,000, specifying the time in seconds.

Related information

Use Brake Relay on page 567

6.23.5 Dynamic Resolution

Parent

Dynamic Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

dynamic_resolution

Limitation

Dynamic Resolution is optimized for the system at delivery. It should normally not be changed.

Allowed values

A predefined value, specified in seconds.

6.23.6 Path Resolution

Parent

Path Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

path_resolution

Description

The parameter corresponds in some sense to the distance between two points in the path. Increasing path resolution means increasing the distance, which leads to a decrease in the resolution of the path!

Increasing path resolution is a way to deal with robot installations that have external axes with long deceleration times due to high CPU load. In such applications the warning "50082 Deceleration limit" can be reported, simultaneously generating a quick-stop. Increasing the path resolution solves the problem.

Prerequisites

It is important to set the path resolution value as low as possible in order to achieve a high path resolution at high speed. Keeping the path resolution low can also give shorter cycle times if the cycle contains many stop points and the move instructions following these stop points have low speeds.

Usage

Path Resolution might require tuning when:

- The acceleration value of an additional axis (and the robot) is decreased using the first parameter of the RAPID instruction `AccSet`.
- The acceleration derivative is decreased using the second parameter of the RAPID instruction `AccSet`.
- The speed is increased.
- The distances between closely programmed positions are decreased.
- The number of simultaneously controlled axes are increased.
- Using coordinated interpolation.
- Using Weldguide.
- Using the option *Conveyor Tracking*.
- Using RAPID controlled path correction.
- Using multitasking with computationally demanding RAPID programs.
- Reorienting with a small or no TCP movement.

Allowed values

A value between 0.1667 to 6.00, specifying the resolution in seconds.

Additional information

There is also a RAPID instruction named `PathResol` which affects the resolution of the path.

Continues on next page

Related information

Technical reference manual - RAPID overview.

Application manual - Controller software IRC5.

6.23.7 Queue Time

Parent

Queue Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

std_servo_queue_time

Description

Increasing *Queue Time* makes the system more tolerant to uneven CPU loads.

Note

The real queue time is a multiple of a sample time related to dynamic resolution. If the parameter value is not an even multiple of the dynamic resolution, the controller will automatically use a queue time as close as possible to the given value.

Allowed values

A value between 0.004032 to 0.290304, specifying the time in seconds.

Additional information

A drawback with increasing the queue time is that the robot reacts more slowly when jogging and when stopping a program execution. However, the emergency brake is not affected. The accuracy of a sensor process, e.g. WeldGuide and Conveyor tracking, may also be affected.

6.23.8 Teach Mode Max Speed

Parent

Teach Mode Max Speed belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

teach_mode_max_speed

Description

Teach Mode Max Speed can be used to set the maximum TCP-speed in manual mode to less than the default value 0.25 m/s.

When the value of this parameter is reduced, the maximum joint speed in teach mode will also be reduced.

If the value is set to 0.2 m/s, all maximum joint speeds in teach mode will be reduced by 0.2/0.25=0.8, i.e. 80% of the previous values.

Allowed values

A value between 0.010 to 0.250, specifying the speed in meter per seconds.

The default value is 0.25 m/s.

6.23.9 Process Update Time

Parent

Process Update Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

process_linearization_time

Description

Process Update Time determines how often the process path information is calculated. This information is used for path following in Conveyor tracking, WeldGuide and Rapid Weave, for example.

Usage

Decreasing the process update time improves accuracy but also increases CPU load. Increasing the parameter decreases the CPU load.

Limitations

When running programs in which the manipulator is moving at high speed, the parameter value should be kept small in order to get the best performance. When the manipulator is moving slowly, the process update time is not critical.

Allowed values

A value between 0.012096 to 1.93536, specifying the time in seconds.

6.23.10 Prefetch Time

Parent

Prefetch Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

ipol_prefetch_time

Description

Prefetch Time affects the point in time at which the controller starts to plan for the motion through a corner zone. If the planning time is too short, the corner zone becomes a fine point. This generates a warning called“ 50024 Corner path failure”.

Usage

If the planning time is too short because of high CPU load, increasing the parameter value may solve the problem. However, it will not solve the problem when it is caused by too many corner zones placed very close together or by incorrect use of instructions, e.g. a corner zone followed by a `WaitDI` instruction. Normally, *Prefetch Time* should only be increased when the corner zone is really needed in the application. When it is not really needed, change the corner zone to a fine point.

Limitations

There is a drawback when increasing the parameter. The difference between the position of the executed RAPID instruction and the current position of the manipulator will increase. This means that after pressing stop during program execution, the program counter on the FlexPendant may show an instruction that has not yet affected the manipulator. When starting again, the manipulator continues along the original path.

Allowed values

A value between 0 to 10, specifying the time in seconds.

6.23.11 Event Preset Time

Parent

Event Preset Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

event_preset_time

Description

Event Preset Time is used to delay the robot to make it possible to activate/control external equipment in advance. This is to compensate for the internal delay of the equipment.

Usage

Adjustment for the internal delay of the equipment can be made with the instruction `TriggEquip`. This takes advantage of the delay between the RAPID commands and the robot movement. In this way an output signal can be set up to about 100 ms in advance. If the delay of the equipment is longer than 100 ms, then *Event Preset Time* must be used to increase the delay of the robot movement.

Configure *Event Preset Time* to the longest equipment delay time needed (if more than 100ms).

Allowed values

A value between 0 and 0.5, specifying the time in seconds.

Additional information

Remember that when using *Event Preset Time*, the start of the robot is delayed and the performance of *WeldGuide,* conveyors, spot welding, and so on will be decreased.

Example

If you use *Fixed Position Event* with the following RAPID instructions, you should configure *Event Preset Time* to 0.2 seconds (the maximum delay required by `TriggEquip`)

```
TriggEquip gunon, 10, 0.2 \DOp:=gun, 1;
TriggL p1, v500, gunon, z50, gun1;
```

Related information

Application manual - Controller software IRC5

6.23.12 Restrict placing of circlepoints

Parent

Restrict placing of circlepoints belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

restricted_circlepoint

Description

Restrict placing of circlepoints adds a supervision that the circle path not turns around more than 240 degrees and that the circle point is placed in the middle part of the circle path.

Usage

If the program is started on a MoveC instruction and the robot is standing between the circle point and the end point then there is a risk that the robot will perform the circle backwards. That is, move to the circle point and complete the circle to the end point in the opposite direction than programmed. This could be dangerous.

The circle path will be better defined if the circle point is near the midth of the path, for example, use the instructions `CirPathMode\CirPointOri` or `SingArea\Wrist`.

To minimize the risk set *Restrict placing of circlepoints* to Yes. Then the robot will stop with an error message if the TCP is not within the safe limits.

Allowed values

Yes or No.

Default value is Yes.

NOTE! The default value is set to No when loading a system created in RW 5.10 or older releases.

Additional information

The following reasons will stop the robot if *Restrict placing of circlepoints* is set to Yes.

- Circle point is too close to start point.
- Circle point is too close to end point.
- Circle is too large, that is more than 240 degrees.

If a circle point is modified (modpos) then the planned path is recalculated so that when restarting the program the robot will follow the new path if the conditions for restricted placing of circlepoints are fulfilled, regardless of if the function is activated or not.

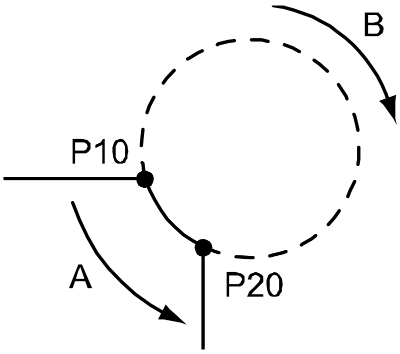
Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

Continues on next page

Example

The example shows a planned path from P10 to P20 in anti clockwise direction (A). If the robot is standing between P10 and P20 when execution is started then the robot might want to use the other direction (B). If *Restrict placing of circlepath* is set to Yes then an error message is displayed that the TCP is not within safe limits.



xx0800000185

6.23.13 Use Motion Supervision

Parent

Use Motion Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

use_motion_sup

Description

Use Motion Supervision defines which set of motion supervision parameters to be used for this motion planner.

Usage

Motion supervision is used to activate, deactivate or adjust the collision detection functionality. For detailed information about collision detection, see the *Application manual - Controller software IRC5*, chapter *Collision Detection*.

Allowed values

A string with maximum 32 characters.

Related information

The Motion Supervision type on page 617.

Application manual - Controller software IRC5.

6.23.14 Motion Supervision Permanent Off

Parent

Motion Supervision Permanent Off belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

motion_sup_permanent_off

Description

Motion Supervision Permanent Off is used to turn off all motion supervision to save CPU power.

Allowed values

Yes

No

6.23.15 Motion Supervision Max Level

Parent

Motion Supervision Max Level belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

motion_sup_max_level

Description

The maximum allowed supervision level, both for program execution and jogging.

Usage

Motion Supervision Max Level stops the operator from tuning the supervision level to values that are too high.

The supervision level for program execution is a combination of the parameter *Path Collision Detection Level* and a tuning value set with the RAPID instruction `MotionSup`. *Motion Supervision Max Level* is a maximum limit for this combined value.

Limitations

Changing this parameter only affects the system if the option *Collision Detection* is installed.

Allowed values

An integer in the interval 10 to 500 (percent).

The default value is 300.

Related information

Path Collision Detection Level on page 621.

Application manual - Controller software IRC5

Example

Motion Supervision Max Level is set to 300.

Path Collision Detection Level is set to 250.

A RAPID program uses the instruction `MotionSup` to tune the supervision level with 200%.

Normally this would lead to a supervision level of 500% (2.5 * 2 = 5), but since *Motion Supervision Max Level* is 300, the supervision level will not exceed 300%.

6.23.16 Remove Corner Path Warning

Parent

Remove Corner Path Warning belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

remove_corner_path_warning

Description

Remove Corner Path Warning is used to disable the corner path failure warnings. Corner warnings will still be executed as fine points but the warning will not be shown in the event log.

Usage

The warning "50024 Corner Path Failure" occurs when RAPID program execution does not provide a new Move instruction while the robot is entering a corner zone. This may be due to a programming oversight or an explicit desire of the programmer.

Allowed values

Yes

No

6.23.17 Time Event Supervision

Parent

Time Event Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

require_event_accuracy

Description

Time Event Supervision is used to detect if a programmed event can be accurately positioned or not. If not, the system will stop and display a warning.

Usage

If the event cannot be accurately positioned, suggested program modifications are to either lower the programmed speed or to increase the distance between the start of the segment and the desired event position.

Allowed values

Yes or No

6.23.18 High Interpolation Priority

Parent

High Interpolation Priority belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

high_interpolation_priority

Description

High Interpolation Priority is used to allow the system to temporarily increase the priority of the path planning in critical situations.

Usage

When the warning "50082 Deceleration limit" occurs at installations, this parameter can be useful. The parameter *Path Resolution* might be useful in this situation.

> **Note**
>
> Using *High Interpolation Priority* might affect the performance of the application, e.g. spot welding or sealing. Thus it is very important to verify the process performance after the parameter has been set.

Allowed values

On or Off

Related information

Path Resolution on page 582.

.

6.23.19 Speed Control Warning

Parent

Speed Control Warning belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

speed_control_warning

Description

By setting *Speed Control Warning* to Yes, a warning will be given when the robot moves slower than the programmed speed.

Usage

When several robots (and other mechanical units) are in synchronized movement mode, in a MultiMove application, all simultaneous move instruction finish at the same time. This means that if one robot has a longer path or a slower programmed speed than another robot, the speed of the second robot is decreased.

If a robot is working with an application where the speed is important (e.g. arc welding or gluing), *Speed Control Warning* can be used to give a warning when the actual speed is slower than the programmed speed.

Limitations

This parameter is only useful when using the RobotWare option MultiMove.

The speed is only supervised for robot TCP speed. No warning is given for the speed of additional axes.

Allowed values

Yes or No.

Additional information

When several tasks are in synchronized movement mode, all these tasks are planned by the same *Motion Planner* (the first *Motion Planner* of those involved in the synchronization). If this *Motion Planner* has *Speed Control Warning* set to Yes, all the synchronized robot speeds are supervised. If it has *Speed Control Warning* set to No, no robot speeds are supervised.

6.23.20 Speed Control Percent

Parent

Speed Control Percent belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

speed_control_percent

Description

If *Speed Control Warning* is set to Yes, a warning will be issued when the actual speed is slower than this percentage of the programmed speed.

Usage

If a robot is working with an application where the speed is important (e.g. arc welding or gluing), *Speed Control Percent* defines the slowest speed (in percent of programmed speed) that is acceptable.

Limitations

This parameter is only useful when using the RobotWare option MultiMove.

The speed is only supervised for robot TCP speed. No warning is given for the speed of additional axes.

Allowed values

A number between 0 and 100 (in percent of programmed speed).

6.23.21 Use spline parameters

Parent

Use spline parameters belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

use_spline_parameters

Description

Use spline parameters defines the value of the spline parameters.

Usage

Use spline parameters defines how long the robot waits when starting from a finepoint, that is, how many positions will be calculated in advance by the motion planner. Default value is *default mp1* for the first robot.

Using *3steps mp1* will give a shorter time when starting from a finepoint. But the robot may stop with the 50024 warning (*Corner zone executed as finepoint*) on the first move.

Allowed values

Following are the allowed values:

- default mp1
- 3steps mp1
- 4steps mp1
- 5steps mp1
- default mp2
- 3steps mp2
- 4steps mp2
- 5steps mp2

NOTE! mp1 stands for motion planner 1, that is, robot 1. mp2 stands for motion planner 2, that is, robot 2.

Limitations

The parameter is valid only for IRB 360.

6.23.22 Use additional interp. object batch

Parent

Use additional interp. object batch belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

extended_dec_dist

Description

Use additional interp. object batch is used to increase the number of interpolation objects available in the system. The value 0 means the default number of interpolation objects is available. Increasing the parameter value by one implies allocating one additional batch of interpolation objects.

Usage

The parameter is useful if *AccSet* is used with very low values or a very slow external axis is used in the system. Typically the value is increased after the error 50426 (*Out of interpolation objects*) is triggered.

Note

The additional interpolation objects use system memory and it is therefore not recommended to add extra safety margin on the number of batches allocated.

Allowed values

A value between 0 and 2 specifying the number of additional batches of interpolation objects that are available in the system.

6.23.23 Bandwidth of path pose filter

Parent

Bandwidth of path pose filter belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

weave_path_pose_filter_bandwidth

Description

Bandwidth of path pose filter is used to set the cut off frequency for a low pass filter that filters the path pose used for weaving. The path pose is constantly calculated from the actual path and the tool Z direction. When this pose changes too rapidly, the robot might jerk slightly or trigger the error message 50375, *Dynamic load too high*. The *Bandwidth of path pose filter* is used to smoothen these changes in the pose.

Usage

Setting this value to a lower value creates a smoother change of the path pose. If a rapid change of pose is needed, a higher value can be set as long as it does not create jerky movements.

