

ROBOTICS

Operating manual

IRC5 with FlexPendant



Trace back information:
Workspace 22D version a4
Checked in 2022-12-05
Skribenta version 5.5.019

Operating manual
IRC5 with FlexPendant

RobotWare 6.14.01

Document ID: 3HAC050941-001

Revision: P

The information in this manual is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this manual.

Except as may be expressly stated anywhere in this manual, nothing herein shall be construed as any kind of guarantee or warranty by ABB for losses, damage to persons or property, fitness for a specific purpose or the like.

In no event shall ABB be liable for incidental or consequential damages arising from use of this manual and products described herein.

This manual and parts thereof must not be reproduced or copied without ABB's written permission.

Keep for future reference.

Additional copies of this manual may be obtained from ABB.

Original instructions.

© Copyright 2022 ABB. All rights reserved.
Specifications subject to change without notice.

Table of contents

Overview of this manual	9
Product documentation	12
1 Welcome to IRC5	15
1.1 About this section	15
1.2 The IRC5 controller	16
1.3 The FlexPendant	17
1.4 RobotStudio Online	24
1.5 RobotStudio	26
1.6 When to use different jogging devices	27
1.7 Buttons and ports on the controller	30
2 Navigating and handling FlexPendant	33
2.1 About this chapter	33
2.2 The Main menu	34
2.2.1 HotEdit menu	34
2.2.2 FlexPendant Explorer	36
2.2.3 Inputs and Outputs, I/O	37
2.2.4 Jogging	38
2.2.5 Production Window	40
2.2.6 Program Data	41
2.2.7 Program Editor	43
2.2.8 Backup and Restore	45
2.2.9 Calibration	46
2.2.10 Control Panel	48
2.2.11 Event Log	49
2.2.12 System Info	51
2.2.13 Restart	53
2.2.14 Log Off	54
2.3 Operator window	55
2.4 Status bar	56
2.5 Quickset	57
2.5.1 The Quickset menu	57
2.5.2 Quickset menu, Mechanical unit	58
2.5.3 Quickset menu, Increment	63
2.5.4 Quickset menu, Run Mode	64
2.5.5 Quickset menu, Step Mode	65
2.5.6 Quickset menu, Speed	66
2.5.7 Quickset menu, Tasks	67
2.6 Basic procedures	68
2.6.1 Using the soft keyboard	68
2.6.2 Messages on the FlexPendant	70
2.6.3 Scrolling and zooming	71
2.6.4 Filtering data	72
2.6.5 Process applications	75
2.6.6 Granting access for RobotStudio	76
2.6.7 Logging on and off	77
2.7 Changing FlexPendant settings	79
2.7.1 System settings	79
2.7.1.1 Setting default paths	79
2.7.1.2 Defining a view to be shown during operating mode change or startup ...	80
2.7.1.3 Changing the background image	81
2.7.1.4 Defining visibility level for UAS protected functions	82
2.7.1.5 Defining an additional test view	83
2.7.1.6 Defining position programming rule	84
2.7.1.7 Defining which tasks should be selectable in the tasks panel	86

Table of contents

2.7.1.8	Managing the display of controller and system name	87
2.7.2	Basic settings	88
2.7.2.1	Changing brightness and contrast	88
2.7.2.2	Adapting the FlexPendant for left-handed users	89
2.7.2.3	Controller settings	91
2.7.2.4	Configuring Most Common I/O	93
2.7.2.5	Changing language	94
2.7.2.6	Changing programmable keys	95
2.7.2.7	Calibrating the touchscreen	97
3	Jogging	99
3.1	Introduction to jogging	99
3.2	Coordinate systems for jogging	101
3.3	Joystick directions	107
3.4	Restrictions to jogging	108
3.5	Coordinated jogging	109
3.6	Basic settings for jogging	110
3.6.1	Selecting mechanical unit for jogging	110
3.6.2	Selecting motion mode	112
3.6.3	Selecting tool, work object, and payload	113
3.6.4	Setting the tool orientation	114
3.6.5	Jog axis by axis	115
3.6.6	Selecting coordinate system	116
3.6.7	Locking the joystick in specific directions	117
3.6.8	Incremental movement for precise positioning	119
3.6.9	Reading the exact position	121
3.6.10	Aligning tools	123
4	Programming and testing	125
4.1	Before you start programming	125
4.2	Using RAPID programs	126
4.3	Programming concept	128
4.3.1	Handling of programs	128
4.3.2	Handling of modules	131
4.3.3	Handling of routines	134
4.3.4	Handling of instructions	139
4.3.5	Example: Add movement instructions	143
4.3.6	About the Program and Motion Pointers	144
4.4	Data types	145
4.4.1	Viewing data in specific tasks, modules, or routines	145
4.4.2	Creating new data instance	146
4.4.3	Editing data instances	148
4.5	Tools	152
4.5.1	What is a tool?	152
4.5.2	What is the tool center point?	154
4.5.3	Creating a tool	156
4.5.4	Defining the tool frame	159
4.5.5	Editing the tool data	163
4.5.6	Editing the tool declaration	166
4.5.7	Deleting a tool	167
4.5.8	Setup for stationary tools	168
4.6	Work objects	170
4.6.1	What is a work object?	170
4.6.2	Creating a work object	171
4.6.3	Defining the work object coordinate system	172
4.6.4	Editing the work object data	176
4.6.5	Editing the work object declaration	177
4.6.6	Deleting a work object	178

4.7	Payloads	179
4.7.1	Creating a payload	179
4.7.2	Editing the payload data	181
4.7.3	Editing the payload declaration	183
4.7.4	Deleting a payload	184
4.8	Testing	185
4.8.1	About the automatic mode	185
4.8.2	About the manual mode	187
4.8.3	Using the hold-to-run function	190
4.8.4	Running the program from a specific instruction	191
4.8.5	Running a specific routine	192
4.8.6	Stepping instruction by instruction	193
4.9	Service routines	196
4.9.1	Running a service routine	196
4.9.2	Battery shutdown service routine	200
4.9.3	Axis Calibration service routine	201
4.9.4	Calibration Pendulum, CalPendulum service routine	202
4.9.5	Service Information System, ServiceInfo service routine	203
4.9.6	LoadIdentify, load identification service routine	204
4.9.7	Brake check service routine	216
4.9.8	Wrist optimization service routine	223
5	Running in production	225
5.1	Basic procedures	225
5.1.1	Starting programs	225
5.1.2	Stopping programs	228
5.1.3	Using multitasking programs	229
5.1.4	Using motion supervision and non motion execution	231
5.1.5	Connecting a FlexPendant	234
5.1.6	Using the hot plug option	236
5.2	Troubleshooting and error recovery	239
5.2.1	General procedure when troubleshooting	239
5.2.2	Returning the robot to the path	240
5.2.3	Running RAPID program with uncalibrated mechanical unit	241
5.3	Operating modes	242
5.3.1	Present operating mode	242
5.3.2	Switching from manual to automatic mode	244
5.3.3	Switching from automatic to manual mode	246
5.3.4	Switching to manual full speed mode	247
5.4	Modifying positions	248
5.4.1	Modifying and tuning positions	248
5.4.2	Modifying positions in the Program Editor or Production Window	249
5.4.3	Tuning positions with HotEdit	253
5.4.4	Working with displacements and offsets	257
5.4.5	Moving the robot to a programmed position	259
6	Handling inputs and outputs, I/O	261
6.1	Viewing signal lists	261
6.2	Simulating and changing signal values	262
6.3	Viewing signal group	265
6.4	Safety signals	266
6.4.1	Safety I/O signals	266
7	Handling the event log	269
7.1	Accessing the event log	269
7.2	Deleting log entries	270
7.3	Saving log entries	271

Table of contents

8	Backup and restore	273
8.1	Back up the system	273
8.2	Important when performing backups	275
8.3	Restore the system	277
9	Calibrating	281
9.1	How to check if the robot needs calibration	281
9.2	Updating revolution counters	282
Index		285

Overview of this manual

About this manual

This manual contains instructions for daily operation of IRC5 based robot systems using a FlexPendant.



Note

It is the responsibility of the integrator to provide safety and user guides for the robot system.

Usage

This manual should be used during operation.

Some actions that are more advanced, or not used in the daily operation, are described in *Operating manual - IRC5 Integrator's guide*.



Note

Before any work on or with the robot is performed, the safety information in the product manual for the controller and manipulator must be read.

Who should read this manual?

This manual is intended for:

- operators
- product technicians
- service technicians
- robot programmers

Prerequisites

The reader should:

- Be familiar with the concepts described in *Operating manual - Getting started, IRC5 and RobotStudio*.
- Be trained in robot operation.

References

<i>Product manual - IRC5</i> IRC5 with main computer DSQC1000 or later.	3HAC047136-001
<i>Product manual - IRC5 Panel Mounted Controller</i> IRC5 with main computer DSQC1000 or later.	3HAC047137-001
<i>Product manual - IRC5 Compact</i> IRC5 with main computer DSQC1000 or later.	3HAC047138-001
<i>Operating manual - Getting started, IRC5 and RobotStudio</i>	3HAC027097-001
<i>Operating manual - RobotStudio</i>	3HAC032104-001
<i>Operating manual - Service Information System</i>	3HAC050944-001

Continues on next page

Operating manual - Troubleshooting IRC5	3HAC020738-001
Operating manual - IRC5 Integrator's guide	3HAC050940-001
Operating manual - Calibration Pendulum	3HAC16578-1
Technical reference manual - System parameters	3HAC050948-001
Technical reference manual - RAPID Overview	3HAC050947-001
Technical reference manual - RAPID Instructions, Functions and Data types	3HAC050917-001
Technical reference manual - RAPID kernel	3HAC050946-001
Application manual - Additional axes and stand alone controller	3HAC051016-001
Application manual - Controller software IRC5	3HAC050798-001
Application manual - MultiMove	3HAC050961-001



Note

The document numbers that are listed for software documents are valid for RobotWare 6. Equivalent documents are available for RobotWare 5.

Revisions

Revision	Description
-	Released with RobotWare 6.0.
A	Released with RobotWare 6.02. <ul style="list-style-type: none"> Added information about time limitation to the section Switching to manual full speed mode on page 247. Added information on Load diagram check in the section LoadIdentify, load identification service routine on page 204. Updated the section Controller settings on page 91.
B	Released with RobotWare 6.03. <ul style="list-style-type: none"> Updated the section Step mode on page 65. Updated the section Illustration Control Panel on page 48. Updated the section System Info on page 51. Added a warning note regarding load data in the sections Before you start programming on page 125, Tools on page 152, Payloads on page 179, and Service routines on page 196.
C	Released with RobotWare 6.04. <ul style="list-style-type: none"> Updated the section Quickset menu, Increment on page 63. Updated the section Restore the system on page 277. Updated the section LoadIdentify, load identification service routine on page 204.
D	Released with RobotWare 6.05. <ul style="list-style-type: none"> Added the new section Managing the display of controller and system name on page 87. Added the new section Brake check service routine on page 216. Updated the section Defining a view to be shown during operating mode change or startup on page 80. Updated descriptions of stops. Removed the information about time limitation in the section Switching to manual full speed mode on page 247.

Continues on next page

Revision	Description
E	Released with RobotWare 6.06. <ul style="list-style-type: none"> Updated the section Brake check service routine on page 216. Updated the section Backup and restore on page 273. Updated the section Example of backward execution on page 194. Updated the section Controller settings on page 91.
F	Released with RobotWare 6.07. <ul style="list-style-type: none"> Safety section restructured. FlexPendant Home Screen background image is updated. Updated the section When is backup possible? on page 275 Updated the section LoadIdentify for 4-axis palletizing and SCARA robots on page 211 Updated the section Simulating and changing signal values on page 262.
G	Released with RobotWare 6.08. <ul style="list-style-type: none"> Updated the section Controller settings on page 91. Updated the section Editing motion supervision settings on page 232. Updated the section Changing programmable keys on page 95. Added information and code example to the section Brake check for MultiMove systems on page 217.
H	Released with RobotWare 6.10. <ul style="list-style-type: none"> Updated the information regarding tool data, see Editing the tool data on page 163.
J	Released with RobotWare 6.10.01. <ul style="list-style-type: none"> Information regarding General SIS data and Mechanical units updated in section System Info on page 51.
K	Released with RobotWare 6.11. <ul style="list-style-type: none"> The safety information is moved to the product manuals for the controller and the manipulator. Updated information about queueing backups. Added Axis Calibration service routine on page 201.
L	Released with RobotWare 6.12. <ul style="list-style-type: none"> Added Wrist optimization service routine on page 223.
M	Released with RobotWare 6.13. <ul style="list-style-type: none"> Added information for delta robots, see LoadIdentify, load identification service routine on page 204.
N	Released with RobotWare 6.14. <ul style="list-style-type: none"> Added information for delta robots, see LoadIdentify, load identification service routine on page 204. The name of the service routine <i>Bat_Shutdown</i> is changed to <i>BatteryShutDown</i>.
P	Released with RobotWare 6.14.01. <ul style="list-style-type: none"> Added information about a new version of the FlexPendant. Removed information about T10.

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

- Safety information.
- Installation and commissioning (descriptions of mechanical installation or electrical connections).
- Maintenance (descriptions of all required preventive maintenance procedures including intervals and expected life time of parts).
- Repair (descriptions of all recommended repair procedures including spare parts).
- Calibration.
- 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 Welcome to IRC5

1.1 About this section

Overview

This section presents an overview of the FlexPendant, the IRC5 controller, and RobotStudio.

A robot consists of a robot controller, the FlexPendant, RobotStudio, and one or several manipulators or other mechanical units.

This manual describes a robot without options, not a robot system. However, in a few places, the manual gives an overview of how options are used or applied. Most options are described in detail in their respective application manual.

1 Welcome to IRC5

1.2 The IRC5 controller

1.2 The IRC5 controller

The IRC5 controller

The IRC5 controller contains all functions needed to move and control the robot. The standard IRC5 controller consists of a single cabinet. The controller is also available in a compact version, *IRC5 Compact*, and it can also be integrated in an external cabinet, *Panel Mounted Controller*.

When running more than one robot with one controller (MultiMove option), an extra drive module must be added for each additional robot. However, a single control module is used.

Related information

Product manual - IRC5, IRC5 of design M2004.

Product manual - IRC5, IRC5 of design 14.

Product manual - IRC5 Panel Mounted Controller, IRC5 of design M2004.

Product manual - IRC5 Panel Mounted Controller, IRC5 of design 14.

Product manual - IRC5 Compact, IRC5 of design M2004.

Product manual - IRC5 Compact, IRC5 of design 14.

Application manual - MultiMove.

1.3 The FlexPendant

Introduction to the FlexPendant

The FlexPendant is a hand held operator unit that is used for many of the tasks when operating a robot: running programs, jogging the manipulator, modifying programs, and so on.

The FlexPendant is designed for continuous operation in harsh industrial environment. Its touchscreen is easy to clean and resistant to water, oil, and accidental welding splashes.

The FlexPendant consists of both hardware and software and is a complete computer in itself. It is connected to the robot controller by an integrated cable and connector.

The hot plug button option makes it possible to disconnect the FlexPendant in automatic mode and continue running without it.

The FlexPendant is available in different versions, as the hardware has been updated over the years. The exact appearance on the graphics might therefore differ slightly from reality.



Note

If protective gloves are used, these must be compatible with touchscreens when using the FlexPendant.

Continues on next page

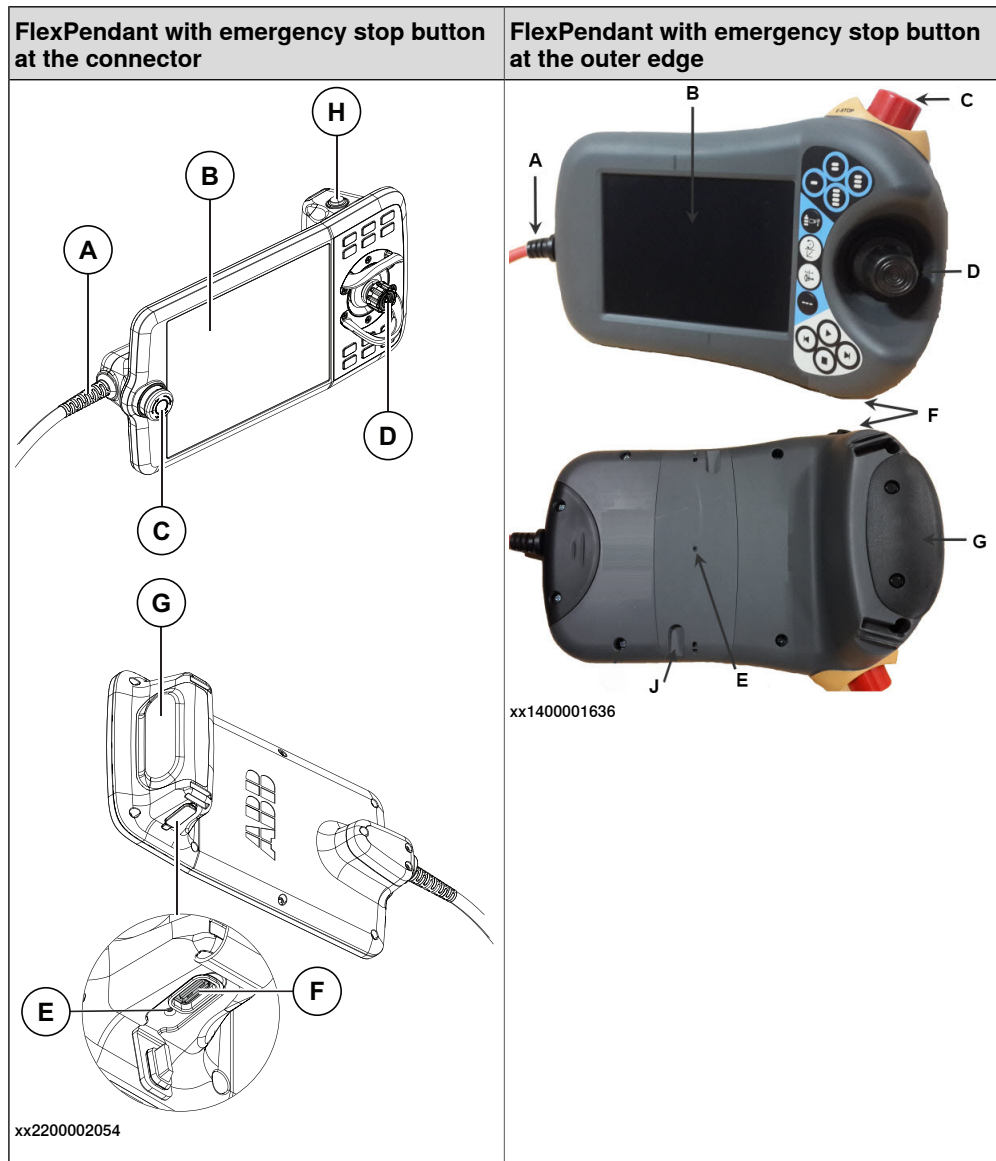
1 Welcome to IRC5

1.3 The FlexPendant

Continued

Main parts

These are the main parts of the FlexPendant.



A	Connector
B	Touchscreen
C	Emergency stop button
D	Joystick
E	Reset button
F	USB port
G	Three-position enabling device
H	Thumb button (Not available on all versions of FlexPendant.)
J	Stylus pen (Not available on all versions of FlexPendant.)

Continues on next page

Joystick

Use the joystick to move the manipulator. This is called jogging the robot. There are several settings for how the joystick will move the manipulator.

Reset button

If the FlexPendant freezes during operation, press the reset button to restart the FlexPendant.

The reset button resets the FlexPendant, not the system on the controller.

USB port

Connect a USB memory to the USB port to read or save files. The USB memory is displayed as drive */USB:Removable* in dialogs and FlexPendant Explorer.



Note

Close the protective cap on the USB port when not used.

Stylus pen

The stylus pen included with the FlexPendant is located on the back. Pull the small handle to release the pen.

Use the stylus pen to tap on the touch screen when using the FlexPendant. Do not use screw drivers or other sharp objects.

(Not available on all versions of FlexPendant.)

Hard buttons

The following hard buttons are available on the FlexPendant.

Button	Description
	Programmable keys, 1 - 4.
	Select mechanical unit.
	Toggle motion mode, reorient or linear.
	Toggle motion mode, axis 1-3 or axis 4-6.
	Toggle increments.
	Step BACKWARD button. Executes one instruction backward as button is pressed.
	START button. Starts program execution.
	Step FORWARD button. Executes one instruction forward as button is pressed.
	STOP button. Stops program execution.

Continues on next page

1 Welcome to IRC5

1.3 The FlexPendant

Continued

Three-position enabling device



CAUTION

The person using the three-position enabling device is responsible to observe the safeguarded space for hazards due to robot motion and any other hazards related to the robot.

The three-position enabling device is located on the FlexPendant. When continuously held in center-enabled position, the three-position enabling device will permit robot motion and any hazards controlled by the robot. Release of or compression past the center-enabled position will stop the robot motion.



CAUTION

For safe use of the three-position enabling device, the following must be implemented:

- The three-position enabling device must never be rendered inoperational in any way.
- If there is a need to enter safeguarded space, always bring the FlexPendant. This is to enforce single point of control.

Thumb button

The thumb button is only available on the FlexPendant with emergency stop located at the connector.

The thumb button is used for hold-to-run.

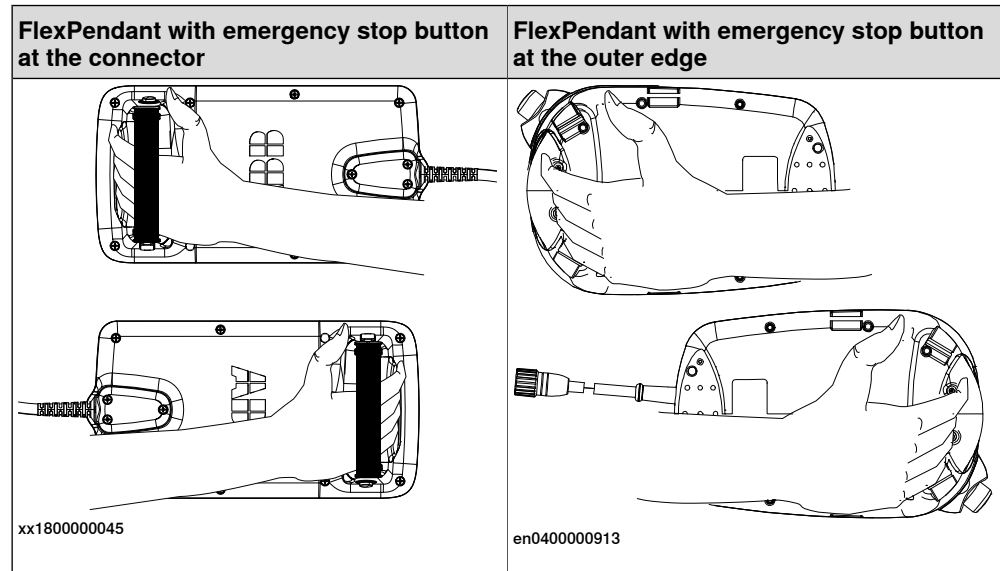
Hold-to-run is described in section [Using the hold-to-run function on page 190](#).

Continues on next page

How to hold the FlexPendant

FlexPendant is typically operated while being held in the hand. The right-handed users use their left-hand to support the FlexPendant while their right-hand performs the operations on the touch screen. However, the left-handed users can easily adapt FlexPendant for their use.

For more details, see the section [Adapting the FlexPendant for left-handed users on page 89](#).



Continues on next page

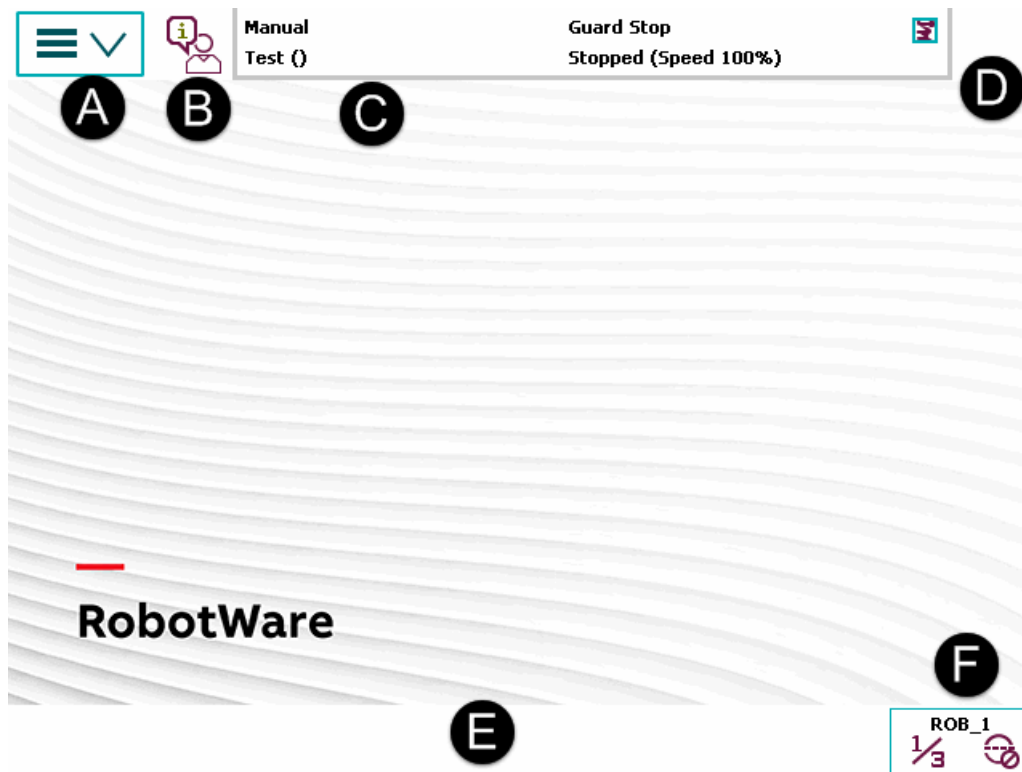
1 Welcome to IRC5

1.3 The FlexPendant

Continued

Touchscreen elements

The illustration shows important elements of the FlexPendant touchscreen.



xx1400001446

A	Main menu
B	Operator window
C	Status bar
D	Close button
E	Task bar
F	Quickset menu

Main menu

The following items can be selected from the Main menu:

- HotEdit
- Inputs and Outputs
- Jogging
- **Production Window**
- **Program Editor**
- **Program Data**
- **Backup and Restore**
- **Calibration**
- **Control Panel**
- **Event Log**

Continues on next page

- FlexPendant Explorer
- System Info
- etc.

This is further described in section [The Main menu on page 34](#).

Operator window

The operator window displays messages from robot programs. This usually happens when the program needs some kind of operator response in order to continue. This is described in section [Operator window on page 55](#).

Status bar

The status bar displays important information about system status, such as operating mode, motors on/off, program state and so on. This is described in section [Status bar on page 56](#).

Close button

Tapping the close button closes the presently active view or application.

Task bar

You can open several views from the Main menu, but only work with one at a time. The task bar displays all open views and is used to switch between these.

Quickset menu

The quickset menu provides settings for jogging and program execution. This is described in section [The Quickset menu on page 57](#).

1 Welcome to IRC5

1.4 RobotStudio Online

1.4 RobotStudio Online

Introduction to RobotStudio Online

RobotStudio Online is a suite of **Windows Store** applications intended to run on **Windows 10** tablets. It provides functionality for the shop floor commissioning of robot systems.


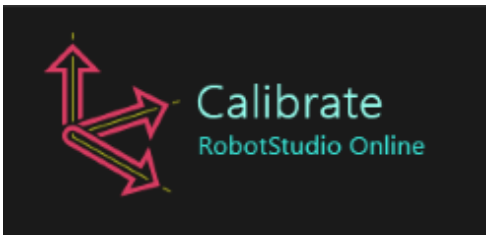
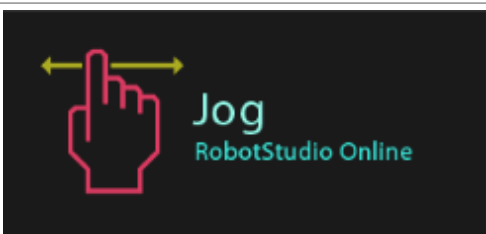
You can run these apps on a tablet that communicates with the robot controller wirelessly. To enable certain functionality, such as entering manual mode and enabling power to the mechanical unit motors, you need a safety device that is connected to the robot using the same plug that alternatively is used to connect the FlexPendant.

The following RobotStudio Online apps are available in the Microsoft [Windows Store](#):

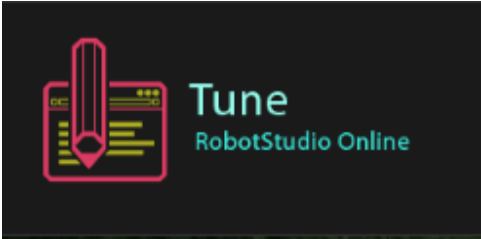
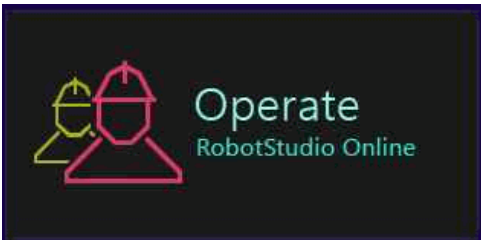
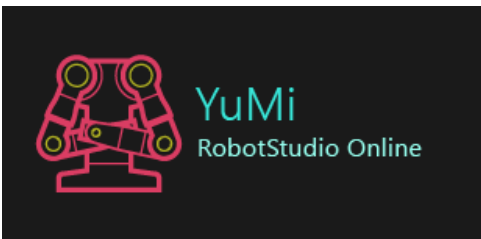


Note

You must have **Windows 8.1** to run these Apps.

RobotStudio Online Apps	Description
 xx1400002047	Manage is a tool to manage IRC5 controllers on a network.
 xx1400002049	Calibrate is a tool for calibration and definition of frames with IRC5 controllers.
 xx1400002048	Jog is a tool for manual positioning (moving or jogging) with IRC5 controllers.

Continues on next page

RobotStudio Online Apps	Description
 <p>xx1400002050</p>	<p>Tune is a tool for shop floor editing of RAPID programs with IRC5 controllers.</p>
 <p>xx1400002511</p>	<p>Operate is a tool used in production to view the program code.</p>
 <p>xx1500000832</p>	<p>YuMi is a tool for programming of the YuMi robot, IRB 14000 from ABB. It will help the users to get a fast introduction to robot programming using lead-thru and graphical programming.</p>

1 Welcome to IRC5

1.5 RobotStudio

1.5 RobotStudio

Overview of RobotStudio

RobotStudio is an engineering tool for the configuration and programming of ABB robots, both real robots on the shop floor and virtual robots in a PC. To achieve true offline programming, RobotStudio utilizes ABB VirtualRobot™ Technology. RobotStudio has adopted the Microsoft Office Fluent User Interface. The Office Fluent UI is also used in Microsoft Office. As in Office, the features of RobotStudio are designed in a workflow-oriented way.

With add-ins, RobotStudio can be extended and customized to suit the specific needs. Add-ins are developed using the RobotStudio SDK. With the SDK, it is also possible to develop custom SmartComponents which exceed the functionality provided by RobotStudio's base components.

For more information, see *Operating manual - RobotStudio*.

RobotStudio for real controllers

RobotStudio allows, for example, the following operations when connected to a real controller:

- Installing and modifying RobotWare systems on controllers, using the **Installation Manager 6**.
- Text-based programming and editing, using the **RAPID Editor**.
- File manager for the controller.
- Administrating the User Authorization System.
- Configuring system parameters.

1.6 When to use different jogging devices

Overview

For operating and managing the robot, you can use any of the following:

- FlexPendant: Optimized for handling robot motions and ordinary operation
- RobotStudio: Optimized for configuration, programming and other tasks not related to the daily operation.
- RobotStudio Online Apps : Optimized for jogging, managing, working with the frames, calibration methods and RAPID programs available in the robot controller.

Start, restart and shut down the controller

To...	Use...
Start the controller	The power switch on the controller's front panel.
Restart the controller	The FlexPendant, RobotStudio, RobotStudio Online Apps or the power switch on the controller's front panel.
Shut down the controller	The power switch on the controller's front panel or the FlexPendant, tap Restart , then Advanced .
Shut down the main computer	The FlexPendant.

Run and control robot programs

To...	Use...
Jog a robot	The FlexPendant.
Start or stop a robot program	The FlexPendant, RobotStudio or RobotStudio Online Apps.
Start and stop background tasks	The FlexPendant, RobotStudio or RobotStudio Online Apps.

Communicate with the controller

To...	Use...
Acknowledge events	The FlexPendant or RobotStudio Online Apps.
View and save the controller's event logs	RobotStudio, FlexPendant or the RobotStudio Online Apps.
Back up the controller's software to files on the PC or a server	RobotStudio, FlexPendant or the RobotStudio Online Apps.
Back up the controller's software to files on the controller	The FlexPendant or RobotStudio Online Apps.
Transfer files between the controller and network drives	RobotStudio, FlexPendant or the RobotStudio Online Apps.

Continues on next page

1 Welcome to IRC5

1.6 When to use different jogging devices

Continued

Program a robot

To...	Use...
Create or edit robot programs in a flexible way. This is suitable for complex programs with a lot of logic, I/O signals or action instructions	RobotStudio to create the program's structure and most of the source code and the FlexPendant to store robot positions and make final adjustments to the program. When programming, RobotStudio provides the following advantages: <ul style="list-style-type: none">• A text editor optimized for RAPID code, with auto-text and tool-tip information about instructions and parameters.• Program check with program error marking.• Close access to configuration and I/O editing.
Create or edit a robot program in a supportive way. This is suitable for programs that mostly consist of move instructions	The FlexPendant. When programming, the FlexPendant provides the following advantages: <ul style="list-style-type: none">• Instruction pick lists• Program check and debug while writing• Possibility to create robot positions while programming
Add or edit robot positions	The FlexPendant with a combination of suitable RobotStudio Online Apps.
Modify robot positions	The FlexPendant with a combination of suitable RobotStudio Online Apps.

Configure the robot's system parameters

To...	Use...
Edit the system parameters of the running system	RobotStudio, FlexPendant or the RobotStudio Online Apps.
Save the robot's system parameters as configuration files	RobotStudio, FlexPendant or the RobotStudio Online Apps.
Load system parameters from configuration files to the running system	RobotStudio, FlexPendant or the RobotStudio Online Apps.
Load calibration data	RobotStudio, FlexPendant or the RobotStudio Online Apps.

Create, modify and install systems

To...	Use...
Create or modify a system	RobotStudio together with RobotWare and a valid RobotWare Key for systems based on RobotWare 5. RobotStudio together with RobotWare and license file for systems based on RobotWare 6.
Install a system on a controller	RobotStudio
Install a system on a controller from a USB memory	The FlexPendant.

Calibration

To...	Use...
Calibrate base frame etc.	The FlexPendant or the RobotStudio Online Apps.

Continues on next page

To...	Use...
Calibrate tools, work objects etc.	The FlexPendant or the RobotStudio Online Apps.

Related information

The table below specifies which manuals to read, when performing the various tasks referred to:

Recommended use...	for details, see manual...	Document number
FlexPendant	<i>Operating manual - IRC5 with Flex-Pendant</i>	<i>3HAC050941-001</i>
RobotStudio	<i>Operating manual - RobotStudio</i>	<i>3HAC032104-001</i>

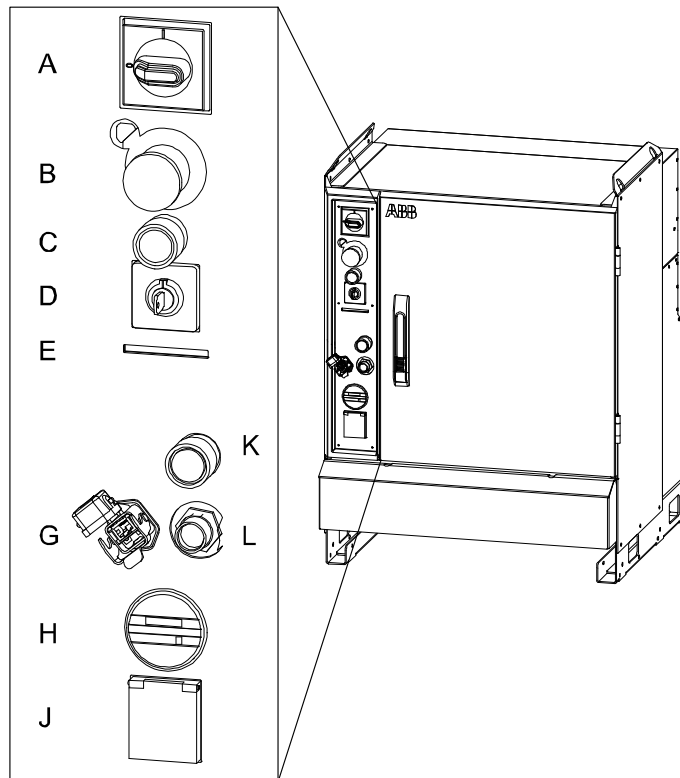
1 Welcome to IRC5

1.7 Buttons and ports on the controller

1.7 Buttons and ports on the controller

Buttons and ports on the controller

These are the buttons and ports on an IRC5 controller. Some buttons and ports are options and might not be available on your controller. The buttons and ports look the same but the placing can differ depending on the controller model (IRC5 Standard, IRC5 Compact, or IRC5 Panel Mounted Controller) and if there is an external operator's panel.



xx0600002782

A	Main switch
B	Emergency stop
C	Motors on
D	Mode switch
E	Safety chain LEDs (option)
G	Service port for PC (option)
H	Duty time counter (option)
J	Service outlet 115/230 V, 200 W (option)
K	Hot plug button (option)
L	Connector for FlexPendant

Related information

Product manual - IRC5, IRC5 of design 14.

Continues on next page

Product manual - IRC5 Panel Mounted Controller, IRC5 of design 14.

Product manual - IRC5 Compact, IRC5 of design 14.

Operating manual - Troubleshooting IRC5.

This page is intentionally left blank

2 Navigating and handling FlexPendant

2.1 About this chapter

Introduction to this chapter

This chapter will help you to work efficiently with the FlexPendant. The important elements for navigation illustrated in [Touchscreen elements on page 22](#) are described here.

All views of the Main menu, the main element for navigation, are described in overview with references to further details on how to use their functions.

In addition, this chapter provides information about basic procedures, such as how to use the soft keyboard for entering text or numbers, how to scroll and zoom the graphical touch screen, and how to use the filtering function. How to log on and log off is also described.

Handling and troubleshooting the FlexPendant

How to handle and clean the FlexPendant is described in the product manual for the robot controller.

Troubleshooting the FlexPendant is described in *Operating manual - Troubleshooting IRC5*.

Hardware and software options

Note that this manual covers only the views of a basic RobotWare system. Process applications such as arc welding, dispense, or plastics are started from the Main menu but not described in this manual. All options are detailed in their respective application manual.

2 Navigating and handling FlexPendant

2.2.1 HotEdit menu

2.2 The Main menu

2.2.1 HotEdit menu

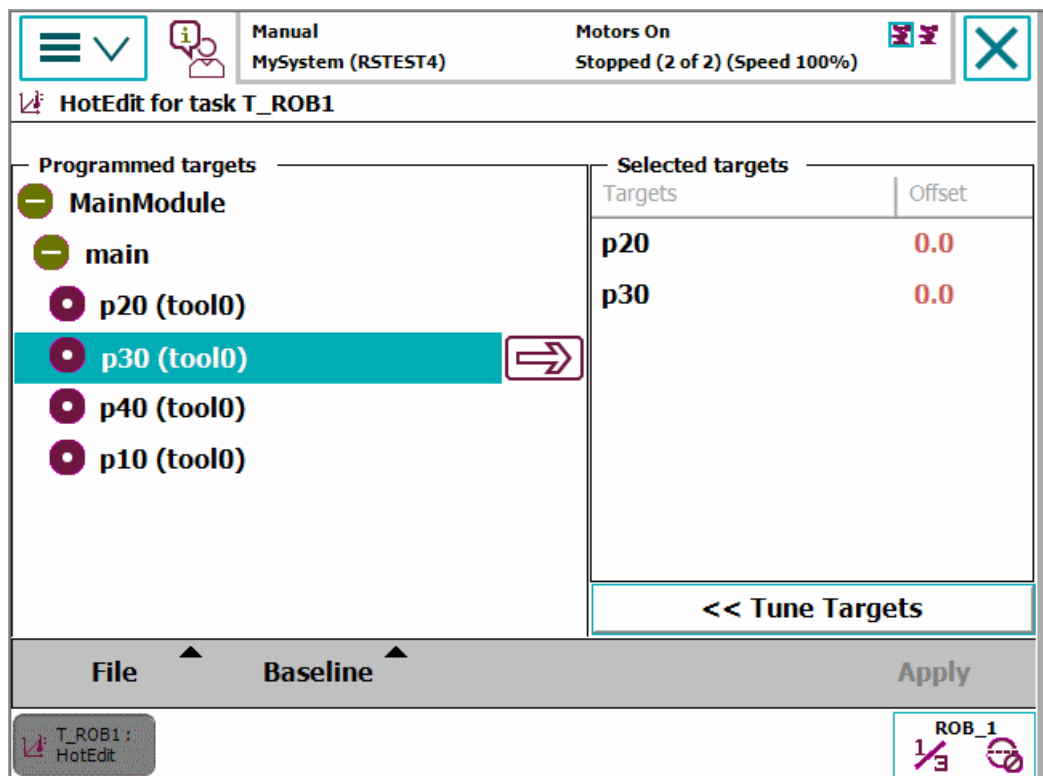
HotEdit

HotEdit is used to tune programmed positions. This can be done in all operating modes and even while the program is running. Both coordinates and orientation can be tuned.

HotEdit can only be used for named positions of the type robtarget (see limitations below).

The functions available in HotEdit may be restricted by the user authorization system, UAS.

Illustration of the HotEdit view



en0500001542

Functions available in HotEdit

Programmed targets	Lists all named positions in a tree view. Select one or several positions to be tuned by tapping the arrow. Notice that if a position is used at several places in your program, any change made to the offset will take effect everywhere it is used.
Selected targets	Lists all selected positions and their current offset. To remove a position from the selection you tap it and then tap the trash.
File	Saves and loads selections of positions to be tuned. If your system uses UAS, this may be the only way to select positions for HotEdit.

Continues on next page

Baseline	Used to apply or reject new offset values to the baseline, which holds the position values currently seen as the original ones. When you are satisfied with your HotEdit session and want to save the new offset values as the original position values, you apply these to the baseline. The old baseline values for these positions are now gone, and cannot be restored.
Tune targets	Displays settings for tuning: Coordinate system, Tuning mode and Tuning increment. Make your choices and then use the plus and minus icons to specify tuning of selected targets.
Apply	Tap Apply to make the settings made in the Tune Targets view take effect. Note that this does not change the baseline values of the positions!



CAUTION

HotEdit offers advanced functionality, which has to be handled carefully. Be aware that new offset values will be used immediately by a running program once the **Apply** button has been tapped.

Before you start using the HotEdit functionality it is strongly recommended to read [Tuning positions with HotEdit on page 253](#), where HotEdit limitations and procedures as well as the baseline concept is detailed.

Related information

See section [Modifying and tuning positions on page 248](#) for a general overview on how to modify programmed positions.

For modifying positions by jogging the robot to the new position, see section [Modifying positions in the Program Editor or Production Window on page 249](#).

For detailed information about HotEdit, see [Tuning positions with HotEdit on page 253](#).

Technical reference manual - RAPID Instructions, Functions and Data types.

Technical reference manual - System parameters, section TopicController - TypeModPos Settings.

2 Navigating and handling FlexPendant

2.2.2 FlexPendant Explorer

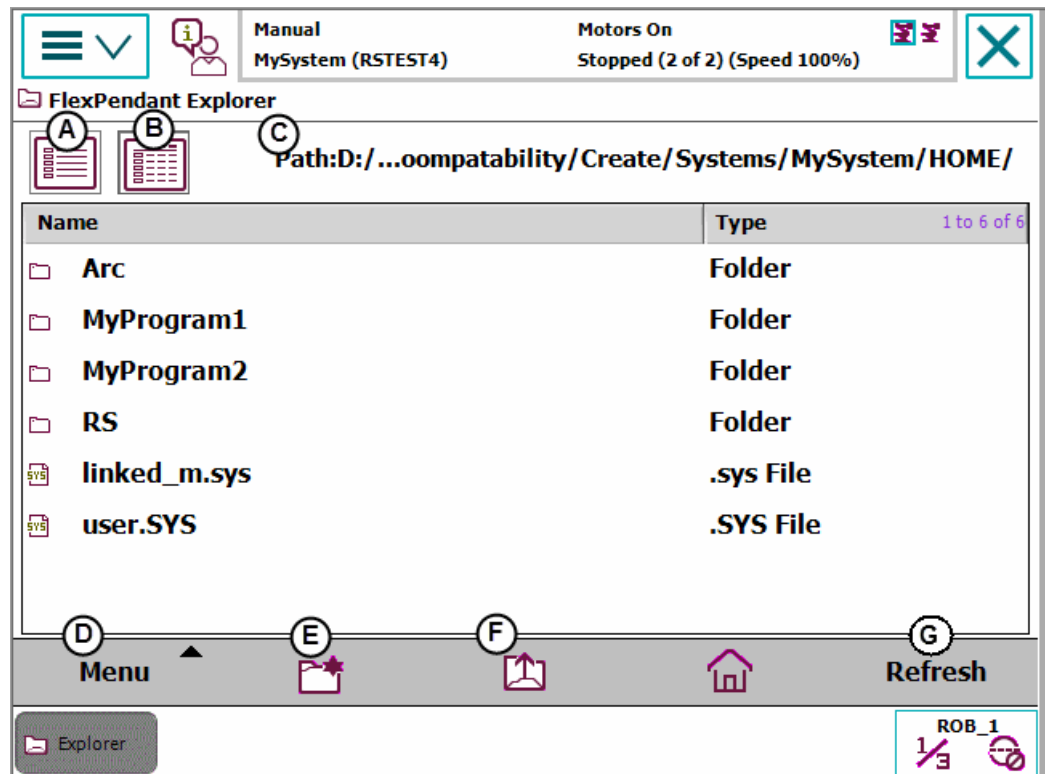
2.2.2 FlexPendant Explorer

FlexPendant Explorer

The FlexPendant Explorer is a file manager, similar to Windows Explorer, with which you can view the file system on the controller. You can also rename, delete, or move files or folders.

Illustration FlexPendant Explorer

The illustration details the FlexPendant Explorer.



en0400001130

A	Simple view. Tap to hide type in the file window.
B	Detailed view. Tap to show type in the file window.
C	Path. Displays folder paths.
D	Menu. Tap to display functions for file handling.
E	New folder. Tap to create a new folder in current folder.
F	Up one level. Tap to change to parent folder.
G	Refresh. Tap to refresh files and folders.

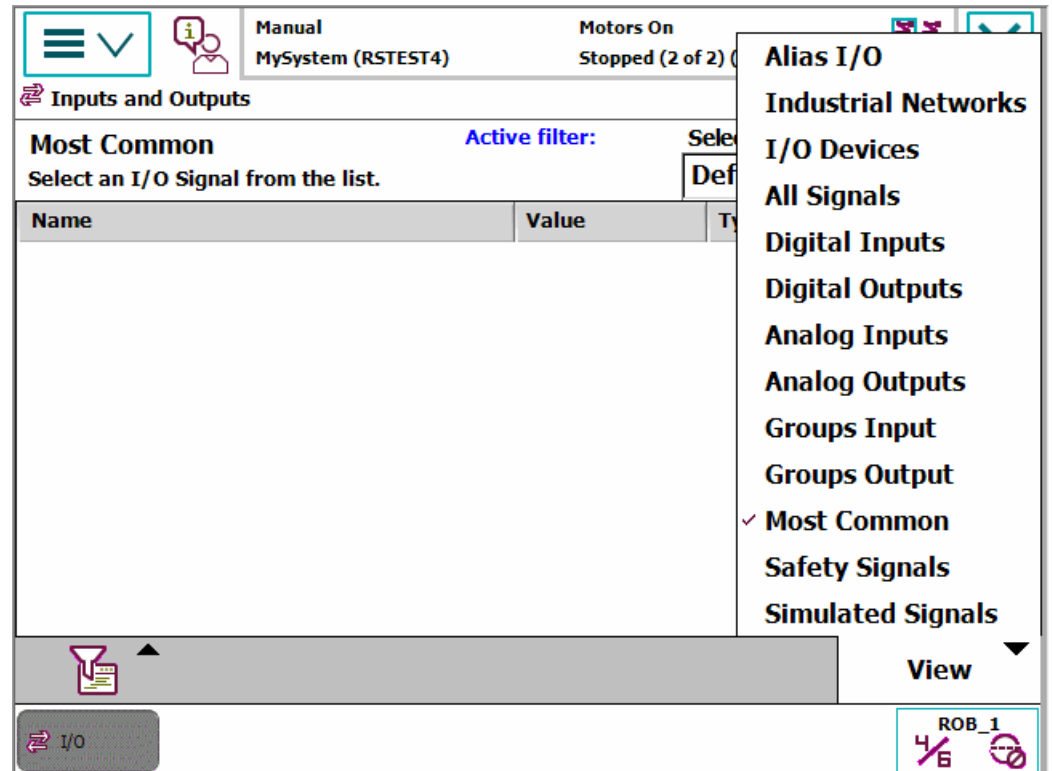
2.2.3 Inputs and Outputs, I/O

Inputs and outputs

Inputs and outputs, I/O, are signals used in the robot system. Signals are configured with system parameters.

Illustration Inputs and Outputs view

This illustration details the Inputs and Outputs view.



en040000770

What is a signal

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

- Inputs or outputs located on an industrial network 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 industrial network I/O device (virtual I/O signal).

By specifying an I/O signal, a logical representation of the real or virtual 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.

2 Navigating and handling FlexPendant

2.2.4 Jogging

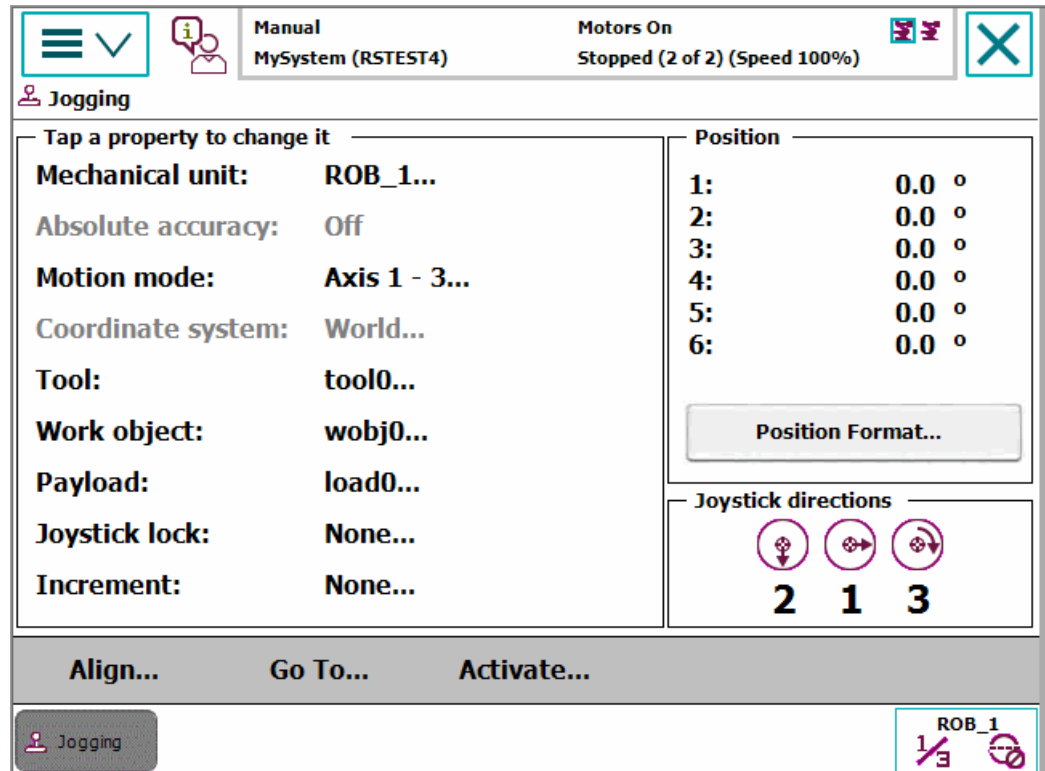
2.2.4 Jogging

Overview

The Jogging functions are found in the Jogging window. The most commonly used are also available under the Quickset menu.

Jogging menu

The illustration shows the functions available under the Jogging menu:



en0400000654

Property/button	Function
Mechanical unit	Select mechanical unit active for jogging, described in section Selecting mechanical unit for jogging on page 110 .
Absolute accuracy	Absolute Accuracy: Off is default. If the robot has the <i>Absolute Accuracy</i> option, then Absolute Accuracy: On is displayed.
Motion mode	Select motion mode, described in section Selecting motion mode on page 112 .
Coordinate system	Select coordinate system, described in section Selecting coordinate system on page 116 .
Tool	Select tool, described in section Selecting tool, work object, and payload on page 113 .
Work object	Select work object, described in section Selecting tool, work object, and payload on page 113 .
Payload	Select payload, described in section Selecting tool, work object, and payload on page 113 .

Continues on next page

Property/button	Function
Joystick lock	Select locking joystick directions, described in section Locking the joystick in specific directions on page 117 .
Increment	Select movement increments, described in section Incremental movement for precise positioning on page 119 .
Position	Displays each axis position in relation to the selected coordinate system, described in section Reading the exact position on page 121 . If the position values are displayed in red, then the revolution counters must be updated. See section Updating revolution counters on page 282 .
Position format	Select position format, described in section Reading the exact position on page 121 .
Joystick directions	Displays current joystick directions, depending on setting in Motion mode. See section Selecting motion mode on page 112 .
Align	Align the current tool to a coordinate system. See section Aligning tools on page 123 .
Go To	Move the robot to a selected position/target. See section Moving the robot to a programmed position on page 259 .
Activate	Activate a mechanical unit.

2 Navigating and handling FlexPendant

2.2.5 Production Window

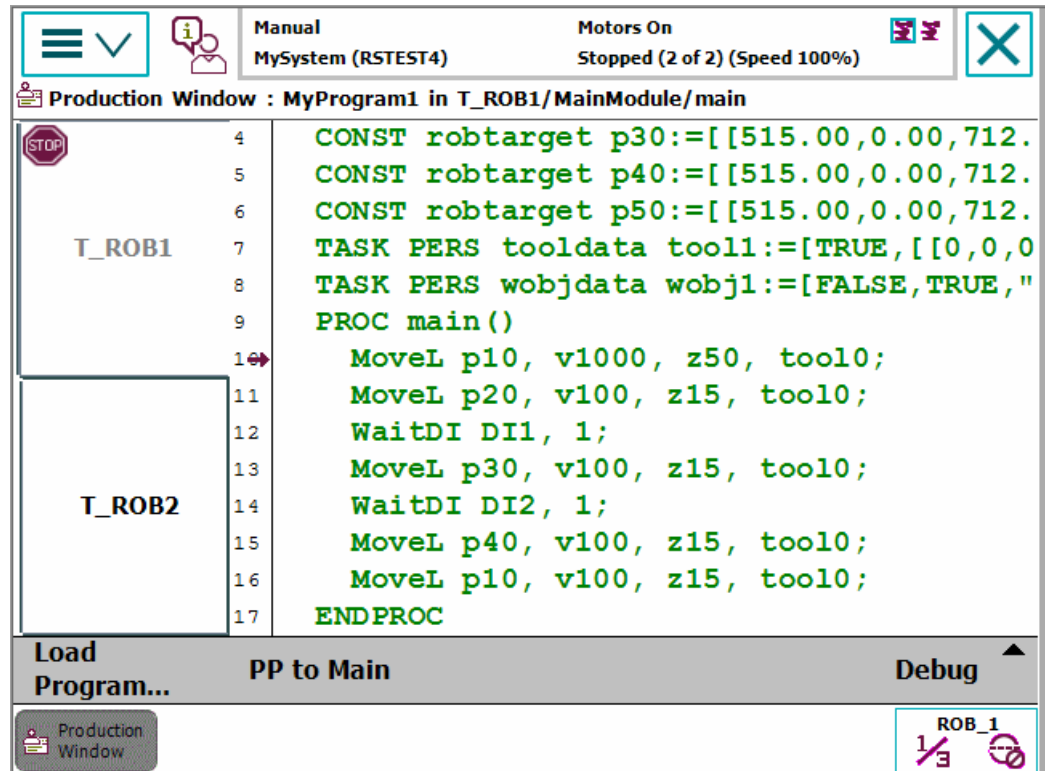
2.2.5 Production Window

Overview


The **Production Window** is used to view the program code while the program is running.

