ROBOTICS

Technical reference manual

System parameters

Trace back information:
Workspace 24D version a9
Checked in 2024-12-18
Skribenta version 5.6.018

Technical reference manual
System parameters

RobotWare 7.17

Document ID: 3HAC065041-001
Revision: T

Original instructions.

Table of contents

Overview of this manual

About this manual

This manual describes the RobotWare 7 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 be performed only 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 - OmniCore	*3HAC065036-001*
Operating manual - RobotStudio	*3HAC032104-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC065038-001*
Technical reference manual - RAPID Overview	*3HAC065040-001*
Technical reference manual - RAPID kernel	*3HAC065039-001*
Application manual - Additional axes	*3HAC082287-001*

Continues on next page

Reference	Document ID
Application manual - Controller software OmniCore	*3HAC066554-001*
Application manual - Connected Services	*3HAC028879-001*
Application manual - Conveyor tracking	*3HAC066561-001*
Application manual - DeviceNet Master/Slave	*3HAC066562-001*
Application manual - EtherNet/IP Scanner/Adapter	*3HAC066565-001*
Application manual - I/O Engineering	*3HAC082346-001*
Application manual - PROFINET Controller/Device	*3HAC066558-001*

Revisions

Revision	Description
A	Released with RobotWare 7.0.
B	Released with RobotWare 7.0. • Information regarding IRB 14050 added in section *Arm Check Point Speed Limit on page 755*.
C	Released with RobotWare 7.01. • Cfg name removed from entire manual. • Updated the section *Type Connected Services on page 35*.
D	Released with RobotWare 7.1. • The following system parameters are added: - *QueueBackup on page 197*. - *Robot In Trusted Position on page 310*. - *Ind collision stop without brake on page 706* • New type in section *The Syslog type on page 139*. • Minor changes in section *How to define I/O signals on page 348* and *How to define an I/O signal group on page 350*. • Added *AllowMoveRobAuto* in *Function on page 222*. • Added new parameters in the section *Type Connected Services on page 35*. • The system output *CPU Fan Not Running* is removed. • The safety input for emergency stop is updated. • The file format for program modules and system modules is changed to UTF-8, which affects, for example, loading of modules. This includes both automatic loading of modules (topic *Controller*) and loading modules with system inputs (topic *I/O System*). More information is also found in *Technical reference manual - RAPID Overview*.
E	Released with RobotWare 7.2. • Updated the section *Type Connected Services on page 35*. • Section *Write Access on page 271* updated with information about single point of control. • Information related to *Speed override* added in sections *Set Speed Override on page 260*, *Argument 9 on page 280*, *Speed Override on page 315*, *Signal Name on page 283*, and *Function on page 222*. • Arguments for system output signal *Robot In Trusted Position* updated in sections *Robot In Trusted Position on page 310*, *Argument 2 on page 323*, *Argument 5 on page 326*, *Argument 6 on page 327*, *Argument 7 on page 328*, and *Argument 8 on page 329*. • Sections *RMQ Max Message Size on page 341* and *RMQ Max No Of Messages on page 342* updated with information about how to adjust the values of the attributes.

Continues on next page

Revision	Description
	• The following system parameters are added: - *Leak Control for Search Signal on page 544* - *Bandwidth of Speed Error Filter on page 545* - *Threshold for Search Trigger on page 546* - *Search reverse distance on page 547* - *Speed During Search on page 559* - *Prop. Gain in Speed Loop During Search on page 560* - *Integration Time in Speed Loop on page 561* - *Use cfx in robtargets for P-rod robots on page 773* - *Server Type on page 113* - *Max difference for gravity compensation on page 814* • New action value for system parameter *Action*: *Verify Local Presence on page 269*. • Updated sections due to remote mounted disk/virtual root changes: *Name on page 93* (FTP Client), *Name on page 111* (NFS Client), *Name on page 131* (SFTP Client). Removed sections: Local Path (FTP Client), Local Path (NFS Client), Local Path (SFTP Client). • Minor corrections in section *The relation between physical Ethernet ports and system parameters on page 34*.
F	Released with RobotWare 7.3. • Updated the section *Type Connected Services on page 35*. • Link to Collision Avoidance corrected in section *Status on page 284*. • *SoftStop* renamed to *Stop* in entire manual. • *Robot Not On Path* removed from the following sections: *Status on page 284*, *Argument 1 on page 322* • The following system parameters are added: - *Lead through stability margin on page 492* - *Lead through stiffness scale on page 774* • Removed incorrect signals from *System Input* type.
G	Released with RobotWare 7.4. • The following system parameters are added: - *Collision Detection Zero Speed Time on page 698* - *Disable SafeMove Assistance on page 707*, *SafeMove assistance speed factor on page 708*, and *SafeMove assistance zone margin on page 709* • Updated the system parameter *PP Moved on page 307*. • Updated the allowed values for the parameters *Transmission Gear High on page 875*, *Transmission Gear Low on page 876*, *Event on page 179*, and *TCP Distance on page 229*.
H	Released with RobotWare 7.5. • Added the parameter *Joint Speed Max Factor 1, 2, 3, 4, 5, 6, 7 on page 680*. • Updated the system parameter *SimMode on page 312*. • Updated the section *The relation between physical Ethernet ports and system parameters on page 34*.
J	Released with RobotWare 7.6. • The following system parameters are added: - *Calibration High Force Priority on page 798*, *Calibration Full Sequence Freq. on page 799*, and *Calibration No Pos Update on page 800* - *Trust Revolution Counter on page 268* and *Revolution Counter Lost on page 309*.

Continues on next page

Revision	Description
	• Updated the value of the parameter *Server Type on page 95* in type *FTP Client* and topic Communication. • I/O Network added to topic Communication: *Enable on I/O Network on page 91*. • Minor corrections in section *Manipulator supervision on page 695*.
K	Released with RobotWare 7.7. • Added the Type *Type Move in Auto on page 219*. • Added new action value for system parameter *Action*: *Verify Move Robot In Auto on page 270*. • Added the new parameter *Fast Device Startup on page 367*. • Moved **Type System Input** and **Type System Output** from the topic **I/O System** to the topic **Controller**. • Updated the **Prerequisites** section in all the action values under *Type System Input on page 236*. • Updated the parameter *Brake on Time on page 641*. • Added limitation for number of instances of the types *Robot* and *Single*, see *The Robot type on page 740* and Single. • Information about Cross Connections removed from section *Topic I/O System on page 345*. • Reference added to Application Manual I/O Engineering.
L	Released with RobotWare 7.8. • Added the section *Topic Process on page 885* with a short description. • Updated the parameter *Trust Revolution Counter on page 268*. • Updated the **Limitations** section for the parameter *Task in Foreground on page 332*. • Updated the description of `ZeroSpeedEMStop` allowed value for the parameter *Function on page 222*. • Added a NOTE in the **Allowed values** section for the parameter *Deactivation Forbidden on page 629*.
M	Released with RobotWare 7.10. • Minor corrections in *The Firewall Manager type on page 86*. • Updated the **Usage** section for the parameter *Task in Foreground on page 332*. • Added the type *The Port Forward type on page 120*. • Missing parameters added: *Missing tip check distance on page 813* and *Act/Deact Only from Rapid on page 633*. Minor corrections in *Max Gun Force on page 807*, *Force matching deflection values on page 809*, *Ramp time matching deflection values on page 810*, *Deflection in z direction (m) on page 811* and *Deflection in x direction (m) on page 812*. • Added limitations for the system input *Limit Speed*. • Added the parameters *Lead through load compensation deadband on page 493* and *Lead through load compensation on page 775*. • Added *RapidInstructionsPerMs on page 198*. • Added limitation for task type in *Hidden on page 339*. • Updated the section *The Syslog type on page 139*. • Updated the section *The Port Forward type on page 120*. • Updated the section *The Firewall Manager type on page 86*. • Minor corrections in section *The relation between physical Ethernet ports and system parameters on page 34*. • Updated the section *Device Mapping on page 401*.

Continues on next page

<table>
<tr><th>Revision</th><th>Description</th></tr>
<tr><td>N</td><td>Released with RobotWare 7.12.
Added the new system output signal Control On State on page 291.
Updated the Additional information section in CollisionErrorHandling on page 192.
Updated the Additional information section in Backup on page 239.
Updated the Description section for the parameters Enable 3G connection on page 51, Enable Wi-Fi connection on page 63, and State on page 68.
The type Safety Run Chain is removed because the possibility to configure the stop category using the system parameters has been removed. This affects users with safety configurations created prior to RobotWare 7.3. Older configurations can still be loaded but the stop category 0 will be used for all protective and emergency stops. Upgrade the safety configuration to a newer version and configure the stop category with Visual SafeMove function in RobotStudio.
Parameters removed from Type Connected Services on page 35.</td></tr>
<tr><td>P</td><td>Released with RobotWare 7.13.
Added the new parameter Torque quota on page 611.
Added the new parameter Enable orientation correction on page 776.
Network segments updated in The relation between physical Ethernet ports and system parameters on page 34.
Information about firewall settings for network services updated in Type Firewall Manager on page 86 and Connection Type on page 37.
Updated the Allowed Values section for the parameter Connected Services Mode on page 49.</td></tr>
<tr><td>Q</td><td>Released with RobotWare 7.14.
Added the Type OPC UA Server on page 118.
Added the parameter Unit has no brake on page 622.</td></tr>
<tr><td>R</td><td>Released with RobotWare 7.15.
Added the system inputs and outputs Enable Energy Saving and Energy Saving Blocked in Topic Controller on page 151.
Added the action value MathLibMemSize on page 193.
Added the new system output signal Stop from client on page 316.
Added the parameters Robot weight in power sharing on page 777 and Single weight in power sharing on page 826.
Added the parameter Joint Id on page 494.
New sections: Enable Energy Saving on page 243 and Energy Saving Blocked.</td></tr>
<tr><td>S</td><td>Released with RobotWare 7.16.
Added the parameter Exclude From Load Modules In AllTasks on page 344.
Added the action value ReportNonActiveSingles on page 200.</td></tr>
<tr><td>T</td><td>Released with RobotWare 7.17.
Added the parameters Ultra Accuracy stability margin on page 495 and
Ultra Accuracy on page 779.
Minor changes.
The names of a number of parameters are updated to match the names visible in RobotStudio and on the FlexPendant.</td></tr>
</table>

Product documentation

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.

Tip

All documents can be found via myABB Business Portal, *www.abb.com/myABB*.

Product manuals

Manipulators, controllers, DressPack, 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.
- Troubleshooting.
- Decommissioning.
- Reference information (safety standards, unit conversions, screw joints, lists of tools).
- Spare parts list with corresponding figures (or references to separate spare parts lists).
- References to circuit diagrams.

Technical reference manuals

The technical reference manuals describe reference information for robotics products, for example lubrication, the RAPID language, and system parameters.

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, software).
- How to install included or required hardware.
- How to use the application.

Continues on next page

- 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 troubleshooters.

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 types.

There are several topics in the controller, each describing an area of the robot system. All parameters are stored in a data base. A separate configuration file is saved for each topic, it can also be generated while creating a backup. These files are known as cfg files (file extension .cfg). See *Configuration files on page 31*.

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 purposes only 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 this manual.

Continues on next page

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

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

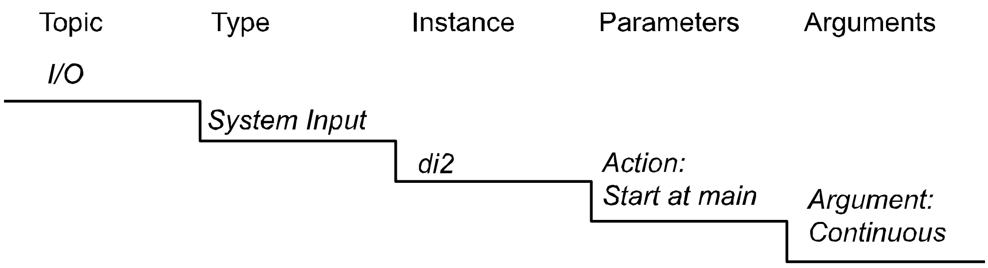
Working with system parameters

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

A parameter can have a defined default value. A parameter with a default value will not be saved in the configuration file, but will be visible in the editors in RobotStudio and FlexPendant.

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, saved as configuration files (*.cfg).

The configuration files are included in system backups. A configuration parameter that is defined with a default value will not be listed in the configuration file.

Note

Configuration files and backups shall not be loaded into systems running an older RobotWare version than the one they were created in.

Configuration files and backups are not guaranteed to be compatible between major releases of RobotWare and may need to be migrated after a RobotWare upgrade.

Topic:	Configuration area:	Configuration file:
Communication	Communication protocols and devices	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 and equipment	PROC.cfg

For RobotWare 7, there is also a topic used by ABB for troubleshooting and debugging purposes, called *Debug* (DBG.cfg). This topic is not described in this manual and is not included in backups.

Note

Only parameters which are visible from RobotStudio are described in this manual.

1.3 File system

Overview

This section describes how paths on the controller can be defined using environment 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*

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 239

Load on page 249

Load and Start on page 251

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 Ethernet ports.

2.2 The relation between physical Ethernet ports and system parameters

Network segment overview

The Ethernet networks used by OmniCore are distributed into the following segments:

Network segment	C30	C90XT V250XT Type A	V250XT Type B V400XT	E10	Usage
Private Network	I/O (Scalable I/O) ETHERNET SWITCH	I/O (Scalable I/O) ETHERNET SWITCH	DEV	DEVICE	Process equipment local to this specific robot.
	MGMT (Management)	MGMT (Management)	MGMT (Management)	MGMT (Management)	ABB service personnel.
	HMI (FlexPendant)	HMI (FlexPendant)	HMI (FlexPendant)	HMI (FlexPendant)	FlexPendant connection.
ABB Connect Network	ABB Connect	ABB Connect	ABB Connect	WAN 2	ABB Connected Services connection.
Public Network	WAN	WAN	WAN 1	WAN 1	Public/factory network.
			WAN 2		
I/O Network	LAN	LAN3	LAN	-	Secondary public/factory network. Isolated from WAN.

Note

For information regarding location of the Ethernet port connectors, see the Product manual for the respective OmniCore controller.

IP addresses

See *Operating manual - Integrator's guide OmniCore*, section *Configuring networks*.

Configuration

See *Operating manual - Integrator's guide OmniCore*, section *Configuring networks*.

2.3 Type Connected Services

2.3.1 The type Connected Services

Overview

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

Type description

RobotWare software comes with built-in functionality that can be activated on each robot system to connect it to ABB Connected Services Cloud. Prerequisites to connect a robot with *Connected Services*:

- A valid service agreement, including each robot that can be connected.
- Network connectivity between the robot and Internet. Note that the connection to ABB Connected Services Cloud is always initiated by RobotWare.

This type contains parameters used by the RobotWare to enable connectivity to ABB Connected Services Cloud.

Related information

Application manual - Controller software OmniCore

2.3.2 Enabled

Parent

Enabled belongs to the type *Connected Services*, in the topic *Communication*.

Description

The parameter *Enabled* specifies if robot will connect to ABB Connected Services Cloud or not.

When enabling the connection for the first time, ensure that all other relevant connectivity parameters are specified. For more information about setup and connectivity of *Connected Services*, see *Application manual - Controller software OmniCore*.

Allowed values

Value	Description
False (No)	Connection shall not be started.
True (Yes)	Connection shall be started.

The default value is True (Yes).

2.3.3 Connection Type

Parent

Connection Type belongs to the type *Connected Services*, in the topic *Communication*.

Description

Connection Type defines what type of network connection will be used by RobotWare to connect to *ABB Connected Services Cloud*. RobotWare insure communication based on the selected *Connection Type*.

Allowed values

Value	Description
ABB Connect	The communication will be done through the ABB Connect network port connected to the Connected Services Gateway Module.
Public	The communication will be done through the Public network port (WAN) according to Gateway and DNS available. **Note** If the *Connection Type* is set to *Public*, field **Enable on Public** in the firewall settings must be set to YES for *Connected Services*. See *The Firewall Manager type on page 86*.
Custom	The communication will be done through a specified gateway and DNS.

The default value is *ABB Connect*.

2.3.4 Internet Gateway IP

Parent

Internet Gateway IP belongs to the type *Connected Services*, in the topic *Communication*. Only if the Connection Type is *Custom.*

Description

Use *Internet Gateway IP* parameter to specify the custom gateway.

Usage

Used to route connected services communication.

Related information

Connection Type on page 37

2.3.5 Internet DNS IP

Parent

Internet DNS IP belongs to the type *Connected Services*, in the topic *Communication*. Only if the Connection Type is *Custom.*

Description

Use the *Internet DNS IP* parameter to specify the custom DNS.

Usage

Used for the domain name resolution.

Related information

Connection Type on page 37

2.3.6 Proxy Used

Parent

Proxy Used belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Proxy Used* parameter specifies if a HTTP proxy should be used or not.

Allowed values

Value	Description
Defined (Yes)	Use HTTP proxy.
Not Defined (No)	Do not use HTTP proxy.

The default value is Not Defined (No).

2.3.7 Proxy Name

Parent

Proxy Name belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Proxy Name* parameter specifies the name of the HTTP proxy server.

Prerequisites

Used only if the value of the *Proxy Used* parameter is *Yes*.

For more details, see *Proxy Used on page 40*.

Allowed values

A string with maximum of 64 characters.

The default value is empty.

2.3.8 Proxy Port

Parent

Proxy Port belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Proxy Port* parameter specifies which port is used by the HTTP proxy server.

Prerequisites

Use only if the value of the *Proxy Used* parameter is *Yes*.

See *Proxy Used on page 40*.

Allowed values

An integer between 1 and 65535.

The default value is 0.

2.3.9 Proxy Auth

Parent

Proxy Auth belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Proxy Auth* parameter defines the proxy authentication type used for connecting with the proxy server.

Allowed values

Value	Description
Basic	Basic authentication method used to connect to the proxy server.
None	No authentication method used.

The default value is *None*.

2.3.10 Proxy User

Parent

Proxy User belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Proxy User* parameter defines the user name that authenticates with the proxy server.

Allowed values

A string with maximum of 64 characters.

The default value is empty.

2.3.11 Proxy Password

Parent

Proxy Password belongs to the type *Connected Services*, in the topic *Communication*.

Description

The Proxy Password parameter defines the password used for the authentication with the proxy server.

Note

The password will be stored encrypted in configuration.

Allowed values

A string with maximum of 64 characters.

The default value is empty.

2.3.12 Server Polling

Parent

Server Polling belongs to the type *Connected Services*, in the topic *Communication*.

Description

The Server Polling parameter defines the frequency of server interactions.

Allowed values

Value	Description
Slow	Will consume less data (10 minutes polling rate).
Fast	Will consume more data (1 minute polling rate).

The default value is *Slow*.

2.3.13 Debug Mode

Parent

Debug Mode belongs to the type *Connected Services*, in the topic *Communication*.

Description

The Debug Mode parameter enables the extensive logging for debugging the issues.

Allowed values

Value	Description
Disabled	Disables the generation of debug logs.
Enabled	Enables the generation of debug logs.

The default value is *Disabled*.

2.3.14 Trace Level

Parent

Trace Level belongs to the type *Connected Services*, in the topic *Communication*.

Description

The Trace Level parameter controls the level of logging if the parameter Debug Mode is enabled.

Prerequisites

Used only if the value of the Debug Mode parameter is *Enabled*. For more details, see *Debug Mode on page 47*.

Allowed values

Value	Description
Normal	Log file contains normal level of debug information.
Verbose	Log file contains detailed level of debug information

The default value is *Normal*.

2.3.15 Connected Services Mode

Parent

Connected Services Mode belongs to the type *Connected Services*, in the topic *Communication*.

Description

The *Connected Services Mode* parameter defines the compatibility for different robot controller's data format, cloud solution, and specific features.

Allowed values

OmniCore Connected Services

2.4 Type Connected Services Gateway 3G

2.4.1 The Connected Services Gateway 3G type

Overview

This section describes the type *Connected Services Gateway 3G*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Description

The Connected Services Gateway 3G type allows to edit and configure the Connected Services Gateway 3G parameters based on the installed SIM card and the Connected Services Gateway 3G module.

The Connected Services Gateway 3G configuration is used when a Connected Services Gateway 3G module is installed in the controller.

2.4.2 Enable 3G connection

Parent

Enable 3G connection belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Enables or disables the communication for the Connected Services Gateway 3G module.

Allowed values

Value	Description
True (Yes)	Enables the Connected Services Gateway 3G module.
False (No)	Disables the Connected Services Gateway 3G module.

The default value is *True (Yes)*.

2.4.3 Roaming

Parent

Roaming belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Enables or disables the roaming. By default, roaming is required for ABB SIM card.

Usage

Allowed values

Value	Description
True (Yes)	Enables the SIM card roaming
False (No)	Disables the SIM card roaming

The default value is *True (Yes)*.

2.4.4 Access Point Name

Parent

Access Point Name belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Access point name used by the SIM card to connect to the network.

Allowed values

The default value is *abbrobotics.com* for ABB SIM card.

2.4.5 User

Parent

User belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

It is the user name of the APN which authenticates with the server.

Allowed values

The default value is empty, which means no user authentication.

2.4.6 Password

Parent

Password belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Password used for login with the user, if required.

Allowed values

The default value is empty, which indicates no password is required.

Additional information

The password will be stored encrypted.

2.4.7 Pin

Parent

Pin belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

If the SIM is secured with PIN, type the PIN number in this field. By default ABB SIM card has no PIN.

Allowed values

The numerical pin number is empty if there is no pin.

Additional information

The pin will be stored encrypted.

2.4.8 Operator

Parent

Operator belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Type the operator ID for force connection to a specific mobile operator.

Allowed values

The default is empty which defines automatic detection.

Additional information

An operator ID is defined by the combination of MCC (Mobile Country Code) and MNC (Mobile Network Code).

Example: 20801 for Orange France

MCC: 208 for France

MNC: 01 for Orange

2.4.9 Band

Parent

Band belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Select the specific network band.

Allowed values

Value	Description
""(Automatic)	Empty string means automatic band detection.
gsm	Means 2G bands forced
umts	Means 3G bands forced

The default value is *Automatic (empty string)*.

2.4.10 Authentication

Parent

Authentication belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Select the authentication method.

Allowed values

Value	Description
""(Automatic)	Empty string means automatic detection of authentication type.
chap	Challenge Handshake Authentication Protocol
pap	Password Authentication Protocol

The default value is *Automatic (empty string)*.

2.4.11 Idle

Parent

Idle belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Description

Type the Idle time (in seconds) to specify the idle time required before hanging up the connection.

Allowed values

The default value is *0*.

Additional information

Not implemented

2.4.12 Delay

Parent

Delay belongs to the type *Connected Services Gateway 3G*, in the topic *Communication*.

Usage

Type the duration (in seconds) between a connection time out and retry.

Allowed values

The default value is *0, which means auto reconnect immediately*.

Additional information

Not implemented

2.5 Type Connected Services Gateway Wi-Fi

2.5.1 The Connected Services Gateway Wi-Fi type

Overview

This section describes the type *Connected Services Gateway Wi-Fi*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Description

The Connected Services Gateway Wi-Fi type allows to edit and configure the Connected Services Gateway Wi-Fi parameters for the Connected Services Gateway Wi-Fi module.

The Connected Services Gateway Wi-Fi configuration is used when a Connected Services Gateway Wi-Fi module is installed in the controller.

2.5.2 Enable Wi-Fi connection

Parent

Enable Wi-Fi connection belongs to the type *Connected Services Gateway Wi-Fi*, in the topic *Communication*.

Description

Enables or disables the communication for the Connected Services Gateway Wi-Fi module.

Allowed values

Value	Description
True (Yes)	Enables the *Connected Services Gateway Wi-Fi* module.
False (No)	Disables the *Connected Services Gateway Wi-Fi* module.

The default value is *True (Yes)*.

2.5.3 SSID

Parent

SSID belongs to the type *Connected Services Gateway Wi-Fi*, in the topic *Communication*.

Description

Type the SSID of the wireless network to which the module need to be connected.

2.5.4 Key

Parent

Key belongs to the type *Connected Services Gateway Wi-Fi*, in the topic *Communication*.

Description

Type the security key of the SSID.

Additional information

The key will be stored encrypted.

2.5.5 Security

Parent

Security belongs to the type *Connected Services Gateway Wi-Fi*, in the topic *Communication*.

Description

Select the type of security required.

Allowed values

Value	Description
"" (Automatic)	Automatic detection of security type.
ieee 802x	IEEE 8021x certificate. Not implemented
none	No security on access point. Not recommended
wep	WEP security on access point. Not recommended
psh	WPA2-PSH security on access point. Recommended
wps	WPS security. Not implemented

The default value is *empty string (Automatic).* .

2.6 Type Connected Services Gateway Wired

2.6.1 The Connected Services Gateway Wired type

Overview

This section describes the type *Connected Services Gateway Wired*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Type description

The Connected Services Gateway Wired type allows to edit and configure Connected Services Gateway Wired parameters for the Connected Services Gateway Wired module.

The Connected Services Gateway Wired configuration is used when a Connected Services Gateway Wired module is installed in the controller.

2.6.2 State

Parent

State belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Enables or disables the communication for the Connected Services Gateway Wired module.

Allowed values

Value	Description
True (Yes)	Enables the *Connected Services Gateway Wired* module.
False (No)	Disables the *Connected Services Gateway Wired* module.

The default value is *True (Yes)*.

2.6.3 IP Address

Parent

IP Address belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Type the IP address of the ABB Connect port on the wired network.

> **Note**
>
> Before assigning the IP address for the wired module make sure the IP address has been assigned to your module by the network administrator.

Allowed values

> **Note**
>
> For the Connected Services 4G gateway, the value must be set to 192.168.126.2

2.6.4 Subnet Mask

Parent

Subnet Mask belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Type the subnet mask of the ABB Connect port.

Allowed values

Note

For the Connected Services 4G gateway, the value must be set to 255.255.255.0

2.6.5 Gateway

Parent

Gateway belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Type the IP address of the external gateway.

Allowed values

> **Note**
>
> For the Connected Services 4G gateway, the value must be set to 192.168.126.1

2.6.6 Primary DNS Server

Parent

Primary DNS Server belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Type the IP address of the external primary DNS server.

Allowed values

> **Note**
>
> For the Connected Services 4G gateway, the value must be set to 192.168.126.1

2.6.7 Secondary DNS Server

Parent

Secondary DNS Server belongs to the type *Connected Services Gateway Wired*, in the topic *Communication*.

Description

Type the IP address of the secondary DNS server, if available.

2.7 Type DNS Client

2.7.1 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.

Type description

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

2.7.2 Name

Parent

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

Description

Must exist and be set to *DNS Client*.

Default value

The default value is *DNSC*.

Allowed values

DNSC

2.7.3 Enabled

Parent

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

Description

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

Default value

The default value is *No*.

Allowed values

Yes or *No*

2.7.4 Domain Name

Parent

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

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.

2.7.5 1st Name Server

Parent

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

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

2.7.6 2nd Name Server

Parent

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

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

2.7.7 3rd Name Server

Parent

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

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

2.7.8 4th Name Server

Parent

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

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

2.7.9 Server Port

Parent

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

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

2.7.10 Retries

Parent

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

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

2.7.11 Timeout

Parent

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

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

2.7.12 IPv4 Zone Name

Parent

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

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.

2.8 Type Firewall Manager

2.8.1 The Firewall Manager type

Overview

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

Type description

The type *Firewall Manager* is used to configure the network firewall on the controller.

Note

Only pre-registered network services can be configured. It is not possible to add new network services.

Default configuration

The following table contains default values for pre-registered network services. These services will always be shown. The list might contain more services depending on which options are installed in the system.

Network Service (pre-registered)	Enable on Public Network	Enable on Private Network	Enable on I/O Network
Bonjour	No	No	No
ConnectedServices	No	Yes	No
DHCP_Client	Yes	No	No
EtherNetIP	No	Yes	No
Netscan	No	Yes [i]	No
RapidSockets	No	No	No
RobAPI	No	Yes [i]	No
RobICI	No	Yes	Yes
RobotWebServices	No	Yes [i]	No
syslog	No	Yes	No
OpcUaServer	No	Yes	No
UDPUC	No	Yes	No

i This value cannot be changed. It must be enabled on the private network for connection of RobotStudio and FlexPendant to the controller.

Note

All default services and application protocols, except DHCP client, are disabled by default on the Public Network. All communication via the Public Network must be manually enabled.

Continues on next page

Tip

In order to be able to connect to the Public network and perform basic RAPID tasks, the following must be enabled in the firewall:

- Bonjour
- Netscan
- RobAPI
- RobotWebServices

Examples

The following list shows a few examples of services that must be enabled for certain functions:

- If RobotStudio should display all available controllers on the network, the service Netscan must be set to Yes on the Public Network.
- For a complete experience of the RobotStudio functionality, Netscan, RobAPI and RobotWebServices must be set to Yes on the Public Network.
- To use Connected Services on Public port Connected Services must be set to Yes on the Public network. If not, the Connected Services status will be blocked.

Related information

Application manual - Controller software OmniCore.

Operating manual - Integrator's guide OmniCore.

2.8.2 Network Service

Parent

Network Service belongs to the type *Firewall Manager*, in the topic *Communication*.

Description

The parameter *Network Service* defines the name of a network service which can be enabled or disabled on the private network, public network or I/O network.

Allowed values

A string with maximum 32 characters.

Examples

The following list shows a few examples of network services names:

```
Bonjour
Netscan
RobotWebServices
```

2.8.3 Enable on Public Network

Parent

Enable on Public Network belongs to the type *Firewall Manager*, in the topic *Communication*.

Description

The parameter *Enable on Public Network* defines if the network service is enabled or disabled on the public network.

Usage

Set the value to *Yes* to enable the communication on the public network.

Allowed values

Yes or No.

Default value is No.

2.8.4 Enable on Private Network

Parent

Enable on Private Network belongs to the type *Firewall Manager*, in the topic *Communication*.

Description

The parameter *Enable on Private Network* defines if the network service is enabled or disabled on the private network.

Usage

Set the value to *Yes* to enable the communication on the private network.

Allowed values

Yes or No.

2.8.5 Enable on I/O Network

Parent

Enable on I/O Network belongs to the type *Firewall Manager*, in the topic *Communication*.

Description

The parameter *Enable on I/O Network* defines if the network service is enabled or disabled on the I/O network.

Usage

Set the value to *Yes* to enable the communication on the I/O network.

Allowed values

Yes or No.

2.9 Type FTP Client

2.9.1 The FTP Client type

Overview

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

Type description

The type *FTP Client* is used to configure FTP.

Appropriate RobotWare options need to be installed. For more details, see *Application manual - Controller software OmniCore*.

Related information

Application manual - Controller software OmniCore.

2.9.2 Name

Parent

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

Description

The name used when accessing the FTP mounted disk.

Usage

When the connection is used from a RAPID program or the FlexPendant, it is referenced with the parameter *Name*.

Allowed values

A string with maximum 40 characters.

Example

The remote mounted disk is used for a connection with unit `C:` on a remote PC. If *Name* is set to `ftp1`, the file `C:\ftp.modx` can be accessed from a RAPID program as `ftp1/ftp.modx`.

2.9.3 Server Address

Parent

Server Address belongs to the type *FTP Client*, in the topic *Communication*.

Description

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

Usage

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

Allowed values

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

2.9.4 Server Type

Parent

Server Type belongs to the type *FTP Client*, in the topic *Communication*.

Description

The type of server the client is connected to.

Allowed values

FileZilla version 0.x, *FileZilla version 1.x*, *MS IIS*, *Serv-U*, *Linux Ubuntu*.

Related information

Application manual - Controller software OmniCore

2.9.5 Trusted

Parent

Trusted belongs to the type FTP Client, in the topic Communication.

Description

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

Usage

A 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 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 OmniCore

2.9.6 Server Path

Parent

Server Path belongs to the type *FTP Client*, in the topic *Communication*.

Description

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

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Controller software OmniCore

2.9.7 Username

Parent

Username belongs to the type *FTP Client*, in the topic *Communication*.

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*. For more information, see *Password on page 99*.

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Controller software OmniCore

2.9.8 Password

Parent

Password belongs to the type *FTP Client*, in the topic *Communication*.

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*. For more information, see *Username on page 98*.

The password will be, if not already, encrypted. The encrypted string will map to the original password only on the original controller.

Allowed values

A string with a maximum of 40 characters. Encrypted strings can be longer.

Related information

Application manual - Controller software OmniCore

2.9.9 Show Device

Parent

Show Device belongs to the type *FTP Client*, in the topic *Communication*.

Description

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

Allowed values

Yes or No.

2.10 Type IP Setting

2.10.1 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.

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:

- 192.168.127.0 (only for internal use by the controller)
- FlexPendant Network
- Private Network
- Public Network (set up using RobotStudio or FlexPendant)

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

2.11 Type IV Camera

2.11.1 The IV Camera type

Overview

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

Type description

The type *IV Camera* is used to configure the camera for *Integrated Vision*.

Related information

Application manual - Integrated Vision.

2.11.2 Name

Parent

Name belongs to the type *IV Camera*, in the topic *Communication*.

Description

The name of the camera in the controller. Each camera must have a unique name.

Allowed values

A string with maximum 40 characters.

2.11.3 Username

Parent

Username belongs to the type *IV Camera*, in the topic *Communication*.

Description

The username used by the controller when it logs on to the camera.

Allowed values

A string with maximum 40 characters.

2.11.4 Password

Parent

Password belongs to the type *IV Camera*, in the topic *Communication*.

Description

The password used by the controller when it logs on to the camera. The password is encrypted.

Allowed values

A string with a maximum of 40 characters. Encrypted strings can be longer.

2.11.5 MAC Address

Parent

MAC Address belongs to the type *IV Camera*, in the topic *Communication*.

Description

The mac address assigned to the NIC of the camera.

This parameter is read-only.

2.11.6 Communication Timeout

Parent

Communication Timeout belongs to the type *IV Camera*, in the topic *Communication*.

Description

The parameter *Communication Timeout* defines the time that the controller will wait for a response from the camera. If a request to the camera results in a communication timeout, the reason can be that the camera is disconnected or that it needs more time to process the result.

Allowed values

A value between 1 and 120000 milliseconds.

The default value is 5000 milliseconds.

2.11.7 Output to Rapid

Parent

Output to Rapid belongs to the type *IV Camera*, in the topic *Communication*.

Description

The parameter *Output to Rapid* defines if the controller will manage the result of a photo request. The camera produces results for each photo request. If *Output to Rapid* is set to *Yes*, the result is converted to RAPID variables, which means that the photo request is managed by the controller and the instruction `CamGetResult` can be used to get the result.

Allowed values

Yes or No.

2.11.8 Max Time Image Request

Parent

Max Time Image Request belongs to the type *IV Camera*, in the topic *Communication*.

Description

The parameter *Max Time Image Result* defines the time that the controller will wait for the result from a photo request. The time for the photo request depends on the complexity of the vision job. If a request to the camera results in a communication timeout, the reason can be that the camera is disconnected or that it needs more time to process the result.