Allowed values

A value between 0.01 and 20, specifying the cut off frequency in Hz.

The default value is 1 Hz.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types, instruction *CorrCon*.

6.23.24 Number of Internal Event Objects

Parent

Number of Internal Event Objects belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

number_of_event_objects

Description

Number of Internal Event Objects defines the number of internal event objects for the motion planner.

Usage

The Number of Internal Event Objects is used to allocate internal event objects. The objects are used in different situations, e.g. when running the `Trigg` instructions in RAPID. When using intensive `TriggLIOs` the controller can get lack of internal event objects, in such event this parameter can be used to solve the problem and increase the number of internal objects.

Allowed values

A value between 0 and 500.

Default value is 100.

Related information

RAPID Instructions, Functions and Data types - TriggLIOs

6.23.25 Enable high accuracy position synchronization

Parent

Enable high accuracy position synchronization belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

enable_high_accuracy_pos_sync

Description

Enable high accuracy position synchronization is used to highly increase the position accuracy when synchronization is made between mechanical units with different bandwidth.

The functionality may have slight negative effect on motion start time after finepoint and may lead to minor increase of cycle time.

The functionality is turned off by default.

Usage

Set this value to Yes to improve the position synchronization.

Allowed values

Yes or No.

Default value is No.

Related information

Application manual - Additional axes and stand alone controller.

Technical reference manual - RAPID Instructions, Functions and Data types.

6.23.26 Setup optimized start from finepoint

Parent

Setup optimized start from finepoint belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

optimized_start_from_finepoint

Description

The parameter *Setup optimized start from finepoint* enables the robot to start faster from a finepoint.

Usage

The default value for *Setup optimized start from finepoint* is *Yes*. When the RAPID command `DeactEventBuffer` is used then the optimized start from finepoint functionality is automatically enabled. And if the event buffer is configured and activated using RAPID command `ActEventBuffer`, the optimized start from finepoint functionality is automatically disabled.

Allowed values

Yes or *No*

Default value

The default value is *Yes*.

Related information

Application manual - Additional axes and stand alone controller.

Technical reference manual - RAPID Instructions, Functions and Data types.

6.24 Type Motion Process Mode

6.24.1 The Motion Process Mode type

Overview

This section describes the type *Motion Process Mode*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MOTION_PROCESS_MODE

Type description

A *Motion Process Mode* consists of a specific set of tuning parameters for a robot. Each *Motion Process Mode* tuning parameter set optimizes the robot tuning for a specific class of applications. There are three predefined modes:

- *Optimal cycle time mode* – this is the default mode and gives the same tuning as the standard robot tuning in RobotWare releases prior to 6.0. This mode gives the shortest possible cycle time.
- *Low speed accuracy mode* – this mode is recommended for applications where path accuracy is important, and for process speeds up to approximately 500 mm/s. The cycle time will be increased compared to *Optimal cycle time mode*.
- *Low speed stiff mode* - this mode is recommended for contact applications where maximum servo stiffness is important. It could also be used in some low speed applications, where a minimum of path vibrations is desired. The cycle time will be increased compared to *Low speed accuracy mode*.

The concept of *Motion Process Mode* simplifies application specific tuning which previously has been performed by using *TuneServo* and *AccSet* in the *RAPID* program. The three predefined modes should be useful in many cases with no further adjustments needed.

The *TuneServo* and *AccSet* instructions can still be used for adjusting the tuning but it is recommended to use *Motion Process Mode* instead.

If a more specific tuning is needed, some tuning parameters can be modified in each *Motion Process Mode*. These parameters are described in the following. In this way, the user can create a specific tuning for a specific application. Note that all parameter settings are relative adjustments of the predefined parameter value.

Relative adjustment of acceleration =
predefined_accset_acc_factor_for_specific_mode x accset_acc_factor x
acc_factor_of_accset_instruction / 100

The *Motion Process Mode* can be changed by changing the parameter *Use Motion Process Mode* for type *Robot*.

Continues on next page

Limitations

Changing the *Motion Process Mode* from RAPID is only possible if the RobotWare option *Advanced Robot Motion* is installed. The *Motion Process Mode* concept is currently available for all six-axes robots except paint robots. The *Mounting Stiffness Factor* parameters are only available for IRB 120, IRB 140, IRB 1200, IRB 1520, IRB 1600, IRB 2600, IRB 4600, IRB 6620 (not LX), IRB 6640, IRB 6700. For IRB 1410, only the three *AccSet* parameters are available.

Related information

Application manual - Controller software IRC5

Technical reference manual - RAPID Instructions, Functions and Data types

WARNING

Incorrect use of *Motion Process Mode* parameters can cause movements and torques that can damage the robot. You must bear this in mind when setting the *Motion Process Mode* parameters.

6.24.2 Name

Parent

Name belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the motion process mode.

Allowed values

A string with maximum 32 characters.

6.24.3 Accset Acc Factor

Parent

Accset Acc Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

accset_acc_factor

Description

Accset Acc Factor changes the acceleration.

Usage

Accset Acc Factor = 0.8 reduces the acceleration by 20%, *Accset Acc Factor* = 1.5 increases the acceleration by 50%. For Optimal cycle time mode, the acceleration is the highest possible and values above 1.0 will not affect the acceleration. Decreased acceleration increases cycle time but reduces path errors, vibrations, and overshoots.

Allowed values

A numeric value between 0.1 and 5.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.4 Accset Ramp Factor

Parent

Accset Ramp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

accset_ramp_factor

Description

Accset Ramp Factor changes the acceleration ramp time (jerk).

Usage

Accset Ramp Factor = 0.5 increases the acceleration ramp time by a factor of 2.
Accset Ramp Factor = 0.2 increases the acceleration ramp time by a factor of 5.
Increased acceleration ramp time, increases cycle time but reduces path errors, vibrations, and overshoots. In most cases, the *Accset Acc Factor* is more efficient for obtaining this and should therefore be the first choice.

Allowed values

A numeric value between 0.1 and 1.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.5 Accset Fine Point Ramp Factor

Parent

Accset Fine Point Ramp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

accset_fp_ramp_factor

Description

Accset Fine Point Ramp Factor changes the deceleration ramp time (jerk) when moving into a fine point.

Usage

Accset Fine Point Ramp Factor = 0.5 increases the deceleration ramp time by a factor of 2, when moving into a fine point. *Accset Fine Point Ramp Factor* = 0.2 increases the deceleration ramp time by a factor of 5. Increased deceleration ramp time in fine point increases cycle time for each fine point but reduces vibrations and overshoots in fine points, and is a more cycle time efficient way to solve such problems (compared to using *Accset Acc Factor* or *Accset Ramp factor*).

Allowed values

A numeric value between 0.1 and 1.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.6 Dh Factor

Parent

Dh Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

dh_factor

Description

Dh factor affects the smoothness of the robot path by adjusting the effective bandwidth of the mechanical unit.

Usage

A *Dh Factor* less than 1 decreases the effective bandwidth of the mechanical unit and increases the smoothness of the robot path. For *Optimal cycle time mode*, the bandwidth is the highest possible and values above 1.0 will not affect the path.

Decreased bandwidth reduces overshoots and path errors due to vibrations. However, at high speed, larger corner zones than programmed will be noticeable.

A decreased *Dh Factor* increases cycle time for each fine point only. Thus, *Dh Factor* is a more cycle time efficient way to reduce vibrations and overshoots than the use of *Accset Acc Factor*.

Allowed values

A numeric value between 0.1 and 5.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.7 Df Factor

Parent

Df Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

df_factor_1, df_factor_2, df_factor_3, df_factor_4, df_factor_5, df_factor_6

Description

Df Factor affects the predicted mechanical resonance frequency of a particular axis.

Usage

Df Factor = 0.95 reduces the predicted mechanical resonance frequency of a particular axis by 5%. The most common use of *Df Factor* is to compensate for a foundation with inadequate stiffness, i.e., a flexible foundation. In this case, the *Df Factor* for axis 1 and 2 is lowered, typically to a value between 0.80 and 0.99. Use of *Df Factor* for axis 3 – 6 is rare and is normally not recommended. *Df Factor* for axis 1 and 2 can be automatically tuned by using TuneMaster. Correctly adjusted, not too low and not too high, *Df Factor* reduces vibrations and overshoots, without affecting cycle time. For robots where *Mounting Stiffness Factor* is available, *Df Factor* is not needed for compensation of flexible foundations.

Allowed values

A numeric value between 0.1 and 1.5.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

Mounting Stiffness Factor X, Mounting Stiffness Factor Y, Mounting Stiffness Factor Z on page 616

6.24.8 Kp Factor

Parent

Kp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

kp_factor_1, kp_factor_2, kp_factor_3, kp_factor_4, kp_factor_5, kp_factor_6

Description

Kp Factor affects the equivalent gain of the position controller.

Usage

An increased *Kp Factor* can reduce path errors and increases the servo stiffness. However, oscillations due to mechanical resonances can be increased in some cases. In most cases where the position or speed controller parameters (*Kp Factor*, *Kv Factor* and *Ti Factor*) need to be changed, *Kv Factor* is the most important parameter and *Kp Factor* is not changed.

Allowed values

A numeric value between 0.2 and 5.0.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.9 Kv Factor

Parent

Kv Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

kv_factor_1, kv_factor_2, kv_factor_3, kv_factor_4, kv_factor_5, kv_factor_6

Description

Kv Factor affects the equivalent gain of the speed controller

Usage

An increased *Kv Factor* can reduce path errors due to, e.g., drive train ripple and friction. An increased *Kv Factor* also increases the servo stiffness. However, oscillations due to mechanical resonances can be increased in some cases. A *Kv Factor* which is too high causes motor vibrations and must be avoided. Always be careful and be observant for increased motor noise level when adjusting *Kv Factor* and do not use higher values than needed for fulfilling the application requirement.

Allowed values

A numeric value between 0.2 and 5.0.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.10 Ti Factor

Parent

Ti Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

ti_factor_1, ti_factor_2, ti_factor_3, ti_factor_4, ti_factor_5, ti_factor_6

Description

Ti Factor affects the integral time of the controller.

Usage

A decreased *Ti Factor* can reduce path errors and increases the servo stiffness. However, oscillations due to mechanical resonances can be increased in some cases. In most cases where the controller parameters (*Kp Factor*, *Kv Factor* and *Ti Factor*) need to be changed, *Kv Factor* is the most important parameter and *Ti Factor* is not changed.

Allowed values

A numeric value between 0.1 and 5.0.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.24.11 Mounting Stiffness Factor X, Mounting Stiffness Factor Y, Mounting Stiffness Factor Z

Parent

Mounting Stiffness Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Cfg name

mounting_stiffness_factor_x, mounting_stiffness_factor_y, mounting_stiffness_factor_z

Description

Mounting stiffness factor describes the stiffness of the robot foundation.

Usage

Mounting Stiffness Factor can be used for compensating for a foundation with inadequate stiffness, i.e., a flexible foundation. Correctly tuned *Mounting Stiffness Factor* will minimize overshoots and reduce vibrations. *Mounting Stiffness Factor* = 1.0 is default and give the best behavior when the foundation is stiff according to the Robot Product Manual (see, requirement on foundation - minimum resonance frequency). A lower value will improve the robot behavior when the requirement on foundation is not fulfilled and a lower value means a more flexible foundation. There are three parameters for the x-, y-, and z-direction (torsional stiffness in base coordinate system). *Mounting Stiffness Factor* can be automatically tuned by TuneMaster.

Allowed values

A numeric value between 0.01 and 1.0333.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.25 Type Motion Supervision

6.25.1 The Motion Supervision type

Overview

This section describes the type *Motion Supervision*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MOTION_SUP

Type description

Motion supervision is used to activate, deactivate or adjust the collision detection functionality. For detailed information about collision detection, see the *Application manual - Controller software IRC5*, chapter *Collision Detection*.

No controller restart required

Most of the motion supervision parameters do not require a restart of the controller when modified.

Limitations

The type *Motion supervision* is mainly used to configure the installed option *Collision detection*. For a system without this option, changing the values for most of the parameters does not affect the system.

Related information

How to tune motion supervision on page 392.

Application manual - Controller software IRC5.

6.25.2 Name

Parent

Name belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the motion supervision setup.

Limitation

This parameter cannot be changed.

Related information

How to tune motion supervision on page 392.

6.25.3 Path Collision Detection

Parent

Path Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

path_col_detect_on

Description

Path Collision Detection turns the collision detection on or off for program execution.

Usage

Setting *Path Collision Detection* to On turns on the collision detection, Off turns off the collision detection.

Allowed values

On or Off

Related information

How to tune motion supervision on page 392.

6.25.4 Jog Collision Detection

Parent

Jog Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

jog_col_detect_on

Description

Jog collision Detection turns the collision detection on or off for jogging.

Limitation

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

On or Off

Related information

How to tune motion supervision on page 392.

6.25.5 Path Collision Detection Level

Parent

Path Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

path_col_detect_level

Description

Path Collision Detection Level modifies the supervision level for the collision detection for program execution by a specified percentage value.

Usage

The supervision level for collision detection in program execution is specified as a percentage. A large value makes the function less sensitive. The default value is 100%. For detailed information, see the *Application manual - Controller software IRC5*.

Limitation

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

A value in the interval 1 to 500, specifying the supervision level in %.

The default value is 100%.

Related information

How to tune motion supervision on page 392.

Application manual - Controller software IRC5

6.25.6 Jog Collision Detection Level

Parent

Jog Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

jog_col_detect_level

Description

Jog Collision Detection Level modifies the supervision level for the collision detection for jogging by a specified percentage value.

Usage

The supervision level for collision detection in jogging is specified as a percentage, where a large value makes the function less sensitive. The default value is 100%. For detailed information, see the *Application manual - Controller software IRC5*.

Limitations

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

A value in the interval 1 to 500, specifying the supervision level in %.

The default level is 100%.

Related information

How to tune motion supervision on page 392.

Application manual - Controller software IRC5.

6.25.7 Collision Detection Memory

Parent

Collision Detection Memory belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

collision_detection_memory

Description

Collision Detection Memory defines how much the robot moves back on the path after a collision.

The parameter requires a restart of the controller when modified.

Usage

The robot movement back on the path after a collision is specified in seconds. If the robot was moving quickly before the collision, it will move further back than if the speed was lower. For detailed information, see the *Application manual - Controller software IRC5*.

Allowed values

A value in the interval 0.025 to 0.5, specifying the movement in seconds.

Related information

How to tune motion supervision on page 392.

Application manual - Controller software IRC5.

6.25.8 Manipulator supervision

Parent

Manipulator supervision belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

manipulator_supervision_on

Description

Manipulator supervision turns the supervision for the loose arm detection on or off for IRB340 and IRB 360.

Usage

Set *Manipulator supervision* to On to turn supervision on. The supervision level is set with parameter *Manipulator supervision level*. A loose arm will stop the robot and cause an error message.

Limitations

Changing this parameter affects the system only if the option *Collision detection* is installed.

For the changes to take effect, a restart is required.