Illustration Production Window

This section illustrates the **Production Window**.



en0400000955

Load Program	Load a new program.
PP to Main	Move the program pointer to the routine Main.
Debug	<p>The Debug menu is available only in manual mode. Modify Position, see Modifying positions in the Program Editor or Production Window on page 249. Show Motion Pointer and Show Program Pointer, see About the Program and Motion Pointers on page 144. Edit Program, see Program Editor on page 43.</p> <p> Note</p> <p>The Visual Step mode icon indicates that the Step Into mode is not selected. This means that if the step mode is Step Out, Step Over, or Next Move the Visual Step mode icon is displayed.</p>

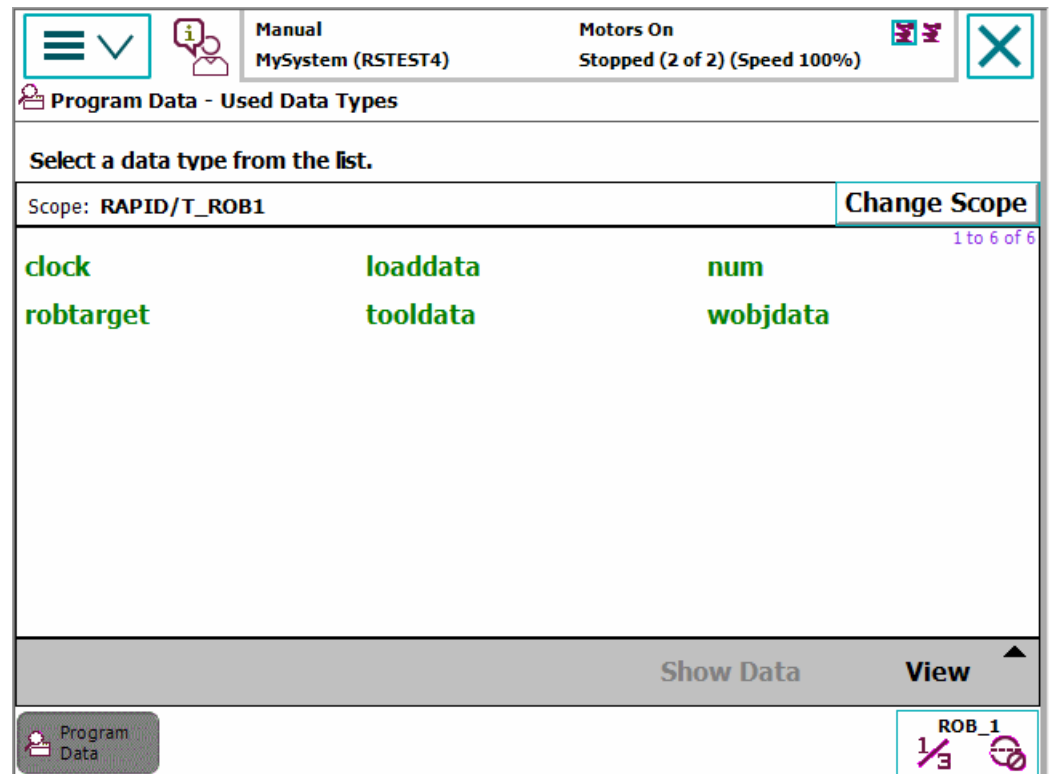
2.2.6 Program Data

Overview

The **Program Data** view contains functions for viewing and working with data types and instances. You can open more than one window of the **Program Data**, which can be useful when working with many instances or data types.

Illustration of Program Data

This section illustrates the **Program Data** view.



en040000659

Change Scope	Changes scope of data types in the list, see Viewing data in specific tasks, modules, or routines on page 145 .
Show Data	Shows all instances of the selected data type.
View	Shows all or only used data types.

Continues on next page

2 Navigating and handling FlexPendant

2.2.6 Program Data

Continued

Illustration of a data type instances

This section illustrates a list of instances for a data type.

Data of type: wobjdata

Select the data you want to edit. **Active filter:**

Scope: RAPID/T_ROB1 **Change Scope**

Name	Value	Module	1 to 2 of 2
wobj0	[FALSE,TRUE,"",[[0...	BASE	Global
wobj1	[FALSE,TRUE,"",[[0...	MainModule	Task

Program Data **ROB_1**

en0500001571

Filter	Filters the instances, see Filtering data on page 72 .
New	Creates a new instance of the selected data type, see Creating new data instance on page 146 .
Edit	Edits the selected instances, see Editing data instances on page 148 .
Refresh	Refreshes the list of instances.
View Data Types	Returns to the Program Data menu.

2.2.7 Program Editor

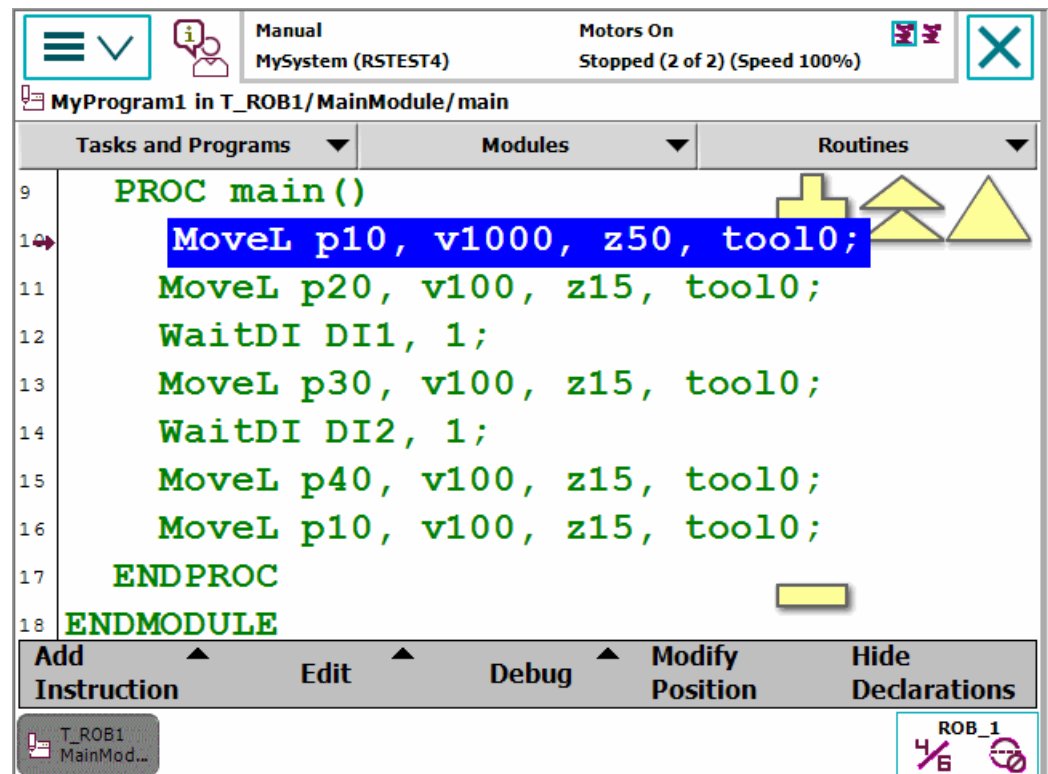
Overview

The **Program Editor** is where you create or modify programs. You can open more than one window of the **Program Editor**, which can be useful with the *Multitasking* option installed.

The **Program Editor** button in the task bar displays the name of the task.

Illustration of Program Editor

This section illustrates the **Program Editor** view.



en0400001143

Tasks and Programs	Menu for program operations, see Handling of programs on page 128 .
Modules	Lists all modules, see Handling of modules on page 131 .
Routines	Lists all routines, see Handling of routines on page 134 .
Add Instruction	Opens instruction menu, see Handling of instructions on page 139 .
Edit	Opens edit menu, see Handling of instructions on page 139 .
Debug	<p>Functions for moving the program pointer, service routines etc., see Running a service routine on page 196, and About the Program and Motion Pointers on page 144.</p> <p>Functions to search routines and view system data.</p> <ul style="list-style-type: none"> Search Routine: Searches all routines across all the modules (except hidden routines). View System Data: Displays all the tasks.

Continues on next page

2 Navigating and handling FlexPendant

2.2.7 Program Editor

Continued

Modify Position	See Modifying positions in the Program Editor or Production Window on page 249 .
Hide Declarations	Hides declarations to make the program code easier to read.

Automatically activate mechanical unit for jogging

If *Multitasking* is installed with more than one mechanical unit and more than one motion task, then when switching between **Program Editor** windows the selection of mechanical unit for jogging is not effected. This means when jogging then the last used mechanical unit will move, which not necessarily is the one used in the active **Program Editor**.

This setting can be changed with system parameters of the type *Automatically Switch Jog Unit* in the topic *Man-machine Communication*. Turn this setting on to automatically activate the mechanical unit last used in a **Program Editor** when switching to that window. This means that when jogging, the mechanical unit last used in the active **Program Editor** moves. Note that when switching between **Program Editors** in the same task, there is no change.

Mechanical units are manually activated for jogging in the **Jogging** window or in the Quickset menu, see [Selecting mechanical unit for jogging on page 110](#).

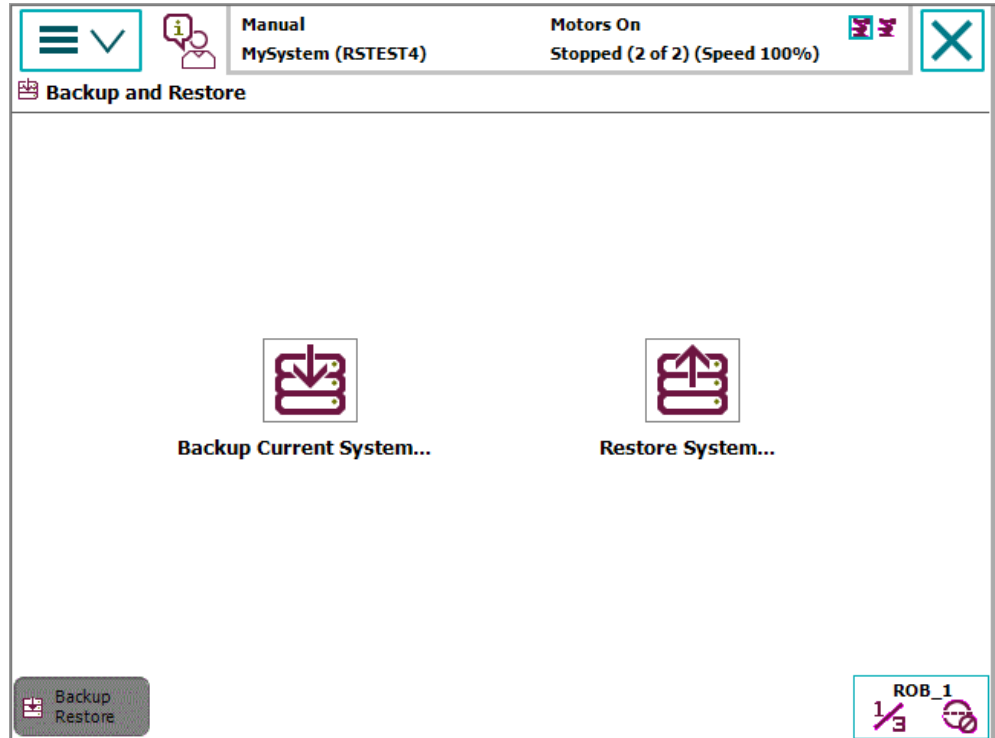
2.2.8 Backup and Restore

About backups

The **Backup and Restore** menu is used for performing backups and restoring the system. See section [Backup and restore on page 273](#).

Illustration of Backup and Restore

This is the **Backup and Restore** menu.



xx030000440

Backup Current System	See Back up the system on page 273 .
Restore System	See Restore the system on page 277 .

2 Navigating and handling FlexPendant

2.2.9 Calibration

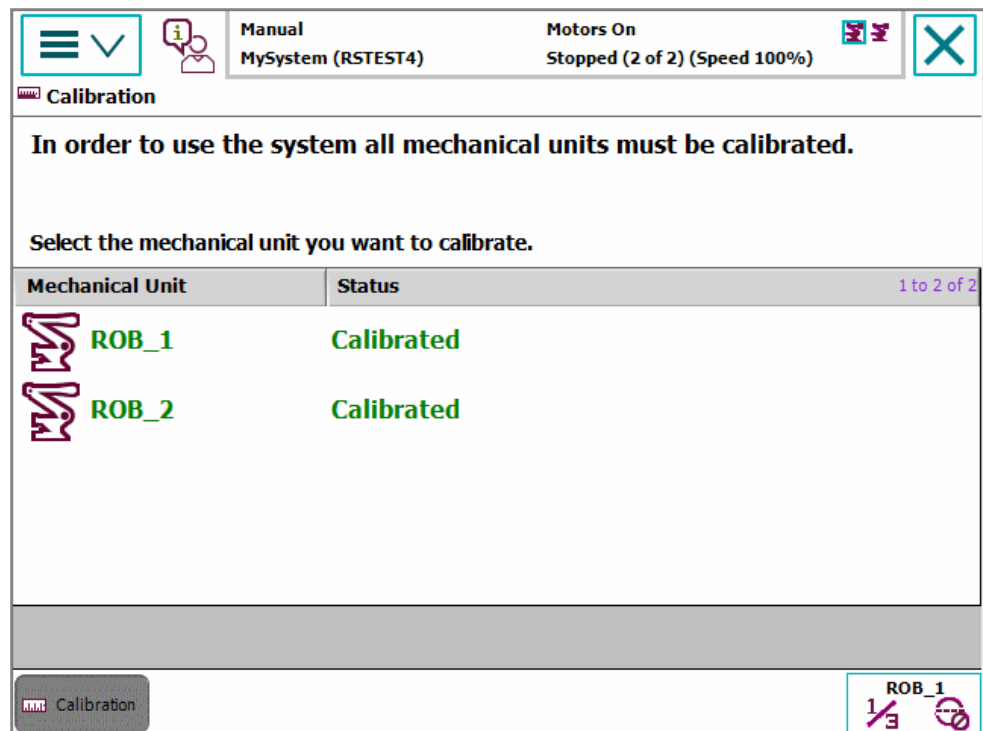
2.2.9 Calibration

About calibration

The **Calibration** menu is used to calibrate mechanical units in the robot system. Calibration can be performed using the option *Calibration Pendulum*. See *Operating manual - Calibration Pendulum*.

Illustration of Calibration menu

This illustration shows the **Calibration** menu. All mechanical units are listed and their calibration status is displayed in the **Status** column.

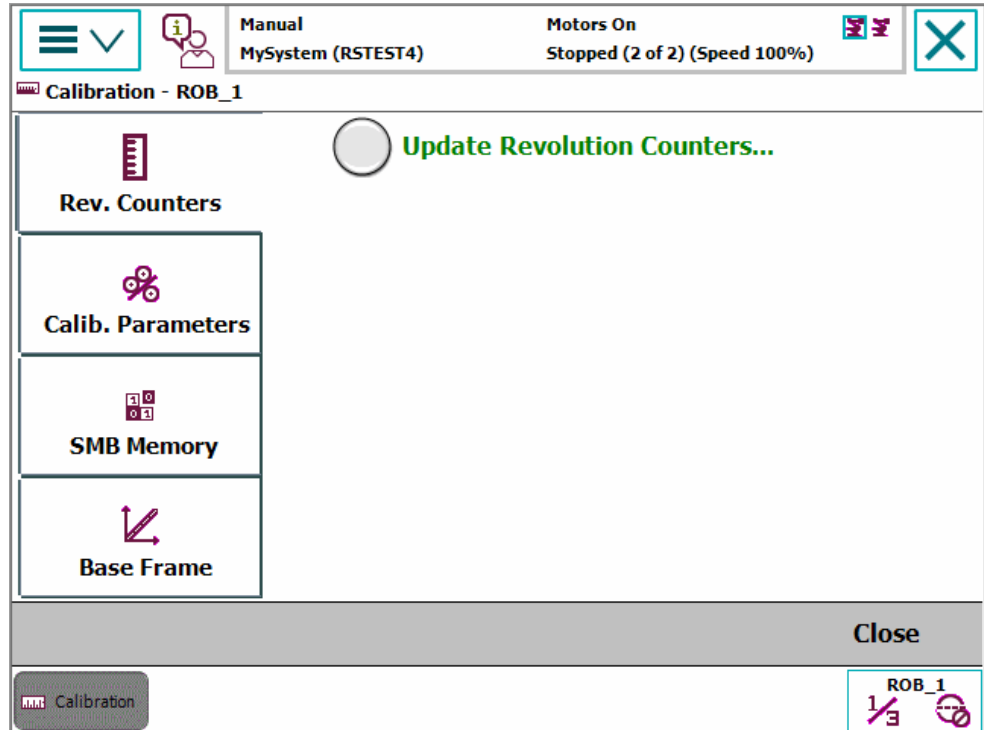


en0400001146

Continues on next page

Calibration menu options

This illustration shows the **Calibration** menu options after selecting mechanical unit.



en040000771

2 Navigating and handling FlexPendant

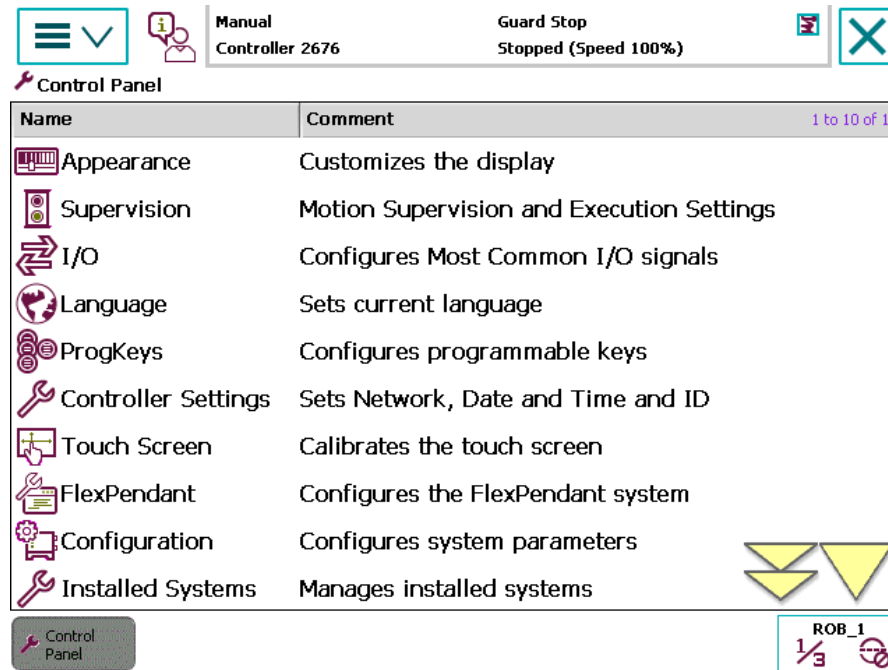
2.2.10 Control Panel

2.2.10 Control Panel

Control Panel

The **Control Panel** contains functions for customizing the robot and the FlexPendant.

Illustration Control Panel



en040000914

Appearance	Settings to customize the brightness of the display. See Changing brightness and contrast on page 88 .
Supervision	Settings for motion supervision and execution settings. See Using motion supervision and non motion execution on page 231 . When changing the collision detection level, it is necessary to go to motors off state to activate the changes.
I/O	See Configuring Most Common I/O on page 93 .
Language	See Changing language on page 94 .
ProgKeys	See Changing programmable keys on page 95 .
Controller Settings	Settings for configuring the network, the date and time for the robot controller, and the ID of the controller. See Controller settings on page 91 .
Touch Screen	See Calibrating the touchscreen on page 97 .
FlexPendant	Configuration of views for operating mode switch and UAS, User Authorization System. See Defining a view to be shown during operating mode change or startup on page 80 .
Configuration	Configuration of the system parameters configuration.

2.2.11 Event Log

The Event Log

The event log stores information about past events for future reference to facilitate troubleshooting.

How to open the event log is described in [Accessing the event log on page 269](#).

Illustration Event Log

Event Log - Common

Tap a message to open it.

Code	Title	Date & Time
10002	Program pointer has been reset	2014-08-06 11:17:19
10011	Motors ON state	2014-08-06 11:14:47
10010	Motors OFF state	2014-08-06 11:14:46
10002	Program pointer has been reset	2014-08-06 11:14:08
10002	Program pointer has been reset	2014-08-06 11:11:19
10015	Manual mode selected	2014-08-06 11:10:55
10012	Safety guard stop state	2014-08-06 11:10:55
10011	Motors ON state	2014-08-06 11:10:50
10017	Automatic mode confirmed	2014-08-06 11:10:49

Save All Logs As... Delete Update View

Program Data I/O ROB_1

xx030000447

Function	Description
View a message	Tap the message. The message structure is described in An event log message on page 50 .
Scroll or zoom a message	See Scrolling and zooming on page 71 .
Delete the log	See Deleting log entries on page 270 .
Save the log	See Saving log entries on page 271 .
Close the log	See Accessing the event log on page 269 .

Continues on next page

2 Navigating and handling FlexPendant

2.2.11 Event Log


Continued

An event log message

Each event log entry consists of a message describing the event in detail, and it often contains advice on how to solve the problem.

Event Log - Event Message

(A) Event Message 10002 (C) 2014-08-06 11:17:19


(B)  Program pointer has been reset

(D) Description
The program pointer of task T_ROB1 has been reset.

(E) Consequences
When started, program execution will start on the first instruction of the task's entry routine. NOTE that the manipulator may move to unexpected position when restarted!

(F) Probable causes
The operator has probably requested this action manually.

(G) Next Previous (H) OK



en0300000454

A	Event number. All errors are listed by numbers.
B	Event title. Briefly states what has happened.
C	Event time marker. Specifies exactly when the event occurred.
D	Description. A brief description of the event. Intended to assist in understanding the causes and implications of the event.
E	Consequences. A brief description of any consequences inflicted on the system, transition to other operation mode, emergency stop, caused by the particular event. Intended to assist in understanding the causes and implications of the event.
F	Probable causes. A list of probable causes, listed in order of probability.
G	Recommended actions. A list of the recommended correcting actions, based on the Probable causes specified above. These may range from <i>Replace the xx...</i> to <i>Run test program xx...</i> , that is, may be actions to isolate the problem as well as fixing it.
H	Acknowledge or OK button.

Related information about logs

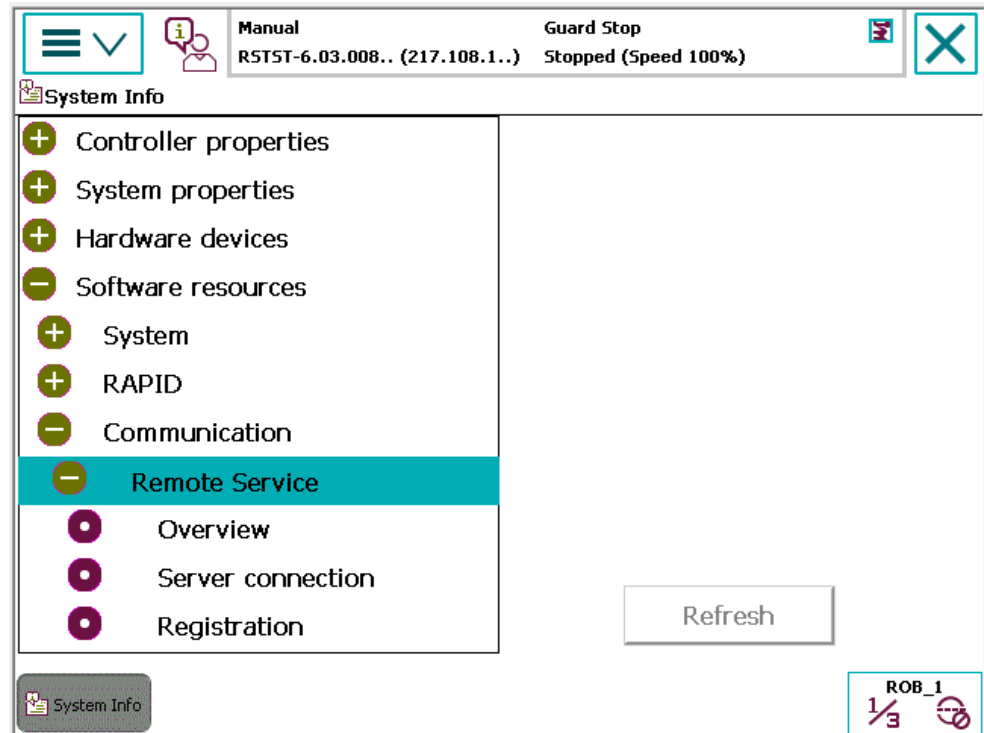
Event log messages and more information about the event log are described in *Operating manual - Troubleshooting IRC5*.

2.2.12 System Info

About System Info

System Info displays information about the controller and the loaded system. Here you can find the RobotWare version and options currently in use, current keys for control and drive modules, network connections, and so on.

Illustration of System Info view



en040000968

Controller properties

It contains controller and network information. When the **Controller properties** is expanded the following are visible:

Network connections	Service port and Local Area Network properties.
Installed systems	List of installed systems.

System properties

It contains information of the system that is currently in use. When the **System properties** is expanded the following are visible:

Control module	Name and key of the Control Module.
Options	Installed RobotWare options and languages.
Drive modules	Lists all Drive Modules.
Drive module x	Name and key of Drive Module x.
Options	Drive Module x options, with type of robot and so on.
Additional options	Any RobotWare options and Process application options.

Continues on next page

2 Navigating and handling FlexPendant

2.2.12 System Info

Continued

Hardware devices

It contains information of all the hardware attached. When the **Hardware devices** is expanded the following are visible:

Controller	Name and key of the Control Module.
Computer system	Contains information of the main computer.
Power supply system	Contains information of the Power supply unit.
Panel board	Provides information about the panel board hardware and software.
Drive module x	It contains information about the axis computer, drive unit, and contactor board.
Mechanical units	Lists the robots or external axes that are connected to the controller.
General SIS data	Lists the service information system data for the connected mechanical units. For more information regarding the SIS data, see <i>Operating manual - Service Information System</i> .

Software resources

It contains information about the RAPID. When the **Software resources** is expanded the following are visible:

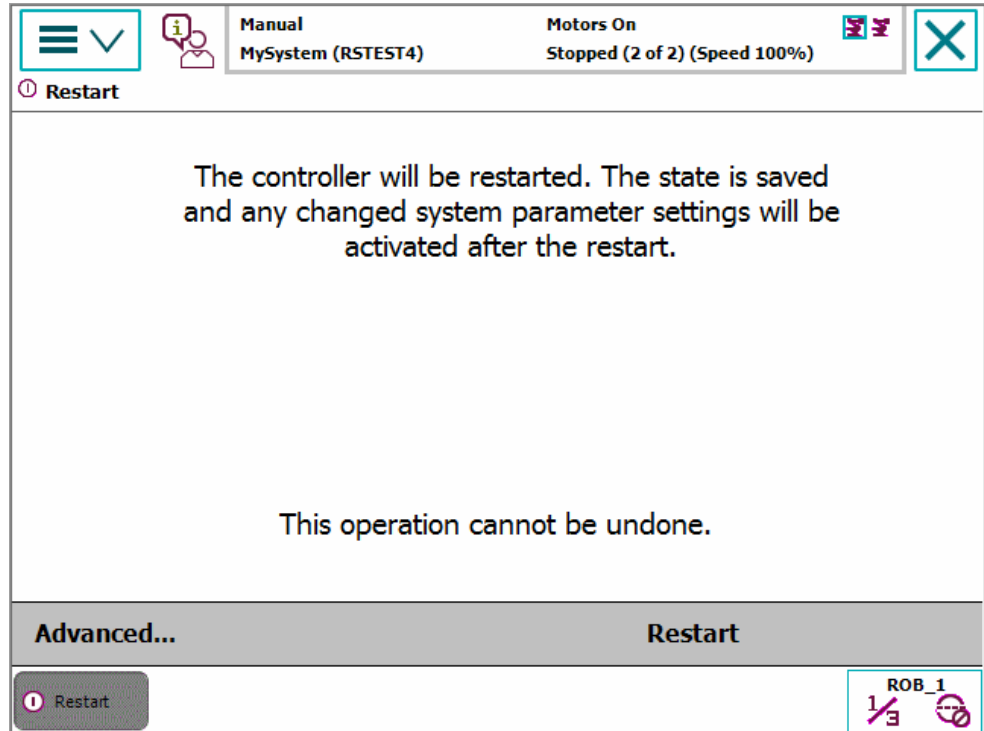
System	Contains information about system uptime and memory.
RAPID	Software used by the controller.
RAPID memory	Memory allocated for RAPID programs.
RAPID performance	Shows the execution load.
Services	Contains information about additional services.
Communication	Contains information about Remote Service Embedded.

2.2.13 Restart

Restart

A running system normally does not need to be restarted.

Tap the **ABB** menu and then **Restart** to restart the system.



en0500001557

2 Navigating and handling FlexPendant

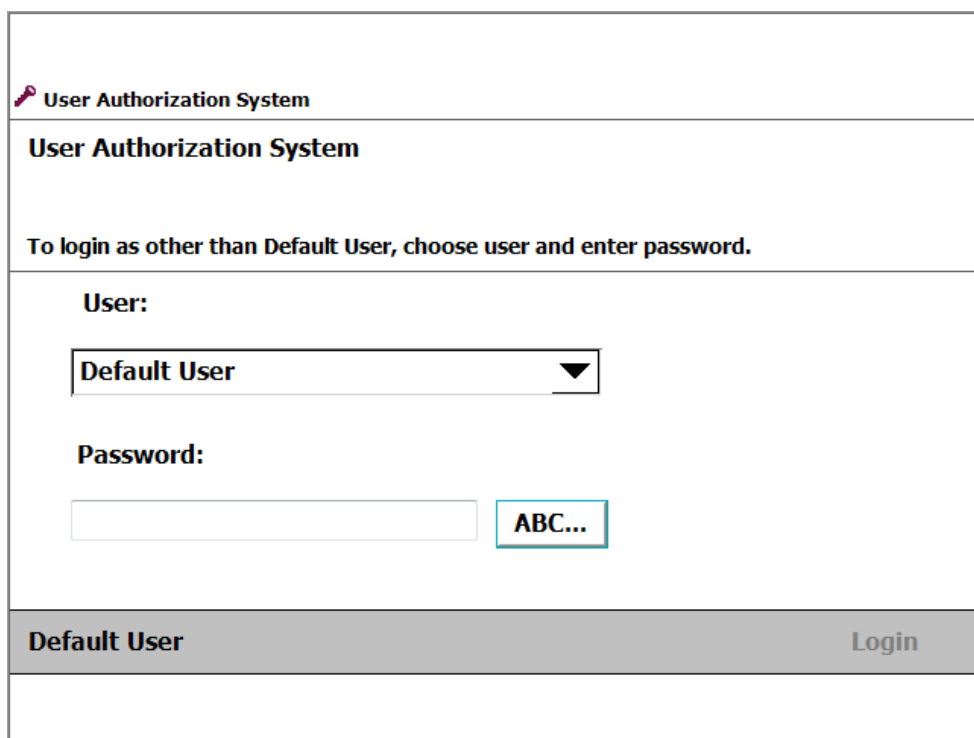
2.2.14 Log Off

2.2.14 Log Off

The Log Off menu

This section details the Log Off menu. More about using this menu is described in [Logging on and off on page 77](#).

Log Off is available under the ABB menu.



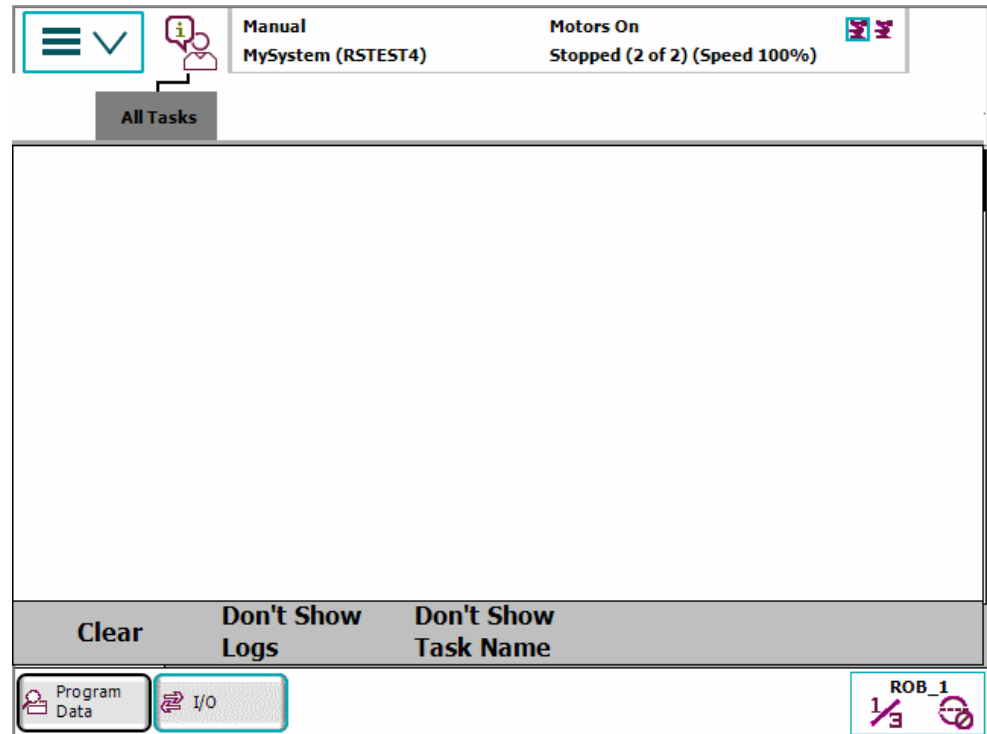
The screenshot shows a web interface for the User Authorization System. At the top, there is a key icon and the text "User Authorization System". Below this, the title "User Authorization System" is displayed. A message states: "To login as other than Default User, choose user and enter password." The form includes a "User:" label followed by a dropdown menu currently showing "Default User". Below that is a "Password:" label followed by a text input field and a button labeled "ABC...". At the bottom of the form, there is a grey bar with "Default User" on the left and "Login" on the right.

en0400000947

2.3 Operator window

Operator window

The operator window displays messages from the program. With *Multitasking* installed, all tasks' messages are displayed in the same operator window. If a message requires action then a separate window for that task will be displayed. The operator window is opened by tapping the icon to the right of the ABB logo in the status bar. The illustration shows an example of an operator window:



en0400000975

Clear	Clears all messages
Don't Show Logs	Clears all messages
Don't Show Task Name	Hides task names

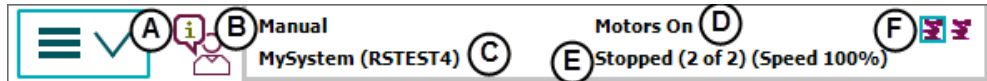
2 Navigating and handling FlexPendant

2.4 Status bar

2.4 Status bar

Illustration of status bar

The Status bar displays information about the current status, such as operational mode, system, and active mechanical unit.



en0300000490

A	Operator window
B	Operating mode
C	System name (and controller name)
D	Controller state
E	Program state
F	Mechanical units. The selected unit (and any unit coordinated with the selected) is marked with a border. Active units are displayed in color, while deactivated units are grey.

2.5 Quickset

2.5.1 The Quickset menu

Quickset menu

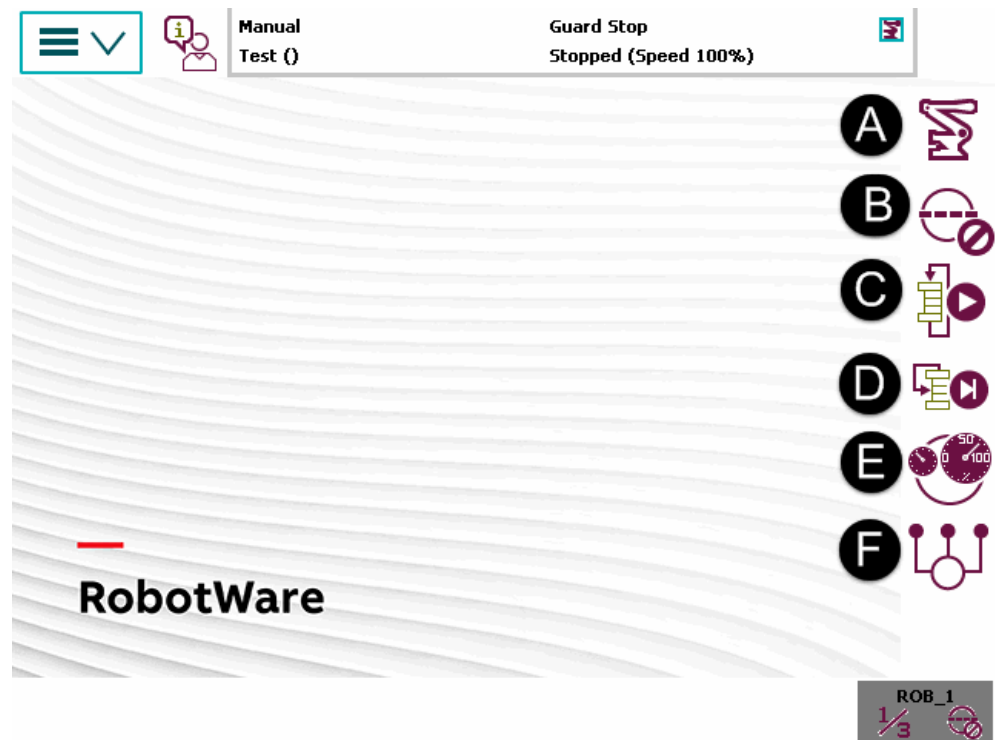
The QuickSet menu provides a quicker way to change among other things jog properties rather than using the **Joggingview**.

Each button on the menu shows the currently selected property value or setting.

In manual mode, the Quickset menu button shows the currently selected mechanical unit, motion mode, and increment size.

Illustration of the Quickset menu

This section describes the buttons in the Quickset menu.



en0300000471

A	Mechanical unit, see Quickset menu, Mechanical unit on page 58 .
B	Increment, see Quickset menu, Increment on page 63 .
C	Run Mode, see Quickset menu, Run Mode on page 64 .
D	Step Mode, see Quickset menu, Step Mode on page 65 .
E	Speed, see Quickset menu, Speed on page 66 .
F	Tasks, see Quickset menu, Tasks on page 67 .

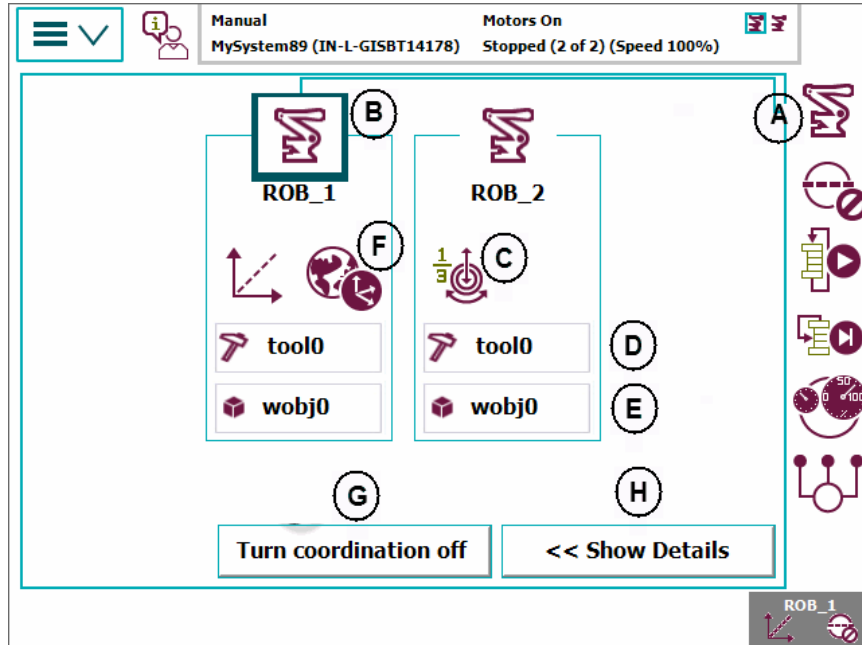
2 Navigating and handling FlexPendant

2.5.2 Quickset menu, Mechanical unit

2.5.2 Quickset menu, Mechanical unit

Illustration Mechanical unit button

On the Quickset menu, tap **Mechanical unit**, then tap to select a mechanical unit.



en0300000539

A	Mechanical unit menu button
B	Mechanical unit, a selected unit is highlighted. See Selecting mechanical unit for jogging on page 110 .
C	Motion mode settings (axes 1-3 motion mode currently selected), further settings are described in Illustration Motion mode settings on page 59 .
D	Tool settings (tool 0 currently selected), further settings are described in Illustration Tool settings on page 60 .
E	Work object settings (work object 0 currently selected), further settings are described in Illustration Work object settings on page 60 .
F	Coordinate system settings (world coordinate currently selected), further settings are described in Illustration Coordinate system settings on page 61 .
G	Turn coordination off , further settings are described in Turn coordination off on page 62 .
H	Show details , further settings are described in Illustration Show Details on page 62 .



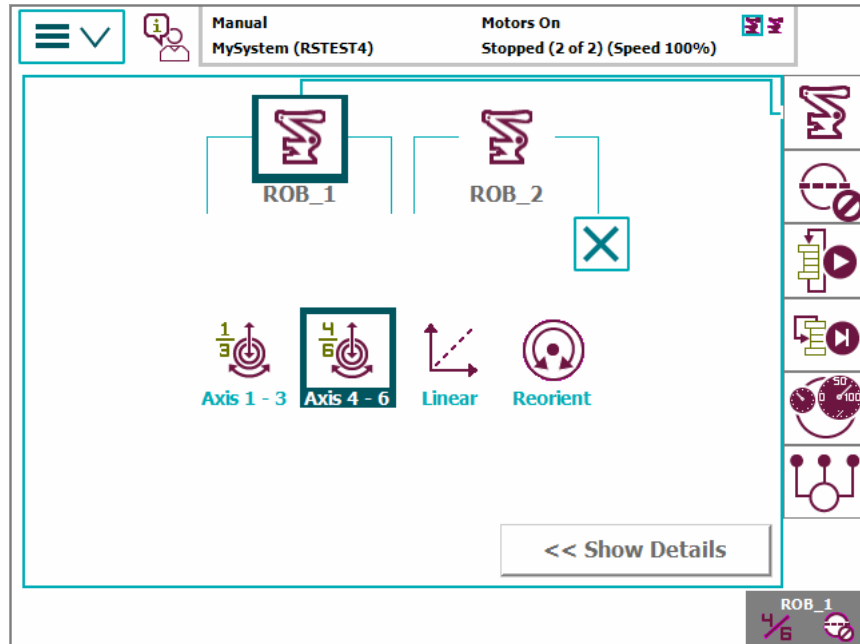
Note

The Mechanical unit menu is only available in manual mode.

Continues on next page

Illustration Motion mode settings

To view or change any motion mode functionality, tap the **Motion mode settings** button. These settings are also available in the Jogging window, see [Selecting motion mode on page 112](#).



en0300000540

Select motion mode setting:

- Axis 1-3
- Axis 4-6
- Linear
- Reorient

Continues on next page

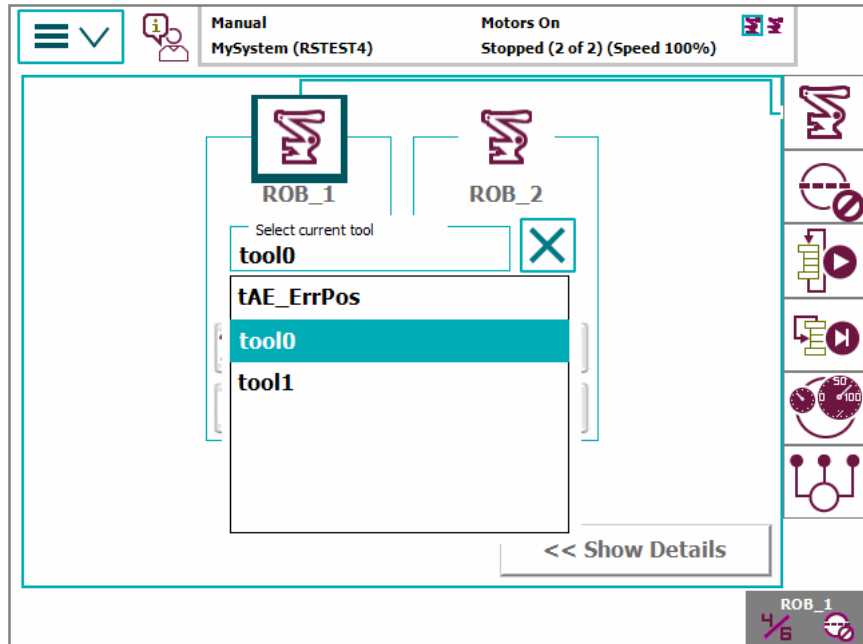
2 Navigating and handling FlexPendant

2.5.2 Quickset menu, Mechanical unit

Continued

Illustration Tool settings

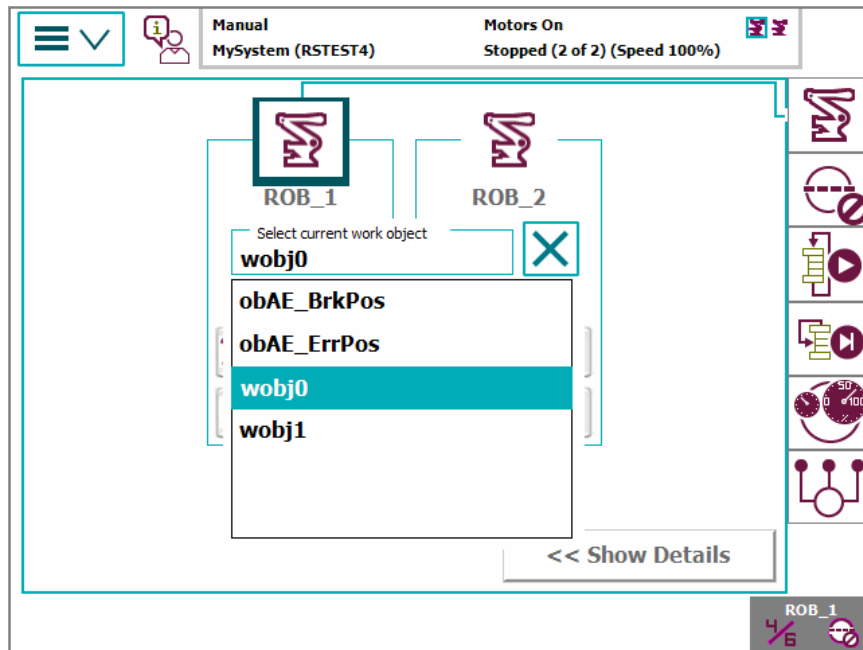
To view or change the available tools, tap the **Tool settings** button. These settings are also available in the Jogging window, see [Selecting tool, work object, and payload on page 113](#).



en0400000988

Illustration Work object settings

To view or change the available work objects, tap the **Work object settings** button. These settings are also available in the Jogging window, see [Selecting tool, work object, and payload on page 113](#).



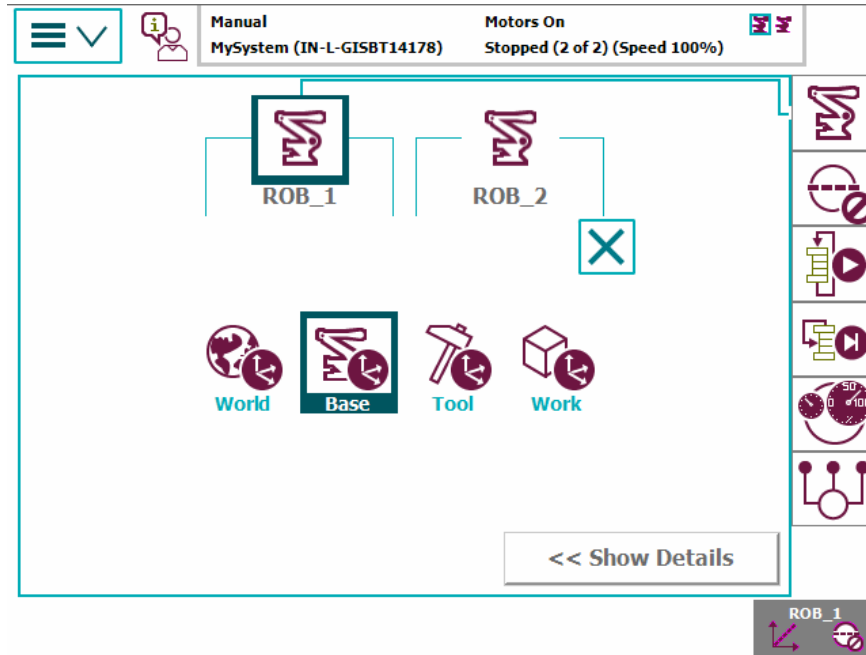
en0400000989

Continues on next page

Select a work object to use.

Illustration Coordinate system settings

To view or change coordinate system functionality, tap the **Coordinate system settings** button. These settings are also available in the Jogging window, see [Selecting coordinate system on page 116](#).



en0300000541

Select a coordinate system setting:

- World coordinate system
- Base coordinate system
- Tool coordinate system
- Work object coordinate system

Continues on next page

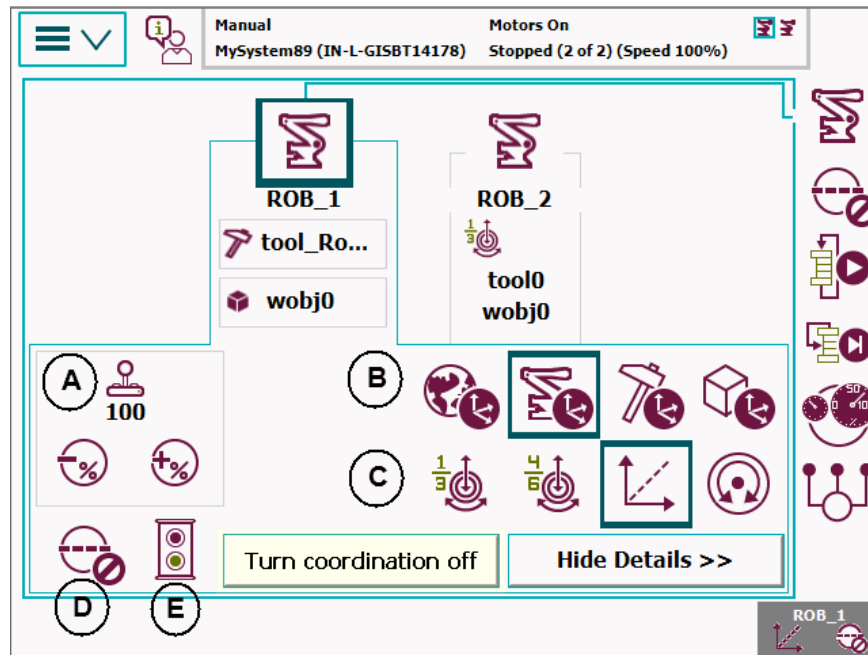
2 Navigating and handling FlexPendant

2.5.2 Quickset menu, Mechanical unit

Continued

Illustration Show Details

Tap Show Details to display the settings available for a mechanical unit.



en0500002354

A	Override jog speed settings (100% currently selected)
B	Coordinate system settings
C	Motion mode settings
D	Turn on or off user increment
E	Turn on or off jog supervision

If any of the settings are not available, they will be crossed over.

Motion mode and coordinate settings can be changed by tapping the buttons.

Tap **Hide Details** to return to the basic display.

Turn coordination off

To quickly change between coordinated and uncoordinated jogging, use the coordination off button.

The button is automatically hidden when you change anything that affects coordination, for example the work object or the coordinate system of the coordinated mechanical unit.

To re-enable the button you must setup coordination again manually.

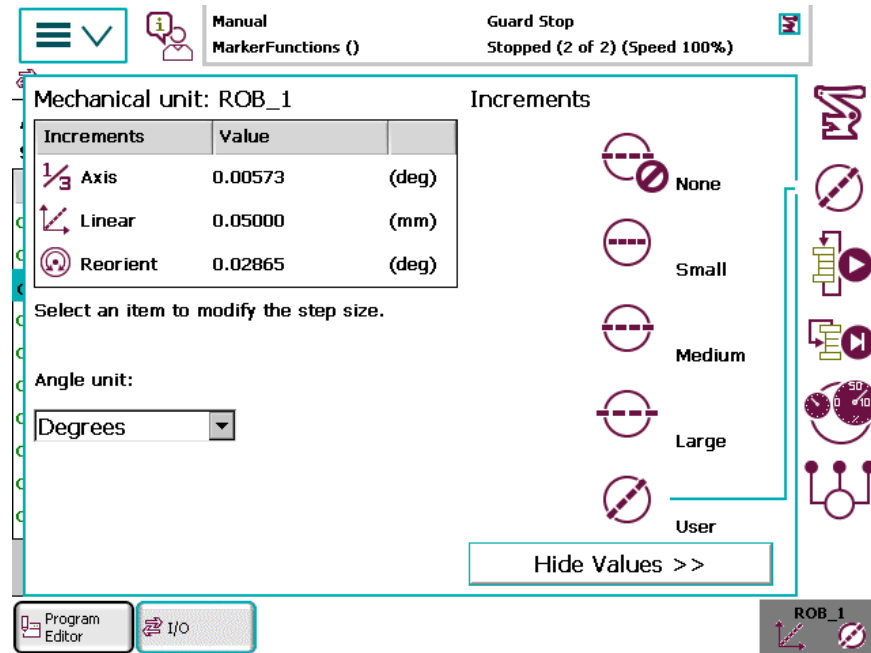
For information about coordination between MultiMove robots, see *Application manual - MultiMove*.

2.5.3 Quickset menu, Increment

Increment settings

The increment settings are also available in the Jogging window. For more details, see [Incremental movement for precise positioning on page 119](#).

Illustration Increment



en0300000542

None	No increments
Small	Small movements
Medium	Medium movement
Large	Large movements
User	User defined movements
Show/Hide Values	Displays/hides increment values
Angle unit	Defines the unit for the angles.



Note

The Increment menu is available only in manual mode.

2 Navigating and handling FlexPendant

2.5.4 Quickset menu, Run Mode

2.5.4 Quickset menu, Run Mode

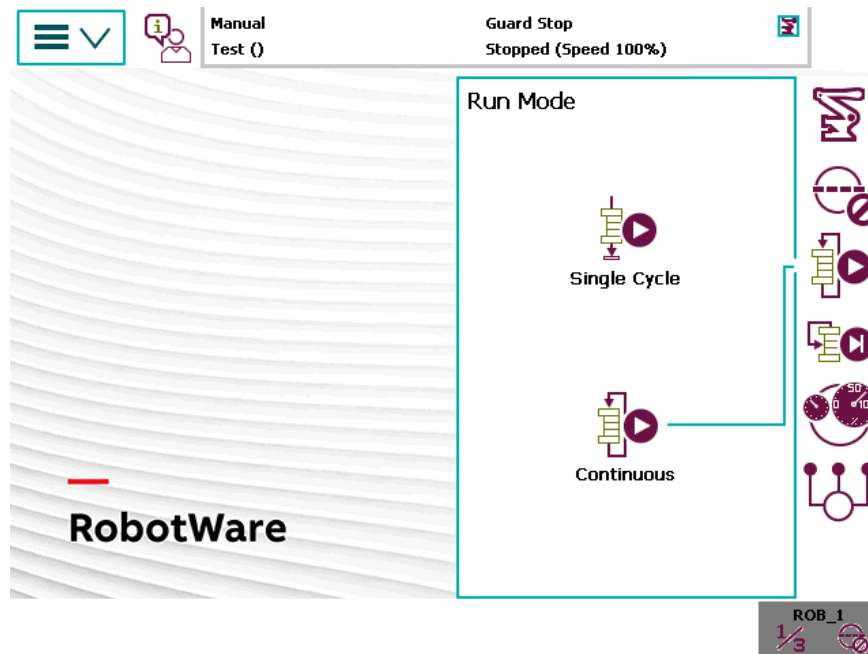
Run mode

By setting run mode you define if the program execution should run once and then stop, or run continuously.

For information about run mode in:

- *Multitasking*, see *Application manual - Controller software IRC5*, section *Multitasking*.
- *MultiMove*, see *Application manual - MultiMove*, section *User interface specific for Multimove*.

Illustration Run mode



en0300000472

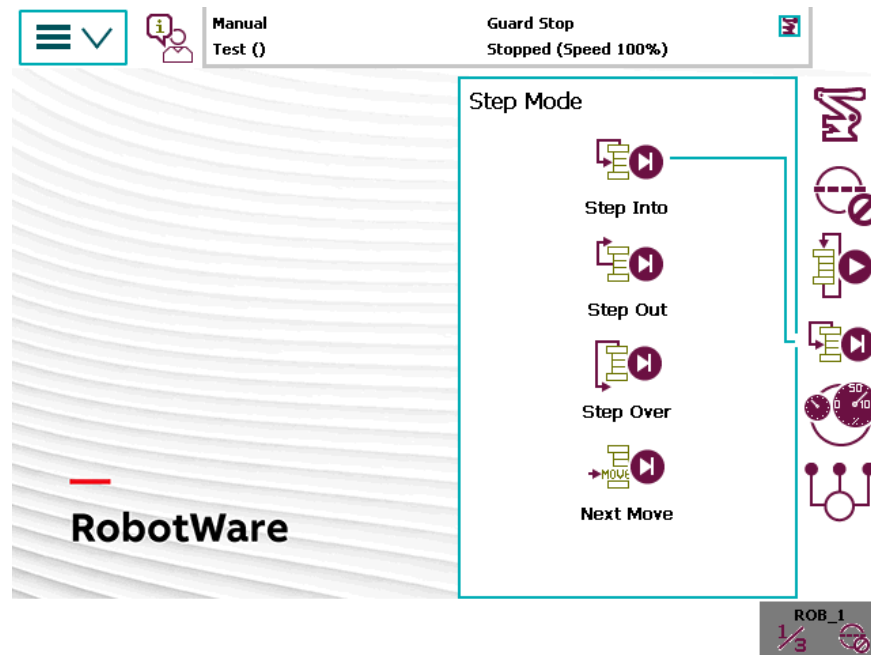
Single Cycle	Runs one cycle then stops execution.
Continuous	Runs continuously.