Allowed values

The default value is 120 seconds.

2.12 Type NFS Client

2.12.1 The NFS Client type

Overview

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

Type description

The type *NFS Client* is used to configure NFS.

Appropriate RobotWare options need to be installed. For more details, see *Application manual - Controller software OmniCore*.

Related information

Application manual - Controller software OmniCore.

2.12.2 Name

Parent

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

Description

The name used when accessing the NFS mounted disk.

Usage

When the connection is used from a RAPID program or the FlexPendant, it is referenced with the parameter *Name*.

Allowed values

A string with maximum 40 characters.

Example

The remote mounted disk is used for a connection with unit `C:` on a remote PC. If *Name* is set to `nfs1`, the file `C:\nfs.modx` can be accessed from a RAPID program as `nfs1/nfs.modx`.

2.12.3 Server Address

Parent

Server Address belongs to the type *NFS Client*, in the topic *Communication*.

Description

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

Usage

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

Allowed values

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

2.12.4 Server Type

Parent

Server Type belongs to the type *NFS Client*, in the topic *Communication*.

Description

The type of server to which the NFS client is connected.

Allowed values

XLink or *Default*.

Note

This value can be left empty for all servers apart from *XLink*.

Related information

Application manual - Controller software OmniCore

2.12.5 Trusted

Parent

Trusted belongs to the type NFS Client, in the topic Communication.

Description

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

Usage

A 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 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 OmniCore

2.12.6 User ID

Parent

User ID belongs to the type *NFS Client*, in the topic *Communication*.

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*. For more information, see *Group ID on page 116*.

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.

Allowed values

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

Default value is 0.

Related information

Application manual - Controller software OmniCore

2.12.7 Group ID

Parent

Group ID belongs to the type *NFS Client*, in the topic *Communication*.

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*. For more information, see *User ID on page 115*.

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.

Allowed values

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

Default value is 0.

Related information

Application manual - Controller software OmniCore

2.12.8 Show Device

Parent

Show Device belongs to the type *NFS Client*, in the topic *Communication*.

Description

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

Allowed values

Yes or No.

2.13 Type OPC UA Server

2.13.1 The OPC UA Server type

Overview

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

Type description

The type *OPC UA Server* is used to configure OPC UA server.

Related information

Application manual - Embedded OPC UA Server

2.13.2 Enabled

Parent

Enabled belongs to the type *OPC UA Server*, in the topic *Communication*.

Description

Enabled defines if the integrated OPC UA server is started or not.

Allowed values

Yes or No.

Default value is No.

2.14 Type Port Forward

2.14.1 The Port Forward type

Overview

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

Type description

The port forwarding configuration can be used to reach a server/device located on a different network in the controller. One example is having a built-in webserver on a device on the IO network that needs to be reached from a Private or Public network.

This functionality uses Network Address Translation (NAT) and will automatically opens the associated port in the controller firewall. For the communication to work in both ways, the server/device must have a default gateway set that points back to the controller. For instance, a server/device on the private network needs to have a default gateway set to 192.168.125.1 which is the controller address on that network.

The following forwarding is possible:

- Public Network -> Private Network
- Public Network -> I/O Network
- Private Network -> I/O Network

Note

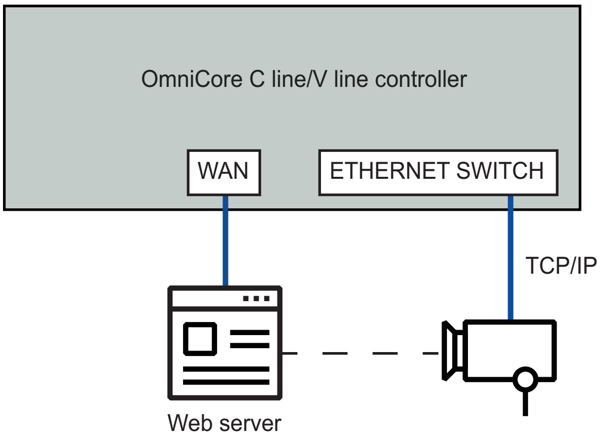
Since network traffic will flow through the controller, high bandwidth applications could negatively affect the controller performance.

Continues on next page

Examples

Public Network -> Private Network on C line/V line

Topology where the web server on a camera on the Private Network needs to be viewed on the web browser on a PC residing on the Public Network:



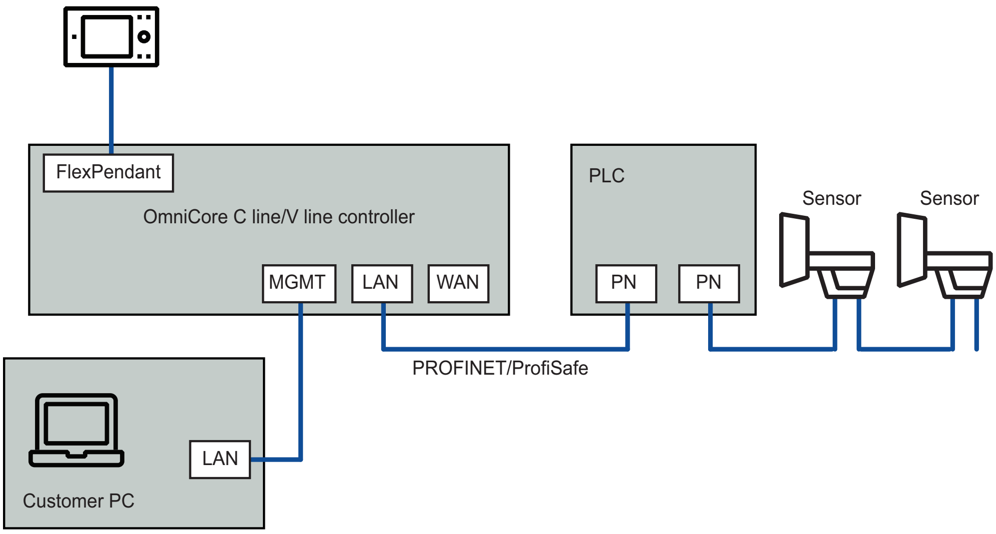
xx2300000737

Corresponding system parameter configuration:

```
SIO:CFG_1.0:7:0::
#
PORT_FORWARD:
-Name "Camera" -ListenNetwork "Public Network" -ListenPort 8080 \
-ForwardToNetwork "Private Network" -ForwardToAddress
     "192.168.125.201" \
-ForwardToPort 80 -Protocol "tcp"
```

Private Network -> I/O Network on C line/V line

Topology where sensors on the I/O network needs to be configured with a PC tool connected to the management port (Private Network):



xx2300000738

Corresponding system parameter configuration:

```
SIO:CFG_1.0:7:0::
```

Continues on next page

```
#
PORT_FORWARD:
-Name "Sensor" -ListenNetwork "Private Network" -ListenPort 8080
     \
-ForwardToNetwork "IO Network" -ForwardToAddress "172.16.0.2" \
-ForwardToPort 80 -Protocol "tcp"
```

Related information

Operating manual - Integrator's guide OmniCore.

2.14.2 Name

Parent

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

Description

Specifies a name that to be used for identification.

Allowed values

A string with maximum 80 characters.

2.14.3 Listen Network

Parent

Listen Network belongs to the type *Port Forward*, in the topic *Communication*.

Description

Specifies which of the controller networks will be listening for the incoming connections.

Allowed values

A string with the value:

- Public Network
- Private Network

2.14.4 Listen Port

Parent

Listen Port belongs to the type *Port Forward*, in the topic *Communication*.

Description

Specifies which port on the controller will be opened to listen for the incoming connections.

Allowed values

A number between 1025 and 65535.

Note

Specifying a port number that is already in use by another service on the controller will block that communication.

2.14.5 Forward to Network

Parent

Forward to Network belongs to the type *Port Forward*, in the topic *Communication*.

Description

Specifies on which of the controller networks the server exists where the communication is forwarded to.

Allowed values

A string with the value:

- Private Network
- IO Network

2.14.6 Forward to Address

Parent

Forward to Address belongs to the type *Port Forward*, in the topic *Communication*.

Description

Specifies the IP address of the server where the communication is forwarded to.

Allowed values

Any IP address between 0.0.0.0 and 255.255.255.255

2.14.7 Forward to Port

Parent

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

Description

Specifies the port number of the server where the communication is forwarded to.

Allowed values

A number between 1 and 65535.

2.14.8 Protocol

Parent

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

Description

Specifies the protocol, which needs to be either TCP or UDP, for the communication.

Allowed values

A string containing tcp or udp.

2.15 Type SFTP Client

2.15.1 The SFTP Client type

Overview

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

Type description

The type *SFTP Client* is used to configure SFTP.

Appropriate RobotWare options need to be installed. For more details, see *Application manual - Controller software OmniCore*.

Related information

Application manual - Controller software OmniCore.

2.15.2 Name

Parent

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

Description

The name used when accessing the SFTP mounted disk.

Usage

When the connection is used from a RAPID program or the FlexPendant, it is referenced with the parameter *Name*.

Allowed values

A string with maximum 40 characters.

Example

The remote mounted disk is used for a connection with unit `C:` on a remote PC. If *Name* is set to `sftp1`, the file `C:\sftp.modx` can be accessed from a RAPID program as `sftp1/sftp.modx`.

2.15.3 Server Address

Parent

Server Address belongs to the type *SFTP Client*, in the topic *Communication*.

Description

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

Usage

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

Allowed values

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

2.15.4 Trusted

Parent

Trusted belongs to the type SFTP Client, in the topic Communication.

Description

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

Usage

A 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 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 OmniCore

2.15.5 Server Path

Parent

Server Path belongs to the type *SFTP Client*, in the topic *Communication*.

Description

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

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Controller software OmniCore

2.15.6 Username

Parent

Username belongs to the type *SFTP Client*, in the topic *Communication*.

Description

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

Usage

Create a user account on the SFTP server. The user name of this account is then specified in *Username*, and the password in *Password*. For more information, see *Password on page 136*.

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Controller software OmniCore

2.15.7 Password

Parent

Password belongs to the type *SFTP Client*, in the topic *Communication*.

Description

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

Usage

Create a user account on the SFTP server. The user name of this account is then specified in *Username*, and the password in *Password*. For more information, see *Username on page 135*.

The password will be, if not already, encrypted. The encrypted string will map to the original password only on the original controller.

Allowed values

A string with a maximum of 40 characters. Encrypted strings can be longer.

Related information

Application manual - Controller software OmniCore

2.15.8 Show Device

Parent

Show Device belongs to the type *SFTP Client*, in the topic *Communication*.

Description

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

Allowed values

Yes or No.

2.15.9 Fingerprint

Parent

Fingerprint belongs to the type *SFTP Client*, in the topic *Communication*.

Description

To guarantee that the controller connects to the expected SFTP server, and not a malicious server, a server fingerprint can be used.

Usage

If *Fingerprint* is set to another value than the default value (only zeros), the fingerprint value from the SFTP server must match this value. If the fingerprint does not match, it will not be possible to connect to the SFTP server.

Limitations

Fingerprint is used only with the RobotWare option *FTP & SFTP Client*.

Allowed values

20 two-digit hexadecimal numbers, separated by colon.

Default value is "00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00".

Related information

Application manual - Controller software OmniCore, section *SFTP Client*.

2.16 Type Syslog

2.16.1 The Syslog type

Overview

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

Type description

The type *Syslog* contains parameters to enable forwarding of log events to a remote syslog server.

If a *Server address* and a *Server port number* are defined, syslog event messages can be sent to a remote syslog server using the transport protocol UDP.

The parameter *Protocol* cannot be modified.

Additional information

A syslog entry is written with the following format:

```
<priority>timestamp hostname application:message
```

The message corresponds to:

```
<event text>|<extra info>|<event id>|<sequence no>|<source>|<product
     name>|<user name>
```

In its entirety, a syslog entry will have the following format:

```
<priority>timestamp hostname application:<event text>|<extra
     info>|<event id>|<sequence no>|<source>|<product name>|<user
     name>
```

Sample events logged in the Syslog

- User login successful
- User login failed, unknown user
- User login failed, incorrect password
- User login failed 3 consecutive attempts
- User logout
- Configuration uploaded
- System parameters changed successfully
- Software updated successfully
- Software failed unsuccessfully
- Control system restarted
- System time updated manually
- Previous configuration successfully restored

Example 1

A sample of the syslog message generated when a user logs in:

```
<13>May 14 11:22:33 VC-1 ABB-UAL:01:Log-in
     successful|-|UAS_UID=983055|1110|00339|-|robot|Default User
```

Continues on next page

Example 2

A sample of the syslog message generated when a user logs out:

```
<13>May 14 11:22:33 VC-1 ABB-UAL:01:Log-out (user logged
     out)|-|Socket=7372|1210|00341|-|robot|
```

2.16.2 Server address

Parent

Server address belongs to the type *Syslog*, in the topic *Communication*.

Description

Server address specifies the IP address of the remote syslog server.

Allowed values

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

Example

An IP address consists of four parts, each with eight bits, separated by dots.

`100.100.100.100` or `138.227.1.45`.

2.16.3 Server port number

Parent

Server port number belongs to the type *Syslog*, in the topic *Communication*.

Description

Server port number specifies the port number on the network node identified by *Server address*.

Allowed values

An integer value between 0 and 65535.

Note

The default port number for syslog events is 514.

Related information

Server address on page 141

2.16.4 Enabled

Parent

Enabled belongs to the type *Syslog*, in the topic *Communication*.

Description

This defines if the transfer to the syslog server is turned on or off.

Default value

The default value is *No*.

Allowed values

Yes or *No*

2.17 Type UDP Unicast Device

2.17.1 The UDP Unicast Device type

Overview

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

Type description

The type *UDP Unicast Device* (UdpUc) is a generic device type, that can be used by different applications.

2.17.2 Name

Parent

Name belongs to the type *UDP Unicast Device*, in the topic *Communication*.

Description

Name specifies the name of the *UDP Unicast Device* instance.

Allowed values

A string with maximum 16 characters.

2.17.3 Type

Parent

Type belongs to the type *UDP Unicast Device*, in the topic *Communication*.

Description

Type defines the type of *UDP Unicast Device* protocol to be used.

Allowed values

The only available *UDP Unicast Device* type is *UDPUC*.

2.17.4 Remote Address

Parent

Remote Address belongs to the type *UDP Unicast Device*, in the topic *Communication*.

Description

Remote Address specifies the IP address of the external device, for example, sensor.

Allowed values

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

Example

An IP address consists of four parts, each with eight bits, separated by dots.

`100.100.100.100` or `138.227.1.45`.

2.17.5 Remote port number

Parent

Remote port number belongs to the type *UDP Unicast Device*, in the topic *Communication*.

Description

Remote port number specifies the port number on the network node identified by *Remote Address*.

Allowed values

An integer value between 0 and 65535.

Related information

Remote Address on page 147

2.17.6 Local port number

Parent

Local port number belongs to the type *UDP Unicast Device*, in the topic *Communication*.

Description

Local port number specifies the port number on which the controller will listen for broadcast messages.

Allowed values

An integer value between 0 and 65535.

0 means that the controller software itself selects an available port number.

Note

The local port number cannot be 6510, since this is reserved for the remote port.

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.

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**. For more information, see *The Operator Safety type on page 221*.
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

Operating manual - OmniCore

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. For more information, see *The Path Return Region type on page 227*.

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.

3.2.3 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 **Controller**, 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.
4. Save the changes.
5. Restart the controller.

For more information, see *The System Input type on page 236* and *The Signal type on page 393*.

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).

3.2.4 How to define system outputs

Overview

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

Prerequisites

A digital output 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 output I/O signal. However, several output 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 output 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 outputs

To define a system output:

1. In the topic **Controller**, choose the type **System Input**.
2. Select the system output to change, delete, or add a new one.
3. Enter, change, or delete the values for the parameters.
4. Save the changes.
5. Restart the controller.

For more information, see *Type System Output on page 282*.

Rejected system outputs

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).

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.

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*.

Allowed values

AllDebugSettings (cannot be changed).

3.3.3 Reset

Parent

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

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.

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**. For more information, see *The Task type on page 330*.

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.

Related information

Technical reference manual - RAPID Overview

Continues on next page

ELOG messages are described in *Technical reference manual - Event logs for RobotWare 7*

Restarts are described in *Operating manual - OmniCore*

3.4.2 File

Parent

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

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:MyModule.modx`

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*.

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*. For more information, see *The Task type on page 330*, *All Tasks on page 165*, and *Shared on page 164*.

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

Application manual - Controller software OmniCore.

3.4.4 Installed

Parent

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

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. For more information, see *All Tasks on page 165*.

Usage

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

Limitations

Cannot be combined with *Shared*.

For more information, see *Shared on page 164*.

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

Technical reference manual - RAPID Overview.

3.4.5 Shared

Parent

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

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*. For more information, see *All Tasks on page 165*, *Task on page 162*, and *Installed on page 163*.

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.

3.4.6 All Tasks

Parent

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

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*. For more information, see *Task on page 162*, *Shared on page 164*, and *The Task type on page 330*.

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).

3.4.7 All Motion Tasks

Parent

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

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*. For more information, see *Task on page 162*, *Shared on page 164*, and *The Task type on page 330*.

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).

3.4.8 Hidden

Parent

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

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 Cyclic Bool Settings

3.5.1 The Cyclic Bool Settings type

Overview

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

Type description

The type *Cyclic Bool Settings* defines the behavior of the cyclic bool functionality.

System restart

All changes in the type *Cyclic Bool Settings* will take effect after a normal restart, or using the restart mode Reset RAPID.

3.5.2 Name

Parent

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

Description

The name of the cyclic bool setting.

Usage

There can be only one instance of each allowed value, that is a maximum of three instances in the system. All three instances will be installed in the system (by default) and cannot be removed.

Allowed values

Value	Description
RemoveAtPpToMain	Defines if connected cyclic bool is to be removed when setting PP to Main
ErrorMode	Defines which error mode to use when evaluation fails
RecoveryMode	Defines which recovery mode to use when evaluation fails

Related information

Value on page 170
ErrorMode on page 171
RecoveryMode on page 172
RemoveAtPpToMain on page 173

3.5.3 Value

Parent

Value belongs to the type *Cyclic bool setting*, in the topic *Controller*.

Description

Defines the cyclic bool values for the value defined in parameter *Name*. For more information, see *Name on page 169*.

Allowed values

The allowed values depend on the parameter *Name*, and are described on the following pages:

- *ErrorMode on page 171*
- *RecoveryMode on page 172*
- *RemoveAtPpToMain on page 173*

3.5.4 Values for the parameter *Value*

3.5.4.1 ErrorMode

Parent

ErrorMode is an action value for the parameter *Name* that belongs to the type *Cyclic Bool Settings*, in the topic *Controller*.

Description

The action value *ErrorMode* is used to configure how to handle failure when evaluating a connected cyclic bool.

Limitations

The behavior can only be configured for all tasks that is, the behavior cannot be different from one task to another.

Allowed values

Name	*Value*	**Description**
ErrorMode	SysStopError (Default)	Stop RAPID execution and produce error log if evaluation of cyclic bool fails.
	Warning	Produce warning log if evaluation of cyclic bool fails.
	None	Ignore any failing cyclic bool.

3.5.4.2 RecoveryMode

Parent

RecoveryMode is an action value for the parameter *Name* that belongs to the type *Cyclic Bool Settings*, in the topic *Controller*.

Description

The action value *RecoveryMode* is used to configure if to recover a failing connected cyclic bool or not.

Limitations

The behavior can only be configured for all tasks that is, the behavior cannot be different from one task to another.

It cannot be disabled if action value *ErrorMode* is set to value *SysStopError*.

Allowed values

Name	*Value*	**Description**
RecoveryMode	On (Default)	Try to recover when evaluation fails.
	Off	Remove cyclic bool that fails during evaluation.

3.5.4.3 RemoveAtPpToMain

Parent

RemoveAtPpToMain is an action value for the parameter *Name* that belongs to the type *Cyclic Bool Settings*, in the topic *Controller*.

Description

The action value *RemoveAtPpToMain* is used to configure if a connected cyclic bool shall be removed or not, when PP is set to Main.

Limitations

The behavior can only be configured for all tasks that is, the behavior cannot be different from one task to another.

Allowed values

Name	*Value*	**Description**
RemoveAtPpToMain	On	Remove all connected cyclic bool setting PP to Main.
	Off	Do not remove all connected cyclic bool when setting PP to Main

3.6 Type Event Routine

3.6.1 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.

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*. For more information, see *The Task type on page 330*.

> **Note**
>
> The RAPID code in the event routine shall be written in such a way that it is executed as fast as possible without any delay.

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 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.

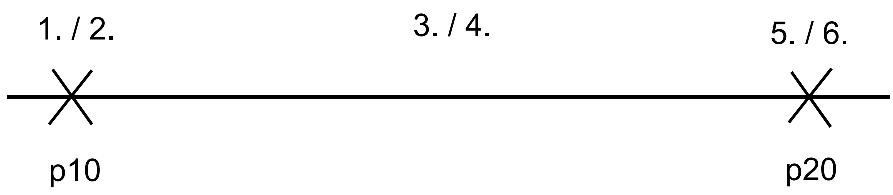
The only way to cancel a stopped event routine from system I/O is to start the program from main.

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 176*.

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

Technical reference manual - RAPID Overview.

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

3.6.2 Routine

Parent

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

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.

3.6.3 Event

Parent

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

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 - OmniCore*.

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 will run when the controller is restarted (restart) from a remote client or by power on. If the routine cannot be started due to a program error, it will run at the next normal start of RAPID if the error is resolved. The Power On event routine will not be stopped by emergency stop, but can be stopped by pressing the stop button on the FlexPendant.
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 stopped with category 0 stop (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 - OmniCore

3.6.4 Sequence Number

Parent

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

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.

3.6.5 Task

Parent

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

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*. For more information, see *The Task type on page 330*, *All Tasks on page 183*, and *All Motion Tasks on page 184*.

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.

3.6.6 All Tasks

Parent

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

Description

All Tasks defines if the routine will run in all configured tasks in the system. For more information, see *Task on page 182* and *The Task type on page 330*.

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.

If *Exclude from Load Modules in All Task* is set for a task, that task will also be excluded when running event routines for *All Tasks*.

3.6.7 All Motion Tasks

Parent

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

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*. For more information, see *Task on page 182* and *The Task type on page 330*.

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.

3.7 Type Fan Control

3.7.1 The Fan Control type

Overview

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

Type description

The cooling fan on OmniCore C30 controllers will work on reduced speed or shut off while the controller is in motors off state to lower the sound level. This is called fan control functionality.

The fan will run with full cooling capacity when the controller is in motors on state. When changing to motors off, the fan will shut off if the temperature on the incoming air is low enough, or run in reduced speed if the temperature is too high.

Related information

Product manual - OmniCore C30

3.7.2 Name

Parent

Name belongs to the type *Fan Control*, in the topic *Controller*.

Description

The default value is Turn Off Fan. The value cannot be changed.

3.7.3 Value

Parent

Value belongs to the type *Fan Control*, in the topic *Controller*.

Description

The parameter *Value* defines if the fan should shut off in motors off state when the temperature on the incoming air to the controller is low enough.

Allowed values

Yes or No.

The default value is Yes.

3.8 Type General Rapid

3.8.1 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.

Type description

General Rapid contains parameters that are general for the controller.

3.8.2 Name

Parent

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

Description

Name defines the ID of the actions listed below. The parameter *Value* defines the value for the instance.

Limitations

There can be only one instance with *Name* set to each of the allowed values.

Allowed values

The following values are allowed and are described on the following pages:

- *BrakeMaintenance on page 191*
- *CollisionErrorHandling on page 192*
- *MathLibMemSize on page 193*
- *ModalPayLoadMode on page 194*
- *NoOfRetry on page 195*
- *PayLoadsInWristCoords on page 196*
- *QueueBackup on page 197*
- *RapidInstructionsPerMs on page 198*
- *RapidLogging on page 199*
- *ReportNonActiveSingles on page 200*
- *SimulateMenu on page 201*
- *StationaryPayLoadMode on page 202*
- *StepOutNoStepin on page 203*
- *TruncateLongRapidStrings on page 204*

Related information

Value on page 190

3.8.3 Value

Parent

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

Description

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

Allowed values

The allowed values depend on the parameter *Name*, and are described on the following pages:

- *BrakeMaintenance on page 191*
- *CollisionErrorHandling on page 192*
- *MathLibMemSize on page 193*
- *ModalPayLoadMode on page 194*
- *NoOfRetry on page 195*
- *PayLoadsInWristCoords on page 196*
- *QueueBackup on page 197*
- *RapidInstructionsPerMs on page 198*
- *RapidLogging on page 199*
- *ReportNonActiveSingles on page 200*
- *SimulateMenu on page 201*
- *StationaryPayLoadMode on page 202*
- *StepOutNoStepin on page 203*
- *TruncateLongRapidStrings on page 204*

3.8.4 Values for the parameter *Name*

3.8.4.1 BrakeMaintenance

Parent

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

Description

BrakeMaintenance (BM) is a feature in the Cyclic Brake Check (CBC) functionality.

Usage

CBC automatically detects if maintenance of the mechanical brakes is needed and then activates the BM functionality during CBC execution. There are event logs that shows if the brake maintenance has been run and also shows the result of the maintenance.

Allowed values

Name	*Value*	**Description**
BrakeMaintenance	Yes	*BrakeMaintenance* is activated. This is the default value.
	No	*BrakeMaintenance* is deactivated. CBC runs as normal, but without brake maintenance.

Additional information

Changes are activated directly. No restart is required.

Related information

For more information about Cyclic Brake Check, see *Application manual - Functional safety and SafeMove*.

3.8.4.2 CollisionErrorHandling

Parent

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

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 will take effect immediately. It does not need a normal restart.

Related information

See *Collision Detection* in *Application manual - Controller software OmniCore*.

Technical reference manual - RAPID kernel

3.8.4.3 MathLibMemSize

Parent

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

Description

The action value *MathLibMemSize* specifies the amount of memory allocated for use by mathematical instructions in RAPID.

Usage

With this parameter it is possible to increase memory used for mathematical RAPID instructions, such as `MatrixX` and `FitX`, so they can handle bigger input data.

Allowed values

An integer value between 500000 and 20000000.

Normally the default value is enough for most calculations and should only be changed if event log *41807, Not enough memory* is reported.

Default value is 500000.

Additional information

Changes are activated after a normal restart.

3.8.4.4 ModalPayLoadMode

Parent

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

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	Yes	ModalPayLoadMode shall be used. Any payload is set by the `GripLoad` instruction. This is a default value.
	No	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*.

3.8.4.5 NoOfRetry

Parent

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

Description

The action value *NoOfRetry* specifies the number of times the routine with a recoverable error is called before the error is reported as fatal and execution is stopped.

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.

Allowed values

An integer value between 0 and 1000.

Additional information

Changes are activated after a normal restart.