The *Manipulator supervision* parameter is used only by IRB 340 and IRB 360.

Allowed values

On or Off

The default value is Off.

Related information

Application manual - Controller software IRC5

6.25.9 Manipulator supervision level

Parent

Manipulator supervision level belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

manipulator_supervision_level

Description

Manipulator supervision level modifies the supervision level for the loose arm detection for the manipulators IRB 340 and IRB 360.

Usage

The supervision level for loose arms is specified as a percentage, where a large value makes the function less sensitive. The default value is 100%.

The supervision function is turned On or Off with parameter *Manipulator supervision*.

Limitations

Changing this parameter only affects the system if the option *Collision detection* is installed.

For the changes to take effect, a restart is required.

The parameter *Manipulator supervision level* is used only by IRB 340 and IRB 360.

Allowed values

A value in the interval 1 to 500, specifying the supervision level in %.

The default value is 100%.

Related information

Application manual - Controller software IRC5

6.26 Type Motion System

6.26.1 The Motion System type

Overview

This section describes the type *Motion System*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MOTION_SYSTEM

Type description

Motion System includes parameters that are common for the entire system.

Non-editable parameters

The following parameters are visible but not editable in the software configuration tools:

- *Sensor Memory Mode*
- *SMB memory update time*

As a consequence, the above parameters are not described in the manual.

6.26.2 Name

Parent

Name belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the *Motion System* type.

Allowed values

A string with maximum 32 characters.

6.26.3 Min Temperature Cabinet

Parent

Min Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

min_temp_ambient_cabinet

Description

Min Temperature Cabinet defines the minimum ambient temperature where the cabinet is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6.26.4 Max Temperature Cabinet

Parent

Max Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

max_temp_ambient_cabinet

Description

Max Temperature Cabinet defines the maximum ambient temperature where the cabinet is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

Additional information

This parameter does not have to be changed if the controller is equipped with an extra fan for the cabinet.

6.26.5 Min Temperature Robot

Parent

Min Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

min_temp_ambient_robot

Description

Min Temperature Robot defines the minimum ambient temperature where the robot is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6.26.6 Max Temperature Robot

Parent

Max Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

max_temp_ambient_robot

Description

Max Temperature Robot defines the maximum ambient temperature where the robot is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6.26.7 Coll-Pred Safety Distance

Parent

Coll-Pred Safety Distance belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

coll_pred_default_safety_distance

Description

The parameter *Coll-Pred Safety Distance* determines at what distance two geometric objects (for example robot-links) are considered to be in collision.

Allowed values

A value between 0.001 to 1 meter.

Default value is 0.001 meter.

Related information

Collision Avoidance on page 299

6.27 Type Motor

6.27.1 The Motor type

Overview

This section describes the *Motor* type which belongs to the topic *Motion*. Each parameter is described in a separate information topic in this section.

Cfg name

MOTOR

Type description

The type *Motor* describes the motor used for each axis. There is one configuration of the type *Motor* for each axis.

Note that only external axes are visible, the robot's axes motors are configured on delivery and should not be changed.

6.27.2 Name

Parent

Name belongs to the type *Motor*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the motor.

Allowed values

A string with maximum 32 characters.

6.27.3 Use Motor Type

Parent

Use Motor Type belongs to the type *Motor*, in the topic *Motion*.

Cfg name

use_motor_type

Description

Use Motor Type defines which type of motor is used for this type.

Usage

The type *Motor Type* defines the motor data.

Allowed values

A string with maximum 32 characters.

Related information

The type Motor Type on page 644.

6.27.4 Use Motor Calibration

Parent

Use Motor Calibration belongs to the type *Motor*, in the topic *Motion*.

Cfg name

use_motor_calib

Description

Use Motor Calibration defines which type of motor calibration to be used.

Usage

The type *Motor Calibration* defines the motor's calibration data.

Allowed values

A string with maximum 32 characters.

Related information

The Motor Calibration type on page 637.

6.28 Type Motor Calibration

6.28.1 The Motor Calibration type

Overview

This section describes the type *Motor Calibration*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MOTOR_CALIB

Type description

With the parameters in the *Motor Calibration* type, you can calibrate the robot's motors by entering the calibration values.

Motor calibration configuration is normally done during robot calibration. However, if the values are known, they can be specified directly.

Limitations

If calibration or commutator offset parameters are set, the corresponding offset valid parameters have to be set to YES, otherwise the offset parameter will not be used.

6.28.2 Name

Parent

Name belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the motor calibration setting it belongs to.

Usage

Name is used to reference the *Motor Calibration* from the parameter *Use Motor Calibration* in the type *Motor*.

Allowed values

A string with maximum 32 characters.

6.28.3 Commutator Offset

Parent

Commutator Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

com_offset

Description

Commutator Offset defines the position of the motor (resolver) when the rotor is in the predefined commutation position relative to the stator.

Usage

ABB motors normally uses *Commutation Offset* value 1.57080.

Allowed values

A value between -6.283186 and 6.283186, specifying the offset in radians.

6.28.4 Commutator Offset Valid

Parent

Commutator Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

valid_com_offset

Description

Commutator Offset Valid specifies whether the commutator offset is defined or not.

Allowed values

Yes or No.

Related information

Commutator Offset on page 639.

6.28.5 Calibration Offset

Parent

Calibration Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

cal_offset

Description

Calibration Offset defines the position of the motor (resolver) when the arm is in the calibration (zero) position.

Allowed values

A value between -6.283186 and 6.283186, specifying the offset in radians.

6.28.6 Calibration Offset Valid

Parent

Calibration Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

valid_cal_offset

Description

Calibration Offset Valid specifies whether the calibration offset is defined or not.

Allowed values

Yes or No.

Related information

Calibration Offset on page 641.

6.28.7 Calibration Sensor Position

Parent

Calibration Sensor Position belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

cal_sensor_position

Description

Calibration Sensor Position defines the calibration sensor position on the arm side.

Usage

The value is set in degrees.

Allowed values

A value between -180 and 180 degrees.

Default value is 0 degrees.

6.29 Type Motor Type

6.29.1 The type Motor Type

Overview

This section describes the type *Motor Type*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MOTOR_TYPE

Type description

The type *Motor Type* is used to describe characteristics for the motor.

Limitations

The parameter values for *Motor Type* can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

6.29.2 Name

Parent

Name belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Motor Type*.

Usage

Name is used to reference a motor type from the parameter *Use Motor Type* in the type *Motor*.

Allowed values

A string with maximum 32 characters.

6.29.3 Pole Pairs

Parent

Pole Pairs belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

pole_pairs

Description

Defines the number of pole pairs for the motor type.

Usage

Set *Pole Pairs* to the number of pole pairs (i.e. number of poles divided with 2) that the motor has.

Limitations

Pole Pairs can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

An integer between 0 and 20.

6.29.4 Stall Torque

Parent

Stall Torque belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

torque_0

Description

The continuous stall torque, i.e. the torque the motor can produce at no speed and during an infinite time.

Usage

Set *Stall Torque* to the stall torque (T_0) specified by the motor manufacturer.

Limitations

Stall Torque can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100000 (Nm).

6.29.5 ke Phase to Phase

Parent

ke Phase to Phase belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

ke

Description

Nominal voltage constant.

Usage

ke Phase to Phase is the induced voltage (phase to phase) that corresponds to the speed 1 rad/s.

Limitations

ke Phase to Phase can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 10 (Vs/rad).

Additional information

Some motor manufacturers specify the value *kt* instead of *ke*. *ke* can then be calculated according to the formula:

ke = kt / ÷3

6.29.6 Max Current

Parent

Max Current belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

i_max

Description

Max current without irreversible magnetization.

Usage

Set *Max Current* to the root-mean-square of the maximum current the motor can withstand without irreversible demagnetization.

Limitations

Max Current can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (A rms).

6.29.7 Phase Resistance

Parent

Phase Resistance belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

r_stator_20

Description

Nominal winding resistance per phase at 20 degrees Celsius.

Usage

Set *Phase Resistance* to the stator phase resistance (R_{20}) specified by the motor manufacturer.

Limitations

Phase Resistance can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (ohm).

6.29.8 Phase Inductance

Parent

Phase Inductance belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

l_stator

Description

Nominal winding inductance per phase at zero current.

Usage

Set *Phase Inductance* to the stator phase inductance (L_0) specified by the motor manufacturer.

Limitations

Phase Inductance can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (H).

6.30 Type Path Sensor Synchronization

6.30.1 The Path Sensor Synchronization type

Parent

This section describes the type *Path Sensor Synchronization* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

PATH_SENSOR_SYNC

Type description

The type *Path Sensor Synchronization* define settings for sensor synchronization. The parameters of this type are used to set limits for the movements of a robot that is synchronized with an external device. Limits can be set for allowed deviation between calculated and actual position, and minimum/maximum TCP speed.

Limitations

Path Sensor Synchronization can only be used if you have the option *Sensor synchronization* installed.

Related information

Application manual - Controller software IRC5, chapter *Sensor synchronization*.

6.30.2 Name

Parent

Name belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name for the path sensor synchronization.

Allowed values

A string with maximum 32 characters.

6.30.3 Max Advance Distance

Parent

Max Advance Distance belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

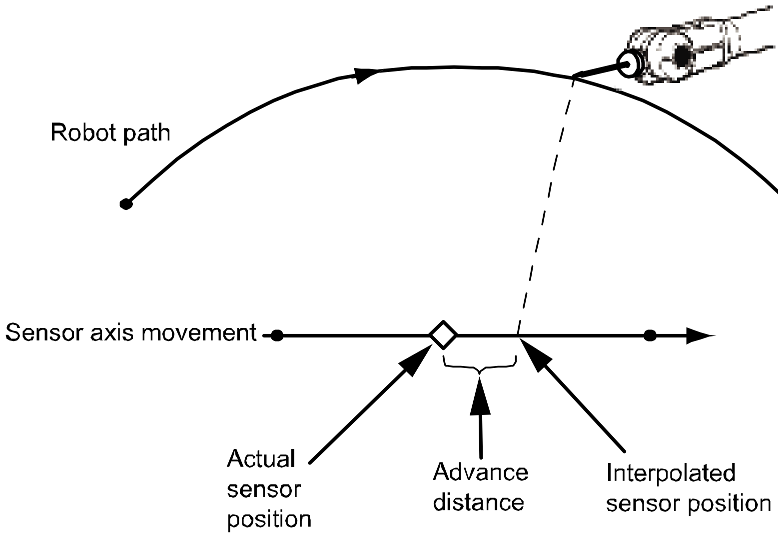
Cfg name

max_adv_dist_for_decel

Description

Max Advance Distance defines the maximum allowed advance distance between the sensor's interpolated position and its actual position.

The interpolated position of the sensor axis corresponds to the robot's position along its path when the robot is synchronized with the sensor.



en0400001243

Usage

If the interpolated position of the sensor axis is ahead of the actual position, a collision may occur. For example, if the robot enter a press based on the information that the press is open, but the press is actually still closed, the robot may move into the closed press. This can be avoided by using *Max Advance Distance*. If *Max Advance Distance* is exceeded, motion and execution is stopped.

Limitations

Max Advance Distance can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A value between 0.01 and 5.0 (meters of movement on the external device that is connected to the sensor).

Default value is 0.1.

6.30.4 Max Delay Distance

Parent

Max Delay Distance belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

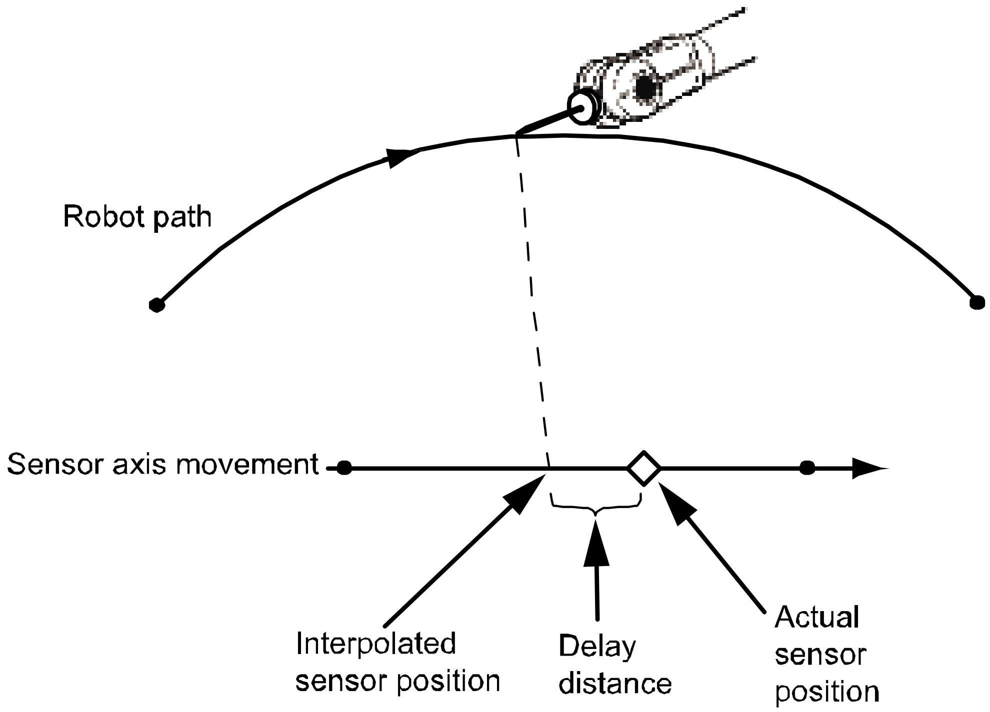
Cfg name

max_delay_dist_for_decel

Description

Max Delay Distance defines the maximum allowed delay distance between sensor's interpolated position and its actual position.

The interpolated position of the sensor axis corresponds to the robot's position along its path when the robot is synchronized with the sensor.



en0400001244

Usage

If the interpolated position of the sensor axis is behind the actual position, a collision may occur. A robot that is moving in an area where the external device will be later in the cycle can collide with the external device because of the incorrect timing. This can be avoided by using *Max Delay Distance*. If *Max Delay Distance* is exceeded, motion and execution is stopped.

Max Delay Distance can be disabled by setting its value to 0.

Limitations

Max Delay Distance can only be used if you have the option *Sensor synchronization* installed.

Continues on next page

Allowed values

A numeric value between 0.0 and 5.0 (meters of movement on the external device that is connected to the sensor).

Default value is 0, which means that the supervision of the delay distance is not used.

6.30.5 Max Synchronization Speed

Parent

Max Synchronization Speed belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

max_sync_speed

Description

Max Synchronization Speed defines the maximum allowed robot TCP speed during synchronization with an external device.

Usage

If the external device (that the robot is synchronized with) moves so fast that the robot should exceed *Max Synchronization Speed*, the robot speed will be limited to *Max Synchronization Speed*. The robot will slip behind, and the interpolated sensor position will be delayed compared to the actual sensor position, until the *Max Delay Distance* is reached.