2.5.5 Quickset menu, Step Mode

Step mode

By setting the step mode you can define how the step-by-step program execution should function. For more details, see [Stepping instruction by instruction on page 193](#)

Illustration Step mode



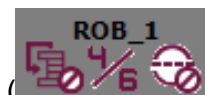
en0300000543

Step Into	Steps into called routines and executes them step-by-step.
Step Out	Executes the remains of the current routine and then stops at the next instruction in the routine from which the current routine was called. Not possible to use in the Main routine.
Step Over	Called routines are executed in one single step.
Next Move	Steps to the next move instruction. Stops before and after movement instructions, for example, to modify positions.



Note

If you select the **Step Out/Step Over/Next Move** step modes, an arrow icon



xx1500001580

) is also displayed in the **QuickSet** menu indicating that the selected mode is not a **Step Into** step mode.

2 Navigating and handling FlexPendant

2.5.6 Quickset menu, Speed

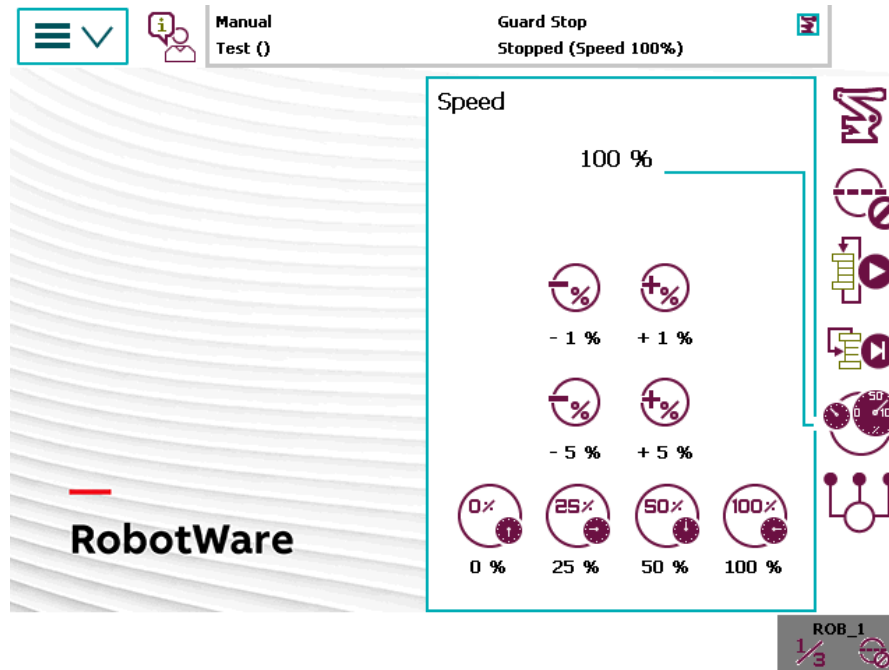
2.5.6 Quickset menu, Speed

Speed button

The speed settings apply to the current operating mode. However, if you decrease the speed in automatic mode, the setting also applies to manual mode if you change mode.

Illustration Speed

Tap the **Speed** button to view or change the speed settings. The current running speed, in relation to max, is displayed above the buttons.



en030000470

-1%	Decrease running speed in steps of 1%
+1%	Increase running speed in steps of 1%
-5%	Decrease running speed in steps of 5%
+5%	Increase running speed in steps of 5%
0%	Set speed to 0%
25%	Run at quarter speed (25%)
50%	Run at half speed (50%)
100%	Run at full speed (100%)

2.5.7 Quickset menu, Tasks

Tasks button

If you have the option *Multitasking* installed there can be more than one task. Otherwise there is only one task.

By default, you can activate/deactivate only **Normal** tasks from the **Quickset** menu. However, from the **Control Panel**, you can change the settings so that **All tasks** can be activated/deactivated.

Activated tasks are started and stopped with the **Start** and **Stop** buttons on the FlexPendant.

The tasks settings are only valid for manual operating mode.

Related information

Application manual - Controller software IRC5, section Multitasking.

How to start and stop multitasking programs is described in section [Using multitasking programs on page 229](#).

TrustLevel for tasks are set with system parameters, see section *Task* in *Technical reference manual - System parameters*.

You can define if all tasks or only normal tasks should be displayed. See section [Defining which tasks should be selectable in the tasks panel on page 86](#).

2 Navigating and handling FlexPendant

2.6.1 Using the soft keyboard

2.6 Basic procedures

2.6.1 Using the soft keyboard

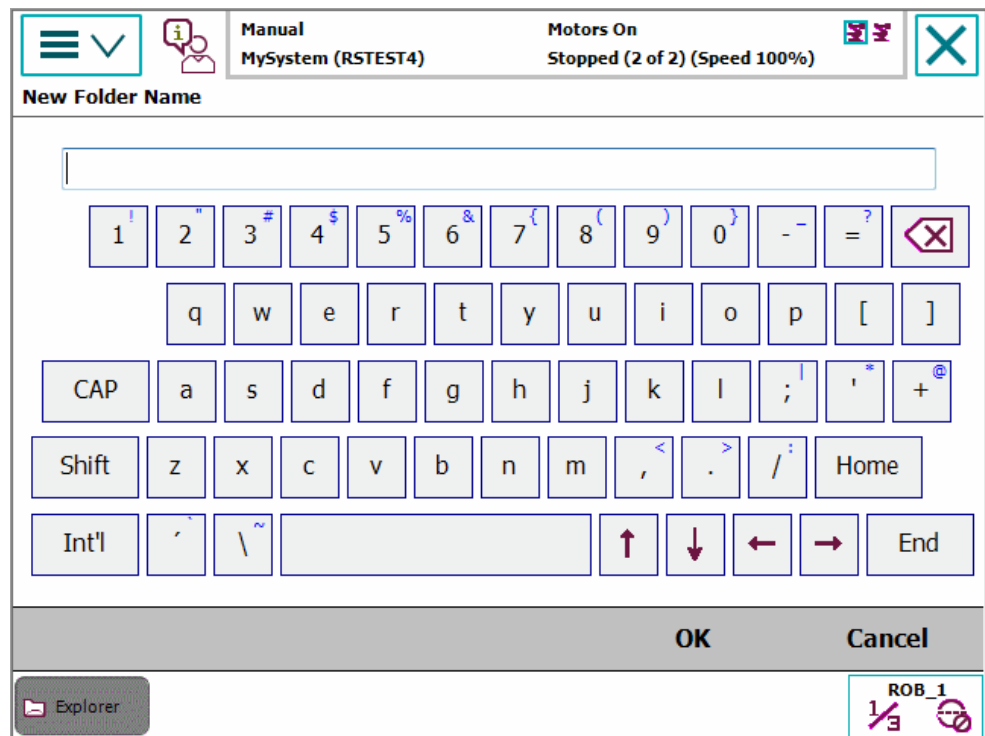
Soft keyboard

The soft keyboard is used frequently when operating the system, for example when entering file names or parameter values.

The soft keyboard works as an ordinary keyboard with which you can place the insertion point, type and correct typing errors. Tap letters, numbers and special characters to enter your text or values.

Illustration soft keyboard

This illustration shows the soft keyboard on the FlexPendant.



en0300000491



Using international characters

All western characters can be used, also in usernames and passwords. To access international characters, tap the **Int'l** button on the soft keyboard.

Continues on next page

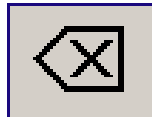
Changing the insertion point

Tap the arrow keys to change the insertion point, for instance when correcting typing errors.

If you need to move...	then tap...
backward	 xx0300000492
forward	 xx0300000493

Deleting

- 1 Tap the **Backspace** key (top right) to delete characters to the left of the insertion point.



xx0300000494

2 Navigating and handling FlexPendant

2.6.2 Messages on the FlexPendant

2.6.2 Messages on the FlexPendant

Overview of messages

The FlexPendant displays messages from the system. These can be status messages, error messages, program messages, or requests for action from the user. Some require actions, and some are plain information.

Event log messages

The event log messages are messages from the RobotWare system about system status, events, or errors.

How to work with the event log messages is described in section [Handling the event log on page 269](#). All messages are also described in *Operating manual - Troubleshooting IRC5*.

System messages

Some messages sent out by the system are not from the event log. They can come from other applications, such as RobotStudio.

To be able to change configurations and settings in the system from RobotStudio, the user must request write access. This generates a message on the FlexPendant where the operator can grant or deny access. The operator can at any time decide to withdraw the write access.

How to request access and work with RobotStudio is described in *Operating manual - RobotStudio*.

Program messages

RAPID programs can send out messages to the Operator window, see section [Operator window on page 55](#).

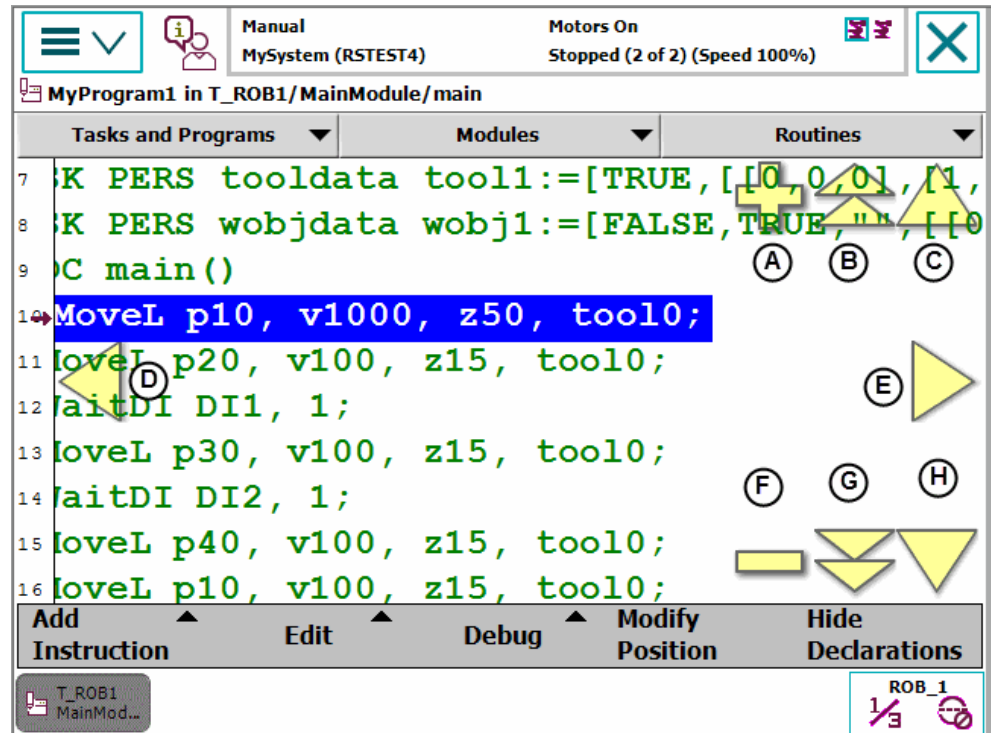
How to generate program messages is described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

2.6.3 Scrolling and zooming

Overview

The entire contents of a screen might not be visible at the same time. To see the entire contents, you can:

- Scroll up/down (and sometimes left/right)
- Zoom in or out (only available in the **Program Editor**)



en040000685

A	Zoom in (larger text)
B	Scroll up (the height of one <i>page</i>)
C	Scroll up (the height of one <i>line</i>)
D	Scroll left
E	Scroll right
F	Zoom out (smaller text)
G	Scroll down (the height of one <i>page</i>)
H	Scroll down (the height of one <i>line</i>)

2 Navigating and handling FlexPendant

2.6.4 Filtering data

2.6.4 Filtering data

Filtering data

In several of the FlexPendant menus you can use filtering. This can be useful when you are looking at instances of a data type, for example, and there are more available than is possible to overlook. By filtering instances starting with a specific character for example, the number can be greatly reduced.

Depending on the type of data, you can filter data either alphabetically or numerically.

Illustration of filtering

The filter function is switched on until the active filter is removed (For example, by tapping **Reset**).

The screenshot shows the FlexPendant interface. At the top, there is a status bar with 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this, the data type is 'robtarget'. The main area displays a table with columns 'Name' and 'Value'. The table lists five entries: p10, p20, p30, p40, and p50, all with values starting with '[[515,0'. An 'Active filter' overlay is present, showing a grid of letters A-Z and a 'Filter' button. The filter is currently set to 'ABC'. Below the table, there are buttons for 'New...', 'Edit', 'Refresh', and 'View Data Types'. At the bottom, there is a 'Program Data' button and a 'ROB_1' button.

Name	Value
p10	[[515,0
p20	[[515,0
p30	[[515,0
p40	[[515,0
p50	[[515,0

en0500001539

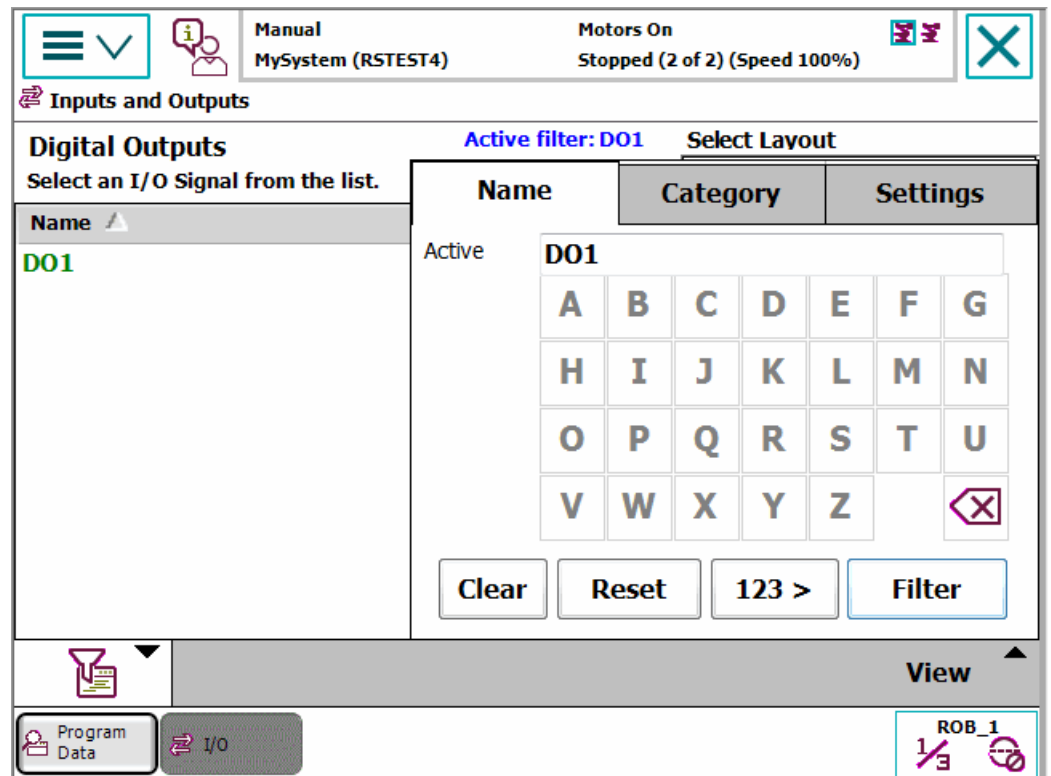


Note

When filtering I/O signals there are more options than for many other types of data. For example,


- you can filter data by **Name** or **Category**.
- filtering function can be displayed automatically if the number of signals displayed exceeds a predefined number. See [Illustration of automatic filter display on page 74](#).

Continues on next page



en120000669

Figure 2.1:

Active filter	Displays the current filter. It is also displayed at the top of the list of items.
Clear	Clears the text within the Active filter text box.
Reset	Removes the filter string.
123 / ABC	Depending on type of data, there may be one or several ways to filter data, e.g. numeric, alphabetic.
Filter	Applies the filter.
	Opens and closes the Active filter menu.

en110000506

Continues on next page

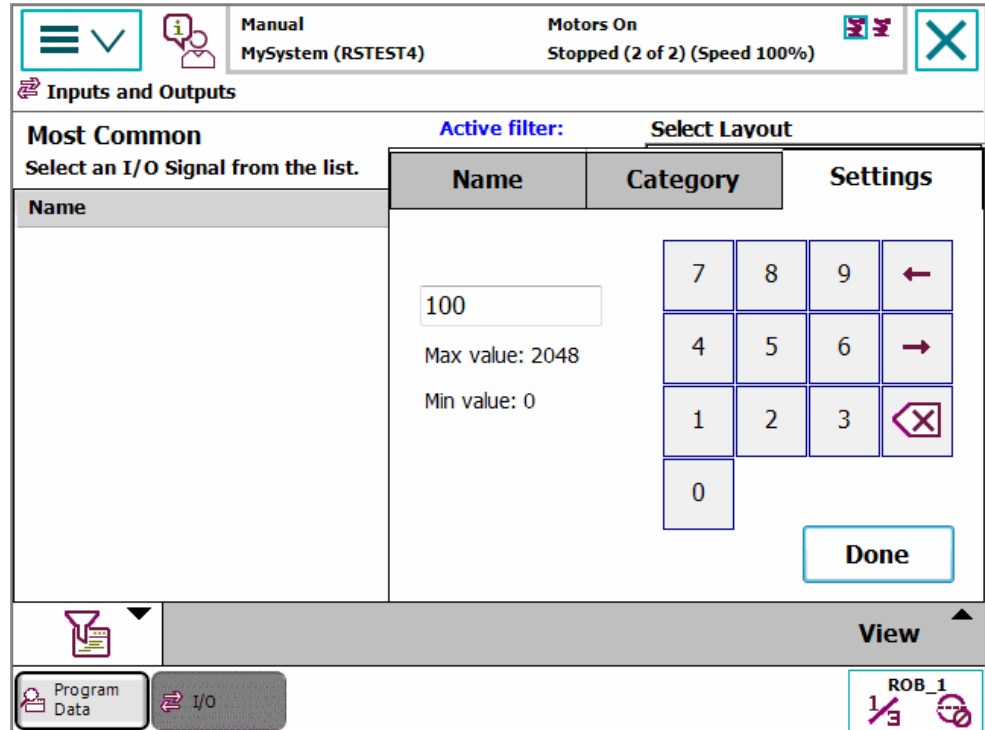
2 Navigating and handling FlexPendant

2.6.4 Filtering data

Continued

Illustration of automatic filter display

The I/O signal filter can be set to be displayed automatically if the number of data exceeds a predefined number.



en0600002643

	Action
1.	Tap Change to edit the setting controlling when the filter dialog should appear.
2.	Enter a new number defining the upper limit for not using the filter. Then tap Done .
3.	Tap Virtuals to select if all, or only virtual, or only non virtual signals should be listed.

2.6.5 Process applications

Process applications

Custom process applications are started from the ABB menu. Each application is listed as a menu item together with the FlexPendant views.

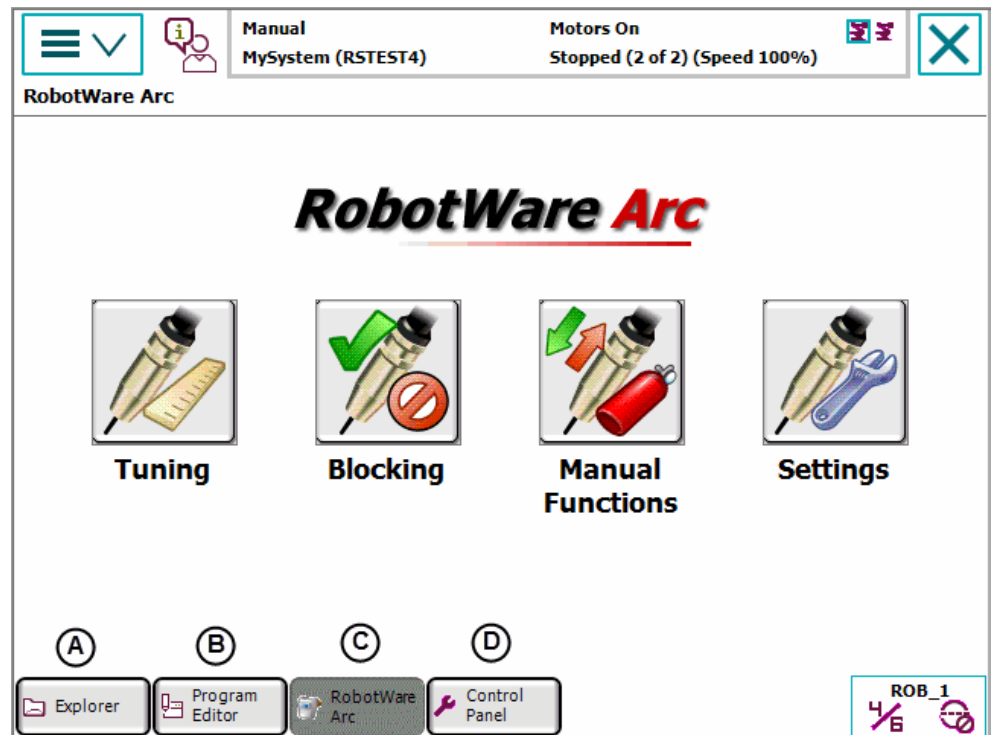
Start a process application

Use this procedure to start a process application.

	Action
1	Tap the ABB button to display the ABB menu. Process applications are listed in the menu.
2	Tap the name of the process application to start.

Switch between started process applications

A started application has a quick-button in the taskbar, just like FlexPendant views. Tap the buttons to switch between the started applications and views.



en0400000768

The views and process application running in this case are:

A	FlexPendant Explorer view
B	Program Editor view
C	RobotWare Arc, a process application
D	Control Panel view

2 Navigating and handling FlexPendant

2.6.6 Granting access for RobotStudio

2.6.6 Granting access for RobotStudio

About write access on the controller

The controller accepts *only* one user with write access at a time. Users in RobotStudio can request write access to the system. If the system is running in manual mode, the request is accepted or rejected on the FlexPendant.

Granting access for RobotStudio

This procedure describes how to grant access for RobotStudio.

	Action
1	When a user in RobotStudio requests access, a message is displayed on the FlexPendant. Decide whether to grant or reject access. If you want to grant access, then tap Grant . The user holds write access until he disconnects or until you reject the access. If you want to deny access, then tap Deny .
2	If you have granted access and want to revoke the access, tap Deny .

2.6.7 Logging on and off

Logout procedure

Use this procedure to log off the system.

	Action
1	On the ABB menu, tap Log Off .
2	Tap Yesto confirm.

Login procedure

Use this procedure to log on to the controller, using the User Authorization System, UAS. UAS can limit the available functions for users.

After a log off, the Login window is displayed automatically.

en040000947

	Action	Information
1	Tap on the User menu and select a user. If there are more than seven users then the menu is replaced with a button.	If you select Default User , then no password is required, and you are logged on automatically.
2	If the user you have chosen has a password you must use the soft keyboard to enter password. Tap ABC... to display the soft keyboard. Enter the password and tap OK .	
3	Tap Login .	

Continues on next page

2 Navigating and handling FlexPendant

2.6.7 Logging on and off

Continued

Handling users and authorization levels

Read more on how to add users or set the authorization in *Operating manual - RobotStudio*.

How to edit what views or functions are hidden for certain users is described in [Defining a view to be shown during operating mode change or startup on page 80](#).

2.7 Changing FlexPendant settings

2.7.1 System settings

2.7.1.1 Setting default paths

Introduction to default paths

You can set individual default paths for some actions using the FlexPendant.

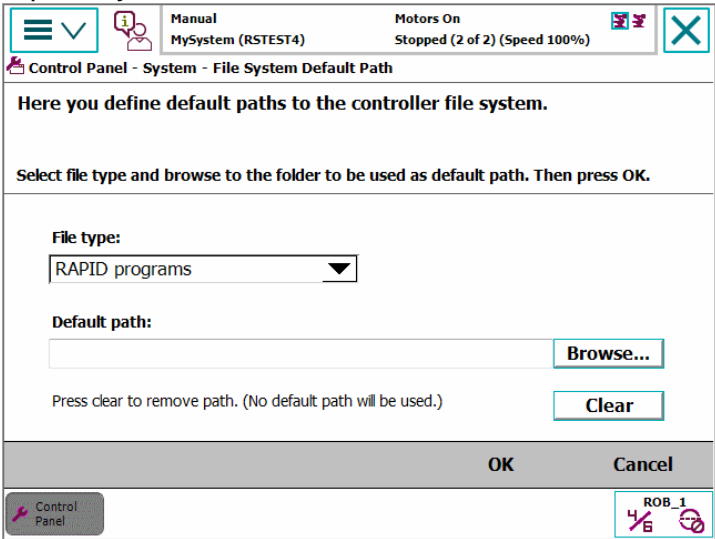
The following default paths can be set:

- Saving and loading RAPID programs.
- Saving and loading RAPID modules.
- Saving and storing configuration files.

This function is available if the user that is logged on is authorized. User authorization is handled via RobotStudio. See *Operating manual - RobotStudio*.

Setting default paths

Use this procedure to set a default path.

	Action
1	On the ABB menu, tap Control Panel and then FlexPendant .
2	<p>Tap File System Default Path.</p>  <p>en0500002361</p>
3	Tap the File type menu to choose type of default path: <ul style="list-style-type: none"> • RAPID programs • RAPID modules • Configurations files
4	Type the default path or tap Browse , to choose the desired location.
5	If required, any previously entered path can be removed by tapping Clear .
6	Tap OK .

2 Navigating and handling FlexPendant

2.7.1.2 Defining a view to be shown during operating mode change or startup

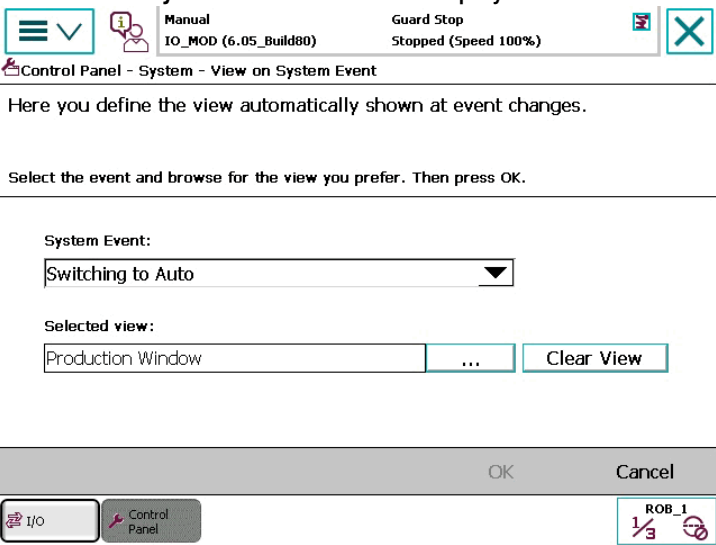
2.7.1.2 Defining a view to be shown during operating mode change or startup

View on operating mode change or startup

This function can be used, for example, if a view other than the **Production Window** is desired when switching to Auto mode.

Defining view on operating mode change

Use the following procedure to configure the FlexPendant to automatically show a specified view during a change in operating mode or during startup.

	Action
1	On the ABB menu, tap Control Panel and then tap FlexPendant .
2	<p>Tap View on System Event. The View on System Event window is displayed.</p>  <p>Control Panel - System - View on System Event</p> <p>Here you define the view automatically shown at event changes.</p> <p>Select the event and browse for the view you prefer. Then press OK.</p> <p>System Event: Switching to Auto</p> <p>Selected view: Production Window ... Clear View</p> <p>OK Cancel</p>
3	<p>Tap the System Event list and select a type of system event.</p> <p>Following are the available types:</p> <ul style="list-style-type: none">• Switching to Auto: Select this option to define the FlexPendant view when switching to auto mode.• Switching to Manual: Select this option to define the FlexPendant view when switching to manual mode.• Switching to Manual Full Speed: Select this option to define the FlexPendant view when switching to manual full speed mode.• FlexPendant Startup: Select this option to define the view when the FlexPendant starts.
4	In the Selected view field, tap ... and select a desired application to be displayed during the selected system event.
5	Tap OK . The changes are saved.



Note

The **Clear View** button will remove the currently selected view if you do not want any view to be automatically shown.

2.7.1.3 Changing the background image

Background images

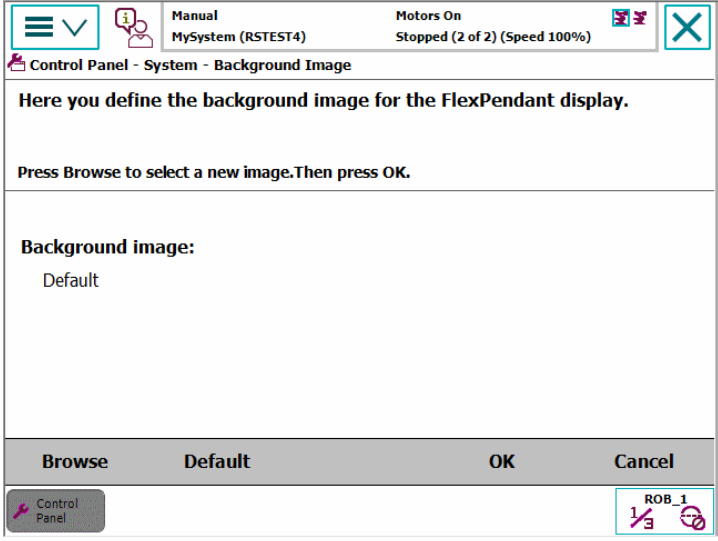
The background image on the FlexPendant can be changed. Any image file on the controller hard disk can be used, a photo as well as an illustration.

For best result, use an image following these recommendations:

- 640 by 390 pixels (width, height)
- Format gif

Changing background image

Use this procedure to change background image on FlexPendant.

Action	
1	On the ABB menu, tap Control panel .
2	<p>Tap FlexPendant and then Background Image.</p> 
3	Tap Browse to locate another picture on the controller hard disk.
4	Tap Default to restore the original background image.
5	Tap OK .

2 Navigating and handling FlexPendant

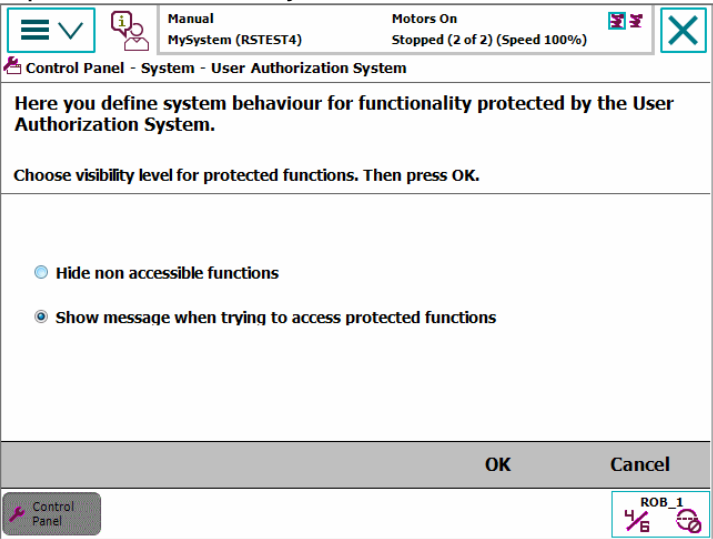
2.7.1.4 Defining visibility level for UAS protected functions

2.7.1.4 Defining visibility level for UAS protected functions

Introduction to visibility levels

This section describes how to define visibility level for functions protected by the user authorization system, UAS. The protected functions can be hidden or displayed but not accessible. All other administration of the user authorization system is done using RobotStudio. See *Operating manual - RobotStudio*.

Defining visibility level for UAS protected functions

	Action
1	On the ABB menu, tap Control Panel and then tap FlexPendant .
2	Tap User Authorization System. 
3	Tap to select the level of visibility for UAS protected functions: <ul style="list-style-type: none">• Hide non accessible functions OR• Show message when trying to access protected functions.
4	Tap OK .

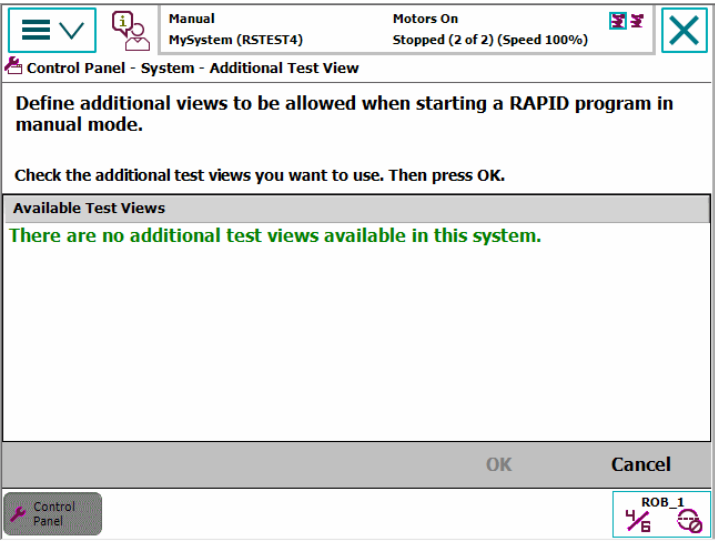
2.7.1.5 Defining an additional test view

Overview

If your system has a customized operator interface, that is one or several applications developed with FlexPendant SDK, it is possible to enable the user to start program execution in manual mode from such an application. If there is no such application, however, the screen for adding other test views will look as in the illustration below.

Defining an additional test view

Use this procedure to define an additional test view.

	Action
1	On the ABB menu, tap Control Panel and then FlexPendant .
2	<p>Tap Additional Test View. The displayed screen might look like this:</p>  <p>en0600003110</p>
3	Usually only the Program Editor and the Production Window are allowed test views. In case there are additional views to choose from, these will appear in the list. Check one or several applications to be used as additional test views.
4	Tap OK .

2 Navigating and handling FlexPendant

2.7.1.6 Defining position programming rule

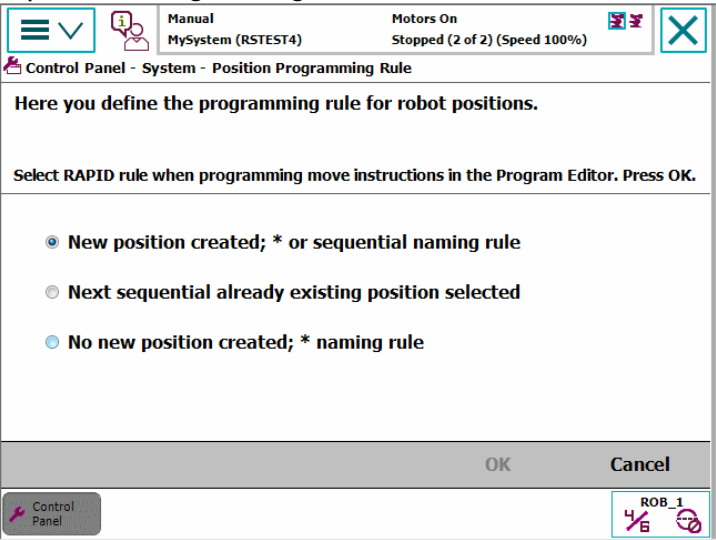
2.7.1.6 Defining position programming rule

Introduction to position naming

Robot positions in a RAPID program are either named variables or not named (using the asterisk character, *). The programmer can choose which naming rule the FlexPendant should use when new move instructions are programmed.

Defining position programming rule

Use this procedure to define a naming rule for new robot positions.

	Action
1	On the ABB menu, tap Control Panel and then FlexPendant .
2	Tap Position Programming Rule. 
3	Tap to select the preferred position programming rule.
4	Tap OK.

Position programming rules

This section gives a detailed description of the options available when programming robot positions, here referred to as *targets*. This signifies the position to which the mechanical unit is programmed to move.

New targets can be named according to any of these principles:

- **New position created; * or sequential naming rule.**
- **Next sequential already existing position selected.**
- **No new position created; * naming rule.**

New position created; * or sequential naming rule

This is the default setting. When a move instruction is programmed, a new target will automatically be created. If the last target was named, that is *not* using an “*”, the new target will be named in sequence with the previous one.

Continues on next page

For example: `MoveJ p10` will be followed by `MoveJ p20`, unless this target already exists in the program. In such a case, `MoveJ p30` (or the next free number) will be used instead.

Next sequential already existing position selected

When a move instruction is programmed, no new target will be created. Instead, the next target in a sequence created beforehand will be selected. The very first target, however, will be an "*", as no sequence yet exists. As soon as the first target has been defined this rule will be applied.

For example: A number of targets have been predefined; p10 to p50. In such a case, `MoveJ p10` will be followed by `MoveJ p20`. The next instruction will use target p30, etc. until p50 is reached. Since no further targets have been defined, p50 will be used for the following targets as well.

No new position created; * naming rule

When a Move instruction is programmed, no new target will be created. Instead, an "*" will always be used. This can be replaced by an existing target at a later stage.

For example: `MoveJ p10` will be followed by `MoveJ *`.

2 Navigating and handling FlexPendant

2.7.1.7 Defining which tasks should be selectable in the tasks panel

2.7.1.7 Defining which tasks should be selectable in the tasks panel

Tasks panel

The tasks panel is found in the Quickset menu. See [Quickset menu, Tasks on page 67](#).



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.

Defining which tasks to show

Use this procedure to define which tasks should be selectable in the tasks panel in the Quickset menu.

	Action
1	On the ABB menu, tap Control Panel and then tap FlexPendant .
2	Tap Task Panel Settings .
3	Select Only Normal tasks or All tasks . When All tasks is selected, then all tasks that are selected in the task panel (including background tasks) will stop when pressing the stop button. The selected background tasks will be treated as if the value of the system parameter <i>Trustlevel</i> is defined as <i>NoSafety</i> .
4	Tap OK .

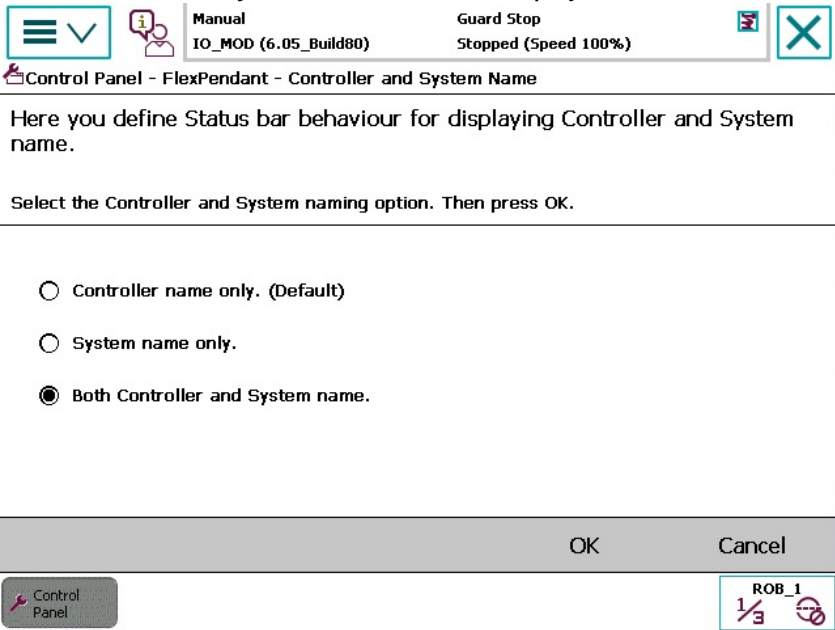
2.7.1.8 Managing the display of controller and system name

Overview

You can manage the display of controller and system name using this feature.

Controller and System name

Use the following procedure to manage the display of controller and system name in the status bar.

	Action
1	On the ABB menu, tap Control Panel and then tap FlexPendant . The FlexPendant Configuration Properties window is displayed.
2	<p>Navigate and tap Controller and System Name. The Controller and System Name window is displayed.</p>  <p>Here you define Status bar behaviour for displaying Controller and System name.</p> <p>Select the Controller and System naming option. Then press OK.</p> <p> <input type="radio"/> Controller name only. (Default) <input type="radio"/> System name only. <input checked="" type="radio"/> Both Controller and System name. </p> <p>OK Cancel</p> <p>Control Panel</p> <p>ROB_1</p> <p>xx1700000330</p>
3	<p>Select a controller and system naming option according to your requirement. Following are the available options and its description:</p> <ul style="list-style-type: none"> • Controller name only. (Default): Displays only the name of the controller in the status bar. This is the default option. • System name only: Displays only the name of the system in the status bar. • Both Controller and System name: Displays both the controller and system name in the status bar.
4	Tap OK . The changes are applied and displayed in the status bar.
5	Tap Close .

2 Navigating and handling FlexPendant

2.7.2.1 Changing brightness and contrast

2.7.2 Basic settings

2.7.2.1 Changing brightness and contrast

Appearance options

This section describes the **Appearance** menu, where you can adjust the screen's brightness and contrast. The contrast can only be adjusted on FlexPendant without USB port.

Changing brightness and contrast

Use this procedure to change brightness and contrast of the screen.

	Action
1	On the ABB menu, tap Control Panel .
2	Tap Appearance .
3	Tap the Plus or Minus button to adjust the levels. Tap Set Default to return to default. The brightness and contrast changes as you change the levels which gives you an instant view of how the new levels will affect the visibility.
4	Tap OK to use the new brightness and contrast levels.



Note

If you change brightness or contrast from the default levels, some screens can appear to be striped. This is however not a sign of a faulty screen. Change back to default settings to avoid the striped appearance.

2.7.2.2 Adapting the FlexPendant for left-handed users

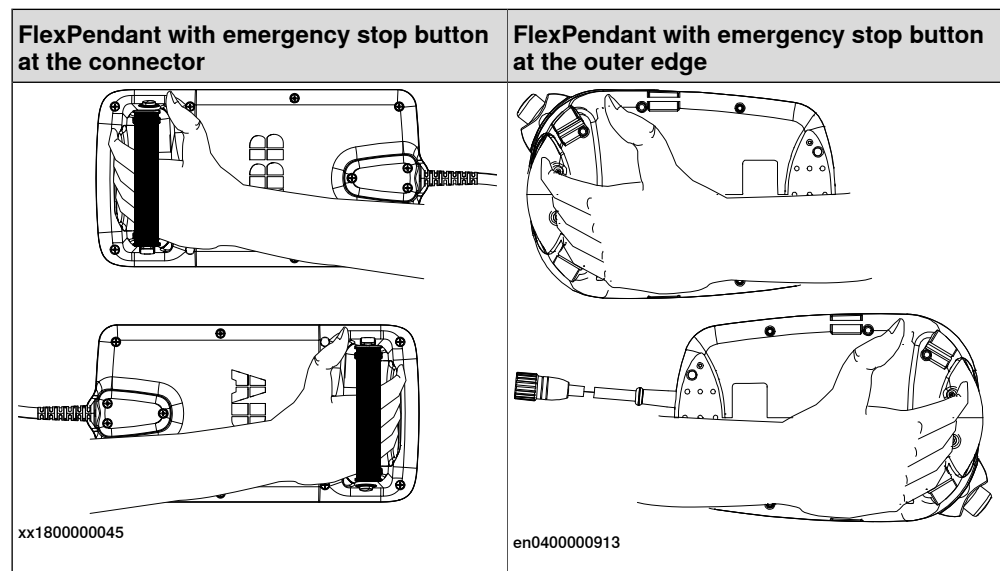
Overview

Left-handed users, normally prefers their left-hand for the touch screen. The FlexPendant can easily be adapted to suit the needs of the left-handed users. They can easily rotate the screen through 180 degrees and use their right-hand to support the device.

Illustration

The FlexPendant operated by a right-handed person at the top and by a left-handed person at the bottom.

On FlexPendant with the emergency stop button located on the outer edge of the unit, note the location of the emergency stop button when the display is rotated for left-hand users.



Rotating the FlexPendant screen

Use this procedure to adapt the FlexPendant to suit a left-handed user.

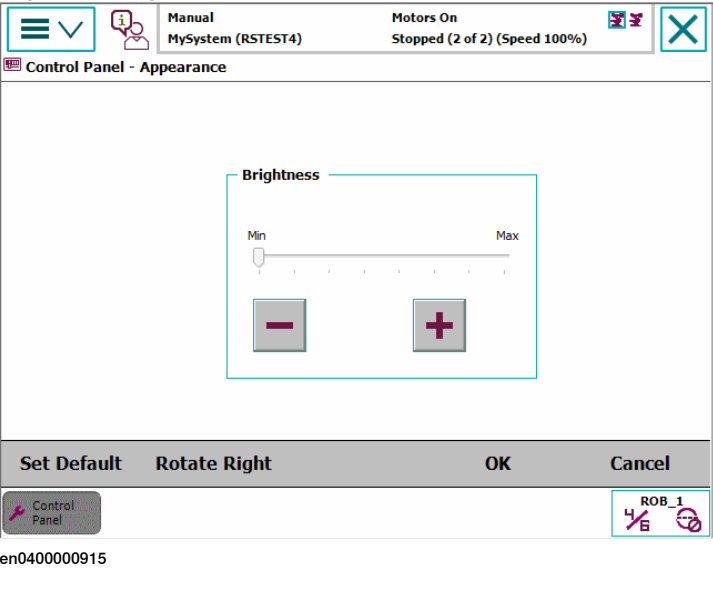
	Action
1	Tap the ABB menu, then tap Control Panel .
2	Tap Appearance .

Continues on next page

2 Navigating and handling FlexPendant


2.7.2.2 Adapting the FlexPendant for left-handed users

Continued

Action	
3	<p>Tap Rotate right.</p>  <p>The screenshot shows the FlexPendant control panel interface. At the top, there is a status bar with 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this is a 'Control Panel - Appearance' window. Inside this window, a 'Brightness' slider is visible, ranging from 'Min' to 'Max', with a slider knob and two buttons: a minus sign (-) and a plus sign (+). At the bottom of the control panel, there are four buttons: 'Set Default', 'Rotate Right', 'OK', and 'Cancel'. A 'Control Panel' button is also visible in the bottom left corner, and a 'ROB_1' button is in the bottom right corner. The ID 'en0400000915' is displayed at the bottom left of the control panel.</p>
4	Rotate the FlexPendant and move it to the other hand.

What is affected?

The following settings are affected when adapting the FlexPendant for a left-handed user.

Setting	Effect	Information
Jogging directions	The joystick directions are adjusted automatically.	The illustrations of jogging directions in the jogging menu are adjusted automatically.
Hardware buttons and programmable keys	No change.  Note Forward and Backward buttons do not change place with each other.	See Hard buttons on page 19 .
Emergency stop	No change.	
Three-position enabling device	No change.	

2.7.2.3 Controller settings

Date and time settings

Use the following procedure to configure date and time.

Step	Action
1	On the ABB menu, tap Control Panel .
2	Tap Controller Settings . The Date and Time settings window is displayed.
3	In the Settings section, select Network Time or Manual Time . Select Network Time for configuring the robot controller for automatic time synchronization using NTP protocol of a time server. The time server is identified by its IP address or DNS name. Select Manual Time if you do not have a time server that is reachable from the controller.
4	Select the required time zone from the Time Zone section.
5	In the Date and Time section, tap the + (Plus) or - (Minus) button and configure the date and time.
6	Tap OK . The selected settings are saved.



Note

The date and time are displayed according to the ISO standard. That is, date is in the year-month-day format and time is in the hour:minute format (24-hour format).

Network settings

Use the following procedure to configure the network.

Step	Action
1	On the ABB menu, tap Control Panel .
2	Tap Controller Settings .
3	Navigate to the bottom menu, tap Settings , and select Network . The Control Panel - Controller Settings - Network window is displayed.
4	Configure the network settings according to your requirements.
5	Tap OK . The selected settings are saved.

Identity settings

Use the following procedure to configure the identity of the controller.

Step	Action
1	On the ABB menu, tap Control Panel .
2	Tap Controller Settings .
3	Navigate to the bottom menu, tap Settings , and select Identity . The Control Panel - Controller Settings - Identity window is displayed.
4	Edit the name of the controller in the Controller Name field according to your requirement.

Continues on next page

2 Navigating and handling FlexPendant

2.7.2.3 Controller settings

Continued

Step	Action
5	Tap OK. The name of the controller is saved.

2.7.2.4 Configuring Most Common I/O

Most Common I/O

Most Common I/O is used in the **Program Editor** to display a list of the most commonly used I/O signals in the robot system. Since there can be many signals, it can be helpful to use this selection.

The sorting in the list can be rearranged manually. By default, the signals are sorted in the order that they are created.

Most Common I/O can also be configured using system parameters in the topic *Man-machine Communication*. However, sorting the list can only be done by using the function under the Control Panel.

Configuring Most Common I/O

Use this procedure to configure the list Most Common I/O.

	Action
1	On the ABB menu, tap Control Panel .
2	Tap I/O . A list of all I/O signals defined in the system is listed with check boxes.
3	Tap the names of the signals to select for the Most Common I/O list. Tap All or None to select all or no signals. Tap Name or Type to sort by name or signal type.
4	Tap Preview to see the list of selected signals and adjust the sort order. Tap to select a signal and then tap the arrows to move the signal up or down in the list, rearranging the sort order. Tap APPLY to save the sort order. Tap Edit to return to the list of all signals.
5	Tap APPLY to save the settings.

2 Navigating and handling FlexPendant

2.7.2.5 Changing language

2.7.2.5 Changing language

Languages

The FlexPendant is installed with and supports twenty different languages. By default, the current language is English.

Switching from one installed languages to another is easy. For more information, see [Changing language on page 94](#).



Note

When you switch to another language, all buttons, menus and dialogs will use the new language. RAPID instructions, variables, system parameters, and I/O signals are not affected.

Changing language

Use this procedure to change language on the FlexPendant.

	Action
1	On the ABB menu, tap Control Panel .
2	Tap Language . A list of all installed languages is displayed.
3	Tap the language that you want to change to.
4	Tap OK . A dialog box is displayed. Tap Yes to proceed and restart the FlexPendant. The current language is replaced by the selected one.

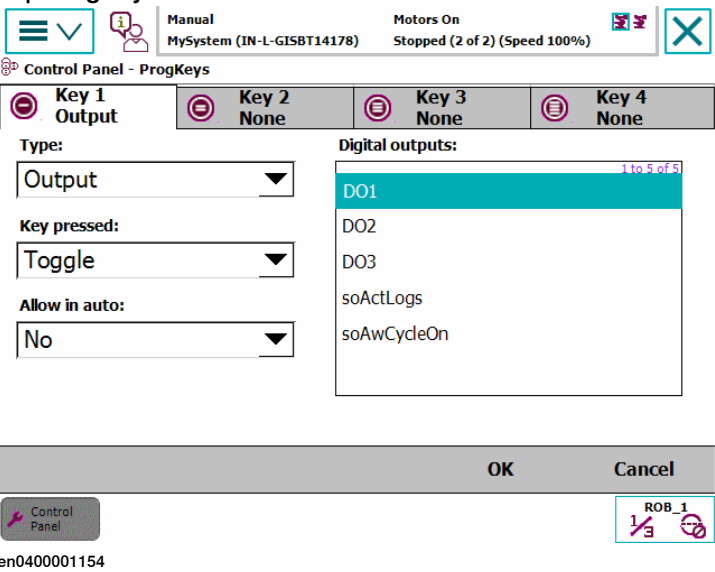

2.7.2.6 Changing programmable keys

Overview

Programmable keys are four hardware buttons on the FlexPendant that can be used for dedicated, specific functions set by the user. See [Hard buttons on page 19](#). The keys can be programmed to simplify programming or testing of programs. They can also be used to activate menus on the FlexPendant.

Change programmable keys

Use the following procedure to set the programmable keys:

	Action
1	On the ABB menu, tap Control Panel .
2	<p>Tap ProgKeys.</p> 
3	Select key to set, Key 1-4 in the upper selection list.
4	<p>Tap the Type menu to select type of action:</p> <ul style="list-style-type: none"> • None • Input • Output • System
5	<p>If Type Input is selected.</p> <ul style="list-style-type: none"> • Tap to select one of the digital inputs from the list. • Tap the Allow in auto menu to select if the function is also allowed in automatic operating mode. <p> Note</p> <p>An input cannot be set by using the programmable keys, its value can only be pulsed. The pulse will be the inverted value of the input.</p>

Continues on next page

2 Navigating and handling FlexPendant

2.7.2.6 Changing programmable keys

Continued

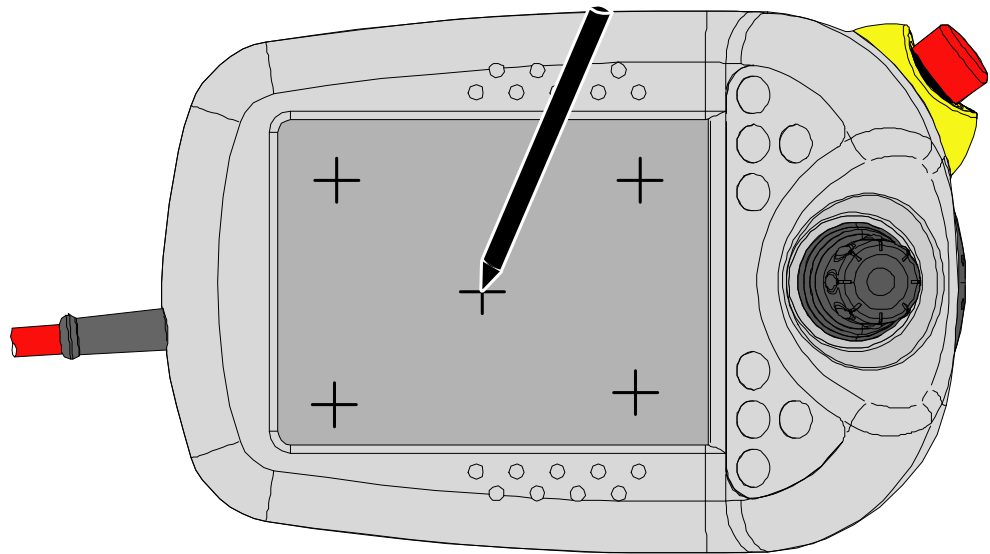
	Action
6	<p>If Type Output is selected.</p> <ul style="list-style-type: none">• Tap to select one of the digital outputs from the list• Tap the Key pressed menu to define how the signal should behave when the key is pressed.• Tap the Allow in auto menu to select if the function is also allowed in automatic operating mode <p>Key pressed functions:</p> <ul style="list-style-type: none">• Toggle - switches signal value from 0 to 1 or vice versa• Set to 1 - sets the signal to 1• Set to 0 - sets the signal to 0• Press/Release - sets signal value to 1 while key is pressed (note that an inverted signal will be set to 0)• Pulse - the signal value pulses once
7	<p>If Type System is selected.</p> <ul style="list-style-type: none">• Tap the Key pressed menu to select Move PP to main• Tap the Allow in auto menu to select if the function is also allowed in automatic operating mode
8	Set the other keys as described in steps 3 to 7 above.
9	Tap OK to save the settings.

2.7.2.7 Calibrating the touchscreen

When to calibrate

The touchscreen is calibrated on delivery and normally never needs to be calibrated. On FlexPendant with the emergency stop button located at the connector, it is not possible to calibrate the touchscreen.


On FlexPendant with the emergency stop button located on the outer edge of the unit, the touchscreen can be calibrated. Depending on the model of the FlexPendant, the appearance of the symbols will vary, the function is however the same.



en040000974

Calibrating the touchscreen

Use this procedure to calibrate the touchscreen.

	Action	Information
1	On the ABB menu, tap Control Panel.	
2	Tap Touch Screen.	
3	Tap Recalibrate. The screen will go blank for a few seconds. A series of symbols will appear on the screen, one at a time.	
4	Tap the center of each symbol with a pointed object.	 CAUTION Do not use a sharp object which can damage the surface of the screen.
5	The recalibration is complete.	

Continues on next page

2 Navigating and handling FlexPendant

2.7.2.7 Calibrating the touchscreen

Continued

About the touch calibration function

The touch calibration function waits on each calibration point for a couple of touch coordinates or that the touch will be released. Then the average of the collected coordinates will be calculated and the symbol moves to the next position.

The touch controller only sends new coordinates to the CPU when the coordinates are changing. If you touch the symbol very accurately with a stylus, the touch coordinates will not change. Then the touch controller sends only one coordinate and the touch calibration function is waiting endlessly for more coordinates.

The best way to avoid this problem is to tap the symbol for only one second and then release.

3 Jogging

3.1 Introduction to jogging

What is jogging?

To jog is to manually position or move robots or external axes.

When can I jog?

You can jog in manual mode but you cannot jog during program execution.

Jogging the robot

This procedure details the main steps for jogging the robot.

	Action	Information
1	It is possible to jog the robot under the following conditions: <ul style="list-style-type: none"> The system has been started as detailed in this manual. No programmed operation is running The system is in Manual mode. The three-position enabling device is pressed and the system is in Motors On state. 	The Manual mode is described in section About the manual mode on page 187 . Starting in the Manual mode is detailed in section Starting programs on page 225 . How to switch to manual mode is detailed in section Switching from automatic to manual mode on page 246 .
2	Determine in which way you want to jog.	The difference between different types of jogging is detailed in section Introduction to jogging on page 99 . How to select coordinate system is detailed in section Selecting coordinate system on page 116 .
3	Select a mechanical unit. The axes can be jogged in different ways.	How to jog the robot axis by axis is detailed in section Jog axis by axis on page 115 .
4	Define the working range for the robot/robots as well as for any other pieces of equipment working in the robot cell.	The robot's working range is defined by system parameters. See <i>Technical reference manual - System parameters</i> .
5	Jog the manipulator using the joystick on the FlexPendant.	The FlexPendant and its various parts and sections are described in section The FlexPendant on page 17 . The joystick and how to map the directions of it, is detailed in section Selecting motion mode on page 112 . How to prevent causing manipulator movements in certain directions while jogging, is detailed in section Locking the joystick in specific directions on page 117 . There might be restrictions to how you can jog, see section Restrictions to jogging on page 108 .
6	In some cases, more than one manipulator can be jogged simultaneously. This requires the option <i>MultiMove</i> .	How to jog multiple manipulators is detailed in section Coordinated jogging on page 109 .