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
```

3.8.4.6 PayLoadsInWristCoords

Parent

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

Description

PayLoadsInWristCoords defines whether or not this mode shall be used. The *PayLoadsInWristCoords* will only have impact when the tool holds an additional payload.

Usage

Can be useful, for example, if several tool/TCP (Tool center Point) or work objects (when stationary tool) are used for one payload. In this case only one Load Identification is needed instead of one for each tool/TCP or work object.

Limitations

The parameter *PayLoadsInWristCoords* will only impact if an additional payload is used beyond the tool.

Allowed values

Name	*Value*	Description
PayLoadsInWristCoords	No	*PayLoadsInWristCoords* shall not be used, any payload is added relative to the TCP or work object.
	Yes	*PayLoadsInWristCoords* shall be used. Any payload is added relative to the wrist. This is the default value.

Additional information

Changes are activated after a normal restart.

The default value is changed compared with IRC5.

Related information

For more information about how loads are added, see*Technical reference manual - RAPID Instructions, Functions and Data types*, `loaddata` and `GripLoad`.

3.8.4.7 QueueBackup

Parent

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

Description

QueueBackup is used when the system input signal *Disable Backup* is frequently used, and there is a risk that a backup not will be taken. If *QueueBackup* is set to *TRUE*, the backup will be queued until the signal is reset.

In automatic mode the backup will also be queued while executing and the system input signal is low. The signal must then be set and reset before a backup is performed. This is to prevent that a backup is started just before entering or during a sensitive path in the program when the signal for *Disable backup* is used.

Usage

To queue a backup can be useful if there is no cross-checking done between taking a backup and running robot movements, where a backup can disturb the process, for example, when gluing.

Allowed values

Name	*Value*	**Description**
QueueBackup	No	*QueueBackup* is deactivated. Any backup that is ordered is neglected if the system parameter *Disable Backup* is set. This is the default value.
	Yes	*QueueBackup* is activated. Any backup that is ordered will be queued until the system parameter *Disable Backup* is reset.

Note

The changes are effective after a normal start.

Related information

Operating manual - OmniCore section *Important when performing backups*.

Operating manual - Integrator's guide OmniCore section *Back up the system*.

Disable Backup on page 242.

3.8.4.8 RapidInstructionsPerMs

Parent

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

Description

RapidInstructionsPerMs defines how many RAPID instructions are executed per ms by the controller.

Allowed values

The default value should not be changed.

3.8.4.9 RapidLogging

Parent

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

Description

The parameter *RapidLogging* enables logging of RAPID execution. It is very useful when sending system diagnostic data for troubleshooting.

Usage

This functionality is useful when finding errors related to the RAPID program. The logging is included in the system diagnostic and can be used by ABB personnel.

Allowed values

Yes or *No*

The default value is Yes.

3.8.4.10 ReportNonActiveSingles

Parent

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

Description

ReportNonActiveSingles is used to warn user and stop execution in case a single, that is not active, has a valid value in RAPID.

Setting this value to *Yes* will stop execution if the position value in RAPID differs from 9E9, for a deactivated single. This can help prevent unexpected movement when a single axis is not consistent between RAPID value and activation status.

Usage

When having mechanical units that can be deactivated, this parameter will prevent the robot from moving.

Allowed values

ReportNonActiveSingles	Description
No	*ReportNonActiveSingles* is deactivated. The default value is No.
Yes	*ReportNonActiveSingles* is activated.

Additional information

Changes are activated after a normal restart.

3.8.4.11 SimulateMenu

Parent

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

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 *No* 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.

3.8.4.12 StationaryPayLoadMode

Parent

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

Description

StationaryPayLoadMode defines whether or not this mode should be used. The *StationaryPayLoadMode* only have effect when 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	No	*StationaryPayLoadMode* shall not be used, any payload is added relative to the work object when a stationary tool is used. This is the default value.
	Yes	*StationaryPayLoadMode* shall be used. Any payload is added relative to the wrist when a stationary tool is used.

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*, `loaddata` and `GripLoad`.

3.8.4.13 StepOutNoStepin

Parent

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

Description

StepOutNoStepin is used when changing direction from forward to backward step when the program pointer is inside a nostepin routine.

Usage

StepOutNoStepin is used to be able to change from forward to backward step when the program pointer is inside a nostepin routine. If any UNDO-handler available in the the nostepin routine, then it is run. Regardless of the UNDO-handler is run or not, the program pointer will be set to the instruction above the call to the nostepin routine.

If *StepOutNoStepin* is not activated, the change of direction will result in rejection of backward step.

Allowed values

Name	*Value*	**Description**
StepOutNoStepin	No	*StepOutNoStepin* is deactivated. This is the default value.
	Yes	*StepOutNoStepin* is activated.

Additional information

Changes are activated after a normal restart.

3.8.4.14 TruncateLongRapidStrings

Parent

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

Description

TruncateLongRapidStrings defines if long RAPID strings should be truncated or not.

The parameter is used in combination with the RAPID instructions `StrFormat` and `TextGet`, as well as when combining two strings (for example, `string1 := string2 +string3;`).

If *TruncateLongRapidStrings* is activated, the program will not be terminated even if the RAPID string is too long. Instead, the string will be truncated and a warning will be displayed.

Usage

Normally, strings that exceed the limit of 80 bytes will cause termination of the program. Using this function, the program will not be terminated even if the strings are too long. Instead, the strings will be truncated.

Allowed values

Name	*Value*	**Description**
TruncateLongRapidStrings	No	*TruncateLongRapidStrings* is deactivated. This is the default value.
	Yes	*TruncateLongRapidStrings* is activated.

Additional information

Changes are activated after a normal restart.

Related information

For more information about `StrFormat`, `TextGet` see*Technical reference manual - RAPID Instructions, Functions and Data types*.

3.9 Type Mechanical Unit Group

3.9.1 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. For more information, see *Use Mechanical Unit Group on page 337*.

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

Application manual - MultiMove.

3.9.2 Name

Parent

Name belongs to the type *Mechanical Unit Group*, in the topic *Controller*. For more information, see *Use Mechanical Unit Group on page 337*.

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.

3.9.3 Robot

Parent

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

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. For more information, see *Name on page 206*.

Limitations

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

Allowed values

A string with maximum 32 characters.

3.9.4 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* belong to the type *Mechanical Unit Group*, in the topic *Controller*.

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. For more information, see *Name on page 206*.

Limitations

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

Allowed values

A string with maximum 32 characters.

3.9.5 Use Motion Planner

Parent

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

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. For more information, see *The Motion Planner type on page 634* in the topic *Motion*.

Limitations

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

Allowed values

A string with maximum 32 characters.

3.10 Type ModPos Settings

3.10.1 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.

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.

3.10.2 Name

Parent

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

Description

Name defines that the parameter configuration is for ModPos.

Allowed values

modpos

Related information

Operating manual - OmniCore

3.10.3 Limited ModPos

Parent

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

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.

3.10.4 Mode

Parent

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

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. For more information, see *Limited ModPos on page 212*.

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.

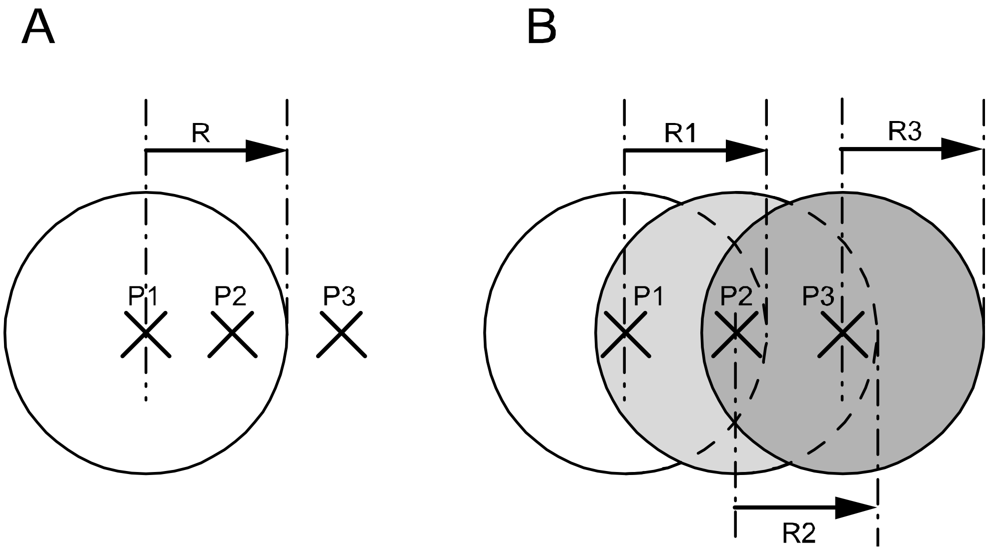
Example

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

3.10.5 Limit Trans

Parent

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

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. For more information, see *Limited ModPos on page 212*.

Allowed values

0 - 1000 mm.

Default value is 5.

3.10.6 Limit Rot

Parent

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

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. For more information, see *Limited ModPos on page 212*.

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)`

3.10.7 Limit External Trans

Parent

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

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. For more information, see *Limited ModPos on page 212*.

Allowed values

0 - 1000 mm.

Default value is 50.

3.10.8 Limit External Rot

Parent

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

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. For more information, see *Limited ModPos on page 212*.

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)`

3.11 Type Move in Auto

3.11.1 The Move In Auto type

Overview

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

Description

Some robots are allowed to be moved in AUTO mode which normally is restricted. To use this functionality, a system input (*MoveInAuto*) must be used to prove the local presence. It is then possible to move the robot, since it will get a certificate key when the system input is set, that are used when jogging. The certificate key will be valid only for a certain time, and that time can be configured with the parameter *MoveInAuto*.

Usage

The type *Move In Auto* defines the timeout or how long the local presence certificate key is valid.

All changes in the type *Move In Auto* will take effect immediately and no restart of the controller is required.

Note

Available only for some robots.

3.11.2 Timeout

Parent

Timeout belongs to the type *Move In Auto*, in the topic *Controller*.

Description

The timeout in seconds for allowing the movement of the robot in AUTO mode and is valid when using a local presence certificate key.

Usage

The default value is 30, and there is normally no reason to change this value.

Limitations

The parameter *Timeout* is used only if you have this functionality .

Allowed values

An integer between 10 and 3000.

Default value is 30.

3.12 Type Operator Safety

3.12.1 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.

Type description

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

Related information

Operating manual - OmniCore

3.12.2 Function

Parent

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

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). For more information, see *How to activate hold-to-run control on page 152*.
AllowMoveRobAuto	*AllowMoveRobAuto* is only available for robots that are allowed to be moved manually in automatic mode. If the function is activated it is possible to move the robot in automatic mode without executing. *AllowMoveRobAuto* is not available for all robots.
HeartBeat	When using an external cableless device to start RAPID execution, there is a risk if the robot starts to move and the communication is lost. For those applications, it is vital to ensure that the device has contact with the controller the whole time during execution. *HeartBeat* must then be enabled.
ZeroSpeedEMStop	When *ZeroSpeedEMStop* is activated, 0% speed is set and kept when the emergency stop button is pressed.

Related information

Operating manual - OmniCore

3.12.3 Active

Parent

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

Description

Active defines whether the value of *Function* is activated. For more information, see *Function on page 222*.

Allowed values

Value	Description
TRUE	Activated
FALSE	Not activated

The default value is FALSE.

3.13 Type Options

3.13.1 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.

Type description

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

3.13.2 Name

Parent

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

Description

Short unique ID of an option.

Usage

Uniquely identifies an option.

Limitations

Read-only

3.13.3 Description

Parent

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

Description

Complete name of an option.

Usage

Human friendly identification of an option.

Limitations

Read-only

3.14 Type Path Return Region

3.14.1 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.

Type description

In a return movement, the path return region specifies the distance from the current robot position to the last executed path.

There are three sets of parameters defined for this type; automatic mode (AUTO), manual mode (MAN), and StartMove. All 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

StartMove

3.14.2 Mode

Parent

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

Description

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

Usage

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

Allowed values

AUTO

MAN

StartMove

3.14.3 TCP Distance

Parent

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

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*. For more information, see *Mode on page 228*.

Allowed values

Mode	Value
Auto	0-2.00 meters, specifying the movement in meters. Default value is 0.5 meter.
StartMove	0-2.00 meters, specifying the movement in meters. Default value is 0.01 meter.
Manual	0-2.00 meters, specifying the movement in meters. Default value is 0.05 meter.

Related information

Application manual - Controller software OmniCore.

3.14.4 TCP Rotation

Parent

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

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*. For more information, see *Mode on page 228*.

Allowed values

Mode	Value
AUTO	0-6.280, specifying the movement in radians. Default value is 1.57 radians.
MAN	0-6.280, specifying the movement in radians. Default value is 0.2 radians.
StartMove	0-6.280, specifying the movement in radians. Default value is 0.35 radians.

Additional information

To convert degrees to radians, use this formula:

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

Related information

Application manual - Controller software OmniCore.

3.14.5 External Distance

Parent

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

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*. For more information, see *Mode on page 228*.

Allowed values

Mode	Value
AUTO	0-2.000, specifying the movement in meters. Default value is 0.5 meter.
MAN	0-2.000, specifying the movement in meters. Default value is 0.05 meter.
StartMove	0-2.000, specifying the movement in meters. Default value is 0.01 meter.

Related information

Application manual - Controller software OmniCore.

3.14.6 External Rotation

Parent

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

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*. For more information, see *Mode on page 228*.

Allowed values

Mode	Value
AUTO	0-2.000, specifying the movement in meters. Default value is 1.57 radians.
MAN	0-6.280, specifying the movement in radians. Default value is 0.2 radians.
StartMove	0-6.280, specifying the movement in radians. Default value is 0.35 radians.

Additional information

To convert degrees to radians, use this formula:

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

Related information

Application manual - Controller software OmniCore.

3.15 Type Run Mode Settings

3.15.1 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.

Type description

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

3.15.2 Name

Parent

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

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.

3.15.3 Switch

Parent

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

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.

3.16 Type System Input

3.16.1 The System Input type

Overview

This section describes the type *System Input* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section. For more information, see *How to define system inputs on page 154*.

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.

Additional information

Most system inputs are 0 to 1 level sensitive. The pulse length must exceed 50 ms or according to the configured filter settings for I/O signals.

The following *System Input* signals are both 0 to 1 and 1 to 0 level sensitive. For more information, see *Filter Time Passive on page 408* and *Filter Time Active on page 409*.

- *Collision avoidance*
- *Enable Energy Saving*
- *Limit Speed*
- *SimMode*
- *Write Access*

3.16.2 Signal Name

Parent

Signal Name belongs to the type *System Input* in the topic *Controller*, see *The Signal type on page 393*.

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, see *The Signal type*.

Allowed values

Available configured digital input I/O signal names.

3.16.3 Action

Parent

Action belongs to the type *System Inputs*, in the topic *Controller*.

Description

The parameter *Action* defines the system action to be triggered by the input signal. The system action is handled by the system without input from the user. A PLC can be used to trigger the system action.

Allowed values

The following values are allowed and described on the following pages:

- *Backup on page 239*.
- *Collision Avoidance on page 241*.
- *Disable Backup on page 242*.
- *Enable Energy Saving on page 243*.
- *Interrupt on page 244*.
- *Limit Speed on page 246*.
- *Load on page 249*.
- *Load and Start on page 251*.
- *Motors Off on page 253*.
- *Motors On on page 254*.
- *Motors On and Start on page 255*.
- *PP to Main on page 256*.
- *ProfiSafeOpAck on page 257*.
- *Quick Stop on page 258*.
- *Reset Execution Error Signal on page 259*.
- *Set Speed Override on page 260*
- *SimMode on page 261*.
- *Start on page 262*.
- *Start at Main on page 263*.
- *Stop on page 264*.
- *Stop at End of Cycle on page 265*.
- *Stop at End of Instruction on page 266*.
- *System Restart on page 267*.
- *Trust Revolution Counter on page 268*.
- *Verify Local Presence on page 269*.
- *Verify Move Robot In Auto on page 270*.
- *Write Access on page 271*.

3.16.4 Values for the parameter *Action*

3.16.4.1 Backup

Parent

Backup is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *Controller*.

Description

The action value *Backup* starts a backup as defined by the parameters *Argument*.

Arguments

When the parameter *Action* is set to *Backup*, the following parameters must also be used. For more information, see *Action on page 238*.

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, see *Argument 1 on page 272*.
Argument 3	Specify a path for the backup. Always define the entire path, for example, BACKUP/sysinBackup, see *Argument 3 on page 274*.
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, see *Argument 4 on page 275*.
Argument 5	*AddDate* means that the backup gets the date in the name automatically. *NoDate* means that the name of the backup does not get the date. The date is in YYYYMMDD format and is put at the end of the name but before any sequence number, see *Argument 5 on page 276*.
Argument 8	*Archive* means that the backup is archived and saved as one file with the file suffix .tar. *No Archive* means that the backup is not archived. See *Argument 8 on page 279*.

Prerequisites

A digital input signal must be available, not used by any other resource (system input).

Limitations

The backup order is ignored with a warning if a backup is already in progress. It is also ignored if the the parameter *Action* is set to *Disable Backup*.

Additional information

The system output *Backup Error* tells if the backup was successful or not, see Backup error, see *Backup Error on page 288*.

The system output *Backup in progress* tells if the backup process is active or not, see *Backup in progress on page 289*.

The ordered backup will take the program control during the operation.

Continues on next page

Be aware that the RAPID execution can be disturbed while taking a backup. Please use the System input signal **Disable Backup** (see, *Disable Backup on page 242*) and the action value **QueueBackup** (see, *QueueBackup on page 197*) during critical movements or other RAPID code that shall not be disturbed.

3.16.4.2 Collision Avoidance

Parent

Collision Avoidance is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Collision Avoidance* is used to activate the *Collision Avoidance* functionality.

A high signal will activate the functionality and a low signal will deactivate the functionality. The functionality is by default active if no signal has been assigned to the system input *Collision Avoidance*.

The function *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* determines at what distance the two objects are considered to be in collision, see *Coll-Pred Safety Distance on page 705*.

The *Collision Avoidance* functionality is configured partly in the system parameters (on/off and distance), and the geometric models are configured in RobotStudio.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.3 Disable Backup

Parent

Disable Backup is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Disable Backup* will prevent starting a backup as long as the signal is set.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

If a backup is prevented, it will not be started when the signal gets low unless the functionality to queue the backup is configured. See *QueueBackup on page 197*.

If a backup is ongoing when the signal is set, the backup will continue until it has finished.

3.16.4.4 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 *Controller*.

Description

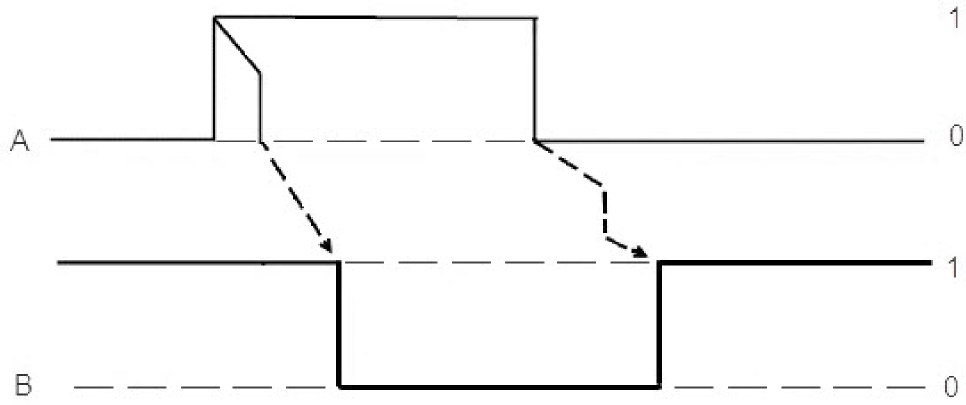
Setting the action value *Enable Energy Saving* enables the controller to enter an energy saving state. Resetting the signal while in an energy saving state will cause the controller to resume.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Signal sequence

The signal sequence for *Enable Energy Saving* is:



xx1500000337

A: Enable Energy Saving (IN)

B: Energy Saving Blocked (OUT)

Additional information

A system output signal (called *Energy Saving Blocked*) can be configured to reflect if energy saving is blocked or not.

It is not only the system input signal *Enable Energy Saving* that can cause the energy saving functionality to be blocked. That is, the system output signal *Energy Saving Blocked* can be set even if the system input action *Enable Energy Saving* is set.

3.16.4.5 Interrupt

Parent

Interrupt is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Interrupt* executes a routine and after running the routine the execution will resume to the same instruction as before. If necessary, 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 following parameters must also be used.

Parameter:	Allowed value:
Argument 1	The name of the routine to be executed, see *Argument 1 on page 272*.
Argument 2	The task in which the routine defined in *Argument 1* should be executed. This is only used with *MultiMove*, see *Argument 2 on page 273*. If *Argument 2* is not set, then the first found motion task is used.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

The parameter has the following limitations:

- The system must be in automatic mode and Motors On.
- You cannot use this action value if the actions *QuickStop*, *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* are set.
- The *Interrupt* action will be applied only when the program execution is stopped.

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
```