Limitations

Max Synchronization Speed can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A numeric value between 1.0 and 10.0 (m/s).

Default value is 4.0.

6.30.6 Min Synchronization Speed

Parent

Min Synchronization Speed belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

min_sync_speed

Description

Min Synchronization Speed defines the minimum allowed robot TCP speed during synchronization with an external device.

Usage

If the external device (that the robot is synchronized with) stops, the robot speed will maintain the *Max Synchronization Speed*. The robot will move ahead, and the interpolated sensor position will be in advance compared to the actual sensor position, until the *Max Advance Distance* is reached.

Limitations

Min Synchronization Speed can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A value between 0.0 and 2.0 (m/s).

Default value is 0.1.

6.30.7 Synchronization Type

Parent

Synchronization Type belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

sync_type

Description

Synchronization Type defines what type of synchronization to be used.

Limitations

Synchronization Type can only be used if you have the option *Sensor synchronization* installed.

Allowed values

Value:	Description:
MINIMAL_DIST	Synchronization based on distance, actual sensor position in corvec.
NOM_SPEED_SENS	Synchronization based on nominal sensor speed, actual sensor position in corvec.
NOM_SPEED_CALC	Synchronization based on nominal sensor speed, calculated sensor position in corvec.
MIN_DIST_CALC	Synchronization based on distance, calculated sensor position in corvec.
LOW_SPEED_SYNC	When robot and sensor speed is lower than 0.2 m/sec.
ROBOT_TO_ROBOT	To synchronize two robots through DeviceNet bus.
ROBOT_TO_PRESS	To synchronize robot with press moved by electric motor.
ROBOT_TO_HPRESS	To synchronize robot with hydraulic press.
SYNC_TO_IMM	To synchronize with injection moulding machine.
HIGH_SPEED_SYNC	To synchronize inside the press for load and unload operation.

6.31 Type Process

6.31.1 The Process type

Overview

This section describes the type *Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

PROCESS

Type description

A process can be called from the parameter *Use Process* in the type *Joint*. The parameters in the type *Process* point out a process in the type *Linked M Process* or *SG Process* that will be used for that joint.

Related information

Use Process on page 520.

The Linked M Process type on page 544.

6.31.2 Name

Parent

Name belongs to the type *Process*, in the topic *Motion*.

Cfg name

name

Description

Name defines the identity of the process.

Usage

The *Name* of the process is used by a joint to call the process.

The process calls a linked motor process (type *Linked M Process*) or a servo gun process (type *SG Process*).

Limitations

This parameter is useful only if you have either of the RobotWare base functionality *Electronically Linked Motors* or option *Spot Servo*.

Allowed values

A string.

6.31.3 Use SG Process

Parent

Use SG Process belongs to the type *Process*, in the topic *Motion*.

Cfg name

use_sg_process

Description

Use SG Process defines which *SG Process* to use.

Usage

Use SG Process refers to a process ID defined by the parameter *Name* in the type *SG Process*.

SG Process is used to define a servo tool's behavior.

Limitations

SG Process can only be used for servo tools.

Allowed values

A string.

6.31.4 Use Linked Motor Process

Parent

Use Linked Motor Process belongs to the type *Process*, in the topic *Motion*.

Cfg name

use_linked_m_proc

Description

Use Linked Motor Process defines which linked motor process to use.

Usage

Use Linked Motor Process points to a process ID defined by the parameter *Name* in the type *Linked M Process*.

The linked motor process is used to define a joint's behavior for *Electronically Linked Motors*.

Allowed values

A string.

6.32 Type Relay

6.32.1 The Relay type

Overview

This section describes the type *Relay* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

RELAY

Type description

The type *Relay* defines the characteristics of the relays that are used for the mechanical units, e.g. brake relays and run relays.

All relays for a robot supplied from ABB are defined on delivery. This means that adding or editing parameters of the *Relay* type is only necessary when additional axes are installed.

Related information

Application manual - Additional axes and stand alone controller.

6.32.2 Name

Parent

Name belongs to the type *Relay*, in the topic *Motion*.

Cfg name

name

Description

The name of the Relay.

Usage

Name is used to refer a Relay from the parameters *Use Activation Relay*, *Use Brake Relay*, and *Use Connection Relay* in the type *Mechanical Unit*.

Allowed values

A string with maximum 32 characters.

6.32.3 Output Signal

Parent

Output Signal belongs to the type *Relay* in the topic *Motion*.

Cfg name

Out_signal

Description

Output Signal defines the logical name of the output signal to the relay.

Usage

Characteristics of relays for manipulators need to be defined when additional axes are installed.

The value of *Output Signal* must be identical to the name of the signal, including upper and lower case letters.

Prerequisites

The logical signal name must be defined in the type *Signal* in the topic *I/O*.

Allowed values

A string with maximum 32 characters.

Related information

The Signal type on page 237.

6.32.4 Input Signal

Parent

Input Signal belongs to the type *Relay* in the topic *Motion*.

Cfg name

in_signal

Description

Input Signal defines the logical name of the input signal to the relay.

Usage

Characteristics of relays for manipulators need to be defined when additional axes are installed.

The value of *Input Signal* must be identical to the name of the signal, including upper and lower case letters.

Prerequisites

The logical signal name must be defined in the type *Signal* in the topic *I/O*.

The signal must be defined as "safety" and "INTERNAL".

Allowed values

A string with maximum 32 characters.

Related information

The Signal type on page 237.

6.33 Type Robot

6.33.1 The Robot type

Overview

This section describes the type *Robot* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ROBOT

Type description

The type *Robot* contains a number of parameters that are common for a robot in the robot system. The robot is a mechanical unit with more than one joint. Parameters of this type are used to define which joints the robot consists of and the base frame of the robot.

6.33.2 Name

Parent

Name belongs to the type *Robot*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the robot.

Limitations

This parameter cannot be changed.

6.33.3 Use Robot Type

Parent

Use Robot Type belongs to the type *Robot*, in the topic *Motion*.

Cfg name

use_robot_type

Description

Use Robot Type defines what robot type is used. The parameter contains information about robot reach (m) and handling capacity (kg).

Allowed values

A string with maximum 32 characters.

6.33.4 Use Old SMB

Parent

Use Old SMB belongs to the type *Robot*, in the topic *Motion*.

Cfg name

use_old_smb

Description

To adapt earlier robot systems, running earlier SMB board versions without flash memory, to later software versions, the parameter *Use Old SMB* is to be set to Yes.

Usage

Earlier systems, in this context, is any robot system delivered with an SMB board of any of these revisions:

- DSQC 313, all revisions
- DSQC 520, revision 5 and earlier
- DSQC 562, revision 2 and earlier

Allowed values

Yes or No.

6.33.5 Use Robot Calibration

Parent

Use Robot Calibration belongs to the type *Robot*, in the topic *Motion*.

Cfg names

use_robot_calib

Description

Use Robot Calibration defines if Absolute Accuracy is active for the robot.

Usage

Set *Use Robot Calibration* to "r1_calib" to activate Absolute Accuracy for the robot.

In a MultiMove system, set the value for robot 2 to "r2_calib", robot 3 to "r3_calib" and robot 4 to "r4_calib".

Allowed values

Value (robot 1)	Value (robot 2)	Value (robot 3)	Value (robot 4)	Description
r1_calib	r2_calib	r3_calib	r4_calib	Absolute Accuracy is activated for the robot.
r1_uncalib	r2_uncalib	r3_uncalib	r4_uncalib	Absolute Accuracy is deactivated for the robot.
not_used_uncalib	not_used_uncalib	not_used_uncalib	not_used_uncalib	Absolute Accuracy is deactivated for the robot. Should only be used if no other value is selectable.

Related information

Absolute Accuracy is described in *Application manual - Controller software IRC5*.

6.33.6 Use Joint 1, 2, 3, 4, 5, 6

Parent

Use Joint 1, *Use Joint 2*, *Use Joint 3*, *Use Joint 4* , *Use Joint 5*, and *Use Joint 6* belong to the type *Robot*, in the topic *Motion*.

Cfg names

use_joint_0

use_joint_1

use_joint_2

use_joint_3

use_joint_4

use_joint_5

Description

Use joint 1 defines which joint data to use as the robot's first joint.

Use joint 2 defines which joint data to use as the robot's second joint.

Use joint 3 defines which joint data to use as the robot's third joint.

Use joint 4 defines which joint data to use as the robot's fourth joint.

Use joint 5 defines which joint data to use as the robot's fifth joint.

Use joint 6 defines which joint data to use as the robot's sixth joint.

Usage

The joints are defined in the type *Joint*.

Allowed values

A string with maximum 32 characters, specifying an already defined joint.

Related information

The Joint type on page 516.

6.33.7 Base Frame x, y, z

Parent

Base Frame x, *Base Frame y*, and *Base Frame z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

base_frame_pos_x

base_frame_pos_y

base_frame_pos_z

Description

Base Frame x defines the x-direction of the base frame position in relation to the world frame (in meters).

Base Frame y defines the y-direction of the base frame position in relation to the world frame (in meters).

Base Frame z defines the z-direction of the base frame position in relation to the world frame (in meters).

Allowed values

A value between -1000 to 1000, specifying the relation in meters.

Related information

How to define base frame on page 381.

6.33.8 Base Frame q1, q2, q3, q4

Parent

Base Frame q1, *Base Frame q2*, *Base Frame q3*, and *Base Frame q4* belongs to the type *Robot*, in the topic *Motion*.

Cfg name

base_frame_orient_u0

base_frame_orient_u1

base_frame_orient_u2

base_frame_orient_u3

Description

Base Frame q1 defines the first quaternion (q1) of the base frame orientation in relation to the world frame.

Base Frame q2 defines the second quaternion (q2) of the base frame orientation in relation to the world frame.

Base Frame q3 defines the third quaternion (q3) of the base frame orientation in relation to the world frame.

Base Frame q4 defines the fourth quaternion (q4) of the base frame orientation in relation to the world frame.

Allowed values

A value between -1 to 1 specifying the orientation.

Related information

How to define base frame on page 381.

6.33.9 Base Frame Moved by

Parent

Base Frame Moved by belongs to the type *Robot*, in the topic *Motion*.

Cfg name

base_frame_coordinated

Description

Base Frame Moved by defines the name of robot or single that moves the base frame of the robot.

Allowed values

A string with maximum 32 characters.

Related information

How to define base frame on page 381.

6.33.10 Gravity Alpha

Parent

Gravity Alpha belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gravity_alpha

Description

Gravity Alpha defines the orientation of the robot with respect to the gravity.

Usage

The *Gravity Alpha* is a positive rotation of the robot around the X-axis in the base coordinate system to define the robot orientation relative to the gravity. The value is set in radians.

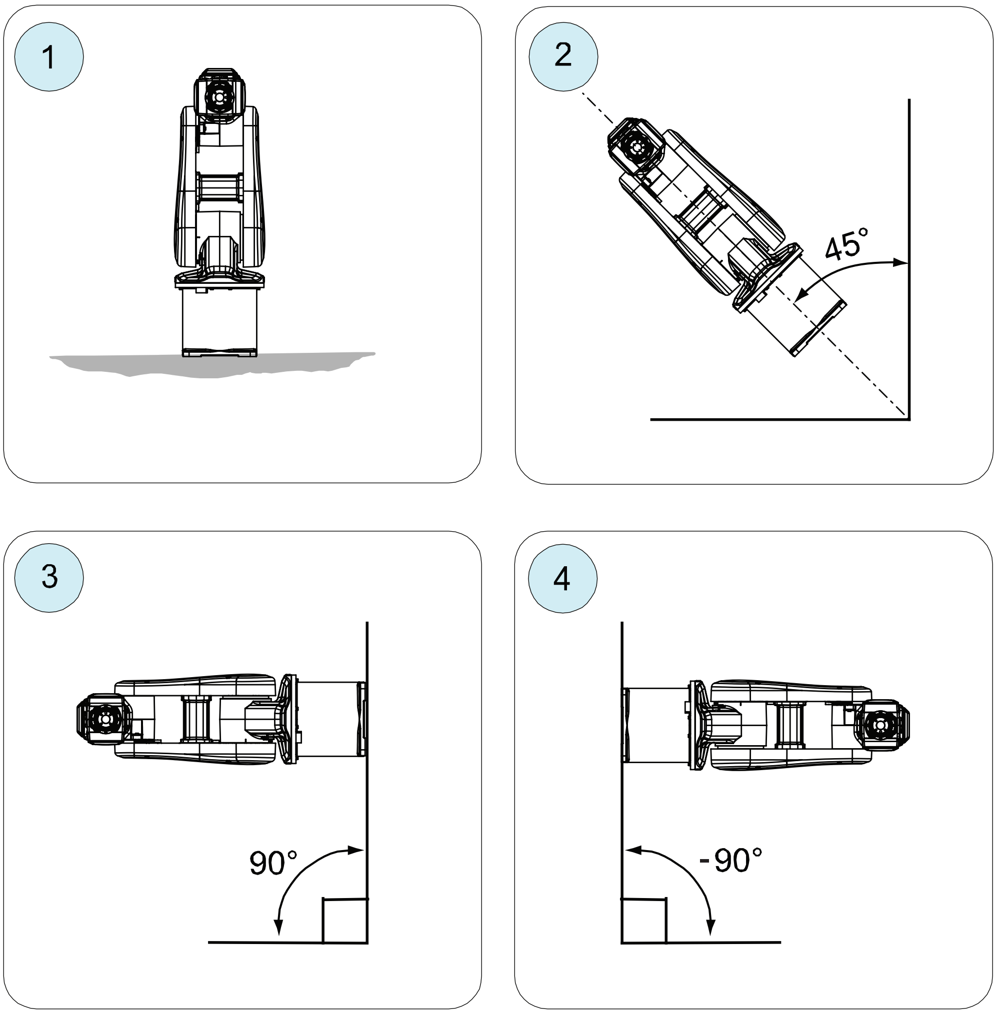
If the robot is mounted on a wall (rotated around the X-axis) the robot base frame and *Gravity Alpha* needs to be changed to reflect the installation. Gravity Alpha should then be ± π/2 (1.570796). For more information about base frame refer to *How to define base frame on page 381*.

Gravity Alpha is calculated in the following way:

`Gravity Alpha = A° x 3.141593/180 = B radians`, where `A` is the mounting angle in degrees and `B` is the mounting angle in radians.

Continues on next page

Examples



xx1500000532

Pos	Mounting angle	Gravity Alpha
1	0° (Floor mounted)	0
2	45° (Tilted)	0.785398
3	90° (Wall)	1.570796
4	-90° (Wall)	-1.570796

Note

For suspended robots (180°), it is recommended to use *Gravity Beta* instead of *Gravity Alpha*.

Prerequisites

The *Gravity Alpha* parameter is not supported for all robot types. It is not supported for IRB 140, IRB 1410, IRB 1600ID No Type, IRB 2400, IRB 4400, IRB 6400R, IRB 6400 except for IRB 6400 200/2.5 and IRB 6400 200/2.8, IRB 6600, IRB 6650, IRB 6650S and IRB 7600 except for IRB 7600 325/3.1.