Continues on next page

3 Jogging

3.1 Introduction to jogging

Continued

About motion modes and robots

The selected motion mode and/or coordinate system determines the way the robot moves.

In linear motion mode the tool center point moves along straight lines in space, in a "move from point A to point B" fashion. The tool center point moves in the direction of the selected coordinate system's axes.

Axis-by-axis mode moves one robot axis at a time. It is then hard to predict how the tool center point moves.

About motion modes and additional axes

Additional axes can only be jogged axis-by-axis. An additional axis can either be designed for some kind of linear motion or for rotational (angular) motion. Linear motion is used in conveyers, rotational motion in many kinds of workpiece handlers.

Additional axes are not affected by the selected coordinate system.

About coordinate systems

Positioning a pin in a hole with a gripper tool can be performed very easily in the tool coordinate system, if one of the coordinates in that system is parallel to the hole. Performing the same task in the base coordinate system may require jogging in both x, y, and z coordinates, making precision much more difficult.

To select the proper coordinate systems to jog in will make jogging easier but there are no simple or single answers to which coordinate system to choose.

A certain coordinate system will make it possible to move the tool center point to the target position with fewer joystick moves than another.

Conditions such as space limitations, obstacles or the physical size of a work object or tool will also guide you to the proper judgement.

Read more about coordinate systems in section [Coordinate systems for jogging on page 101](#).

3.2 Coordinate systems for jogging

Coordinate systems

A coordinate system defines a plane or space by axes from a fixed point called the origin. Robot targets and positions are located by measurements along the axes of coordinate systems.

A robot uses several coordinate systems, each suitable for specific types of jogging or programming.

- The *base coordinate system* is located at the base of the robot. It is the easiest one for just moving the robot from one position to another.
- The *work object coordinate system* is related to the work piece and is often the best one for programming the robot.
- The *tool coordinate system* defines the position of the tool the robot uses when reaching the programmed targets.
- The *world coordinate system* that defines the robot cell, all other coordinate systems are related to the world coordinate system, either directly or indirectly. It is useful for jogging, general movements and for handling stations and cells with several robots or robots moved by external axes.
- The *user coordinate system* is useful for representing equipment that holds other coordinate systems, like work objects.

Default settings

If you change coordinate system in the jogging properties, this will automatically be reset to default settings after a restart.

Linear mode

For each mechanical unit the system will by default use the base coordinate system for the linear motion mode.

Reorient mode

For each mechanical unit the system will by default use the tool coordinate system for the reorientation motion mode.

Continues on next page

3 Jogging

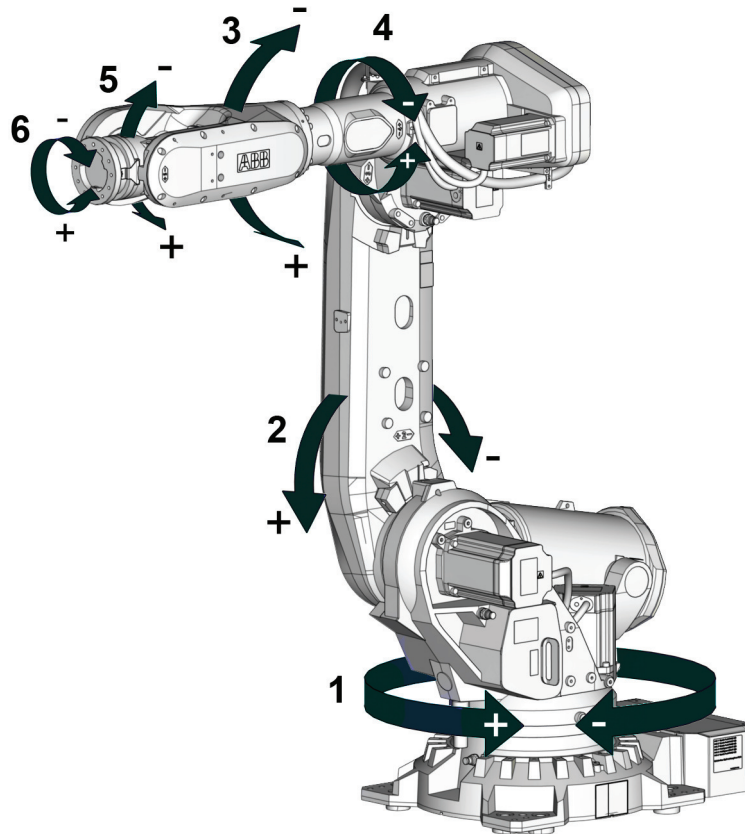
3.2 Coordinate systems for jogging

Continued

Illustration of axes and joystick directions

The axes of a generic six axis manipulator can be jogged manually using the joystick. Please check your plant or cell documentation to determine the physical orientation of any additional axes.

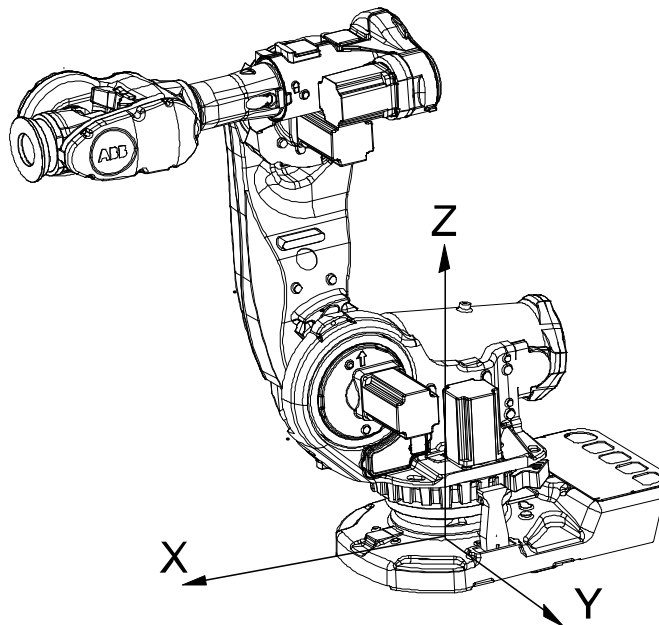
The illustration shows the movement patterns for each manipulator axis.



xx0300000520

Continues on next page

The base coordinate system



xx0300000495

The base coordinate system has its zero point in the base of the robot, which makes movements predictable for fixed mounted robots. It is therefore useful for jogging a robot from one position to another. For programming a robot, other coordinate systems, like the work object coordinate system are often better choices. See [The work object coordinate system on page 104](#) for more information.

When you are standing in front of the robot and jog in the base coordinate system, in a normally configured robot system, pulling the joystick towards you will move the robot along the X axis, while moving the joystick to the sides will move the robot along the Y axis. Twisting the joystick will move the robot along the Z axis.

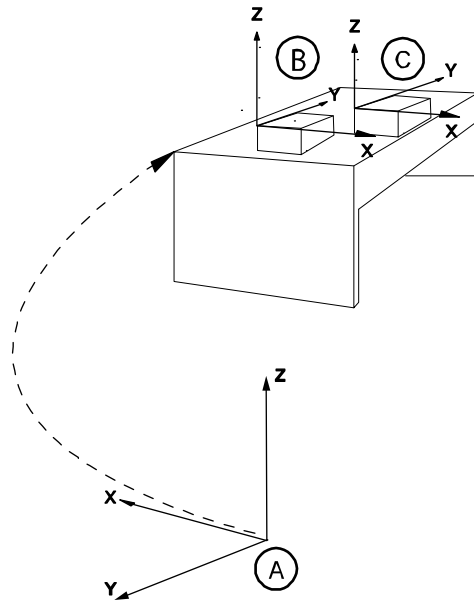
Continues on next page

3 Jogging

3.2 Coordinate systems for jogging

Continued

The work object coordinate system



xx0600002738

A	World coordinate system
B	Work Object coordinate system 1
C	Work Object coordinate system 2

The work object coordinate system corresponds to the work piece: It defines the placement of the work piece in relation to the world coordinate system (or any other coordinate system).

The work object coordinate system must be defined in two frames, the user frame (related to the world frame) and the object frame (related to the user frame).

A robot can have several work object coordinate systems, either for representing different work pieces or several copies of the same work piece at different locations.

It is in work object coordinate systems you create targets and paths when programming the robot. This gives a lot of advantages:

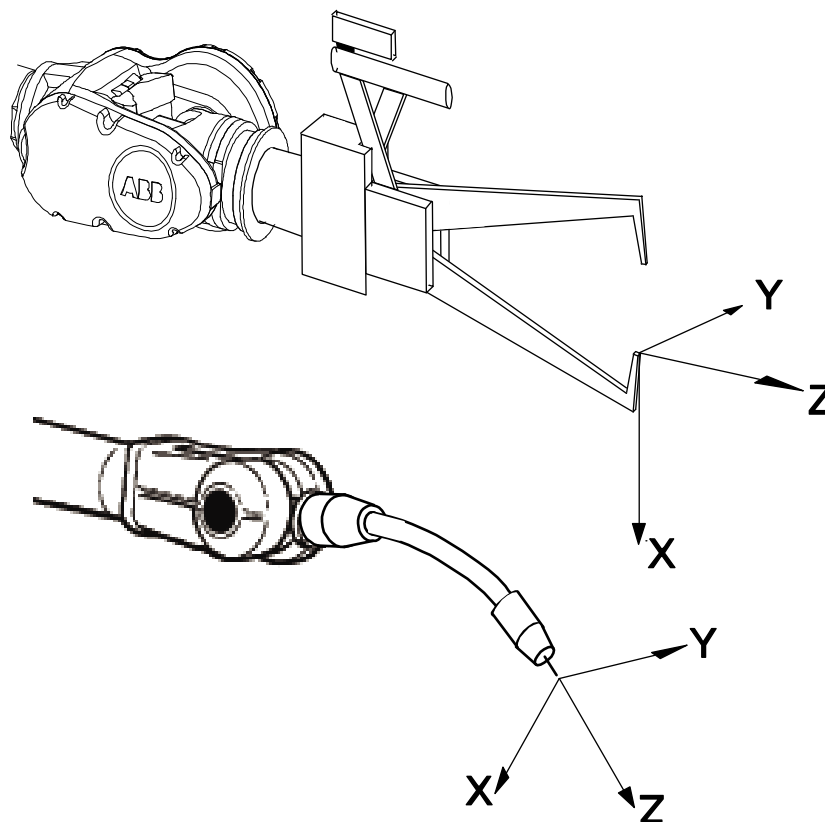
- When repositioning the work piece in the station you just change the position of the work object coordinate system and all paths are updated at once.
- Enables work on work pieces moved by external axes or conveyor tracks, since the entire work object with its paths can be moved.

Examples of use

For example, you are determining the positions of a number of holes to be drilled along the edge of the work object.

You are creating a weld between two walls in a box.

Continues on next page

The tool coordinate system

en0300000497

The tool coordinate system has its zero position at the center point of the tool. It thereby defines the position and orientation of the tool. The tool coordinate system is often abbreviated TCPF (Tool Center Point Frame) and the center of the tool coordinate system is abbreviated TCP (Tool Center Point).

It is the TCP the robot moves to the programmed positions, when executing programs. This means that if you change the tool (and the tool coordinate system) the robot's movements will be changed so that the new TCP will reach the target.

All robots have a predefined tool coordinate system, called `tool0`, located at the wrist of the robot. One or many new tool coordinate systems can then be defined as offsets from `tool0`.

When jogging a robot the tool coordinate system is useful when you don't want to change the orientation of the tool during the movement, for instance moving a saw blade without bending it.

Examples of use

Use the tool coordinate system when you need to program or adjust operations for threading, drilling, milling or sawing.

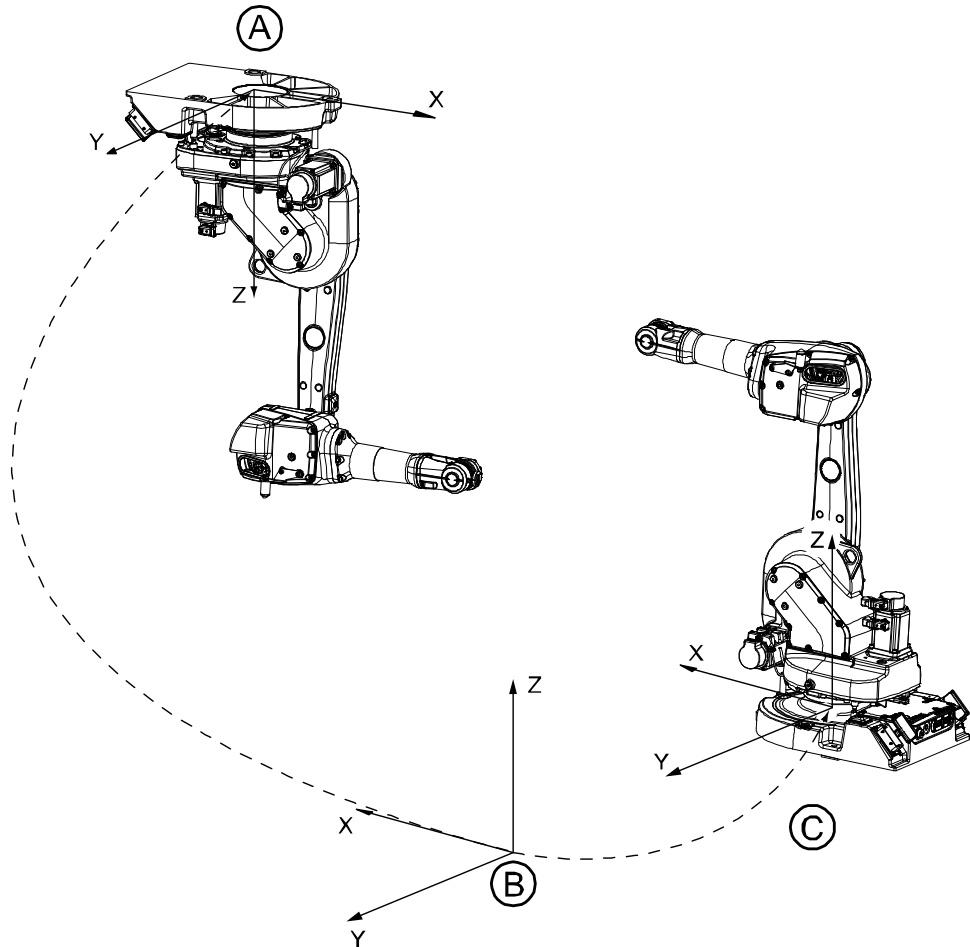
Continues on next page

3 Jogging

3.2 Coordinate systems for jogging

Continued

The world coordinate system



en030000496

A	Base coordinate system for robot 1
B	World coordinate
C	Base coordinate system for robot 2

The world coordinate system has its zero point on a fixed position in the cell or station. This makes it useful for handling several robots or robots moved by external axes.

By default the world coordinate system coincides with the base coordinate system.

Examples of use

For example, you have two robots, one floor mounted and one inverted. The base coordinate system for the inverted robot would be upside down.

If you jog in the base coordinate system for the inverted robot, movements will be very difficult to predict. Choose the shared world coordinate system instead.

3.3 Joystick directions

Introduction to joystick directions

The Joystick Directions area shows how joystick axes correspond to the selected coordinate system's axes.



CAUTION

The Directions properties is not intended to show the direction in which the mechanical unit will move. Always try jogging with small joystick movements so that you learn the true directions of the mechanical unit.

Joystick directions

The significance of the joystick directions depends on the selected motion mode.

Motion mode	Illustration joystick	Description
Linear	<p>en0400001131</p>	Linear mode is described in section Setting the tool orientation on page 114 .
Axis 1, 2, and 3 (default for robots)	<p>en0300000536</p>	Axis 1-3 mode is described in section Jog axis by axis on page 115 .
Axis 4, 5, and 6	<p>en0300000537</p>	Axis 4-6 mode is described in section Jog axis by axis on page 115 .
Reorient	<p>en0400001131</p>	Reorient mode is described in section Setting the tool orientation on page 114 .

3 Jogging

3.4 Restrictions to jogging

3.4 Restrictions to jogging

Jog additional axes

Additional axes can only be jogged axis-by-axis. Please see *Application manual - Additional axes and stand alone controller*.

Jog mechanical units that are not calibrated

If the mechanical unit is not calibrated the text **Unit not calibrated** will be displayed in the **Position** area of the **Jogging** window.

An uncalibrated mechanical unit can only be jogged axis-by-axis. Its working range will not be checked.

When the robot is not calibrated, incremental movement is restricted to one step per joystick deflection. A calibrated robot performs 10 steps/sec when deflecting the joystick.



CAUTION

Mechanical units whose working range is not controlled by the robot system can be moved to dangerous positions. Mechanical stops should be used and configured to avoid danger to equipment or personnel.

Jog robot axes in independent mode

It is not possible to jog axes in independent mode. You need to return the axes to normal mode in order to jog. Please see *Application manual - Controller software IRC5* for details.

Jog while using world zones

With the option *World Zones* installed, defined zones will restrict motion while you jog. Please see *Application manual - Controller software IRC5* for details.

Jog with axis loads not set

If equipment is mounted on any of the robot axes, then axes loads must be set. Otherwise overload errors might occur when jogging.

How to set axis loads are described in the Product Manuals delivered with your robot.

Jog with tool or payload weights not set

If the weight of tools and payloads is not set, then overload errors might occur when jogging. Loads for additional axes controlled by specific software (dynamic models) can only be set in programming.

3.5 Coordinated jogging

Coordination

A robot that is coordinated to a work object will follow the movements of that work object.

Coordinated jogging

If the mechanical unit moving the work object is jogged, any robot that is currently coordinated with the work object will move so that it maintains its relative position to the work object.

Set up coordination

	Action	Information
1	Select the robot that is to be coordinated to another mechanical unit.	See Selecting mechanical unit for jogging on page 110 .
2	Set Coordinate system to Work Object.	See Selecting coordinate system on page 116 .
3	Set Work object to the work object moved by the other mechanical unit.	See Selecting tool, work object, and payload on page 113 .
4	Select the mechanical unit that moves the work object.	Any jogging, while this mechanical unit is selected, will also affect the robot that is coordinated with it.

Coordinating robots

Coordinating robots, so that when jogging one robot another robot will follow, requires the option *MultiMove*. See *Application manual - MultiMove*.

3 Jogging

3.6.1 Selecting mechanical unit for jogging

3.6 Basic settings for jogging

3.6.1 Selecting mechanical unit for jogging

Jogging properties

If your system has more than one robot, that is additional robots or additional axes, then you need to select which mechanical unit to jog when using the joystick.

There are three ways to select mechanical unit:

- Using the **Select mechanical unit** button.
- Using the **Jogging** window on the **ABB** menu.
- Using the **Quickset** menu **Mechanical unit**, see [Quickset menu, Mechanical unit on page 58](#).

Any changes you make to jogging properties only affects the currently selected mechanical unit.

All jogging properties are saved and restored when you return to jog that mechanical unit.

Identifying mechanical units

Each mechanical unit that can be jogged is represented in the mechanical units list. The name of the unit is defined in the system configuration. Each unit also has a symbol that is used in the Status bar, see section [Status bar on page 56](#).

In manual mode, the Quickset menu button displays which mechanical unit is selected.

Please consult your plant or cell documentation to see which mechanical units are available in your robot system.

Selecting mechanical unit using the hard button

Press the **Select mechanical unit** button to change unit. One press on the button changes to the next mechanical unit, as steps in a cycle.

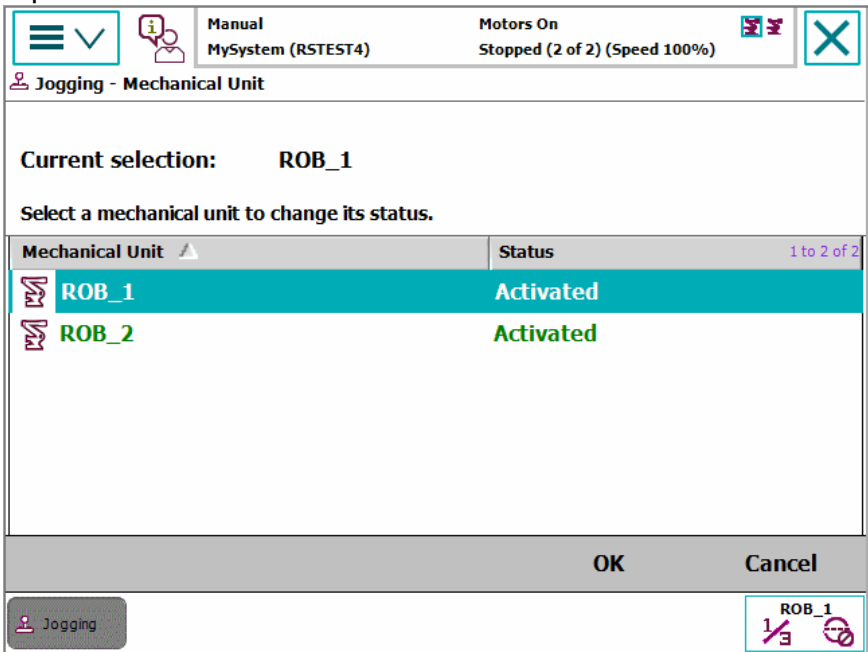
Selecting mechanical unit in the Jogging window

Use this procedure to select a mechanical unit to jog in the **Jogging** window.

	Action
1	On the ABB menu, tap Jogging .

Continues on next page

3.6.1 Selecting mechanical unit for jogging
Continued

Action	
2	<p>Tap Mechanical Unit.</p>  <p>The screenshot shows a dialog box titled 'Tap Mechanical Unit'. At the top, there are status indicators: 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this, the text 'Jogging - Mechanical Unit' is displayed. The current selection is 'ROB_1'. A table lists two mechanical units: 'ROB_1' and 'ROB_2', both with a status of 'Activated'. At the bottom of the dialog are 'OK' and 'Cancel' buttons. A 'Jogging' button is visible at the bottom left, and a 'ROB_1' button with a jog icon is at the bottom right.</p>
3	<p>Tap the mechanical unit to be jogged, and then tap OK. The selected mechanical unit is active until you select another unit, even if you close the Jogging window.</p>

Examples of use

Your robot system may consist of more than a single robot. There can also be other mechanical units such as workpiece handlers or additional axes mounted on the robot that can also be jogged.

Related information

If the system uses *Multitasking*, and has more than one motion task, and uses more than one mechanical unit, then the selected mechanical unit can be switched automatically when switching between **Program Editor** windows. See section [Program Editor on page 43](#).

Mechanical units can be activated or deactivated with the **Activate** function in the **Jogging** window.

3 Jogging

3.6.2 Selecting motion mode

3.6.2 Selecting motion mode

Motion mode

There are three ways to select motion mode:

- 1 Using the **Toggle motion mode** button.
- 2 Using the **Jogging** window on the **ABB** menu.
- 3 Using the **Quickset** menu **Mechanical** unit, see [Quickset menu, Mechanical unit on page 58](#).

Selecting motion mode using the toggle button

Press the **Toggle motion mode reorient/linear** button to switch motion mode.

Selecting motion mode in the Jogging window

Use this procedure to select motion mode in the **Jogging** window.

	Action	Information
1	On the ABB menu, tap Jogging .	
2	Tap Motion mode .	
3	Tap on the mode you want and then tap OK .	The significance of the joystick directions are shown in Joystick direction after making the selection.

Related information

[Joystick directions on page 107](#).

3.6.3 Selecting tool, work object, and payload

Overview

It is always important to choose the proper tool, work object, or payload. It is absolutely vital when you create a program by jogging to the target positions.

Failing to do so will most likely result in overload errors and/or incorrect positioning either when you jog or when you run the program in production.

Selecting tool, work object, and payload

	Action
1	On the ABB menu, choose Jogging to view jogging properties.
2	Tap Tool , Work object , or Payload to display the lists of available tools, work objects or payloads.
3	Tap the tool, work object, or payload of choice followed by OK .

3 Jogging

3.6.4 Setting the tool orientation

3.6.4 Setting the tool orientation

Examples of use

Tools for arc welding, grinding and dispensing must be oriented in a particular angle to the work piece to obtain the best result. You also need to set up the angle for drilling, milling or sawing.

In most cases you set the tool orientation when you have jogged the tool center point to a specific position such as the starting point for a tool operation. After you have set the tool orientation you continue to jog in linear motion to complete the path and the supposed operation.

Definition of tool orientation

The tool orientation is relative to the currently selected coordinate system. From a user perspective however this is not noticeable.

Setting the tool orientation

	Action
1	On the ABB menu, tap Jogging .
2	Tap Motion Mode , then tap Reorient followed by OK .
3	If not already selected, select the proper tool by following the procedure in Selecting tool, work object, and payload on page 113 .
4	Press and hold the three-position enabling device to activate the mechanical unit's motors. Move the joystick and the tool's orientation changes.



Tip

Use the **QuickSet** menu to select jogging mode faster.

3.6.5 Jog axis by axis

Jogging axis by axis

There are three ways to select axis for jogging.

- Using the **Toggle motion mode axis group** button.
- Using the **Jogging** window on the **ABB** menu.
- Using the **Quickset** menu **Mechanical unit**, see [Quickset menu, Mechanical unit on page 58](#).

In manual mode, the Quickset menu button displays which axis group is selected.

How to use the joystick when jogging axis by axis is displayed in the **Joystick directions** area. See [Illustration of axes and joystick directions on page 102](#).

Examples of use

Use axis by axis jogging when you need to:

- Move the mechanical unit out of a hazardous position.
- Move robot axes out of singularities.
- Position axes for fine calibration.

Selecting axis group using the toggle button

Press the **Toggle motion mode axis group** button to switch motion mode.

Selecting axis group in the Jogging window

Use this procedure to select axis group in the **Jogging** window.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Motion Mode .
3	Tap on the axis group 1-3 or 4-6 and then tap OK .



CAUTION

The orientation of any mounted tool will be affected by this procedure. If the resulting orientation is important, perform the procedure described in [Setting the tool orientation on page 114](#) when finished.

3 Jogging

3.6.6 Selecting coordinate system

3.6.6 Selecting coordinate system

Coordinate systems for jogging

The coordinate system most suitable for your jogging depends on many things. See section [Coordinate systems for jogging on page 101](#), for more information.

There are two ways to select coordinate system:

- Using the **Jogging** window on the **ABB** menu.
- Using the **Quickset** menu **Mechanical unit**, see [Quickset menu, Mechanical unit on page 58](#).

Prerequisites

Select motion mode suitable for the intended jogging.

Stationary tools in the tool coordinate system

If your robot system uses stationary tools, you must select both the proper tool and the proper work object (held by the robot) to jog in tool coordinates.

The tool coordinate system is defined by the position and orientation of the stationary tool and is fixed in space. To perform the intended operations you move the work object. This way positions can be expressed in the tool coordinate system.

Selecting coordinate system

Use this procedure to select coordinate system in the **Jogging** window.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Coordinate System .
3	Tap to select a coordinate system.
4	Tap OK .

3.6.7 Locking the joystick in specific directions

Overview

The joystick can be locked in specific directions to prevent movement for one or more axes.

This may be useful for instance while fine tuning positions or when programming operations that should only be performed in the direction of a specific coordinate system axis.

Note that the axes locked depends on the currently selected motion mode.

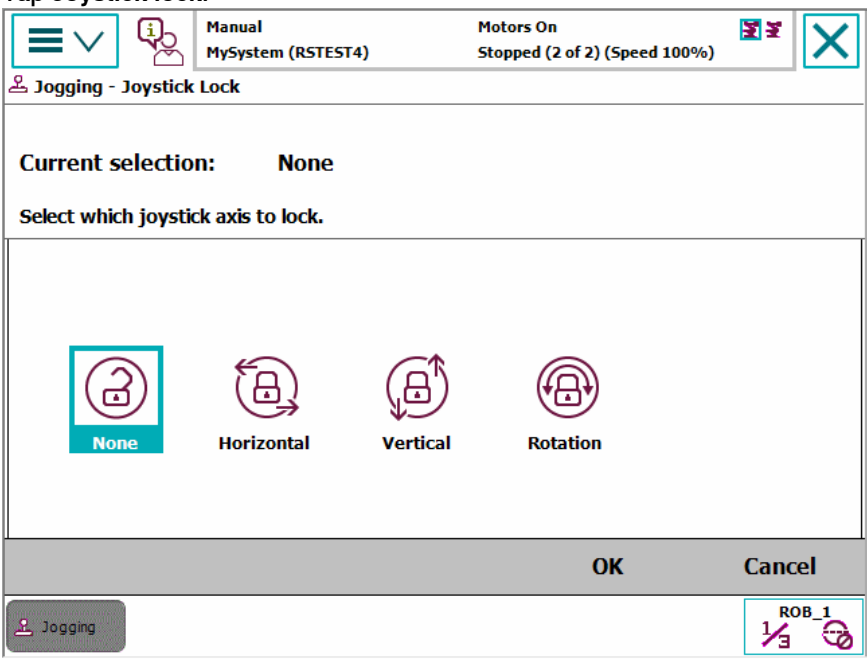
Which axes are locked?

This section describes how to see which joystick directions are locked

	Action
1	On the ABB menu, tap Jogging to view jogging properties.
2	Tap Joystick lock to check the joystick properties, or check the Joystick directions area properties in the right hand corner of the window. A padlock symbol is displayed for locked axes.

Locking the joystick in specific directions

This section describes how to lock the joystick in specific directions.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Joystick lock . 
3	Tap the joystick axis or axes that should be locked. The axis toggles between locked and unlocked each time you tap.
4	Tap OK to lock.

Continues on next page

3 Jogging

3.6.7 Locking the joystick in specific directions

Continued

Unlocking all axes

This section describes how to unlock all axes from the joystick directions lock.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Joystick lock .
3	Tap None , then tap OK .

3.6.8 Incremental movement for precise positioning

Incremental movement

Use incremental movement to jog the robot in small steps, which enables very precise positioning.

This means that each time the joystick is deflected, the robot moves one step (increment). If the joystick is deflected for one or more seconds, a sequence of steps, (at a rate of 10 steps per second), will be performed as long as the joystick is deflected.

Default mode is no increment, then the robot moves continuously when the joystick is deflected.

There are three ways to select the increment size:

- Using the **Toggle increments** button.
- Using the **Jogging** window on the **ABB** menu.
- Using the **Quickset** menu **Increments**, see [Quickset menu, Increment on page 63](#).

To use the toggle button you must first select an increment size in the **Jogging** window or **Quickset** menu.

Selecting increments using the toggle button

Press the **Toggle increments** button to switch increment size, you toggle between no increments and the increment size you previously selected in the **Jogging** window.

Selecting increments in the Jogging window

Use this procedure to select the incremental movement size using the **Jogging** window.

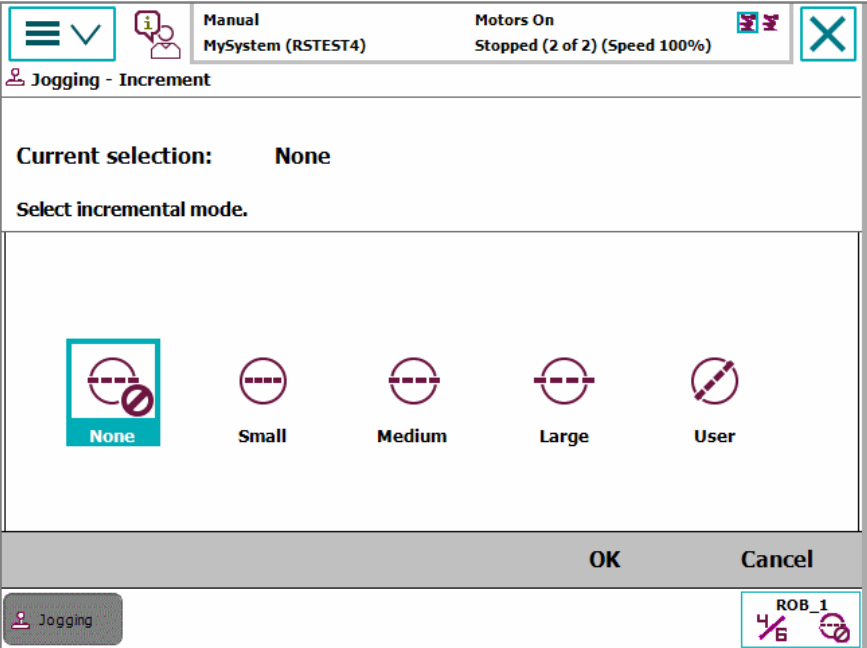
	Action
1	On the ABB menu, tap Jogging .

Continues on next page

3 Jogging

3.6.8 Incremental movement for precise positioning

Continued

Action	
2	<p>Tap Increment.</p>  <p>en0400000971</p>
3	<p>Tap the desired increment mode, see description in section Incremental movement sizes on page 120.</p>
4	<p>Tap OK.</p>

Incremental movement sizes

Choose between small, medium or large increments. You can also define your own increment movement sizes.

Increment	Distance	Angular
Small	0.05 mm	0.005°
Medium	1 mm	0.02°
Large	5 mm	0.2°
User		

3.6.9 Reading the exact position

About positions and revolution counters

The exact position of the robot is determined using the position of the resolvers and counters that count the number of resolver revolutions. These are called revolution counters.

If the robot is correctly calibrated then the current position is automatically calculated at start.



CAUTION

If the positions are displayed in red text then the values from the revolution counters are lost and instead the values stored on the SMB are displayed. Be careful when jogging the robot if the values are displayed in red text. Watch the robot closely and do not use the displayed values! If the mechanical unit is uncalibrated then the actual position can be very different from the position values stored by the SMB. You must update the revolution counters before a program can be started. See [Updating revolution counters on page 282](#).



Note

If no positions are displayed then the mechanical unit is uncalibrated. Instead the text **Selected mechanical unit is not calibrated** is displayed.



Note

When updating the revolution counters, the ongoing RAPID instruction or function is interrupted, and the path is cleared.

How robot positions are displayed

Positions are always displayed as:

- The point in space expressed in the x, y, and z tool center point coordinates.
- The angular rotation of the tool center point expressed in Euler angles or as a quaternion.

How additional axes' positions are displayed

When an additional axis is moved, only the axis position is displayed.

Linear axis positions are displayed in millimeters expressed as the distance to the calibration position.

Rotating axis positions are displayed in degrees expressed as the angle to the calibration position.

Reading the exact position

This procedure describes how to read the exact position.

	Action
1	On the ABB menu tap .Jogging .

Continues on next page

3 Jogging

3.6.9 Reading the exact position

Continued

	Action
2	The position is displayed in the Position area properties in the right hand side of the window. See illustration in Jogging on page 38 .

Position format

The position can be displayed in different formats. Tap **Position Format** to change settings.

The **Position** can be displayed relative the following frames:

- World
- Base
- Work object

The **Orientation** format can be set to:

- Quaternion
- Euler angles

The **Position angle** format can be set to:

- Angles

The **Presentation angle unit** can be set to:

- Degrees
- Radians

3.6.10 Aligning tools

Overview

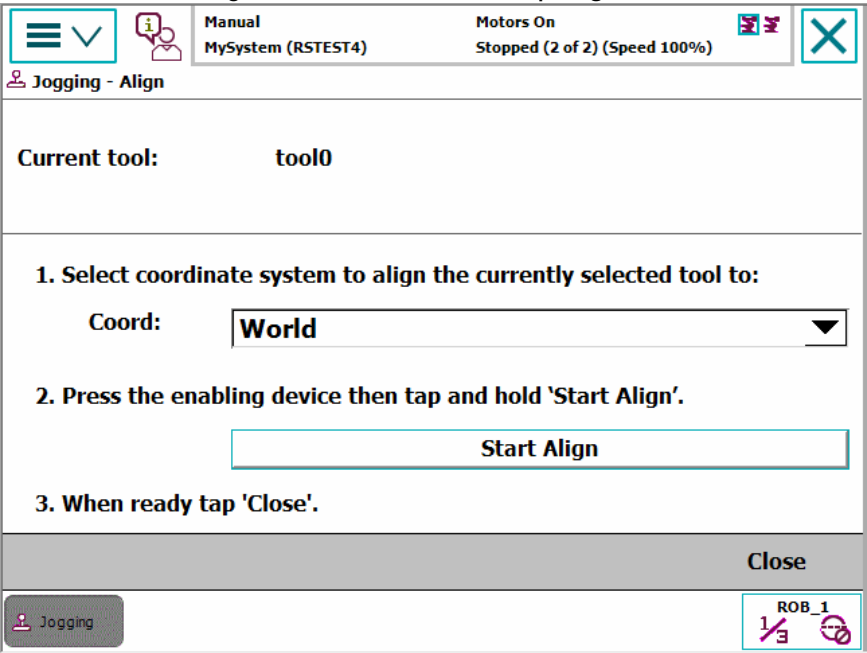
A tool can be aligned with another coordinate system.

When aligning a tool, the tool's z axis is aligned to the selected coordinate system's nearest axis. Therefore it is recommended to first jog the tool so it is close to the desired coordinates.

Note that the tool's data is not changed!

Aligning mechanical units

This procedure describes how to align tools.

	Action
1	On the ABB menu, tap Jogging .
2	<p>Make sure that the right tool is active and then tap Align....</p>  <p>1. Select coordinate system to align the currently selected tool to:</p> <p>Coord: <input type="text" value="World"/></p> <p>2. Press the enabling device then tap and hold 'Start Align'.</p> <p><input type="button" value="Start Align"/></p> <p>3. When ready tap 'Close'.</p> <p><input type="button" value="Close"/></p> <p><input type="button" value="Jogging"/></p> <p>en0500001548</p>
3	Select a coordinate system to align the selected tool to.
4	Press and hold the three-position enabling device and then tap and hold Start Align to start aligning the tool.
5	Tap Close when completed.

This page is intentionally left blank

4 Programming and testing

4.1 Before you start programming

Programming tools

You can use both the FlexPendant and RobotStudio for programming. The FlexPendant is best suited for modifying programs, such as positions and paths, while RobotStudio is preferred for more complex programming.

How to program using RobotStudio is described in *Operating manual - RobotStudio*.

Define tools, payloads, and work objects

Define tools, payloads and work objects before you start programming. You can always go back and define more objects later, but you should define your basic objects in advance.



WARNING

It is important to always define the actual tool load and, when used, the payload of the robot (for example, a gripped part). Incorrect definitions of load data can result in overloading of the robot mechanical structure. There is also a risk that the speed in manual reduced speed mode can be exceeded.

When incorrect load data is specified, it can often lead to the following consequences:

- The robot may not use its maximum capacity.
- Impaired path accuracy including a risk of overshooting.
- Risk of overloading the mechanical structure.

The controller continuously monitors the load and writes an event log if the load is higher than expected. This event log is saved and logged in the controller memory.

Define coordinate systems

Make sure the base and world coordinate systems have been set up properly during the installation of your robot system. Also make sure that additional axes have been set up.

Define tool and work object coordinate systems before you start programming. As you add more objects later you also need to define the corresponding coordinate systems.



Tip

For more details about the RAPID language and structure, see *Technical reference manual - RAPID Overview* and *Technical reference manual - RAPID Instructions, Functions and Data types*.

4 Programming and testing

4.2 Using RAPID programs

4.2 Using RAPID programs

Using the RAPID program

This procedure describes the main steps required in creating, saving, editing and debugging any RAPID program.

Note that there is more information available, than the one referred to in the procedure.

	Action	Information
1	Start by creating a RAPID program.	How to create a RAPID program is detailed in section Handling of programs on page 128 .
2	Edit your program.	Proceed as detailed in section Handling of instructions on page 139 .
3	To simplify programming and keep an overview of the program, you may want to divide the program into more than one module.	How to view, add, or delete a module is detailed in section Handling of modules on page 131 .
4	To further simplify programming, you may want to divide the module into more than one routine.	How to add or delete a routine is detailed in section Handling of routines on page 134 .
5	When programming you may want to work with: <ul style="list-style-type: none">• Tools• Work objects• Payloads	Also read the following sections: <ul style="list-style-type: none">• Creating a tool on page 156.• Creating a work object on page 171.• Creating a payload on page 179.
6	In order to deal with potential errors that may occur during program execution, you may want to create an error handler.	Error handlers are described in the <i>RAPID</i> manuals.
7	After completing the actual RAPID program, it will require testing before being put into production.	Proceed as detailed in section Testing on page 185 .
8	After test running your RAPID program, it may require altering. You may want to modify, or tune, programmed positions, the TCP positions, or paths.	How to modify positions while the program is running is described in section HotEdit menu on page 34 . How to modify positions in manual mode is described in section Modifying positions in the Program Editor or Production Window on page 249 .
9	Programs that are no longer required may be removed.	

Running the program

This procedure specifies how to use an existing RAPID program.

	Action	Information
1	Load an existing program.	Described in section Starting programs on page 225 .
2	When starting program execution, you may choose between running the program once, or running it continuously.	Described in section Quickset menu, Run Mode on page 64 .

Continues on next page

	Action	Information
3	Once the program has been loaded, you may start program execution.	Described in section Starting programs on page 225 and in Using multitasking programs on page 229 .
4	After program execution is completed, the program may be stopped.	Proceed as detailed in section Stopping programs on page 228 .

4 Programming and testing

4.3.1 Handling of programs

4.3 Programming concept

4.3.1 Handling of programs

Overview

This section details how to perform normal handling of robot programs. It describes how to:

- create a new program
- load an existing program
- save a program
- rename a program
- delete a program

Each task must contain *one* program, no more, no less. Note that the following procedures describe a single task system, i.e. only one task is available.

How to create a new program *when no program is available* is detailed in section [Creating a new program on page 128](#).

About program files

When saving a program to the controller hard disk, it is by default saved to the directory HOME in the system's folder unless otherwise stated. How to set another default path is detailed in section [Setting default paths on page 79](#).

The program is saved as a folder, named as the program, containing the actual program file, of type pgf.

When loading a program you open the program folder and select the pgf file.

When renaming a program you rename the program folder and the program file.

When saving a loaded program which is already saved to the hard disk, you must not open the existing program folder. Instead, you should save the program folder again and overwrite the old version, or rename the program.

Creating a new program

This section describes how to create a new program.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Tasks and Programs .
3	Tap File , then New Program . If there was already a program loaded, a warning dialog appears. <ul style="list-style-type: none">• Tap Save to save the loaded program.• Tap Don't Save to close loaded program without saving it, i.e. delete from program memory.• Tap Cancel to leave the program loaded.
4	Continue by adding instructions, routines, or modules. A new program is created.

Continues on next page

Loading an existing program

This section describes how to load an existing program.

	Action
1	On the ABB menu, tap Program Editor.
2	Tap Tasks and Programs.
3	Tap File, then Load Program. If there was already a program loaded, a warning dialog appears. <ul style="list-style-type: none"> • Tap Save to save the loaded program. • Tap Don't save to close loaded program without saving it, i.e. delete from program memory. • Tap Cancel to leave the loaded program.
4	Use the file searching tool to locate the program file to be loaded (file type pgf). Then tap OK. The program is loaded and the program code is displayed.

```

9  PROC main()
10 MoveL p10, v1000, z50, tool0;
11 MoveL p20, v100, z15, tool0;
12 WaitDI DI1, 1;
13 MoveL p30, v100, z15, tool0;
14 WaitDI DI2, 1;
15 MoveL p40, v100, z15, tool0;
16 MoveL p10, v100, z15, tool0;
17 ENDPROC
18 ENDMODULE
  
```

Saving a program

This section describes how to save a loaded program to the controller's hard disk.

A loaded program is automatically saved in the program memory, but saving to the controller hard disk is an extra precaution.

	Action
1	On the ABB menu, tap Program Editor.
2	Tap Tasks and Programs.
3	Tap File and select Save Program As....
4	Use the suggested program name or tap ... to open the soft keyboard and enter a new name. Then tap OK.

Continues on next page

4 Programming and testing

4.3.1 Handling of programs

Continued

Renaming a loaded program

This section describes how to rename a loaded program.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Tasks and Programs .
3	Tap File and select Rename Program . A soft keyboard is displayed.
4	Use the soft keyboard to enter the new name of the program. Then tap OK .

Deleting a program

This section describes how to delete a program.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Tasks and Programs .
3	Tap File and select Delete Program . A confirmation dialog is displayed.
4	Tap OK to delete, or Cancel to keep the program intact.

4.3.2 Handling of modules

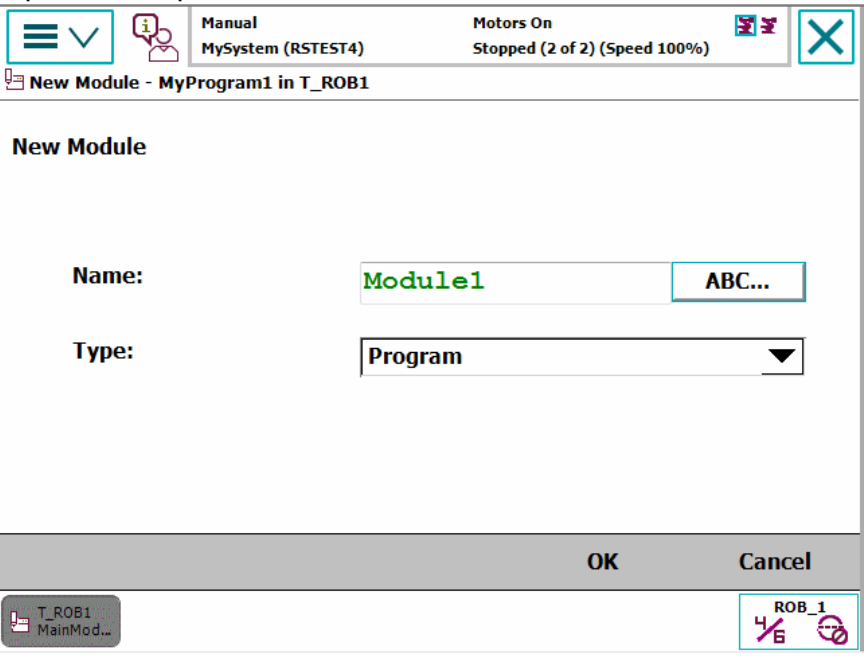
Overview

This section details how to handle program modules. i.e.:

- create a new module
- load an existing module
- save a module
- rename a module
- delete a module

Creating a new module

This section describes how to create a new module.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Modules .
3	<p>Tap File, then tap New Module.</p> 
4	Tap ABC... and use the soft keyboard to enter the new module's name. Then tap OK to close the soft keyboard.
5	<p>Select which type of module to be created:</p> <ul style="list-style-type: none"> • Program • System <p>Then tap OK.</p> <p>How to later switch between these types is detailed in section Changing type of module on page 133.</p>

Continues on next page

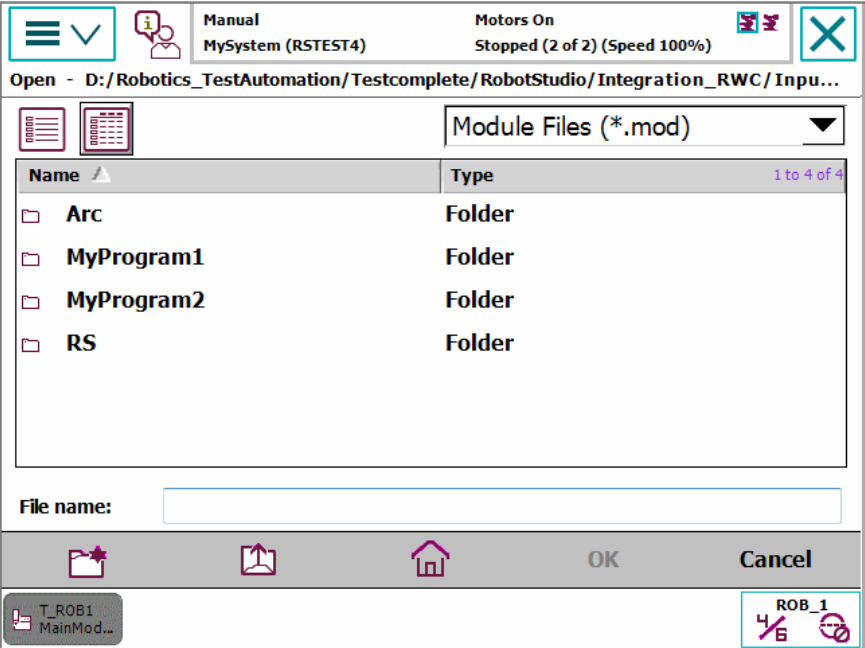
4 Programming and testing

4.3.2 Handling of modules

Continued

Loading an existing module

This section describes how to load an existing module.

Action	
1	On the ABB menu, tap Program Editor .
2	Tap Modules .
3	Tap File , then Load Module .  <p>en040000689</p> <p>Locate the module to be loaded. See section FlexPendant Explorer on page 36. A default path may be defined as detailed in section Setting default paths on page 79.</p>
4	Tap OK to load the selected module. The module is loaded.

Saving a module

This section describes how to save a module.

Action	
1	On the ABB menu, tap Program Editor .
2	Tap Modules and tap to select the module you want to load.
3	Tap File , then Save Module As...
4	Tap on the suggested file name and use the soft keyboard to enter the module's name. Then tap OK .
5	Use the file searching tool to locate where you want to save the module. See section FlexPendant Explorer on page 36 . The default location is on the controller disk, but any other location may be set as default as detailed in section Setting default paths on page 79 . Then tap OK . The module is saved.

Continues on next page

Renaming a module

This section describes how to rename a module.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Modules .
3	Tap File , then Rename Module... The soft keyboard is displayed.
4	Use the soft keyboard to enter the module's name. Then tap OK .

Changing type of module

This section describes how to change the type of module.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Modules and select the module to be changed.
3	Tap File , then Change declaration...
4	Tap Type and select module type.
5	Tap OK .

Deleting a module

This section describes how to delete a module from memory. If the module has been saved to disk, it will not be erased from the disk.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Modules and tap to select the module you want to delete.
3	Tap File , then Delete Module... A dialog box is displayed.
4	Tap OK to delete the module without saving it. If you want to save the module first, tap Cancel and save the module first. How to save the module is detailed in section Saving a module on page 132 .

4 Programming and testing

4.3.3 Handling of routines

4.3.3 Handling of routines

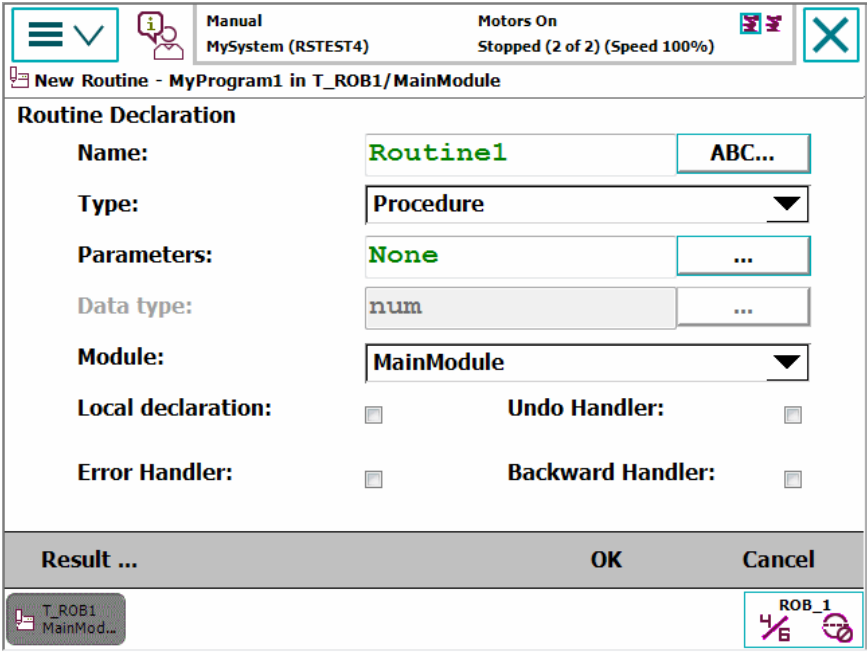
Overview

This section details how to handle program routines. i.e.:

- create a new routine
- create a copy of a routine
- change the declaration of a routine
- delete a routine

Creating a new routine

This section details how to create a new routine, set the declaration, and add it to a module.

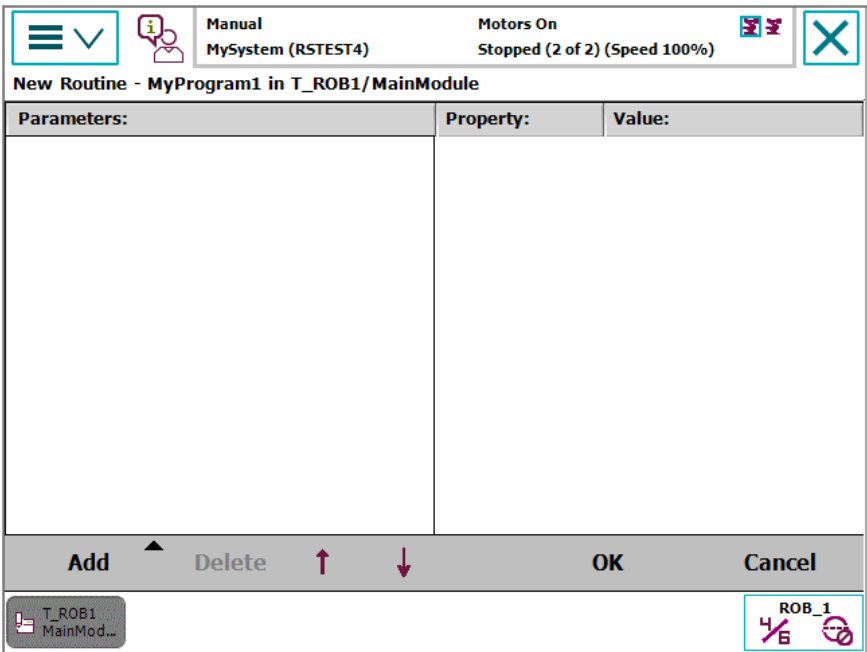
	Action
1	On the ABB menu, tap Program Editor .
2	Tap Routines .
3	Tap File , then New Routine . A new routine is created and displayed with default declaration values. 
4	Tap ABC... and use the soft keyboard to enter the new routines' name. Then tap OK .
5	Select the type of routine: <ul style="list-style-type: none">• Procedure: used for a normal routine without return value• Function: used for a normal routine with return value• Trap: used for an interrupt routine
6	Do you need to use any parameters? If YES; tap ... and proceed as detailed in section Defining parameters in routine on page 135 . If NO; proceed to the next step.
7	Select module to add the routine to.

Continues on next page

Action	
8	Tap the checkbox to select Local declaration if the routine should be local. A local routine can only be used in the selected module.
9	Tap OK.

Defining parameters in routine

This section describes how to define parameters in a routine.