Continues on next page

```
  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.

3.16.4.6 Limit Speed

Parent

Limit Speed is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *LimitSpeed* should be set when the speed of one or all motion task should be reduced. The reduction of the speed is considered to be completed when the system output signal *LimitSpeed* is set to 1.

WARNING

There is a time lag of 0.35 to 0.5 seconds for the *LimitSpeed* to be triggered in the robot. Hence, this additional time should be considered when setting up the robot, for example, safety distance for an operator.

The speed limitation is set up with RAPID instructions `SpeedLimAxis` and `SpeedLimCheckPoint` (see *Technical reference manual - RAPID Instructions, Functions and Data types* for further details) or the manual mode default values will be used. The default value for manual mode is defined by the parameter *Teach Mode Max Speed*.

Arguments

When the parameter *Action* is set to *Limit Speed*, the following parameters must also be used.

Parameter:	Allowed value:
Argument 6	The parameter specifies a mechanical unit, see *Argument 6 on page 277*.

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 *Limit Speed* (connected to same robot) can cause unpredictable behavior during power failure restart.

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.

Continues on next page

The maximum allowed acceleration during ramping up is controlled by the system parameter *Limit Speed Acc Limitation* in the type *Motion Planner*.

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.

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.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

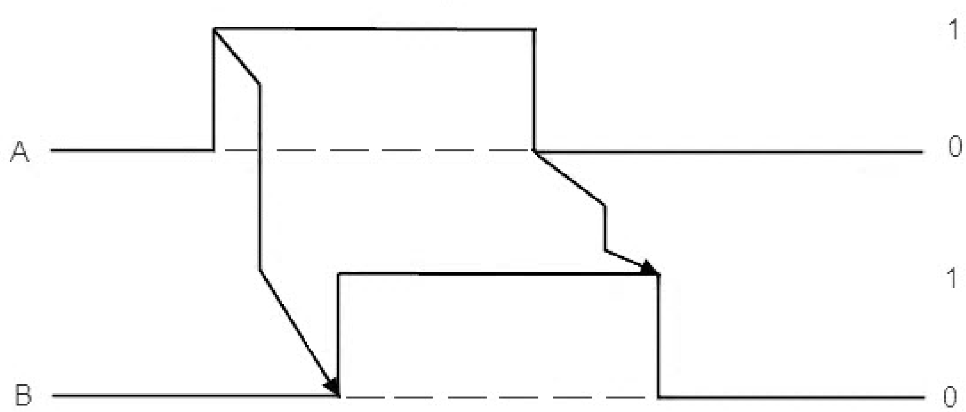
The system input *Limit Speed* should not be used together with applications where it is critical that the robot maintains a specific TCP speed. Also, the system output *Limit Speed* should not be used together with external equipment that requires that a large equipment lag is set up.

Specifically, the following features should not be used together with Limit Speed:

- Motion system parameter *Event Preset Time*
- System outputs *TCP Speed* and *TCP Speed Reference*
- Dispensing, that is, RAPID instructions `DispL`, `DispC`, and `TriggSpeed`
- Arc welding

Signal sequence

The signal sequence for *Limit Speed* is:



en1200000680

A: LimitSpeed (IN)

B: LimitSpeed (OUT)

Continues on next page

Related information

Argument 6 on page 277

System output *Limit Speed on page 296*

Teach Mode Max Speed on page 645

Max acc when ramping up speed on page 670

Technical reference manual - RAPID Instructions, Functions and Data types

3.16.4.7 Load

Parent

Load is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Load* loads a RAPID program (files of type .modx, .mod, .sysx, .sys, and .pgf). The program starts from the beginning.

Note

The previously loaded files (of type .pgf) will be unloaded.

Note

In RobotWare 7.0 and earlier, the formats were `.mod` and `.sys`. When loading these in a RobotWare 7.1 controller or later using RobotStudio, they are automatically converted when saved. When saved, the new file extensions are `.sysx` and `.modx`. Note that the files must be converted, not just renamed.

To convert a file manually, the file must be saved as UTF-8 without BOM (Byte Order Mark).

The program pointer is set to the main entry routine after the program is loaded. Program pointers in other tasks are not affected.

Arguments

When the parameter *Action* is set to *Load*, the following parameters must also be used.

Parameter:	Allowed value:
Argument 1	The name of the program file to load, including the file format (.modx, .sysx, or .pgf). Always define the path to the file, e.g. `HOME:ModuleA.modx`, see *Argument 1 on page 272*.
Argument 2	The task in which the program defined in *Argument 1* should be loaded, see *Argument 2 on page 273*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

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 that program is unloaded.

Continues on next page

Additional information

When a program is loading, all routines connected to the event routine *Reset* will be executed. It is only possible to initiate loading of programs in one task at a time by using system inputs. To load in several tasks, the system output *System Input Busy* can be used so that the current execution of *Reset* routines are finished before the next load is initiated. See *The Event Routine type on page 174*.

3.16.4.8 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 *Controller*.

Description

The action value *Load and Start* loads a RAPID program (files of type .modx, .mod, .sysx, .sys, and .pgf). The program starts from the beginning.

Note

If a program is loaded, all modules of type .modx will be unloaded.

It 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 program is loaded. Program pointers in other tasks are not affected.

Note

In RobotWare 7.0 and earlier, the formats were `.mod` and `.sys`. When loading these in a RobotWare 7.1 controller or later using RobotStudio, they are automatically converted when saved. When saved, the new file extensions are `.sysx` and `.modx`. Note that the files must be converted, not just renamed.

To convert a file manually, the file must be saved as UTF-8 without BOM (Byte Order Mark).

Arguments

When the parameter *Action* is set to *Load and Start*, the following parameters must also be used.

Parameter:	Allowed value:
Argument 1	The name of the program file to load, including the file format (.modx, .sysx, or .pgf). Always define the path to the file, for example `HOME:ModuleA.modx`, see *Argument 1 on page 272*.
Argument 2	The task in which the program defined in *Argument 1* should be loaded, see *Argument 2 on page 273*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

This action value has the following limitations:

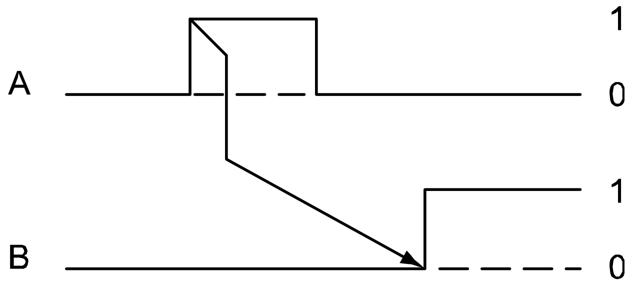
- The controller must be in automatic mode.
- You cannot use this action value if the actions *QuickStop*, *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* are set.

Continues on next page

- *Load and Start* action cannot be executed 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)

3.16.4.9 Motors Off

Parent

Motors Off is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

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.

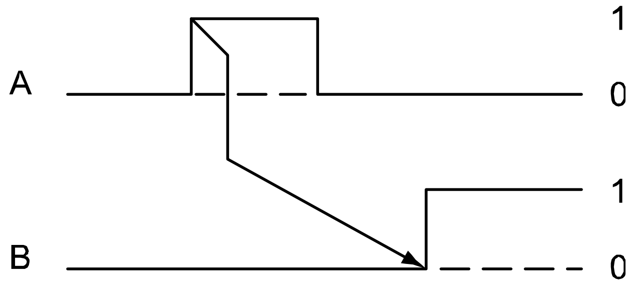
We recommend stopping the program execution before using the action *Motors Off* to secure a controlled stop.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Additional information

The signal sequence for *Motors Off* is:



xx0400000949

A: Motors Off (IN)

B: Motors Off (OUT)

3.16.4.10 Motors On

Parent

Motors On is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Motors On* sets the controller in the Motors On state.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

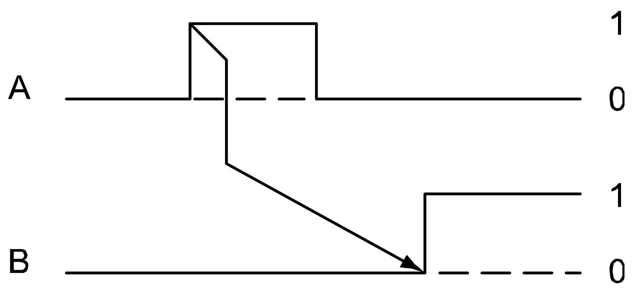
Limitations

The action value has the following limitations:

- The controller has to be in automatic mode, see *Run Chain OK on page 311*.
- If the system input I/O signal action *Motors Off* is high, then setting *Motors On* high has no effect, see *Motors Off on page 253*.
- 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)

3.16.4.11 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 *Controller*. For more information, see *Action on page 238*.

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 *Argument 1* must also be used.

, specifying continuous or cycle. The default value is continuous. For more information, see *Argument 1 on page 272*.

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies the run mode, *Continuous* or *Cycle*. See *Argument 1 on page 272*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

The action value has the following limitations:

- The controller must be in automatic mode.
- You cannot use this action value if the actions *Motors Off*, *QuickStop*, *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* are set.
- The *Motors On and Start* action is not valid during program execution.

Related information

Operating manual - OmniCore

3.16.4.12 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 *Controller*. For more information, see *Action on page 238*.

Description

The action value *PP to Main* sets the program pointer to the configured production entry that is the main routine.

Arguments

When the parameter *Action* is set to *PP to Main*, the following parameter must also be used:

Parameter	Allowed value
Argument 7	The parameter can be used to set PP to Main in a specific task. If the parameter *Argument 7* is not defined, all tasks will be affected. For more information, see *Argument 7 on page 278*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

PP to Main can only be used with normal tasks.

3.16.4.13 ProfiSafeOpAck

Parent

ProfiSafeOpAck is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

ProfiSafeOpAck is used for *PROFIsafe F-Host*. It is used to acknowledge a change in the PROFIsafe communication, for example if the communication cable was disconnected and reconnected.

The system input *ProfiSafeOpAck* has the same effect as tapping the button **F-Host Op. Ack.** on the FlexPendant.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

The action value *ProfiSafeOpAck* can only be used with the option *PROFIsafe F-host*.

Related information

Application manual - Functional safety and SafeMove.

3.16.4.14 Quick Stop

Parent

Quick Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *Controller*.

Description

The action value *Quick Stop* stops the RAPID program execution quickly. 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.

A program cannot start when this signal is high.

Note

This stop should not be used for normal program stops as this causes extra, unnecessary wear on the robot.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.15 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 *Controller*. For more information, see *Execution Error on page 295*.

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.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.16 Set Speed Override

Parent

Set Speed Override is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Set Speed Override* should be set to a specific percentage, and block any other user settings of speed override until the signal is reset.

Arguments

When the parameter *Action* is set to *Set Speed Override*, the following parameter must also be used:

Parameter	Allowed value
Argument 9	The parameter specifies the speed in percentage, see *Argument 9 on page 280*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.17 SimMode

Parent

SimMode is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *SimMode* shall be set when the simulation mode shall be entered.

Arguments

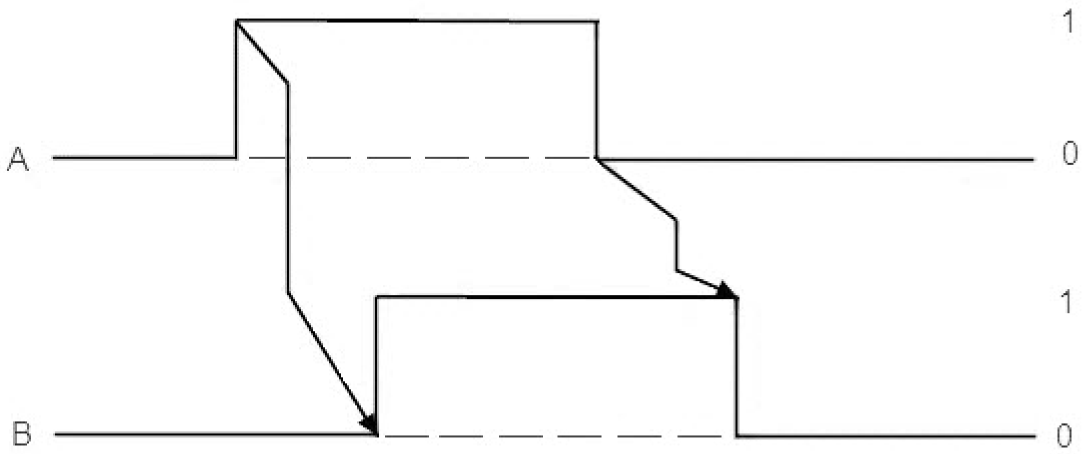
Parameter	Allowed value
Argument 1	*LOAD*, see *Argument 1 on page 272*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Signal sequence

The signal sequence for *SimMode* is:



en1100000964

A: SimMode (IN)

B: SimMode (OUT)

Additional information

A system output signal (also called *SimMode*) can be configured that reflects the status of the system state *SimMode*, see *SimMode on page 312*.

3.16.4.18 Start

Parent

Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*. For more information, see *Action on page 238*.

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 must 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 following parameter must also be used:

Parameter	Allowed value
Argument 1	*Argument 1* specifies the run mode, *Continuous* or *Cycle*. See *Argument 1 on page 272*. Default value is *Continuous*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

The controller must be in motors on state and the program control must be available, that is not used by any other resource.

Limitations

This action value has the following limitations:

- The controller must be in automatic mode.
- You cannot use this action value if the actions *QuickStop*, *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* are set.
- The *Start* action is not valid during program execution.

3.16.4.19 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 *Controller*. For more information, see *Action on page 238*.

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 following parameter must also be used:

Parameter	Allowed value
Argument 1	*Argument 1* specifies the run mode, *Continuous* or *Cycle*. See *Argument 1 on page 272*. Default value is *Continuous*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

The controller must be in motors on state and the program control must be available, that is not used by any other resource.

Limitations

This action value has the following limitations:

- The controller must be in automatic mode.
- You cannot use this action value if the actions *QuickStop*, *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* are set.
- *Start at Main* action is not valid during program execution.

3.16.4.20 Stop

Parent

Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *Controller*.

Description

The action value *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.

A program cannot start when this signal is high.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.21 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 *Controller*.

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 signal must be defined and this signal should not used by any other system input.

3.16.4.22 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 *Controller*.

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 signal must be defined and this signal should not used by any other system input.

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.

Example

If a `WaitTime` instruction is executed, it can take a while before the execution is stopped.

3.16.4.23 System Restart

Parent

System Restart is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

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

A digital input signal must be defined and this signal should not used by any other system input.

All RAPID programs should be stopped before using the action.

3.16.4.24 Trust Revolution Counter

Parent

Trust Revolution Counter is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Trust Revolution Counter* should be set when the controller reports that revolution counters must be updated, but nothing has changed and setup for the revolution counters are correct. This can happen if the SMB has been switched during power off. The signal can then be set, to trust the revolution counters.

Note

The *Trust Revolution Counter* action value is available only for IRB 365.

Arguments

Argument	Description
Argument 6	*Argument 6* defines the mechanical unit.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

3.16.4.25 Verify Local Presence

Parent

Verify Local Presence is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Verify Local Presence* is used to verify local presence when no FlexPendant is available.

Changing the value 3 times rapidly (for example 0 -> 1 -> 0 -> 1), corresponds to clicking the enabling device on the FlexPendant two times when you log in for the first time.

Note

The action value must be changed 3 times within 40 seconds.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

CAUTION

The action value *Verify Local Presence* shall not be changed remotely. In order to verify local presence, you must have visual control of the robot.

It is the responsibility of the integrator to set up this input so that local presence is always achieved.

It is only possible to configure one system input with the action value *Verify Local Presence*.

3.16.4.26 Verify Move Robot In Auto

Parent

Verify Move Robot In Auto is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

Description

The action value *Verify Move Robot In Auto* is used for verifying local presence and provides opportunity to move the robot in AUTO mode for a limited time.

The value of *Verify Move Robot In Auto* must be changed to 1 within 30 seconds after a request for moving the robot in Auto mode, to enable move in Auto.

> **CAUTION**
>
> The action value *Verify Move Robot In Auto* shall not be changed remotely. In order to verify local presence, you must have visual control of the robot. It is the responsibility of the integrator to set up this input so that local presence is always achieved.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

Limitations

Moving the robot in Auto mode is not available for all the robots.

3.16.4.27 Write Access

Parent

Write Access is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *Controller*.

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.

This system input signal can be used to obtain *single point of control*.

Prerequisites

A digital input signal must be defined and this signal should not used by any other system input.

The signal can be used only in automatic mode. The write access is released while entering the manual mode.

Additional information

The system output signal *Write Access* can reflect if the I/O client has write access or not, see *Write Access on page 321*.

3.16.5 Argument 1

Parent

Argument 1 belongs to the type *System Inputs*, in the topic *Controller*.

Description

The parameter *Argument 1* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 1*
Backup	The name of the backup. If the value is *SYSTEM:* or not defined, then the name is set to be the system name.
Interrupt	The name of the routine to be executed (without path), e.g. *HOME*. Cannot be undefined. See *The Event Routine type on page 174*.
Load	The name of the program file to load, including the file format (.modx, .sysx, .pgf) and the path, e.g. *HOME:ModuleA.modx*. Cannot be undefined.
Load and Start	The name of the program file to load, including the file format (.modx, .sysx, .pgf) and the path, e.g. *HOME:ModuleA.modx*. Cannot be undefined.
Motors On and Start Start Start at Main	*Argument 1* specifies the run mode, *Continuous* or *Cycle*. See *The Run Mode Settings type*. Default value is *Continuous*.
SimMode	*LOAD*, no other value is allowed. See *SimMode on page 261*.

Related information

Backup on page 239.

Interrupt on page 244.

Load on page 249.

Load and Start on page 251.

Motors On and Start on page 255.

Start on page 262.

Start at Main on page 263.

SimMode on page 261.

3.16.6 Argument 2

Parent

Argument 2 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 2* is an argument required to perform some of the system actions.

Limitations

Argument 2 is only used with the option *MultiMove*.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 2*
Interrupt	The task in which the routine defined in *Argument 1* should be executed. This is only used with *MultiMove*, see *The Event Routine type on page 174*. If *Argument 2* is not set, then the first found motion task is used.
Load	The task in which the program defined in *Argument 1* should be loaded, see *The Task type*.
Load and Start	The task in which the program defined in *Argument 1* should be loaded, see *The Task type*.

If *MultiMove* is not installed, then *Argument 2* must be set to T_ROB1.

Related information

Interrupt on page 244.

Load on page 249.

Load and Start on page 251.

3.16.7 Argument 3

Parent

Argument 3 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 3* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 3*
Backup	The path of the backup.

Related information

Backup on page 239.

3.16.8 Argument 4

Parent

Argument 4 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 4* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 4*
Backup	*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. The default value *Overwrite* means that a backup will overwrite an existing backup with the same name.

Related information

Backup on page 239.

3.16.9 Argument 5

Parent

Argument 5 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 5* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 5*
Backup	*AddDate* means that the date is added at the end of the name. *NoDate* means that no date is added. Default value is *NoDate*. If the parameter *Argument 4* is defined as *UniqueName*, the sequence number is added after the date.

Related information

Backup on page 239.

3.16.10 Argument 6

Parent

Argument 6 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 6* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 6*
Limit Speed	A mechanical unit from the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*.

Related information

Limit Speed on page 246.

3.16.11 Argument 7

Parent

Argument 7 belongs to the type *System Input*, in the topic *Controller*.

Description

Argument 7 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*. For more information, see *PP to Main on page 256*.

Usage

Task Name can be used to specify a RAPID task or can be left blank for all normal tasks.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 7*
PP to Main	A task from the type *Task* in the topic *Controller*, see *The Task type*.

Related information

PP to Main on page 256.

3.16.12 Argument 8

Parent

Argument 8 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 8* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 8*
Backup	*Archive* means that the backup is archived and saved as one file with the file suffix .tar. The default value *No Archive* means that the backup is not archived.

Related information

Backup on page 239.

3.16.13 Argument 9

Parent

Argument 9 belongs to the type *System Input*, in the topic *Controller*.

Description

The parameter *Argument 9* is an argument required to perform some of the system actions.

Allowed values

Value of parameter *Action*	Allowed value of parameter *Argument 9*
Set Speed Override	The speed in percentage.

Related information

Set Speed Override on page 260.

3.16.14 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]	See 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	See note *v*		X	X
ResetEstop			X	X	X		X	X
SimMode		X	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]
Verify Local Presence	X	X	X	X	X	X	X	X

i The cause of the System Failure can have impact on the function for the given System Input Actions
ii Does not affect the ongoing backup
iii Only load of the program module is performed
iv MotorOn only
v Execution error triggered during program execution
vi Ongoing backup will be deleted

3.17 Type System Output

3.17.1 The System Output type

Overview

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

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. For more information, see *The Signal type on page 393*.

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.

3.17.2 Signal Name

Parent

Signal Name belongs to the type *System Output*, in the topic *Controller*.

Description

Signal Name is the name of the configured output I/O signal to use. It connects the system output with a configured output I/O signal, see *The Signal type on page 393*.

Allowed values

A name of an already configured output I/O signal.

Prerequisites

The signal must be digital for all outputs excepts *TCP Speed*, *TCP Speed Reference* and *Speed Override* that uses analog signals.

3.17.3 Status

Parent

Status belongs to the type *System Output*, in the topic *Controller*.

Description

The parameter *Status* defines what state the output signal will reflect.

Allowed values

The following values are allowed and are described on the following pages:

- *Absolute Accuracy Active on page 286*.
- *Auto On on page 287*.
- *Backup Error on page 288*.
- *Backup in progress on page 289*.
- *Collision Avoidance on page 290*.
- *Control On State on page 291*.
- *Cycle On on page 292*.
- *Emergency Stop on page 293*.
- *Energy Saving Blocked on page 294*.
- *Execution Error on page 295*.
- *Limit Speed on page 296*.
- *Mechanical Unit Active on page 297*.
- *Mechanical Unit Not Moving on page 298*.
- *Motion Supervision On on page 299*.
- *Motion Supervision Triggered on page 300*.
- *Motors Off on page 301*.
- *Motors Off State on page 302*.
- *Motors On on page 303*.
- *Motors On State on page 304*.
- *Path Return Region Error on page 305*.
- *Power Fail Error on page 306*.
- *PP Moved on page 307*.
- *Production Execution Error on page 308*.
- *Revolution Counter Lost on page 309*.
- *Robot In Trusted Position on page 310*.
- *Run Chain OK on page 311*.
- *SimMode on page 312*.
- *Simulated I/O on page 313*.
- *SMB Battery Charge Low on page 314*.
- *Speed Override on page 315*.

Continues on next page

- *Stop from client on page 316*.
- *System Input Busy on page 317*.
- *TaskExecuting on page 318*.
- *TCP Speed on page 319*.
- *TCP Speed Reference on page 320*.
- *Write Access on page 321*.

3.17.4 Values for the parameter *Status*

3.17.4.1 Absolute Accuracy Active

Parent

Absolute Accuracy Active is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

3.17.4.2 Auto On

Parent

Auto On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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 - OmniCore.

3.17.4.3 Backup Error

Parent

Backup Error is a value for the parameter *Status* and belongs to the type *System Output*, in the topic *Controller*. For more information, see *Backup on page 239*.

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*.

3.17.4.4 Backup in progress

Parent

Backup in progress is a value for the parameter *Status* and belongs to the type *System Output*, in the topic *Controller*. For more information, see *Backup on page 239*.

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*.

3.17.4.5 Collision Avoidance

Parent

Collision Avoidance is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

The I/O signal is set when the functionality for *Collision Avoidance* is activated. The signal is cleared when the functionality for *Collision Avoidance* is deactivated.

The function *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* determines at what distance the two objects are considered to be in collision, see *Coll-Pred Safety Distance on page 705*.

3.17.4.6 Control On State

Parent

Control On State is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Control On State*, the signal is set when the controller is in the state Motors On and the brakes are released. The signal is reset when the brakes are applied.

> **Note**
>
> The brakes are applied in the state Motors Off. The brakes can also be applied in the state Motors On, for example, at Brake On Timeout.

3.17.4.7 Cycle On

Parent

Cycle On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

3.17.4.8 Emergency Stop

Parent

Emergency Stop is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Emergency Stop*, the I/O signal is set when the controller is in emergency stop state.

3.17.4.9 Energy Saving Blocked

Parent

Energy Saving Blocked is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Energy Saving Blocked*, the I/O signal is set when the energy saving functionality is blocked (disabled).

Additional information

It is not only the system input signal *Enable Energy Saving* that can cause the energy saving functionality to be blocked.

3.17.4.10 Execution Error

Parent

Execution Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

The I/O signal is set to high until one 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). For more information, see *Reset Execution Error Signal on page 259*.

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).

Arguments

When the parameter *Status* is set to *Execution Error*, the following parameters must also be used.

Parameter	Allowed value
Argument 2	If *Argument 2* is specified with a task name, the I/O signal will only react on execution errors for that task. For more information, see *Argument 2 on page 323*.

3.17.4.11 Limit Speed

Parent

Limit Speed is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *LimitSpeed*, the I/O signal is set when the specified mechanical unit is running with reduced speed triggered by the system input signal *Limit Speed*.

See description for *Limit Speed on page 246*.

Arguments

When the parameter *Status* is set to *Limit Speed*, the following parameters must also be used.

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies which mechanical unit the signal is used for, see *Argument 1 on page 322*.
Argument 4	*Argument 4* specifies a delay when setting the output to minimize the risk of faulty triggering by SafeMove when the output is used to start the supervision. *Argument 4 on page 325*.

Related information

System input *Limit Speed on page 246*.

3.17.4.12 Mechanical Unit Active

Parent

Mechanical Unit Active is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Mechanical Unit Active*, the I/O signal is set when the configured mechanical unit is active.

Arguments

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies which mechanical unit the signal is used for, see *Argument 1 on page 322*.

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.

3.17.4.13 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 *Controller*.

Description

If *Status* has the value *Mechanical Unit Not Moving*, the I/O signal is set high when the configured mechanical unit is not moving. 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. For more information, see *Mech.Unit Not Moving Detection Level on page 766*, in the topic *Motion*, type *Robot* and *Mech.Unit Not Moving Detection Level on page 824*, in the topic *Motion*, type *Single*.

Arguments

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies which mechanical unit the signal is used for, see *Argument 1 on page 322*. If *Argument 1* 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.

Note

The drop-down list in the FlexPendant or RobotStudio configuration tool shows only TCP robots. Use ABC... to add any other mechanical unit.

Limitations

For conveyors and mechanical units that are moved using independent move or sensor synchronization the system output remains high if the robot is not moving.

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 option *SafeMove* can be used.

3.17.4.14 Motion Supervision On

Parent

Motion Supervision On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Motion Supervision On*, the I/O signal is set when the motion supervision function is active.

Arguments

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies which mechanical unit the supervision is used for, see *Argument 1 on page 322*.

Additional information

Motion Supervision On is only valid when the robot is in status motors on.

After motion supervision has triggered, the robot moves away from the collision, see *Technical reference manual - RAPID Overview*, section *Collision detection*. To make sure that it is possible to back away, the motion supervision is always turned off during this movement, which means that the I/O signal will be set to low directly after motion supervision is triggered. The I/O signal will then be reset to the previous value after one of the following actions:

- The program is restarted.
- The program pointer is manually moved to Main.

Related information

Application manual - Controller software OmniCore

3.17.4.15 Motion Supervision Triggered

Parent

Motion Supervision Triggered is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

Arguments

Parameter:	Allowed value:
Argument 1	*Argument 1* specifies which mechanical unit the supervision is used for, see *Argument 1 on page 322*.

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. For more information, see *CollisionErrorHandling on page 192*.

Related information

Application manual - Controller software OmniCore

3.17.4.16 Motors Off

Parent

Motors Off is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Motors Off*, the I/O signal is set when the controller is in motors off state.

Additional information

If the controller is in guard stop[1], 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.

If only motors off state is of interest, the action value *Motors Off State* is preferred.

Related information

Motors Off State on page 302.

Run Chain OK on page 311.

1 The controller is in motors off state and a safety chain is not closed.

3.17.4.17 Motors Off State

Parent

Motors Off State is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Motors Off State*, the I/O signal is set when the controller is in motors off state.

3.17.4.18 Motors On

Parent

Motors On is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Motors On*, the I/O signal is set when the controller is in motors on state.

For more information, see *Motors On State on page 304*.

Additional information

If the controller is in guard stop[2], 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.

2 The controller is in motors off state and a safety chain is not closed.

3.17.4.19 Motors On State

Parent

Motors On State is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Motors On State*, the I/O signal is set when the controller is in motors on state.

3.17.4.20 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 *Controller*.

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.

Arguments

Parameter	Allowed value
Argument 1	*Argument 1* specifies which mechanical unit the supervision is used for, see *Argument 1 on page 322*.

Additional information

The value *Path Return Region Error* is set, for example, 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*, see *The Path Return Region type on page 227*.

3.17.4.21 Power Fail Error

Parent

Power Fail Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

3.17.4.22 PP Moved

Parent

PP Moved is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

The signal is set when PP is moved. Argument 3 decides what kind of PP movement that will trigger the signal and how the signal is set.

If Argument 3 is set to MAIN, the signal will be set when PP is moved to main. The signal will stay high until the *Cycle On* parameter is set and the execution starts.

If Argument 3 is not set, the signal will pulse once when you move the program pointer to Main, routine, or within a routine.

Arguments

Argument	Description
Argument 2	*Argument 2* defines a RAPID task. For more information, see *Argument 2 on page 323*.
Argument 3	*Argument 3* defines if the the output shall reflect only PP moved to main, and will then be set instead of pulsed. For more information, see *Argument 3 on page 324*.

3.17.4.23 Production Execution Error

Parent

Production Execution Error is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*. For more information, see *Execution Error on page 295*.

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[3]
- A system error: SysFail, SysHalt, or SysStop RapidBlock.

The I/O signal is reset by:

- Program start.
- Program restart.

The I/O signal value is not kept after a restart.

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

Motion Supervision Triggered on page 300

System errors are described in parameter *TrustLevel on page 336*.

The instruction `SystemStopAction`, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

3 This is *not* a replacement for *Motion Supervision Triggered*.

3.17.4.24 Revolution Counter Lost

Parent

Revolution Counter Lost is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

The signal will be set when revolution counters must be updated.

Arguments

Argument	Description
Argument 1	*Argument 1* defines the mechanical unit.

3.17.4.25 Robot In Trusted Position

Parent

Robot In Trusted Position is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Robot In Trusted Position*, the I/O signal is set when the robot is on the programmed path.

Arguments

Parameter	Allowed value
Argument 2	*Argument 2* specifies which RAPID task controls the mechanical unit that the signal is used for, see *Argument 2 on page 323*.
Argument 5	*Argument 5* defines the maximum TCP distance that the robot can diverge from the programmed path, see *Argument 5 on page 326*.
Argument 6	*Argument 6* defines the maximum TCP rotation, see *Argument 6 on page 327*.
Argument 7	*Argument 7* defines the maximum TCP distance that the external axis can diverge from the programmed path, see *Argument 7 on page 328*.
Argument 8	*Argument 8* defines the maximum TCP rotation that the external axis can diverge from the programmed path, see *Argument 8 on page 329*.

Additional information

The value *Robot In Trusted Position* is reset if:

- The robot is jogged too far from the programed path.
- The robot has slid too far away from its programmed path, for example, caused by an unplanned stop.
- The program pointer is moved to either Main, Routine, or Cursor.
- A position in the program is modified or the program is edited so that the program pointer is lost.

The I/O signal is set by one of the following actions:

- When a fine point is reached.
- If the first position is a zone, the signal is set when leaving the zone.

If the signal is set and the controller is restarted, then the signal will not be set until the controller is in Motors On state.

Note

Robot In Trusted Position will not be updated correctly in case the robot has deviated from the programmed path due to corrections such as conveyor tracking, soft servo, etc.

3.17.4.26 Run Chain OK

Parent

Run Chain OK is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

Example

In manual mode the safety chain is opened and *Run Chain OK* is not set.

3.17.4.27 SimMode

Parent

SimMode is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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

Parameter	Allowed value
Argument 3	*LOAD*, see *Argument 3 on page 324*.

Additional information

After a restart, the system output signal *SimMode* will also reflect the state *SimMode*.

3.17.4.28 Simulated I/O

Parent

Simulated I/O is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

I/O signals can be set to simulated mode during testing, using the FlexPendant.

Related information

Operating manual - OmniCore

3.17.4.29 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 *Controller*.

Description

If *Status* has the value *SMB Battery Charge Low*, the I/O signal is set when the SMB battery capacity is running low and the battery needs to be replaced soon. The signal is cleared when the SMB battery charge is okay.

There is only one battery in a normal single robot system. However, there can be up to 16 SMB batteries in a MultiMove system or when using external axes. The output is activated if any of the batteries need replacement.

An event log message gives information about which battery should be replaced.

Additional information

SMB batteries are cyclically supervised every 10th hour. After replacing a battery, it can take up to 10 hours for the signal to reset. The value is saved during a restart.

3.17.4.30 Speed Override

Parent

Speed Override is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Speed Override*, the I/O signal reflects the speed override in percent.

Prerequisites

An analogue output signal with the signal name defined as *Speed Override* must be available, and it cannot be used by any other resource.

3.17.4.31 Stop from client

Parent

Stop From Client is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *Stop From Client*, the I/O signal will pulse for one second when a stop is done by an external client, such as the TPU, RobotStudio, and so on. A stop made from a PLC with a system input signal Stop, an execution error, an emergency stop or any other action not taken by a client will not affect the system output signal.

3.17.4.32 System Input Busy

Parent

System Input Busy is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

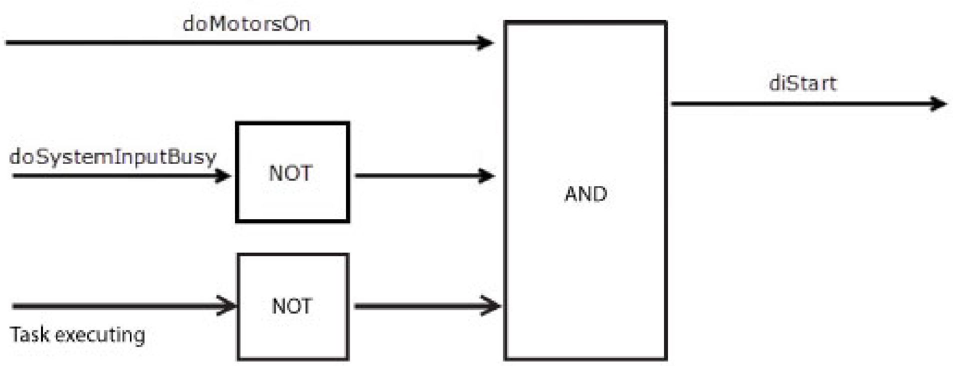
If *Status* has the value *System Input Busy*, that means the I/O signal is set when the system input mechanism is busy.

For some actions the controller is busy for some time and cannot receive any commands, thus rejects any order. A new command must be sent when the controller is ready again. The status *System Input Busy* can be used to show if the incoming system input request will be rejected or not.

Example

In this example the controller is set to motors on and a program is started by setting a system output status *Motors On*. This signal is cross connected to a system input signal configured with the action *Start*. As the controller is busy with changing state to motors on, the start order will be rejected while the controller is still busy with the state change.

A solution to this is to use the system input action *System Input Busy* and add an AND operator with *System Input Busy* inverted to the cross connection. This will delay the start request until the motors on action is completed. Since the start order also will make the system busy, that is, *doSystemInputBusy* will be set again, an extra inverted system output signal for *Task Execution* must be added so the logic does not end up in a loop.



xx1600001282

3.17.4.33 TaskExecuting

Parent

TaskExecuting is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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.

Arguments

When the parameter *Status* is set to *TaskExecuting*, the following parameters must also be used.

Parameter	Allowed value
Argument 2	The parameter has to be defined with a task name. For more information, see *Argument 2 on page 323*. The parameter *Argument 2* can only be configured with the name of a normal task.

3.17.4.34 TCP Speed

Parent

TCP Speed is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *TCP Speed*, the I/O signal reflects the speed of the robot's TCP.

Arguments

Parameter	Allowed value
Argument 1	*Argument 1* specifies which mechanical unit the speed refers to, see *Argument 1 on page 322*.

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. For more information, see *Maximum Logical Value on page 412* and *Maximum Physical Value on page 413*.

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 *Event Preset Time* parameter affects the time interval between the setting up of the analog output and the occurrence 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 occurrence of the TCP speed. For more information, see *Event Preset Time on page 648*.

This system output should not be used together with the system input *Limit Speed*. The system output will not reflect the change in TCP speed that is activated when the *Limit Speed* signal is set.

3.17.4.35 TCP Speed Reference

Parent

TCP Speed Reference is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

Description

If *Status* has the value *TCP Speed Reference*, the I/O signal reflects the programmed speed of the robot's TCP.

Arguments

Parameter	Allowed value
Argument 1	*Argument 1* specifies which mechanical unit the programmed speed refers to, see *Argument 1 on page 322*.

Additional information

TCP Speed Reference works in the same way as *TCP Speed* but uses the programmed speed. For more information, see *TCP Speed on page 319*.

Note

TCP Speed can differ from *TCP Speed Reference*, for example at acceleration or if the override speed has been changed.

This system output should not be used together with the system input *Limit Speed*. The system output will not reflect the change in TCP speed that is activated when the *Limit Speed* signal is set.

3.17.4.36 Write Access

Parent

Write Access is a value for the parameter *Status* that belongs to the type *System Output* in the topic *Controller*.

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 input *Write Access*, see *Write Access on page 271*.

3.17.5 Argument 1

Parent

Argument 1 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 1* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 1*
Motion Supervision On *TCP Speed* *TCP Speed Reference*	A mechanical unit of the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*. Default value is *ROB_1*.
Motion Supervision Triggered *Path Return Region Error*	A mechanical unit of the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*. If no mechanical unit is specified, the I/O signal reacts on any mechanical unit in the system.
Mechanical Unit Active	A mechanical unit of the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*. Default value is *ROB_1*.
Mechanical Unit Not Moving	A mechanical unit of the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*. If no unit is specified, the I/O signal reacts on any mechanical unit in the system. Default value is empty.
Limit Speed	A mechanical unit of the type *Mechanical Unit* in the topic *Motion*, see *The Mechanical Unit type on page 619*. The value cannot be empty.

Related information

Motion Supervision On on page 299.

TCP Speed on page 319.

TCP Speed Reference on page 320.

Motion Supervision Triggered on page 300.

Path Return Region Error on page 305.

Mechanical Unit Active on page 297.

Limit Speed on page 296.

Mechanical Unit Not Moving on page 298.

3.17.6 Argument 2

Parent

Argument 2 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 2* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	**Allowed value of parameter *Argument 2***
Execution Error *PP Moved*	A task from the type *Task* in the topic *Controller*, see *The Task type*. If no task is specified, the I/O signal reacts on any task in the system.
TaskExecuting	A task from the type *Task* in the topic *Controller*, see *The Task type*.
Robot In Trusted Position	A task from the type *Task* in the topic *Controller*, see *The Task type*.

Related information

Execution Error on page 295.

PP Moved on page 307.

TaskExecuting on page 318.

Robot In Trusted Position on page 310.

3.17.7 Argument 3

Parent

Argument 3 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 3* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 3*
SimMode	*LOAD*, see SimMode on page 312.
PP Moved	*MAIN*, see PP Moved on page 307.

3.17.8 Argument 4

Parent

Argument 4 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 4* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 4*
LimitSpeed	*Argument 4* specifies a delay when setting the output to minimize the risk of faulty triggering by SafeMove when the output is used to start the supervision. The default value is 250 ms.

Related information

Limit Speed on page 296.

3.17.9 Argument 5

Parent

Argument 5 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 5* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 5*
Robot In Trusted Position	*Argument 5* defines the maximum TCP distance that the robot can diverge from the programmed path. Allowed values are 0-2.00, specified in meters. Default value is 0.05 meters.

Related information

Robot In Trusted Position on page 310.

3.17.10 Argument 6

Parent

Argument 6 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 6* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 6*
Robot In Trusted Position	*Argument 6* defines the maximum TCP rotation. Allowed values are 0-6.280, specified in radians. Default value is 1.57 radians.

Related information

Robot In Trusted Position on page 310.

3.17.11 Argument 7

Parent

Argument 7 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 7* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 7*
Robot In Trusted Position	*Argument 7* defines the maximum TCP distance that the external axis can diverge from the programmed path. Allowed values are 0-2.00, specified in meters. Default value is 0.05 meters.

Related information

Robot In Trusted Position on page 310.

3.17.12 Argument 8

Parent

Argument 8 belongs to the type *System Outputs*, in the topic *Controller*.

Description

The parameter *Argument 8* is an argument required for some of the system output status values.

Allowed values

Value of parameter *Status*	Allowed value of parameter *Argument 8*
Robot In Trusted Position	*Argument 8* defines the maximum TCP rotation that the external axis can diverge from the programmed path. Allowed values are 0-6.280, specified in radians. Default value is 1.57 radians.

Related information

Robot In Trusted Position on page 310.

3.18 Type Task

3.18.1 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.

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 OmniCore chapter *Multitasking*.

3.18.2 Task

Parent

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

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.

3.18.3 Task in Foreground

Parent

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

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, it runs at the highest priority. That is, no other task can suspend its execution.

Limitations

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

A motion task cannot have another task in the foreground. Any such eventual task set will be ignored.

Allowed values

A string with maximum 30 characters.

3.18.4 Type

Parent

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

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. 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.

3.18.5 Check Unresolved References

Parent

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

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.

3.18.6 Main Entry

Parent

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

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.

3.18.7 TrustLevel

Parent

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

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 - OmniCore

3.18.8 Use Mechanical Unit Group

Parent

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

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 (for example, the TCP position) for the mechanical units in the mechanical unit group. For more information, see *MotionTask on page 338*.

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*. For more information, see *Name on page 206*.

A string with maximum 32 characters.

3.18.9 MotionTask

Parent

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

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 OmniCore.

3.18.10 Hidden

Parent

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

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

Only tasks with *Type* defined as *Static* or *Semistatic* can be hidden. A *Normal* task cannot be hidden.

This parameter is only available for multitasking systems.

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

Allowed values

YES or NO.

Default value is NO.

3.18.11 RMQ Type

Parent

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

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 OmniCore*.

3.18.12 RMQ Max Message Size

Parent

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

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.

> **Note**
>
> The value cannot be changed in RobotStudio. The only way to change the value is to edit the sys.cfg file by adding the attribute *RmqMaxMsgSize* with the desired value.

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 OmniCore*.

3.18.13 RMQ Max No Of Messages

Parent

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

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.

Note

The value cannot be changed in RobotStudio. The only way to change the value is to edit the sys.cfg file by adding the attribute *RmqMaxNoOfMsg* with the desired value.

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 OmniCore*.

3.18.14 RMQ Mode

Parent

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

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 OmniCore*.

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

3.18.15 Exclude From Load Modules In AllTasks

Parent

Exclude From Load Modules In AllTasks belongs to the type *Task*, in the topic *Controller*.

Description

For the *Automatic Loading Of Modules* type, RAPID modules can be configured to load AllTask. With this argument it is possible to suppress that for a certain task. If the value of *Exclude From Load Modules In AllTasks* is set to YES, modules that are set to be loaded AllTask will not be loaded into this task.

Allowed values

YES or NO.

Default value is NO.

Related information

Type Automatic Loading of Modules on page 159

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 types and parameters that are specific for the industrial networks are described in the respective application manual.

Configuration results

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

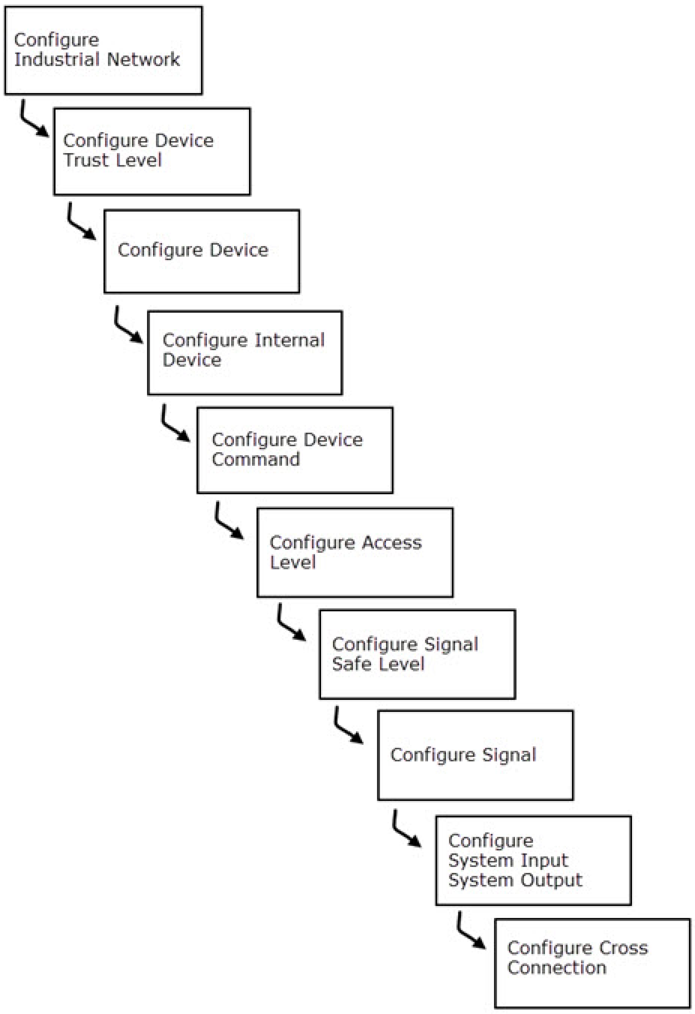
4.2 Workflows

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. For more information, see *The Device type on page 359*.

The following are examples of available I/O devices:

- Digital I/O
- Analog I/O
- AD Combi I/O
- Relay I/O
- Gateways
- Simulated I/O
- Encoder interface devices

Prerequisites

Before defining an I/O device, you must:

1. Configure 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. For more information, see *The Device Trust Level type on page 375*.

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.

4.2.3 How to define 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

For more information, see *The Signal type on page 393*.

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. For more information, see *How to define an I/O signal group on page 350*.

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. For more information, see *The Signal Safe Level type on page 421*.

How to define I/O signals

To define I/O signals:

1. In I/O Engineering Tool, select the I/O device or I/O module to which signals should be assigned.

Continues on next page

2. On the row below the existing signals, add a new signal by completing the empty field **Name**. Assign properties to the new signal.
3. Save the changes.
4. Restart the controller.

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 I/O Engineering Tool, select the I/O device or I/O module to which the group signal should be assigned.
2. Add a new I/O signal, 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.

For more information, see *How to define I/O signals on page 348*, *The Signal type on page 393*, and *The Signal Safe Level type on page 421*.

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.

4.3 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.

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	Signals that are installed with access level internal cannot be viewed or accessed from user applications. 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
Rapid	Write enabled
Local client in manual mode	Write enabled
Local client in auto mode	Read only

Continues on next page

Parameter:	Value:
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*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.3.3 Rapid

Parent

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

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*.

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*.

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*.

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*.

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 Device

4.4.1 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.

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 *The Internal Device type on page 384* and *Connected to Industrial Network on page 362*.

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.

For more information, see *The Device Command type on page 368*.

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).

For more information, see *The Device Trust Level type on page 375*.

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).

Related information

For more information on safety signals, see *Operating manual - OmniCore*.

Continues on next page

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

4.4.2 Name

Parent

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

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.4.3 Connected to Industrial Network

Parent

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

Description

The parameter *Connected to Industrial Network* specifies which industrial network this I/O device is physically connected to.

4.4.4 Identification Label

Parent

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

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.

4.4.5 Trust Level

Parent

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

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.4.6 State when System Startup

Parent

State when System Startup belongs to the type *Device*, in the topic *I/O System*.

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

4.4.7 Simulated

Parent

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

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

4.4.8 Fast Device Startup

Parent

Fast Device Startup belongs to the type *Device*, in the topic *I/O System*.

Description

The parameter *Fast Device Startup* specifies if the I/O device should use a faster connection attempt algorithm or not.

Usage

The parameter *Fast Device Startup* is used mainly to speed up tool change applications. The usual PROFINET connection attempt takes a few seconds to complete, but with Fast Device Startup enabled devices, this time is shortened to less than a second. For more information, see *Application manual - PROFINET Controller/Device*.

Prerequisites

The option *3020-1 PROFINET Controller* and *3020-2 PROFINET Device* must be installed.

Limitations

The Ethernet switches between the controller and the I/O device that uses the *Fast Device Startup* functionality. It must be configured to disable the auto crossover and automatic speed detection functions on used connectors. The speed rate is set to 100Mbps (full duplex).

Default value

The default value is *Deactivated*.

Allowed values

- *Deactivated*
- *Activated*
- *Support*

Note

Select *Support* to set the desired port speed. For port speed, select *100 Mbps* and the port speed is adjusted to *100 Mbps*, so autonegotiation is turned off for the port.

Hence, it is possible to change the settings on a built-in switch for a PROFINET I/O device.

4.5 Type Device Command

4.5.1 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.

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

4.5.2 Name

Parent

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

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.

4.5.3 Device

Parent

Device belongs to the type *Device Command*, in the topic *I/O System*.

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

The Device type on page 359.

4.5.4 Download Order

Parent

Download Order belongs to the type *Device Command*, in the topic *I/O System*.

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.

4.5.5 Path

Parent

Path belongs to the type *Device Command*, in the topic *I/O System*.

Description

Path specifies the network path to the parameter.

Allowed values

A string defining the path with maximum 30 characters.

Example