Continues on next page

The parameter is supported for all robots on track when the *7 axes high performance motion* parameter is set.

If the robot does not support *Gravity Alpha*, use *Gravity Beta* along with the recalibration of axis 1 to define the rotation of the robot around the X-axis.

To define the rotation of the robot around the X-axis with help of *Gravity Beta*:

1. Install the robot.
2. Move axis 1 to one of the two positions where the rotational axis for joint 2 is parallel to the floor.
3. Note the axis 1 angle for this position (normally ± 90 degrees). This is needed in Step 6.
4. Make a fine calibration of axis 1 to set this position as the new zero position.
5. Update *Gravity Beta* to the correct tilting angle of the installation. If the robot is tilted forward around axis 2 in the new calibration position, the beta value should be positive. If the robot is tilted backward around axis 2 in the new calibration position, the beta value should be negative.
6. Update the working range of the robot since the zero position for axis 1 is changed. Otherwise, axis 1 may run into its mechanical stops. If the calibration position is positive, reduce the *Upper Joint Bound* angle by the angle as measured during the calibration. If the calibration position is positive, reduce the *Lower Joint Bound* angle by the angle as measured during the calibration.
7. Restart the controller.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

Related information

How to define gravity on page 383

How to define base frame on page 381

Gravity Beta on page 680

Upper Joint Bound on page 407.

Lower Joint Bound on page 408.

6.33.11 Gravity Beta

Parent

Gravity Beta belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gravity_beta

Description

Gravity Beta defines the orientation of the robot with respect to the gravity.

Usage

The *Gravity Beta* is a positive rotation of the robot around the Y-axis in the base coordinate system to define the robot orientation relative to the gravity. The value is set in radians.

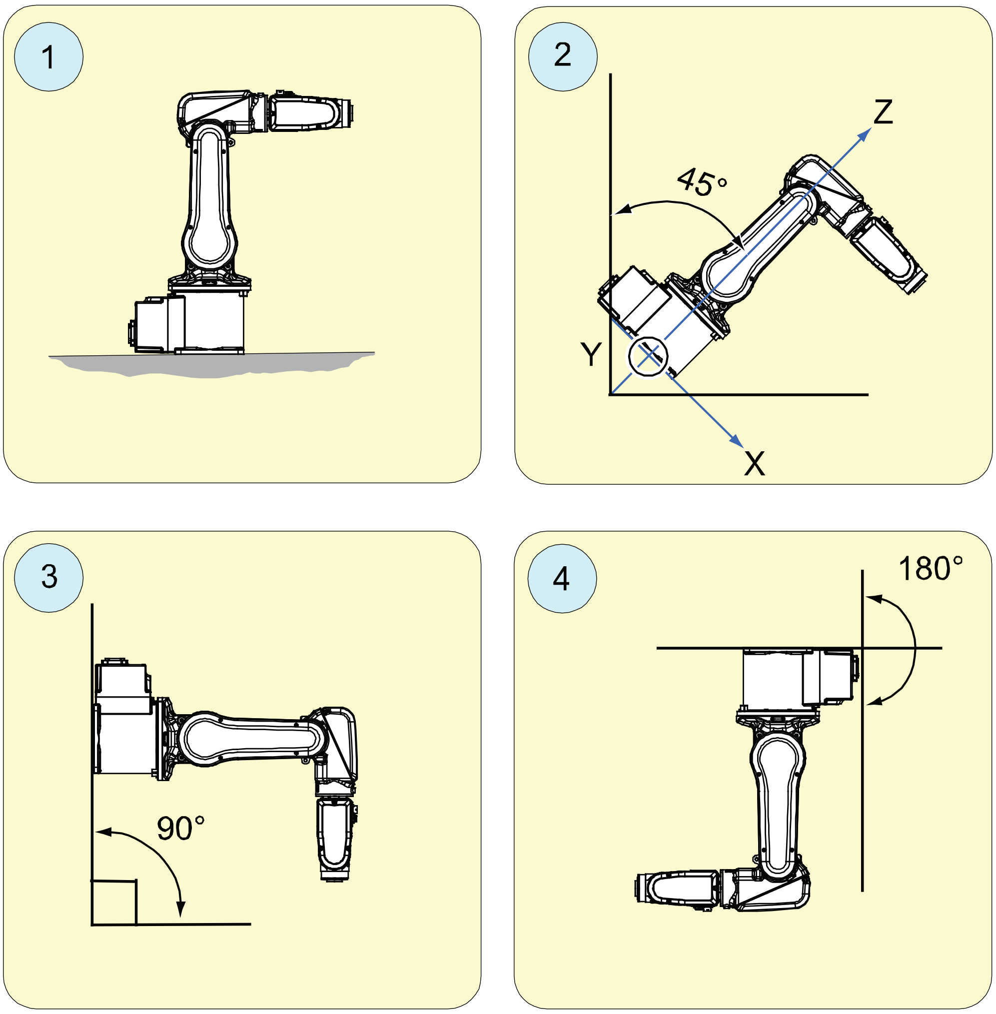
If the robot mounted upside down or on a wall (rotated around the Y-axis) the robot base frame and *Gravity Beta* needs to be changed to reflect the installation. *Gravity Beta* should be π(3.141593) if mounted upside down and ± π/2(1.570796) if mounted on a wall. For more information about base frame refer to *How to define base frame on page 381*.

Gravity Beta is calculated in the following way:

`Gravity Beta = A° x 3.141593/180 = B radians`, where `A` is the mounting angle in degrees and `B` is the mounting angle in radians.

Continues on next page

Examples



xx1000000126

Pos	Mounting angle	Gravity Beta
Pos 1	0° (Floor mounted)	0
Pos 2	45° (Tilted)	0.785398
Pos 3	90° (Wall)	1.570796
Pos 4	180° (Suspended)	3.141593

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

Related information

How to define gravity on page 383

How to define base frame on page 381

Gravity Alpha on page 677

6.33.12 Gamma Rotation

Parent

Gamma Rotation belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gamma_rotation

Description

Gamma Rotation defines the orientation of the robot foot on the travel carriage.

Usage

The *Gamma Rotation* is a rotation of the robot around its Z-axis. It defines the robot rotation relative to the positive direction of the travel carriage (track motion). The value is set in radians.

Prerequisites

The *Gamma Rotation* parameter is useful only for robots on track when the *7 axes high performance motion* parameter is set. This parameter is not used for all robot types.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default values is 0.

6.33.13 Upper Work Area x, y, z

Parent

Upper Work Area x, *Upper Work Area y*, and *Upper Work Area z* belong to the type *Robot*, in the topic *Motion*

Cfg names

upper_work_area_x

upper_work_area_y

upper_work_area_z

Description

Upper work area x defines the x-coordinate of the upper bound of the work area for the robot.

Upper work area y defines the y-coordinate of the upper bound of the work area for the robot.

Upper work area z defines the z-coordinate of the upper bound of the work area for the robot.

Limitations

This parameter is valid only for parallel arm robots.

Allowed values

A numeric value higher than the respective *Lower Work Area* value in meters.

Related information

How to restrict the work area for parallel arm robots on page 385.

Lower Work Area x, y, z on page 684.

How to define base frame on page 381.

6.33.14 Lower Work Area x, y, z

Parent

Lower Work Area x, *Lower Work Area y*, and *Lower Work Area z* belong to the type *Robot*, in the topic *Motion*.

Cfg names

lower_work_area_x

lower_work_area_y

lower_work_area_z

Description

Lower work area x defines the x-coordinate of the lower bound of the work area for the robot.

Lower work area y defines the y-coordinate of the lower bound of the work area for the robot.

Lower work area z defines the z-coordinate of the lower bound of the work area for the robot.

Limitations

This parameter is valid only for parallel arm robots.

Allowed values

A numeric value lower than the respective *Upper Work Area* value in meters.

Related information

How to restrict the work area for parallel arm robots on page 385.

Upper Work Area x, y, z on page 683.

How to define base frame on page 381.

6.33.15 Arm Check Point Speed Limit

Parent

Arm Check Point Speed Limit belongs to the type *Robot*, in the topic *Motion*.

Cfg name

Global_max_speed_limit_acp_custom

Description

Arm Check Point Speed Limit sets the speed limit in meter per second for the arm check point (ACP).

Note

This parameter is used to configure the safety function Cartesian speed supervision.

Note

When changing this safety related system parameter, an event message will take focus on the FlexPendant after restart to notify the user of the change. The user then has to verify that the intended setting was made.

Limitations

Arm Check Point Speed Limit is only used for the following robots:

- IRB 14000

Setting this parameter for any other robot will not have any effect.

Arm Check Point Speed Limit can only be used to lower the speed limit from a maximum speed limit for each robot type. If a higher value is set, the maximum value for the robot type is used.

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000	1.0 m/s

Allowed values

A number between 0.1 and 20.

The default value is 20.

6.33.16 Check Point Bound Limit Outside Cube

Parent

Check Point Bound Limit Outside Cube belongs to the type *Robot*, in the topic *Motion*.

Cfg names

cp_bound_limit_outside

Description

Check Point Bound Limit Outside Cube determines if the robot should be limited to stay outside or inside the cube.

Usage

The check point can be restricted to stay outside or inside a defined cube when the robot is moving. The cube is defined by two coordinates, upper and lower related to the robot base coordinate system. Thus, the defined cube will work as a stationary world zone, where the inside or outside of the cube is the forbidden area for the arm check point. If the parameter is *Yes*, then the check point is limited to being outside the cube. If the parameter is *No*, then the check point is limited to being inside the cube.

Prerequisites

The arm check point bounds must be configured before setting *Check Point Bound Limit Outside Cube*.

Limitations

Check Point Bound Limit Outside Cube can only be used for articulated robots.

Allowed values

Yes or *No*.

Default value

Default value is *No*, limited to stay outside the cube.

Related information

How to define arm check point on page 386.

6.33.17 Upper Check Point Bound x, y, z

Parent

Upper Check Point Bound x, *Upper Check Point Bound y*, and *Upper Check Point Bound z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

upper_arm_cp_bound_x

upper_arm_cp_bound_y

upper_arm_cp_bound_z

Description

Upper Check Point Bound x defines the cartesian x-coordinate upper check point bound on arm check point.

Upper Check Point Bound y defines the cartesian y-coordinate upper check point bound on arm check point.

Upper Check Point Bound z defines the cartesian z-coordinate upper check point bound on arm check point.

Usage

The arm check point can be bound to restrict the movement area.

Allowed values

A numeric value higher than the respective coordinate *Lower Check Point Bound* in meters.

Related information

How to define arm check point on page 386.

Lower Check Point Bound x, y, z on page 688.

6.33.18 Lower Check Point Bound x, y, z

Parent

Lower Check Point Bound x, *Lower Check Point Bound y*, and *Lower Check Point Bound z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

lower_arm_cp_bound_x

lower_arm_cp_bound_y

lower_arm_cp_bound_z

Description

Lower Check Point Bound x defines the cartesian x-coordinate lower check point bound on arm check point.

Lower Check Point Bound y defines the cartesian y-coordinate lower check point bound on arm check point.

Lower Check Point Bound z defines the cartesian z-coordinate lower check point bound on arm check point.

Usage

The arm check point can be bound to restrict the movement area.

Allowed values

A numeric value lower than the respective coordinate *Upper Check Point Bound* in meters.

Related information

How to define arm check point on page 386.

Upper Check Point Bound x, y, z on page 687.

6.33.19 Track Conveyor with Robot

Parent

Track Conveyor with Robot belongs to the type *Robot*, in the topic *Motion*.

Cfg name

track_convey_with_robot

Description

Defines if the robot should track the conveyor.

Usage

Set *Track Conveyor with Robot* to Yes if the robot should track the conveyor without using the track axis, even if robot is coordinated with track. Default value is No.

Limitations

Track Conveyor with Robot can only be used with option *Conveyor tracking* installed.

Allowed values

Yes or No.

Related information

Application manual - Conveyor tracking.

6.33.20 Max External Pos Adjustment

Parent

Max External Pos Adjustment belongs to the type *Robot*, in the topic *Motion*.

Cfg name

max_external_pos_adjustment

Description

Max External Pos Adjustment defines the maximum position adjustment allowed in conveyor direction while tracking a conveyor. The unit is meter.

Usage

If error 50163 occurs, the value of this parameter can be increased for the robots with heavy load and high conveyor speed. Before increasing the parameter value, verify that the parameters *Adjustment speed* and *Adjustment accel* (type *Conveyor systems* in the topic *Process*) are correctly defined.

If the value of this parameter is increased, the value of the parameters *Start ramp* and *Stop ramp* parameters should also be increased to 20 or 30 (type *Conveyor systems* in the topic *Process*).

Allowed values

The minimum value is 0.1 and the maximum value is 0.8.

The default value is 0.2.

6.33.21 7 axes high performance motion

Parent

7 axes high performance motion belongs to the type *Robot*, in the topic *Motion*.

Cfg name

seven_axes_hp_motion

Description

7 axes high performance motion defines the name of the single that moves the robot.

Usage

This parameter should only be set if a "high performance track motion"-additional package is present in your mediapool.

Allowed values

A string with maximum 32 characters, specifying the unit name.

6.33.22 Time to Inposition

Parent

Time to Inposition belongs to the type *Robot*, in the topic *Motion*.

Cfg name

time_to_inpos

Description

Time to Inposition defines the delay time between the last position reference and the inposition event when reaching a fine point.

Limitations

Time to Inposition is only used by the option *Conveyor tracking*.

Allowed values

A value between 0 and 2.0 seconds.

Default value is 0.08 seconds. This should not be changed!

Related information

Application manual - Conveyor tracking.

6.33.23 Orientation Supervision Off

Parent

Orientation Supervision Off belongs to the type *Robot*, in the topic *Motion*.

Cfg name

ori_superv_off

Description

The *Orientation Supervision Off* system parameter defines whether the orientation supervision is Off or On. The parameter is valid only for IRB 340 and IRB 360.

Usage

The orientation supervision is normally On and hence the value of the *Orientation Supervision Off* system parameter is noramlly No. If the orientation supervision is triggered in a system and if the system was working in a previous release of RobotWare, the supervision can be switched off by setting the value of *Orientation Supervision Off* system parameter to Yes.

Note! Switching off the orientation supervision can cause an incorrect behavior in the tool orientation of the robot. The supervision is triggered due to an error in the RAPID program and the first action to be taken is to correct the error rather than switching off the orientation supervision.

Allowed values

Yes or No

6.33.24 Mech.Unit Not Moving Detection Level

Parent

Mech.Unit Not Moving Detection Level belongs to the type *Robot*, in the topic *Motion*.

Cfg name

not_moving_speed_level

Description

Mech.Unit Not Moving Detection Level defines the detection level for the axes of a *Robot* for the system output *Mechanical Unit Not Moving*.

Usage

Normally the output of *Mechanical Unit Not Moving* will be set only when the robot is stopped. The output will also be set if the speed of all axes of the robot are lower than the defined level.