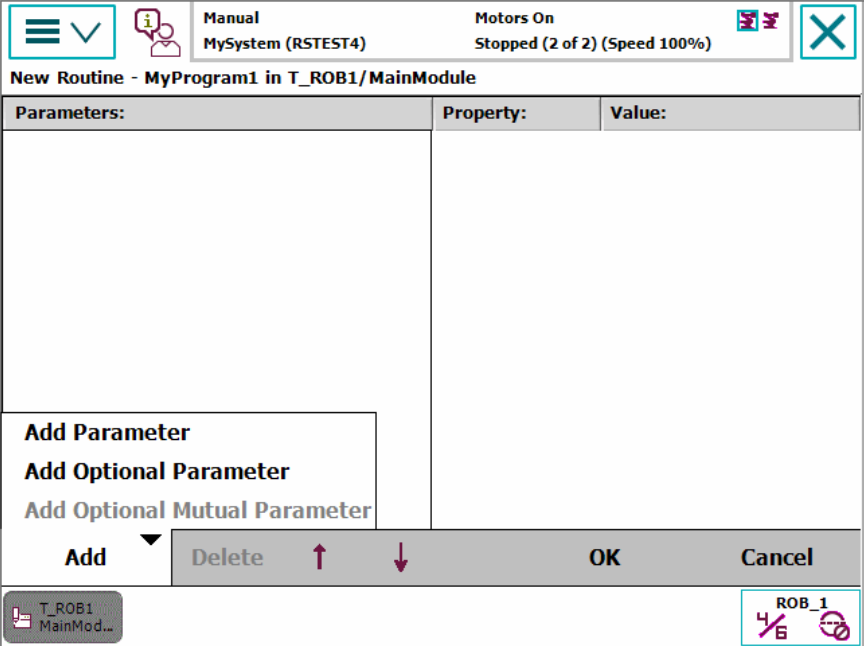
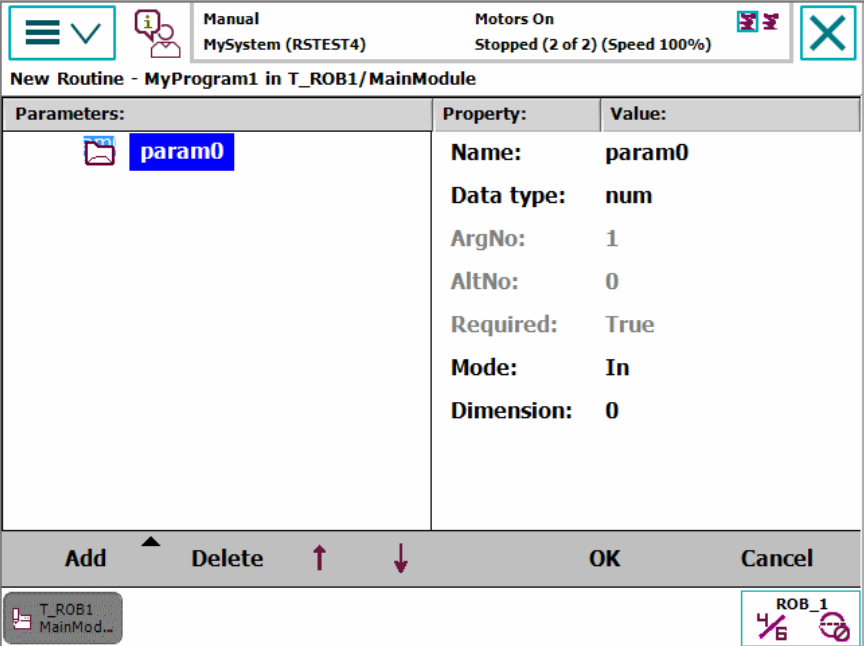
Action	
1	<p>In the routine declaration, tap ... to define parameters. A list of defined parameters is displayed.</p>  <p>The screenshot shows a software interface for defining parameters in a routine. At the top, there are navigation icons and status information: 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this is a title bar 'New Routine - MyProgram1 in T_ROB1/MainModule'. The main area contains a table with three columns: 'Parameters:', 'Property:', and 'Value:'. The table is currently empty. At the bottom of the dialog, there are several buttons: 'Add', 'Delete', 'OK', and 'Cancel'. There are also navigation arrows (up and down) between 'Delete' and 'OK'. At the very bottom, there is a button labeled 'T_ROB1 MainMod...' and a status indicator for 'ROB_1' showing a battery level of 4/6 and a stop icon.</p>

Continues on next page

4 Programming and testing

4.3.3 Handling of routines

Continued

Action	
2	<p>If no parameters are shown, tap Add to add a new parameter.</p> <ul style="list-style-type: none"> • Add optional parameter adds a parameter that is optional • Add optional mutual parameter adds a parameter that is mutually optional with another parameter <p>Read more about routine parameters in the RAPID reference manuals.</p>  <p>en0400000695</p>
3	<p>Use the soft keyboard to enter the name of the new parameter and then tap OK. The new parameter is displayed in the list.</p>  <p>en0400000696</p>
4	<p>Tap to select a parameter. To edit values, tap the value.</p>

Continues on next page

	Action
5	Tap OK to return to the routine declaration.

Creating a copy of a routine

This section describes how to create a copy of a routine.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Routines .
3	Highlight the routine by tapping it.
4	Tap File , then Copy Routine . The new routine is displayed. The name of the new routine is set to the same as the original with the suffix <i>Copy</i> .
5	Make any changes in the declarations for the new routine copy. Then tap OK . How to make all declarations is detailed in section Creating a new routine on page 134 .

Changing the declaration of a routine

This section describes how to change the declaration of a routine.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Routines .
3	Highlight the routine by tapping it.
4	Tap File , then Change Declaration
5	Change any declaration values for the routine. Then tap OK . Declaration settings are described in section Creating a new routine on page 134 .

Moving a routine

This section describes how to move a routine to another module.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Routines .
3	Highlight the routine by tapping it.
4	Tap File , then Move Routine...
5	Select task and module. Then tap OK .

Deleting a routine

This section describes how to delete a routine from memory.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Routines .
3	Highlight the routine by tapping it.
4	Tap File , then Delete Routine... A dialog box is displayed.

Continues on next page

4 Programming and testing

4.3.3 Handling of routines

Continued

	Action
5	Tap: <ul style="list-style-type: none">• OK to delete the routine without saving any changes made to it.• Cancel to revert without deleting the routine.

4.3.4 Handling of instructions

Instructions

A RAPID program consists of instructions. An instruction can, for example, move the robot, set an I/O signal, or write a message to the operator.

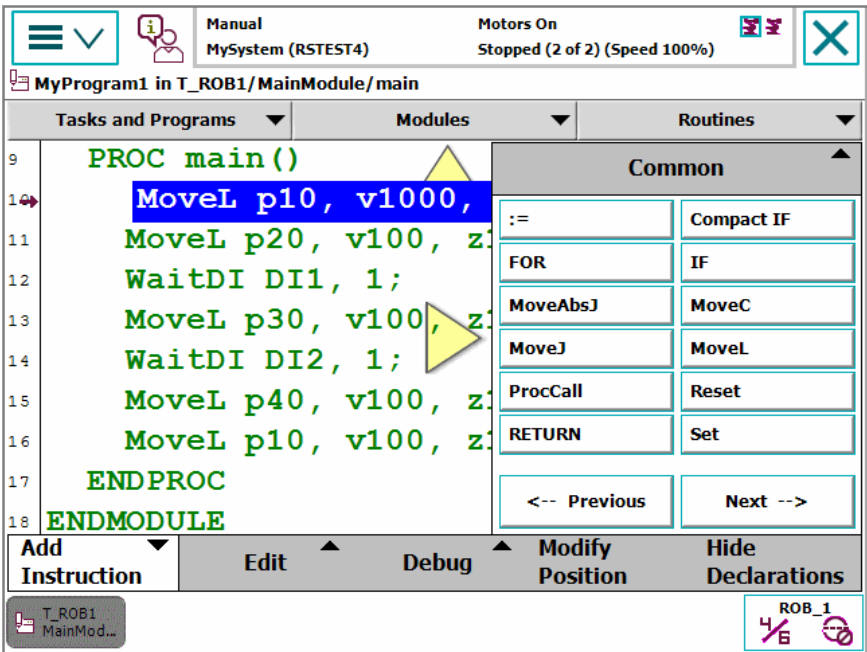
A large number of instructions are available, and these are listed in *Technical reference manual - RAPID Instructions, Functions and Data types*. The basic procedure for adding instructions are, however, identical.

Undo and redo

When editing programs in the Program editor, you can undo and redo up to three steps. This function is available in the **Edit** menu.

Adding instructions

This section describes how to add instructions.

Action	
1	On the ABB menu, tap Program Editor .
2	Tap to highlight the instruction under which you want to add a new instruction.
3	<p>Tap Add instruction. A category of instructions is displayed.</p>  <p>The screenshot shows the ABB RAPID program editor interface. At the top, there are status indicators for 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this, the program name 'MyProgram1 in T_ROB1/MainModule/main' is displayed. The main editing area shows a list of instructions: 'PROC main()', 'MoveL p10, v1000,', 'MoveL p20, v100, z', 'WaitDI DI1, 1;', 'MoveL p30, v100, z', 'WaitDI DI2, 1;', 'MoveL p40, v100, z', 'MoveL p10, v100, z', 'ENDPROC', and 'ENDMODULE'. A blue highlight is over the 'MoveL p10, v1000,' instruction. To the right, a 'Common' category list is open, showing various instruction types like ':=', 'FOR', 'MoveAbsJ', 'MoveJ', 'ProcCall', 'RETURN', 'Compact IF', 'IF', 'MoveC', 'MoveL', 'Reset', and 'Set'. Navigation buttons '<-- Previous' and 'Next -->' are at the bottom of the list.</p>
4	<p>Tap Common to display a list of the available categories. You can also tap Previous/Next at the bottom of the list of instructions to move to the next/previous category.</p>

Continues on next page

4 Programming and testing

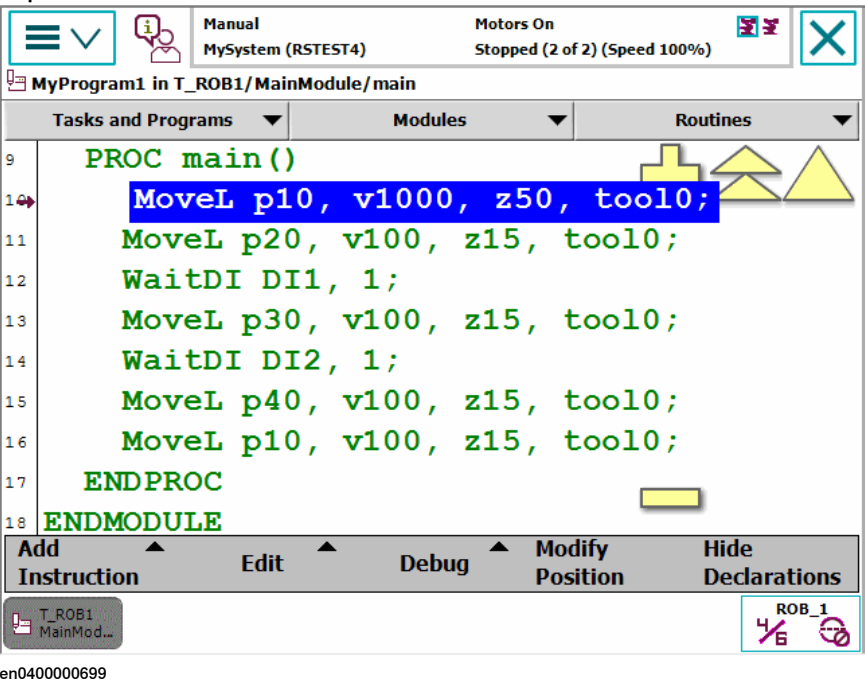
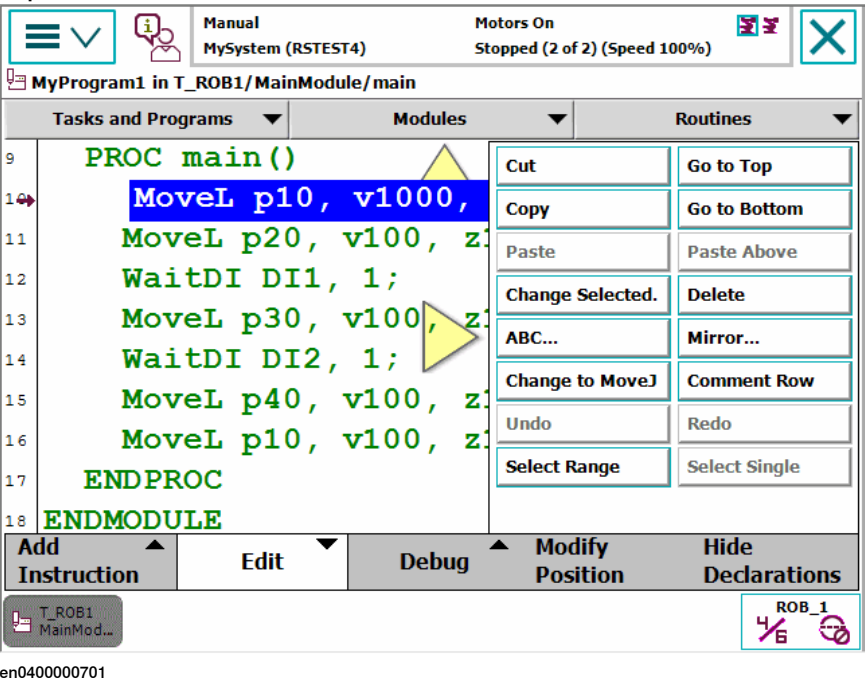
4.3.4 Handling of instructions

Continued

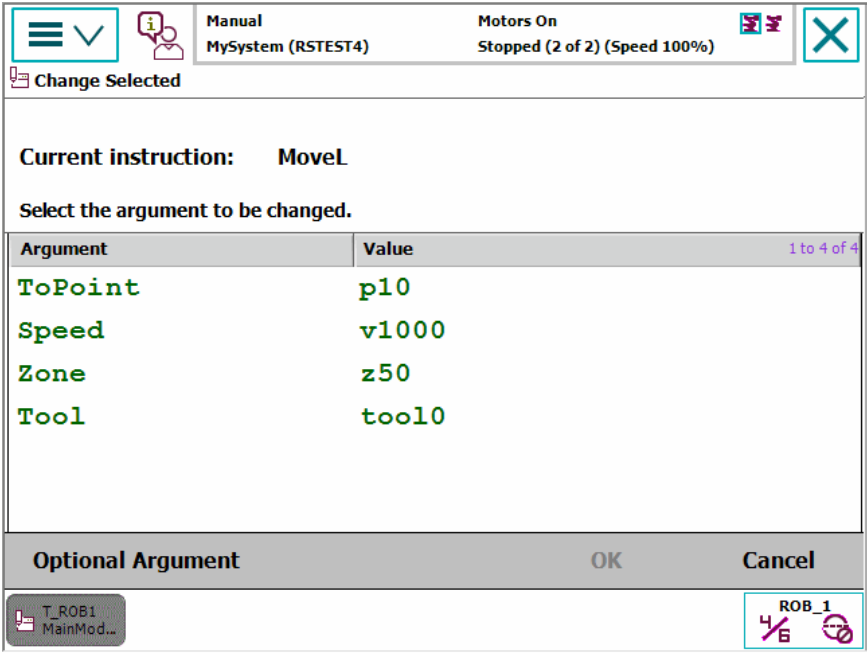
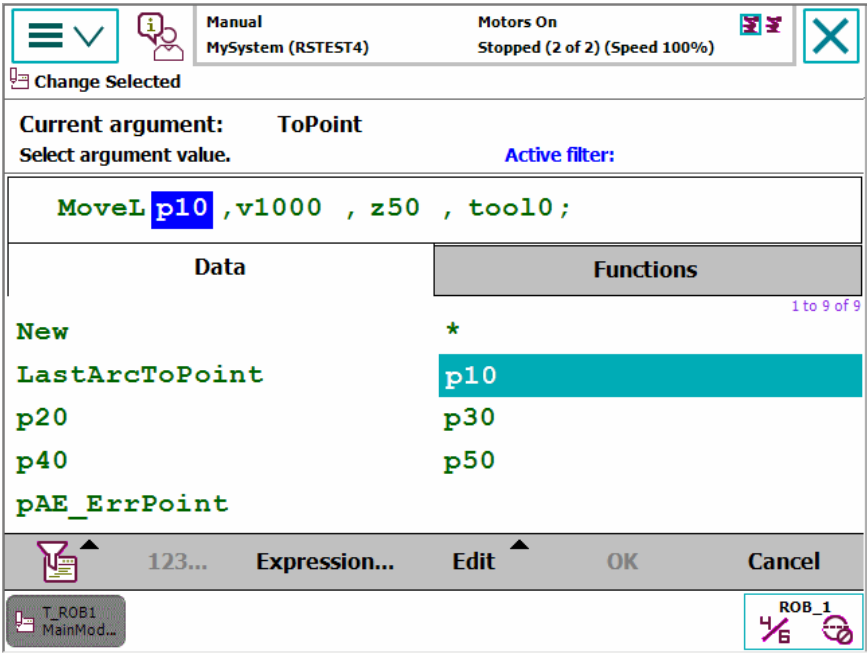
Action	
5	Tap the instruction you want to add. The instruction is added to the code.

Editing instruction arguments

This section describes how to edit instruction arguments.

Action	
1	<p>Tap the instruction to edit.</p> 
2	<p>Tap Edit.</p> 

Continues on next page

Action													
3	<p>Tap Change Selected. Depending on the type of instruction, the arguments have different data types. Use the soft keyboard to change string values or proceed to the next steps for other data types or multiple argument instructions.</p>  <p>Current instruction: MoveL</p> <p>Select the argument to be changed.</p> <table border="1"> <thead> <tr> <th>Argument</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>ToPoint</td> <td>p10</td> </tr> <tr> <td>Speed</td> <td>v1000</td> </tr> <tr> <td>Zone</td> <td>z50</td> </tr> <tr> <td>Tool</td> <td>tool0</td> </tr> </tbody> </table> <p>Optional Argument OK Cancel</p> <p>T_ROB1 MainMod... ROB_1</p> <p>en0400000702</p>	Argument	Value	ToPoint	p10	Speed	v1000	Zone	z50	Tool	tool0		
Argument	Value												
ToPoint	p10												
Speed	v1000												
Zone	z50												
Tool	tool0												
4	<p>Tap the argument to be changed. A number of options are displayed.</p>  <p>Current argument: ToPoint</p> <p>Select argument value. Active filter:</p> <p>MoveL p10, v1000, z50, tool0;</p> <table border="1"> <thead> <tr> <th>Data</th> <th>Functions</th> </tr> </thead> <tbody> <tr> <td>New</td> <td>*</td> </tr> <tr> <td>LastArcToPoint</td> <td>p10</td> </tr> <tr> <td>p20</td> <td>p30</td> </tr> <tr> <td>p40</td> <td>p50</td> </tr> <tr> <td>pAE_ErrPoint</td> <td></td> </tr> </tbody> </table> <p>123... Expression... Edit OK Cancel</p> <p>T_ROB1 MainMod... ROB_1</p> <p>en0400000703</p>	Data	Functions	New	*	LastArcToPoint	p10	p20	p30	p40	p50	pAE_ErrPoint	
Data	Functions												
New	*												
LastArcToPoint	p10												
p20	p30												
p40	p50												
pAE_ErrPoint													
5	<p>Tap an existing data instance to select and then tap OK to complete, or tap Expression</p>												

Continues on next page

4 Programming and testing

4.3.4 Handling of instructions

Continued



Tip

Tapping twice on an instruction will automatically launch the Change selected option. Tapping twice on an instruction argument will automatically launch the argument editor.

Copying and pasting instructions or arguments

This section describes how to paste instructions or arguments.

	Action
1	Tap to select the argument or instruction you want to copy. To select more than one row: select the first row, tap Select Range in the Edit menu and then tap the last row.
2	Tap Edit and then tap Copy .
3	Place the cursor on the instruction above where you want to paste the instruction or argument, or tap on the argument or instruction you want to change and tap Paste .

Cutting an instruction

This section describes how to cut an instruction.

	Action
1	Tap to select the instruction you want to cut. To select more than one row: select the first row, tap Select Range in the Edit menu and then tap the last row.
2	Tap Edit and then tap Cut .

Changing motion mode for a move instruction

This section describes how to change the motion mode for a move instruction.

	Action
1	Tap to select the move instruction you want to change and then tap Edit .
2	Tap Change to MoveJ or Change to MoveL . The change is performed.

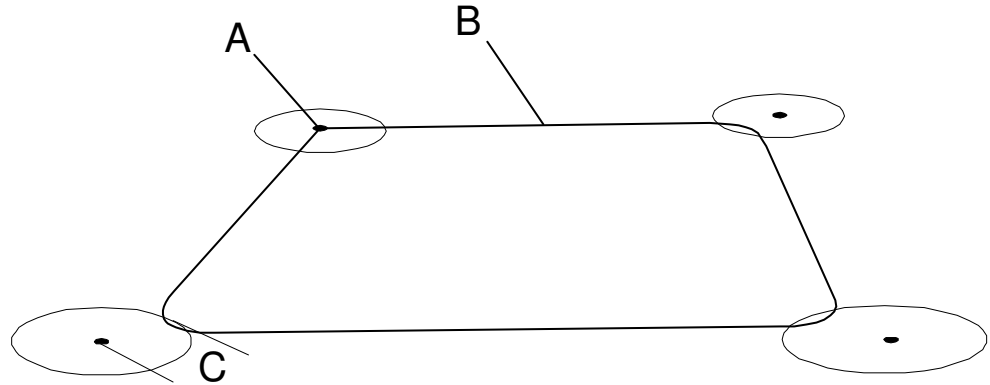
Commenting instruction rows

Instruction rows can be commented, i.e. skipped in the program execution. The comment/uncomment command is found under the **Edit** menu in the **Program Editor**.

4.3.5 Example: Add movement instructions

Overview

In this example you will create a simple program that makes the robot move in a square. You need four movement instructions to complete this program.



en0400000801

A	First point
B	Robot movement Speed data v50 = speed 50mm/s
C	Zone z50 = (50mm)

Add movement instructions

This section details how to add movement instructions.

	Action	Information
1	Jog the robot to the first point.	Tip: Use only left-right/up-down joystick movements to jog in a square.
2	In the program editor, tap Add Instruction .	
3	Tap MoveL to insert a <code>MoveL</code> instruction.	
4	Repeat for the next four positions of the square.	
5	For the first and last instruction. Tap z50 in the instruction, tap Edit and then Change selected to Fine . Tap OK	

Result

Your program code should look like this:

```

Proc main()
  MoveL *, v50, fine, tool0;
  MoveL *, v50, z50, tool0;
  MoveL *, v50, z50, tool0;
  MoveL *, v50, z50, tool0;
  MoveL *, v50, fine, tool0;
End Proc;

```

4 Programming and testing

4.3.6 About the Program and Motion Pointers

4.3.6 About the Program and Motion Pointers

The Program Pointer

The Program Pointer (PP) indicates the instruction with which the program will start when you press any of the **Start**, **Forward**, or **Backward** buttons on the FlexPendant.

Program execution continues from the instruction where the Program Pointer is. However, if the cursor is moved to another instruction when the program is stopped, the Program Pointer can be moved to the position of the cursor (or the cursor can be moved to the Program Pointer), and execution can be restarted from there.

The Program Pointer is shown as a yellow arrow to the left of the program code in the **Program Editor** and **Production Window**.

The Motion Pointer

The Motion Pointer (MP) indicates the instruction that the robot is currently executing. This is normally one or more instructions after the Program Pointer, as the system executes and calculates the robot path faster than the robot moves.

The Motion Pointer is shown as a small robot to the left of the program code in the **Program Editor** and in the **Production Window**.

The cursor

The cursor can indicate a complete instruction or any of the arguments.

The cursor is shown as blue highlighting of the program code in the **Program Editor**.

Program Editor

If you toggle between the **Program Editor** and another view and back again, the **Program Editor** will show the same part of the code as long as the program pointer has not been moved. If the program pointer is moved, the **Program Editor** shows the code at the position of the program pointer.

The same behavior applies to the **Production Window**.

Related information

[Production Window on page 40.](#)

[Program Editor on page 43.](#)

[Stepping instruction by instruction on page 193.](#)

[Starting programs on page 225.](#)

4.4 Data types

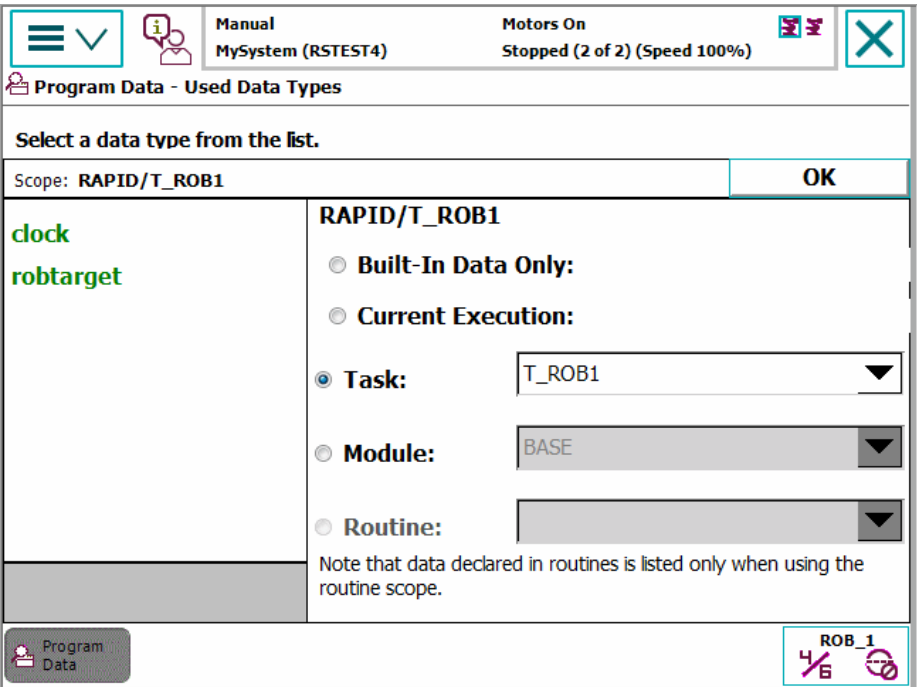
4.4.1 Viewing data in specific tasks, modules, or routines

Overview

It is possible to view selections of data types by selecting a specific scope.

Viewing data in specific tasks, modules, or routines

This section details how to view data instances in specific modules or routines.

	Action
1	On the ABB menu, tap Program Data .
2	<p>Tap Change Scope. The following screen is displayed:</p>  <p>en040000661</p>
3	<p>Select the required scope by selecting:</p> <ul style="list-style-type: none"> • Built-In Data Only: Shows all data types used by the specific system • Current execution: Shows all data types used in the current execution • Task: Shows all data types used by a specific task • Module: Shows all data types used by a specific module • Routine: Shows all data types used by a specific routine
4	Tap OK to confirm your choice.
5	Tap twice to select a data type and view its instances.

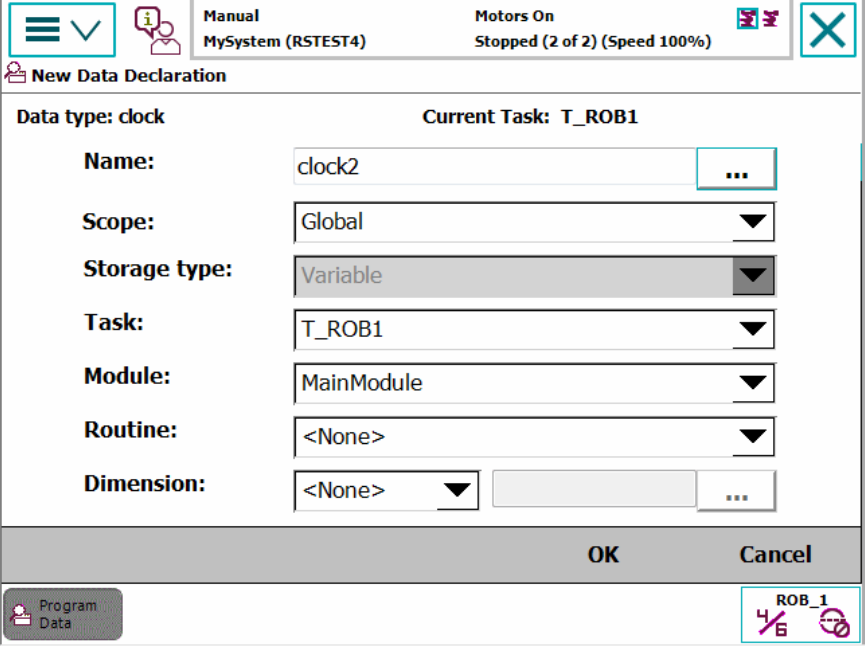
4 Programming and testing

4.4.2 Creating new data instance

4.4.2 Creating new data instance

Creating new data instance

This section details how to create new data instances of data types.

Action	
1	On the ABB menu, tap Program Data . A list of all available data types is displayed.
2	Tap the data instance type to be created, i.e. bool and then tap Show data . A list of all instances of the data type is displayed.
3	Tap New .  en040000663
4	Tap ... the right of Name to define the data instance's name. Name
5	Tap the Scope menu to set accessibility for the data instance. Select: <ul style="list-style-type: none">• Global - reachable by all tasks• Local - reachable within the module• Task - reachable within the task
6	Tap the Storage type menu to select type of memory used for the data instance. Select: <ul style="list-style-type: none">• Persistent if the data instance is persistent• Variable if the data instance is variable• Constant if the data instance is constant
7	Tap the Module menu to select module.
8	Tap the Routine menu to select routine.
9	If you want to create an array of data instances, then tap the Dimensions menu and select the number of dimensions in the array, 1-3. <ul style="list-style-type: none">• 1• 2• 3• None Then tap ... to set the Size of the array's axes.

Continues on next page

4 Programming and testing

4.4.2 Creating new data instance

Continued

	Action
10	Tap OK.

4 Programming and testing

4.4.3 Editing data instances

4.4.3 Editing data instances

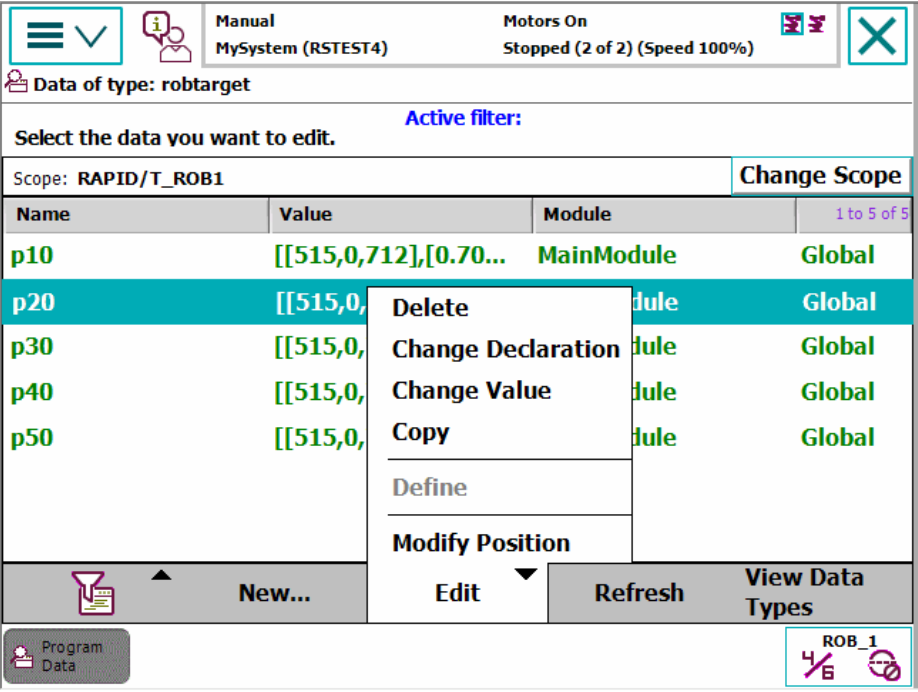
Overview

This section describes how to view data instances in the **Program Data** window. It also details how to edit, delete, change declaration of, copy, and define a data instance.

For the data types `tooldata`, `wobjdata` and `loaddata` also see sections [Tools on page 152](#), [Work objects on page 170](#) or [Payloads on page 179](#).

Viewing data instances

This section details how to view the available instances of a data type.

Action																									
1	On the ABB menu, tap Program Data .																								
2	Tap the data type you want view and then tap Show Data .																								
3	<p>Tap the data instance you want to edit, and then tap Edit.</p>  <p>The screenshot shows the 'Program Data' window for 'robtarget' data. It displays a table with the following data:</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Module</th> <th>Scope</th> </tr> </thead> <tbody> <tr> <td>p10</td> <td>[[515,0,712],[0.70...</td> <td>MainModule</td> <td>Global</td> </tr> <tr> <td>p20</td> <td>[[515,0,</td> <td>Module</td> <td>Global</td> </tr> <tr> <td>p30</td> <td>[[515,0,</td> <td>Module</td> <td>Global</td> </tr> <tr> <td>p40</td> <td>[[515,0,</td> <td>Module</td> <td>Global</td> </tr> <tr> <td>p50</td> <td>[[515,0,</td> <td>Module</td> <td>Global</td> </tr> </tbody> </table> <p>The context menu for p20 includes: Delete, Change Declaration, Change Value, Copy, Define, and Modify Position.</p>	Name	Value	Module	Scope	p10	[[515,0,712],[0.70...	MainModule	Global	p20	[[515,0,	Module	Global	p30	[[515,0,	Module	Global	p40	[[515,0,	Module	Global	p50	[[515,0,	Module	Global
Name	Value	Module	Scope																						
p10	[[515,0,712],[0.70...	MainModule	Global																						
p20	[[515,0,	Module	Global																						
p30	[[515,0,	Module	Global																						
p40	[[515,0,	Module	Global																						
p50	[[515,0,	Module	Global																						
4	<p>Depending on what you want to do, tap one of the following menu items:</p> <ul style="list-style-type: none"> • Tap Delete to remove the data instance. • Tap Change Declaration to change the declaration of the data instance. • Tap Change Value to edit the value of the data instance. • Tap Copy to copy the data instance. • Tap Define to define the instance (only available for <code>tooldata</code>, <code>wobjdata</code> and <code>loaddata</code>). • Tap Modify Position to modify a position (only available for <code>robtarget</code> and <code>jointtarget</code>). <p>Proceed as described in the respective section following below.</p>																								

Continues on next page

Editing the value of a data instance

This section describes how to edit a data instance value.

	Action	Information
1	Tap Change Value to open the instance.	
2	Tap the value to open a keyboard or list of choices.	The way to edit a value depends on the data type and possible values, for instance text, numbers, predefined values etc.
3	Select or enter a new value.	
4	Tap OK .	



Note

If the value of a persistent variable is changed at any point in a running program, the **Program Editor** will still show the old value until the program stops. The **Program Data** view, however, always shows the current value of persistent variables. See *Persistent declaration* in the *Technical reference manual - RAPID Overview* for further information.

Deleting a data instance

This section details how to delete a data instance.



Note

A data instance can be of type **tool**, **work object**, **payload** or others.

	Action
1	Tap Delete in the menu for the data instance to be deleted, as detailed in section Viewing data instances on page 148 . A dialog box is displayed.
2	Tap Yes if you are sure the data instance is to be deleted.



CAUTION

A deleted tool, work object or payload cannot be recovered, and all related data will be lost. If the tool, work object or payload is referenced by any program, those programs cannot run without changes.

If you delete a tool you cannot continue the program from the current position.

Continues on next page

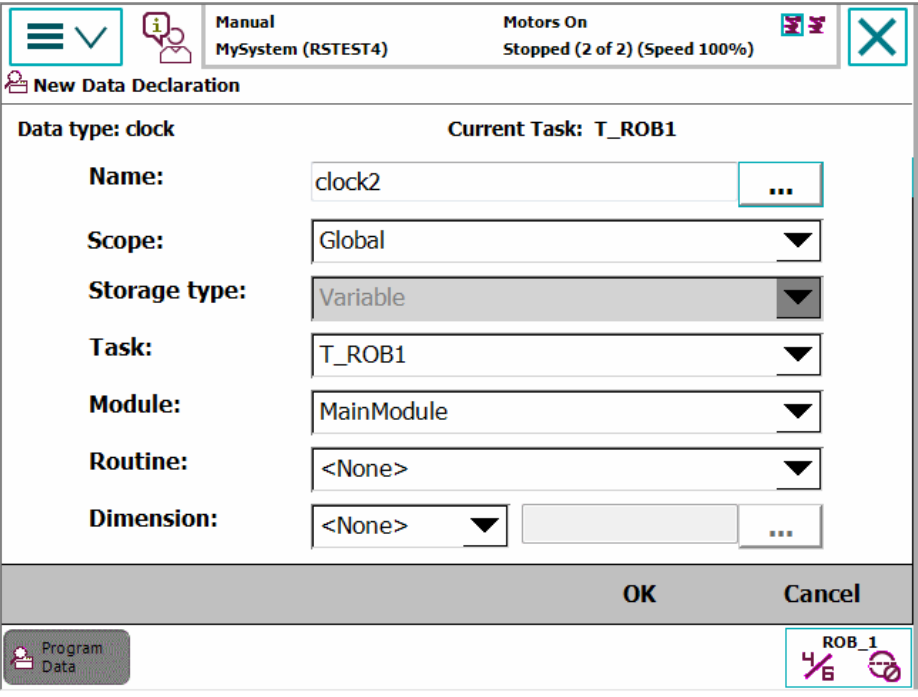
4 Programming and testing

4.4.3 Editing data instances

Continued

Changing the declaration of a data instance

This section details how to change the declaration of a data instance.

Action	
1	<p>Tap Change Declaration in the menu for the data instance to be deleted, as detailed in section Viewing data instances on page 148.</p>  <p>The screenshot shows a software interface for defining a data instance. At the top, there's a status bar with 'Manual MySystem (RSTEST4)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below that is a title bar 'New Data Declaration'. The main area contains several fields: 'Data type: clock', 'Current Task: T_ROB1', 'Name: clock2', 'Scope: Global', 'Storage type: Variable', 'Task: T_ROB1', 'Module: MainModule', 'Routine: <None>', and 'Dimension: <None>'. There are 'OK' and 'Cancel' buttons at the bottom. A 'Program Data' button is on the left, and a 'ROB_1' button is on the right.</p>
2	<p>Select what data instance values to be changed:</p> <ul style="list-style-type: none"> • Name: Tap ... to bring out the soft keyboard and change the name. • Scope • Storage type • Module • Routine

Copying a data instance

This section details how to copy a data instance.

Action	
1	<p>Tap Copy in the menu for the data instance to be copied, as detailed in section Viewing data instances on page 148. A copy of the data instance is created. The copy has the same values as the original, but the name is unique.</p>

Defining a data instance

How to define the tool frame or work object frame is described in the sections [Defining the tool frame on page 159](#) and [Defining the work object coordinate system on page 172](#).

Continues on next page

Modifying position of a data instance

Only instances of data types `robtarget` and `jointtarget` can use the function **Modify Position**. The currently active work object and tool will be used in the operation.

More information about modifying positions is detailed in [Modifying and tuning positions on page 248](#).



Note

Make sure that the correct work object and tool are selected when modifying positions in the **Program Data** window. This is not verified automatically by the system.

4 Programming and testing

4.5.1 What is a tool?

4.5 Tools

4.5.1 What is a tool?

Tool

A tool is an object that can be mounted directly or indirectly on the robot turning disk or fitted in a fixed position within the robot working range.



Note

A fixture (jig) is not a tool.

All tools must be defined with a TCP (Tool Center Point).

Each tool that can be used by the robot must be measured and its data stored in order to achieve accurate positioning of the tool center point.



WARNING

It is important to always define the actual tool load and, when used, the payload of the robot (for example, a gripped part). Incorrect definitions of load data can result in overloading of the robot mechanical structure. There is also a risk that the speed in manual reduced speed mode can be exceeded.

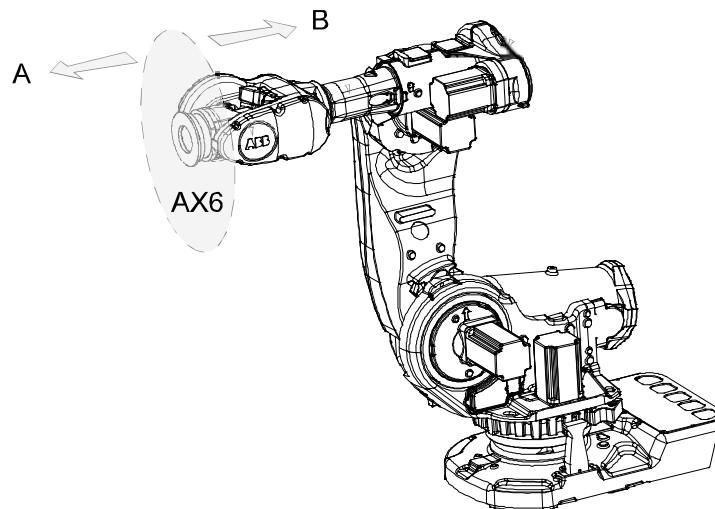
When incorrect load data is specified, it can often lead to the following consequences:

- The robot may not use its maximum capacity.
- Impaired path accuracy including a risk of overshooting.
- Risk of overloading the mechanical structure.

The controller continuously monitors the load and writes an event log if the load is higher than expected. This event log is saved and logged in the controller memory.

Continues on next page

Illustration



en040000803

A	Tool side
B	Robot side

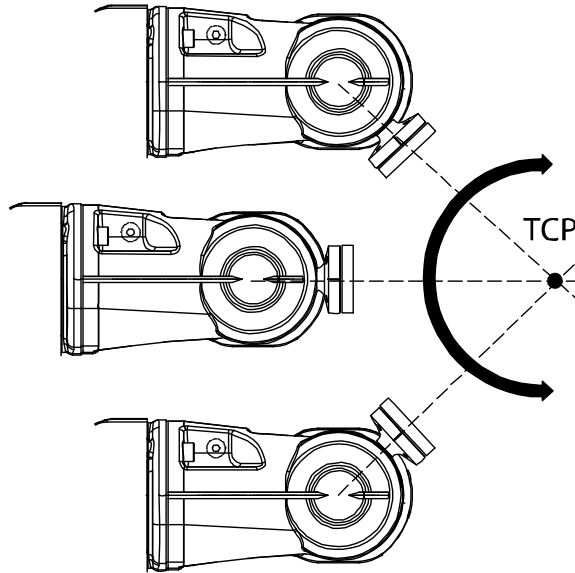
4 Programming and testing

4.5.2 What is the tool center point?

4.5.2 What is the tool center point?

Illustration

The illustration shows how the tool center point (TCP) is the point around which the orientation of the tool/manipulator wrist is being defined.



xx030000604

Description

The tool center point (TCP) is the point in relation to which all robot positioning is defined. Usually the TCP is defined as relative to a position on the manipulator turning disk.



CAUTION

Incorrect settings for the TCP will result in incorrect speed. Always verify the speed after changing the settings.

The TCP will be jogged or moved to the programmed target position. The tool center point also constitutes the origin of the tool coordinate system.

The robot system can handle a number of TCP definitions, but only one can be active at any one time.

There are two basic types of TCPs: moveable or stationary.

Moving TCP

The vast majority of all applications deal with moving TCP, i.e. a TCP that moves in space along with the manipulator.

A typical moving TCP can be defined in relation to, for example the tip of a arc welding gun, the center of a spot welding gun, or the end of a grading tool.

Continues on next page

Stationary TCP

In some applications a stationary TCP is used, for example when a stationary spot welding gun is used. In such cases the TCP can be defined in relation to the stationary equipment instead of the moving manipulator.

4 Programming and testing

4.5.3 Creating a tool

4.5.3 Creating a tool

What happens when you create a tool?

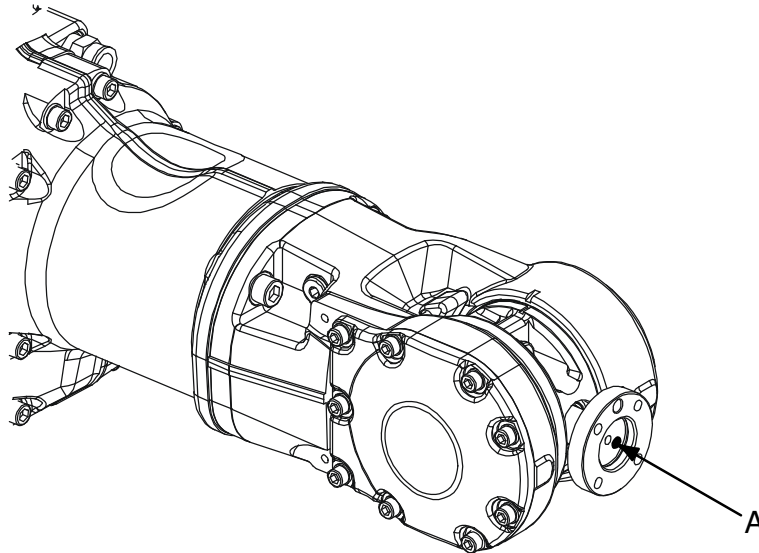
When you create a new tool a variable of the data type `tooldata` is created. The variable name will be the name of the tool. For more information on data types, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

The new tool has initial default values for mass, frame, orientation etc., which must be defined before the tool can be used.

How to create a tool

The tool center point of the default tool (`tool0`) is in the center of the robot's mounting flange and shares the orientation of the robot base.

By creating a new tool you define another tool center point. For more information about tools and the tool center points see [What is a tool? on page 152](#) and [What is the tool center point? on page 154](#).

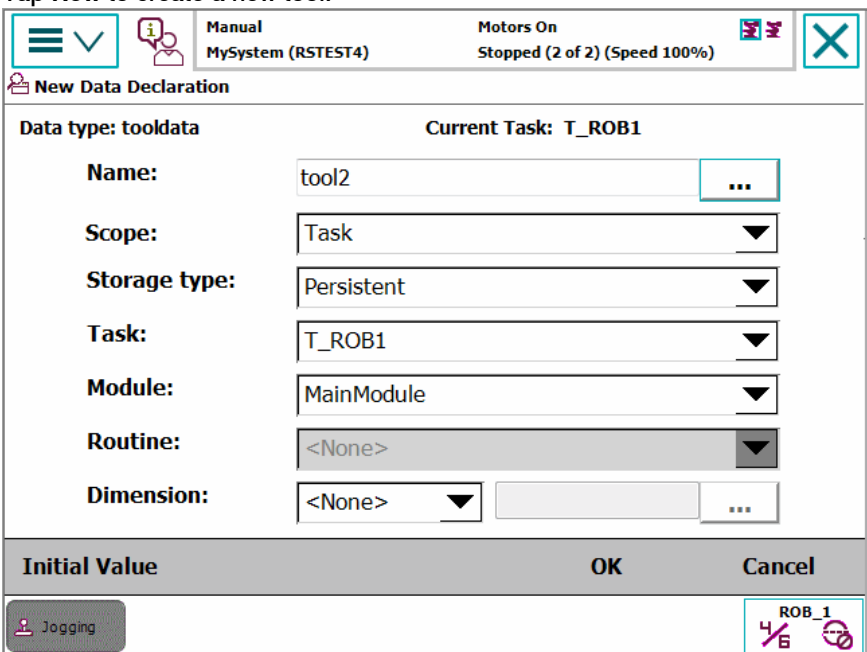


en0400000779


A	Tool center point, TCP, for tool0
---	-----------------------------------

Action	
1	On the ABB menu, tap Jogging .
2	Tap Tool to display the list of available tools.

Continues on next page

Action	
3	<p>Tap New to create a new tool.</p>  <p>en0300000544</p> <p>Enter values for each field, see table below.</p>
4	Tap OK.

Tool declaration settings

If you want to change...	then...	Recommendation
the name of the tool	tap ... button next to Name	<p>Tools are automatically named <code>tool</code> followed by a running number, for example <code>tool10</code> or <code>tool21</code>.</p> <p>You are recommended to change this to something more descriptive such as gun, gripper or welder.</p> <p> Note</p> <p>If you change the name of a tool after it is referenced in any program you must also change all occurrences of that tool.</p>
the scope	select the preferred scope from the menu	Tools should always be global, as to be available to all modules in the program.
the storage type	-	Tool variables must always be persistent.
the module	select the module in which this tool should be declared from the menu	

Continues on next page

4 Programming and testing

4.5.3 Creating a tool

Continued

If you want to change...	then...	Recommendation
the size of the data array's axes	tap ... button next to Dimension	



Note

The created tool is not useful until you have defined the tool data (TCP coordinates, orientation, weight etc.). See [Editing the tool data on page 163](#) and [LoadIdentify, load identification service routine on page 204](#) to learn more about how to do it.

4.5.4 Defining the tool frame

Preparations

To define the tool frame, you first need a reference point in the world coordinate system. If you need to set the tool center point orientation, you also need to affix elongators to the tool.

You also need to decide which method to use for the tool frame definition.

Available methods

There are three different methods which can be used when defining the tool frame. All three require that you define the cartesian coordinates of the tool center point. What differs is how the orientation is defined.

If you want to...	...then select
set the orientation the same as the orientation of the robot's mounting plate	TCP (default orient.)
set the orientation in Z axis	TCP&Z
set the orientation in X and Z axes	TCP&Z,X

How to select a method

This procedure describes how to select the method to be used when defining the tool frame.

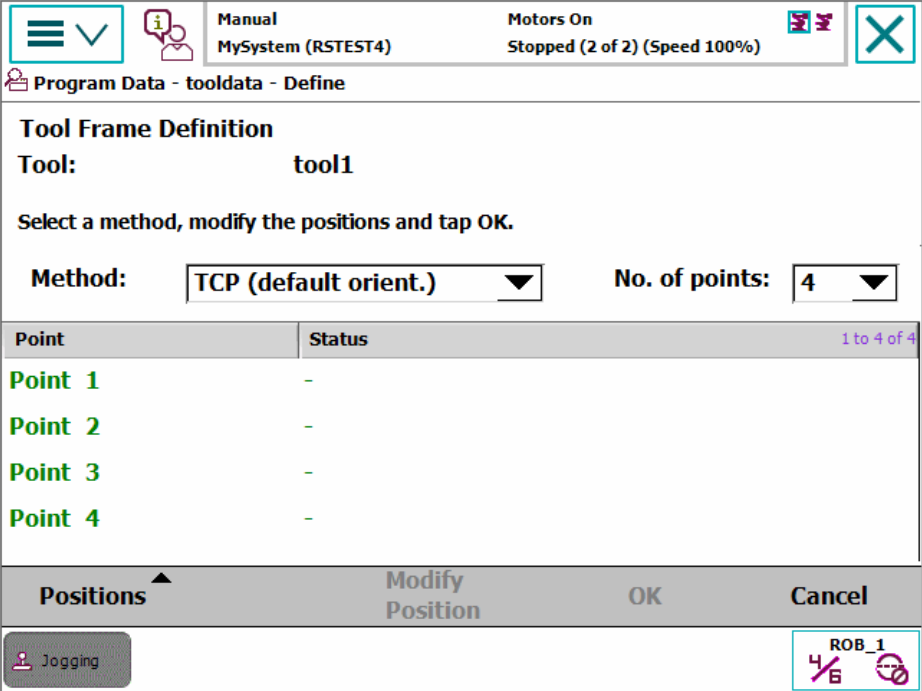
	Action
1	On the ABB menu, tap Jogging .
2	Tap Tool to display a list of available tools.
3	Select the tool you want to define.
4	In the Edit menu, tap Define

Continues on next page

4 Programming and testing

4.5.4 Defining the tool frame

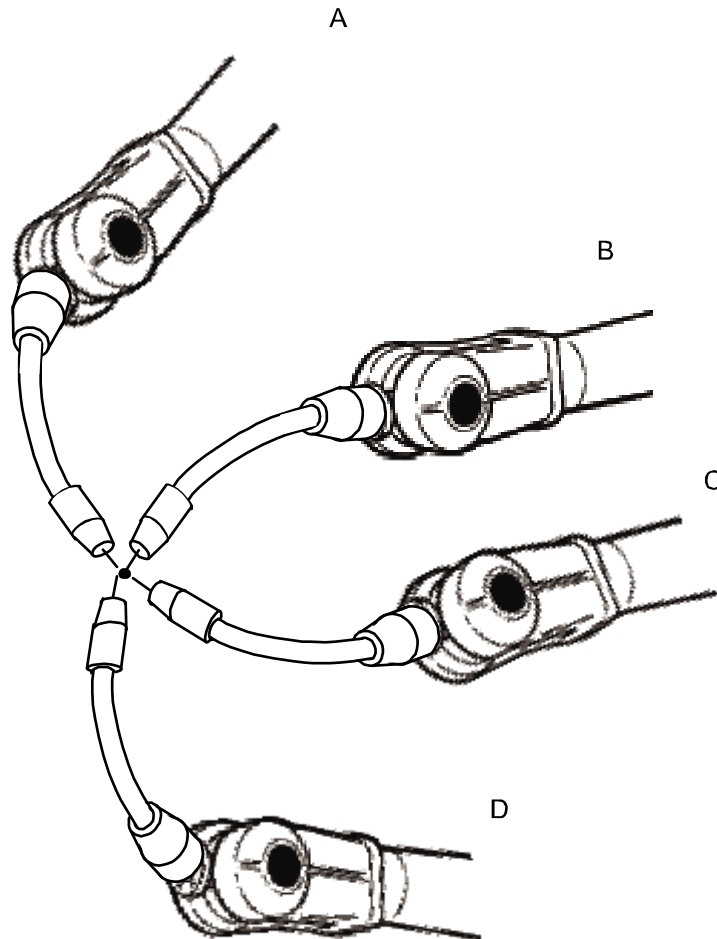
Continued

Action											
5	<p>In the dialog box which appears, select the method to use.</p>  <p>Program Data - tooldata - Define</p> <p>Tool Frame Definition</p> <p>Tool: tool1</p> <p>Select a method, modify the positions and tap OK.</p> <p>Method: TCP (default orient.) No. of points: 4</p> <table border="1"><thead><tr><th>Point</th><th>Status</th></tr></thead><tbody><tr><td>Point 1</td><td>-</td></tr><tr><td>Point 2</td><td>-</td></tr><tr><td>Point 3</td><td>-</td></tr><tr><td>Point 4</td><td>-</td></tr></tbody></table> <p>Positions Modify Position OK Cancel</p> <p>Jogging ROB_1</p> <p>en0600003147</p>	Point	Status	Point 1	-	Point 2	-	Point 3	-	Point 4	-
Point	Status										
Point 1	-										
Point 2	-										
Point 3	-										
Point 4	-										
6	<p>Select the number of approach points to use. Usually 4 points is enough. If you choose more points to get a more accurate result, you should be equally careful when defining all of them.</p>										
7	<p>See How to proceed with tool frame definition on page 161 for information on how to gather positions and perform the tool frame definition.</p>										

Continues on next page

How to proceed with tool frame definition

This procedure describes how to define the tool center point in Cartesian coordinates.



en040000906

	Action	Information
1	Jog the robot to an appropriate position, A, for the first approach point.	Use small increments to accurately position the tool tip as close to the reference point as possible.
2	Tap Modify Position to define the point.	
3	Repeat step 1 and 2 for each approach point to be defined, positions B, C, and D.	Jog away from the fixed world point to achieve the best result. Just changing the tool orientation will not give as good a result.
4	If the method you are using is TCP&Z or TCP&Z,X orientation must be defined as well.	Follow the instructions in How to define elongator points on page 162 .
5	If, for some reason, you want to redo the calibration procedure described in step 1-4, tap Positions and then Reset All .	

Continues on next page

4 Programming and testing

4.5.4 Defining the tool frame

Continued

	Action	Information
6	When all points are defined you can save them to file, which enables you to reuse them later. On the Positions menu, tap Save .	
7	Tap OK . The Calculation Result dialog box will now be displayed, asking you to cancel or to confirm the result before it is written to the controller.	For further information see Is the calculated result good enough? on page 162

How to define elongator points

This procedure describes how to define the orientation of the tool frame by specifying the direction of the z and/or x axis. You need to do this only if you the tool orientation should differ from that of the robot base. The tool coordinate system by default resembles the coordinate system of tool0, as illustrated in [Measuring the tool center point on page 164](#).

	Action
1	Without changing the orientation of the tool, jog the robot so that the reference world point becomes a point on the desired positive axis of the rotated tool coordinate system.
2	Tap Modify Position to define the point.
3	Repeat step 1 and 2 for the second axis if it should be defined.

Is the calculated result good enough?

The **Calculation Result** dialog box displays the calculated result of the tool frame definition. You have to confirm that you accept the result before it can take effect in the controller. The alternative is to redo the frame definition in order to achieve a better result. The result **Mean Error** is the average distance of the approach points from the calculated TCP (tool center point). **Max Error** is the maximum error among all approach points.

It is hard to tell exactly what result is acceptable. It depends on the tool, robot type etc. you are using. Usually a mean error of a few tenths of a millimeter is a good result. If the positioning has been undertaken with reasonable accuracy the result will be okay.

As the robot is used as a measuring machine, the result is also dependent on where in the robot's working area the positioning has been done. Variation of the actual TCP up to a couple of millimeters (for large robots) can be found between definitions in different parts of the working area. The repeatability of any following TCP calibrations will thus increase if these are done close to the preceding ones. Note that the result is the optimal TCP for the robot in that working area, taking into account any discrepancies of the robot in the configuration at hand.



Tip

A common way to check that the tool frame has been correctly defined is to perform a reorientation test when the definition is ready. Select the reorient motion mode and the tool coordinate system and jog the robot. Verify that the tool tip stays very close to the selected reference point as the robot moves.

4.5.5 Editing the tool data

Tool data

Use the value settings to set the tool center point position and physical properties of the tool such as weight and center of gravity.

This can also be done automatically with the service routine LoadIdentify. See sections [Running a service routine on page 196](#), or [LoadIdentify, load identification service routine on page 204](#).



CAUTION

If the tool data is incorrectly defined there is a risk that the speed is higher than expected. This is particularly important in manual mode.

Displaying the tool data

This section details how to display the tool data.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Tool to display the list of available tools.
3	Tap the tool you want to edit, then tap Edit . A menu appears. <ul style="list-style-type: none"> • Change Declaration • Change Value • Delete • Define
4	In the menu, tap Change Value . The data that defines the tool appears. Green text indicates that the value can be changed.
5	Proceed with changing the data as described below.

Continues on next page

4 Programming and testing

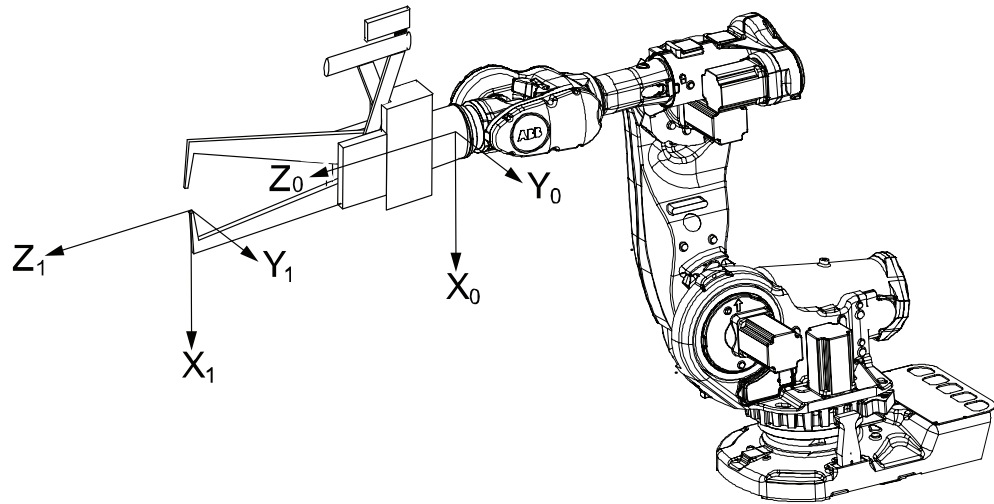
4.5.5 Editing the tool data

Continued

Measuring the tool center point

The easiest way to define the tool center point, TCP, is usually to use the predefined method described in [Defining the tool frame on page 159](#). If you use this method, you do not have to write any values for the frame as these are supplied by the method.