```
6,20 01 24 08 30 01,C6,1
```

Description of example:

- 6 is the length of the path - that is, the number of hexadecimal figures until the next comma.
- Path (20 01 24 08 30 01) is a software description of DeviceNet class, instance and attribute. A further description can be found in the ODVA DeviceNet Specification 2.0.
- C6 is the hexadecimal value for the data type identifier.
- 1 is the data size - that is, the number of bytes as a hexadecimal value.

4.5.6 Service

Parent

Service belongs to the type *Device Command*, in the topic *I/O System*.

Description

Service defines the explicit service that should be performed on DeviceNet or EtherNet/IP object instance or attribute pointed out in *Path*. For more information, see *Path on page 372*.

Usage

Service is used to define the type of action to be used.

Prerequisites

The option *DeviceNet Master/Slave* or *EtherNet/IP* must be installed

Default value

The default value is *Set_Attribute_Single*.

Allowed values

Following values are allowed:

- *Reset*
- *Create*
- *Apply_Attributes*
- *Set_Attribute_Single*

4.5.7 Value

Parent

Value belongs to the type *Device Command*, in the topic *I/O System*.

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.

4.6 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.

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 and event generation for I/O devices.

Limitations

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

Predefined device trust levels

Device Trust Level:	Description:
DefaultTrustLevel	Default for an I/O device if nothing else is defined. 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.
InternalDeviceTrustLevel	Default for an internal I/O device if nothing else is defined. Using this level - • there is no system action performed when the I/O device is disconnected. • there is no event reported, when the I/O device is reconnected. • I/O devices with this trust level are not allowed to be deactivated. It is always set to *Deny Deactivate*. **Note** *InternalDeviceTrustLevel* shall only be used for internal devices which cannot be disconnected.
SafetyTrustLevel	Default for a safety I/O device if nothing else is defined. 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*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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*.

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*.

Description

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

Default value

Default value is *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*.

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

Technical reference manual - Event logs for RobotWare 7

4.6.6 Action when Faulty

Parent

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

Description

Specifies the system action to perform when the signals are not accessible and I/O device is changed to a bad state.

Default value

Default value is *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.

4.6.7 Report when Faulty

Parent

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

Description

Specifies the event reporting when an I/O device is changed to bad state.

Default value

Default value is *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.

4.6.8 Report when Reconnected

Parent

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

Description

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

Default value

Default value is *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.

4.6.9 Energy Saving Active

Parent

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

Description

Specifies if the I/O device shall be selected for energy saving or not.

Default value

Default value is No.

Allowed values

Yes or *No*

4.7 Type Internal Device

4.7.1 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.

Type description

For the internal slave device and the anybus industrial network options, a predefined *Internal Device* is created at system startup. For more information, see *The Device type on page 359*.

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

4.7.2 Vendor Name

Parent

Vendor Name belongs to the type *Internal Device*, in the topic *I/O System*.

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.

4.7.3 Product Name

Parent

Product Name belongs to the type *Internal Device*, in the topic *I/O System*.

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.

4.7.4 Identification Label

Parent

Identification Label belongs to the type *Internal Device*, in the topic *I/O System*.

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.

4.7.5 Simulated

Parent

Simulated belongs to the type *Internal Device*, in the topic *I/O System*.

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

4.7.6 Connection Input Size

Parent

Connection Input Size belongs to the type *Internal Device*, in the topic *I/O System*.

Description

Connection Input Size defines the data size in bytes for the input area received from the connected EtherNet/IP scanner.

Usage

Connection Input Size is an EtherNet/IP specific parameter.

Prerequisites

The option *EtherNet/IP Scanner/Adapter*, *Ethernet/IP Anybus Adapter*, *DeviceNet Master/Slave* or *DeviceNet Anybus Slave* must be installed.

Default value

For option *DeviceNet Master/Slave*, default value is 8.

For option *Ethernet/IP Anybus Adapter*, default value is 64

Allowed values

For option *Ethernet/IP Anybus Adapter*, allowed values are the integers 0-255 (0-2040 signal bits), specifying the data size in bytes

For option *DeviceNet Master/Slave*, allowed values ranges from -1 to 64.

For option *EtherNet/IP Scanner/Adapter*, allowed values ranges from 0 to 505.

4.7.7 Connection Output Size

Parent

Connection Output Size belongs to the type *Internal Device*, in the topic *I/O System*.

Description

Connection Output Size defines the data size in bytes for the output area sent to the connected EtherNet/IP scanner

Usage

Connection Output Size is an EtherNet/IP specific parameter.

Prerequisites

The option *EtherNet/IP Scanner/Adapter*, *EtherNet/IP Anybus Adapter*, *DeviceNet Master/Slave* or *DeviceNet Anybus Slave* must be installed.

Default value

For option *DeviceNet Master/Slave*, default value is 8.

For option *EtherNet/IP Anybus Adapter*, default value is 64

Allowed values

For option *EtherNet/IP Anybus Adapter*, allowed values are the integers 0-255 (0-2040 signal bits), specifying the data size in bytes

For option *DeviceNet Master/Slave* and *DeviceNet Anybus Slave* allowed values ranges from -1 to 64.

For option *EtherNet/IP Scanner/Adapter*, allowed values ranges from 0 to 505.

4.7.8 Input Size

Parent

Input Size belongs to the type *Internal Device*, in the topic *I/O System*.

Description

The parameter *Input Size* is used to configure the input slot configuration of the PROFINET internal device.

Usage

It will configure the input slot size for the PROFINET internal device. This size must match the connecting PLC’s or other PROFINET controller’s defined output slot size. .

Prerequisites

The option *PROFINET Controller/Device* or *PROFINET Device* must be installed.

Default value

The default value is 64 bytes (512 signal bits).

Allowed values

8, 16, 32, 64, 128 or 256 Bytes (64, 128, 512, 1024 or 2048 signal bits).

4.7.9 Output Size

Parent

Output Size belongs to the type *Internal Device*, in the topic *I/O System*.

Description

The parameter *Output Size* is used to configure the output slot configuration of the PROFINET internal device.

Usage

The parameter *Output Size* is only valid for the PN_Internal_Device. It will configure the output slot size for the PROFINET internal device. This size must match the connecting PLC's or other PROFINET controller's defined input slot size. .

Prerequisites

The option *PROFINET Controller/Device* or *PROFINET Device* must be installed.

Default value

The default value is 64 bytes (512 signal bits).

Allowed values

8, 16, 32, 64, 128 or 256 bytes (64, 128, 512, 1024 or 2048 signal bits).

4.8 Type Signal

4.8.1 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.

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).

For more information, see *The Device type on page 359*.

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). For more information, see *The Access Level type on page 351* and *The Signal Safe Level type on page 421*.

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
Device Mapping	11

Continues on next page

Parameter	Value
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
Device Mapping	0-7

Continues on next page

Parameter	Value
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
Maximum Physical Value Limit	10

Continues on next page

Parameter	Value
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

Operating manual - OmniCore

4.8.2 Name

Parent

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

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, for more information, see *Application manual - I/O Engineering*.
- Configuring system inputs and system outputs, for more information, see *The System Input type on page 236* and *The System Output type on page 282*.

Default value

The default value is an empty string.

Allowed values

A string with maximum 32 characters.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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 Type of Signal

Parent

Type of Signal belongs to the type *Signal*, in the topic *I/O System*.

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

4.8.4 Assigned to Device

Parent

Assigned to Device belongs to the type *Signal*, in the topic *I/O System*.

Description

The parameter *Assigned to Device* specifies which I/O device the I/O signal is connected to (if any). For more information, see *The Device type on page 359*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.8.5 Signal Identification Label

Parent

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

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

4.8.6 Device Mapping

Parent

Device Mapping belongs to the type *Signal*, in the topic *I/O System*.

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
- Group Input/Digital Input
- Group Output/Group Output

Continues on next page

- 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. For more information, see *Number Of Bits on page 420*. 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*.

Note

For safety group signals, the *Device Mapping* will automatically be rearranged when *Number Of Bits* is selected. The signal with the highest size will be placed first, and the signal with the lowest size placed last.

Continues on next page

Example

Examples of valid mapping of a digital signal (1 bit):

- 0
- 13

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

4.8.7 Category

Parent

Category belongs to the type *Signal*, in the topic *I/O System*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.8.8 Access Level

Parent

Access Level belongs to the type *Signal*, in the topic *I/O System*. For more information, see *The Access Level type on page 351*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.8.9 Default Value

Parent

The parameter *Default Value* belongs to the type *Signal*, in the topic *I/O System*.

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 *The Signal Safe Level type on page 421*.
- 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.

Limitations

It is not possible to set digital input values on a real I/O device unless it is *Simulated*. Even the *Default Value* set for a digital input in the configuration file will not get reflected after loading it to the controller.

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*. For more information, see *Minimum Logical Value on page 416* and *Maximum Logical Value on page 412*.
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). For more information, see *Device Mapping on page 401*.

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 401*.

4.8.10 Safe Level

Parent

Safe Level belongs to the type *Signal*, in the topic *I/O System*. For more information, see *The Signal Safe Level type on page 421*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.8.11 Filter Time Passive

Parent

Filter Time Passive belongs to the type *Signal*, in the topic *I/O System*. For more information, see *Filter Time Active on page 409* and *Type of Signal on page 398*.

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*.

4.8.12 Filter Time Active

Parent

Filter Time Active belongs to the type *Signal*, in the topic *I/O System*. For more information, see *Filter Time Active on page 409* and *Type of Signal on page 398*.

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*.

4.8.13 Invert Physical Value

Parent

Invert Physical Value belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

Default value

The default value is No.

Allowed values

Yes

No

4.8.14 Analog Encoding Type

Parent

Analog Encoding Type belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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.

4.8.15 Maximum Logical Value

Parent

Maximum Logical Value belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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 (For more information, see *Minimum Logical Value on page 416* and *Minimum Physical Value on page 417*.)
- maximum logical value = maximum physical value (For more information, see *Maximum Logical Value on page 412* and *Maximum Physical Value on page 413*.)

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).

4.8.16 Maximum Physical Value

Parent

Maximum Physical Value belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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, (for more information, see *Minimum Physical Value on page 417* and *Minimum Bit Value on page 419*.)
- maximum physical value = maximum bit value, (for more information, see *Maximum Physical Value on page 413* and *Maximum Bit Value on page 415*.)

Hence there is no scaling or offset factor between the physical and bit representation of the value of an I/O signal.

4.8.17 Maximum Physical Value Limit

Parent

Maximum Physical Value Limit belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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, (for more information, see *Minimum Physical Value on page 417*.)
- maximum physical value limit = maximum physical value, (for more information, see *Maximum Physical Value on page 413*.)

4.8.18 Maximum Bit Value

Parent

Maximum Bit Value belongs to the type *Signal*, in the topic *I/O System*.

Description

The parameter *Maximum Bit Value* specifies the bit value that will correspond to the *Maximum Logical Value*. For more information, see *Maximum Logical Value on page 412*.

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. For more information, see *Maximum Physical Value on page 413*.

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. For more information, see *Type of Signal on page 398*.

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*. For more information, see *Minimum Bit Value on page 419* and *Analog Encoding Type on page 411*.

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

4.8.19 Minimum Logical Value

Parent

Minimum Logical Value belongs to the type *Signal*, in the topic *I/O System*.

Description

The parameter *Minimum Logical Value* specifies the logical value that will correspond to the *Minimum Physical Value*. For more information, see *Minimum Physical Value on page 417*.

Usage

See *Maximum Logical Value on page 412*.

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. For more information, see *Type of Signal on page 398*.

Limitations

The value must be less than the value of the *Maximum Logical Value*.

Default value

The default value is 0.

4.8.20 Minimum Physical Value

Parent

Minimum Physical Value belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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 on page 413*.

4.8.21 Minimum Physical Value Limit

Parent

Minimum Physical Value Limit belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Type of Signal on page 398*.

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 on page 414*.

4.8.22 Minimum Bit Value

Parent

Minimum Bit Value belongs to the type *Signal*, in the topic *I/O System*.

Description

The parameter *Minimum Bit Value* specifies the bit value that will correspond to the *Minimum Logical Value*.

Usage

See *Maximum Bit Value on page 415*.

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. For more information, see *Type of Signal on page 398*.

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*.

4.8.23 Number Of Bits

Parent

Number Of Bits belongs to the type *Signal*, in the topic *I/O System*.

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. For more information, see *Device Mapping on page 401*.

Default value

The default value is 23.

Allowed values

1 to 32.

4.9 Type Signal Safe Level

4.9.1 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.

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. For more information, see *Safe Level on page 407*.

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. For more information, see *Default Value on page 406*.
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.

Example

This is an example of a signal safe level.

Parameter:	Value:
Name	MySafeLevel

Continues on next page

Parameter:	Value:
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

Operating manual - OmniCore

4.9.2 Name

Parent

Name belongs to the type *Signal Safe Level*, in the topic *I/O System*.

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.

The string must follow the RAPID rules described in *Technical reference manual - RAPID Overview*.

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.

4.9.3 Action When Startup

Parent

Action When Startup belongs to the type *Signal Safe Level*, in the topic *I/O System*.

Description

Specifies the value for a logical output signal after startup of the robot system.

Default value

Set default value, for more information, see *Default Value on page 406*.

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.

For more information, see *Device Mapping on page 401*.

4.9.4 Action when Signal Accessible

Parent

Action when Signal Accessible belongs to the type *Signal safe Level*, in the topic *I/O System*.

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, for more information, see *Default Value on page 406*.

Set last value

Set zero value

4.9.5 Action when Signal Not Accessible

Parent

Action when Signal Not Accessible belongs to the type *Signal Safe Level*, in the topic *I/O System*.

Description

Specifies the value for a logical output signal when its physical state becomes not accessible.

Default value

Set default value, for more information, see *Default Value on page 406*.

Allowed values

Set default value

Set last value

Set zero value

4.9.6 Action when System Shutdown

Parent

Action when System Shutdown belongs to the type *Signal SafeLevel*, in the topic *I/O System*.

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, for more information, *Default Value on page 406*.

Set last value

Set zero value

This page is intentionally left blank

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 types for *Most Common Instructions* are identical and therefore only described in one section, but valid for all three types.

5.2 Type Backup Settings

5.2.1 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.

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.

5.2.2 Name

Parent

Name belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

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. For more information, see *Unique name on page 433*.

If the *Name* parameter is undefined, the default backup name SystemName_Backup_Date (for example, SystemX_Backup_20100101) is suggested.

5.2.3 Path

Parent

Path belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

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

5.2.4 Unique name

Parent

Unique name belongs to the type *Backup Settings*, in the topic *Man-machine communication*.

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.

5.2.5 Disable name change

Parent

Disable name change belongs to the type *Backup Settings*, in the topic *Man-machine communication.*

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.

5.3 Type Block IO in MotorsOff

5.3.1 The Block IO in MotorsOff type

Overview

This section describes the type *Block IO in MotorsOff* which belongs to the topic *Man-machine communication*. Each parameter of this type is described in a separate information topic in this section.

Type description

The type *Block IO in MotorsOff* enables a function that blocks changes of I/O signals in manual mode unless the controller is in motors on state. This restriction is only applicable in RobotStudio I/O views. This is not a safety function, and does not guarantee that a signal is not changed on the controller when the I/O view is blocked.

The controller does not need to be restarted for this functionality to take effect.

Usage

This function is useful to restrict incidental changes on the I/O of the controller when the robot cell is not ready. As it causes unexpected behavior of the robot or connected devices.

Limitations

There can be only one instance of the type *Block IO in MotorsOff* in the system. The name of the instance must not be changed.

5.3.2 Enabled

Parent

Enabled belongs to the type *Block IO in MotorsOff*, in the topic *Man-machine communication*.

Description

Set *Enabled* to *True* to activate the function *Block IO in MotorsOff*.

Allowed values

True or False.

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.

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	

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

Continues on next page

Parameter:	Value:
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*.

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. For more information, see *Instruction Name on page 442*.

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.

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*. For more information, see *Instruction Name on page 442*.

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. For more information, see *Alternative Number on page 441*.

Related information

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*. For more information, see *Instruction Name on page 442*.

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. For more information, see *Parameter Number on page 440*.

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

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*.

Description

Instruction Name defines which instruction to use if the parameter *Name* contains more information than only the instruction. For more information, see *Name on page 439*.

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

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*.

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.

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*. For more information, see *The Signal type on page 393*.

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.

5.5.3 Signal Type

Parent

Signal Type belongs to the type *Most Common I/O Signal*, in the topic *Man-machine communication*.

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

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.

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

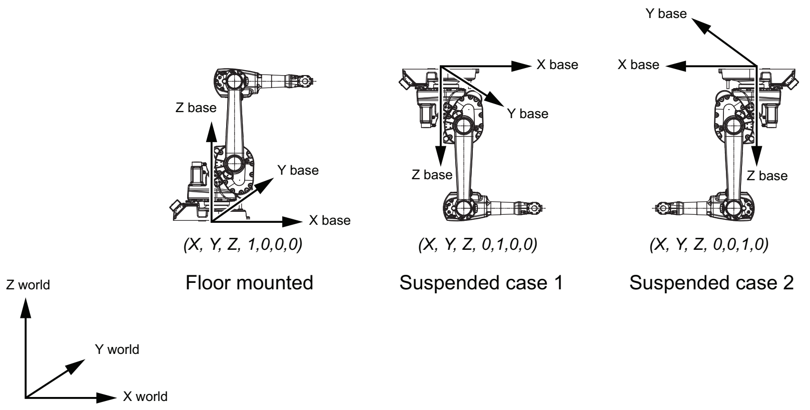
To define the base frame:

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 on page 740*.
4. Save the changes.

Additional information

The illustration shows some examples of frame definitions.



en0300000423

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*, for more information, see *Gravity Alpha on page 749*.
   - *Gravity Beta*, for more information, see *Gravity Beta on page 752*.

   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.

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:

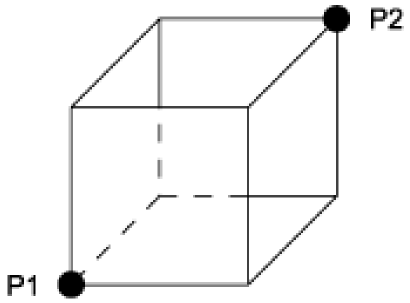
1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to edit.
3. Edit the parameters *Upper Joint Bound* and *Lower Joint Bound* to set the respective limit of the work area for this joint in radians. For more information, see *Upper Joint Bound on page 471* and *Lower Joint Bound on page 472*.
4. Save the changes.

For more information, see *How to restrict the work area for parallel arm robots on page 451*.

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. For more information, see *Upper Work Area x, y, z on page 756* and *Lower Work Area x, y, z on page 757*.
3. Save the changes.

Note

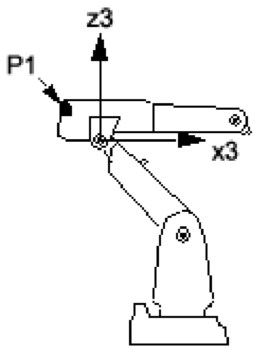
The system parameters that define the work area for parallel robot are valid only for IRB 340 and IRB 360 robots.

For more information, see *How to restrict the work area for articulated robots on page 450*.

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

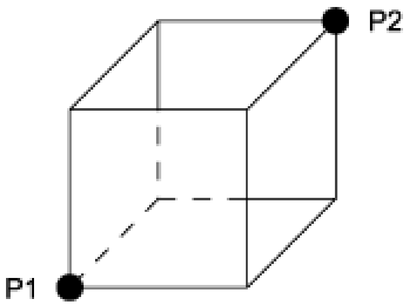
P1	Arm check point
z3	z-axis for arm 3
x3	x-axis for arm 3

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 Arm Check Point type on page 496*.
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, see *The Arm type on page 468*.
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.

   For detailed information, see *Upper Check Point Bound x, y, z on page 759* and *Lower Check Point Bound x, y, z on page 760*.
10. Save the changes.

Related information

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.

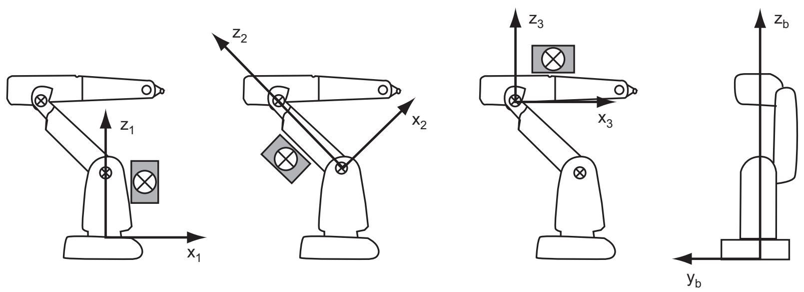
For more information about arm loads, see the type *Arm Load*.

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 Arm Load type on page 499*.

Continues on next page

4. In the topic **Motion**, choose the type **Arm** and select the arm that the load is mounted on.

   For detailed information, see *The Arm type on page 468*.
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 service routine *LoadIdentify* is described in *Operating manual - OmniCore*.

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 Mains type on page 612*.
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.

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**.

   For more information, see *The Motion Supervision type on page 688*.
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

Application manual - Controller software OmniCore

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 (for example, 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.

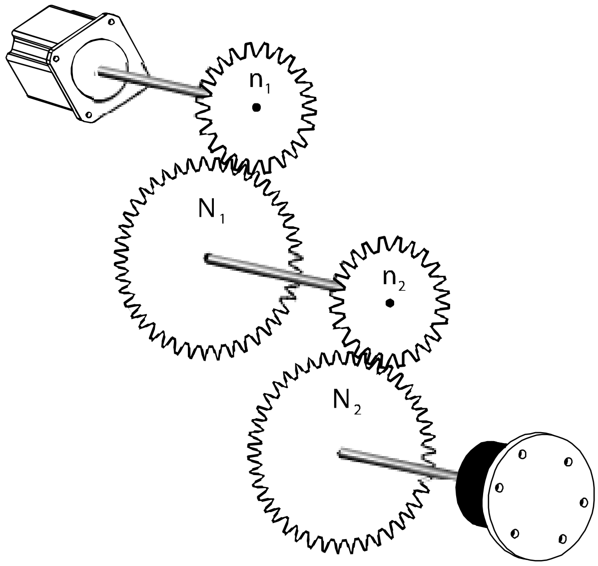
Limitations

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.

For detailed information, see *The Transmission type on page 871*.

Related information

Application manual - Controller software OmniCore

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:

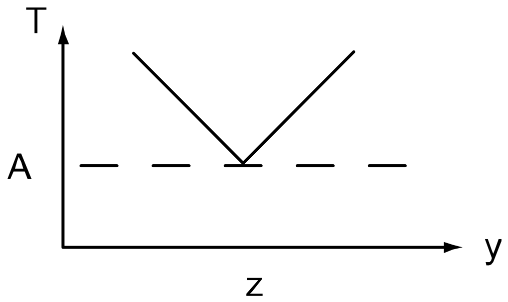
$T = A + |k \times (0 - \theta_0)|$

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**.

   For more information, see *The Arm type on page 468*.
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*.

   For more information, see *External Const Torque on page 481*, *External Proportional Torque on page 484*, and *External Torque Zero Angle on page 485*.
4. Save the changes.

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, θ_0 = 0, k = 5 / (200 × (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. For example, 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**.

   For more information, see *The Arm type on page 468*.
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*. For more information, see *Jam Supervision Trim Factor on page 477*, *Load Supervision Trim Factor on page 478*, *Speed Supervision Trim Factor on page 479*, and *Position Supervision Trim Factor on page 480*.
4. Save the changes.

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.

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*.

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*.

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*.

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.

6.4 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.

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.

6.4.2 Name

Parent

Name belongs to the type *Arm*, in the topic *Motion*.

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*.

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 OmniCore.

6.4.4 Upper Joint Bound

Parent

Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

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

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. For more information, see *How to restrict the work area for articulated robots on page 450*.

Allowed values

A value between +-1256640 radians.

6.4.5 Lower Joint Bound

Parent

Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

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

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. For more information, see *How to restrict the work area for articulated robots on page 450*.

Allowed values

A value between +-1256640 radians.

6.4.6 Independent Upper Joint Bound

Parent

Independent Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

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 OmniCore.

6.4.7 Independent Lower Joint Bound

Parent

Independent Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

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 OmniCore.

6.4.8 Calibration Position

Parent

Calibration Position belongs to the type *Arm*, in the topic *Motion*.

Description

Calibration Position defines the position of the axis when it was fine calibrated.

Usage

This value should specify a well-defined position in which the axis can be positioned repeatedly. This position is then used when updating *Calibration Offset* and revolution counter. For more information, see *Calibration Offset on page 718*.

Allowed values

A value between -1000 and 1000, specifying the position in radians.

Related information

Product Manual for the manipulator.

6.4.9 Performance Quota

Parent

Performance Quota belongs to the type *Arm*, in the topic *Motion*.

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*.

Description

Jam Supervision Trim Factor defines the tune factor for jam supervision. For more information, see *How to define supervision level on page 463*.

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.

6.4.11 Load Supervision Trim Factor

Parent

Load Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Description

Load Supervision Trim Factor defines the tune factor for load supervision. For more information, see *How to define supervision level on page 463*.

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.

6.4.12 Speed Supervision Trim Factor

Parent

Speed Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Description

Speed Supervision Trim Factor defines the tune factor for speed supervision. For more information, see *How to define supervision level on page 463*.

Usage

The factor influences the maximum allowed speed error.

Allowed values

A number between 0.05 and 10.0.

6.4.13 Position Supervision Trim Factor

Parent

Position Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Description

Position Supervision Trim Factor defines the tune factor for position supervision.
For more information, see *How to define supervision level on page 463*.

Usage

The factor influences the maximum allowed position error.

Allowed values

A number between 0.1 and 10.0.

6.4.14 External Const Torque

Parent

External Const Torque belongs to the type *Arm*, in the topic *Motion*.

Description

External Const Torque defines the external constant torque. For more information, see *How to define external torque on page 461*.

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.

6.4.15 Use Arm Load

Parent

Use Arm Load belongs to the type *Arm*, in the topic *Motion*.

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. For more information, see *The Arm Load type on page 499*.

6.4.16 Use Check Point

Parent

Use Check Point belongs to the type *Arm*, in the topic *Motion*.

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

The Arm Check Point type on page 496.

6.4.17 External Proportional Torque

Parent

External Proportional Torque belongs to the type *Arm*, in the topic *Motion*.

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. For more information, see *How to define external torque on page 461*.

Allowed values

A value between -100,000 and 100,000, specifying the scale factor in Nm/rad.

6.4.18 External Torque Zero Angle

Parent

External Torque Zero Angle belongs to the type *Arm*, in the topic *Motion*.

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. For more information, see *How to define external torque on page 461*.

Allowed values

A value between -100,000 and 100,000, specifying the position in radians.

6.4.19 Load Id Acceleration Ratio

Parent

Load Id Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

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 0.02 and 1.0.

6.4.20 Angle Acceleration Ratio

Parent

Angle Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

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*.

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 - Functional safety and 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 - Functional safety and SafeMove

6.4.22 Change to Logical Axis

Parent

Change to Logical Axis belongs to the type *Arm*, in the topic *Motion*.

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 track motions (IRBT). If the value is zero, then no change will happen and the value in the Joint will be used as normal. For more information, see *Logical Axis on page 575*.

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.

6.4.23 Thermal Supervision Sensitivity Ratio

Parent

Thermal Supervision Sensitivity Ratio belongs to the type *Arm*, in the topic *Motion*.

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.

6.4.24 Brake applied movement detection factor

Parent

Brake applied movement detection factor belongs to the type *Arm*, in the topic *Motion*.

Description

Brake applied movement detection factor defines the factor of the default detection level of movement when brake applied.

Usage

For example, set this factor to 0.1 to make the detection ten times more sensitive of the movement when the brake is applied, and set the factor to more than 1 to make the detection less sensitive. A zero value means that the function is deactivated.

Allowed values

A value between 0 and 10. Default value is 1 for robots where this detection is activated and 0 otherwise.

6.4.25 Lead through stability margin

Parent

Lead through stability margin belongs to the type *Arm*, in the topic *Motion*.

Description

The parameter *Lead through stability margin* can be used to adjust the behavior of the low-level joint control.

Usage

The parameter *Lead through stability margin* is used joint-wise and should be decremented in case of unexpected buzzing noise coming from the joint.

Limitations

Lead through stability margin is only used for the following robots:

- CRB 15000

Setting this parameter for any other robot will not have any effect.

Allowed values

A value between 0.01 and 1.

Default value 0.8.

Related information

Lead through stiffness scale on page 774

6.4.26 Lead through load compensation deadband

Parent

Lead through load compensation deadband belongs to the type *Arm*, in the topic *Motion*.

Description

The parameter *Lead through load compensation deadband* can be used to tune the assumed maximum joint torque from cables and other disturbances affecting the robot arm, when the parameter *Lead through load compensation* in the type *Robot* is set to *Always*.

Usage

The value is set joint-wise. A larger value makes the robot less sensitive to drift caused by torque disturbances acting on the joint, but also makes the robot feel less soft.

Note

This parameter does not need a restart of the controller to apply the changes. Hence, it is possible to test directly after changing the parameter value.

Limitations

Lead through load compensation deadband is only used for the following robots:

- CRB 15000

Setting this parameter for any other robot will not have any effect.

The parameter is only used when *Lead-through Mode* is set to *Axis 1-6*.

This parameter only has effect when the parameter *Lead through load compensation* in the type *Robot* is set to *Always*.

Allowed values

A value between 0 and 50.

Default value 10.

Related information

Lead through load compensation on page 775

6.4.27 Joint Id

Parent

Joint Id belongs to the type *Arm*, in the topic *Motion*.

Description

The parameter *Joint Id* can be used to change the joint id number used when creating a safety configuration. If the value is zero, the value of the logical axis (*Logical Axis*) configured in the type *Joint* will be used as the default value. Changing the default value may be required when more than one joint in the same cabinet share the same logical axis number, for example, in ABB positioners (IRP/IRBP). For more information, see *Logical Axis on page 575*.

Usage

The value of *Joint Id* is used by the robot controller to identify individual joints in the safety configuration. For each joint included in the safety configuration, the *Joint Id* number must be unique within its cabinet. Robots from ABB always use the values from 1 to 6.

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

Application manual - Functional safety and SafeMove

6.4.28 Ultra Accuracy stability margin

Parent

Ultra Accuracy stability margin belongs to the type *Arm*, in the topic *Motion*.

Description

The parameter *Ultra Accuracy stability margin* can be used to adjust the behavior of the low-level joint control.

Usage

The parameter *Ultra Accuracy stability margin* is used joint-wise and should be decremented in case of unexpected buzzing noise coming from the joint when running in ultra accuracy mode.

Decrement in steps of 0.1 until the joint is stable.

Limitations

The ultra accuracy mode is only used for GoFa CRB 15000 and requires the option 3101-10.

Setting this parameter for any other robot will not have any effect.

Allowed values

A value between 0.01 and 1.

The default value is 1.

Related information

Ultra Accuracy on page 779

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. For more information, see *How to define arm check point on page 452*.

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. For more information, see *Check Point Bound Limit Outside Cube on page 761*.

6.5.2 Name

Parent

Name belongs to the type *Arm Check Point*, in the topic *Motion*. For more information, see *How to define arm check point on page 452*.

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.

6.5.3 Position x, y, z

Parent

Position x, *Position y*, and *Position z* belong to the type *Arm Check Point*, in the topic *Motion*. For more information, see *How to define arm check point on page 452*.

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.

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. For more information, see *How to define arm loads on page 454*.

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.

6.6.2 Name

Parent

Name belongs to the type *Arm Load*, in the topic *Motion*. For more information, see *How to define arm loads on page 454*.

Description

Name specifies the name of the arm load setting it belongs to.

Allowed values

A string with maximum 32 characters.

6.6.3 Mass

Parent

Mass belongs to the type *Arm Load*, in the topic *Motion*. For more information, see *How to define arm loads on page 454*.

Description

Mass specifies the mass of the equipment mounted on a robot arm.

Allowed values

A value between 0 and 50000, specifying the weight in kg.

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*. For more information, see *How to define arm loads on page 454*.

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 -30 and +30, specifying the coordinate in meters.

6.6.5 Inertia x, y, z

Parent

Inertia x, *Inertia y*, and *Inertia z* belong to the type *Arm Load*, in the topic *Motion*.
For more information, see *How to define arm loads on page 454*.

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 1000, specifying the moment of inertia in kgm^2.

6.7 Type Brake

6.7.1 The Brake type

Overview

This section describes the type *Brake* which belongs to the topic *Motion*.

Type description

The type *Brake* is used to specify brake parameters for a specific joint. For more information, see *The Joint type on page 573*.

6.7.2 Name

Parent

Name belongs to the type *Brake*, in the topic *Motion*.

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*.

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*.

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.

6.7.5 Brake Control On Delay

Parent

Brake Control On Delay belongs to the type *Brake*, in the topic *Motion*.

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*. For more information, see *Control Off Delay on page 507*.

Allowed values

A value between 0 and 30 seconds.

Default value is 0.

6.7.6 Brake Control Min Delay

Parent

Brake Control Min Delay belongs to the type *Brake*, in the topic *Motion*.

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*.

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 100000 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*.

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*.

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 decelerate 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

The default value is 5.

6.7.10 Max Static Arm Torque

Parent

Max Static Arm Torque belongs to the type *Brake*, in the topic *Motion*.

Description

The parameter static torque should be highest that the brake needs to withstand, when the additional axis is positioned in maximum gravity. The value is entered in [Nm] and calculated to the motor side.

Usage

The parameter *Max Static Arm Torque* needs to be calculated and entered into the configuration to run the Cyclic Brake Check (CBC) on ABB motor units. CBC uses this value when testing the brake at error-level.

Allowed values

To calculate the parameter for an axis that has no gravity, for example a track, the below formula may be used:

Max Static Arm Torque = Tbrake min/1.35

Tbrake min can be found in the product specification for the specific motor unit, see *Product specification - Motor Units and Gear Units*.

Related information

For more information about Cyclic Brake Check, see *Application manual - Functional safety and SafeMove*.

6.7.11 Max Brake Release Time

Parent

Max Brake Release Time belongs to the type *Brake*, in the topic *Motion*.

Description

The parameter is the maximum time for release of the brake.

Usage

The parameter is used to wait until the brakes are released. The time can be increased if the brakes are slow.

Allowed values

0-2 and the default value is 0.15 sec.

6.7.12 Use Brake Type

Parent

Use Brake Type belongs to the type *Brake*, in the topic *Motion*.

Description

Use Brake Type defines which type of brake is used for this type.

Allowed values

A string with maximum 32 characters.

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.

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 OmniCore, chapter *Advanced Shape Tuning*.

6.8.2 Name

Parent

Name belongs to the type *Control Parameters*, in the topic *Motion*.

Description

Name defines the name to use for the control parameters.

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*.

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*.

Allowed values

Yes or No.

Related information

Application manual - Controller software OmniCore.

6.8.4 Friction FFW Level

Parent

Friction FFW Level belongs to the type *Control Parameters*, in the topic *Motion*.

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.

Allowed values

A decimal number between 0 and 15 in Nm.