If the detection level is set both for a robot and a single running in the same motion group, all the axes of the robot and the single must move slower than its level to set the output.

Mechanical units with the detection level defined as 0 can run at high speed also when the output is set. For example, if a robot with a track motion has the detection level defined with a value other than 0 only for the track and the robot axis 1, then the other axes of the robot (with detection level = 0) can run at high speed when the output is set.

Allowed values

A value between 0 and 1.

0.01 = 1% of motor max speed, disabled if 0.

The default value is 0.

Related information

Mechanical Unit Not Moving on page 325, in the topic *I/O*, type *System Output*.

Mech.Unit Not Moving Detection Level on page 731, in the topic *Motion*, type *Single*.

6.33.25 LoadIdentify test-speed

Parent

LoadIdentify test-speed belongs to the type *Robot*, in the topic *Motion*.

Cfg name

load_id_test_speed_factor

Description

LoadIdentify test-speed determines the Load Identification speed during the slow test.

Usage

This factor can be used to increase or decrease the axis speed used during the slow-test sequence.

Allowed values

A value between 1 and 6.

The default value is 4, meaning the axis speed will be four times faster than the slowest movement used during the real Load Identification sequence.

6.33.26 Encoder high temp shall generate error

Parent

Encoder high temp shall generate error belongs to the type *Robot*, in the topic *Motion*.

Cfg name

encoder_hi_temp_generate_error

Description

Defines if encoder high temperature shall stop the robot and generate an error in the event log.

Usage

When this parameter is:

Set to Yes, the robot stops and an error is reported in the event log.

Set to No, there is only warning report in the event log.

Note

Changing the parameter to No can result in overheated motors.

Default value

No

Allowed values

Yes

No

6.33.27 Global Speed Limit

Parent

Global Speed Limit belongs to the type *Robot*, in the topic *Motion*.

Cfg name

Global_max_speed_limit_custom

Description

Global Speed Limit sets the speed limit in meters per second for the tool center point (TCP), the arm check point (ACP), and the wrist center point (WCP).

Note

This parameter is used to configure the safety function Cartesian speed supervision.

Note

When changing this safety related system parameter, an event message will take focus on the FlexPendant after restart to notify the user of the change. The user then has to verify that the intended setting was made.

Limitations

Global Speed Limit is only used for the following robots:

- IRB 14000

Setting this parameter for any other robot will not have any effect.

Global Speed Limit can only be used to lower the speed limit from maximum speed limit for each robot type. If a higher value is set, the maximum value for the robot type is used.

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000	1.5 m/s

Allowed values

A number between 0.1 and 20.

The default value is 20.

6.33.28 Arm-Angle Reference Direction

Parent

Arm-Angle Reference Direction belongs to the type *Robot*, in the topic *Motion*.

Cfg name

arm_angle_ref_dir

Description

Arm-Angle Reference Direction controls how the arm-angle property is calculated and affects the location of certain singularities for seven-axis robots.

Usage

In addition to position and orientation, seven-axis robots also depend on the arm-angle concept to fully specify a `robtarget`.

The calculation of the arm-angle depends on a chosen reference direction, and by default this reference direction is chosen as the line passing through axis 2 of the robot and being parallel with the Y-axis of the world frame. When the TCP is on the axis chosen as the reference direction, the arm-angle becomes undefined. Hence, the inverse kinematics is singular for all positions with the TCP on the line, and linear movement on and across this line will not work.

If linear movement in this area of the workspace is important for your application, then you can configure the robot to use another reference direction. The choices available are: the world Y-axis, the world Z-axis, and the line passing through axis 1 of the robot.

Note

A RAPID program created with one value for this parameter will behave differently or maybe not work at all if the parameter value is changed.

Allowed values

Arm-Angle Reference Direction can have the following values:

Value:	Name:	Description:
0	World Y	Reference direction parallel with the Y-axis of the world frame.
1	World Z	Reference direction parallel with the Z-axis of the world frame.
2	Axis 1	Reference direction parallel with a line passing through axis 1 of the robot.

The default value is 0.

Related information

Operating manual - IRB 14000

6.34 Type Robot Serial Number

6.34.1 The Robot Serial Number type

Overview

This section describes the type *Robot Serial Number*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ROBOT_SERIAL_NUMBER

Type description

The type *Robot Serial Number* contains parameters that define the robot's serial number.

Related information

The Robot type on page 668.

6.34.2 Name

Parent

Name belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the robot that the serial number belongs to.

Allowed values

A string with maximum 32 characters.

6.34.3 Robot Serial Number High Part

Parent

Robot Serial Number High Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

robot_serial_number_high_part

Description

Robot Serial Number High Part defines the high part of the robot's serial number.

Usage

The high part is the first four characters of the serial number.

The serial number can be found on the robot's identification plate.

Allowed values

A string with maximum four characters.

Default value is 0000.

6.34.4 Robot Serial Number Low Part

Parent

Robot Serial Number Low Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

robot_serial_number_low_part

Description

Robot Serial Number Low Part defines the low part of the robot's serial number.

Usage

The low integer part of the serial number.

The serial number can be found on the robot's identification plate.

Allowed values

An integer value with maximum nine digits.

Default value is 0.

6.35 Type SG Process

6.35.1 The SG Process type

Overview

This section describes the type *SG Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SG_PROCESS

Type description

The type *SG Process* contains parameters to configure the behavior of a servo gun (or other servo tool). There are parameters for adjusting the timing, force and thickness when closing and opening a servo gun. It is also possible to specify how the tip wear calibration will be performed. The relation between tip force and motor torque is configured as shown below.

Limitations

SG Process can only be used if you have servo tools.

Force-torque relation

Tip Force 1-10 and *Motor Torque 1-10* are used to define the motor torque the motor should apply when a gun closing is ordered with a certain tip force. Due to friction, the relation between force and torque is not always linear.

Between 2 and 10 points can be used to define the motor torque as a function of the tip force. The number of points used is defined in *Number of Stored Forces*.

Ordered closing tip force:	Resulting motor torque:
Tip Force 1	Motor Torque 1
Tip Force 2	Motor Torque 2
Tip Force 3	Motor Torque 3
Tip Force 4	Motor Torque 4
Tip Force 5	Motor Torque 5
Tip Force 6	Motor Torque 6
Tip Force 7	Motor Torque 7
Tip Force 8	Motor Torque 8
Tip Force 9	Motor Torque 9
Tip Force 10	Motor Torque 10

When calculating the force-torque function, the origin (force=0, torque=0) is considered to be an extra point in the diagram. For tip force values between points,

Continues on next page

linear interpolation is used. For tip force values higher than the highest defined tip force, extrapolation from the last two points is used.

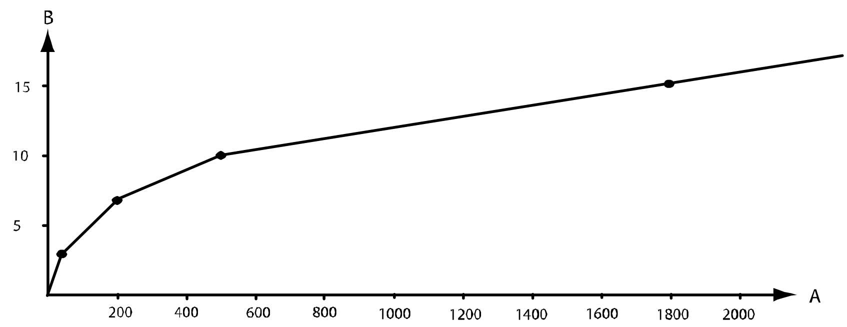
Example

In this example, four points are used to define the relation between tip force and motor torque. Any values given for point 5 to 10 are ignored.

These parameters and values are configured:

Parameter:	Value:
Number of Stored Forces	4
Tip Force 1	50
Tip Force 2	200
Tip Force 3	500
Tip Force 4	1800
Motor Torque 1	3
Motor Torque 2	7
Motor Torque 3	10
Motor Torque 4	15

The results of this configuration is the following graph for motor torque as function of tip force:



xx0400000938

A	Tip force (N)
B	Motor torque (Nm)

6.35.2 Name

Parent

Name belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

name

Description

The name of the *SG Process*.

Usage

Name is used to reference a *SG Process* from the parameter *Use SG Process* in the type *Process*.

Allowed values

A string with maximum 32 characters.

6.35.3 Use Force Master

Parent

Use Force Master belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

use_force_master

Description

Use Force Master determines which *Force Master* should be used.

Usage

Use Force Master is a reference to the parameter *Name* in the type *Force Master*.

Prerequisites

A *Force Master* must be configured before *Use Force Master* can refer to it.

Limitations

Use Force Master can only be used for servo tools.

Allowed values

A string with maximum 32 characters.

Related information

The Force Master type on page 468.

6.35.4 Sync Check Off

Parent

Sync Check Off belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

sync_check_off

Description

Defines if the servo tool synchronization check is turned off.

Usage

Set *Sync Check Off* to Yes to disable the servo tool synchronization check. This can be useful to do to manage the servo tool before having done the service calibration.

Limitations

Sync Check Off can only be used for servo tools.

Allowed values

Yes or No.

Related information

Application manual - Controller software IRC5

Example

To turn off the synchronization check, use this RAPID code:

```
STTune SERVOGUN, 1, SyncCheckOff;
```

To turn on the synchronization check again:

```
STTuneReset SERVOGUN;
```

6.35.5 Close Time Adjust.

Parent

Close Time Adjust. belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

min_close_time_adjust

Description

Adjustment of the ordered minimum close time of the gun.

Usage

If the servo gun is ordered to start closing before the robot is in position, the tips might touch the work piece too early. By setting *Close Time Adjust.* to a positive value, this can be avoided.

If there is a waiting period when the robot is in position but before the servo gun is closing, the cycle time can be reduced by setting *Close Time Adjust.* to a negative value.

Close Time Adjust. may be used to delay the closing slightly when the synchronized pre closing is used for welding.

Limitations

Close Time Adjust. can only be used if you have servo tools.

Allowed values

Numerical value between -100 and 100 (seconds).

6.35.6 Close Position Adjust.

Parent

Close Position Adjust. belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

close_position_adjust

Description

Adjustment of the ordered position when closing the gun to a position and force.

When the tool tips reach the position (plate thickness) ordered by the close instruction, the force control starts. This tool tip position can be adjusted with *Close Position Adjust.* to make the force control start earlier.

Usage

To make sure the tool tips do not touch the work piece before the force control starts, *Close Position Adjust.* can be used to leave some space between the tool tips and the work object.

Limitations

Close Position Adjust. can only be used if you have servo tools.

Allowed values

Numeric value between 0 and 0.005 (meters).

6.35.7 Force Ready Delay

Parent

Force Ready Delay belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

pre_sync_delay_time

Description

Force Ready Delay is used to delay the close ready event. This will make the servo gun wait some extra time when the closing is finished and the ordered force is achieved.

Usage

Force Ready Delay can be used if the servo gun needs some extra time for the force to be stabilized.

Limitations

Force Ready Delay can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 30 (seconds).

6.35.8 Max Force Control Motor Torque

Parent

Max Force Control Motor Torque belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

max_motor_torque

Description

Max allowed motor torque for force control. Commanded force will be reduced, if the required motor torque is higher than this value.

Usage

Max Force Control Motor Torque is used to protect the gun from mechanical overload.

Limitations

Max Force Control Motor Torque can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 100 (Nm).

The default value is 7 Nm.

6.35.9 Post-synchronization Time

Parent

Post-synchronization Time belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

post_sync_time

Description

Post-synchronization Time is used to anticipate the open ready event. The open instruction will be considered ready before the servo gun is completely open.

Usage

Post-synchronization Time can be used to save cycle time. The waiting time between the opening of the servo gun and the execution of the next instruction can be reduced.

The synchronization may fail if *Post-synchronization Time* is set too high.

Limitations

Post-synchronization Time can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 0.5 (seconds).

6.35.10 Calibration Mode

Parent

Calibration Mode belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_mode

Description

Number of tip wear calibration points, i.e. the number of times the servo gun closes during a tip wear calibration.

Usage

If the flexibility of a servo gun is not linearly dependent of the force, more than two measurement points may be necessary. This will improve the plate thickness detection.

Limitations

Calibration Mode can only be used if you have servo tools.

Allowed values

An integer between 2 and 10.

The default value is 2.

6.35.11 Calibration Force High

Parent

Calibration Force High belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_force_high

Description

The force used for the last closing when calibrating the tip wear of a servo gun.

Calibration Force High affects the gun stiffness calibration.

Usage

Set *Calibration Force High* to a value close to the highest force you intend to use the servo gun for. This way it will be well calibrated for forces of that size.

Limitations

Calibration Force High can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 12000 (N).

The default value is 3500 N.

Additional information

The force of the first gun closing in a tip wear calibration is specified in *Calibration Force Low*. If more than two measurement points are used, the force of these measurement points are evenly distributed between *Calibration Force Low* and *Calibration Force High*.

6.35.12 Calibration Force Low

Parent

Calibration Force Low belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_force_low

Description

The force used for:

- the second gun closing of a new tips calibration
- the second gun closing of a tool change calibration
- the first gun closing of a tip wear calibration.

Calibration Force Low affects the gun position calibration.

Usage

It is recommended to set *Calibration Force Low* to a value close to the lowest force you intend to use the servo gun for, but not a higher value than half the value of *Calibration Force High*.

Limitations

Calibration Force Low can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 12000 (N).

The default value is 1500 N.

6.35.13 Calibration Time

Parent

Calibration Time belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_time

Description

The time that the servo gun waits in closed position during calibration.

Usage

If the servo gun needs more time to stabilize, *Calibration Time* can be increased. This can improve the gun position calibration.

In order to make the calibrations faster, *Calibration Time* can be decreased.

Limitations

Calibration Time can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 30 (seconds).

The default value is 0.5 seconds.

6.35.14 Number of Stored Forces

Parent

Number of Stored Forces belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

no_of_active_db_posts

Description

Used to define the relation between tip force and motor torque for a servo gun. *Number of Stored Forces* defines for how many tip force values you want to define the motor torque, i.e. the number of points in the force-torque graph (see *Force-torque relation on page 703*).

Usage

Measure the tip force and motor torque for a number of points. Set *Number of Stored Forces* to the number of points you want to specify.

Limitations

Number of Stored Forces can only be used if you have servo tools.

Allowed values

An integer between 2 and 10.

The default value is 3.

6.35.15 Tip Force 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Parent

Tip Force 1, *Tip Force 2*, *Tip Force 3*, *Tip Force 4*, *Tip Force 5*, *Tip Force 6*, *Tip Force 7*, *Tip Force 8*, *Tip Force 9*, and *Tip Force 10* belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

squeeze_force_1

squeeze_force_2

squeeze_force_3

squeeze_force_4

squeeze_force_5

squeeze_force_6

squeeze_force_7

squeeze_force_8

squeeze_force_9

squeeze_force_10

Description

Used to define the relation between tip force and motor torque for a servo gun (see *Force-torque relation on page 703*).

Tip Force 1 defines the ordered closing force for the first point in the force-torque graph.