If you already have the measurements of the tool, or for some reason want to measure them manually, the values can be entered in the tool data.



en040000881

X	X axis for tool0
Y	Y axis for tool0
Z	Z axis for tool0
X	X axis for the tool you want to define
Y	Y axis for the tool you want to define
Z	Z axis for the tool you want to define

Action	
1	Measure the distance from the center of the robot's mounting flange to the tool's center point along the X axis of tool0.
2	Measure the distance from the center of the robot's mounting flange to the tool's center point along the Y axis of tool0.
3	Measure the distance from the center of the robot's mounting flange to the tool's center point along the Z axis of tool0.

Editing the tool definition

Action	Instance	Unit
1	Enter the cartesian coordinates of the tool center point's position. tframe.trans.x tframe.trans.y tframe.trans.z	[mm]

Continues on next page

4 Programming and testing

4.5.5 Editing the tool data

Continued

	Action	Instance	Unit
2	If necessary, enter the tool frame orientation.	tframe.rot.q1 tframe.rot.q2 tframe.rot.q3 tframe.rot.q4	None
3	Enter the weight of the tool.	tload.mass	[kg]
4	If necessary, enter the tool's center of gravity.	tload.cog.x tload.cog.y tload.cog.z	[mm]
5	If necessary, enter the orientation of the axis of moment	tload.aom.q1 tload.aom.q2 tload.aom.q3 tload.aom.q4	None
6	If necessary, enter the tool's moment of inertia.	tload.ix tload.iy tload.iz	[kgm ²]
7	Tap OK to use the new values, Cancel to leave the definition unchanged.		

4 Programming and testing

4.5.6 Editing the tool declaration

4.5.6 Editing the tool declaration

Tool declaration

Use the declaration to change how the tool variable can be used in the program's modules.

Displaying the tool declaration

	Action
1	On the ABB menu, tap Jogging .
2	Tap Tool to see the list of available tools.
3	Tap the tool you want to edit, then tap Edit . A menu appears. <ul style="list-style-type: none">• Change Declaration• Change Value• Delete• Define
4	In the menu, tap Change Declaration . The tool's declaration appears.
5	Edit the tool declaration as listed in section Creating a tool on page 156 .



Note

If you change the name of a tool after it is referenced in any program you must also change all occurrences of that tool.

4.5.7 Deleting a tool

Deleting a tool

For more information about deleting a tool, see [Deleting a data instance on page 149](#).

4 Programming and testing

4.5.8 Setup for stationary tools

4.5.8 Setup for stationary tools

Stationary tools

Stationary tools are used, for instance, in applications that involve large machines such as cutters, presses and punch cutters. You may use stationary tools to perform any operation that would be difficult or inconvenient to perform with the tool on the robot.

With stationary tools, the robot holds the work object.

Make a tool stationary

This section describes how to make a tool stationary.

	Action
1	On the ABB menu, tap Jogging .
2	Tap Tool to display the list of available tools.
3	Tap the tool you want to edit, then tap Edit . A menu appears.
4	In the menu, tap Change value . The data that defines the tool appears.
5	Tap the instance <code>robhold</code> .
6	Tap FALSE to make this tool stationary.
7	Tap OK to use the new setup, Cancel to leave the tool unchanged.

Make a work object robot held

This section describes how to make a work object robot held.

	Action
1	In the Jogging window, tap Work object to display the list of available work objects.
2	Tap the work object you want to edit, then tap Edit . A menu appears.
3	In the menu, tap Change value . The data that defines the work object appears.
4	Tap the instance <code>robhold</code> .
5	Tap TRUE to indicate that this work object is held by the robot.
6	Tap OK to use the new setup, Cancel to leave the work object unchanged.

Differences in coordinate system referencing

This section describes differences in coordinate system referencing.

The...	...normally references the...	...but now references the...
work object coordinate system	user coordinate system	user coordinate system (no change)
user coordinate system	world coordinate system	robot's mounting plate
tool coordinate system	robot's mounting plate	world coordinate system

Continues on next page

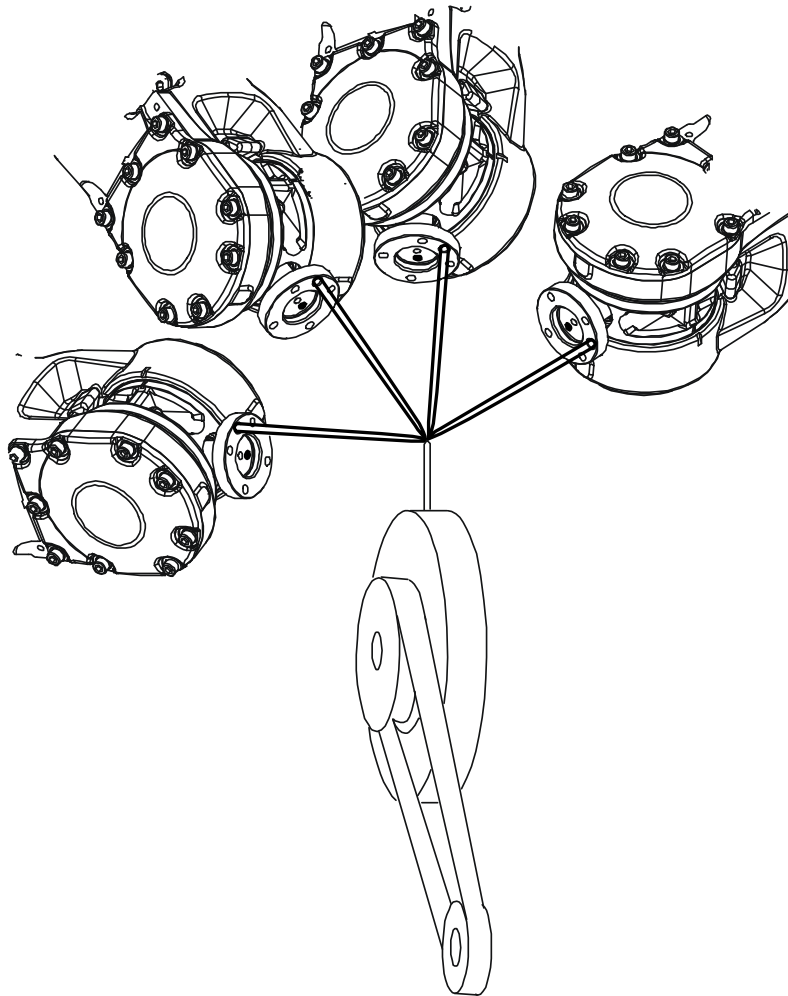
Set up the tool coordinate system

You use the same measurement methods to set up a stationary tool coordinate system as with tools mounted on the robot.

The world reference tip must, in this case, be attached to the robot. Define and use a tool with the reference tip's measurements when you create approach points. You also need to attach elongators to the stationary tool if you need to set up the orientation.

You should enter the reference tip's tool definition manually to minimize errors when calculating the stationary tool's coordinate system.

You may enter the stationary tool's definition manually.



en040000990

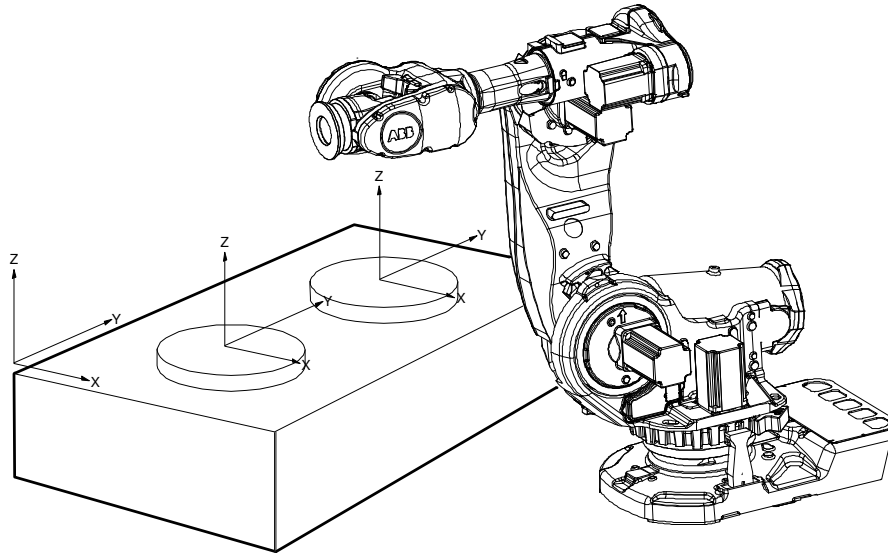
4 Programming and testing

4.6.1 What is a work object?

4.6 Work objects

4.6.1 What is a work object?

Illustration



en0400000819

Description

A work object is a coordinate system with specific properties attached to it. It is mainly used to simplify programming when editing programs due to displacements of specific tasks, objects processes etc.

The work object coordinate system must be defined in two frames, the user frame (related to the world frame) and the object frame (related to the user frame).

Work objects are often created to simplify jogging along the object's surfaces. There might be several different work objects created so you must choose which one to use for jogging.

Payloads are important when working with grippers. In order to position and manipulate an object as accurate as possible its weight must be accounted for. You must choose which one to use for jogging.

4.6.2 Creating a work object

What happens when I create a work object?

A variable of the type `wobjdata` is created. The variable's name will be the name of the work object. For more information on data types, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

This is detailed in section [What is a work object? on page 170](#).

Creating a work object

The work object's coordinate system is now identical with the world coordinate system. To define the position and orientation of the work object's coordinate system, see [Editing the work object declaration on page 177](#).

	Action
1	On the ABB menu, tap Jogging .
2	Tap Work Object to display the list of available work objects.
3	Tap New... to create a new work object.
4	Tap OK .

Work object declaration settings

If you want to change...	then...	Recommendation
the work object's name	tap the ... button next to it	Work objects are automatically named <code>wobj</code> followed by a running number, for example <code>wobj10</code> , <code>wobj27</code> . You should change this to something more descriptive. If you change the name of a work object after it is referenced in any program you must also change all occurrences of that work object.
the scope	select the scope of choice from the menu	Work objects should always be global to be available to all modules in the program.
the storage type	-	Work object variables must always be persistent.
the module	select the module in which this work object should be declared from the menu	

4 Programming and testing

4.6.3 Defining the work object coordinate system

4.6.3 Defining the work object coordinate system

Overview

Defining a work object means that the robot is used to point out the location of it. This is done by defining three positions, two on the x-axis and one on the y-axis. When defining a work object you can use either the user frame or the object frame or both. The user select frame and the object frame usually coincides. If not, the object frame is displaced from the user frame.

How to select method

This procedure describes how to select method for defining either user frame or object frame or both. Note that this only works for a user created work object, not the default work object, wobj0. Defining work object can also be done from the **Program Data** window.

Action	
1	On the ABB menu, tap Jogging
2	Tap Work object to display the list of available work objects.
3	Tap the work object you want to define, then tap Edit .
4	In the menu, tap Define.....
5	Select method from the User method and/or the Object method menu. See How to define the user frame on page 173 and How to define the object frame on page 174

Manual MySystem (RSTEST4) Motors On Stopped (2 of 2) (Speed 100%)

Program Data - wobjdata - Define

Work Object Frame Definition
Work object: wobj1 Active tool: tool0

Select a method for each frame, modify the positions and tap OK.

User No Change Object No Change

Point	Status
-------	--------

Positions Modify Position OK Cancel

Jogging

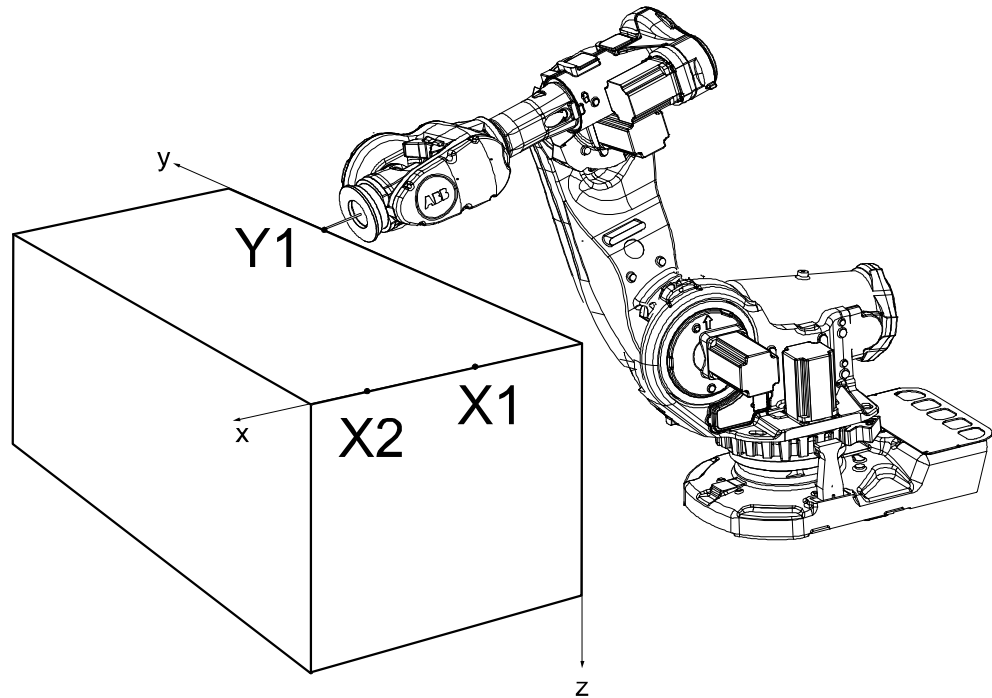
en0400000893

ROB_1

Continues on next page

How to define the user frame

This section details how to define the user frame.



en040000887

The x axis will go through points X1-X2, and the y axis through Y1.

	Action	Information
1	In the User method pop up menu, tap 3 points .	
2	Press the three-position enabling device and jog the robot to the first (X1, X2 or Y1) point that you want to define.	Large distance between X1 and X2 is preferable for a more precise definition.
3	Select the point in the list.	
4	Tap Modify Position to define the point.	
5	Repeat steps 2 to 4 for the remaining points.	

Continues on next page

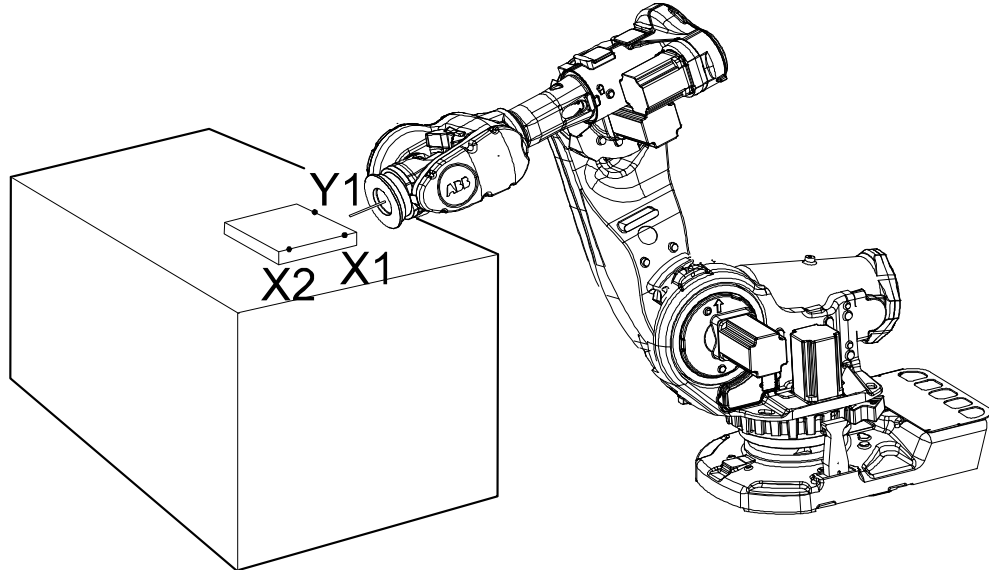
4 Programming and testing

4.6.3 Defining the work object coordinate system

Continued

How to define the object frame

This section describes how to define the object frame if you want to displace it from the user frame.



en0400000899

The x axis will go through points X1-X2, and the y axis through Y1.

	Action
1	In the Object method pop up menu, tap 3 points.
2	See steps 2 to 4 in the description of How to define the user frame on page 173 .

How to save the defined positions

Normally the defined positions are only used as temporary positions by the controller to calculate the position of the work object and are then discarded. However, the positions can also be saved to a program module for later use or analysis.

When saving the positions a new program module is created where the positions are stored with predefined names given by the controller. The names of the positions can be changed afterwards, but when loading the positions it is recommended to use the predefined names.



Note

Only the positions (robtargets) are saved. Make sure to note which tool was used when modifying the defined positions.

	Action
1	When the work object frame definition is completed and all positions have been modified, tap OK .
2	In the Save Modified Points dialog, tap Yes .
3	Tap ABC to change the name of the program module, tap OK to accept the name.

Continues on next page

	Action
4	The names of the positions and the module is displayed in the Save dialog, tap OK .

How to load defined positions

In some cases it is not practical or possible to use the robot to define the positions. Then the positions can be defined or calculated elsewhere and loaded to the **Work Object Frame Definition** dialog.

Positions from any program module can be can be loaded, but is recommended to use the module from the **Save Modified Points** dialog with predefined position names given by the controller.



CAUTION

Make sure that the correct tool and work object is activated in the **Work Object Frame Definition** dialog before loading any positions.

	Action
1	In the Work Object Frame Definition dialog, tap Positions and Load .
2	Tap the module that holds the calibration points, tap OK .
3	If the controller finds all or any predefined positions in the module, the positions are automatically loaded to the correct user or object point. In the Load dialog, tap OK .
4	If some positions are missing or do not have the correct names, the controller cannot load the positions automatically so the user is asked to match the positions manually. Tap each point in the list to assign the positions manually from the drop down list. Tap OK .
5	If necessary, use Modify Position to define any remaining points that could not be loaded.

4 Programming and testing

4.6.4 Editing the work object data

4.6.4 Editing the work object data

Overview

Use the work object data definition to set the position and rotation of the user and object frames.

How to display the work object data

	Action
1	On the ABB menu, tap Jogging .
2	Tap Work object to display the list of available work objects.
3	Tap the work object you want to edit, then tap Edit .
4	Tap Change Value . The data that defines the work object appears.

How to set user and object frame values manually

The easiest way to set the work object and user coordinate systems position is to use the method described in [Defining the work object coordinate system on page 172](#). You can however edit the values manually using the guide below.

Values	Instance	Unit
The cartesian coordinates of the position of the object frame	<code>oframe.trans.x</code> <code>oframe.trans.y</code> <code>oframe.trans.z</code>	mm
The object frame orientation	<code>oframe.rot.q1</code> <code>oframe.rot.q2</code> <code>oframe.rot.q3</code> <code>oframe.rot.q4</code>	-
The cartesian coordinates of the position of the user frame	<code>uframe.trans.x</code> <code>uframe.trans.y</code> <code>uframe.trans.z</code>	mm
The user frame orientation	<code>uframe.rot.q1</code> <code>uframe.rot.q2</code> <code>uframe.rot.q3</code> <code>uframe.rot.q4</code>	-



Note

Editing work object data can also be done from the **Program Data** window.

4.6.5 Editing the work object declaration

Overview

Use the declaration to change how the work object variable can be used in the program's modules.

Displaying the work object declaration

	Action
1	On the ABB menu, tap Jogging .
2	Tap Work object to see the list of available work objects.
3	Tap the work object you want to edit, then tap Edit .
4	In the menu, tap Change Declaration .
5	The work object's declaration appears.
6	Edit the tool declaration as listed in section Creating a work object on page 171 .



Note

If you change the name of a work object after it is referenced in any program you must also change all occurrences of that work object.

4 Programming and testing

4.6.6 Deleting a work object

4.6.6 Deleting a work object

Deleting a work object

For more information about deleting a work object, see [Deleting a data instance on page 149](#).

4.7 Payloads


4.7.1 Creating a payload

What happens when I create a payload?

A variable of the type `loaddata` is created. The variables name will be the name of the payload. For more information on data types, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

Adding a new payload and setting data declaration

The payloads coordinate system will be set to the position, including orientation, of the world coordinate system.

	Action
1	On the ABB menu tap Jogging .
2	Tap Payload or Total Load to display the list of available payloads.  Note Total Load is displayed only when the value of ModalPayloadMode is set to 0 and the mechanical units are TCP robots. See Setting the value for ModalPayloadMode on page 180 .
3	Tap New to create a new payload and enter the data. See Payload declaration settings on page 180 .
4	Tap OK .



WARNING

It is important to always define the actual tool load and, when used, the payload of the robot (for example, a gripped part). Incorrect definitions of load data can result in overloading of the robot mechanical structure. There is also a risk that the speed in manual reduced speed mode can be exceeded.

When incorrect load data is specified, it can often lead to the following consequences:

- The robot may not use its maximum capacity.
- Impaired path accuracy including a risk of overshooting.
- Risk of overloading the mechanical structure.

The controller continuously monitors the load and writes an event log if the load is higher than expected. This event log is saved and logged in the controller memory.

Continues on next page

4 Programming and testing

4.7.1 Creating a payload

Continued

Payload declaration settings

If you want to change...	...then...	Recommendation
the payload's name	tap the ... button next to it	Payloads are automatically named <code>load</code> followed by a running number, for example <code>load10</code> , <code>load31</code> . You should change this to something more descriptive. If you change the name of a payload after it is referenced in any program you must also change all occurrences of that payload's name.
the scope	select the scope of choice from the menu	Payloads should always be global to be available to all modules in the program.
the storage type	-	Payload variables must always be persistent.
the module	select the module in which this payload should be declared from the menu	-

Setting the value for `ModalPayloadMode`

This procedure describes how to modify the value of `ModalPayloadMode`:

- 1 On the **ABB** menu, tap **Control Panel** and then **Configuration**.
- 2 Select **Controller**.
- 3 Select the type **System Misc** and tap.
- 4 Select **ModalPayloadMode** and then tap **Edit**.
- 5 Tap the parameter **Value** twice and set to **0**.
- 6 Click **OK**.
- 7 Tap **Yes** to the question **The changes will not take effect until the controller is warm started. Do you want to restart now?**

4.7.2 Editing the payload data

Overview

Use the payload data to set physical properties of the payload such as weight and center of gravity.

This can also be done automatically with the service routine LoadIdentify. See sections [Running a service routine on page 196](#), or [LoadIdentify, load identification service routine on page 204](#).

Displaying the payload definition

	Action
1	On the ABB menu, tap Jogging .
2	Tap Payload to display the list of available payloads.
3	Tap the payload you want to edit, then tap Edit .
4	Tap Change Value . The data that defines the payload appears.

Changing the payload data

This procedure describes how to manually enter the payload data. This can also be done automatically by running the service routine LoadIdentify. How to run a service routine is described in section [Running a service routine on page 196](#).

	Action	Instance	Unit
1	Enter the weight of the payload.	load.mass	[kg]
2	Enter the payload's center of gravity.	load.cog.x load.cog.y load.cog.z	[mm]
3	Enter the orientation of the axis of moment.	load.aom.q1 load.aom.q2 load.aom.q3 load.aom.q3	
4	Enter the payload's moment of inertia.	ix iy iz	[kgm ²]
5	Tap OK to use the new values, Cancel to leave the data unchanged.	-	-

Using the `PayLoadsInWristCoords` parameter

By using the `PayLoadsInWristCoords` parameter, the loaddata for payloads can be specified relative to the wrist instead of the active TCP or work object. This can be useful if several tool or TCP or work objects (when tool is stationary) are used for one payload. In this case only one load identification is needed instead of one for each tool or TCP or work object. Thus it is possible to use the same payload loaddata for any robhold or stationary tool being active. This saves the time (for example, during commissioning).

Continues on next page

4 Programming and testing

4.7.2 Editing the payload data

Continued

For more information about `PayLoadsInWristCoords`, see *Technical reference manual - System parameters* and *Technical reference manual - RAPID Instructions, Functions and Data types*.

4.7.3 Editing the payload declaration

Overview

Use the declaration to change how the payload variable can be used in the program's modules.

Displaying the payload declaration

	Action
1	On the ABB menu, tap Jogging .
2	Tap Payload to see the list of available payloads.
3	Tap the payload you want to edit, then tap Edit .
4	In the menu, tap Change declaration .
5	The payload's declaration appears. See Creating a payload on page 179 .



Note

If you change the name of a payload after it is referenced in any program you must also change all occurrences of that payload's name.

4 Programming and testing

4.7.4 Deleting a payload

4.7.4 Deleting a payload

Deleting a payload

For more information about deleting a payload, see [Deleting a data instance on page 149](#).

4.8 Testing

4.8.1 About the automatic mode

What is the automatic mode?

In automatic mode the safety function of the three-position enabling device is bypassed so that the manipulator is allowed to move without human intervention.



WARNING

Prior to selecting automatic mode, any suspended safeguards shall be returned to their full functionality.

Tasks normally performed in the automatic mode

The following tasks are normally performed in automatic mode:

- Starting and stopping processes.
- Loading, starting, and stopping RAPID programs.
- Returning the manipulator to its path when returning to operation after an emergency stop.
- Backing up the system.
- Restoring backups.
- Cleaning tools.
- Preparing or replacing work objects.
- Performing other process oriented tasks.

Limitations in automatic mode

Jogging is not possible in automatic mode. There may be other specific tasks that should not be performed in automatic mode.

Consult your plant or system documentation to find out which specific tasks should not be performed in automatic mode.

Active safeguard mechanisms

For IRC5, both the general mode stop (GS) mechanisms, the automatic mode stop (AS) mechanisms, and the superior stop (SS) are all active while operating in automatic mode.

Coping with process disturbances

Process disturbances may not only affect a specific manipulator cell but an entire chain of systems even if the problem originates in a specific cell.

Extra care must be taken during such a disturbance since that chain of events may create hazardous operations not seen when operating the single manipulator cell.

All remedial actions must be performed by personnel with good knowledge of the entire production line, not only the malfunctioning manipulator.

Continues on next page

4 Programming and testing

4.8.1 About the automatic mode

Continued

Process disturbance examples

A manipulator picking components from a conveyer might be taken out of production due to a mechanical malfunction, while the conveyer must remain running in order to continue production in the rest of the production line. This means, of course, that extra care must be taken by the personnel preparing the manipulator in close proximity to the running conveyor.

A welding manipulator needs maintenance. Taking the welding manipulator out of production also means that a work bench as well as a material handling manipulator must be taken out of production to avoid personnel hazards.

4.8.2 About the manual mode

What is the manual mode?

The manual mode is used when programming and for program verification.

In manual mode the manipulator movement is under manual control. The three-position enabling device must be pressed to activate the motors of the manipulator, that is, enabling movement.

There are two manual modes:

- Manual reduced speed mode, usually called manual mode.
- Manual full speed mode (not available in USA or Canada).

What is the manual reduced speed mode?

In manual reduced speed mode the movement of the TCP is limited to 250 mm/s. In addition, there is a limitation on the maximum allowed speed for each axis. These axis limitations are robot dependent and cannot be changed.

The three-position enabling device must be pressed to center-position to activate the motors of the manipulator.



WARNING

Wherever possible, the manual mode of operation shall be performed with all persons outside the safeguarded space.

What is the manual full speed mode?

The manual full speed mode is used for program verification only.

In manual full speed mode, the manipulator can move in programmed speed but only under manual control.

In manual full speed mode the initial speed limit is up to, but not exceeding, 250 mm/s. This is achieved by limiting the speed to 3% of the programmed speed. Through manual control the speed can be increased up to 100%.

The three-position enabling device must be pressed to center-position and the hold-to-run button pressed to enable program execution.



WARNING

Wherever possible, the manual mode of operation shall be performed with all persons outside the safeguarded space.

Continues on next page

4 Programming and testing

4.8.2 About the manual mode

Continued

Note that the manual full speed mode is optional and therefore not available in all robots.



Note

As per the updated standard, ISO 10218-1:2011 *Robots and robotic devices – Safety requirements for industrial robots – Part 1 Robots*, the following adaptations are made to the manual full speed mode.

- Resetting the speed to 250 mm/s every time the three-position enabling device is re-initiated by placing the switch in the center-enabled position after either having been released or fully compressed.
- Editing RAPID programs and jogging the manipulator are disabled.

Bypassed safeguard mechanisms

Automatic mode safeguarded stop (AS) mechanisms are bypassed while operating in manual mode.

The three-position enabling device

In **manual mode**, the motors of the manipulator are activated by the three-position enabling device on the FlexPendant. This way, the manipulator can only move as long as the device is pressed.

The three-position enabling device is designed so that its push-button must be pressed just half-way to activate the motors of the manipulator. The manipulator cannot be operated in the fully-in and fully-out positions.

The hold-to-run function

The hold-to-run function allows stepping or running a program in manual full speed mode. For safety reasons, it is necessary to keep pressing both the three-position enabling device and the **Start** button.

Note that jogging does not require the hold-to-run function, regardless of operating mode. The hold-to-run function can also be activated for manual reduced speed mode.

Tasks normally performed in manual reduced speed mode

The following tasks are normally performed in manual reduced speed mode.

- Jogging the manipulator back on its path when returning to operation after an emergency stop
- Correcting the value of I/O signals after error conditions
- Creating and editing RAPID programs
- Starting, stepping, and stopping program execution, for example while testing a program
- Tuning programmed positions

Continues on next page

Tasks normally performed in manual full speed mode

As per the standard, ISO 10218-1:2011, the following tasks can be performed in the manual full speed mode.

- Starting and stopping program execution for final program verification
- Stepping program execution
- Setting speed (0–100%)
- Setting program pointer (to Main, to routine, to cursor, to service routine, etc.)

The following tasks cannot be performed in the manual full speed mode:

- Changing system parameter values
- Editing system data

4 Programming and testing

4.8.3 Using the hold-to-run function

4.8.3 Using the hold-to-run function

When to use the hold-to-run function

The hold-to-run function is used to run or step programs in manual full speed mode, in combination with the three-position enabling device.

In order to run a program in manual full speed mode it is necessary, for safety reasons, to keep pressing both the three-position enabling device and the hold-to-run button. This hold-to-run function also applies when stepping through a program in manual full speed mode.

Some versions of the FlexPendant have a separate hold-to-run button (thumb button). If there is no separate hold-to-run button, the program execution buttons are used as hold-to-run buttons. When **Start**, **Forward**, and **Backward** buttons are used like this (press and hold) they are referred to as hold-to-run buttons.

Operating mode	Function
Manual reduced speed mode	Normally, hold-to-run has no effect in the manual reduced speed mode. However, it is possible to activate for manual reduced speed mode by changing a system parameter.
Manual full speed mode	Pressing hold-to-run AND pressing the three-position enabling device enables running a program. It may be run continuously or step-by-step. Releasing hold-to-run in this mode immediately stops manipulator movement as well as program execution. When pressing it again, execution is resumed from that position.
Automatic mode	Hold-to-run is not used in automatic mode.



Note

If the FlexPendant has a separate hold-to-run button, pressing both that button and the **Start** button will stop the program execution.

Using the hold-to-run function

This instruction details how use the hold-to-run function in manual full speed mode.

	Action
1	Press the three-position enabling device on the FlexPendant.
2	Choose execution mode by pressing and holding either: <ul style="list-style-type: none">• Start (continuous program execution) or thumb button• Forward (step-by step program execution forwards)• Backward (step-by step program execution backwards)
3	If Start or the thumb button was pressed, then the program execution continues as long as the button is pressed. If Forward or Backward was pressed, the program is executed step-by-step by alternately releasing and pressing the Forward/Backward button. Note that the button must be pressed and held until the instruction is executed. If the button is released, program execution will stop immediately!
4	If the three-position enabling device is released, intentionally or by accident, the complete procedure must be repeated to enable running.

4.8.4 Running the program from a specific instruction

Overview


When starting a program the execution starts from the program pointer. To start from another instruction, move the program pointer to the cursor.



WARNING

When execution is started the robot will move to the first programmed position in the program. Make sure that the robot does not risk running into any obstacles.

Running the program from a specific instruction

	Action
1	On the ABB menu, tap Program Editor .
2	Tap on the program step where you want to start, then tap Debug and then PP to Cursor .
3	 DANGER Make sure that no personnel are in the safeguarded space.
4	Press the Start button on the FlexPendant.

4 Programming and testing

4.8.5 Running a specific routine

4.8.5 Running a specific routine

Overview

When starting a program the execution starts from the program pointer. To start from another routine, move the program pointer to the routine.

Prerequisites

In order to run a specific routine the module with the routine must be loaded and the controller must be in manual stopped mode.

Running a specific routine

This procedure describes how to run a specific routine by moving the program pointer.

	Action
1	On the ABB menu, tap Program Editor .
2	Tap Debug and then PP to Routine to place the program pointer at the start of the routine.
3	Press the Start button on the FlexPendant.

Related information

How to run a service routine is described in [Running a service routine on page 196](#). The same method can be used to run a specific routine in the task scope. See [Running a service routine on page 196](#) for detailed information.

4.8.6 Stepping instruction by instruction

Overview

In all operating modes the program may be executed step by step forwards or backwards.

Stepping backwards is limited, see *Technical reference manual - RAPID Overview* for more details.

Select step mode

This section details how to select step mode. Stepping can be done in three ways; step in, step over, and motion step.

	Action	Information
1	Select step mode using the Quickset menu.	Described in Quickset menu, Step Mode on page 65 .

Stepping

This section details how to step forwards and backwards.

If you want to step...	then press...
forward	Forward button on FlexPendant
backward	Backward button on FlexPendant

Limitations of backward execution

There are some restrictions for the backward execution:

- When stepping backwards through a `MoveC` instruction, the execution does not stop in the circular point.
- It is not possible to step backwards out of a `IF`, `FOR`, `WHILE` and `TEST` statement.
- It is not possible to step backwards out of a routine when reaching the beginning of the routine.
- There are instructions affecting the motion that cannot be executed backwards (e.g. `ActUnit`, `ConfL` and `PDispOn`). If attempting to execute these backwards, an alert box will inform you that this is not possible.

Backward execution behavior

When stepping forward through the program code, a program pointer indicates the next instruction to execute and a motion pointer indicates the move instruction that the robot is performing.

When stepping backward through the program code, the program pointer indicates the instruction above the motion pointer. When the program pointer indicates one move instruction and the motion pointer indicates another, the next backward movement will move to the target indicated by the program pointer, using the type of movement and speed indicated by the motion pointer.

Continues on next page

4 Programming and testing

4.8.6 Stepping instruction by instruction

Continued

Example of backward execution

The following example illustrates the behavior when stepping backwards through move instructions. The program pointer and motion pointer helps you keep track of where the RAPID execution is and where the robot is.

MoveL, MoveJ, and MoveC are move instructions in RAPID, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

The screenshot shows the RAPID editor interface. At the top, there is a status bar with a menu icon, a manual icon, and text: "Manual RW6.05 (10.140.60.195)", "Guard Stop", and "Stopped (Speed 100%)". Below this is the program name: "NewProgramName in T_ROB1/MainModule/main". The main area displays the RAPID code:

```

5  CONST robtarget p40:=[[119.73,-680.44,575.86],[0.
6  CONST robtarget p50:=[[119.73,-637.95,575.87],[0.
7  CONST robtarget p60:=[[97.43,-637.95,575.87],[0.5
8  CONST robtarget p80:=[[146.54,-582.14,628.12],[0.
9  CONST robtarget p70:=[[146.54,-582.14,628.12],[0.
10 PROC main()
11   MoveJ p10, v1000, z50, tool0;
12   MoveL p20, v1000, z50, tool0;
13   MoveC p30, p40, v300, z10, tool0;
14   MoveL p50, v1000, z50, tool0;
15   MoveL p60, v1000, z50, tool0;
16   ENDPROC
17 ENDMODULE
  
```

Annotations in the image: (A) points to line 15 (MoveL p60), (B) points to line 13 (MoveC p30), and (C) points to line 12 (MoveL p20). The target 'p50' in line 14 is highlighted in blue. The interface includes a toolbar at the bottom with buttons: "Add Instruction", "Edit", "Debug", "Modify Position", "Hide Declarations", "Jogging", "Production Window", "T_ROB1 MainModule", "Program Data", and "ROB_1".

en0400001204

A	Program pointer
B	Motion pointer
C	Highlighting of the robtarget that the robot is moving towards, or already has reached.

When...	then...
stepping forward until the robot is in p50	the motion pointer will indicate p50 and the program pointer will indicate the next move instruction (MoveL p60).
pressing the Backward button once	the robot will not move but the program pointer will move to the previous instruction (MoveC p30, p40). This indicates that this is the instruction that will be executed the next time Backward is pressed.
pressing the Backward button again	the robot will move to p40 linearly with the speed v300. The target for this movement (p40) is taken from the MoveC instruction. The type of movement (linear) and the speed are taken from the instruction below (MoveL p50). The motion pointer will indicate p40 and the program pointer will move up to MoveL p20.

Continues on next page

When...	then...
pressing the Backward button again	the robot will move circularly, via p30, to p20 with the speed v1000. The target p20 is taken from the instruction <code>MoveL p20</code> . The type of movement (circular), the circular point (p30) and the speed are taken from the <code>MoveC</code> instruction. The motion pointer will indicate p40 and the program pointer will move up to <code>MoveL p10</code> .
pressing the Backward button again	the robot will move linearly to p10 with the speed v1000. The motion pointer will indicate p10 and the program pointer will move up to <code>MoveJ p10</code> .
pressing the Forward button once	the robot will not move but the program pointer will move to the next instruction (<code>MoveL p20</code>).
pressing the Forward button again	the robot will move to p20 with the speed v1000.

4 Programming and testing

4.9.1 Running a service routine

4.9 Service routines

4.9.1 Running a service routine

Service routines

Service routines perform a number of common services. The service routines available to you depends on your system setup and available options.

The service routines described in this manual are some of the commonly available.

Prerequisites

Service routines can only be started in manual reduced speed mode or in manual full speed mode.

The program must be stopped and there has to be a program pointer.

It is not possible to call a routine when in synchronized mode.

If the service routine contains parts that must be carried out in automatic mode, then the program pointer must not be moved manually before starting the service routine. The program pointer should be where the program flow was stopped.



WARNING

If a service routine is started in the middle of a stopped movement instruction (that is, before the end position is reached), then the movement will be resumed when the execution of the service routine starts.



CAUTION

Note that once a service routine has started, aborting it might not resume the system to its previous state, as the routine may have moved the robot arm.

Running a service routine

This section describes how to execute a service routine or another routine in the task scope using **Call Routine**.

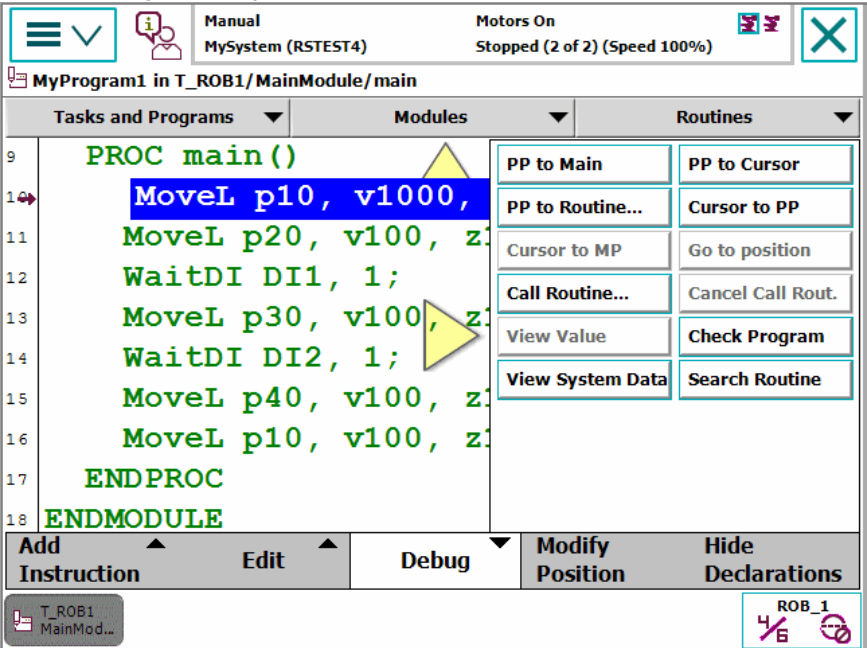
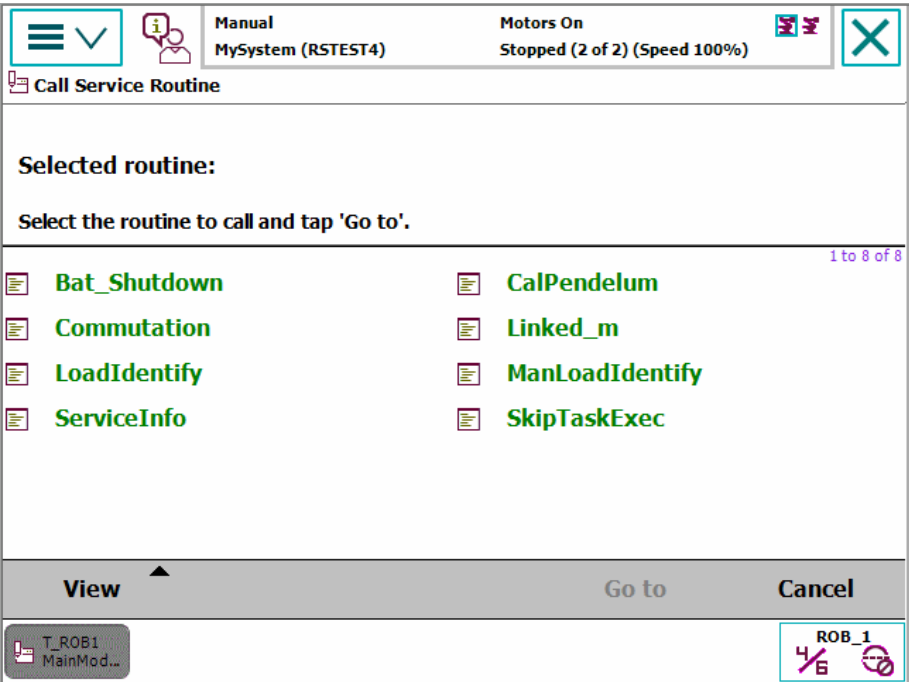
	Action
1	On the ABB menu tap Program Editor .

Continues on next page

4 Programming and testing

4.9.1 Running a service routine

Continued

Action	
2	<p data-bbox="504 315 925 342">On the Debug menu tap Call Routine.</p>  <p data-bbox="504 999 612 1021">en040000884</p>
3	<p data-bbox="504 1059 1442 1144">The Call Service Routine dialog lists all predefined service routines. The same dialog can however be used to run any routine in the task scope. Select All Routines on the View menu to see all available routines.</p>  <p data-bbox="504 1834 612 1856">en040000885</p>
4	<p data-bbox="504 1895 957 1921">Tap a service routine and then tap Go to.</p> <p data-bbox="504 1928 1442 1980">The Program Editor will be displayed with the program pointer moved to the beginning of the selected routine.</p>

Continues on next page

4 Programming and testing

4.9.1 Running a service routine

Continued

Action	
5	Press the Start button on the FlexPendant and follow the instructions displayed on the FlexPendant. After execution of the routine the task is stopped and the program pointer is returned to where it was before the service routine started.



CAUTION

Press **Cancel Call Rout** if you need to interrupt the routine before it has finished executing. Before resuming normal program flow, however, you must see to it that the robot is correctly positioned. If the interrupted routine has moved it, you will need to take actions to return the robot to its position. See [Returning the robot to the path on page 240](#) for further information.



WARNING

Do not execute a service routine in the middle of a move or a weld.

If you execute a service routine in the middle of a movement, the unfinished movements will be completed before the called routine is executed. This can result in an unwanted movement.

If possible, step and complete the interrupted movement before the service routine is called. Otherwise save the current movement by adding `StorePath` and `RestoPath` in the service routine. The movement will then be completed after the service routine has ended and the program starts again.

However, it is not possible to save more than one interrupted movement each time as wanted, if the service routine would be called from an error handler with `StorePath` and `RestoPath`.

Limitations

Besides service routines, **Call Routine** applies to all routines with the following criteria:

- Must be a procedure with empty parameter list. This means not a function and not a trap routine.
- Must be in the task scope, not local. If the procedure is local in a module the scope is restricted to that module, and the procedure is not visible from the task level.
- Must be in a loaded module, not installed. (Check the system parameter **Installed** in the type *Automatic Loading of Modules* in the *Controller* topic.)

Related information

[Battery shutdown service routine on page 200.](#)

[LoadIdentify, load identification service routine on page 204.](#)

[Service Information System, ServiceInfo service routine on page 203.](#)

[Calibration Pendulum, CalPendulum service routine on page 202.](#)

Continues on next page

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

4 Programming and testing

4.9.2 Battery shutdown service routine

4.9.2 Battery shutdown service routine

When to use this service routine

For SMB units with 2-pole battery contact, it is possible to shut down the battery backup of the serial measurement board to save battery power during transportation or storage. This is the *BatteryShutDown* service routine.

For SMB units with 3-pole contact, this function shall not be used since the power consumption is so low that it is not needed.



Tip

The service routine was earlier called *Bat_Shutdown*.

BatteryShutDown

When the system is powered on again, the function is reset. The revolution counters will be lost and need an update but the calibration values will remain.

The consumption in ordinary shutdowns is then approximately 1 mA. When using sleep mode the consumption is reduced to 0.3 mA. When the battery is nearly discharged, with less than 3 Ah left, an alert is given on the FlexPendant and the battery should be replaced.



Tip

Before starting the service routine *BatteryShutDown*, run the robot to its calibration position. This will make it easier to recover after the sleep mode.

Related information

How to start a service routine is described in [Running a service routine on page 196](#).

How to update the revolution counters is described in [Updating revolution counters on page 282](#).

4.9.3 Axis Calibration service routine

Description of the service routine

Axis Calibration is a standard calibration method for calibration of ABB robots and is the most accurate method for the standard calibration.

The following routines are available for the Axis Calibration method:

- Fine calibration
- Update revolution counters
- Reference calibration

The calibration equipment for Axis Calibration is delivered as a toolkit.

The actual instructions of how to perform the calibration procedure and what to do at each step is given on the FlexPendant. You will be guided through the calibration procedure, step by step.

Related information

[Running a service routine on page 196.](#)

A more detailed description of the calibration method is given in the product manual for the respective manipulator.

4 Programming and testing

4.9.4 Calibration Pendulum, CalPendulum service routine

4.9.4 Calibration Pendulum, CalPendulum service routine

Description of the service routine

Calibration Pendulum is a standard calibration method for calibration of ABB robots (except some models).

Two different routines are available for the Calibration Pendulum method:

- Calibration Pendulum II
- Reference calibration

The calibration equipment for Calibration Pendulum is delivered as a complete toolkit, including the *Operating manual - Calibration Pendulum*, which describes the method and the different routines further.

Related information

[Running a service routine on page 196.](#)

Calibration Pendulum is described in full in the manual *Operating manual - Calibration Pendulum*. Specific information for each robot is described in the robot's product manual.

4.9.5 Service Information System, ServiceInfo service routine

When to use this service routine

ServiceInfo is a service routine based on Service Information System, SIS, a software function which simplifies maintenance of the robot system. It supervises the operating time and mode of the robot, and alerts the operator when a maintenance activity is scheduled.

ServiceInfo

Maintenance is scheduled by setting the system parameters of the type *SIS Parameters*. All system parameters are described in *Technical reference manual - System parameters*.

More details about SIS is described in *Operating manual - Service Information System*.

Supervised functions

The following counters are available:

- Calendar time counter
- Operation time counter
- Gearbox operation time counters
- Moved distance counter¹

¹ The moved distance value cannot be reset through the service routines.

Counters are reset when maintenance has been performed.



CAUTION

Resetting counters cannot be undone.

The counter status is displayed after running the ServiceInfo routine for maintenance. Status **OK** indicates that no service interval limit has been exceeded by that counter. Status **NOK** indicates that service interval limit has been exceeded by that counter.

Related information

[Running a service routine on page 196.](#)

4 Programming and testing

4.9.6 LoadIdentify, load identification service routine

4.9.6 LoadIdentify, load identification service routine

When to use this service routine

The service routine LoadIdentify is used to automatically identify the data of loads mounted on the robot. The data can also be entered manually, but this requires information that may be difficult to calculate.

To run LoadIdentify, there are a number of things to consider. These are described on the following pages. There is also information on error handling and limitations described in this chapter.



WARNING

It is important to always define the actual tool load and, when used, the payload of the robot (for example, a gripped part). Incorrect definitions of load data can result in overloading of the robot mechanical structure. There is also a risk that the speed in manual reduced speed mode can be exceeded.

When incorrect load data is specified, it can often lead to the following consequences:

- The robot may not use its maximum capacity.
- Impaired path accuracy including a risk of overshooting.
- Risk of overloading the mechanical structure.

The controller continuously monitors the load and writes an event log if the load is higher than expected. This event log is saved and logged in the controller memory.



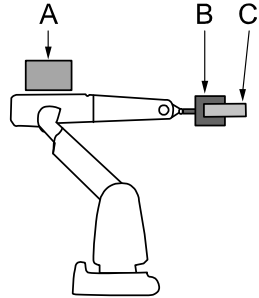
Note

When using LoadId or LoadIdentification to identify the load of a tool or payload with unknown mass, the mass is estimated using the manipulator and the result can deviate from the actual mass. This is due to tolerances and variations between mechanical units. This does not necessarily mean that the identified payload or tool will cause issues with motion performance. If a very accurate value for the mass is required, it is recommended to weigh the tool or payload and use known mass in the identification.

Continues on next page

LoadIdentify

LoadIdentify can identify the tool load and the payload. The data that can be identified are mass, center of gravity, and moments of inertia.



en0500001535

A	Upper arm load
B	Tool load
C	Payload

Before running the load identification for the payload, make sure that the tool load data is correctly defined first, for example by running LoadIdentify for the tool.

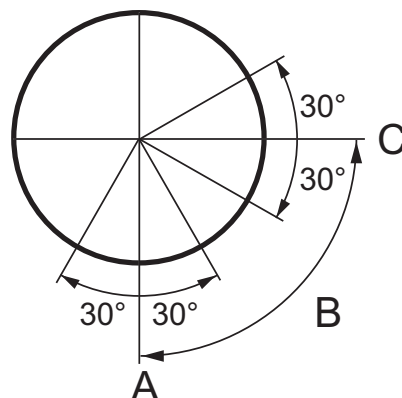
To identify the mass of B and C, axis 3 has to perform some movement. This means that to identify the mass, the upper arm load A must be known and correctly defined first.

To improve accuracy if the upper arm load A is mounted, input the known mass of B and C and choose the *known mass* method when identifying.

Configuration angles

To perform the identification the robot moves the load after a specific pattern and calculates the data. The axes that move are 3, 5 and 6. At the identification position, the motion for axis 3 is approximately ± 3 degrees and for axis 5 it is approximately ± 30 degrees. For axis 6 the motion is performed around two configuration points.

The optimum value for the configuration angle is either +90 degrees or -90 degrees.



en0500001537

A	Configuration 1 (start position)
B	Configuration angle

Continues on next page

4 Programming and testing

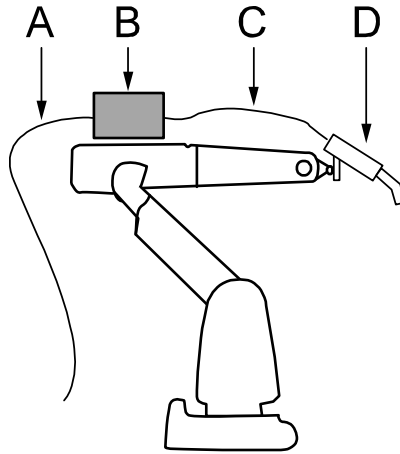
4.9.6 LoadIdentify, load identification service routine

Continued

C	Configuration 2
---	-----------------

LoadIdentify with arm loads mounted

The best way to perform load identification is to use a robot with no arm loads mounted. If this is not possible, good accuracy can still be achieved. Consider, for example, the robot in the figure below, which has arc welding equipment mounted on it.



en0500001536

A	Cable 1
B	Load 1
C	Cable 2
D	Load 2

If we want to use load identification to find the data of load 2, the most important thing to remember is to make sure that the upper arm load is correctly defined, in particular its mass and center of gravity along the robot arm. The arm load includes everything that is mounted on the robot, except tool load and payload. In the figure above, cable 1, cable 2, and load 1 are included in the arm load, the total weight and center of gravity have to be calculated.

When performing the load identification, cable 2 should be disconnected since it will otherwise put an extra force on load 2. When identifying load 2 with such a force present, the result may differ considerably from the correct load. Ideally, cable 2 should be disconnected from load 2 and fastened on the upper arm. If this is not possible, the cable can also be disconnected at load 1 and fastened to the upper arm in such a way that the resulting force on load 2 is minimized.

Prerequisites for tool loads

Before running the LoadIdentify service routine for a tool load, make sure that:

- The tool is selected in the jogging menu.
- The tool is correctly mounted.
- Axis 6 is close to horizontal.
- The upper arm load is defined, if the tool mass is to be identified.
- The axes 3, 5, and 6 are not close to their corresponding working range limits.

Continues on next page

- The speed is set to 100%.
- The system is in manual mode.

Note that LoadIdentify cannot be used for `tool0`.

Prerequisites for payloads

Before running the LoadIdentify service routine for a payload, make sure that:

- The tool and payload are correctly mounted.
- Axis 6 is close to horizontal.
- The tool load is known (run LoadIdentify for the tool first).
- The upper arm load is defined, if the payload mass is to be identified.
- When using a moving TCP, the tool must be calibrated (TCP).
- When using a stationary TCP, the corresponding work object must be calibrated (user frame and object frame).
- The axes 3, 5, and 6 are not close to their corresponding working range limits.
- The speed is set to 100%.
- The system is in manual mode.

Note that LoadIdentify cannot be used for `load0`.

Running LoadIdentify

To start the load identification service routine you must have an active program in manual mode and the tool and payload that you want to identify must be defined and active in the **Jogging** window.



Tip

Always run load identification with cold motors (no warm-up).


	Action	Information
1	Start LoadIdentify from the Program Editor . Press the three-position enabling device and then press the Start button on the FlexPendant.	How to start service routines is described in Running a service routine on page 196 .
2	Tap OK to confirm that current path will be cleared and that the program pointer will be lost.	Tap Cancel and then Cancel Call Rout to quit the service routine without losing the program pointer.
3	Tap Tool or Payload .	
4	Tap OK to confirm that the correct tool and/or payload is active in the jogging menu and that the tool load/payload is correctly mounted.	If it is not correct, release the three-position enabling device and select the correct tool/payload in the Jogging menu. Then return to LoadIdentify, press the three-position enabling device, and press Start . Tap Retry and confirm that the new tool/payload is correct.

Continues on next page

4 Programming and testing

4.9.6 LoadIdentify, load identification service routine

Continued

	Action	Information
5	When identifying tool loads, confirm that the tool is active. When identifying payloads, confirm that the payload's tool is active and calibrated.	See step 4 .
6	When identifying payloads with stationary TCP, confirm that the correct work object is active and (preferably) calibrated. If it is correct, tap OK to confirm.	See step 4 .
7	Select identification method. If you select the method where the mass is assumed to be known, remember that the tool/payload that you use must have the correct mass defined. Tap OK to confirm.	
8	Select configuration angle. The optimum is +90 or -90 degrees. If this is impossible, tap Other and set the angle. The minimum is +30 or -30 degrees.	
9	If the robot is not in a correct position for load identification, you will be asked to jog one or more axes roughly to a specified position. When you have done this tap OK to confirm. If the robot is still not in a correct position for load identification, the robot will slowly move to the correct position. Press Move to start the movement.	Axis 1 to 3 must not be more than 10 degrees from proposed position.
10	The robot can go through the load identification movements slowly before performing the load identification (slow test). Tap Yes if you want a slow test and No to proceed to the identification.	This is useful for ensuring that the robot will not hit anything during the identification. However, this will take a lot longer time.  Note If the load identification is planned to be run in manual full speed, then the slow test is required before the actual measurement can start.
11	The setup for load identification is now complete. To start the motion, switch to Automatic mode and Motors On. Then tap Move to start the load identification movements.	
12	When the identification is finished, switch back to manual mode, press the three-position enabling device and the Start button. Tap OK to confirm.	
13	The result of the load identification is now presented on the FlexPendant. For robots that support the <i>Load diagram check</i> functionality, there is a message if the load is approved or not, and an Analyze button to view more information.	See Load diagram check on page 209 .
14	Tap Yes to update the selected tool or payload with the identified parameters. Tap No to exit LoadIdentify without saving the parameters.	

Continues on next page

Load diagram check

For robots that support the *Load diagram check* functionality, the combination of the arm load, tool and payload, will be assessed against the load diagram. The ratings of total handling weight as well as of the center of gravity distance to the load diagram in Z and L directions will be provided both for wrist-up and wrist-down configurations.

There is a message if the load is approved or not, and an **Analyze** button to view more information.

- Load approved
- Load not approved
- Load approved only with wrist down

Running LoadIdentify with ModalPayloadMode deactivated

When the system parameter ModalPayloadMode is deactivated, set to 0, LoadIdentify will identify the tool load and the total load. It is no longer possible to define the payload.