Related information

Application manual - Controller software OmniCore.

6.8.5 Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Control Parameters*, in the topic *Motion*.

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.

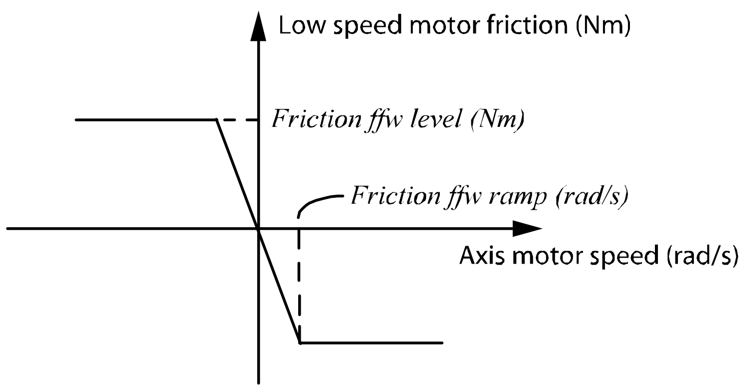
Allowed values

A number between 0.001 and 10 in radians/second.

Related information

Application manual - Controller software OmniCore.

Illustration



en0300000278

6.9 Type Drive System

6.9.1 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.

Type description

The type *Drive System* is used to identify and specify each drive system used in the robot system.

6.9.2 Name

Parent

Name belongs to the type *Drive System*, in the topic *Motion*.

Description

Defines the name for the drive system.

Allowed values

A string with maximum 32 characters.

6.9.3 Use DC-link

Parent

Use DC-link belongs to the type *Drive System*, in the topic *Motion*.

Description

Use DC-link determines which dc-link (rectifier) unit should be used.

Allowed values

A string with maximum 32 characters.

6.9.4 Use Trafo

Parent

Use Trafo belongs to the type *Drive System*, in the topic *Motion*.

Description

Use Trafo determines which transformer should be used.

Allowed values

A string with maximum 32 characters.

6.9.5 Use Drive Unit

Parent

Use Drive Unit belongs to the type *Drive System*, in the topic *Motion*. For more information, see *The Drive Unit type on page 526*.

Description

Use Drive Unit determines which drive unit should be used.

Allowed values

A string with maximum 32 characters.

Related information

Application manual - Additional axes

6.10 Type Drive Unit

6.10.1 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.

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 521.

6.10.2 Name

Parent

Name belongs to the type *Drive Unit*, in the topic *Motion*.

Description

Defines the name for the drive unit.

Allowed values

A string with maximum 32 characters.

6.10.3 Drive Unit Position

Parent

Drive Unit Position belongs to the type *Drive Unit*, in the topic *Motion*.

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 0 and 9.

6.11 Type Force Master

6.11.1 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.

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 OmniCore

6.11.2 Name

Parent

Name belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.3 Use Force Master Control

Parent

Use Force Master Control belongs to the type *Force Master*, in the topic *Motion*.

Description

Use Force Master Control determines which *Force Master Control* should be used. For more information, see *The Force Master Control type on page 548*.

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.

6.11.4 References Bandwidth

Parent

References Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.5 Use Ramp Time

Parent

Use Ramp Time belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.6 Ramp when Increasing Force

Parent

Ramp when Increasing Force belongs to the type *Force Master*, in the topic *Motion*.

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 70 Nm/s.

6.11.7 Ramp Time

Parent

Ramp Time belongs to the type *Force Control*, in the topic *Motion*.

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.

6.11.8 Collision LP Bandwidth

Parent

Collision LP Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.9 Collision Alarm Torque

Parent

*Collision Alarm Torque*belongs to the type*Force Master*, in the topic *Motion*.

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.

6.11.10 Collision Speed (m/s)

Parent

Collision Speed (m/s) belongs to the type *Force Master*, in the topic *Motion*.

Description

Collision Speed (m/s) determines the servo gun speed during the first gun closing of new tip calibrations and tool change calibrations. These calibrations affect the position calibration.

Usage

The only reason for changing *Collision Speed (m/s)* is if repetitive tip wear calibrations give varying results. A lower speed can improve the repeatability.

Limitations

Collision Speed (m/s) 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.

6.11.11 Collision Delta Position (m)

Parent

Collision Delta Position (m) belongs to the type *Force Master*, in the topic *Motion*.

Description

Collision Delta Position (m) 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 (m) is used for the first gun closing of new tip 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 (m)* can remove a constant calibration error, but does not have any effect if repetitive tip wear calibrations give varying results.

Limitations

Collision Delta Position (m) can only be used for servo tools.

Allowed values

A value between 0 and 1 meters.

The default value is 0.0019 m.

6.11.12 Force Detection Bandwidth

Parent

Force Detection Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.13 Delay Ramp

Parent

Delay Ramp belongs to the type *Force Master*, in the topic *Motion*.

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).

6.11.14 Ramp to Real Contact

Parent

Ramp to Real Contact belongs to the type *Force Master*, in the topic *Motion*.

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.

6.11.15 Force Detection Min Time

Parent

Force Detection Min Time belongs to the type *Force Master*, in the topic *Motion*.

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 Setup

6.11.16 Leak Control for Search Signal

Parent

Leak Control for Search Signal belongs to the type *Force Master*, in the topic *Motion*.

Description

Leak Control for Search Signal defines the leak factor of the search.

Usage

When search for plate is activate in a spot welding instruction, the servo gun will perform a movement towards the plate and stops immediately when the plate is found. The plate is considered to be found when the signal value is bigger than *Threshold for Search Trigger*.

Limitations

Leak Control for Search Signal is only used for servo tools.

Allowed values

A value between 0 and 10000.

Default value is 0.1. This must be tuned depending on gun characteristics.

Related information

Threshold for Search Trigger on page 546.

Application manual - Servo Gun Setup

6.11.17 Bandwidth of Speed Error Filter

Parent

Bandwidth of Speed Error Filter belongs to the type *Force Master*, in the topic *Motion*.

Description

Bandwidth of Speed Error Filter defines the bandwidth of the Low Pass filter used in the search process.

Usage

To avoid false search stops due to noisy signals the speed error is filtered by a Low Pass filter. The cut-off frequency is set by the parameter *Bandwidth of Speed Error Filter*.

Limitations

Bandwidth of Speed Error Filter is only used for servo tools.

Allowed values

An value between 0.1 and 250 Hz.

Default value is 10 Hz.

Related information

Application manual - Servo Gun Setup

6.11.18 Threshold for Search Trigger

Parent

Threshold for Search Trigger belongs to the type *Force Master*, in the topic *Motion*.

Description

Threshold for Search Trigger defines the trigger level of the search hit.

Usage

When search for plate is activate in a spot welding instruction, the servo gun will perform a movement towards the plate and stops immediately when the plate is found. The plate is considered to be found when the signal value is bigger than *Threshold for Search Trigger*.

Limitations

Threshold for Search Trigger is only used for servo tools.

Allowed values

A value between 0 and 10000.

Default value is 3. This must be tuned depending on gun characteristics.

Related information

Application manual - Servo Gun Setup

6.11.19 Search reverse distance

Parent

Search reverse distance belongs to the type *Force Master*, in the topic *Motion*.

Description

Search reverse distance defines how long distance the gun arm should move in opposite direction directly after a search hit.

Usage

When search for plate is activate in a spot welding instruction, the servo gun will perform a movement towards the plate and stops immediately when the plate is found. To reduce search impact as much as possible the gun will automatically move in the opposite direction directly after the plate is found.

The return distance is set by the parameter *Search reverse distance*.

Limitations

Search reverse distance is only used for servo tools.

Allowed values

A value between 0 and 0.01 meters.

Default value is 0.002 (2 mm).

Related information

Application manual - Servo Gun Setup

6.12 Type Force Master Control

6.12.1 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.

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 damage 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

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 OmniCore

Example

In this example, two points are used to define the speed limit and speed loop gain. Any values given for point 3 to 6 are ignored.

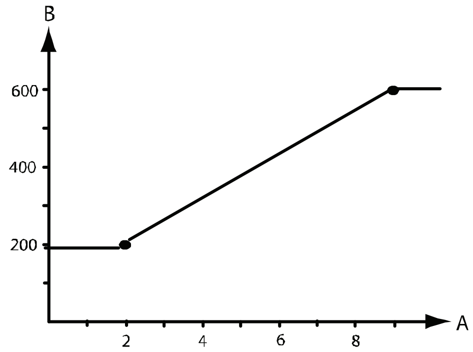
The parameters in the type *Force Master Control* are set to the following values:

Parameter:	Value:
No. of speed limits	2

Continues on next page

Parameter:	Value:
Torque 1	2
Torque 2	8
Speed Limit 1	200
Speed Limit 2	600
Kv 1	0.3
Kv 2	0.8

The results of this configuration are the following graphs for speed limit and speed loop gain:



xx1600001321

A	Torque (Nm)
B	Speed limit (rad/s on motor)
C	Speed loop gain (Nms/rad)

6.12.2 Name

Parent

Name belongs to the type *Force Master Control*, in the topic *Motion*.

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.

6.12.3 No. of Speed Limits

Parent

No. of Speed Limits belongs to the type *Force Master Control*, in the topic *Motion*.

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 548*).

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 OmniCore

6.12.4 Torque 1

Parent

Torque 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Torque 1 defines the ordered closing torque for the first point in the speed limit graph (see *Example on page 548*).

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 *Tool Control*.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 1 Nm.

Related information

Application manual - Controller software OmniCore

6.12.5 Torque 2

Parent

Torque 2 belongs to the type *Force Master Control*, in the topic *Motion*.

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 548*).

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.

It is possible to change the values to index 6 manually by changing a MOC.cfg.

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Torque 2* is not used. For more information, see *No. of Speed Limits on page 551*.

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

Application manual - Controller software OmniCore

6.12.6 Speed Limit 1

Parent

Speed Limit 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Speed Limit 1 defines the maximum allowed speed for the torque specified in *torque 1*. For more information, see *Torque 1 on page 552*.

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 548*).

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

Application manual - Controller software OmniCore

6.12.7 Speed Limit 2

Parent

Speed Limit 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Speed Limit 2 defines the maximum allowed speed for the torque specified in *torque 2*. For more information, see *Torque 2 on page 553*.

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 548*).

It is possible to change the values to index 6 manually by changing a MOC.cfg.

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Speed Limit 2* is not used. For more information, see *No. of Speed Limits on page 551*.

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

Application manual - Controller software OmniCore

6.12.8 Kv 1

Parent

Kv 1 belongs to the type *Force Master Control*, in the topic *Motion*.

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. For more information, see *Torque 1 on page 552*.

Usage

Set *Kv 1* to the proportional gain you want for the first point in the speed limit graph (see *Example on page 548*).

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

Application manual - Controller software OmniCore

6.12.9 Kv 2

Parent

Kv 2 belongs to the type *Force Master Control*, in the topic *Motion*.

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. For more information, see *Torque 2 on page 553*.

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 548*).

It is possible to change the values to index 6 manually by changing a MOC.cfg.

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Kv 2* is not used. For more information, see *No. of Speed Limits on page 551*.

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

Application manual - Controller software OmniCore

6.12.10 Speed limit factor in force mode

Parent

Speed limit factor in force mode belongs to the type *Force Master Control*, in the topic *Motion*.

Description

When force is built up and tool is squeezing, the *Speed limit factor in force mode* defines a factor the speed limitation is multiplied with.

Usage

The parameter *Speed limit factor in force mode* is used in the processes where the material collapses during the squeezing process. For example, during spot welding wire to wire.

By setting *Speed limit factor in force mode* to a higher value than 1, it is possible to disable or reduce the speed limitation if the material is collapsing at the end of the squeezing process. This will reduce the risk of unintendedly release of the squeezing force.

Example

If speed limit 2 = 50 rad/s.

If speed limit factor in force mode = 2.

Then during the force buildup 50 rad/s will be the speed limit that will reduce the torque.

As soon as the force is fully built up, the speed limit will be increased to 100 rad/s to avoid any force reduction if the material is collapsing at a later stage of the process.

Limitations

Speed limit factor in force mode can only be used if you have servo tools.

Allowed values

A value between 0.1 to 10.

The default value is 1.

6.12.11 Speed During Search

Parent

Speed During Search belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Speed During Search defines the search speed used in the search process.

Usage

When a spot instruction is using a search argument, the gun will start a movement toward the plate with the speed defined in the parameter *Speed During Search*.

Limitations

Speed During Search is only used for servo tools.

Allowed values

A value between 0.001 and 0.1 m/s.

Default value is 0.05 (50 mm/s).

Related information

Application manual - Servo Gun Setup

6.12.12 Prop. Gain in Speed Loop During Search

Parent

Prop. Gain in Speed Loop During Search belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Prop. Gain in Speed Loop During Search is the proportional gain in the speed loop during the search process.

Usage

To be able to have a fixed search tuning of the speed loop a special proportional gain is used in the speed loop. This can in many cases be same value as in the parameter *Kv* of *Lag Control Master*, but might in some cases have to be tuned.

Limitations

Prop. Gain in Speed Loop During Search is only used for servo tools.

Allowed values

A value between 0.001 and 100.

Default value is 1.

Related information

Application manual - Servo Gun Setup

6.12.13 Integration Time in Speed Loop

Parent

Integration Time in Speed Loop belongs to the type *Force Master Control*, in the topic *Motion*.

Description

Integration Time in Speed Loop defines the integration time in the speed loop during the search process.

Usage

To be able to have a fixed search tuning of the speed loop an integration time can be used in the speed loop. This can in many cases be same value as in the parameter *Ti* of *Lag Control Master*, but might in some cases have to be tuned.

Limitations

Integration Time in Speed Loop is only used for servo tools.

Allowed values

A value between 0.1 and 250 Hz.

Default value is 15 Hz.

Related information

Application manual - Servo Gun Setup

6.13 Type Friction Compensation

6.13.1 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.

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 OmniCore, chapter *Advanced Shape Tuning*.

6.13.2 Name

Parent

Name belongs to the type *Friction Compensation*, in the topic *Motion*.

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.

6.13.3 Friction FFW On

Parent

Friction FFW On belongs to the type *Friction Compensation*, in the topic *Motion*.

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 OmniCore

6.13.4 Friction FFW Level

Parent

Friction FFW Level belongs to the type *Friction Compensation*, in the topic *Motion*.

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 OmniCore*.

Limitations

Friction FFW Level is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A decimal number between 0 and 100 (in Nm).

Related information

Application manual - Controller software OmniCore

6.13.5 Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Friction Compensation*, in the topic *Motion*.

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 OmniCore*.

Limitations

Friction FFW Ramp is only useful if you have the RobotWare option *Advanced Shape Tuning*.

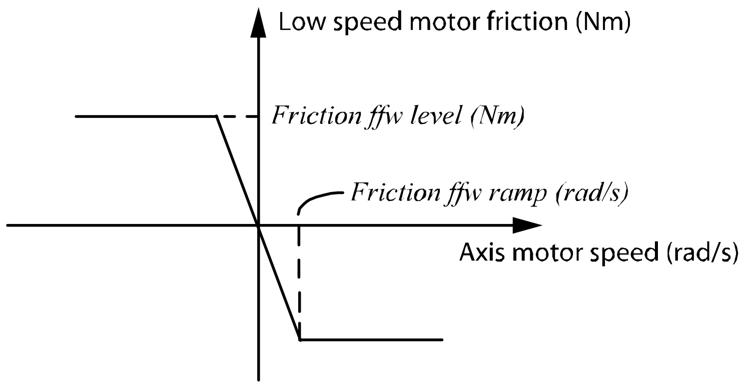
Allowed values

A number between 0.001 and 10 radians/second.

Related information

Application manual - Controller software OmniCore

Illustration



en0300000278

6.14 Type Jog Parameters

6.14.1 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.

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).

6.14.2 Name

Parent

Name belongs to the type *Jog Parameters*, in the topic *Motion*.

Description

Name defines the name of the Jog parameters data.

Allowed values

A string with maximum 32 characters.

6.14.3 Configurable Linear Step Size (m)

Parent

Configurable Linear Step Size (m) belongs to the type *Jog Parameters*, in the topic *Motion*.

Description

Configurable Linear Step Size (m) 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.

6.14.4 Configurable Reorient Step Size (rad)

Parent

Configurable Reorient Step Size (rad) belongs to the type *Jog Parameters*, in the topic *Motion*.

Description

Configurable Reorient Step Size (rad) 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.

6.14.5 Configurable Joint Step Size (rad)

Parent

Configurable Joint Step Size (rad) belongs to the type *Jog Parameters*, in the topic *Motion*.

Description

Configurable Joint Step Size (rad) 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.

6.14.6 Jog Mode

Parent

Jog Mode belongs to the type *Jog Parameters*, in the topic *Motion*.

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. However, Responsive is activated for some robot types.

Allowed values

Standard

Responsive

6.15 Type Joint

6.15.1 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.

Type description

The *Joint* type contains parameters that define a joint.

Related information

The Arm type on page 468.

The Measurement Channel type on page 616.

6.15.2 Name

Parent

Name belongs to the type *Joint*, in the topic *Motion*.

Description

Name defines the unique name to use for this joint.

Allowed values

A string with maximum 32 characters.

6.15.3 Logical Axis

Parent

Logical Axis belongs to the type *Joint*, in the topic *Motion*.

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

6.15.4 Use Drive System

Parent

Use Drive System belongs to the type *Joint*, in the topic *Motion*.

Description

Use Drive System determines which drive system should be used. For more information, see *The Drive System type on page 521*.

Allowed values

A string with maximum 32 characters.

6.15.5 Use Process

Parent

Use Process belongs to the type *Joint*, in the topic *Motion*.

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*. For more information, see *Name on page 733*.

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

Application manual - Controller software OmniCore

6.15.6 Lock Joint in Ipol

Parent

Lock Joint in Ipol belongs to the type *Joint*, in the topic *Motion*.

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 OmniCore

6.15.7 Follower to Joint

Parent

Follower to Joint belongs to the type *Joint*, in the topic *Motion*.

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 OmniCore

6.15.8 Drive Module Number

Parent

Drive Module Number belongs to the type *Joint*, in the topic *Motion*.

Description

Drive Module Number defines the drive module number that should be used. For more information, see *Use Drive Module on page 581*.

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.

6.15.9 Use Drive Module

Parent

Use Drive Module belongs to the type *Joint*, in the topic *Motion*.

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.

6.15.10 Use Measurement Channel

Parent

Use Measurement Channel belongs to the type *Joint*, in the topic *Motion*.

Description

Use Measurement Channel determines which measurement channel should be used. For more information, see *The Measurement Channel type on page 616*.

Usage

Use Measurement Channel points to a measurement channel ID defined by the parameter *Name* in the Type *Measurement Channel*.

Allowed values

A string with maximum 32 characters.

6.16 Type Lag Control Master 0

6.16.1 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.

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.

6.16.2 Name

Parent

Name belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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.

6.16.3 Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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).

6.16.4 Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Lag Control Master*, in the topic *Motion*.

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 1000 (Nms/rad).

6.16.5 Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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.

6.16.6 Forced Control Active

Parent

Forced Control Active belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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. For more information, see *Forced Factor for Kp on page 589* and *Forced Factor for Ki on page 590*.

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. For more information, see *Affects Forced Control on page 854*.

Allowed values

Yes or No.

Related information

Application manual - Additional axes

6.16.7 Forced Factor for Kp

Parent

Forced Factor for Kp belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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 10.

6.16.8 Forced Factor for Ki

Parent

Forced Factor for Ki belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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 10.

6.16.9 Raise Time for Kp

Parent

Raise Time for Kp belongs to the type *Lag Control Master*, in the topic *Motion*.

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.

6.16.10 FFW Mode

Parent

FFW Mode belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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

6.16.11 Bandwidth

Parent

Bandwidth belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Defines the controller bandwidth when *FFW Mode* is set to 1 or 2. For more information, see *FFW Mode on page 592*.

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.

6.16.12 Df

Parent

Df belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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 on page 586*, *Kp, Gain Position Loop on page 585*, *Ti Integration Time Speed Loop on page 587*, and *Inertia on page 597*).

Df is only used when *FFW Mode* is set to 2. For more information, see *FFW Mode on page 592*.

Allowed values

A value between 1 and 100. Default value is 100.

6.16.13 Dw

Parent

Dw belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Can reduce oscillations further when *Df* is set. For more information, see *Df on page 594*.

Usage

The default value of *Dw* is recommended.

Allowed values

A value between 0.002 to 1. Default value is 0.01.

6.16.14 Delay

Parent

Delay belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Reduces overshoot.

Usage

Delay can be used when *Df* is set, to reduce overshoot but it impairs the axis coordination when increased. For more information, see *Df on page 594*.

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.

6.16.15 Inertia

Parent

Inertia belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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. For more information, see *FFW Mode on page 592*.

Allowed values

A value between 0.0 and 10,000.

6.16.16 K Soft Max Factor

Parent

K Soft Max Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Determines the value of the product *Kp Gain Position Loop * Kv Gain Speed Loop* when the soft servo is used with softness 0%. For more information, see *Kp, Gain Position Loop on page 585* and *Kv, Gain Speed Loop on page 586*.

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 0.001 and 1000. Default value is 1.0.

6.16.17 K Soft Min Factor

Parent

K Soft Min Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Determines the value of the product *Kp Gain Position Loop * Kv Gain Speed Loop* if the soft servo is used with softness 100%. For more information, see *Kp, Gain Position Loop on page 585* and *Kv, Gain Speed Loop on page 586*.

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 1000. Default value is 0.01.

6.16.18 Kp/Kv Ratio Factor

Parent

Kp/Kv Ratio Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Description

Defines the factor used to tune the *Kp Gain Position Loop*/*Kv Gain Speed Loop* ratio. For more information, see *Kp, Gain Position Loop on page 585* and *Kv, Gain Speed Loop on page 586*.

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 0.001 and 1000.

6.16.19 Ramp Time

Parent

Ramp Time belongs to the type *Lag Control Master 0*, in the topic *Motion*.

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.

6.17 Type Linked M Process

6.17.1 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.

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 OmniCore, chapter *Electronically Linked Motors*.

6.17.2 Name

Parent

Name belongs to the type *Linked M Process*, in the topic *Motion*.

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.

6.17.3 Offset Adjust. Delay Time

Parent

Offset Adjust. Delay Time belongs to the type *Linked M Process*, in the topic *Motion*.

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 20, specifying the delay in seconds.

Default value: 0.2

6.17.4 Max Follower Offset

Parent

Max Follower Offset belongs to the type *Linked M Process*, in the topic *Motion*.

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 50, specifying the maximum offset in radians (for rotational axes) or meters (for linear axes) on the arm side.

Default value: 0.05.

6.17.5 Max Offset Speed

Parent

Max Offset Speed belongs to the type *Linked M Process*, in the topic *Motion*.

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.

6.17.6 Offset Speed Ratio

Parent

Offset Speed Ratio belongs to the type *Linked M Process*, in the topic *Motion*.

Description

Offset Speed Ratio defines how large a part of the *Max Offset Speed* can be used to compensate for position error. For more information, see *Max Offset Speed on page 606*.

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.

6.17.7 Ramp Time

Parent

Ramp Time belongs to the type *Linked M Process*, in the topic *Motion*.

Description

Ramp Time defines the acceleration up to *Max Offset Speed*. For more information, see *Max Offset Speed on page 606*.

Usage

The proportion constant for position regulation is ramped from zero up to its final value (*Master Follower Kp on page 609*) during *Ramp Time*.

Allowed values

A value between 0.01 and 100, specifying the time in seconds.

Default value: 1

6.17.8 Master Follower Kp

Parent

Master Follower Kp belongs to the type *Linked M Process*, in the topic *Motion*.

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.

6.17.9 Torque follower

Parent

Torque follower belongs to the type *Linked M Process*, in the topic *Motion*.

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.

6.17.10 Torque quota

Parent

Torque quota belongs to the type *Linked M Process*, in the topic *Motion*.

Description

Torque quota is a quota defining how much of the total torque should be applied by the follower axis.

Usage

The follower axis will get its torque reference according to the master axis torque multiplied with the quota.

If motors and gears are identical, it is recommend to set this value a bit lower than 1, for example, 0.95.

This parameter will have no effect if *Torque follower* is set to No.

Allowed values

A value between 0 and 1.

Default value is 1.

6.18 Type Mains

6.18.1 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.

Type description

The type *Mains* defines the drive system's mains power tolerance. The parameters of the Mains type have nominal values. For more information, see *Mains Tolerance Min on page 614*, *Mains Tolerance Max on page 615*, and *How to optimize drive system parameters on page 456*.

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.

6.18.2 Name

Parent

Name belongs to the type *Mains*, in the topic *Motion*.

Description

Name specifies the name of the mains tolerance setting it belongs to.

Allowed values

A string with maximum 32 characters.

6.18.3 Mains Tolerance Min

Parent

Mains Tolerance Min belongs to the type *Mains*, in the topic *Motion*.

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 456*.

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%).

6.18.4 Mains Tolerance Max

Parent

Mains Tolerance Max belongs to the type *Mains*, in the topic *Motion*.

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%).

For more information, see *How to optimize drive system parameters on page 456*.

Allowed values

The default value is 0.1.

6.19 Type Measurement Channel

6.19.1 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.

Type description

The type *Measurement Channel* describes which position sensor on the SMB to use for a joint.

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.

6.19.2 Name

Parent

Name belongs to the type *Measurement Channel*, in the topic *Motion*.

Description

Name defines the name of the used channel.

Allowed values

A string with maximum 32 characters.

6.19.3 Disconnect at Deactivate

Parent

Disconnect at Deactivate belongs to the type *Measurement Channel*, in the topic *Motion*.

Description

Disconnect at Deactivate shall be set if it the physical signals from position sensor is intended to be disconnected 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

6.20 Type Mechanical Unit

6.20.1 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.

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

6.20.2 Name

Parent

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

Description

Name defines the name for the mechanical unit.

Allowed values

A string with maximum 32 characters.

6.20.3 Use Activation Relay

Parent

Use Activation Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

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

6.20.4 Unit has no brake

Parent

Unit has no brake belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

The parameter *Unit has no brake* defines if the mechanical unit has a brake or not.

Usage

Set *Unit has no brake* to *Yes* if the mechanical unit has no brake.

The default value is *No*, that is, that the mechanical unit has a brake.

Allowed values

Yes or No.

6.20.5 Use Brake Relay

Parent

Use Brake Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

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.

6.20.6 Use Connection Relay

Parent

Use Connection Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

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.

6.20.7 Use Robot

Parent

Use Robot belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

Use Robot defines which robot is part of the mechanical unit.

Usage

The robot is defined in the type *Robot*.

For more information, see *Name on page 741*, of the type *Robot*.

Allowed values

A string with maximum 32 characters.

6.20.8 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* belong to the type *Mechanical Unit*, in the topic *Motion*.

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 is a string with maximum 32 characters.

Related information

Name on page 816, in the type *Single*.

6.20.9 Allow Move of User Frame

Parent

Allow Move of User Frame belongs to the type *Mechanical Unit*, in the topic *Motion*.

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

6.20.10 Activate at Start Up

Parent

Activate at Start Up belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

Activate at Start Up defines if the mechanical unit should be activated at start.

Usage

Set the value to Yes to activate the mechanical unit at start.

Allowed values

Yes or *No*.

The default value is *No*.

6.20.11 Deactivation Forbidden

Parent

Deactivation Forbidden belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

Deactivation Forbidden defines if the mechanical unit is allowed to be deactivated.

Usage

Set *Deactivation Forbidden* to Yes if the mechanical unit should be allowed to be deactivated. 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.

The default value for ABB robots is Yes.

Note

The default value is No in case a new external axis is added manually.

6.20.12 Deactivate PTC superv. at disconnect

Parent

Deactivate PTC superv. at disconnect belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

Set the parameter *Deactivate PTC superv. at disconnect* to Yes, to disable the PTC supervision 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 *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.

6.20.13 Activate from any motion task

Parent

Activate from any motion task belongs to the type *Mechanical Unit*, in the topic *Motion*.

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 575*.

The mechanical unit must still belong to a mechanical unit group, see *The Mechanical Unit Group type on page 205*. This configuration determines which task that will control the mechanical unit at start.

Allowed values

Yes

No

The default value is No.

Continues on next page

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.

6.20.14 Act/Deact Only from Rapid

Parent

Act/Deact Only from Rapid belongs to the type *Mechanical Unit*, in the topic *Motion*.

Description

The parameter *Act/Deact Only from Rapid* defines if activation and deactivation of a mechanical unit shall only be allowed from RAPID, that is, the activation state for a mechanical unit shall not change when the RAPID program pointer is moved.

Usage

Set *Act/Deact Only from Rapid* when the activation state shall not be changed when the RAPID program pointer is moved.

The value *Yes* is only used for additional axes.

Allowed values

Yes or No.

Default value is No.

6.21 Type Motion Planner

6.21.1 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.

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

6.21.2 Name

Parent

Name belongs to the type *Motion Planner*, in the topic *Motion*.

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*. For more information, see *The Mechanical Unit Group type on page 205* in the topic *Controller*.

Allowed values

A string with maximum 32 characters.

The name must not be changed.

6.21.3 AbsAcc Speed Adjust

Parent

AbsAcc Speed Adjust belongs to the type *Motion Planner*, in the topic *Motion*.

Description

The parameter *AbsAcc Speed Adjust* makes a robot with AbsAcc, follow the path with the same speed as the corresponding nominal robot. The path based on the the AbsAcc adjusted target, can be longer or shorter than the nominal path, thus results in a different cycle time.

When the parameter is set to *Yes*, the cycle time for a nominal robot and for AbsAcc calibrated robots is much the same.

Allowed values

Yes or *No*.

The default value is *Yes*.

Related information

Application manual - Controller software OmniCore, chapter *Motion performance*.

6.21.4 TCP Linear Max Speed (m/s)

Parent

TCP Linear Max Speed (m/s) belongs to the type *Motion Planner*, in the topic *Motion*.

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.

For more information, see *TCP Reorient Max Speed (deg/s) on page 638*, *Ext. Axis Rotational Max Speed (deg/s) on page 640*, and *Ext. Axis Linear Max Speed (m/s) on page 639*.

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, motset, MaxRobSpeed, and vmax

6.21.5 TCP Reorient Max Speed (deg/s)

Parent

TCP Reorient Max Speed (m/s) belongs to the type *Motion Planner*, in the topic *Motion*.

Description

It defines the maximum reorientation speed (deg/s) in RAPID speeddata vmax.

Usage

The parameter is used to define v_ori in RAPID speeddata vmax. For more information, see *TCP Linear Max Speed (m/s) on page 637*, *Ext. Axis Linear Max Speed (m/s) on page 639*, and *Ext. Axis Rotational Max Speed (deg/s) on page 640*.

Allowed values

A value between 0.01 and 100000.

The default value is 500.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.6 Ext. Axis Linear Max Speed (m/s)

Parent

Ext. Axis Linear Max Speed (m/s) belongs to the type *Motion Planner*, in the topic *Motion*.

Description

It defines the maximum external axis linear speed (m/s) in RAPID speeddata vmax.

Usage

The parameter is used to define v_leax in RAPID speeddata vmax. For more information, see *TCP Linear Max Speed (m/s) on page 637*, *TCP Reorient Max Speed (deg/s) on page 638*, and *Ext. Axis Rotational Max Speed (deg/s) on page 640*.

Allowed values

A value between 0.01 and 339.

The default value is 5.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.7 Ext. Axis Rotational Max Speed (deg/s)

Parent

Ext. Axis Rotational Max Speed (deg/s) belongs to the type *Motion Planner*, in the topic *Motion*.

Description

It defines the maximum external axis reorientation speed (deg/s) in RAPID speeddata vmax.

Usage

The parameter is used to define v_reax in RAPID speeddata vmax. For more information, see *TCP Linear Max Speed (m/s) on page 637*, *TCP Reorient Max Speed (deg/s) on page 638*, and *Ext. Axis Linear Max Speed (m/s) on page 639*.

Allowed values

A value between 0.01 and 100000.

The default value is 1000.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.8 Brake on Time

Parent

Brake on Time belongs to the type *Motion Planner*, in the topic *Motion*.

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

It is necessary that all Mechanical Units in the system has a *Use Brake Relay* defined, else this parameter will have no effect.

Values for *Brake On Timeout* will only be considered for Motion Planners with configured Mechanical Units. This means that for systems without any MultiMove option, it will only be necessary to set *Brake On Timeout* for Motion Planner 1. For MultiMove systems, different values for *Brake On Timeout* can be set and all Motion Planners with configured Mechanical Units must agree that the system should apply the brakes before an order to apply the brakes is executed.

For more information, see *Use Brake Relay on page 623*.

Allowed values

A value between 0.3 to 3,600,000, specifying the time in seconds.

6.21.9 Dynamic Resolution

Parent

Dynamic Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Dynamic Resolution is optimized for the system at delivery. It should normally not be changed.

The dynamic resolution can be reduced from the nominal value 1.0, which reduces the cycle time in some applications. However, a lower dynamic resolution will increase the CPU load.

Limitation

Dynamic Resolution is optimized for the system at delivery. It should normally not be changed.

Allowed values

A value between 0.1667 to 1.00, specifying a factor of the resolution.

6.21.10 Path Resolution

Parent

Path Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Description

The parameter is used for specific applications such as conveyer tracking or synchronization with press equipment.

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:

- Using coordinated interpolation.
- Using Weldguide.
- Using the option *Conveyor Tracking*.

Allowed values

A value between 0.1667 to 6.00, specifying a factor of the resolution.

Additional information

There is also a RAPID instruction named `PathResol` which affects the resolution of the path.

Related information

Technical reference manual - RAPID Overview

Application manual - Controller software OmniCore

6.21.11 Queue Time

Parent

Queue Time belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.12 Teach Mode Max Speed

Parent

Teach Mode Max Speed belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.13 Process Update Time

Parent

Process Update Time belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.14 Prefetch Time

Parent

Prefetch Time belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.15 Event Preset Time

Parent

Event Preset Time belongs to the type *Motion Planner*, in the topic *Motion*.

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.

If *Event Preset Time* is defined, the system input *Limit Speed* should not be used. The actions that need the *Event Preset Time* may be handled incorrectly when the *Limit Speed* signal is set.

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 OmniCore

6.21.16 Restrict Placing of Circlepoints

Parent

Restrict Placing of Circlepoints belongs to the type *Motion Planner*, in the topic *Motion*.

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.

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.

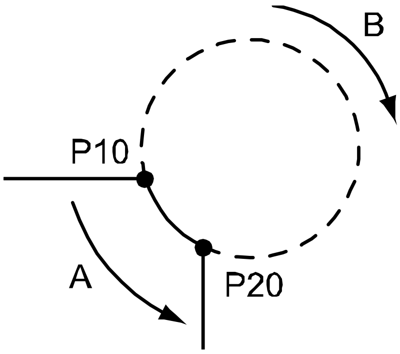
Related information

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 Circlepoints* is set to Yes then an error message is displayed that the TCP is not within safe limits.



xx0800000185

6.21.17 Use Motion Supervision

Parent

Use Motion Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Use Motion Supervision defines which set of motion supervision parameters to be used for this motion planner. For more information, see *The Motion Supervision type on page 688*.

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 OmniCore*, chapter *Collision Detection*.

Allowed values

A string with maximum 32 characters.

6.21.18 Motion Supervision Permanent Off

Parent

Motion Supervision Permanent Off belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Motion Supervision Permanent Off is used to turn off all motion supervision to save CPU power.

Allowed values

Yes

No

6.21.19 Motion Supervision Max Level

Parent

Motion Supervision Max Level belongs to the type *Motion Planner*, in the topic *Motion*.

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. For more information, see *Path Collision Detection Level on page 692*.

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

Application manual - Controller software OmniCore

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%.

6.21.20 Time Event Supervision

Parent

Time Event Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

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

6.21.21 High Interpolation Priority

Parent

High Interpolation Priority belongs to the type *Motion Planner*, in the topic *Motion*.

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. See also *Path Resolution on page 643*.

Note

Using *High Interpolation Priority* might affect the performance of the application, for example, 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

6.21.22 Speed Control Warning

Parent

Speed Control Warning belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.23 Speed Control Percent

Parent

Speed Control Percent belongs to the type *Motion Planner*, in the topic *Motion*.

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).

6.21.24 Interpolation Buffer Startup Adjust

Parent

Interpolation Buffer Startup Adjust belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Interpolation Buffer Startup Adjust defines how to adjust the default value for the interpolation buffer created at start from finepoint.