Tip Force 2 defines the ordered closing force for the second point in the force-torque graph.

Tip Force 3 defines the ordered closing force for the third point in the force-torque graph.

Tip Force 4 defines the ordered closing force for the fourth point in the force-torque graph.

Tip Force 5 defines the ordered closing force for the fifth point in the force-torque graph.

Tip Force 6 defines the ordered closing force for the sixth point in the force-torque graph.

Tip Force 7 defines the ordered closing force for the seventh point in the force-torque graph.

Tip Force 8 defines the ordered closing force for the eighth point in the force-torque graph.

Tip Force 9 defines the ordered closing force for the ninth point in the force-torque graph.

Tip Force 10 defines the ordered closing force for the tenth point in the force-torque graph.

Continues on next page

Usage

Measure the tip force and the motor torque for some different values.

Set *Tip Force 1* to the tip force value of the first point you want to specify, and *Motor Torque 1* to the corresponding motor torque.

Set *Tip Force 2* to the tip force value of the second point you want to specify, and *Motor Torque 2* to the corresponding motor torque.

Set *Tip Force 3* to the tip force value of the third point you want to specify, and *Motor Torque 3* to the corresponding motor torque.

Set *Tip Force 4* to the tip force value of the fourth point you want to specify, and *Motor Torque 4* to the corresponding motor torque.

Set *Tip Force 5* to the tip force value of the fifth point you want to specify, and *Motor Torque 5* to the corresponding motor torque.

Set *Tip Force 61* to the tip force value of the sixth point you want to specify, and *Motor Torque 6* to the corresponding motor torque.

Set *Tip Force 7* to the tip force value of the seventh point you want to specify, and *Motor Torque 7* to the corresponding motor torque.

Set *Tip Force 8* to the tip force value of the eighth point you want to specify, and *Motor Torque 8* to the corresponding motor torque.

Set *Tip Force 9* to the tip force value of the ninth point you want to specify, and *Motor Torque 9* to the corresponding motor torque.

Set *Tip Force 10* to the tip force value of the tenth point you want to specify, and *Motor Torque 10* to the corresponding motor torque.

Limitations

Tip Force can only be used for servo tools.

Allowed values

A numeric value between 0 and 20000 (N).

6.35.16 Motor Torque 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Parent

Motor Torque 1, *Motor Torque 2*, *Motor Torque 3*, *Motor Torque 4*, *Motor Torque 5*, *Motor Torque 6*, *Motor Torque 7*, *Motor Torque 8*, *Motor Torque 9*, and *Motor Torque 10* belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

squeeze_torque_1

squeeze_torque_2

squeeze_torque_3

squeeze_torque_4

squeeze_torque_5

squeeze_torque_6

squeeze_torque_7

squeeze_torque_8

squeeze_torque_9

squeeze_torque_10

Description

Used to define the relation between tip force and motor torque for a servo gun (see *Force-torque relation on page 703*).

Motor Torque 1 defines the motor torque for the first point in the force-torque graph.

Motor Torque 2 defines the motor torque for the second point in the force-torque graph.

Motor Torque 3 defines the motor torque for the third point in the force-torque graph.

Motor Torque 4 defines the motor torque for the fourth point in the force-torque graph.

Motor Torque 5 defines the motor torque for the fifth point in the force-torque graph.

Motor Torque 6 defines the motor torque for the sixth point in the force-torque graph.

Motor Torque 7 defines the motor torque for the seventh point in the force-torque graph.

Motor Torque 8 defines the motor torque for the eighth point in the force-torque graph.

Motor Torque 9 defines the motor torque for the ninth point in the force-torque graph.

Motor Torque 10 defines the motor torque for the tenth point in the force-torque graph.

Usage

Measure the tip force and the motor torque for some different values

Continues on next page

Set *Motor Torque 1* to the motor torque value of the first point you want to specify, and *Tip Force 1* to the corresponding tip force.

Set *Motor Torque 2* to the motor torque value of the second point you want to specify, and *Tip Force 2* to the corresponding tip force.

Set *Motor Torque 3* to the motor torque value of the third point you want to specify, and *Tip Force 3* to the corresponding tip force.

Set *Motor Torque 4* to the motor torque value of the fourth point you want to specify, and *Tip Force 4* to the corresponding tip force.

Set *Motor Torque 5* to the motor torque value of the fifth point you want to specify, and *Tip Force 5* to the corresponding tip force.

Set *Motor Torque 6* to the motor torque value of the sixth point you want to specify, and *Tip Force 6* to the corresponding tip force.

Set *Motor Torque 7* to the motor torque value of the seventh point you want to specify, and *Tip Force 7* to the corresponding tip force.

Set *Motor Torque 8* to the motor torque value of the eighth point you want to specify, and *Tip Force 8* to the corresponding tip force.

Set *Motor Torque 9* to the motor torque value of the ninth point you want to specify, and *Tip Force 91* to the corresponding tip force.

Set *Motor Torque 10* to the motor torque value of the tenth point you want to specify, and *Tip Force 10* to the corresponding tip force.

Limitations

Motor Torque can only be used for servo tools.

Allowed values

A numeric value between -100 and 100 (Nm).

6.35.17 Squeeze Position 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Parent

Squeeze Position 1 to *Squeeze Position 10* belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

squeeze_pos_1

squeeze_pos_2

squeeze_pos_3

squeeze_pos_4

squeeze_pos_5

squeeze_pos_6

squeeze_pos_7

squeeze_pos_8

squeeze_pos_9

squeeze_pos_10

Description

Used to define the joint position for a servo gun in relation to a given tip force and motor torque (see *Force-torque relation on page 703*).

Squeeze Position defines the joint position for the servo gun in the force-torque graph.

Usage

Squeeze Position is used to control the servo gun when a change of force is ordered during welding.

Limitations

Squeeze Position can only be used for servo tools.

Allowed values

A numeric value typically between -0.02 and 0.02 (meters).

The default value is 0.

6.35.18 Soft Stop Timeout

Parent

Soft Stop Timeout belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

soft_stop_timeout

Description

If a soft stop occurs during constant force, *Soft Stop Timeout* defines how long the force will be maintained. The force will be reduced after this time-out, or when opening is commanded.

Usage

If you want the gun to remain closed a short period after a soft stop, set *Soft Stop Timeout* to the desired time-out value.

Setting *Soft Stop Timeout* to 0 will make the gun release its force immediately when a soft stop occurs.

Limitations

Soft Stop Timeout can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 1.2 (seconds).

The default value is 0.3 seconds.

6.36 Type Single

6.36.1 The Single type

Overview

This section describes the type *Single*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SINGLE

Type description

The type *Single* contains a number of parameters that are common for a single in the robot system. The single is a mechanical unit with one joint. Parameters of this type are used to define which joint the single consist of and the base frame of the single.

6.36.2 Name

Parent

Name belongs to the type *Single*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the single.

Allowed values

A string with maximum 32 characters.

6.36.3 Use Single Type

Parent

Use Single Type belongs to the type *Single*, in the topic *Motion*.

Cfg name

use_single_type

Description

Use Single Type defines what single type is used.

Usage

The single type is defined in the type *Single Type*.

Allowed values

A string with maximum 32 characters.

Related information

The type Single Type on page 733.

6.36.4 Use Joint

Parent

Use Joint belongs to the type *Single*, in the topic *Motion*.

Cfg name

use_joint

Description

Use Joint defines which joint data to use for the single.

Usage

The joints are defined in the type *Joint*.

Allowed values

A string with maximum 32 characters.

Related information

The Joint type on page 516.

6.36.5 Base Frame x, y, z

Parent

Base Frame x, *Base Frame y*, and *Base Frame z* belongs to the type *Single* in the topic *Motion*.

Cfg names

base_frame_pos_x

base_frame_pos_y

base_frame_pos_z

Description

Base Frame x defines the x-direction of the base frame position in relation to the world frame (in meters).

Base Frame y defines the y-direction of the base frame position in relation to the world frame (in meters).

Base Frame z defines the z-direction of the base frame position in relation to the world frame (in meters).

Allowed values

A value between -1,000 and 1,000 meters.

Related information

How to define base frame on page 381.

6.36.6 Base Frame q1, q2, q3, q4

Parent

Base Frame q1, *Base Frame q2*, *Base Frame q3*, and *Base Frame q4* belongs to the type *Single* in the topic *Motion*.

Cfg names

base_frame_orient_u0

base_frame_orient_u1

base_frame_orient_u2

base_frame_orient_u3

Description

Base Frame q1 defines the first quaternion (q1) of the base frame orientation in relation to the world frame.

Base Frame q2 defines the second quaternion (q2) of the base frame orientation in relation to the world frame.

Base Frame q3 defines the third quaternion (q3) of the base frame orientation in relation to the world frame.

Base Frame q4 defines the fourth quaternion (q4) of the base frame orientation in relation to the world frame.

Allowed values

A value between -1 and 1 specifying the orientation.

Related information

How to define base frame on page 381.

6.36.7 Base Frame Coordinated

Parent

Base Frame Coordinated belongs to the type *Single* in the topic *Motion*.

Cfg name

base_frame_coordinated

Description

Base Frame Coordinated defines the name of robot or single that moves the base frame of this single.

Allowed values

A string with maximum 32 characters.

Related information

How to define base frame on page 381.

6.36.8 Mech.Unit Not Moving Detection Level

Parent

Mech.Unit Not Moving Detection Level belongs to the type *Single*, in the topic *Motion*.

Cfg name

not_moving_speed_level

Description

Mech.Unit Not Moving Detection Level defines the detection level for a *Single* for the system output *Mechanical Unit Not Moving*.

Usage

Normally the output of *Mechanical Unit Not Moving* will be set only when the single is stopped. If the detection level is set for the speed of the single, the output will also be set when the speed of the single are lower than the defined level.

If the detection level is set both for a robot and a single running in the same motion group, all the axes of the robot and the single must move slower than its level to set the output.

If the detection level is set only for the single but not for the robot, the output will be set when the speed of the single is lower than the level regardless of the speed of the robot.

Allowed values

A value between 0 and 1.

0.01 = 1% of motor max speed, disabled if 0.

The default value is 0.

Related information

Mechanical Unit Not Moving on page 325, in the topic *I/O*, type *System Output*.

Mech.Unit Not Moving Detection Level on page 694, in the topic *Motion*, type *Robot*.

6.36.9 Ignore Joint World Zones

Parent

Ignore Joint World Zones belongs to the type *Single*, in the topic *Motion*.

Cfg name

ignore_joint_world_zone

Description

If *Ignore Joint World Zones* is set, this axis will be excluded from consideration in all joint WorldZones, overriding any setting in WZHomeJointDef and WZLimJointDef.

Usage

This parameter is useful if the system has an external axis. For example, a servo gun or a track motion, that should be excluded from the checks done by WZHomeJointDef and WZLimJointDef.

Allowed values

Yes or No.

Default value is No.

6.37 Type Single Type

6.37.1 The type Single Type

Overview

This section describes the type *Single Type* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SINGLE_TYPE

Type description

The type *Single Type* contains a number of parameters that are common for a single type in the robot system. The single is a mechanical unit with one joint.

Related information

The Single type on page 724.

6.37.2 Name

Parent

Name belongs to the type *Single Type* in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the single type.

Allowed values

A string with maximum 32 characters.

6.37.3 Mechanics

Parent

Mechanics belongs to the type *Single Type* in the topic *Motion*.

Cfg name

mechanics

Description

Mechanics defines what type of mechanics the single type uses.

Allowed values

The following mechanics are available/allowed:

Value:	Description:
TRACK	Linear track motion
FREE_ROT	Rotating axis
SG_LIN	Servo Gun
EXT_LIN	Conveyor, linear
EXT_ROT	Conveyor, rotating
SS_LIN	Sensor synchronization, linear movement
SS_ROT	Sensor synchronization, rotating movement

Related information

Application manual - Additional axes and stand alone controller.

6.38 Type Stress Duty Cycle

6.38.1 The Stress Duty Cycle type

Overview

This section describes the type *Stress Duty Cycle*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

STRESS_DUTY_CYCLE

Type description

The type *Stress Duty Cycle* is used to protect axes, gearboxes, etc. Damages due to too high mechanical forces are avoided by setting limits for speed and torque.

Limitations

Parameters of the type *Stress Duty Cycle* can only be defined for additional axes.

6.38.2 Name

Parent

Name belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Stress Duty Cycle*.

Usage

Name is used to reference a *Stress Duty Cycle* from the parameter *Use Stress Duty Cycle* in the type *Drive System*.

Allowed values

A string with maximum 32 characters.

6.38.3 Speed Absolute Max

Parent

Speed Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

speed_absolute_max

Description

The absolute highest motor speed to be used.

Usage

Limit the motor speed with *Speed Absolute Max* to avoid too much stress on the axis. If, for example, the gearbox is the limiter for the speed, set *Speed Absolute Max* to a value that will protect the gearbox.

Allowed values

A numeric value between 0 and 1500 (rad/s on motor side).

6.38.4 Torque Absolute Max

Parent

Torque Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

torque_absolute_max

Description

The absolute highest motor torque to be used.

Usage

Limit the motor torque with *Torque Absolute Max* to avoid too much stress on the axis. If, for example, the gearbox is the limiter for the torque, set *Torque Absolute Max* to a value that will protect the gearbox.

Limitation

Torque Absolute Max can only be defined for additional axes.

Allowed values

A numeric value between 0 and 100000 (Nm on motor side).

6.39 Type Supervision

6.39.1 The Supervision type

Overview

This section describes the type *Supervision*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SUPERVISION

Type description

The type *Supervision* is used for supervision of joints. Each joint has one set of parameters of the type *Supervision*.

Limitation

Parameters of the type *Supervision* can only be defined for additional axes.

Related information

The Joint type on page 516.

6.39.2 Name

Parent

Name belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

name

Description

The name of the supervision.

Allowed values

A string with maximum 32 characters.

6.39.3 Brake Release Supervision On

Parent

Brake Release Supervision On belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

brake_release_supervision_on

Description

Brake Release Supervision On defines if the brake release supervision is on or off.

Usage

Set *Brake Release Supervision On* to On to turn on brake release supervision. This activates a position supervision algorithm during brake release.

Allowed values

On or Off

6.39.4 Speed Supervision

Parent

Speed Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

speed_supervision_on

Description

Defines if the speed supervision should be activated or not.

Usage

Speed supervision should normally be On.

NOTE! Deactivating the speed supervision can be dangerous.

Allowed values

On or Off

6.39.5 Position Supervision

Parent

Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

position_supervision_on

Description

Defines if the position supervision should be activated or not.

Usage

The position supervision should normally be On.

NOTE! Deactivating the position supervision can be dangerous.

Allowed values

On or Off

6.39.6 Counter Supervision

Parent

Counter Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

counter_supervision_on

Description

Defines if the measurement system supervision should be activated or not.

Usage

The counter supervision should normally be On.

NOTE! Deactivating the counter supervision can be dangerous.