With ModalPayloadMode deactivated it is possible to use the \TLoad argument in movement instructions. The \TLoad argument describes the total load used in the movement. The total load is the tool load together with the payload that the tool is carrying. If the \TLoad argument is used, then the loaddata in the current tooldata is not considered. For more information about ModalPayloadMode in movement instructions, see the section MoveL in *Technical reference manual - RAPID Instructions, Functions and Data types*.

To start the load identification service routine you must have an active program in manual mode and the tool and payload that you want to identify must be defined and active in the Jogging window.



Tip

Always run load identification with cold motors (no warm-up).


	Action	Information
1	Start LoadIdentify from the Program Editor . Press the three-position enabling device and then press the Start button on the FlexPendant.	How to start service routines is described in Running a service routine on page 196 .
2	Tap OK to confirm that current path will be cleared and that the program pointer will be lost.	Tap Cancel and then Cancel Call Rout to quit the service routine without losing the program pointer.
3	Tap OK to continue with the LoadIdentify process.	The selection to update the tool load or the total load is done at a later stage.

Continues on next page

4 Programming and testing

4.9.6 LoadIdentify, load identification service routine

Continued

	Action	Information
4	Tap OK to confirm that the correct tool and/or total load is active in the jogging menu and that the tool load/total load is correctly mounted.	If it is not correct, release the three-position enabling device and select the correct tool/payload in the Jogging menu. Then return to LoadIdentify , press the three-position enabling device, and press Start . Tap Retry and confirm that the new tool/payload is correct.
5	When identifying tool loads, confirm that the tool is active.	See step 4 .
6	Select identification method. If you select the method where the mass is assumed to be known, remember that the tool/total load that you use must have the correct mass defined. Tap OK to confirm.	
7	Select configuration angle. The optimum is +90 or -90 degrees. If this is impossible, tap Other and set the angle. The minimum is +30 or -30 degrees.	
8	If the robot is not in correct position for load identification, you are asked to jog one or more axes roughly to a specified position. When you have done this tap OK to confirm. If the robot is still not in a correct position for load identification, the robot will slowly move to the correct position. Press Move to start the movement.	Axis 1 to 3 must not be more than 10 degrees from proposed position.
9	The robot can go through the load identification movements slowly before performing the load identification (slow test). Tap Yes if you want a slow test and No to proceed to the identification.	This is useful for ensuring that the robot will not hit anything during the identification. However, this will take a lot longer time.  Note If the load identification is planned to be run in manual full speed, then the slow test is required before the actual measurement can start.
10	The setup for load identification is now complete. To start the motion, switch to Automatic mode and Motors On. Then tap Move to start the load identification movements.	
11	When the identification is finished, switch back to manual mode, press the three-position enabling device and the Start button. Tap OK to confirm.	
12	The result of the load identification is now presented on the FlexPendant. Tap Tool if you want to update the selected tool, tap Loaddata if you want to update the total load, or tap No if you want to quit without saving.	
13	If Loaddata is selected it is possible to update the total load to either an existing or a new <code>loaddata</code> persistent variable.	

Continues on next page

Error handling

If the three-position enabling device is released during the load identification (before the movements start), the routine can always be restarted by pressing the three-position enabling device again and then pressing the Start button.

If an error should occur during the load identification movements, the routine must be restarted from the beginning. This is done automatically by pressing Start after confirming the error. To interrupt and leave the load identification procedure, tap **Cancel Call Routine** in **Program Editor** debug menu.

Limitations for LoadIdentify

Only tool loads and payloads can be identified with LoadIdentify. Thus arm loads cannot be identified.

If the load identification movements are interrupted by any kind of stop (program stop, emergency stop, etc.), the load identification must be restarted from the beginning. Confirm the error and press **Start** to automatically restart.

If the robot is stopped on a path with program stop and load identification is performed at the stop point, the path will be cleared. This means that no regain movement will be performed to return the robot back to the path.

The load identification ends with an EXIT instruction. That means that the program pointer is lost and must be set to main before starting any program execution.



Tip

The tool and/or payload data can be set manually if the load is small (10% or less of the maximum load) or symmetrical, for example if the tool load is symmetrical around axis 6.



Tip

If the mass of the tool or payload is unknown, the service routine LoadIdentify can in some cases identify a 0 kg mass. If the load is very small compared to the maximum load for the robot, then a 0 kg mass can be ok. Otherwise, try the following to identify the mass.

- Check that the arm loads are correctly defined and redo the identification.
- Find the weight of the load in some other way and perform a load identification with known mass to remove the dependency on arm loads.

LoadIdentify for 4-axis palletizing and SCARA robots

When running LoadIdentify on a robot with 4 instead of 6 axes, there are some differences. In this description of the differences the robot type is assumed to be similar to IRB 260, IRB 460, IRB 660, IRB 760, or IRB 910SC.

The main differences are:

- The used axes are:
 - 1 (or 2 for some robot models)
 - 3 (for all robot models)

Continues on next page

4 Programming and testing

4.9.6 LoadIdentify, load identification service routine

Continued

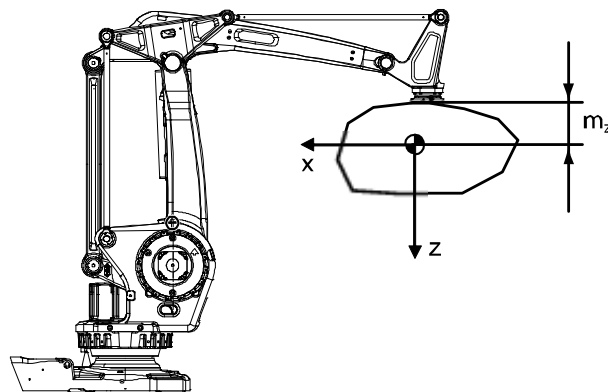
- 6 (or 4 for some robot models)
- Because the first axis (axis 1 or 2) is used, the resulting movements can be large.
- Not all load parameters can be identified.

The first axis (axis 1 or 2) will move approximately ± 23 degrees from its current position. Therefore, the load can move a large distance during the identification. Axes 3 and 6 (or 4) will move as for 6-axis robots. The configuration angle for axis 6 (or 4) works exactly as for 6-axis robots.

Because there is not 6 axes, a 4-axis robot cannot identify all parameters of the load. The following parameters cannot be identified:

- I_x - The inertia around the x-axis.
- I_y - The inertia around the y-axis.
- m_z - The z-coordinate for the center of mass.

However, for this type of robot the above parameters have negligible effect on the motion performance. See the definition of the load coordinate system in the following figure.



xx0900000021



Tip

It is possible that the identification procedure fails to estimate the center of gravity if the measured torque data has too high variance. If this happens, it should still be possible to get good results by running the LoadIdentify routine again, preferably with another position of the last axis.

LoadIdentify for 4-axis delta robots

When running LoadIdentify on a delta (picker/packer) robot with 4 axes, there are some differences. In this description of the differences the robot type is assumed to be similar to IRB 360, IRB 365, or IRB 390.

The main differences are:

- The used axes are:
 - 3 (for all robot models)
 - 4 (for all robot models)

Continues on next page

- Not all load parameters can be identified.

Axis 4 will move approximately +60 degrees from its current position. Therefore, the load will move a large rotational distance during the identification.

Because there is not 6 axes, a 4-axis delta robot cannot identify all parameters of the load. In fact, only the following parameters are possible to identify:

- I_z , the inertia around the z-axis.
- m , the total mass.

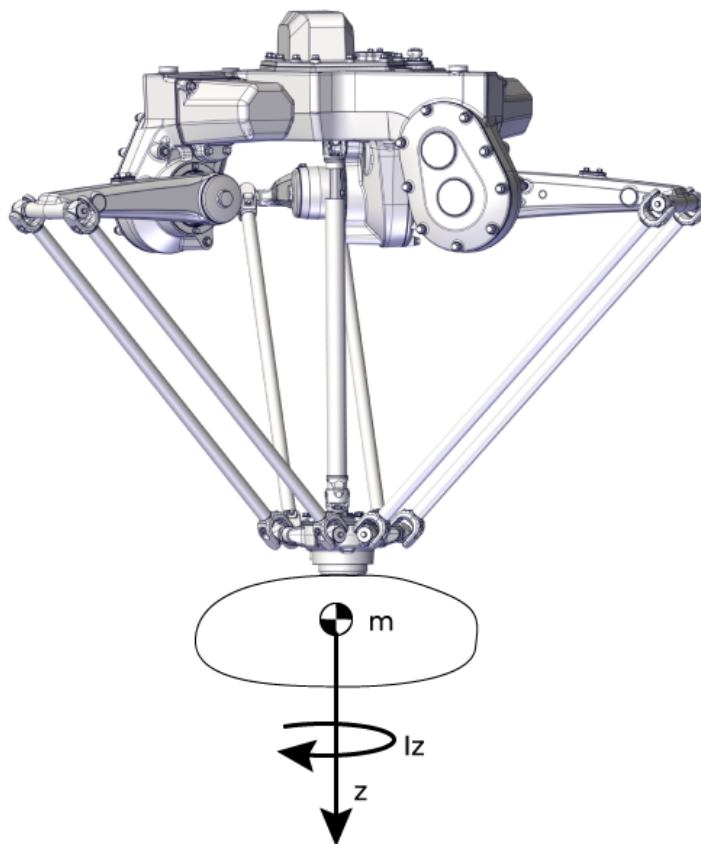
However, for this type of robot the total mass and the inertia around the z-axis is enough to give good motion performance. All other `loaddata` parameters will be set to zero.

**Note**

An x,y-offset in the mass center (m_x, m_y) should not be manually entered since this information will be included in the identified I_z .

(However, if known, an offset m_z (z-coordinate for the center of mass) can be manually entered to slightly enhance motion performance.)

See the resulting `loaddata` from LoadIdentify for 4-axis delta robots in the following figure.



xx2100002581

Continues on next page

4 Programming and testing

4.9.6 LoadIdentify, load identification service routine

Continued

LoadIdentify for 5-axis delta robots

When running LoadIdentify on a delta (picker/packer) robot with 5 axes, there are some differences. In this description of the differences the robot type is assumed to be similar to IRB 365 or IRB 390.

The main differences are:

- The used axes are:
 - - 3 (for all robot models)
 - - 4 (for all robot models)
 - - 5 (for all robot models)
- Not all load parameters can be identified.

Axis 4 will move approximately +45 degrees from its current position. Therefore, the load will move a large rotational distance during the identification.

Axis 5 will move approximately +35 degrees from its current position. Therefore, the load will move a large rotational distance during the identification.

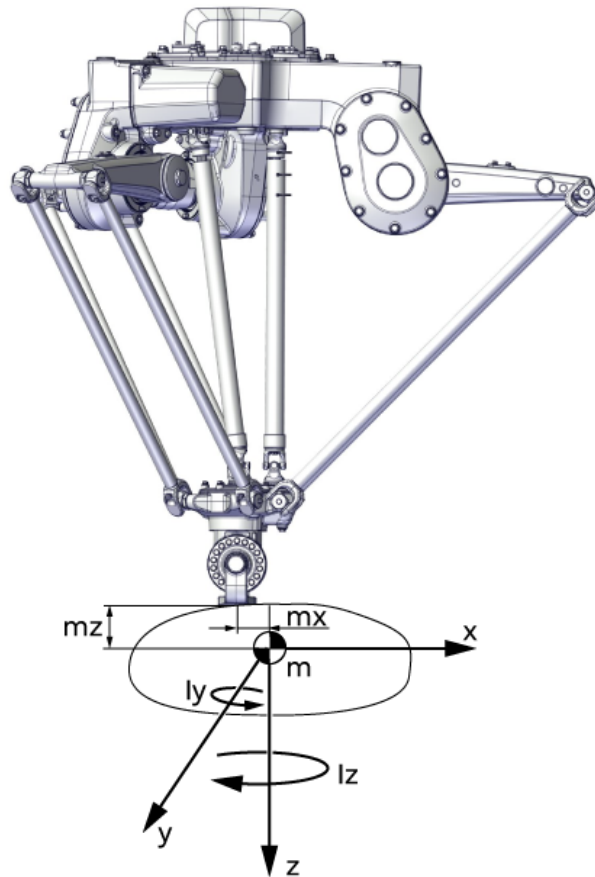
Instead of the usual 0-90 degree configuration movement, the axis 5 will move ± 45 degrees.

Because there is not 6 axes, a 5-axis delta robot cannot identify all parameters of the load. The following parameters are possible to identify:

- I_y , the inertia around the y-axis.
- I_z , the inertia around the z-axis.
- m_x , the x-coordinate for the center of mass.
- m_z , the z-coordinate for the center of mass.
- m , the total mass.

Continues on next page

See the resulting `loaddata` from `LoadIdentify` for a 5-axis delta robot in the following figure.



xx2100002620

Related information

It is also possible to include `LoadIdentify` in a program by using RAPID instructions. See `LoadID` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

How to enter the data manually is described in [Editing the tool data on page 163](#), and [Editing the payload data on page 181](#).

The product manual for the robot contain information on how and where to mount the loads.

Load identification for positioners is done with the service routine `ManLoadIdentify`, see *Product manual - IRBP /D2009*.

How to define the system parameters for arm loads is described in *Technical reference manual - System parameters*.

4 Programming and testing

4.9.7 Brake check service routine

4.9.7 Brake check service routine

Overview

The BrakeCheck service routine is used to verify whether the mechanical brakes work correctly.

The BrakeCheck service routine is included in the RobotWare installation if the controller does not have SafeMove or EPS option.



Note

If the controller has SafeMove or EPS option, the RobotWare installation includes *Cyclic Brake Check* service routine. For more details, see *Application manual - Functional safety and SafeMove2*.

While running the BrakeCheck service routine the brakes are tested in consecutive order and each test takes 10-15 seconds.

Prerequisites for running the BrakeCheck service routine

Following are the prerequisites for running the BrakeCheck service routine:

- The robot and all additional axes must be moved to a safe and relaxed position (away from people, equipment and not too much stretched) before performing a brake check. Normally the robot moves only a few centimeters during the brake check.
- Move the robot to a stop point before performing a brake check.
- The brake check can be performed only at normal execution level (not from a trap routine, error handler, event routine, or store path level).

Exclude individual axes from the brake check

It is possible to exclude individual axes from the brake check. For this, set the value of system parameter `Deactivate Cyclic Brake Check for axis` to On. For more details, see [Configure system parameters on page 222](#).

Running the brake check

Following are the two ways to initiate a BrakeCheck service routine:

- Call the BrakeCheck service routine from FlexPendant. The controller must be in manual mode.
- Call the procedure BrakeCheck from the RAPID program.



WARNING

While the brake check routine is active, do not change the speed from the FlexPendant and do not use the instructions `VelSet`, `AccSet`, `SpeedRefresh`, or any other instruction that affects the motion performance in trap routines or event routines.

Continues on next page

**Note**

The RAPID function `IsBrakeCheckActive` can be used to check if `BrakeCheck` is active.

Interrupt the brake check

It is not recommended, but it is possible to stop the execution while running a brake check.

If the brake check is interrupted, it will be resumed when the program execution starts again. The brake check can be resumed up to 3 times.

Brake maintenance

Brake maintenance is a feature in the brake check functionality.

The `BrakeCheck` routine automatically detects if maintenance of the mechanical brakes is needed and activates the *Brake maintenance* functionality during execution. *Brake maintenance* applies the brake and turns the motor shaft 1 radian five times, which gives a movement of the robot arm of less than 1 degree.

There are event logs that tell if *Brake maintenance* is needed, and if it has been run.

For more information see parameter *Brake Maintenance*, type *General Rapid*, topic *Controller*, in *Technical reference manual - System parameters*.

Event logs

When `BrakeCheck` is executed, the following event logs will be shown:

Event log	Title
10272	Brake Check Done
10273	Brake Check Started

If there is a problem with one or several mechanical brakes, an event log that is describing which mechanical unit and which axis that has bad brakes will be shown:

Event log	Title
37234	Brake Performance Warning
37235	Brake Performance Error

Brake check for MultiMove systems**Note**

Make sure that all mechanical units are standing still before starting a `BrakeCheck` routine.

One of the motion tasks call the routine `BrakeCheck` to perform brake check for all mechanical units in all tasks.

The brake check must not be performed while any tasks are in synchronized mode (coordinated movement). It is necessary to synchronize all motion tasks with `WaitSyncTask` instructions before and after the actual brake check. If running a

Continues on next page

4 Programming and testing

4.9.7 Brake check service routine

Continued

movement instruction when one motion task is execution a `BrakeCheck`, you will have an error (41888) and all execution will stop. Instruction `ExitCycle` is also forbidden to use during an active `BrakeCheck`.

The RAPID function `IsBrakeCheckActive` can be used to check if there is an ongoing `BrakeCheck`.

Only one call to `BrakeCheck` can be done at a time. This is checked by the service routine and if more than one RAPID task or client try to execute the routine, you will have an error (41886).

Program example

```
T_ROB1
PERS tasks task_list{2} := [ ["T_ROB1"], ["T_ROB2"] ];
VAR syncident sync1;
VAR syncident sync2;
...
IF PLC_dil_DO_CBC = 1 THEN
    WaitSyncTask sync1, task_list;
    BrakeCheck;
    WaitSyncTask sync2, task_list;
ENDIF

T_ROB2
PERS tasks task_list{2} := [ ["T_ROB1"], ["T_ROB2"] ];
VAR syncident sync1;
VAR syncident sync2;
...
IF PLC_dil_DO_CBC = 1 THEN
    WaitSyncTask sync1, task_list;
    ! Wait for T_ROB1 to be ready with BrakeCheck
    WaitSyncTask sync2, task_list;
ENDIF
```

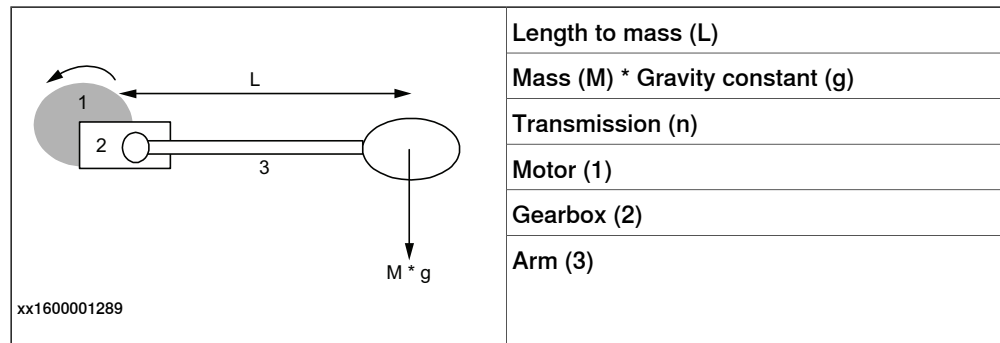
Brake check on additional axes

To be able to run brake check on additional axes, the parameter *Max Static Arm Torque* (in topic *Motion* and type *Brake*) needs to be calculated for the additional axis and entered into the configuration. Brake check uses this value when testing the brake at error-level.

The parameter should be the maximum static torque that the brake needs to withstand when the additional axis is positioned in maximum gravity. The following formula should be used:

$$\text{Max Static Arm Torque} = (M \cdot L \cdot g) / n$$

Continues on next page



To calculate the parameter for an axis that has no gravity, for example a track, the below formula may be used:

$$\text{Max Static Arm Torque} = T_{\text{brake min}}/1.35$$

$T_{\text{brake min}}$ for ABB motor units can be found in the product specification for the specific motor unit, see *Product specification - Motor Units and Gear Units*.

For more information about parameter *Max Static Arm Torque*, see topic *Motion*, type *Brake* in *Technical reference manual - System parameters*.



Note

Note that the calculated value should be entered in [Nm] and calculated to the motor side.

Description of the I/O setup

Signal configuration

It is possible to configure digital output signals that reflect the status of the mechanical brakes in an open RAPID module. The digital output signals that can be configured are OK, WARNING, ERROR, and ACT (brake check active) for each drive module.

The signal configuration should be done in the RAPID module *BC_config_IO.sys*, see [Description of the I/O setup on page 219](#).

The file *BC_config_IO.sys* can be found in directory *hd0a\<active system>\PRODUCTS\RobotWare_6.0x.xxxx\utility\BrakeCheck*, and must then be copied to the *HOME* directory of the active system.



Note

Remember to update the I/O configuration with the digital output signals.

In a MultiMove system you need to define an OK, WARNING, ERROR, and ACT digital output signals for each drive module.



Note

If the signals should keep their values after a power fail, the power fail settings in the system parameters must also be updated, see [Description of the I/O setup on page 219](#).

Continues on next page

4 Programming and testing

4.9.7 Brake check service routine

Continued

Description of the BC_config_IO module

```
MODULE BC_config_IO(SYSMODULE,NOVIEW)
  PROC BC_config_IO_proc(VAR string user_io_names{*,*})
    !TPWrite "BC_config_IO_proc";
    ! Define your own signals. The signal
    ! names must be signals defined in EIO.cfg

    ! If 1 drive module
    user_io_names{1, 1}:="BCACT1";
    user_io_names{1, 2}:="BCOK1";
    user_io_names{1, 3}:="BCWAR1";
    user_io_names{1, 4}:="BCERR1";

    ! If 2 drive modules
    !user_io_names{2, 1}:="BCACT2";
    !user_io_names{2, 2}:="BCOK2";
    !user_io_names{2, 3}:="BCWAR2";
    !user_io_names{2, 4}:="BCERR2";

    ! If 3 drive modules
    !user_io_names{3, 1}:="BCACT3";
    !user_io_names{3, 2}:="BCOK3";
    !user_io_names{3, 3}:="BCWAR3";
    !user_io_names{3, 4}:="BCERR3";

    ! If 4 drive modules
    !user_io_names{4, 1}:="BCACT4";
    !user_io_names{4, 2}:="BCOK4";
    !user_io_names{4, 3}:="BCWAR4";
    !user_io_names{4, 4}:="BCERR4";
  ENDPROC
ENDMODULE
```

Description of the EIO.cfg file

```
EIO:CFG_1.0:6:1::
...
#
EIO_SIGNAL:

-Name "BCACT1" -SignalType "DO"

-Name "BCOK1" -SignalType "DO"

-Name "BCWAR1" -SignalType "DO"

-Name "BCERR1" -SignalType "DO"
```

Continues on next page

Brake check signal description

Introduction

Description of different signal states for brake check in the BrakeCheck routine. The signal names are according to [Description of the I/O setup on page 219](#).

Timing sequence for brake check signals

Description of which signals are set at different times during the BrakeCheck execution.

Beginning of brake check

The following signals are set in the beginning of the BrakeCheck execution.

Signal	Set to
BCOK	0
BCACT	1
BCERR	0
BCWAR	0

End of brake check

The following signals are set in the end of the BrakeCheck execution.

Signal	BrakeCheck test OK Set to	BrakeCheck test WARNING Set to	BrakeCheck test ERROR Set to
BCOK	1	0	0
BCERR	0	0	1
BCWAR	0	1	0
BCACT	0	0	0

Program Pointer moved to Main after interrupted brake check

When the program pointer is moved to Main after an interrupted BrakeCheck execution, the following signals are set.

Signal	Set to
BCOK	0
BCACT	0

During the first brake check test

Signal	Signal state
BCOK	0
BCERR	0
BCWAR	0
BCACT	1

Interrupted brake check test, program pointer still in BrakeCheck routine

Signal	Signal state
BCOK	0

Continues on next page

4 Programming and testing

4.9.7 Brake check service routine

Continued

Signal	Signal state
BCERR	0
BCWAR	0
BCACT	1

Interrupted brake check test, program pointer moved from BrakeCheck routine

Signal	Signal state
BCOK	0
BCERR	0
BCWAR	0
BCACT	0

Configure system parameters

About the system parameters

The configuration of system parameters required for a robot system should be made before running the brake check.



Note

The controller must be restarted after changing the system parameters.

Type Mechanical Unit

All mechanical units for additional axes that shall be supervised must have the parameters *Activate at Start Up* and *Deactivation Forbidden* set to On. (Supervised mechanical units must always be active.)

Type Arm

If an axis should be excluded from brake check, set the parameter *Deactivate Cyclic Brake Check for axis* to On.

Type Brake

If brake check is executed on an additional axis, a lowest safe brake torque must be defined. A 5% margin is added during the test for setting the fail limit. The parameter used is *Max Static Arm Torque* defined in on motor side. A warning limit is set with a higher torque value (depending on the brake).

4.9.8 Wrist optimization service routine

Overview

The wrist optimization service routine is used to improve the TCP reorientation performance. More information is available in the product manual for the manipulator.

This page is intentionally left blank

5 Running in production

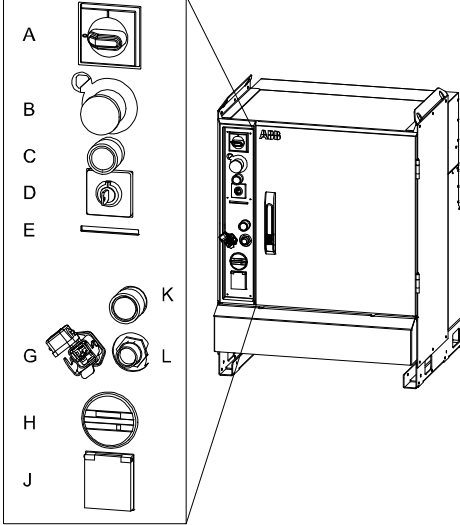
5.1 Basic procedures

5.1.1 Starting programs

Starting programs

Use this procedure to start a program for the first time or to continue running a program that has been stopped.

If your robot system has the option *Multitasking* installed, also see [Using multitasking programs on page 229](#).

	Action	Information
1	Check that all necessary preparations are done to the robot and in the robot cell and that no obstacles exist within the robot work area.	
2	Make sure no personnel are inside the robot cell.	
3	Select operating mode on the controller with the mode switch.	 <p data-bbox="975 1514 1082 1532">xx0600002782</p> <p data-bbox="975 1552 1198 1574">C: Motors on button</p> <p data-bbox="975 1585 1145 1608">D: Mode switch</p>
4	Press the Motors on button on the controller to activate the robot.	
5	Is a program loaded? If yes, proceed to the next step. If no, load a program.	How to load programs is described in section Handling of programs on page 128 .
6	If needed, select run mode and speed using the Quickset menu.	See Quickset menu, Run Mode on page 64 and Quickset menu, Speed on page 66 .

Continues on next page

5 Running in production

5.1.1 Starting programs

Continued

	Action	Information
7	In Auto mode: 1 Press the Start button on the Flex-Pendant to start the program. In manual mode: 1 Select start mode. 2 Press and hold the three-position enabling device. 3 Press the Start button on the Flex-Pendant to start the program.	The button is shown in the section Hard buttons on page 19 . How to select start mode is detailed in section Using the hold-to-run function on page 190 .
8	Is the Regain Request dialog box displayed? If yes, return the robot to the path using a suitable method. If no, proceed.	Returning the robot to the path is described in section Returning the robot to the path on page 240 .
9	If the Cursor does not coincide with PP dialog box is displayed then tap PP or Cursor to select from where the program should start. Then press the Start button again.	This dialog box is only displayed if the system parameters of type <i>Warning at start</i> are defined. See <i>Technical reference manual - System parameters</i> .

Continue running after the program is changed

You can always continue a program even if it has been changed.

In automatic mode, a warning dialog may appear to avoid restarting the program if the consequences are unknown.

If you...	then tap...
Are sure the changes you have made are not in conflict with the current robot position and that the program can continue without danger to equipment or personnel	Yes
Are unsure of the consequences your changes might have and want to investigate further	No

Restart from the beginning

A program can be restarted from the **Production Window** or the **Program Editor**. **PP to Main** from the **Production Window** will reset the program pointer to the production entry in all normal tasks, including tasks deactivated in the task selection panel.

PP to Main from the **Program Editor** will reset the program pointer to the production entry in the specified task only, even if the task is deactivated in the task selection panel.

Use this procedure to restart a program from the **Production Window**.

	Action
1	On the ABB menu, tap Production Window .
2	Tap PP to Main .
3	Start the program by pressing the Start button on the FlexPendant.

Use this procedure to restart a program from the **Program Editor**.

	Action
1	On the ABB menu, tap Program Editor .

Continues on next page

	Action
2	Tap Debug .
3	Tap PP to Main .
4	Start the program by pressing the Start button on the FlexPendant.

Limitations

Only one program at a time can be executed, unless your system has the *Multitasking* option. If so several programs can be executed simultaneously. See how to select tasks in [Quickset menu, Tasks on page 67](#).

If the robot system encounters program code errors while the program is running, it will stop the program and the error is logged in the event log.

5 Running in production

5.1.2 Stopping programs

5.1.2 Stopping programs

Stopping programs

If your robot system has the *Multitasking* option installed, see [Using multitasking programs on page 229](#).

	Action
1	Check that the ongoing operation is in such a state that it can be interrupted.
2	Make sure it is safe to stop the program.
3	Press the Stop button on the hardware button set of the jogging device.



DANGER

Do not use the **Stop** button in an emergency. Use the emergency stop button. Stopping a program with the **Stop** button does not mean that the robot will stop moving immediately.

Stopping execution when using hold-to-run or step-by-step execution

When using hold-to-run or step-by-step execution, execution can be stopped according to the following.

Mode	Action	Information
Operation <i>with</i> hold-to-run	Release the Start button	The hold-to-run function is described in section The FlexPendant on page 17 .
Step-by-step mode	The robot will stop after executing each instruction. Execute the next instruction by pressing the Forward button again.	The STOP and Forward button are described in section The FlexPendant on page 17 . If you press the STOP button while executing a move instruction, the robot will stop without completing the move.

5.1.3 Using multitasking programs

Overview

In a system with the option *Multitasking* installed, you may have one or several programs running in parallel, for instance in a *MultiMove* cell with more than one robot where each robot has its own task and program (multitasking).

For general information on program handling, see [Handling of programs on page 128](#). Multitasking is described in *Application manual - Controller software IRC5*.

Manually set up tasks

Tasks need to be set up in order to run as planned. Normally, all tasks are set up on delivery. Setting up tasks is done by defining system parameters of the type *Controller*, see *Technical reference manual - System parameters*.

You need detailed information to set up tasks manually. Please read your plant or cell documentation for details.

How tasks are run

Tasks may be defined as Normal, Static, or Semistatic. Static and Semistatic tasks are automatically started as soon as a program is loaded into that task.

Normal tasks are started when you press the **Start** button of the FlexPendant, and stopped when you press the **Stop** button.

To be able to step, start and stop a Static or Semistatic task: set **Task Panel Settings** to **All tasks** and activate the task using the **Quickset** menu. See *Application manual - Controller software IRC5*, section *Multitasking*.

The concepts of Static, Semistatic, and Normal are described in *Technical reference manual - System parameters*, type *Tasks*.

Load, run, and stop multitasking programs

This section describes how to load, run, and stop multitasking programs.

	Action
1	Make sure there is more than one task set up. This is done using system parameters, see <i>Technical reference manual - System parameters</i> .
2	Load programs to respective task using the Program Editor or the Production Window , this is described in section Loading an existing program on page 129 .
3	If one or more task should be disabled, go to the Quickset menu to do this. See section Quickset menu, Tasks on page 67 . Deselecting tasks can only be done in manual mode. When switching to automatic mode, an alert box will appear warning that not all tasks are selected to run.
4	Start program execution by pressing the start button. All active tasks are started.
5	Stop program execution by pressing the stop button. All active tasks are stopped.

Continues on next page

5 Running in production

5.1.3 Using multitasking programs

Continued

How to load a program to a task

This section describes how to load a program to a task in a multitasking system. It is assumed that the tasks have been configured.

Load a program from the **Production Window**

	Action
1.	On the ABB menu, tap Production Window .
2.	Tap the task into which you want to load a program.
3.	Tap Load Program.... If you want to open a program in another folder, locate and open that folder. See description in FlexPendant Explorer on page 36 . The file dialog box appears.
4.	Tap the program you want to load followed by OK .

Load a program from the **Program Editor**

	Action
1.	On the ABB menu, tap Program Editor .
2.	Tap Tasks and Programs .
3.	Tap the task into which you want to load a program.
4.	On the File menu, tap Load Program.... If you want to open a program in another folder, locate and open that folder. See description in FlexPendant Explorer on page 36 . The file dialog box appears.
5.	Tap the program you want to load followed by OK .
6.	Tap Close to close the Program Editor.

Viewing multitasking programs

In the **Production Window**, there is one tab for each task. To switch between viewing the different tasks, tap on the tabs.

To edit several tasks in parallel, open one **Program Editor** for each task. To edit static and semistatic tasks, see *Application manual - Controller software IRC5*, section *Multitasking*.

5.1.4 Using motion supervision and non motion execution

Motion supervision

The controller software has functionality aiming at reducing collision impact forces on the robot. This helps protecting the robot and external equipment from severe damage if a collision occurs.

Motion supervision during program execution is by default always active, regardless which options are installed in the controller. When a collision is detected, the robot will immediately stop and relieve the residual forces by moving in reversed direction a short distance along its path. The program execution will stop with an error message. The robot remains in the state *Motors on* so that program execution can be resumed after the collision error message has been acknowledged.

Moreover, there is a software option called *Collision Detection*, which has extra features such as supervision during jogging. To find out if your system has this option installed, tap **System Info** on the ABB menu. Expand the node *System Properties* and tap *Options* under *Control Module*.

Functions in RobotWare base

Description of functions in RobotWare base:

- *Path Supervision* in automatic and manual full speed mode used to prevent mechanical damage due to the robot running into an obstacle during program execution.
- *Non motion execution* used to run a program without robot motion.

Functions in Collision Detection

A RobotWare system with *Collision Detection* has additional functionality:

- *Path Supervision* in manual mode and the possibility to tune supervision in all modes.
- *Jog Supervision* used to prevent mechanical damage to the robot during jogging.
- RAPID instruction `MotionSup` used to activate/deactivate collision detection and to tune sensitivity during program execution.



Note

All motion supervision must be set for each task separately.

Continues on next page







5 Running in production

5.1.4 Using motion supervision and non motion execution

Continued

Editing motion supervision settings

This section describes how to modify settings for motion supervision.

	Action	Information
1	On the ABB menu tap Control Panel and then Supervision .	
2	Tap the Task list and select a task.	If you have more than one task, you need to set the desired values for each task separately.
3	<p>Tap OFF/ON to remove or activate path supervision. Tap the 123... button to open the soft number pad and type a value to adjust the sensitivity of path supervision.</p> <p> Note</p> <p>If the option <i>Collision Detection</i> is not installed,</p> <ul style="list-style-type: none"> • sensitivity setting will have no effect. • path supervision affects only the robot in auto and manual full speed mode. 	<p> Tip</p> <p>Sensitivity can be set between 0 and 300. If it is set lower than 80, however, the robot may stop due to internal drag.</p> <p> Note</p> <p>You can modify the sensitivity of <i>Path supervision</i>. For more information, see Setting sensitivity of Motion Supervision on page 233.</p>
4	<p>Tap OFF/ON to remove or activate jog supervision. Tap the 123... button to open the soft number pad and type a value to adjust the sensitivity of jog supervision.</p> <p> Note</p> <p>If the option <i>Collision Detection</i> is not installed, these settings will have no effect.</p>	<p> Tip</p> <p>Sensitivity can be set between 0 and 300. If it is set lower than 80, however, the robot may stop due to internal drag.</p> <p> Note</p> <p>You can modify the sensitivity of <i>Path supervision</i>. For more information, see Setting sensitivity of Motion Supervision on page 233.</p>
5	Under Execution Settings , tap OFF/ON to deactivate or activate non motion execution. This is a separate function, not a part of motion supervision.	See Non motion execution on page 233 for information about this function.

Continues on next page

Setting sensitivity of Motion Supervision

Use the following procedure to set the sensitivity of *Path Supervision* and *Jog Supervision*.

	Action	Information
1	On the ABB menu, tap Control Panel and then Configuration .	
2	Tap Topics and select Motion .	
3	Select the type Motion Supervision and tap.	
4	Select one from the list and tap Edit .	For example: rob1
5	Select <i>Path Collision Detection Level</i> , tap twice and set a value.	The maximum value that can be set is 500.
6	Click OK .	
7	Select <i>Jog Collision Detection Level</i> , tap twice and set a value.	The maximum value that can be set is 500
8	Click OK .	

Non motion execution

Non motion execution enables you to run a RAPID program without robot motion. All other functions work normally; current cycle times, I/O, TCP speed calculation and so on.

Non motion execution can be used for program debugging or cycle time evaluation. It also represents a solution if you need to measure for example glue or paint consumption during a cycle.

When non motion execution is activated it can be executed in:

- manual mode
- manual full speed mode
- auto mode

Cycle times will be simulated according to the selected mode.



Note

Non motion execution can only be activated when the system is in **Motors Off** state.



CAUTION

Non motion execution is reset after a reboot. If you intend to run the program in non motion mode, do not restart without checking the status of **Non motion execution**. Starting the program incorrectly may cause serious injury, or damage the robot or other equipment.

Related information

For more information on *Collision Detection*, see *Application manual - Controller software IRC5*.

5 Running in production

5.1.5 Connecting a FlexPendant

5.1.5 Connecting a FlexPendant

Location of FlexPendant connector

The FlexPendant connector is located on the operator's panel on the controller, or on an external operator's panel. The Panel Mounted Controller has a connector on the front.

Connecting a FlexPendant



CAUTION

Always inspect the connector for dirt or damage before connecting it to the controller. Clean or replace any damaged parts.

	Action	Information
1	Locate the FlexPendant socket connector on the controller or operator's panel.	The controller must be in manual mode. If your system has the option Hot plug, then you can also disconnect in auto mode. See section Using the hot plug option on page 236 . The controller must be in manual mode.
2	Plug in the FlexPendant cable connector.	
3	Screw the connector lock ring firmly by turning it clockwise.	
4	The FlexPendant starts automatically when connected and verifies that it has the correct software installed. If an update is needed, this is shown.	Updating the add-in FlexPendant SxTPU4 Software on page 234

Updating the add-in FlexPendant SxTPU4 Software



Note

The add-in is only available for the FlexPendant with the emergency stop located at the connector. All other FlexPendant versions will automatically update their software via the controller (if needed).

The FlexPendant with the emergency stop located at the connector has an add-in that enables support for different RobotWare versions. This is the **FlexPendant SxTPU4 Software** add-in. The version of the add-in is shown during start-up.

When connecting the FlexPendant, the add-in verifies that it has support for the RobotWare version on the controller. If the RobotWare version is not supported by default, then the add-in requires an update. There are two methods to update the FlexPendant add-in. The update is distributed as a software package.

- The update can be installed using a USB drive.
- If the update is available on the controller, then the FlexPendant will update itself when connecting it to the controller.

Continues on next page

Once the add-in is updated, the FlexPendant can be connected to other IRC5 controllers with the same RobotWare version without requiring additional updates.

Update using a USB drive

Use the following procedure to update the add-in using a USB drive.

- 1 Download the update from RobotStudio, in the tab **Add-Ins**.
- 2 Save the software package (.rspak) on a USB drive in the folder `SxTPU4`, located in the root folder.
- 3 With the FlexPendant connected to the controller, reset the FlexPendant with the USB drive connected.
- 4 The update starts automatically and takes approximately 3-4 minutes.

Update from the controller

Use the following procedure to update the add-in from the controller.

- 1 In RobotStudio, use **Installation Manager 6** to create or update a system on your controller. Add the product *FlexPendantSxTPU4Software*.
- 2 Connect the FlexPendant to the controller.
- 3 The update starts automatically and takes approximately 3-4 minutes.

Handling the FlexPendant cables

FlexPendant cables are allowed to be rolled up by hand with a minimum bending radius of 10 times the cable diameter. This also applies to the extension cable. For example, if the cable is 9.5 mm then it is allowed to roll it with a radius of 95 mm.

Extension cables are not allowed to be used in chains.

5 Running in production

5.1.6 Using the hot plug option

5.1.6 Using the hot plug option

Hot plug option

The hot plug option makes it possible to:

- Disconnect the FlexPendant from a system in automatic mode and thereby run the system without a FlexPendant connected.
- Temporarily connect and operate a FlexPendant without interrupting the application running on the system.



WARNING

Pressing the hot plug button disables the emergency stop button on the FlexPendant. Only press the hot plug button while connecting or disconnecting the FlexPendant.



WARNING

A disconnected FlexPendant cannot initiate a protective or emergency stop. It must be stored out of sight so that it cannot be mistaken for being in use.

Connect and disconnect the FlexPendant using the hot plug button

The following procedure describes how to connect or disconnect the FlexPendant on a system in automatic mode using the hot plug button option.

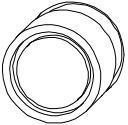

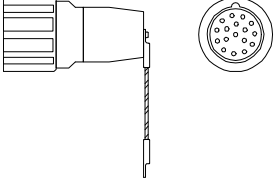


Note

Do not switch to manual mode (or manual full speed mode) while the system is running without the FlexPendant. The FlexPendant must be connected when you switch to automatic mode otherwise you cannot confirm the mode change.

	Action	Information
1	Make sure that the system is in automatic mode.	
2	Press and hold the hot plug button.	A red lamp inside the button indicates when pressed.

Continues on next page

	Action	Information
3	Keep pressing the hot plug button and at the same time, switch the jumper plug with the FlexPendant plug.	<p data-bbox="978 353 1002 387">A</p>  <p data-bbox="978 504 1002 537">B</p>  <p data-bbox="978 600 1082 611">xx0600002784</p> <p data-bbox="978 638 1177 660">A: Hot plug button</p> <p data-bbox="978 672 1257 694">B: FlexPendant connector</p>  <p data-bbox="978 891 1082 902">xx0600002796</p> <p data-bbox="978 929 1114 952">Jumper plug</p>
4	Release the hot plug button.	Make sure that the button is not stuck in the actuated position since this disables the FlexPendant emergency stop button.
5	If the connected FlexPendant does not have support for the RobotWare version running on the controller, then a dialog is shown that the add-in must be updated. See Updating the add-in FlexPendant Sx-TPU4 Software on page 234 .	The three-position enabling device and emergency stop button are active even if the add-in dialog is shown.



Note

When the FlexPendant is disconnected, the jumper plug must be connected in its place.



Note

If the hot plug button is released while neither the jumper plug, nor the FlexPendant is connected, the robot movements will be stopped since the emergency stop chains are opened.

5 Running in production

5.1.6 Using the hot plug option

Continued

Limitations for messages on the FlexPendant

When using the hot plug option, the following limitations apply to messages on the FlexPendant:

Operator messages

Some applications may require input from the operator by using the FlexPendant (e.g. applications using RAPID instructions `TPReadNum`, `UIMsgBox`, etc.). If the application encounters such an operator message, program execution will wait. After connecting the FlexPendant you must then stop and start the program execution to be able to see and respond to these messages. They are not displayed automatically by just connecting the FlexPendant.

If possible, avoid using these types of instructions when programming systems that are using the hot plug button option.

Event log messages

When connecting the FlexPendant, event log messages can be viewed also for the period when the FlexPendant was disconnected, since these are stored on the controller.

5.2 Troubleshooting and error recovery

5.2.1 General procedure when troubleshooting

Types of faults

Faults occurring in the robot system may be of two categories:

- Faults detected by the built-in diagnostics system. These faults are described in section *Event log messages* in *Operating manual - Troubleshooting IRC5*.
- Faults NOT detected by the built-in diagnostics system. These faults are described in section *Other types of faults* in *Operating manual - Troubleshooting IRC5*.

Faults causing error message on the FlexPendant

The control system is supplied with diagnostic software to facilitate troubleshooting and to reduce downtime. Any errors detected by the diagnostics are displayed in plain language with a code number on the FlexPendant.

All system and error messages are logged in a common log in which the last 150 messages are saved. The log can be accessed from the Status bar on the FlexPendant.

To facilitate troubleshooting, it is important that some basic principles are followed. These are specified in *troubleshooting principles* in *Operating manual - Troubleshooting IRC5*.

Faults NOT causing error messages on the FlexPendant

These faults are not detected by the diagnostic system and are handled in other ways. The way the symptom of the fault is observed greatly influences the type of fault. Instructions are given in section *Other types of faults* in *Operating manual - Troubleshooting IRC5*.

To troubleshoot faults NOT causing error messages on the FlexPendant, follow steps 3 and 4 in the procedure above.

Other possible actions

Some errors may require running a service routine. See section [Service routines on page 196](#).

5 Running in production

5.2.2 Returning the robot to the path

5.2.2 Returning the robot to the path

About paths and return regions

While a program is running, the robot or additional axis is considered to be *on path*, which means that it follows the desired sequence of positions.

If you stop the program the robot is still on path, unless you change its position. It is then considered to be *off path*. If the robot is stopped by an emergency or safety stop it may also be off path.

If the stopped robot is within the *path return region* you can start the program again, and the robot will return to the path and continue program execution.

Note that there is no way to predict the exact return movement for the robot.



Tip

The path return region is set with system parameters, see *Technical reference manual - System parameters, Type Path Return Region*.

Returning to path

Turning off the power to the robot motors often results in the robot slipping from its programmed path. This may occur after an uncontrolled emergency or safety stop. The allowed slip distance is configured with system parameters. The distance can be different depending on operating mode.

If the robot is not within the configured allowed distance, you may choose to let the robot return to the programmed path or continue to the next programmed point in the path. Then the program execution continues automatically in programmed speed.

- 1 Make sure there are no obstacles blocking the way and that payload and work objects are properly placed.
- 2 If necessary, put the system in automatic mode and press the Motors on button on the controller to activate the robot motors.
- 3 Press the Start button on the FlexPendant to continue execution from where it stopped. One of these things will happen:
 - The robot or axis slowly returns to the path and the execution continues.
 - The **Regain Request** dialog will be displayed.
- 4 If the **Regain Request** dialog is displayed, select the proper action.
 - Tap **Yes** to return to the path and continue the program.
 - Tap **No** to return to the next target position and continue the program.
 - Tap **Cancel** to cancel the program.

5.2.3 Running RAPID program with uncalibrated mechanical unit

When is this useful?

If a servo gun is damaged or uncalibrated, you may want to run a service routine. In order to run the service routine (or any RAPID code), even though an additional axis is uncalibrated, the steps in this description must be followed.

How to get the program started

	Action
1	Set the system parameter <i>Active at Start Up</i> (in type <i>Mechanical Unit</i> , topic <i>Motion</i>) to No. Set the system parameter <i>Disconnect at Deactivate</i> (in type <i>Measurement Channel</i> , topic <i>Motion</i>) to Yes.
2	If any of the system parameter values are changed, restart the controller.
3	Deactivate the uncalibrated mechanical unit.
4	Move the program pointer to Main (otherwise the mechanical unit will be automatically activated).
5	Run the service routine or other RAPID code.

5 Running in production

5.3.1 Present operating mode

5.3 Operating modes

5.3.1 Present operating mode

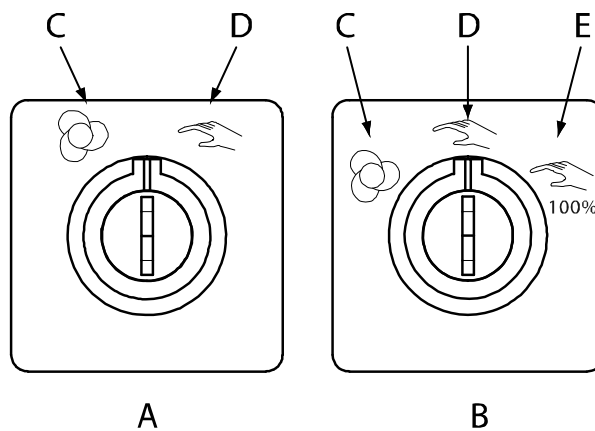
Overview

Check the position of the controller's mode switch or the status bar of the FlexPendant.

Operational mode changes are also logged in the event log.

The mode switch

The mode switch should be in the position as illustrated:



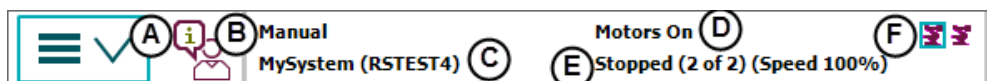
xx0300000466

A	Two position mode switch
B	Three position mode switch
C	Automatic mode
D	Manual reduced speed mode
E	Manual full speed mode

	Action	Information
1	To switch from manual to automatic mode	detailed in Switching from manual to automatic mode on page 244 .
2	To switch from automatic to manual mode	detailed in Switching from automatic to manual mode on page 246 .

Viewing present mode on the FlexPendant

On the FlexPendant, you can view the present operating mode in the status bar. An example of the status bar is shown below:



en0300000490

A	Operator window
---	-----------------

Continues on next page

B	Operating mode
C	Active system
D	Controller state
E	Program state
F	Mechanical units, active is highlighted

Related information

[About the automatic mode on page 185](#)

[About the manual mode on page 187](#)

5 Running in production

5.3.2 Switching from manual to automatic mode

5.3.2 Switching from manual to automatic mode

When should I put the system in automatic mode?

Put the system in automatic mode when you have a process application or a RAPID program that is ready to be run in production.

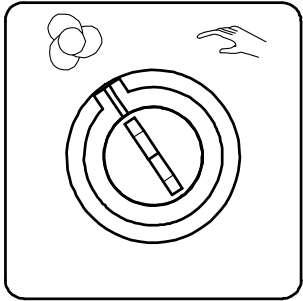


DANGER

When put in automatic mode the robot may move without warning.

Make sure no personnel are in safeguarded space before you change operating mode.

Switching from manual to automatic mode

	Action	Information
1	Set the mode switch in the automatic position. A mode change dialog is displayed.	 xx0300000467
2	If any debug settings have been changed, a dialog informs about the changes and if these values will be reset. Tap Acknowledge .	If these values are reset or not is defined by system parameters in the type <i>Auto Condition Reset</i> in the topic <i>Controller</i> .
3	Tap OK to close the dialog. If you change the switch back to manual mode the dialog will be closed automatically.	
4	Did the system change mode without errors? If yes, then resume or start the process application or RAPID program. If no, stop and troubleshoot the problem.	How to start programs is described in Starting programs on page 225 .



Note

If your specific system uses a distributed operator's panel, controls and indicators may not be placed exactly as described in this manual. Consult your plant or cell documentation for details.

Controls and indicators do however look and function the same way.

When can I start using the robot system?

As long as the mode change dialog is displayed programs cannot be started and the robot's motors cannot be activated either manually or remotely.

Continues on next page

Exceptions

In automatic mode it is possible to start a RAPID program and turn motors on remotely. This means that the system will never enter a safe standby state and the robot may move at any time.

Please consult your plant or cell documentation for details on how your system is configured.

Related information

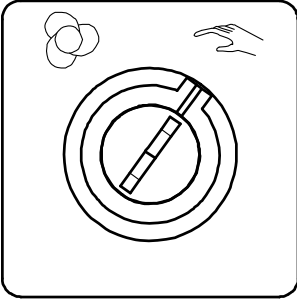
A number of conditions can be set or reset when switching to automatic mode, see *Technical reference manual - System parameters*, sections *Auto Condition Reset* and *Run Mode Settings*.

5 Running in production

5.3.3 Switching from automatic to manual mode

5.3.3 Switching from automatic to manual mode

Switching from automatic to manual mode

	Action	Information
1	Set the mode switch in the manual position.	 <p>xx0300000468</p>
2	Did the system change mode without errors? If yes, then this procedure is completed. If no, try to locate the error.	Error handling is detailed in <i>Operating manual - Troubleshooting IRC5</i> .



Note

If your specific system uses a distributed operator's panel, controls and indicators may not be placed exactly as described in this manual. Please consult your plant or cell documentation for details.

Controls and indicators do however look and function the same way.

5.3.4 Switching to manual full speed mode

When should I use the manual full speed mode?

Use full speed manual mode when the program is to be tested at full speed.

The manual full speed mode allows you to run the program at full speed while still having access to all the available debugging functions of the program editor.



DANGER

Testing at full speed is dangerous.

Make sure no personnel are in safeguarded space when starting the program.

Switching to manual full speed mode

	Action	Information
1	Set the mode switch to the manual full speed position.	
2	Did the system change mode without errors? If yes, then this procedure is completed. If no, try to locate the error.	Error handling is detailed in <i>Operating manual - Troubleshooting IRC5</i> .



Note

When you switch to manual full speed mode, all the functionality except for **Start**, **Stop**, and **Step** are disabled.

FlexPendant alert

When changing mode, a dialog is displayed on the FlexPendant to alert you about the change of mode. Tap **OK** to close the dialog.

If you switch back to the previous mode the dialog is closed automatically and there is no change in the mode.

5 Running in production

5.4.1 Modifying and tuning positions

5.4 Modifying positions

5.4.1 Modifying and tuning positions

Overview

Positions are instances of the data type `robtarg` or `jointtarg`. See *Technical reference manual - RAPID Instructions, Functions and Data types*.

The positions can be tuned using the function `HotEdit`, where you enter offset values using a soft keyboard. The offset value is used together with the original position value. See [Tuning positions with HotEdit on page 253](#). The `HotEdit` menu is described in section [HotEdit menu on page 34](#).

The positions can also be modified using the **Modify positions** function in the **Program Editor** or the **Production Window** where you step and jog the robot to the new position. A modified position value overwrites the original value. See [Modifying positions in the Program Editor or Production Window on page 249](#).



CAUTION

Changing programmed positions may significantly alter the robot's movement pattern.

Always make sure any changes are safe for both equipment and personnel.

Positions in arrays

If a position is declared as an array, then the procedure for modifying or tuning may differ slightly depending on how the array is indexed in the move instruction.

Limitations

Note that `jointtarg`s can only be modified using the **Modify positions** method in the **Program Editor** and the **Production Window**, i.e. not with `HotEdit`.



Note

Your system can have restrictions on how positions can be modified. Restrictions can apply to distance using system parameters (topic *Controller*, type *ModPos Settings*) and which positions can be modified using UAS.

5.4.2 Modifying positions in the Program Editor or Production Window

Overview

When modifying positions by jogging the robot to the new position you can either single-step through the program to the position(s) you want to modify, or jog directly to the new position and change the corresponding position argument of the instruction.

The recommendation is to step through the program to the position, but if you know your robot program well and the new position is known, it is faster to use the jogging method.



Note

Do not use this method to change orientation values.

Prerequisites

To modify positions using the Program Editor or Production Window.

- the system must be in manual mode
- the target position must have a initial value. For example: `CONST robtarget p10:=[[515.00,0.00,712.00],[0.707107,0,0.707107,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]]; CONST jointtarget jpos10:=[[-0,-0,0,-0,-0,-0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];`



Note

To modify positions in the Production Window, you must have started the program so that the motion pointer is set.

Applying modified positions

The modified position values will normally be used when you restart the program. If the robot cannot use the values directly at start, a warning is displayed. Then the modified position will be used the next time that the position is used in the program.

Modifying positions

This procedure describes how to modify positions, either by single-stepping to the positions or jogging. You can use the **Program Editor** or the **Production Window**, the functionality is the same.


	Action	Information
1	On the ABB menu, tap Program Editor .	
2	Stop the program, if running.	

Continues on next page

5 Running in production

5.4.2 Modifying positions in the Program Editor or Production Window

Continued

	Action	Information
3	Do you want to single-step to the position or jog? If <i>single-stepping</i> , step through the program to the position you want to change. Make sure the correct argument is selected. If <i>jogging</i> , use the Jogging view to make sure that the same work object and tool that are used in the instruction are selected.	When single-stepping, if the instruction or procedure call has more than one position argument, continue to step to reach each argument.
4	Jog to the new position.	
5	When using the jogging method, tap to select the position argument you want to change.	
6	In the Program Editor, tap Modify Position . In the Production Window, tap Debug and then Modify Position . A confirmation dialog appears.	When modifying a position in an array that is indexed with a variable you will have to select which element in the array to modify before the modification is executed.
7	Tap Modify to use the new position, Cancel to keep the original.	If you select the check box Don't show this dialog again in the confirmation dialog, then you will not get any more confirmation dialogs when modifying positions.  Note This is only valid for the current Program Editor .
8	Repeat step 3 through 7 for each position argument you want to change.	

Limitations

The **Modify position** button in the **Program Editor** is disabled until you select a position argument (that is possible to modify).

The **Modify position** button in the **Production Window** is disabled until the motion pointer is set and a position is selected. To set the motion pointer, the program must be started and then stopped.

The maximum movement or change in orientation, may be restricted by the system parameters (topic *Controller*, type *ModPos Settings*) in the system design. Please read your cell or plant documentation for details.

If the system parameters are setup to use absolute limits for position changes, then the original positions can only be restored or changed using the baseline menu in HotEdit. The baseline concept is described in section [Tuning positions with HotEdit on page 253](#).

If a named position is modified, all other instructions using that position will be affected.

In the **Production Window**, circle points cannot be modified in synchronized mode. See *Application manual - MultiMove*.

Differences between Program Editor and Production Window

The procedure for modifying positions is the same in the **Program Editor** and the **Production Window**. However, there are differences in how positions are selected.

Continues on next page

Also, if your system uses MultiMove, then the result from the Program Editor and the Production Window will differ. See *Application manual - MultiMove*.

Program Editor selections

To select a position for modification in the Program Editor, tap the desired position.

Production Window selections

To select a position for modification in the Production Window you must step the program to the desired position.



Note

If you have executed the program from another window and then switch back to the Production Window, the selected position will be changed to the position where the motion pointer now is. Make sure the correct position is selected before making the modification!

Related information

For an overview on how to modify positions, see [Modifying and tuning positions on page 248](#).

HotEdit and baseline are described in [Tuning positions with HotEdit on page 253](#).

The HotEdit menu is also described in [HotEdit menu on page 34](#).

Modifying positions in the Program Data window is described in [Editing data instances on page 148](#).

Technical reference manual - RAPID Instructions, Functions and Data types.

Technical reference manual - System parameters.