Usage

Interpolation Buffer Startup Adjust changes the default value by increasing or decreasing the number of steps in the buffer, calculated by the motion planner at start from finepoint.

A value less than zero will decrease the number of steps and this will reduce the time to start from finepoint (see additional information for risks). A value greater than zero will increase the number of steps. This can be used if there are unexpected corner path failures (code 50024) in the first move instruction after a finepoint.

Allowed values

An integer in the range -2 to 2.

Default value is 0 and default number of steps will be used.

Additional information

Reducing the number of steps in the buffer will increase the risk that the robot stops with the corner path failure warning (50024) on the first move instruction after a finepoint. A reduced value can in some cases result in deceleration limit error (50082). In these cases, the value should be increased.

6.21.25 Use Additional Interp. Object Batch

Parent

Use Additional Interp. Object Batch belongs to the type *Motion Planner*, in the topic *Motion*.

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.

6.21.26 Bandwidth of path pose filter

Parent

Bandwidth of path pose filter belongs to the type *Motion Planner*, in the topic *Motion*.

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*.

6.21.27 Number of Internal Event Objects

Parent

Number of Internal Event Objects belongs to the type *Motion Planner*, in the topic *Motion*.

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

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.28 Enable High Accuracy Pos Sync

Parent

Enable High Accuracy Pos Sync belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Enable High Accuracy Pos Sync 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

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.29 Setup Optimized Start from Finepoint

Parent

Setup Optimized Start from Finepoint belongs to the type *Motion Planner*, in the topic *Motion*.

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 instruction `DeactEventBuffer` is used then the optimized start from finepoint functionality is automatically enabled. And if the event buffer is configured and activated using the RAPID instruction `ActEventBuffer`, the optimized start from finepoint functionality is automatically disabled.

Allowed values

Yes or *No*

The default value is Yes.

Related information

Application manual - Additional axes

Technical reference manual - RAPID Instructions, Functions and Data types

6.21.30 Use check point limitation in world

Parent

Use check point limitation in world belongs to the type *Motion Planner*, in the topic *Motion*.

Description

The parameter *Use check point limitation in world* enables the robot to limit check point speed in world coordinate system in teach mode. Thus, the additional speed from a track motion is added to the check point speed and the robot speed is reduced.

Usage

The default value for *Use check point limitation in world* is *No*. This function is especially useful when combining SafeMove with a robot on track. SafeMove supervises the speed of check points in world coordinate system. If this function is not active, there is a risk that SafeMove will trigger overspeed error when robot and track is moved simultaneously. For more information, see *The Arm Check Point type on page 496* and *How to define arm check point on page 452*.

Allowed values

Yes or *No*

Default value

The default value is *Yes*.

Related information

Application manual - Functional safety and SafeMove

6.21.31 Cartesian threshold for short segments

Parent

Cartesian threshold for short segments belongs to the type *Motion Planner*, in the topic *Motion*.

Description

For a movement to be considered short, the TCP movement, the maximum rotating axis movement, and the maximum linear axis movement must simultaneously be smaller than the *Cartesian threshold for short segments*, the *Threshold for short segments in rad*, and the *Threshold for short segments in m*, respectively.

Usage

Cartesian threshold for short segments is used to detect and warn you about extremely short movement instructions. Short movement instructions can lead to problems like high CPU load and events being executed out of order.

Allowed values

A value between 0.0 to 0.1, specifying the distance in meters.

The default value is 0.0001 m.

6.21.32 Threshold for short segments in rad

Parent

Threshold for short segments in rad belongs to the type *Motion Planner*, in the topic *Motion*.

Description

For a movement to be considered short, the TCP movement, the maximum rotating axis movement, and the maximum linear axis movement must simultaneously be smaller than the *Cartesian threshold for short segments*, the *Threshold for short segments in rad*, and the *Threshold for short segments in m*, respectively.

Usage

Threshold for short segments in rad is used to detect and warn you about extremely short movement instructions. Short movement instructions can lead to problems like high CPU load and events being executed out of order.

Allowed values

A value between 0.0 to 0.1, specifying the angle in radians.

The default value is 0.001 radians.

6.21.33 Threshold for short segments in m

Parent

Threshold for short segments in m belongs to the type *Motion Planner*, in the topic *Motion*.

Description

For a movement to be considered short, the TCP movement, the maximum rotating axis movement, and the maximum linear axis movement must simultaneously be smaller than the *Cartesian threshold for short segments*, the *Threshold for short segments in rad*, and the *Threshold for short segments in m*, respectively.

Usage

Threshold for short segments in m is used to detect and warn you about extremely short movement instructions. Short movement instructions can lead to problems like high CPU load and events being executed out of order.

Allowed values

A value between 0.0 to 0.1, specifying the distance in meters.

The default value is 0.0001 m.

6.21.34 Max allowed short segments

Parent

Max allowed short segments belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Max allowed short segments determines the maximum number of short consecutive movement instructions allowed before a warning is displayed. When a non-short movement instruction is executed, the internal counter for short movements is reset to zero.

Usage

This parameter can be increased to allow up to 100 short subsequent movement instructions. Set the value to -1 to stop the internal counter for short movements.

Allowed values

A value between -1 to 100.

The default value is 1.

6.21.35 Maximum allowed path correction

Parent

Maximum allowed path correction belongs to the type *Motion Planner*, in the topic *Motion*.

Description

Maximum allowed path correction defines the maximum allowed path correction for the robot to follow the actual path.

Allowed values

A value between 0.01 to 0.5, specifying the distance in meters.

The default value is 0.05.

6.21.36 Max acc when ramping up speed

Parent

Max acc when ramping up speed belongs to the type *Motion Planner*, in the topic *Motion*.

Description

When deactivating the system input *LimitSpeed*, the parameter *Max acc when ramping up speed* defines an upper limit of the path acceleration that the mechanical unit can have when ramping to the programmed speed. So, when activating *LimitSpeed*, the speed is ramped down as quickly as possible, but when deactivating *LimitSpeed*, the acceleration is limited by this parameter.

Allowed values

Default value is 10 m/s^2.

Related information

System input *Limit Speed on page 246*.

Technical reference manual - RAPID Instructions, Functions and Data types.

6.22 Type Motion Process Mode

6.22.1 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.

> **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.

Available motion process modes

A motion process mode consists of a specific set of tuning parameters for a robot. Each tuning parameter set, that is each mode, optimizes the robot tuning for a specific class of applications.

There following modes are predefined:

- *Optimal cycle time mode* – this mode gives the shortest possible cycle time and is normally the default mode.
- *Accuracy mode* – this mode improves path accuracy. The cycle time will be slightly increased compared to *Optimal cycle time mode*.
- *Low speed accuracy mode* – this mode improves path accuracy. The cycle time will be slightly increased compared to *Accuracy mode*.
- *Low speed stiff mode* - this mode is recommended for contact applications where maximum servo stiffness is important. 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*.
- *Press tending mode* – Changes the *Kv Factor*, *Kp Factor* and *Ti Factor* in order to mitigate tool vibrations. This mode is primarily intended for use in press tending applications where flexible grippers with a large extension in the y-direction are used.
- *Collaborative mode* – This mode is recommended for collaborative applications where robot should run smoothly. The cycle time will be increased compared to optimal cycle time mode. This will only have any effect on GoFa CRB 15000.

There are also four modes available for application specific user tuning:

- *MPM User mode 1 – 4*

Continues on next page

Type description

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 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*.

Limitations

- The *Motion Process Mode* concept is currently available for all six- and seven-axes robots except paint robots with TrueMove1.
- The *Mounting Stiffness Factor* parameters are only available for the following robots:

  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 *Accset* and the geometric accuracy parameters are available.
- The following robot models do not support the use of *World Acc Factor* (i.e. only *World Acc Factor = -1* is allowed):

  IRB 340, IRB 360, IRB 540, IRB 1400, IRB 1410

Related information

Application manual - Controller software OmniCore

Technical reference manual - RAPID Instructions, Functions and Data types

6.22.2 Name

Parent

Name belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

Name defines the name of the motion process mode.

Allowed values

A string with maximum 32 characters.

6.22.3 Use Motion Process Mode Type

Parent

Use Motion Process Mode Type belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

Choice of predefined mode parameters.

Usage

This parameter determines the set of predefined parameters for a user defined mode. The value must be one of the following strings:

- rob1_optimal_cycle_time_mode
- rob1_low_speed_accuracy_mode
- rob1_low_speed_stiff_mode
- rob1_accuracy_mode
- rob1_press_tending_mode
- rob1_collaborative_mode

If the system has multiple robots it is necessary to replace rob1 by rob2, rob3, etc.

Allowed values

- rob1_optimal_cycle_time_mode
- rob1_low_speed_accuracy_mode
- rob1_low_speed_stiff_mode
- rob1_accuracy_mode
- rob1_press_tending_mode
- rob1_collaborative_mode

6.22.4 Accset Acc Factor

Parent

Accset Acc Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.5 Accset Ramp Factor

Parent

Accset Ramp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.6 Accset Fine Point Ramp Factor

Parent

Accset Fine Point Ramp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.7 Dh Factor

Parent

Dh Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.8 Joint Acc Factor 1, 2, 3, 4, 5, 6, 7

Parent

Joint Acc Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

Joint Acc Factor reduces the acceleration for a specific joint.

Usage

Joint Acc Factor = 0.6 reduces the acceleration for a specific joint by 40%. *Joint Acc Factor* can be used to reduce path errors and vibrations caused by the acceleration of specific joints. For example, axis 4-6 during TCP reorientation.

Allowed values

A numeric value between 0.01 and 1.

6.22.9 Joint Speed Max Factor 1, 2, 3, 4, 5, 6, 7

Parent

Joint Speed Max Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

Joint Speed Max Factor reduces the maximum speed for a specific joint.

Usage

The allowed max speed for a specific joint is reduced by a particular percentage depending on the value of the *Joint Speed Max Factor* parameter. For example, if the value of *Joint Speed Max Factor* is 0.6, it reduces the allowed max speed for a specific joint by 40%.

Allowed values

A numeric value between 0.01 and 1.

Related information

Joint Acc Factor 1, 2, 3, 4, 5, 6, 7 on page 679

6.22.10 World Acc Factor

Parent

World Acc Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

A positive value of *World Acc Factor* activates a function that reduces the world acceleration dynamically. Use of *World Acc Factor* decreases path errors and increases the cycle time slightly. However, since the world acceleration reduction is dynamic and depends on the path characteristics, the use of *World Acc Factor* is often a cycle-time efficient way of improving path accuracy, compared to the use of *Accset Acc Factor* or *Accset Ramp Factor*.

Usage

The recommended setting for improving path accuracy is *World Acc Factor* = 1. *World Acc Factor* = -1 deactivates this function. Path accuracy can be further improved, to the cost of longer cycle time, by decreasing the recommended value (for example, = 0.75). Cycle time can be shortened, to the cost of less accuracy, by increasing the recommended value (for example, *World Acc Factor* = 1.5). The use of *World Acc Factor* is recommended for cutting applications and other applications where path accuracy is important.

Limitations

The following robot models do not support the use of *World Acc Factor* (that is, only *World Acc Factor* = -1 is allowed):

IRB 340, IRB 360, IRB 540, IRB 1400, IRB 1410

Allowed values

A numeric value between -1 and 100.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types - `WorldAccLim`.

6.22.11 Geometric Accuracy Factor

Parent

Geometric Accuracy Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

Description

Geometric Accuracy Factor can be used to adjust the geometric accuracy of the path. The final geometric accuracy is the default setting for a specific motion mode multiplied by *Geometric Accuracy Factor*.

Usage

For *Motion Process Mode* = *Accuracy Mode*, the default setting (*Geometric Accuracy Factor* = 1) is recommended. For other modes, the accuracy can be improved by setting *Geometric Accuracy Factor* = 0.1.

Allowed values

A numeric value between 0.1 and 10.

6.22.12 Df Factor 1, 2, 3, 4, 5, 6, 7

Parent

Df Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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.

For more information, see *Mounting Stiffness Factor X, Y, Z on page 687*.

Allowed values

A numeric value between 0.1 and 1.5.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6.22.13 Kp Factor 1, 2, 3, 4, 5, 6, 7

Parent

Kp Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.14 Kv Factor 1, 2, 3, 4, 5, 6, 7

Parent

Kv Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.15 Ti Factor 1, 2, 3, 4, 5, 6, 7

Parent

Ti Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.22.16 Mounting Stiffness Factor X, Y, Z

Parent

Mounting Stiffness Factor belongs to the type *Motion Process Mode*, in the topic *Motion*.

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

6.23 Type Motion Supervision

6.23.1 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.

Type description

The type *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 OmniCore*, 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. For more information, see *How to tune motion supervision on page 458*.

6.23.2 Name

Parent

Name belongs to the type *Motion Supervision*, in the topic *Motion*.

Description

Name defines the name of the motion supervision setup.

Limitation

This parameter cannot be changed.

6.23.3 Path Collision Detection

Parent

Path Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

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

6.23.4 Jog Collision Detection

Parent

Jog Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

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

6.23.5 Path Collision Detection Level

Parent

Path Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

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 OmniCore* and *How to tune motion supervision on page 458*.

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%.

6.23.6 Jog Collision Detection Level

Parent

Jog Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

Description

Jog Collision Detection Level modifies the supervision level for the collision detection for jogging by a specified percentage value.

For more information, see *How to tune motion supervision on page 458*.

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 OmniCore*.

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%.

6.23.7 Collision Detection Memory

Parent

Collision Detection Memory belongs to the type *Motion Supervision*, in the topic *Motion*.

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 movement of robot 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 OmniCore*.

Allowed values

A value in the interval 0 to 0.5, specifying the movement in seconds.

For the IRB 14050 and CRB 15000 robots the default value is 0 s and hence the robot does not back off.

Setting the value to 0 s (disabling backing after collision) may leave the robot in a state with residual forces remaining after a collision. This could trigger new collisions when trying to move away from that position. To move away robustly after a collision, the following are some of the recommended solutions:

- Enable lead-through for a short period of time to release the tension.
- Set the value of `MotionSup\` to `Off` before executing the `move` instructions.
- Use `ContactL` instead of `MoveL`.

Related information

Application manual - Controller software OmniCore

6.23.8 Manipulator supervision

Parent

Manipulator supervision belongs to the type *Motion Supervision*, in the topic *Motion*.

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

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 On.

Related information

Application manual - Controller software OmniCore

6.23.9 Manipulator supervision level

Parent

Manipulator supervision level belongs to the type *Motion Supervision*, in the topic *Motion*.

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 OmniCore

6.23.10 Collision detection at standstill

Parent

Collision detection at standstill belongs to the type *Motion Supervision*, in the topic *Motion*.

Description

The parameter *Collision detection at standstill* enables the detection of any collision, even at standstill, when the value is set to TRUE.

Allowed values

TRUE or FALSE.

Default value is FALSE. The default value for the IRB 14050/CRB 15000 robot is TRUE.

6.23.11 Collision Detection Zero Speed Time

Parent

Collision Detection Zero Speed Time belongs to the type *Motion Supervision*, in the topic *Motion*.

Description

Collision Detection Zero Speed Time modifies the wait time after a collision. This is needed to make sure that the Robot is standing still before backing away.

Usage

If Motion Supervision is used in time critical applications and when the collisions are typically happening at low speeds with the tools that are not flexible, the value of *Collision Detection Zero Speed Time* can be reduced. Otherwise, it is not recommended to change this parameter since a low value of *Collision Detection Zero Speed Time* can cause additional errors when backing away from the collision.

The main effect of this parameter is seen if the Motion system parameter *Ind collision stop without brake*" is set to ON since then there is no need to apply and release the brakes.

Limitations

Changes in this parameter affect the system only if the Collision detection option is installed.

Allowed values

A value in the interval 50 ms to 1000 ms. The default value is 400 ms.

6.24 Type Motion System

6.24.1 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.

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.

6.24.2 Name

Parent

Name belongs to the type *Motion System*, in the topic *Motion*.

Description

Name specifies the name of the *Motion System* type.

Allowed values

A string with maximum 32 characters.

6.24.3 Min Ambient Temperature Cabinet

Parent

Min Ambient Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Description

Min Ambient 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.

6.24.4 Max Ambient Temperature Cabinet

Parent

Max Ambient Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Description

Max Ambient 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.

6.24.5 Min Ambient Temperature Robot

Parent

Min Ambient Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Description

Min Ambient 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.

6.24.6 Max Ambient Temperature Robot

Parent

Max Ambient Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Description

Max Ambient 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.

6.24.7 Coll-Pred Safety Distance

Parent

Coll-Pred Safety Distance belongs to the type *Motion System*, in the topic *Motion*.

Description

The function *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* determines at what distance the two objects are considered to be in collision.

The geometric model for the robot is integrated in RobotWare. The geometric models for external or surrounding equipment are set up in RobotStudio.

The functionality is activated by a system input, see *Collision Avoidance on page 241*.

Allowed values

A value between 0.001 and 1 meters.

The default value is 0.001.

Limitation

Coll-Pred Safety Distance is only activated together with the option *Collision Avoidance*.

6.24.8 Ind collision stop without brake

Parent

Ind collision stop without brake belongs to the type *Motion System*, in the topic *Motion*.

Description

The parameter *Ind collision stop without brake* is used to set that detected collisions can be handled independently in RAPID tasks that are executed independently. For example, if ROB_1 and ROB_2 are running in independent RAPID tasks and ROB_2 detects a collision, only the RAPID task for ROB_2 is stopped.

The independent stop does not use the physical brake.

Usage

The main usage for *Ind collision stop without brake* is for MultiMove configurations but when used for a non-MultiMove configuration the stop method without physical brake will be used anyway.

Prerequisites

This requires the option *Collision Detection* in the system.

Limitations

The collision stop without brake is slightly slower than when using the physical brakes. This can cause the robot to get stuck against a fixed object after a collision which in turn can trigger other supervision functions. To avoid that the robot gets stuck after a collision, the value of the system parameter *Collision Detection Memory* can be increased, see *Collision Detection Memory on page 694*.

Allowed values

TRUE or FALSE.

Default value is FALSE.

Related information

Collision Detection Memory on page 694

6.24.9 Disable SafeMove Assistance

Parent

Disable SafeMove Assistance belongs to the type *Motion System*, in the topic *Motion*.

Description

The parameter *Disable SafeMove Assistance* is used to turn off the functionality *SafeMove Assistant*.

Use the parameter *SafeMove assistance speed factor* to set the speed reduction factor.

The functionality *SafeMove Assistant* is only active in automatic mode.

Allowed values

Yes or No.

The default value is *No*, meaning that *SafeMove Assistant* is enabled.

Related information

SafeMove assistance speed factor on page 708

6.24.10 SafeMove assistance speed factor

Parent

SafeMove assistance speed factor belongs to the type *Motion System*, in the topic *Motion*.

Description

If the robot has a minor overshoot or in any other way triggers a SafeMove speed violation, the parameter *SafeMove assistance speed factor* can be reduced to avoid unnecessary violations. The default setting of 0.96 corresponds to that the path planner will use 96% of the speed limit in the active safety configuration.

Avoid programming movement with different speed data at the exact border of a safety zone.

Allowed values

A numerical value between 0 and 1.

Default value is 0.96.

6.24.11 SafeMove assistance zone margin

Parent

SafeMove assistance zone margin belongs to the type *Motion System*, in the topic *Motion*.

Description

The parameter *SafeMove assistance zone margin* is used as a margin distance, in meters, to avoid triggering speed violations in SafeMove. It determines roughly how far before a speed-limited zone the robot will have reached the actual speed limit. Likewise, when the robot is moving out of a speed-limited zone, it will not start accelerating towards the programmed speed until it has moved this distance outside the zone.

Allowed values

A numerical value between 0 and 1.

Default value is 0.01.

6.25 Type Motor

6.25.1 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.

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.

6.25.2 Name

Parent

Name belongs to the type *Motor*, in the topic *Motion*.

Description

Name defines the name of the motor.

Allowed values

A string with maximum 32 characters.

6.25.3 Use Motor Type

Parent

Use Motor Type belongs to the type *Motor*, in the topic *Motion*.

Description

Use Motor Type defines which type of motor is used for this type.

Usage

The type *Motor Type* defines the motor data. For more information, see *The type Motor Type on page 723*.

Allowed values

A string with maximum 32 characters.

6.25.4 Use Motor Calibration

Parent

Use Motor Calibration belongs to the type *Motor*, in the topic *Motion*.

Description

Use Motor Calibration defines which type of motor calibration to be used.

Usage

The type *Motor Calibration* defines the motor's calibration data. For more information, see *The Motor Calibration type on page 714*.

Allowed values

A string with maximum 32 characters.

6.26 Type Motor Calibration

6.26.1 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.

Type description

With the parameters in the *Motor Calibration* type, you can calibrate the robot's motors by entering the calibration values.

The robot is calibrated on delivery. If needed, the motor calibration configuration is 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.

6.26.2 Name

Parent

Name belongs to the type *Motor Calibration*, in the topic *Motion*.

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.

6.26.3 Commutator Offset

Parent

Commutator Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

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.

6.26.4 Commutator Offset Valid

Parent

Commutator Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Description

Commutator Offset Valid specifies whether the commutator offset is defined or not. For more information, see *Commutator Offset on page 716*.

Allowed values

Yes or No.

6.26.5 Calibration Offset

Parent

Calibration Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

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.

6.26.6 Calibration Offset Valid

Parent

Calibration Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Description

Calibration Offset Valid specifies whether the calibration offset is defined or not.

For more information, see *Calibration Offset on page 718*.

Allowed values

Yes or No.

6.26.7 Calibration Sensor Position

Parent

Calibration Sensor Position belongs to the type *Motor Calibration*, in the topic *Motion*.

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.

6.26.8 Factory Calibration Method

Overview

Factory Calibration Method belongs to the type *Motor Calibration*, in the topic *Motion*.

Description

The factory calibration is done when the robot is calibrated in the factory, before delivery from ABB.

Using the values in the below table you can determine which method was used to calibrate the robot's motor.

Value	Calibration method
0	Undefined
1	Manual calibration
2	Calibration Pendulum
3	Axis Calibration
4	Axis Calibration (YuMi)
5	Default calibration (GoFa)

Usage

This parameter cannot be modified.

6.26.9 Latest Calibration Method

Overview

Latest Calibration Method belongs to the type *Motor Calibration*, in the topic *Motion*.

Description

The parameter *Latest Calibration Method* defines what method that was last used to calibrate the motors of the robot.

Value	Calibration method
0	Undefined
1	Manual calibration
2	Calibration Pendulum
3	Axis Calibration
4	Axis Calibration (YuMi)
5	Default calibration (GoFa)

Usage

This parameter cannot be modified.

6.27 Type Motor Type

6.27.1 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.

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.

6.27.2 Name

Parent

Name belongs to the type *Motor Type*, in the topic *Motion*.

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.

6.27.3 Pole Pairs

Parent

Pole Pairs belongs to the type *Motor Type*, in the topic *Motion*.

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 are visible for robot motors, but cannot be changed.

Allowed values

An integer between 0 and 20.

6.27.4 Inertia (kgm**2)

Parent

*Inertia (kgm**2)* belongs to the type *Motor Type*, in the topic *Motion*.

Description

Motor and resolver inertia on motor side. The unit is kgm^2.

Usage

For a rotating object, the inertia describes the tendency to resist a change in rotational speed (corresponding to mass for an object moving linearly). For a motor, the inertia depends on the mass and the mass distribution of the rotor. The value of inertia is used for advanced servo control and can be found in the motor specification.

Allowed values

A value between 0 and 10.

The default value is 0.

6.27.5 Stall Torque (Nm)

Parent

Stall Torque (Nm) belongs to the type *Motor Type*, in the topic *Motion*.

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 (Nm)* to the stall torque (T_0) specified by the motor manufacturer.

Limitations

Stall Torque (Nm) can only be changed for additional axis motors. The values are visible for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100000 Nm.

6.27.6 ke Phase to Phase (Vs/rad)

Parent

ke Phase to Phase (Vs/rad) belongs to the type *Motor Type*, in the topic *Motion*.

Description

Nominal voltage constant.

Usage

ke Phase to Phase (Vs/rad) is the induced voltage (phase to phase) that corresponds to the speed 1 rad/s.

Limitations

ke Phase to Phase (Vs/rad) can only be changed for additional axis motors. The values are visiblie 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:

$$ke = kt/\sqrt{3}$$

6.27.7 Max Current (A rms)

Parent

Max Current (A rms) belongs to the type *Motor Type*, in the topic *Motion*.

Description

Max current without irreversible magnetization.

Usage

Set *Max Current (A rms)* to the root-mean-square of the maximum current the motor can withstand without irreversible demagnetization.

Limitations

Max Current (A rms) can only be changed for additional axis motors. The values are visible for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (A rms).

6.27.8 Phase Resistance (ohm)

Parent

Phase Resistance (ohm) belongs to the type *Motor Type*, in the topic *Motion*.

Description

Nominal winding resistance per phase at 20 degrees Celsius.

Usage

Set *Phase Resistance (ohm)* 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 are visible for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 ohm.

6.27.9 Phase Inductance (H)

Parent

Phase Inductance (H) belongs to the type *Motor Type*, in the topic *Motion*.

Description

Nominal winding inductance per phase at zero current.

Usage

Set *Phase Inductance (H)* to the stator phase inductance (L_0) specified by the motor manufacturer. The value should be measured at a frequency of about 120 Hz to correspond to what the drive expects. If the inductance is measured phase to phase the value is divided by 2.

Limitations

Phase Inductance (H) can only be changed for additional axis motors. The values are visible for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 H.

6.28 Type Process

6.28.1 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.

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. For more information, see *Use Process on page 577* and *The Linked M Process type on page 602*.

6.28.2 Name

Parent

Name belongs to the type *Process*, in the topic *Motion*.

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.

6.28.3 Use SG Process

Parent

Use SG Process belongs to the type *Process*, in the topic *Motion*.

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.

6.28.4 Use Linked Motor Process

Parent

Use Linked Motor Process belongs to the type *Process*, in the topic *Motion*.

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.

6.29 Type Relay

6.29.1 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.

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

6.29.2 Name

Parent

Name belongs to the type *Relay*, in the topic *Motion*.

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.

6.29.3 Output Signal

Parent

Output Signal belongs to the type *Relay* in the topic *Motion*.

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*. For more information, see *The Signal type on page 393*.

Allowed values

A string with maximum 32 characters.

6.29.4 Input Signal

Parent

Input Signal belongs to the type *Relay* in the topic *Motion*.

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".

For more information, see *The Signal type on page 393*.

Allowed values

A string with maximum 32 characters.

6.30 Type Robot

6.30.1 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.

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.

A maximum of 8 instances of the type *Robot* can be configured in a system.

6.30.2 Name

Parent

Name belongs to the type *Robot*, in the topic *Motion*.

Description

Name defines the name of the robot.

Limitations

This parameter cannot be changed.

6.30.3 Use Robot Type

Parent

Use Robot Type belongs to the type *Robot*, in the topic *Motion*.

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.

6.30.4 Use Old SMB

Parent

Use Old SMB belongs to the type *Robot*, in the topic *Motion*.

Description

This parameter is not used.

6.30.5 Use Robot Calibration

Parent

Use Robot Calibration belongs to the type *Robot*, in the topic *Motion*.

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_un-calib	not_used_un-calib	not_used_un-calib	not_used_un-calib	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 OmniCore*.

6.30.6 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*.

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*. For more information, see *The Joint type on page 573*.

Allowed values

A string with maximum 32 characters, specifying an already defined joint.

6.30.7 Base Frame x, y, z

Parent

Base Frame x, *Base Frame y*, and *Base Frame z* belong to the type *Robot*, in the topic *Motion*.

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).

For more information, see *How to define base frame on page 448*.

Allowed values

A value between -1000 and 1000, specifying the relation in meters.

6.30.8 Base Frame q1, q2, q3, q4

Parent

Base Frame q1, *Base Frame q2*, *Base Frame q3*, and *Base Frame q4* belong to the type *Robot*, in the topic *Motion*.

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.

For more information, see *How to define base frame on page 448*.

Allowed values

A value between -1 and 1 specifying the orientation.

6.30.9 Base Frame Moved by

Parent

Base Frame Moved by belongs to the type *Robot*, in the topic *Motion*.

Description

Base Frame Moved by defines the name of robot or single that moves the base frame of the robot. For more information, see *How to define base frame on page 448*.

Note

When a robot is coordinated with 3rd party track, the performance of the track will be reduced to match limitations from the robot.

Allowed values

A string with maximum 32 characters.

6.30.10 Gravity Alpha

Parent

Gravity Alpha belongs to the type *Robot*, in the topic *Motion*.

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. For more information, see *How to define gravity on page 449*.

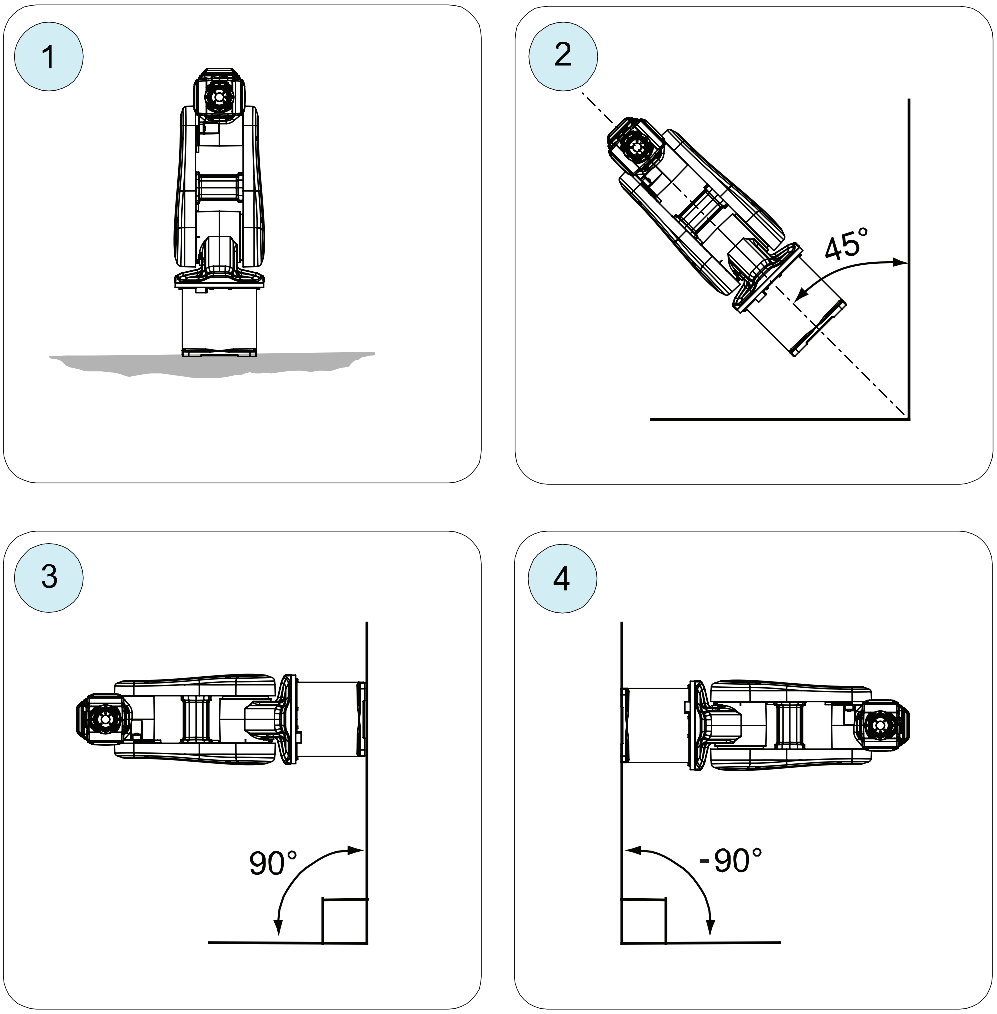
If the robot is mounted on a wall (rotated around the X-axis) the robot base frame and *Gravity Alpha* needs to be changed to reflect the installation. *Gravity Alpha* should then be ± π/2 (1.570796). For more information about base frame, see *How to define base frame on page 448*.

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 mounting)	0
2	45° (tilted mounting)	0.785398
3	90° (wall mounting)	1.570796
4	-90° (wall mounting)	-1.570796

Note

For suspended robots (180°), use *Gravity Beta* instead of *Gravity Alpha*, see *Gravity Beta on page 752*.

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 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.

For more information, see *Upper Joint Bound on page 471* and *Lower Joint Bound on page 472*.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

Additional information

The value for *Gravity Alpha* can be changed in runtime (without restart) with the RAPID instruction `WriteCfgData`. See *Technical reference manual - RAPID Instructions, Functions and Data types*.

Note

To be able to change *Gravity Alpha* in runtime, the initial value of *Gravity Alpha* at startup must be < > 0 (not zero).

Values smaller than 0.0001 at startup is rounded off to zero and voids the ability to change the value in runtime.

6.30.11 Gravity Beta

Parent

Gravity Beta belongs to the type *Robot*, in the topic *Motion*.

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. For more information, see *How to define gravity on page 449*.

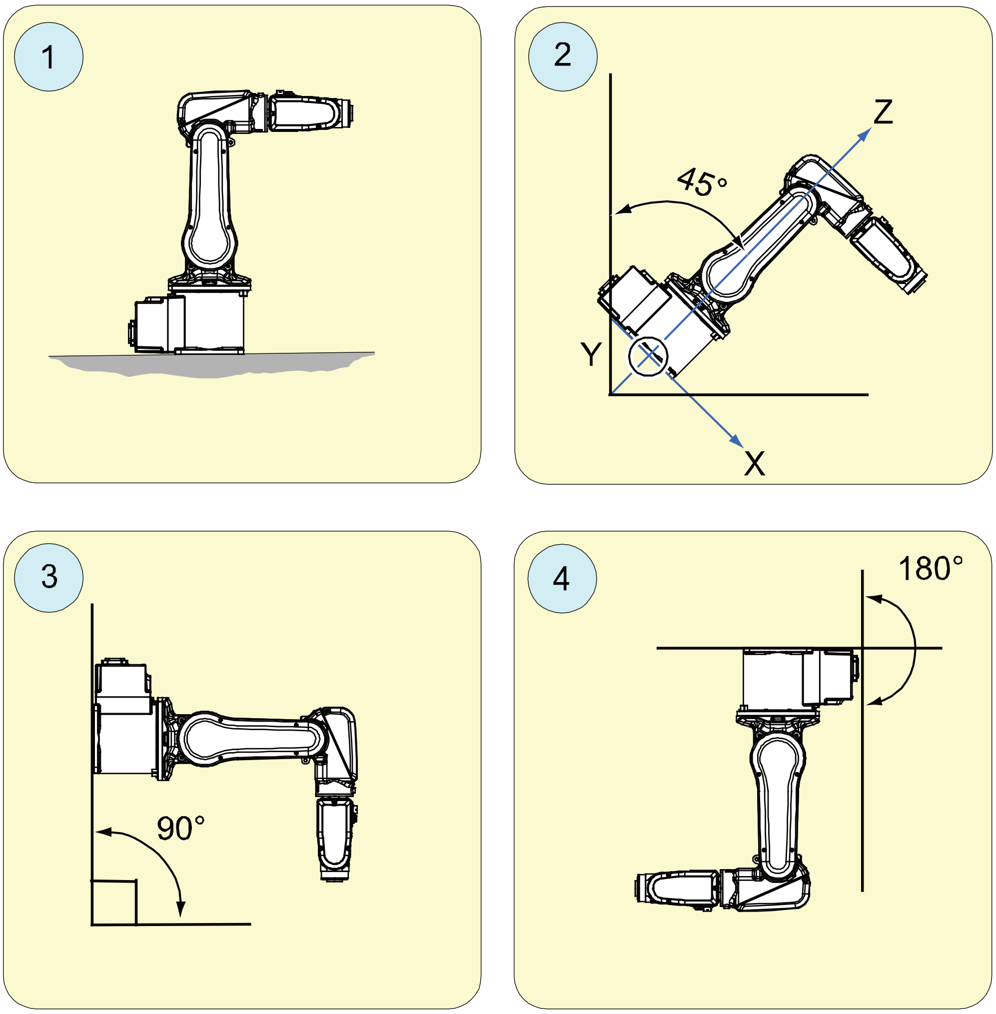
If the robot mounted upside down or on a wall (rotated around the Y-axis) the robot base frame and *Gravity Beta* needs to be changed to reflect the installation. *Gravity Beta* should be π(3.141593) if mounted upside down and ± π/2(1.570796) if mounted on a wall. For more information about base frame refer to *How to define base frame on page 448*.

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
1	0° (floor mounting)	0
2	45° (tilted mounting)	0.785398
3	90° (wall mounting)	1.570796
4	180° (suspended mounting)	3.141593

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

Additional information

The value for *Gravity Beta* can be changed in runtime (without restart) with the RAPID instruction `WriteCfgData`. See *Technical reference manual - RAPID Instructions, Functions and Data types*.

6.30.12 Gamma Rotation

Parent

Gamma Rotation belongs to the type *Robot*, in the topic *Motion*.

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.

6.30.13 Arm Check Point Speed Limit

Parent

Arm Check Point Speed Limit belongs to the type *Robot*, in the topic *Motion*.

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
- IRB 14050

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	0.75 m/s
IRB 14050	0.75 m/s

Allowed values

A number between 0.1 and 20.

The default value is 0.75.

6.30.14 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*.

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.

For more information, see *How to restrict the work area for parallel arm robots on page 451* and *How to define base frame on page 448*.

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. For more information, see *Lower Work Area x, y, z on page 757*.

6.30.15 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*.

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.

For more information, see *How to restrict the work area for parallel arm robots on page 451* and *How to define base frame on page 448*.

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. For more information, see *Upper Work Area x, y, z on page 756*.

6.30.16 Use Motion Process Mode

Parent

Use Motion Process Mode belongs to the type *Robot*, in the topic *Motion*.

Description

Use Motion Process Mode defines the choice of motion process mode that is used for the robot. For more information, see *Use Motion Process Mode Type on page 674*.

Allowed values

A string with maximum 32 characters.

6.30.17 Upper Check Point Bound x, y, z

Parent

Upper Check Point Bound x, *Upper Check Point Bound y*, and *Upper Check Point Bound z* belong to the type *Robot*, in the topic *Motion*.

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. For more information, see *How to define arm check point on page 452*.

Allowed values

A numeric value higher than the respective coordinate *Lower Check Point Bound* in meters. For more information, see *Lower Check Point Bound x, y, z on page 760*.

Related information

Check Point Bound Limit Outside Cube on page 761

6.30.18 Lower Check Point Bound x, y, z

Parent

Lower Check Point Bound x, *Lower Check Point Bound y*, and *Lower Check Point Bound z* belong to the type *Robot*, in the topic *Motion*.

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. For more information, see *How to define arm check point on page 452*.

Allowed values

A numeric value lower than the respective coordinate *Upper Check Point Bound* in meters. For more information, see *Upper Check Point Bound x, y, z on page 759*.

Related information

Check Point Bound Limit Outside Cube on page 761

6.30.19 Check Point Bound Limit Outside Cube

Parent

Check Point Bound Limit Outside Cube belongs to the type *Robot*, in the topic *Motion*.

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 and supervised in robot base coordinates, by an upper and a lower coordinate (or corner). 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.

For applications with a 7th axis, the base frame of the robot could be moved by the 7th axis. This will pose a limitation on using arm check points for avoiding fixed objects along the direction of the additional axis.

If the parameter is *Yes*, then the check point is limited to stay outside the cube. If the parameter is *No*, then the check point is limited to stay inside the cube.

Prerequisites

The arm check point bounds must be configured before setting *Check Point Bound Limit Outside Cube*. For more information, see *How to define arm check point on page 452*.

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

Upper Check Point Bound x, y, z on page 759

Lower Check Point Bound x, y, z on page 760

6.30.20 Track Conveyor with Robot

Parent

Track Conveyor with Robot belongs to the type *Robot*, in the topic *Motion*.

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

6.30.21 Max External Pos Adjustment

Parent

Max External Pos Adjustment belongs to the type *Robot*, in the topic *Motion*.

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

A value between 0.1 and 0.8.

The default value is 0.2.

6.30.22 Time to Inposition

Parent

Time to Inposition belongs to the type *Robot*, in the topic *Motion*.

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

6.30.23 Orientation Supervision Off

Parent

Orientation Supervision Off belongs to the type *Robot*, in the topic *Motion*.

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

6.30.24 Mech.Unit Not Moving Detection Level

Parent

Mech.Unit Not Moving Detection Level belongs to the type *Robot*, in the topic *Motion*.

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*. For more information, see *Mechanical Unit Not Moving on page 298* and *Mech.Unit Not Moving Detection Level on page 824*.

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.

6.30.25 LoadIdentify test-speed

Parent

LoadIdentify test-speed belongs to the type *Robot*, in the topic *Motion*.

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.

6.30.26 Encoder high temp shall generate error

Parent

Encoder high temp shall generate error belongs to the type *Robot*, in the topic *Motion*.

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

6.30.27 Global Speed Limit