Allowed values

On or Off

6.39.7 Jam Supervision

Parent

Jam Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

jam_supervision_on

Description

Defines if the jam supervision should be activated or not.

Usage

The jam supervision should normally be activated (On).

NOTE! Deactivating the jam supervision can be dangerous.

Allowed values

On or Off

6.39.8 Load Supervision

Parent

Load Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

load_supervision_on

Description

Defines if the load supervision should be activated or not.

Usage

The load supervision should normally be On.

Allowed values

On or Off

6.39.9 Power Up Position Supervision

Parent

Power Up Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

power_up_position_on

Description

Defines if the power up position supervision should be activated or not.

Usage

The power up position supervision should normally be On.

NOTE! Deactivating the power up position supervision can be dangerous.

Allowed values

On or Off

6.39.10 In Position Range

Parent

In Position Range belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

in_position_range

Description

Defines the allowed position deviation from fine point when the axis is considered to have reached the fine point.

Usage

Normally set to 1.

Allowed values

A value between 0 and 1000000 radians on motor side.

6.39.11 Zero Speed

Parent

Zero Speed belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

normalized_zero_speed

Description

Defines the maximum speed when the axis is considered to be standing still.

Usage

Normally set to 0.02.

Allowed values

A value between 0 and 1, where 1 equals max speed.

6.39.12 Affects Forced Control

Parent

Affects Forced Control belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

joint_affect_forced_Kp

Description

Defines if the joint affects the in position forced control used in fine point.

Usage

Set to No if the joint should affect the in position forced control.

The forced control is used to reduce time for axis to go into the fine point.

Allowed values

Yes or No

Related information

Forced Control Active on page 530, in the type *Lag Control Master 0*.

6.39.13 Forced on Position Limit

Parent

Forced on Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

Kp_forced_on_limit

Description

The upper position limit for activation of forced control, measured from the fine point.

Usage

The upper position limit is measured in radians on the motor shaft.

Allowed values

A value between 0 and 5.

Related information

Affects Forced Control on page 751.

6.39.14 Forced off Position Limit

Parent

Forced off Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

Kp_forced_off_limit

Description

The lower position limit for deactivation of forced control used close to the fine point.

Usage

The lower position limit is measured in radians on the motor shaft.

Limitations

Must have a lower value than *Forced on Position Limit*.

Allowed values

A value between 0 and 5.

Related information

Forced on Position Limit on page 752.

Affects Forced Control on page 751.

6.39.15 Thermal Supervision Sensitivity Ratio

Parent

Thermal Supervision Sensitivity Ratio belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

thermal_supervision_sensitivity_ratio

Usage

Parameter used for tuning the thermal motor model. High value increases the temperature in the model.

Limitations

The thermal supervision is only available for motor units (MU 200, MU 300, and MU 400) and gear units (MTD 250, MTD 500, MTD 750, 200 MID 500, and MID 1000).

Allowed values

A value between 0.5 and 2.

6.40 Type Supervision Type

6.40.1 The type Supervision Type

Overview

This section describes the type *Supervision Type*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SUPERVISION_TYPE

Type description

The type *Supervision Type* is used for continuos supervision of position, speed and torque. These values should follow the planned path, within a tolerance interval, or the movement is stopped.

Limitations

Parameters of the type *Supervision Type* can only be defined for additional axes.

6.40.2 Name

Parent

Name belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Supervision Type*.

Usage

Name is used to reference a *Supervision Type* from the parameter *Use Supervision Type* in the type *Supervision*.

Allowed values

A string with maximum 32 characters.

6.40.3 Max Force Control Position Error

Parent

Max Force Control Position Error belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

fc_position_limit

Description

Max allowed position error during force control.

If the position error is larger than *Max Force Control Position Error*, all movement is stopped.

Usage

When a servo gun is in force control mode it is not allowed to move more than the distance specified in *Max Force Control Position Error*.

The most common reasons for a servo gun to move during force control are:

- the servo gun is flexible and can give in when high forces are applied
- the force control may start before the gun has closed around the plate, e.g. because the ordered plate thickness is larger than the real plate thickness, or because the parameter *Close position adjust* is set to a value larger than 0.

Limitations

Max Force Control Position Error can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 0.10 (meter).

The default value is 0.03 m.

6.40.4 Max Force Control Speed Limit

Parent

Max Force Control Speed Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

fc_speed_limit_factor

Description

Speed error factor during force control.

The speed limits for force control is defined in the type *Force Master Control*. If this speed limit multiplied with *Max Force Control Speed Limit* is exceeded, all movement is stopped.

Usage

The speed may for a short period of time exceed the speed limit (defined in type *Force Master Control*) before it is regulated to a value within the limits. To allow the speed to exceed the limit during this regulation without stopping all movement, *Max Force Control Speed Limit* must be set to a value larger than 1. How much the speed is allowed to over-shoot the limit is determined by *Max Force Control Speed Limit*.

Limitations

Max Force Control Speed Limit can only be used if you have servo tools.

Allowed values

A numeric value between 1 and 10. The value has no unit, but is a ratio of the speed limit defined in the type *Force Master Control*.

The default value is 1.1.

Related information

The Force Master Control type on page 483.

6.40.5 Dynamic Power Up Position Limit

Parent

Dynamic Power Up Position Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

dynamic_power_up_position_limit

Description

Defines the maximum accepted power up position error at maximum speed.

Usage

Dynamic Power Up Position Limit sets a dynamic limit for measurement system supervision of moment during power fail.

A typical value is 120% of the maximum brake distance.

Allowed values

A value between 0 and 1000 in radians.

6.40.6 Teach Max Speed Main

Parent

Teach Max Speed Main belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

teach_mode_speed_max_main

Description

Defines maximum ordered speed in manual mode.

Usage

Teach Max Speed Main is used to limit the maximum speed in manual mode.

The value of *Teach Max Speed Main* should be set so that the arm speed does not exceeds 250 mm/s.

Allowed values

A ratio value between 0 and 1, where 1 equals max speed.

6.40.7 Teach Max Speed DSP

Parent

Teach Max Speed DSP belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

teach_mode_speed_max_dsp

Description

Defines the motor speed supervision level in manual mode.

Usage

Teach Max Speed DSP is used for speed supervision in Axis Computer during manual mode. The value of *Teach Max Speed DSP* should be set to the same value as *Teach Max Speed Main* added with a margin for noise and vibrations. Typical value is the largest value of (*Teach Max Speed Main* * 1.20) or (*Teach Max Speed Main* + 8/*Speed Absolute Max*).

Allowed values

A ratio value between 0 and 1, where 1 equals max speed.

6.40.8 Max Jam Time

Parent

Max Jam Time belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

max_jam_time

Description

Defines the maximum allowed time with maximum torque at zero speed.

Usage

Set *Max Jam Time* to protect the robot and equipment from faults and damage that may occur if the torque is high while the speed is zero.

Allowed values

A value between 0 and 2.0 seconds.

A typical value is 0.5.

6.40.9 Max Overload Time

Parent

Max Overload Time belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

max_overload_time

Description

Defines the maximum allowed time with maximum torque while moving.

Usage

Set *Max Overload Time* to protect the robot and equipment from faults and damage. If *Max Overload Time* is exceeded, the controller will indicate an error in hardware, robot, load, or programming.

Allowed values

A value between 0 and 20 seconds.

A typical value is 0.2.

6.40.10 Auto Max Speed Supervision Limit

Parent

Auto Max Speed Supervision Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

auto_mode_max_speed_sup_limit

Description

Defines the maximum speed supervision limit in automatic mode.

Usage

Auto Max Speed Supervision Limit is typically set to 1.2 to allow margin against speed overshoot, interference from external forces, etc.

Allowed values

A value between 0 and 5, where 1 equals max speed.

A typical value is 1.2.

6.40.11 Influence Group

Parent

Influence Group belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

influence_group

Description

Defines the type of influence group for the *Supervision Type*. An influence group is a group of axes, mechanically affecting each other.

Usage

Influence Group is used to calculate supervision levels.

Normally, for axes not affecting each other, deactivate the function by setting *Influence Group* to 0.

Allowed values

An integer between 0 and 10.

6.40.12 Alarm Position Limit for Brake Release

Parent

Alarm Position Limit for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

brake_release_position_alarm_limit

Description

Alarm Position Limit for Brake Release defines the emergency stop limit for position supervision during brake release.

Usage

An emergency stop is generated if the axis motor moves more than the defined value of *Alarm Position Limit for Brake Release* directly after brake release.

Allowed values

A value between 0 and 1000, defined in radians on motor side.

Default value is 1.0.

6.40.13 Position OK Ratio for Brake Release

Parent

Position OK Ratio for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

brake_release_position_ok_ratio

Description

Position OK Ratio for Brake Release defines the maximum position error for the axis when the axis should leave the brake supervision state and change to normal operation.

Usage

The value of *Position OK Ratio for Brake Release* is a ratio of the value of parameter *Alarm Position Limit for Brake Release*.

Allowed values

A value between 0 and 1.

Default value is 0.2, a normal value is 0.2 - 0.5.

Related information

Alarm Position Limit for Brake Release on page 766.

6.41 Type Transmission

6.41.1 The Transmission type

Overview

This section describes the type *Transmission*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

TRANSMISSION

Type description

Each set of parameters of the type *Transmission* belongs to a joint (robot joint or additional axis).

The parameters in *Transmission* determine the transmission gear ratio between the motor and the axis.

Limitations

The transmission gear ratio can only be defined for additional axes.

The transmission gear ratio for the robot joints are defined by ABB and cannot be changed.

6.41.2 Name

Parent

Name belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Transmission*.

Usage

Name is used to reference a *Transmission* from the parameter *Use Transmission* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6.41.3 Rotating Move

Parent

Rotating Move belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

rotating_move

Description

Rotating Move defines if the axis is rotating or linear.

Usage

For rotating axes, set *Rotating Move* to Yes. For linear axes, set *Rotating Move* to No.

Rotating Move affects if the transmission gear ratio is defined as motor radians per joint radians, or motor radian per joint meter.

Allowed values

Yes or No.

The default value is No (i.e. that the axis is linear).

6.41.4 Transmission Gear Ratio

Parent

Transmission Gear Ratio belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

transm_joint

Description

Transmission Gear Ratio defines the transmission gear ratio between motor and joint.

Usage

For rotating axes, set *Transmission Gear Ratio* to the number of revolutions the motor performs for every revolution of the joint. For linear axes, set *Transmission Gear Ratio* to motor radians per meter.

Limitations

Transmission Gear Ratio can only be defined for external axes. *Transmission Gear Ratio* for the robot joints are defined by ABB and cannot be changed.

Allowed values

A numeric value between -100000 and +100000.

6.41.5 Transmission Gear High

Parent

Transmission Gear High belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

high_gear

Description

When a joint is in independent mode, *Transmission Gear High* is the numerator in the fraction representing the transmission gear ratio between motor and joint. The denominator is the parameter *Transmission Gear Low*.

Usage

When a joint is set to independent mode, the transmission gear ratio is represented as *Transmission Gear High* divided by *Transmission Gear Low*. See *How to define transmission gear ratio for independent joints* for more information on how to use these parameters.

Limitations

The parameter *Transmission Gear High* is only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

Allowed values

An integer value.

Related information

How to define transmission gear ratio for independent joints on page 393.

Transmission Gear Low on page 773.

Application manual - Controller software IRC5.

6.41.6 Transmission Gear Low

Parent

Transmission Gear Low belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

low_gear

Description

When a joint is in independent mode, *Transmission Gear Low* is the denominator in the fraction representing the transmission gear ratio between motor and joint. The numerator is the parameter *Transmission Gear High*.

Usage

When a joint is set to independent mode, the transmission gear ratio is represented as *Transmission Gear High* divided by *Transmission Gear Low*. See *How to define transmission gear ratio for independent joints* for more information on how to use these parameters.

Limitations

The parameter *Transmission Gear Low* is only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

Allowed values

An integer value.

Related information

How to define transmission gear ratio for independent joints on page 393.

Transmission Gear High on page 772.

Application manual - Controller software IRC5.

6.42 Type Uncalibrated Control Master 0

6.42.1 The Uncalibrated Control Master 0 type

Overview

This section describes the type *Uncalibrated Control Master 0*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

UCCM0

Type description

The type *Uncalibrated Control Master 0* is used to regulate uncalibrated axes. If one axis in a mechanical unit is uncalibrated, *Uncalibrated Control Master 0* is used to regulate all axes in that mechanical unit.

6.42.2 Name

Parent

Name belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

UCCM0 name

Description

The name of the *Uncalibrated Control Master 0*.

Usage

Name is used to reference an *Uncalibrated Control Master 0* from the parameter *Uncalibrated Control Master* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6.42.3 Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Kp

Description

Proportional gain in the position regulation loop.

Usage

The higher the value of *Kp, Gain Position Loop*, the better tracking and disturbance rejection.

If the position regulation overshoots, decrease *Kp, Gain Position Loop*.

Limitations

Kp, Gain Position Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 1000 (1/s).

6.42.4 Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Kv

Description

Proportional gain in the speed regulation loop.

Usage

The higher the value of *Kv, Gain Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, decrease *Kv, Gain Speed Loop*.

Limitations

Kv, Gain Speed Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 100 (Nms/rad).

6.42.5 Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Ti

Description

Integration time in the speed regulation loop.

Usage

The lower the value of *Ti Integration Time Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, increase *Ti Integration Time Speed Loop*.

Limitations

Ti Integration Time Speed Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10 (seconds).

The default value is 10 seconds.

6.42.6 Speed Max Uncalibrated

Parent

Speed Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

speed_max_n

Description

Speed Max Uncalibrated defines the maximum allowed speed for an uncalibrated axis.

Usage

Use *Speed Max Uncalibrated* as a limit for the speed of the axis when it is regulated as an uncalibrated axis.

Limitations

Speed Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 670 (rad/s on motor side).

6.42.7 Acceleration Max Uncalibrated

Parent

Acceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

acc_max_n

Description

Acceleration Max Uncalibrated defines the maximum allowed acceleration for an uncalibrated axis.

Usage

Use *Acceleration Max Uncalibrated* as a limit for the acceleration of the axis when it is regulated as an uncalibrated axis.

Limitations

Acceleration Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10000 (rad/s^2 on motor side).

6.42.8 Deceleration Max Uncalibrated

Parent

Deceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

dec_max_n

Description

Deceleration Max Uncalibrated defines the maximum allowed deceleration for an uncalibrated axis.

Usage

Use *Deceleration Max Uncalibrated* as a limit for the deceleration of the axis when it is regulated as an uncalibrated axis.

Limitations

Deceleration Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10000 (rad/s^2 on motor side).

This page is intentionally left blank

Index

Contact us

ABB AB
Discrete Automation and Motion
Robotics
S-721 68 VÄSTERÅS, Sweden
Telephone +46 (0) 21 344 400

ABB AS, Robotics
Discrete Automation and Motion
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 51489000

ABB Engineering (Shanghai) Ltd.
No. 4528 Kangxin Hingway
PuDong District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

www.abb.com/robotics

3HAC050948-001, Rev C, en