Parent

Global Speed Limit belongs to the type *Robot*, in the topic *Motion*.

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
- IRB 14050

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 and IRB 14050	1.5 m/s

Allowed values

A number between 0.1 and 20.

The default value is 20.

6.30.28 Arm-Angle Reference Direction

Parent

Arm-Angle Reference Direction belongs to the type *Robot*, in the topic *Motion*.

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 origin of the robot and being parallel with the Y-axis of the world frame. When the WCP 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 WCP 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

Product manual for the robot.

6.30.29 Limit avoidance distance

Parent

Limit avoidance distance belongs to the type *Robot*, in the topic *Motion*.

Description

Limit avoidance distance controls the distance to the nearest singularity or joint limit when automatically adjusting the arm-angle.

Usage

The singularities that can be handled are where axis 2 or axis 5 is equal to zero.

Allowed values

A value between -1 to 100 radians.

The default value is 0.017453 radians.

Setting a negative value will disable the functionality.

Related information

Product manual for the robot.

6.30.30 Friction comp. lead through factor

Parent

Friction comp. lead through factor belongs to the type *Robot*, in the topic *Motion*.

Description

Friction comp. lead through factor determines how soft a robot should be in lead through mode.

Usage

A higher value makes the robot softer in lead through mode and a lower value makes the robot less soft.

Setting a high value can make the robot sensitive to errors such as wrong payload in the tool definition. The robot can then start to drift by itself.

Setting the value to 0 removes all friction compensation in lead through mode.

Note

This parameter does not need a reboot to apply the changes. Hence the tests of different levels can be done directly after changing the parameter value.

Limitations

Friction comp. lead through factor is only used for the following robots:

- IRB 14000
- IRB 14050

Configuring this parameter in any other robot will not have any effect.

Allowed values

A value between 0.0 and 1.0.

Default value is 0.6.

6.30.31 Use cfx in robtargets for P-rod robots

Parent

Use cfx in robtargets for P-rod robots belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Use cfx in robtargets for P-rod robots* can be used to enable use of configuration *cfx* in the axis configuration definition on parallel rod robots.

Usage

Set *Use cfx in robtargets for P-rod robots* to *Yes* to have the same configuration data as serial link robots.

Allowed values

Yes or No.

Default value is Yes.

Related information

See the datatype `confdata`, *Technical reference manual - RAPID Instructions, Functions and Data types*.

6.30.32 Lead through stiffness scale

Parent

Lead through stiffness scale belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Lead through stiffness scale* can be used to tune the overall lead-through stiffness experience.

Usage

Decreasing the value of *Lead through stiffness scale* will make the lead-through movement more fluid but more sensitive to speed supervision.

Limitations

Lead through stiffness scale is only used for the following robots:

- CRB 15000

Setting this parameter for any other robot will not have any effect.

Allowed values

A value between 0.01 and 1.

Default value 1.

Related information

Lead through stability margin on page 492

6.30.33 Lead through load compensation

Parent

Lead through load compensation belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Lead through load compensation* is used to determine the type of load compensation during lead through.

Usage

If the value is *Default*, the load compensation is based on the torques measured during activation of lead through.

If the value is *Always*, the load compensation is continuously updated based on the measured joint torques. This mode can be used to minimize drift caused by cables and other disturbances affecting the robot arm. The parameter *Lead through load compensation deadband* in the type *Arm* can be used to tune the behavior for each robot joint.

If the value is *Never*, no load compensation is used. The delay when activating lead through is minimized, but the robot becomes more sensitive to drift caused by calibration errors and incorrectly defined loads.

Note

This parameter does not need a restart of the controller to apply the changes. Hence, it is possible to test directly after changing the parameter value.

Limitations

Lead through load compensation is only used for the following robots:

- CRB 15000

Setting this parameter for any other robot will not have any effect.

The parameter is only used when *Lead-through Mode* is set to *Axis 1-6*.

Allowed values

Default, Always, or Never.

Default value is Default.

Related information

Lead through load compensation deadband on page 493

6.30.34 Enable orientation correction

Parent

Enable orientation correction belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Enable orientation correction* is used to correct the orientation of a 5-axis delta robot, to make it possible to reach positions that are hard to program, for example, inside boxes or bins.

Usage

Setting the value to *Yes* will allow the robot to use internal correction so that it can reach a pose, even if the programmed orientation cannot be reached with a 5-axis robot.

Limitations

Enable orientation correction is only used for 5-axis delta robots.

Setting this parameter for any other robot will not have any effect.

Allowed values

Yes or No

Default value is No.

6.30.35 Robot weight in power sharing

Parent

Robot weight in power sharing belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Robot weight in power sharing* is used to decide how power should be distributed between robots and singles in case the power supply is a limiting factor.

Usage

If a movement with both a robot and a positioner needs to be limited due to power supply then the robot's part of the total power will be decided by this parameter.

If robot performance is prioritized then *Robot weight in power sharing* should be increased and vice versa if single performance is preferred.

Example 1

If the power supply has 10 kW with a robot on a track motion and the total required power for a movement would be, for the robot 10 kW and track 2 kW. Then the parameters can be set as follows:

- *Robot weight in power sharing*: 80
- *Single weight in power sharing*: 20

In this case the distribution of the available power would be 8 kW for robot and 2 kW for track.

Example 2

If the power supply has 10 kW with a robot with 3 additional axis and the total required power for a movement would be, for the robot 10 kW and for each single 2 kW. Then the

parameters can be set as follows:

- *Robot weight in power sharing*: 90
- *Single weight in power sharing*: 10

Then actual sharing for each unit would be:

Robot 90/(90+10+10+10) = 75

Single 10/(90+10+10+10) = 8.3

In this case, the distribution of the available power would be 7.5 kW for robot and 0.83 kW for each single.

Note

Note that 0.83 kW might not be enough and another single weight should be considered.

Allowed values

A value between 0 and 1000000.

Default value is 90.

Continues on next page

Additional information

This parameter does not need a restart of the controller to apply the changes. Hence, it is possible to test directly after changing the parameter value.

Related information

Single weight in power sharing on page 826

6.30.36 Ultra Accuracy

Parent

Ultra Accuracy belongs to the type *Robot*, in the topic *Motion*.

Description

The parameter *Ultra Accuracy* enables the ultra accuracy mode for GoFa CRB 15000.

Usage

Setting the value to *Active* will make robot run with ultra accuracy mode.

Limitations

The ultra accuracy mode is only used for GoFa CRB 15000 and requires the option 3101-10.

Setting this parameter for any other robot will not have any effect.

Allowed values

Active or *Inactive*.

The default value is *Inactive*.

Related information

Ultra Accuracy stability margin on page 495

6.31 Type Robot Serial Number

6.31.1 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.

Type description

The type *Robot Serial Number* contains parameters that define the robot's serial number.

6.31.2 Name

Parent

Name belongs to the type *Robot Serial Number*, in the topic *Motion*.

Description

Name specifies the name of the robot that the serial number belongs to.

Allowed values

A string with maximum 32 characters.

6.31.3 Robot Serial Number High Part

Parent

Robot Serial Number High Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

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.

6.31.4 Robot Serial Number Low Part

Parent

Robot Serial Number Low Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

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.

6.32 Type SG Process

6.32.1 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.

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 1-5 and *Motor Torque 1-5* 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 5 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

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, 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.

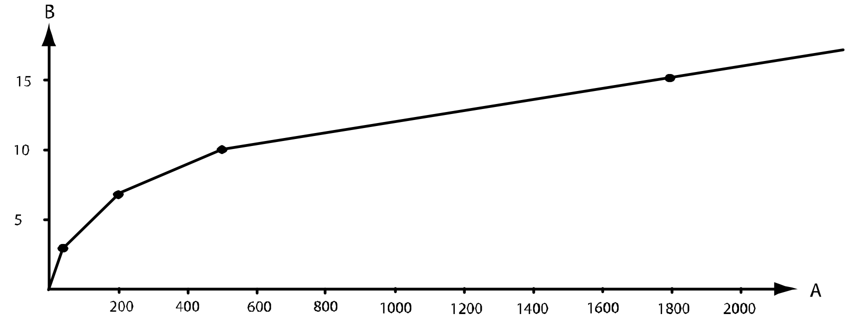
These parameters and values are configured:

Parameter:	Value:
Number of Stored Forces	4
Tip Force 1	50
Tip Force 2	200

Continues on next page

Parameter:	Value:
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)

6.32.2 Name

Parent

Name belongs to the type *SG Process*, in the topic *Motion*.

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.

6.32.3 Serial Number

Parent

Serial Number belongs to the type *SG Process*, in the topic *Motion*.

Description

This is an identification label or serial number that can be used by the manufacturer to identify each servo gun.

Allowed values

A string with maximum 32 characters.

6.32.4 Use Force Master

Parent

Use Force Master belongs to the type *SG Process*, in the topic *Motion*.

Description

Use Force Master determines which *Force Master* should be used. For more information, see *The Force Master type on page 529*.

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.

6.32.5 Close Time Adjust.

Parent

Close Time Adjust. belongs to the type *SG Process*, in the topic *Motion*.

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).

6.32.6 Close Position Adjust.

Parent

Close Position Adjust. belongs to the type *SG Process*, in the topic *Motion*.

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).

6.32.7 Force Ready Delay

Parent

Force Ready Delay belongs to the type *SG Process*, in the topic *Motion*.

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).

6.32.8 Max Force Control Motor Torque

Parent

Max Force Control Motor Torque belongs to the type *SG Process*, in the topic *Motion*.

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.

6.32.9 Post-synchronization Time

Parent

Post-synchronization Time belongs to the type *SG Process*, in the topic *Motion*.

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).

6.32.10 Calibration Mode

Parent

Calibration Mode belongs to the type *SG Process*, in the topic *Motion*.

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.

6.32.11 Calibration Force High

Parent

Calibration Force High belongs to the type *SG Process*, in the topic *Motion*.

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.

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*.

6.32.12 Calibration Force Low

Parent

Calibration Force Low belongs to the type *SG Process*, in the topic *Motion*.

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.

6.32.13 Calibration Time

Parent

Calibration Time belongs to the type *SG Process*, in the topic *Motion*.

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.

6.32.14 Calibration High Force Priority

Parent

Calibration High Force Priority belongs to the type *SG Process*, in the topic *Motion*.

Description

Calibration High Force Priority defines if the high force in the calibration sequence is prioritized. This will make the calibration start with the higher force.

In combination with *Calibration Full Sequence Freq.*, it is possible to choose only the high force in the sequence.

Usage

Set to YES to have the high force have priority

Limitations

Calibration High Force Priority can only be used if you have servo tools.

Allowed values

Yes or No.

Default value is No.

Related information

Calibration Force High on page 795

Calibration Full Sequence Freq. on page 799

6.32.15 Calibration Full Sequence Freq.

Parent

Calibration Full Sequence Freq. belongs to the type *SG Process*, in the topic *Motion*.

Description

Calibration Full Sequence Freq. is used to set how often the gun should use multiple closing at 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 Full Sequence Freq. can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 1000.

0 means gun will always close with a single closing except for the first closing after a tip change.

1 means gun will always close with a multiple closing always.

2 means gun will always close with a multiple every second time.

3 means gun will always close with a multiple every third time, etc.

Default value is 1.

Related information

Calibration Force High on page 795 and *Calibration Force Low on page 796*.

6.32.16 Calibration No Pos Update

Parent

Calibration No Pos Update belongs to the type *SG Process*, in the topic *Motion*.

Description

Calibration No Pos Update will prevent the axis position of the gun to be updated at servo tool calibrations. As a consequence, opened gun arm locations are not affected by tip wear changes. On the other hand, gun closing distances increases with tip wear growth.

Usage

Set to YES to prevent update of gun position.

Limitations

Calibration No Pos Update can only be used if you have servo tools.

Allowed values

Yes or No.

Default value is No.

6.32.17 Number of Stored Forces

Parent

Number of Stored Forces belongs to the type *SG Process*, in the topic *Motion*.

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 784*).

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.

6.32.18 Soft Stop Timeout

Parent

Soft Stop Timeout belongs to the type *SG Process*, in the topic *Motion*.

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 100000 (seconds).

The default value is 0.3.

6.32.19 Brake release delay

Parent

Brake release delay belongs to the type *SG Process*, in the topic *Motion*.

Description

Servo Tool independent restart movement will be delayed until this time has passed after control on.

Usage

Set *Brake release delay* to the desired delay value.

Limitations

Brake release delay can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 100000 (seconds).

The default value is 0.1.

6.32.20 Tip Force 1, 2, 3, 4, 5

Parent

Tip Force 1, *Tip Force 2*, *Tip Force 3*, *Tip Force 4*, and *Tip Force 5* belong to the type *SG Process*, in the topic *Motion*.

Description

Used to define the relation between tip force and motor torque for a servo gun (see *Force-torque relation on page 784*).

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.

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.

It is possible to change the values for index 6-10 manually by changing a MOC.cfg.

Limitations

Tip Force can only be used for servo tools.

Allowed values

A numeric value between 0 and 20000 (N).

6.32.21 Motor Torque 1, 2, 3, 4, 5

Parent

Motor Torque 1, *Motor Torque 2*, *Motor Torque 3*, *Motor Torque 4*, and *Motor Torque 5* belong to the type *SG Process*, in the topic *Motion*.

Description

Used to define the relation between tip force and motor torque for a servo gun (see *Force-torque relation on page 784*).

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.

Usage

Measure the tip force and the motor torque for some different values

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.

It is possible to change the values for index 6-10 manually by changing a MOC.cfg.

Limitations

Motor Torque can only be used for servo tools.

Allowed values

A numeric value between -1000 and 1000 (Nm).

6.32.22 Position 1, 2, 3, 4, 5

Parent

Position 1, *Position 2*, *Position 3*, *Position 4*, and *Position 5* belong to the type *SG Process*, in the topic *Motion*.

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 784*).

Position defines the joint position for the servo gun in the force-torque graph.

Usage

Position is used to control the servo gun when a change of force is ordered during welding.

It is possible to change the values for index 6-10 manually by changing a MOC.cfg.

Limitations

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.

6.32.23 Max Gun Force

Parent

Max Gun Force belongs to the type *SG Process*, in the topic *Motion*.

Description

Max Gun Force is set to highest force value that the gun can use.

Usage

Max Gun Force can be used in RAPID to supervise the maximum force of the servo gun.

Limitations

Max Gun Force can only be used for servo tools.

Allowed values

A numeric value.

The unit of the value depends on the servo gun calibration.

6.32.24 Automatic open disabled

Parent

Automatic open disabled belongs to the type *SG Process*, in the topic *Motion*.

Description

Automatic open disabled defines if the tool should open or not when stop occurs. It is recommended for servo grippers, but not for servo guns.

Usage

If you want the servo tool to remain closed when a stop occurs, this parameter should be set.

Limitations

Automatic open disabled can only be used if you have servo tools.

Allowed values

Yes or No.

Default value is No.

6.32.25 Force matching deflection values

Parent

Force matching deflection values belongs to the type *SG Process*, in the topic *Motion*.

Description

The applied force that result in the deflections defined in *Deflection in z direction (m) on page 811* and *Deflection in x direction (m) on page 812*.

Usage

Apply the force *Force matching deflection values* and measure the deflection in z and x direction.

Limitations

Force matching deflection values can only be used for servo tools.

Allowed values

A numeric value.

The unit of the value depends on the servo gun calibration.

6.32.26 Ramp time matching deflection values

Parent

Ramp time matching deflection values belongs to the type *SG Process*, in the topic *Motion*.

Description

The time it takes for the servo gun to build up the force specified in *Force matching deflection values on page 809*.

Usage

This data is used to calculate the robot movement for the servo gun deflection compensation.

Limitations

Ramp time matching deflection values can only be used for servo tools.

Allowed values

A value between 0 and 1 (s).

Default value is 0.1.

6.32.27 Deflection in z direction (m)

Parent

Deflection in z direction (m) belongs to the type *SG Process*, in the topic *Motion*.

Description

TCP deviation in z-direction caused of gun arm deflection when the gun is closed with the force specified in *Force matching deflection values on page 809*. This data is used for the deflection compensation movement of the robot.

Usage

Apply the force defined in *Force matching deflection values* and measure the deflection in z direction. Define this measured deflection in *Deflection in z direction (m)*.

Limitations

Deflection in z direction (m) can only be used for servo tools.

Allowed values

A value between 0 and 0.02 m.

6.32.28 Deflection in x direction (m)

Parent

Deflection in x direction (m) belongs to the type *SG Process*, in the topic *Motion*.

Description

TCP deviation in x-direction caused of gun arm deflection when the gun is closed with the force specified in *Force matching deflection values on page 809*. This data is used for the deflection compensation movement of the robot.

Usage

Apply the force defined in *Force matching deflection values* and measure the deflection in x direction. Define this measured deflection in *Deflection in x direction (m)*.

Limitations

Deflection in x direction (m) can only be used for servo tools.

Allowed values

A value between -0.02 and 0.02 m.

6.32.29 Missing tip check distance

Parent

Missing tip check distance belongs to the type *SG Process*, in the topic *Motion*.

Description

The parameter *Missing tip check distance* defines the distance for when to trigger an error that a tip is missing from the tool.

Usage

For example, set this value to 75% of the tip size. If a tip is missing, the tool will continue to close after the normal squeeze position. If the distance to the normal squeeze position is larger than this value, an error message will appear, and the system goes to motor off.

Consider the motor movement during force squeeze and this parameter to avoid false triggering.

Limitations

Missing tip check distance can only be used for servo tools.

Allowed values

A value between 0.001 and 5 (m).

Default value is 5.

6.32.30 Max difference for gravity compensation

Parent

Max difference for gravity compensation belongs to the type *SG Process*, in the topic *Motion*.

Description

Max difference for gravity compensation is set as the difference in force when the spot welding gun is closing in negative direction with respect to the gravity direction, compared to closing in positive direction with respect to the gravity direction.

This parameter is optional.

Usage

Max difference for gravity compensation can be used in spot welding applications to compensate force depending on servo gun movement with respect to gravity direction. It will be calculated by the calibration method in SpotWare.

Limitations

Max difference for gravity compensation can only be used for servo tools and will only have an effect with SpotWare.

Allowed values

A numeric value between 0 and 1000.

The unit of the value depends on the servo gun calibration.

6.33 Type Single

6.33.1 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.

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.

A maximum of 12 instances of the type *Single* can be configured in a system.

6.33.2 Name

Parent

Name belongs to the type *Single*, in the topic *Motion*.

Description

Name defines the name of the single.

Allowed values

A string with maximum 32 characters.

6.33.3 Serial Number Low Part

Parent

Serial Number Low Part belongs to the type *Single*, in the topic *Motion*.

Description

Serial Number Low Part defines the low part of the single's serial number.

Usage

The low integer part of the serial number.

Allowed values

An integer value with maximum nine digits.

Default value is 0.

6.33.4 Serial Number High Part

Parent

Serial Number High Part belongs to the type *Single*, in the topic *Motion*.

Description

Serial Number High Part defines the high part of the single's serial number.

Usage

The high part is the first four characters of the serial number.

Allowed values

A string with maximum four characters.

Default value is 0000.

6.33.5 Use Single Type

Parent

Use Single Type belongs to the type *Single*, in the topic *Motion*.

Description

Use Single Type defines what single type is used. For more information, see *The type Single Type on page 827*.

Usage

The single type is defined in the type *Single Type*.

Allowed values

A string with maximum 32 characters.

6.33.6 Use Joint

Parent

Use Joint belongs to the type *Single*, in the topic *Motion*.

Description

Use Joint defines which joint data to use for the single. For more information, see *The Joint type on page 573*.

Usage

The joints are defined in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6.33.7 Base Frame x, y, z

Parent

Base Frame x, *Base Frame y*, and *Base Frame z* belong to the type *Single* in the topic *Motion*.

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).

For more information, see *How to define base frame on page 448*.

Allowed values

A value between -1,000 and 1,000 meters.

6.33.8 Base Frame q1, q2, q3, q4

Parent

Base Frame q1, *Base Frame q2*, *Base Frame q3*, and *Base Frame q4* belong to the type *Single* in the topic *Motion*.

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.

For more information, see *How to define base frame on page 448*.

Allowed values

A value between -1 and 1 specifying the orientation.

6.33.9 Base Frame Coordinated

Parent

Base Frame Coordinated belongs to the type *Single* in the topic *Motion*.

Description

Base Frame Coordinated defines the name of robot or single that moves the base frame of this single. For more information, see *How to define base frame on page 448*.

Allowed values

A string with maximum 32 characters.

6.33.10 Mech.Unit Not Moving Detection Level

Parent

Mech.Unit Not Moving Detection Level belongs to the type *Single*, in the topic *Motion*.

Description

Mech.Unit Not Moving Detection Level defines the detection level for a *Single* for the system output *Mechanical Unit Not Moving*. For more information, see *Mechanical Unit Not Moving on page 298* and *Mech.Unit Not Moving Detection Level on page 766*.

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 is 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.

6.33.11 Ignore joint world zones

Parent

Ignore joint world zones belongs to the type *Single*, in the topic *Motion*.

Description

If *Ignore joint world zones* is set, this axis will be excluded from consideration in all joint WorldZones, overriding any setting in the instructions `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.

6.33.12 Single weight in power sharing

Parent

Single weight in power sharing belongs to the type *Single*, in the topic *Motion*.

Description

The parameter *Single weight in power sharing* is used to decide how power should be distributed between robots and singles in case the power supply in the controller is a limiting factor.

Usage

If a movement with both a robot and a positioner needs to be limited due to power supply then the single's part of the total power will be decided by this parameter.

If robot performance is prioritized then *Robot weight in power sharing* should be increased and vice versa if single performance is preferred.

Example

If the power supply has 10 kW with a robot on a track motion and the total required power for a movement would be, for the robot 10 kW and track 2 kW. Then the parameters can be set as follows:

- *Robot weight in power sharing*: 90
- *Single weight in power sharing*: 10

In this case the distribution of the available power would be 9 kW for robot and 1 kW for track.

Allowed values

A value between 0 and 1000000.

Default value is 1.

Additional information

This parameter does not need a restart of the controller to apply the changes. Hence, it is possible to test directly after changing the parameter value.

Related information

Robot weight in power sharing on page 777

6.34 Type Single Type

6.34.1 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.

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. For more information, see *The Single type on page 815*.

6.34.2 Name

Parent

Name belongs to the type *Single Type* in the topic *Motion*.

Description

Name defines the name of the single type.

Allowed values

A string with maximum 32 characters.

6.34.3 Mechanics

Parent

Mechanics belongs to the type *Single Type* in the topic *Motion*.

Description

Mechanics defines what type of mechanics the single type uses.

Limitation

The value *TRACK* is only used when a robot is mounted on a track. For a linear axis without a mounted robot, the value *FREE_ROT* shall be used and the parameter *Rotating Move* shall be set to *No* (type *Transmission*).

Allowed values

The following mechanics are available/allowed:

Value:	Description:
TRACK	Linear track motion, only used when a robot is mounted on the track
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

Rotating Move on page 873

6.35 Type SIS Parameters and SIS Single Parameters

6.35.1 The SIS Parameters type and the SIS Single Parameters type

Overview

This section describes the types *SIS Parameters* and *SIS Single Parameters* which belong to the topic *Motion*. Each parameter of these types is described in a separate information topic in this section.

Type description

The type *SIS Parameters* describes the service intervals and warning levels for the robot. The service interval can be set in both production time and calendar time.

The type *SIS Single Parameters* describes the service intervals and warning levels for external axes. The service interval can be set in both production time and calendar time.

The parameters for the types *SIS Parameters* and *SIS Single Parameters* are identical in usage and allowed values. Therefore they are described together in this manual.

Limitations

Changing the parameter values in *SIS Single Parameters* is only useful if you have one or more external axes.

Related information

See the product manual for the robot.

6.35.2 Name

Parent

Name belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Name defines the SIS parameter name.

Allowed values

A string with maximum 32 characters.

6.35.3 Operational Limit (h)

Parent

Operational Limit (h) belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Operational Limit (h) describes the service interval measured in production time.

Usage

The service interval for production time, *Operational Limit (h)*, for ABB robots is normally set on delivery and should be changed if the value differs from the maintenance schedule in the product manual.

When the *Operational Limit (h)* is reached, the FlexPendant displays a message from the elog.

If *Operational Limit (h)* is set to 0, the function is disabled.

Allowed values

A value between 0 and 50000 hours.

6.35.4 Calendar Limit (years)

Parent

Calendar Limit (years) belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Calendar Limit (years) defines the service interval in calendar time.

Usage

The service interval for calendar time, *Calendar Limit (years)*, for ABB robots is normally set on delivery and should be changed if the value differs from the maintenance schedule in the product manual.

When the *Calendar Limit (years)* is reached, the FlexPendant displays a message from the elog.

If *Calendar Limit (years)* is set to 0, the function is disabled.

Allowed values

A value between 0 and 20 years.

6.35.5 Operational Warning (%)

Parent

Operational Warning (%) belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Operational Warning (%) defines when the warning before reached service level for production time should occur.

Usage

The value of *Operational Warning (%)* is a percentage of the *Operational Limit (h)*. A higher number gives a shorter time between the warning and the reached service level.

If *Operational Warning (%)* is set to 0, the warning is disabled.

Allowed values

A value between 0 and 100 %.

Related information

Operational Limit (h) on page 832.

6.35.6 Calendar Warning (%)

Parent

Calendar Warning (%) belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Calendar Warning (%) defines when the warning before reached service level for calender time should occur.

Usage

The value of *Calendar Warning (%)* is a percentage of the *Calendar Limit (years)*. A higher number gives a shorter time between the warning and the reached service level.

If *Calendar Warning (%)* is set to 0, the warning is disabled.

Allowed values

A value between 0 and 100 %.

Related information

Calendar Limit (years) on page 833.

6.35.7 Gearbox Warning (%)

Parent

Gearbox Warning (%) belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Gearbox Warning (%) defines when the warning before reached service level for gearbox should occur.

Usage

The estimated gearbox service interval (remaining lifetime) is calculated automatically. The value of *Gearbox Warning (%)* is a percentage of the estimated gearbox service interval. A higher number gives a shorter time between the warning and the reached service level.

For an ABB robot using SIS, the value is typically set to 100.

If *Gearbox Warning (%)* is set to 0, the warning is disabled.

Allowed values

A value between 0 and 100 %.

6.35.8 Robot temperature (C), Single temperature (C)

Parent

Robot temperature (C) and *Single temperature (C)* belong to the type *SIS Parameters*, in the topic *Motion*.

Description

Robot temperature (C) and *Single temperature (C)* defines the ambient temperature (°C).

Usage

The gearbox service interval (remaining lifetime) is calculated automatically, using among other things *Robot temperature (C)* or *Single temperature (C)*. For an ABB robot using SIS, the value is typically set to 50 and should be changed if the value differs from the value defined in the product manual.

The temperature value can be changed to the actual temperature of the location where the robot is used.

Allowed values

A value between 35 and 50.

If a value lower than 35 is defined, then 35 will be used.

6.35.9 Events as Warnings

Parent

Events as Warnings belongs to the type *SIS Parameters* in the topic *Motion*.

Description

Events as Warnings defines whether SIS event logs should be warnings instead of errors.

Usage

Set to *Yes* when you want the SIS event logs to be reported as warnings instead of errors. The main difference is that warnings don't take focus on the FlexPendant, and that they have a different icon. This will affect calendar time events, production time events, and gearbox events.

Allowed values

Yes or *No*.

6.36 Type Stress Duty Cycle

6.36.1 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.

Type description

The type *Stress Duty Cycle* is used to protect axes, gearboxes, etc. Damage 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.

6.36.2 Name

Parent

Name belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

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.

6.36.3 Speed Absolute Max

Parent

Speed Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

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).

6.36.4 Torque Absolute Max

Parent

Torque Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

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).

6.37 Type Supervision

6.37.1 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.

Type description

The type *Supervision* is used for supervision of joints. Each joint has one set of parameters of the type *Supervision*. For more information, see *The Joint type on page 573*.

Limitation

Parameters of the type *Supervision* can only be defined for additional axes.

6.37.2 Name

Parent

Name belongs to the type *Supervision*, in the topic *Motion*.

Description

The name of the supervision.

Allowed values

A string with maximum 32 characters.

6.37.3 Brake Release Supervision On

Parent

Brake Release Supervision On belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.4 Speed Supervision

Parent

Speed Supervision belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.5 Position Supervision

Parent

Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.6 Counter Supervision

Parent

Counter Supervision belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.7 Jam Supervision

Parent

Jam Supervision belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.8 Load Supervision

Parent

Load Supervision belongs to the type *Supervision*, in the topic *Motion*.

Description

Defines if the load supervision should be activated or not.

Usage

The load supervision should normally be On.

Allowed values

On or Off

6.37.9 Power Up Position Supervision

Parent

Power Up Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

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

6.37.10 In Position Range

Parent

In Position Range belongs to the type *Supervision*, in the topic *Motion*.

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.

6.37.11 Zero Speed (%)

Parent

Zero Speed (%) belongs to the type *Supervision*, in the topic *Motion*.

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.

6.37.12 Affects Forced Control

Parent

Affects Forced Control belongs to the type *Supervision*, in the topic *Motion*.

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.

For more information, see *Forced Control Active on page 588*.

Allowed values

Yes or No

6.37.13 Forced on Position Limit

Parent

Forced on Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Description

The upper position limit for activation of forced control, measured from the fine point. For more information, see *Affects Forced Control on page 854*.

Usage

The upper position limit is measured in radians on the motor shaft.

Allowed values

A value between 0 and 5.

6.37.14 Forced off Position Limit

Parent

Forced off Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Description

The lower position limit for deactivation of forced control used close to the fine point. For more information, see *Affects Forced Control on page 854*.

Usage

The lower position limit is measured in radians on the motor shaft.

Limitations

Must have a lower value than *Forced on Position Limit*. For more information, see *Forced on Position Limit on page 855*.

Allowed values

A value between 0 and 5.

6.37.15 Thermal Supervision Sensitivity Ratio

Parent

Thermal Supervision Sensitivity Ratio belongs to the type *Supervision*, in the topic *Motion*.

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, MU 400) and gear units (MTD 250, MTD 500, MTD 750, 200 MID 500, MID 1000).

Allowed values

A value between 0.5 and 2.

6.38 Type Supervision Type

6.38.1 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.

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.

6.38.2 Name

Parent

Name belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.3 Max Force Control Position Error

Parent

Max Force Control Position Error belongs to the type *Supervision Type*, in the topic *Motion*.

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 0.2 (meter).

The default value is 0.03.

6.38.4 Max Force Control Speed Limit

Parent

Max Force Control Speed Limit belongs to the type *Supervision Type*, in the topic *Motion*.

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. For more information, see *The Force Master Control type on page 548*.

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.

6.38.5 Dynamic Power Up Position Limit

Parent

Dynamic Power Up Position Limit belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.6 Teach Max Speed Main

Parent

Teach Max Speed Main belongs to the type *Supervision Type*, in the topic *Motion*.

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.

CAUTION

Incorrectly defined parameters will result in incorrect speed. Always verify the speed after changing these parameters.

There is a hazard that the speed 250 mm/s is exceeded in manual reduced speed mode.

6.38.7 Teach Max Speed DSP

Parent

Teach Max Speed DSP belongs to the type *Supervision Type*, in the topic *Motion*.

Description

Defines the motor speed supervision level in manual mode.

Usage

Teach Max Speed DSP is used for speed supervision in 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.

CAUTION

Incorrectly defined parameters will result in incorrect speed. Always verify the speed after changing these parameters.

There is a hazard that the speed 250 mm/s is exceeded in manual reduced speed mode.

6.38.8 Max Jam Time

Parent

Max Jam Time belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.9 Max Overload Time

Parent

Max Overload Time belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.10 Auto Max Speed Supervision Limit

Parent

Auto Max Speed Supervision Limit belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.11 Influence Group

Parent

Influence Group belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.12 Alarm Position Limit for Brake Release

Parent

Alarm Position Limit for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

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.

6.38.13 Position OK Ratio for Brake Release

Parent

Position OK Ratio for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

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*. For more information, see *Alarm Position Limit for Brake Release on page 869*.

Allowed values

A value between 0 and 1.

Default value is 0.2, a normal value is 0.2-0.5.

6.39 Type Transmission

6.39.1 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.

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.

6.39.2 Name

Parent

Name belongs to the type *Transmission*, in the topic *Motion*.

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.

6.39.3 Rotating Move

Parent

Rotating Move belongs to the type *Transmission*, in the topic *Motion*.

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.

6.39.4 Transmission Gear Ratio

Parent

Transmission Gear Ratio belongs to the type *Transmission*, in the topic *Motion*.

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.

> **CAUTION**
>
> Incorrectly defined parameters will result in incorrect speed. Always verify the speed after changing these parameters.
>
> There is a hazard that the speed 250 mm/s is exceeded in manual reduced speed mode.

6.39.5 Transmission Gear High

Parent

Transmission Gear High belongs to the type *Transmission*, in the topic *Motion*.

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 on page 459* 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*.

For more information, see *Transmission Gear Low on page 876*.

Allowed values

An integer value except 0.

Related information

Application manual - Controller software OmniCore

6.39.6 Transmission Gear Low

Parent

Transmission Gear Low belongs to the type *Transmission*, in the topic *Motion*.

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 on page 459* 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*.

For more information, see *Transmission Gear High on page 875*.

Allowed values

An integer value except 0.

Related information

Application manual - Controller software OmniCore

6.40 Type Uncalibrated Control Master 0

6.40.1 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.

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.

6.40.2 Name

Parent

Name belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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.

6.40.3 Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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).

6.40.4 Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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).

6.40.5 Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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.

6.40.6 Speed Max Uncalibrated

Parent

Speed Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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).

6.40.7 Acceleration Max Uncalibrated

Parent

Acceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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).

6.40.8 Deceleration Max Uncalibrated

Parent

Deceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

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).

7 Topic Process

7.1 The Process topic

Overview

The *Process* topic contains parameters for configuring various process applications. These parameters are described in the application manuals for the respective options.

This page is intentionally left blank

Index

ABB AB
Robotics & Discrete Automation
S-721 68 VÄSTERÅS, Sweden
Telephone +46 10-732 50 00

ABB AS
Robotics & Discrete Automation
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 22 87 2000

ABB Engineering (Shanghai) Ltd.
Robotics & Discrete Automation
No. 4528 Kangxin Highway
PuDong New District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

ABB Inc.
Robotics & Discrete Automation
1250 Brown Road
Auburn Hills, MI 48326
USA
Telephone: +1 248 391 9000

abb.com/robotics

3HAC065041-001, Rev T, en