Application manual - MultiMove

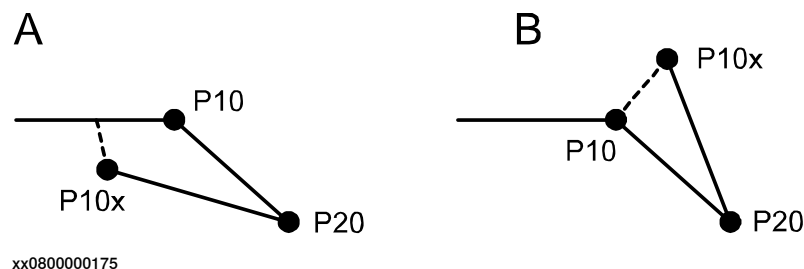
Examples planned path

The following examples show how the planned path will be effected when modifying positions.

Linear movement

In example A the robot is stopped on path before reaching the position P10. The robot is jogged off path to the new position (P10x) and the position P10 is modified.

In example B the robot is stopped on path in position P10. The robot is jogged off path to the new position (P10x) and the position P10 is modified.



Continues on next page

5 Running in production

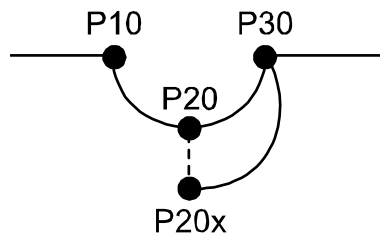
5.4.2 Modifying positions in the Program Editor or Production Window

Continued

In both examples, when restarting the program the robot continues from the new P10 (that is now the same as P10x) directly to P20 without returning to the previous planned path (via the old P10).

Circular movement

In this example the robot is stopped on path in position P20 (circle point) and then jogged to the new position P20x. The position P20 is modified.



xx080000176

In single robot systems or *MultiMove* systems in unsynchronized mode: When restarting the program the robot continues directly from the new P20 (that is now the same as P20x) to P30 without returning to the previous planned path (via the old P20). The new planned path from P20 (P20x) to P30 is calculated using these two positions and position P10.

In *MultiMove* synchronized mode: When restarting the program the robot returns to the old P20 and uses the previously planned path to P30. In the next cycle only the new P20 (P20x) is used.

5.4.3 Tuning positions with HotEdit

Overview

HotEdit is used to tune programmed positions. This can be done in all operating modes and even while the program is running. Both coordinates and orientation can be tuned.

HotEdit can only be used for named positions of the type robtarget (see limitations below).

The functions available in HotEdit may be restricted by the user authorization system, UAS.

The HotEdit menu is described in the section [HotEdit menu on page 34](#).

Applying tuned positions

Tuning values are used directly by an executing program when you tap **Apply**. If tuning is done close to the program or motion pointer it may be hard to predict exactly when it will take effect. It is therefore important that you know where in the program the robot is if applying offset values while the program is running.

However, the new values are not stored in the baseline until you use a **Commit** command.

How to tune positions

This is how you tune programmed positions using HotEdit:

	Action
1	In the Programmed targets window, select the positions to be tuned and add them to Selected targets by tapping the arrow.
2	Tap Tune Targets and select tuning mode (linear, reorient, or external axes) and then coordinate system (tool or work object).
3	Tap + and - to specify the exact tuning of the position(s) in x, y and z direction. Select Increment to define the step size of these buttons.
4	To activate the new values, tap APPLY . The offset will be used directly if the program is running.
5	If you are satisfied with the result and want the tuned positions to become part of the baseline, tap Baseline and then Commit Selection .
6	If, however, the selected targets need further tuning, you can tap Baseline and then Restore Selection and start all over again, or you can simply continue tuning until you are satisfied.

Working with selections

A selection of positions to be tuned later can be saved on the controller mass memory unit. If your system uses UAS, this may be the only way to select positions for tuning.

The commands for working with selections are located in the **File** menu:

Save Selection As	Make sure that the window Selected targets shows nothing but the positions to be saved. Tap File and Save Selection As . Enter the name and optionally a description of the file, then tap OK .
--------------------------	---

Continues on next page

5 Running in production

5.4.3 Tuning positions with HotEdit

Continued

Open Selection	Tap File and Open Selection . Then tap the selection you want to use and tap OK .
Clear Selection	Clear the Selected targets area by tapping File and Clear Selection .

Baseline concept

A baseline can be defined as a reference against which future changes are measured. The baseline concept makes it possible to undo tuning and revert to the position values stored in the latest baseline. To do this you use a **Restore** command.

When a **Commit** command is performed the baseline is updated with new offset values, and the old values no longer exist in program memory.

Use the baseline menu to apply or reject tuning.

- **Restore Selection** will discard all tuning of the currently selected positions and revert them to the values of the latest baseline, meaning that their offset values will be 0,0.
- **Restore Entire Program** will discard ALL tuning to programmed positions since the latest **Commit** command. This may include several HotEdit sessions for the same task. If the system uses *Absolute Limit ModPos* any **Modify Position** command from the Program Editor will also be undone.
- **Commit Selection** will apply the offset of the currently selected positions to the baseline.
- **Commit Entire Program** will apply ALL tuning to programmed positions. This may include several HotEdit sessions for the same task. If the system uses *Absolute Limit ModPos* it also includes **Modify Position** performed in the Program Editor.

Baseline target criteria

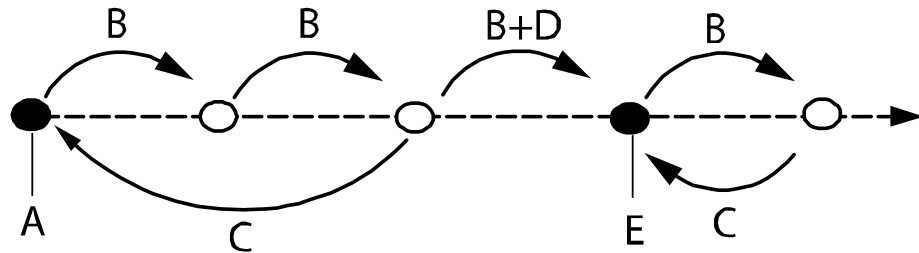
Targets that fulfil all of the following criteria are part of the baseline:

- The data type must be `robtarg` or `jointtarg`
- It must **not** be declared locally in a routine
- It must **not** be declared as part of an array of targets

Continues on next page

Illustration of baseline concept

The baseline concept is illustrated below, where a point is moved, restored and committed. Starting out from the original baseline (A), let us assume that you move the point (B) twice. If you regret the changes you perform a restore command (C). But if you instead continue moving the point and perform a commit command (B +D), you will have created a new baseline (E) and there is no way to revert to the original baseline. If you move the point one more time and then restore, the point is moved back to the latest baseline (E).



xx0600002620

A	Original baseline
B	Move selected point
C	Restore
D	Commit
E	New baseline

Restore Selection or Restore Entire Program

The following example shows the difference between **Restore Selection** and **Restore Entire Program** to original. The same idea applies for **Commit Selection** and **Commit Entire Program**.

Action	
1.	The robtargets p10 and p30 are added to Selected Targets and tuned once.
2.	p10 is removed from Selected Targets
3.	p30 is tuned again.
4.	<ul style="list-style-type: none"> Restore Selection sets the currently selected position, p30, to its value in the latest baseline. p10 is not affected, thus still tuned. Restore Entire Program sets all tuned positions, that is both p10 and p30 to their baseline values.

HotEdit for external axes

External axes can be tuned with HotEdit if they are activated in at least one of the selected robtargets. Only axes with active values are tuned.

Limitations

HotEdit tuning is only possible for named (e.g. p10, p20) robtargets. (* robtargets are not visible in the treeview.)

If a robtarget is declared as an array, it must be indexed with a number to be modified in HotEdit.

Continues on next page

5 Running in production

5.4.3 Tuning positions with HotEdit

Continued

It is only possible to perform HotEdit tuning on targets that are part of the baseline. Targets that are NOT part of the baseline will not be shown in the HotEdit treeview, as they cannot be selected for tuning. This means that a target declared locally in a routine, for example, will not be displayed.

HotEdit tuning is possible for robtargets only. (Jointtargets can only be tuned by using **Modify Position** in the **Program Editor**.) If the system uses *Absolute limit ModPos* these jointtargets are however part of the baseline and will be affected when **Restore Entire Program** and **Commit Entire Program** are used.



Note

For more information about *Absolute Limit ModPos*, see the *Technical reference manual - System parameters*, section *Topic Controller - Type ModPos Settings*.

Using UAS in HotEdit

The user authorization system can be used to restrict the Hot Edit functionality and only allow a user to edit pre-selected positions. These are loaded by tapping **File** and then **Open Selection**. The selected positions can then be tuned in the usual way.

Related information

Technical reference manual - System parameters.

5.4.4 Working with displacements and offsets

About displacements

Sometimes, the same path is to be performed at several places on the same object, or on several work pieces located next to each other. To avoid having to reprogram all positions each time a displacement coordinate system can be defined.

This coordinate system can also be used in conjunction with searches, to compensate for differences in the positions of the individual parts.

The displacement coordinate system is defined based on the object coordinate system.

The displacement coordinate system is described in section [Coordinate systems for jogging on page 101](#).

Select displacement method

Depending on how, when, and how often you want to use displacements, the best method may vary.

Moving a work object

Moving a work object is suitable when you do not need to move or displace the work object very often.

This is detailed in section [Defining the work object coordinate system on page 172](#).

Displace a work object

A work object consists of a user frame and a object frame. You can move one or both of these frames. If you move both frames, then the whole work object is moved. It can be useful to displace the object frame from the user frame for instance when using one fixture for several work objects. Then you can keep the user frame and displace the object frame for the work objects.

See procedure *How to define object frame* in section [Defining the work object coordinate system on page 172](#).

Displace and rotate a work object

You may want to displace and rotate the object frame from the user frame if the displacement is not in just x, y, and z.

To displace in x, y, and z, you can use the same method as above. To rotate the work object, follow the procedure in section [Editing the work object data on page 176](#).

About offsets

Sometimes it is easier to define a position as an offset from a given position. If, for example, you know the exact dimensions of a work object, it will only be necessary to jog to one position.

The offset is programmed with the displacement distance in x, y, and z direction, in relation to the work object. For instance:

```
MoveL Offs(p10, 100, 50, 0), v50...
```

Define the offset for the position with the following expressions:

- 1 Original position / starting point

Continues on next page

5 Running in production

5.4.4 Working with displacements and offsets

Continued

- 2 Displacement in x direction
- 3 Displacement in y direction
- 4 Displacement in z direction

Examples

This example shows the move instructions with offsets to move the robot in a square (clockwise), starting at p10, with a 100 mm displacement in x and y.

```
MoveL p10, v50...  
MoveL Offs(p10, 100, 0, 0), v50...  
MoveL Offs(p10, 100, 100, 0), v50...  
MoveL Offs(p10, 0, 100, 0), v50...  
MoveL p10, v50...
```

How to create position offsets

This procedure details how to change a position to become an offset position.

	Action	Information
1	In the Program Editor, tap to select the position argument to edit.	
2	Tap Edit and then Change Selected .	
3	Tap Functions and then Offs .	
4	Tap to select each expression, <EXP>, and then tap any of the desired available data or functions. You can also tap Edit to access more functions. Tap All to open the soft keyboard and edit all expressions at the same time, or tap Only Selected to edit one at a time with the soft keyboard.	You can use the filter to narrow down the available data. You can also change data type of the available data.
5	Tap OK to save changes.	

Related information

There are a number of functions in RAPID that may be useful. See *Technical reference manual - RAPID Instructions, Functions and Data types*, and *Technical reference manual - RAPID Overview*.

5.4.5 Moving the robot to a programmed position

Positions

A robot program usually contain programmed positions. The robot can move automatically to a programmed position using a function in the **Jogging** menu. The robot will move at 250 mm/s.



DANGER

When moving the robot automatically, the robot arm may move without warning. Make sure no personnel are in safeguarded space and that no objects are in the way between the current position and the programmed position.

Moving the robot to a programmed position

This procedure describes how to move a robot automatically to a programmed position.

	Action	Information
1	On the ABB menu, tap Jogging .	
2	Make sure the correct mechanical unit is selected and then tap Go To....	
3	Tap to select a programmed position.	If you have many programmed positions you can use a filter to narrow down the visible positions. See section Filtering data on page 72 .
4	Press and hold the three-position enabling device and then tap and hold the Go To button. The robot now moves directly from the current position to the programmed position. Make sure no objects are in the way.	

This page is intentionally left blank

6 Handling inputs and outputs, I/O

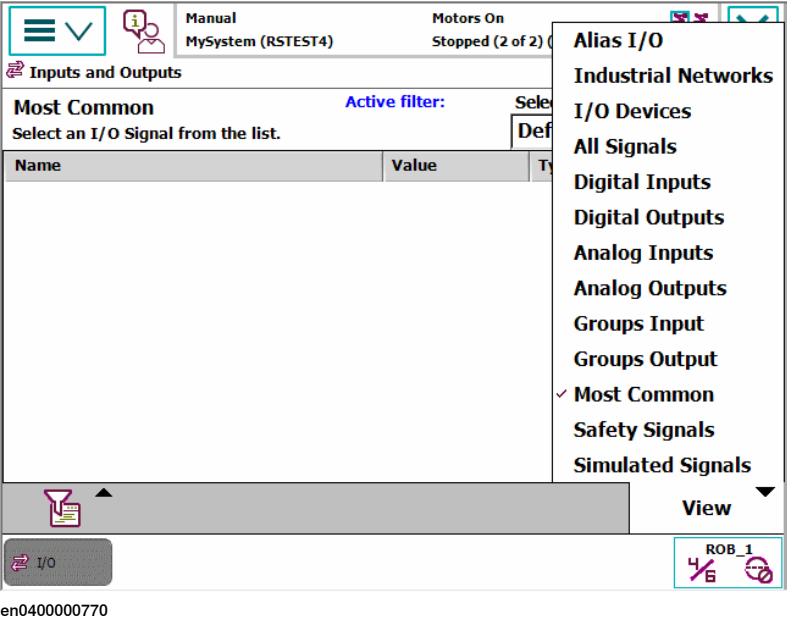
6.1 Viewing signal lists

Overview

I/O signal properties is used to view the input and output signals and their values. Signals are configured with system parameters .

How to view signal lists

This section details how to view a list of signals.

	Action
1	<p>On the ABB menu tap Inputs and Outputs. The list of Most Common I/O signals is displayed.</p> 
2	Tap View to change the selection of signals in the list.



Tip

Tap the **Select Layout** menu if you want to view signal labels in the list.

Related information

[Simulating and changing signal values on page 262.](#)

[Filtering data on page 72.](#)

[Configuring Most Common I/O on page 93.](#)

6 Handling inputs and outputs, I/O

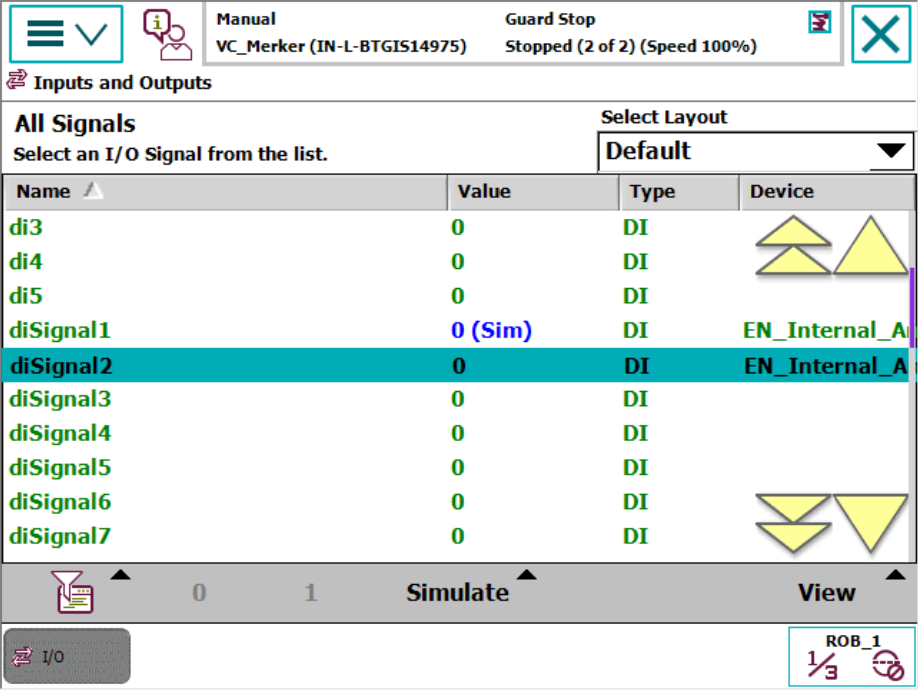
6.2 Simulating and changing signal values

6.2 Simulating and changing signal values


Simulating and changing signal values

A signal can be changed into a simulated signal. The value of the signal can also be changed.

Use the following procedure to simulate a signal or to change the value of a signal:

Action																																													
1	<p>On the ABB menu, tap Inputs and Outputs. A list of most common signals is displayed. See section Configuring Most Common I/O on page 93.</p>																																												
2	<p>Tap View and select a signal filter. The signal list is updated.</p>  <p>The screenshot shows the 'Inputs and Outputs' screen. At the top, there are status indicators: 'Manual VC_Merker (IN-L-BTGIS14975)' and 'Guard Stop Stopped (2 of 2) (Speed 100%)'. Below this is the 'Inputs and Outputs' title. The main section is titled 'All Signals' and includes a 'Select Layout' dropdown set to 'Default'. A table lists signals with columns for Name, Value, Type, and Device. The signal 'diSignal2' is selected and highlighted in blue, with a value of '0'. The 'Simulate' button is visible at the bottom of the list. The screen also shows a 'View' button and a 'ROB_1' indicator.</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> <th>Device</th> </tr> </thead> <tbody> <tr> <td>di3</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>di4</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>di5</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>diSignal1</td> <td>0 (Sim)</td> <td>DI</td> <td>EN_Internal_A</td> </tr> <tr> <td>diSignal2</td> <td>0</td> <td>DI</td> <td>EN_Internal_A</td> </tr> <tr> <td>diSignal3</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>diSignal4</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>diSignal5</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>diSignal6</td> <td>0</td> <td>DI</td> <td></td> </tr> <tr> <td>diSignal7</td> <td>0</td> <td>DI</td> <td></td> </tr> </tbody> </table> <p>Note Tap All Signals to view all the signals.</p>	Name	Value	Type	Device	di3	0	DI		di4	0	DI		di5	0	DI		diSignal1	0 (Sim)	DI	EN_Internal_A	diSignal2	0	DI	EN_Internal_A	diSignal3	0	DI		diSignal4	0	DI		diSignal5	0	DI		diSignal6	0	DI		diSignal7	0	DI	
Name	Value	Type	Device																																										
di3	0	DI																																											
di4	0	DI																																											
di5	0	DI																																											
diSignal1	0 (Sim)	DI	EN_Internal_A																																										
diSignal2	0	DI	EN_Internal_A																																										
diSignal3	0	DI																																											
diSignal4	0	DI																																											
diSignal5	0	DI																																											
diSignal6	0	DI																																											
diSignal7	0	DI																																											
3	<p>Tap on a signal.</p>																																												
4	<p>Tap on Simulate to change the signal into a simulated signal. Tap on Remove Simulation to remove the simulation from the signal.</p> <p>Note When you remove the simulation from a signal, the signal value returns to the same value it had before it was simulated. Also the cross connections are updated accordingly.</p>																																												

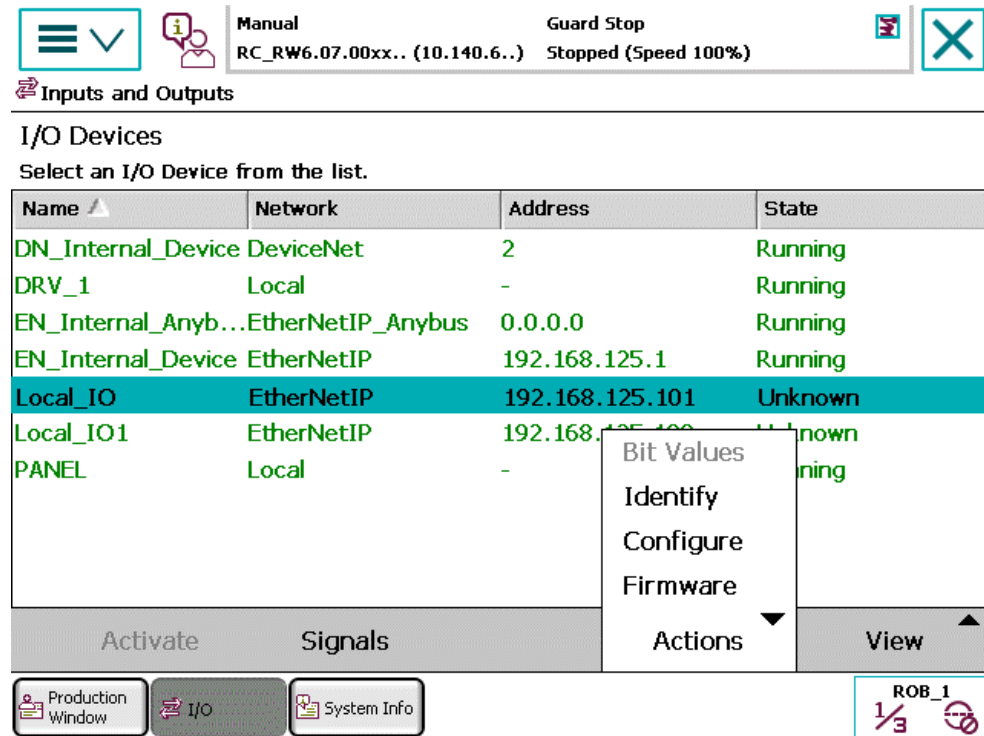
Continues on next page

Action	
5	<p>For a digital signal, tap 0 or 1 to change the value of a signal. For analog signals and groups, tap on 123... to change the signal value. The soft numeric keyboard is displayed. Enter the new value and tap OK.</p> <p> Note</p> <p>If there is a cross connection created between two signals, changing the value of one signal automatically changes the value of the corresponding cross connected signal.</p>

For more information about changing the signal properties see the section Control Panel, [Configuring Most Common I/O on page 93](#).

Managing I/O devices

You can manage I/O devices from the I/O Devices page.



I/O Devices
Select an I/O Device from the list.

Name ▲	Network	Address	State
DN_Internal_Device	DeviceNet	2	Running
DRV_1	Local	-	Running
EN_Internal_Anyb...	EtherNetIP_Anybus	0.0.0.0	Running
EN_Internal_Device	EtherNetIP	192.168.125.1	Running
Local_IO	EtherNetIP	192.168.125.101	Unknown
Local_IO1	EtherNetIP	192.168.125.100	Unknown
PANEL	Local	-	Running


Bit Values
Identify
Configure
Firmware
Actions

Activate Signals Actions View

Production Window I/O System Info ROB_1 1/3

xx1700002253

Following are the actions that you can perform from I/O Devices page:



Action	Output
Activate/Deactivate	Activates or deactivates a selected I/O device.
Signals	Displays the signals associated with the selected I/O device.
Actions - Bit Values	Displays the input and output bit values of the selected device.
Actions - Identify	Displays the MAC ID of the selected device.
	<p> Note</p> <p>Actions - Identify option is available only for the network type EtherNet IP.</p>

Continues on next page

6 Handling inputs and outputs, I/O


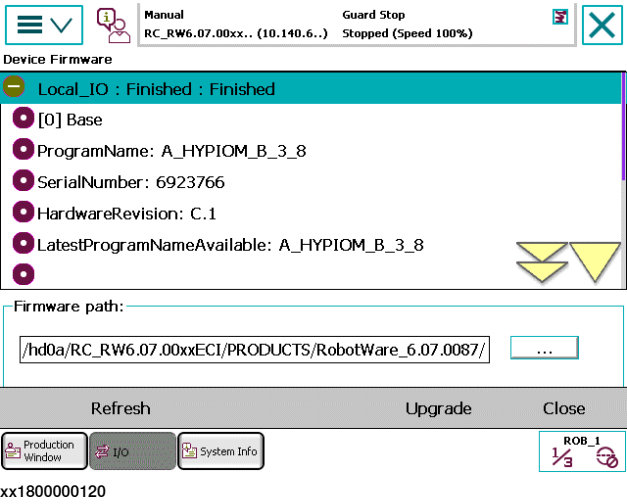
6.2 Simulating and changing signal values

Continued

Action	Output
Actions - Configure	<p>Allows you to configure the selected local I/O device.</p> <p> Note</p> <p>Actions - Configure option is available only for the network type EtherNet IP.</p>
Actions - Firmware	<p>Allows you to upgrade the firmware.</p> <p> Note</p> <p>Actions - Firmware option is available only for the network type EtherNet IP.</p>

Upgrading firmware

You can upgrade the firmware of I/O device of type EtherNet IP. Use the following procedure to upgrade the firmware.

	Action	Description
1	Tap Main > Inputs and Outputs.	The Input & Output window is displayed.
2	Tap View > I/O Devices.	The I/O Devices window is displayed. It displays the available I/O devices.
3	Select the EtherNet IP I/O device.	<p> Note</p> <p>The Firmware upgrade option is available only for the network type EtherNet IP.</p>
4	Tap Actions > Firmware.	<p>The Device Firmware window is displayed with the details of the firmware.</p> 
5	Tap Upgrade.	The firmware for the selected EtherNet IP I/O device is upgraded.

6.3 Viewing signal group

Viewing signal group

This section details how to view signal groups.

	Action
1	On the ABB menu, tap I/O . A list of most common signals is displayed. See section Configuring Most Common I/O on page 93 .
2	In the View menu, tap Groups .
3	Tap on the signal group's name in the list and then tap Properties . Or tap twice on the signal group's name. The signal group's properties is displayed.

6 Handling inputs and outputs, I/O

6.4.1 Safety I/O signals

6.4 Safety signals

6.4.1 Safety I/O signals

General

In the IRC5 controller's basic and standard form, certain I/O signals are dedicated to specific safety functions. These are listed below with a brief description of each. All signals can be viewed in the I/O menu on the FlexPendant.

Safety I/O signals

The list below contains the safety I/O signals as used by the standard system.

Signal name	Description	Bit value condition	From - To
ES1	Emergency stop, chain 1	1 = Chain closed	From panel board to main computer
ES2	Emergency stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTESI	Soft Emergency stop	1 = Soft stop enabled	From panel board to main computer
EN1	Enabling device1&2, chain 1	1 = Enabled	From panel board to main computer
EN2	Enabling device1&2, chain 2	1 = Enabled	From panel board to main computer
AUTO1	Op mode selector, chain 1	1 = Auto selected	From panel board to main computer
AUTO2	Op mode selector, chain 2	1 = Auto selected	From panel board to main computer
MAN1	Op mode selector, chain 1	1 = MAN selected	From panel board to main computer
MANFS1	Op mode selector, chain 1	1 = Man. full speed selected	From panel board to main computer
MAN2	Op mode selector, chain 2	1 = MAN selected	From panel board to main computer
MANFS2	Op mode selector, chain 2	1 = Man. full speed selected	From panel board to main computer
USERDOOVL	Over load, user DO	1 = Error, 0 = OK	From panel board to main computer
MONPB	Motors-on pushbutton	1 = Pushbutton pressed	From panel board to main computer
AS1	Auto stop, chain 1	1 = Chain closed	From panel board to main computer
AS2	Auto stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTASI	Soft Auto stop	1 = Soft stop enabled	From panel board to main computer
GS1	General stop, chain 1	1 = Chain closed	From panel board to main computer

Continues on next page

Signal name	Description	Bit value condition	From - To
GS2	General stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTGSI	Soft General stop	1 = Soft stop enabled	From panel board to main computer
SS1	Superior stop, chain1	1 = Chain closed	From panel board to main computer
SS2	Superior stop, chain2	1 = Chain closed	From panel board to main computer
SOFTSSI	Soft Superior stop	1 = Soft stop enabled	From panel board to main computer
CH1	All switches in run chain 1 closed	1 = Chain closed	From panel board to main computer
CH2	All switches in run chain 2 closed	1 = Chain closed	From panel board to main computer
ENABLE1	Enable from MC (read back)	1 = Enable, 0 = break chain 1	From panel board to main computer
ENABLE2_1	Enable from AXC1	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_2	Enable from AXC2	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_3	Enable from AXC3	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_4	Enable from AXC4	1 = Enable, 0 = break chain 2	From panel board to main computer
PANEL24OVL	Overload, panel 24V	1 = Error, 0 = OK	From panel board to main computer
DRVOVL	Overload, drive modules	1 = Error, 0 = OK	From panel board to main computer
DRV1LIM1	Read back of chain 1 after limit switches	1 = Chain 1 closed	From axis computer to main computer
DRV1LIM2	Read back of chain 2 after limit switches	1 = Chain 2 closed	From axis computer to main computer
DRV1K1	Read back of contactor K1, chain 1	1 = K1 closed	From axis computer to main computer
DRV1K2	Read back of contactor K2, chain 2	1 = K2 closed	From axis computer to main computer
DRV1EXTCONT	External contactors closed	1 = Contactors closed	From axis computer to main computer
DRV1TEST1	A dip in run chain 1 has been detected	Toggled	From axis computer to main computer
DRV1TEST2	A dip in run chain 2 has been detected	Toggled	From axis computer to main computer
SOFTESO	Soft Emergency stop	1 = Set soft E-stop	From main computer to panel board
SOFTASO	Soft Auto stop	1 = Set soft Auto stop	From main computer to panel board
SOFTGSO	Soft General stop	1 = Set soft General stop	From main computer to panel board

Continues on next page

6 Handling inputs and outputs, I/O

6.4.1 Safety I/O signals

Continued

Signal name	Description	Bit value condition	From - To
SOFTSSO	Soft Superior stop	1 = Set soft Sup. E-stop	From main computer to panel board
MOTLMP	Motors-on lamp	1 = Lamp on	From main computer to panel board
TESTEN1	Test of Enable1	1 = Start test	From main computer to panel board
DRV1CHAIN1	Signal to interlocking circuit	1 = Close chain 1	From main computer to axis computer 1
DRV1CHAIN2	Signal to interlocking circuit	1 = Close chain 2	From main computer to axis computer 1
DRV1BRAKE	Signal to brake-release coil	1 = Release brake	From main computer to axis computer 1

7 Handling the event log

7.1 Accessing the event log

Event log

Open the event log to:

- view all present entries.
- study specific entries in detail.
- handle the log entries, such as saving or deleting.

The log can be printed by using RobotStudio.

Open and close the event log

This section details how to open the event log.

	Action
1	Tap the status bar. The status window is displayed.
2	Tap Event Log . The event log list is displayed.
3	If the log contents do not fit into a single screen, it can be scrolled.
4	Tap a log entry to view the event message.
5	Tap the status bar again to close the log.

Related information

Operating manual - RobotStudio.

7 Handling the event log

7.2 Deleting log entries

7.2 Deleting log entries

Why should I delete log entries?

Logs can be deleted to increase available disk space. Deleting log entries is often a good way to trace faults since you remove old and insignificant log entries not related to the problem you are trying to solve.

Delete all log entries

	Action
1	Tap the status bar, then the Event Log tab to open the event log.
2	On the View menu, tap Common .
3	Tap Delete and then Delete all logs . A confirmation dialog is displayed.
4	Tap Yes to delete, or No to keep the log intact.

Delete log entries of a specific category

	Action
1	Tap the status bar, then the Event Log tab to open the event log.
2	On the View menu, tap the category of choice.
3	Tap Delete and then Delete log . A confirmation dialog is displayed.
4	Tap Yes to delete, or No to keep the log intact.

7.3 Saving log entries

Why should I save log entries?

You should save log entries when:

- you need to clear the log but want to keep the current entries to be viewed later.
- you want to send log entries to support to solve a problem.
- you want to keep log entries for future reference.



Note

The log can keep up to 20 entries per category and up to 1000 entries in the all events list. When the buffer is full the oldest entries will be overwritten and lost. There is no way to retrieve these lost log entries.

Save all log entries

This section details how to save all log entries.

	Action
1	Tap the status bar to open the event log.
2	Tap Save all logs as . The file dialog is displayed.
3	If you want to save the log in a different folder, locate and open the folder.
4	In the File name box, type a name for the file.
5	Tap Save .

This page is intentionally left blank

8 Backup and restore

8.1 Back up the system

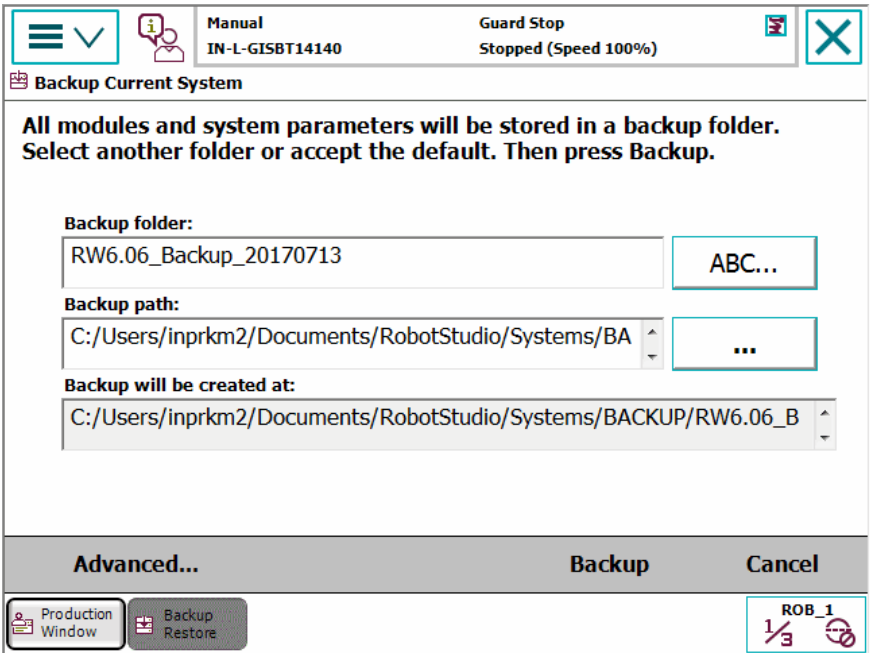
When do I need this?

We recommend performing a backup:

- *Before* installing new RobotWare.
- *Before* making any major changes to instructions and/or parameters to make it possible to return to the previous setting.
- *After* making any major changes to instructions and/or parameters and testing the new settings to retain the new successful setting.

Back up the system

This section describes how to back up the system.


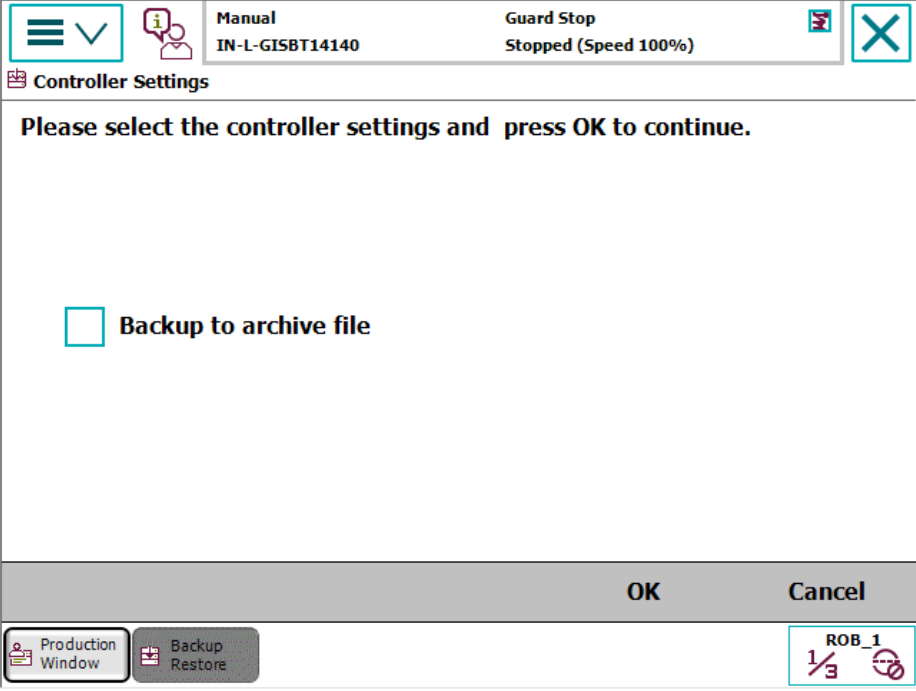
	Action
1	Tap the ABB menu and then tap Backup and Restore .
2	<p>Tap Backup Current System. The Backup Current System window is displayed. If a default path has been defined as detailed in the section Setting default paths on page 79, this is displayed.</p>  <p>xx030000441</p> <p>Note</p> <ul style="list-style-type: none"> • By default, a name for the Backup folder is created which can be renamed by the user later. • While renaming, ensure that the name does not start with a space. • If the folder name starts with a space, a warning dialog appears.

Continues on next page

8 Backup and restore

8.1 Back up the system

Continued

	Action
3	<p>Tap Advanced... The Controller Settings window is displayed.</p> <p> Note</p> <p>Use this step to create the backup as a zip file. If this is not required navigate directly to the last step.</p>  <p>xx1700001302</p>
4	Select the Backup to archive file checkbox.
5	Tap OK .
6	<p>Is the displayed backup path the correct one? If YES: Tap Backup to perform the backup to the selected directory. A backup file named according to the current date is created. If NO: Tap ... to the right of the backup path and select directory. Then tap Backup. A backup folder named according to the current date is created.</p>

Disable or queue backup

Backing up the system during production can interfere with the RAPID execution. To avoid that a backup is taken during critical process steps or sensitive robot movements, a system input (*Disable Backup*, type *System Input*) can be set during these critical steps. When the critical steps are done, the input should be reset to allow backups again.

If needed, the backup can be queued while *Disable Backup* is set, using the system parameter *General RAPID*, with action value *QueueBackup* set to *TRUE*. Then the backup will be queued until the signal is reset.

Disable Backup and *QueueBackup* are described in *Technical reference manual - System parameters*.

8.2 Important when performing backups

BACKUP directory

A local default backup directory, `BACKUP`, is automatically created by the system. We recommend using this directory for saving backups.

Such backups are not copied to the directory `HOME` in following backups.

Never change the name of the `BACKUP` directory.

Never change the name of the actual backup to `BACKUP`, since this will cause interference with this directory.

A default path can be created to any location on the network where the backup should be stored, see [Setting default paths on page 79](#).

When is backup possible?

A backup of a system can be performed during program execution, with a few limitations:

- Start program, load program, load module, close program, and erase module cannot be done during backup in executing state. The RAPID instructions `Load` and `StartLoad` can, however, be used.
- Do not create backups while performing critical process steps or sensitive robot movements. This may affect the accuracy and performance of the movement. To make sure that no backup is requested, use a system input with the action value `Disable Backup` (type *System Input*). When the critical steps are done, the input should be reset to allow backups again.

If needed, the backup can be queued while `Disable Backup` is set, using the system parameter *General RAPID*, with action value `QueueBackup` set to `TRUE`. Then the backup will be queued until the signal is reset.

(Queueing functionality available from RobotWare 6.11.)

`Disable Backup` and `QueueBackup` are described in *Technical reference manual - System parameters*.

The system input signal can be set from RAPID for the parts of the code that are critical for disturbances.

What happens during backup?

During the backup process, background tasks continue to execute.

Duplicated modules?

No save operation is performed in the backup command. This implies that two revisions of the same module can exist in the backup, one from the program memory saved in `Rapid\Task\Progmod\` directory and one from the `HOME` directory copied to the backup's home directory. Restoring such a backup will restore both revisions of the module, so the status remains unchanged.

Continues on next page

8 Backup and restore

8.2 Important when performing backups

Continued

Large data amount

Since the HOME directory is included in the backup, large files contained in this folder will make the backup larger. To avoid this situation, you should either clean the HOME directory on regular basis removing the unnecessary files, or keep large files in some other location.

Faults during backup

If a fault occurs during the backup, for example full disk or power failure, the whole current backup is deleted to make sure that only valid fully saved backups are present on the disk.

8.3 Restore the system

When do I need this?

We recommend performing a restore:

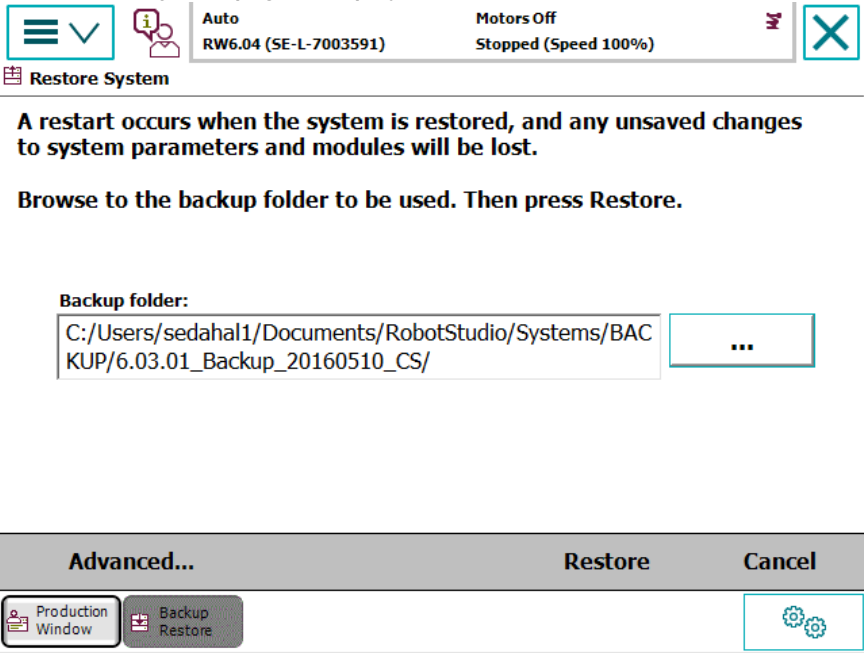
- If you suspect that the program file is corrupt.
- If any changes made to the instructions and/or parameters settings did not prove successful, and you want to return to the previous settings.

During the restore, all system parameters are replaced and all the modules from the backup directory are loaded.

The Home directory is copied back to the new system's HOME directory during the restart.

Restore the system

This section describes how to restore the system.

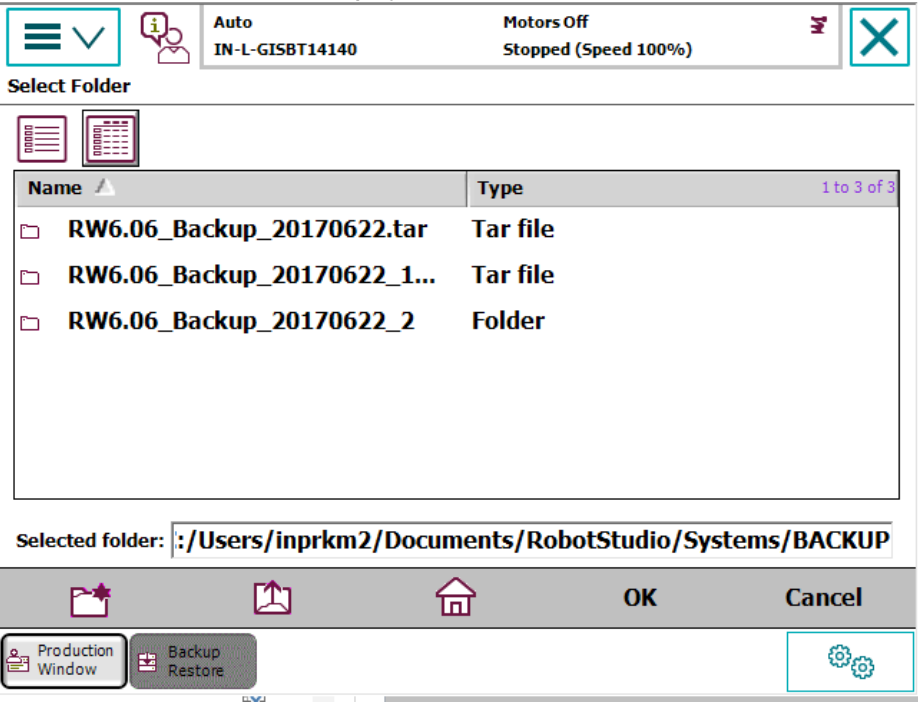
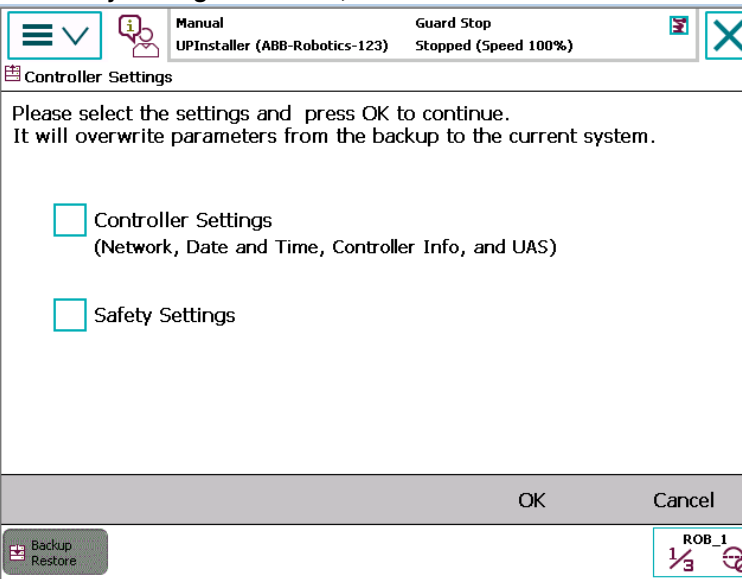
	Action
1	On the ABB menu, tap Backup and Restore .
2	<p>Tap Restore System. The Restore System page is displayed.</p>  <p>A restart occurs when the system is restored, and any unsaved changes to system parameters and modules will be lost.</p> <p>Browse to the backup folder to be used. Then press Restore.</p> <p>Backup folder: C:/Users/sedah1/Documents/RobotStudio/Systems/BAC KUP/6.03.01_Backup_20160510_CS/</p> <p>Advanced... Restore Cancel</p> <p>Production Window Backup Restore</p> <p>xx0300000442</p> <p>Note</p> <p>If a default path has been defined as explained in the section Setting default paths on page 79, that path is displayed in the Backup folder path.</p>

Continues on next page

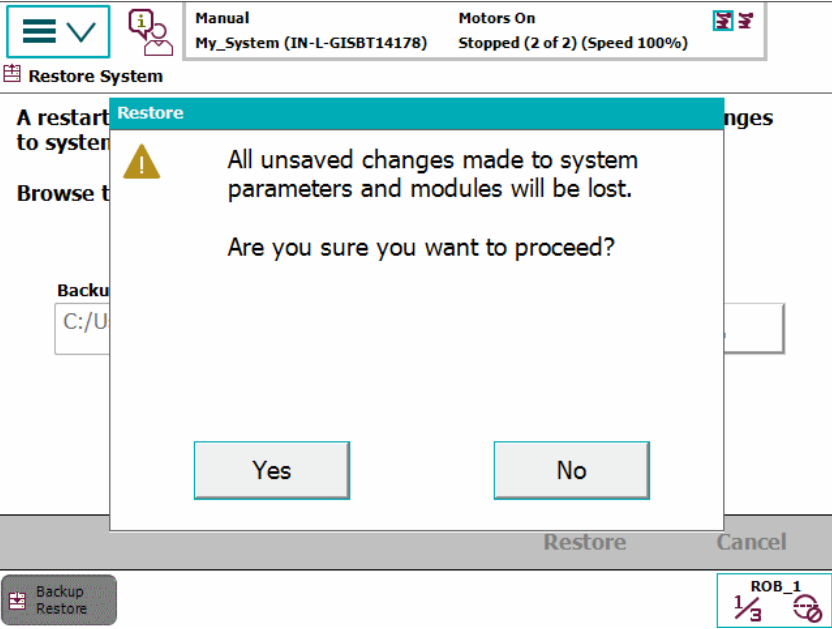
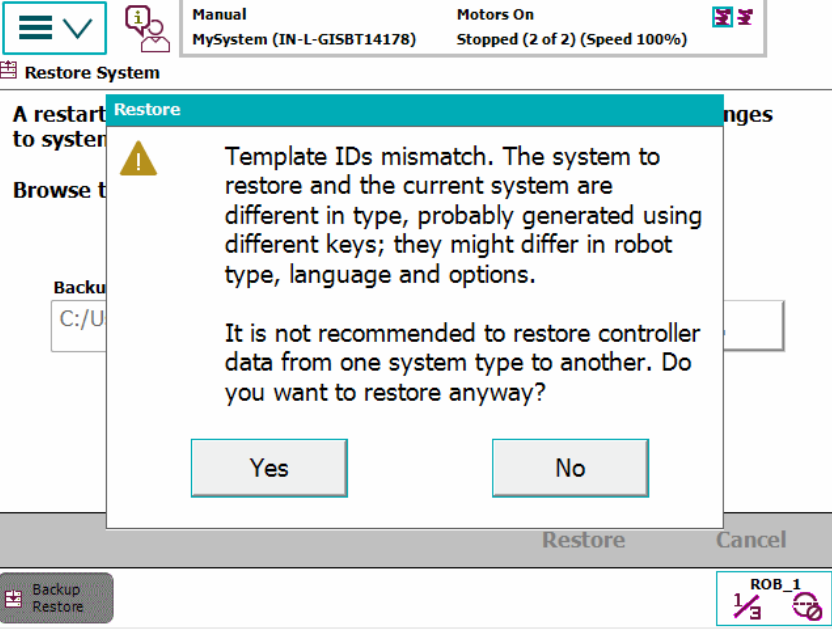
8 Backup and restore

8.3 Restore the system

Continued

Action	
3	<p>To select the backup folder path or to select the TAR backup file format, tap The Select Folder window is displayed.</p>  <p>Selected folder: <code>:/Users/inprkm2/Documents/RobotStudio/Systems/BACKUP</code></p> <p>xx1700001303</p>
4	<p>Select the backup folder or select the backup file which is in TAR format.</p>
5	<p>Tap OK. The Restore System page is displayed again with the selected backup folder or file path details. If you want to replace the current controller and safety settings with the selected backup controller and safety settings, click Advanced..., select the Controller Settings and Safety Settings check box, and click OK.</p>  <p>Please select the settings and press OK to continue. It will overwrite parameters from the backup to the current system.</p> <p><input type="checkbox"/> Controller Settings (Network, Date and Time, Controller Info, and UAS)</p> <p><input type="checkbox"/> Safety Settings</p> <p>xx1600001082</p>

Continues on next page

Action	
6	<p>Tap Restore. The following screen is displayed.</p>  <p>xx1400002325</p>
7	<p>Tap Yes. The restore is performed, and the system is restarted.</p> <p>Note</p> <p>If there is a mismatch between the backup and the current system, the following warning message is displayed.</p>  <p>xx1400002326</p>

This page is intentionally left blank

9 Calibrating

9.1 How to check if the robot needs calibration

Check robot calibration status

This section describes how to check the robot's calibration status.

	Action
1	On the ABB menu, tap Calibration .
2	In the list of mechanical units, check the calibration status.

What kind of calibration is needed?

If the calibration status is...	then...
Not calibrated	the robot must be calibrated by a qualified service technician.
Rev. counter update needed	You must update the revolution counters. How to update the revolution counters is described in the product manual for the robot.
Calibrated	No calibration is needed.



DANGER

Do not attempt to perform the fine calibration procedure without the proper training and tools. Doing so may result in incorrect positioning that may cause injuries and property damage.

9 Calibrating

9.2 Updating revolution counters

9.2 Updating revolution counters

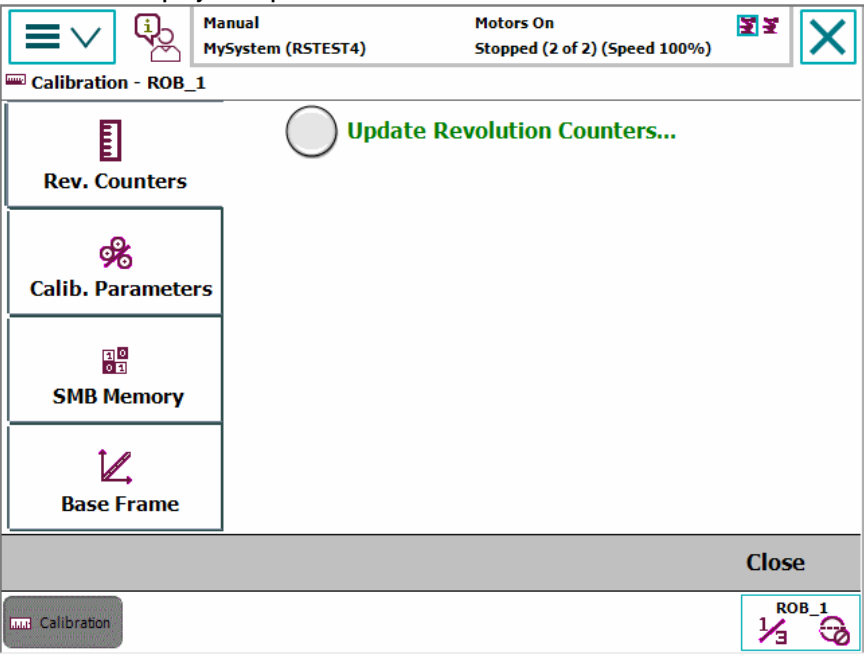
Overview

This section details how to perform a rough calibration of each robot axis, that is updating the revolution counter value for each axis, using the FlexPendant. Detailed information about revolution counters and how to update them, with calibration positions and scales, can be found in the respective robot product manual. Also, see the manual *Operating manual - Calibration Pendulum* for information on calibration.



For robots using the *Absolute Accuracy* option, the calibration data file *absacc.cfg* must be loaded first.

Storing the revolution counter setting

This procedure details the second step when updating the revolution counter; storing the revolution counter setting.

	Action
1	On the ABB menu, tap Calibration . All mechanical units connected to the system are shown along with their calibration status.
2	Tap the mechanical unit in question. A screen is displayed: tap Rev. Counters . 

Continues on next page

	Action
3	<p>Tap Update Revolution Counters. A dialog box is displayed, warning that updating the revolution counters may change programmed robot positions:</p> <ul style="list-style-type: none"> • Tap Yes to update the revolution counters. • Tap No to cancel updating the revolution counters. <p>Tapping Yes displays the axis selection window.</p> <p> Note</p> <p>When updating the revolution counters, the ongoing RAPID instruction or function is interrupted, and the path is cleared.</p>
4	<p>Select the axis to have its revolution counter updated by:</p> <ul style="list-style-type: none"> • Ticking in the box to the left • Tapping Select all to update all axes. <p>Then tap Update.</p>
5	<p>A dialog box is displayed, warning that the updating operation cannot be undone:</p> <ul style="list-style-type: none"> • Tap Update to proceed with updating the revolution counters. • Tap Cancel to cancel updating the revolution counters. <p>Tapping Update updates the selected revolution counters and removes the tick from the list of axes.</p>
6	<p> CAUTION</p> <p>If a revolution counter is incorrectly updated, it will cause incorrect manipulator positioning, which in turn may cause damage or injury!</p> <p>Check the calibration position very carefully after each update.</p> <p>See section <i>Checking the calibration position</i> in either of the calibration manuals, depending on which calibration method to be used. The Product manual for the robot also contains more information about calibration.</p>

Related information

Operating manual - Calibration Pendulum

This page is intentionally left blank

Index

A

ABB menu, 22
 approach points, 160
 array
 modify positions, 151
 automatic mode
 limitations, 185
 switching to, 244
 axes
 illustration, 102
 Axis Calibration
 service routine, 201

B

background
 changing, 81
 backup
 default file path, 79
 directory, 275
 important, 275
 menu, 45
 system, 273
 backward button, 19
 backward execution
 about, 193
 limitations, 193
 base coordinates
 default settings, 101
 selecting, 116
 baseline
 concept, 254
 target criteria, 254
 battery shutdown
 service routine, 200
 buttons
 controller, 30

C

cabinets, 16
 calculation result, 162
 calender time counter, 203
 calibrating
 Axis Calibration, 201
 CalPendulum, 202
 LoadIdentify, 204
 calibration, 28
 status, 281
 touchscreen, 97
 calibration menu, 46
 CalPendulum
 service routine, 202
 characters
 entering, 68
 international, 68
 close button, 22
 connection
 FlexPendant, 234
 connector, 18
 controller
 buttons, 30
 ports, 30
 single cabinet, 16
 control tools, overview, 27
 coordinate systems

 default settings, 101
 overview, 101
 Quickset, 61
 selecting, 61, 116

cursor
 about, 144

D

data instance, 41, 146
 data types
 creating new, 146
 editing, 148
 menu, 41
 viewing, 145
 date and time, 91
 default paths
 setting, 79
 disable backup, 274
 displacements
 about, 257
 work object, 172

E

elongator points
 define, 162
 emergency stop button
 FlexPendant, 18
 enabling device, 18, 20
 using, 188
 error messages, 70
 event log
 message, 50
 expressions
 offs, 257
 positions, 257

F

fieldbuses, 37
 files
 programs, 128
 filtering
 about, 72
 data types, 72
 files, 72
 programs, 72
 FlexPendant
 connecting, 234
 connecting in operation, 236
 connector, 30
 disconnecting, 236
 hardware buttons, 19
 hot plug, 236
 how to hold, 21, 89
 jumper plug, 236
 left-hander, 21
 main parts, 18
 overview, 17
 rotating, 90
 screen, 22
 FlexPendant Explorer, 36
 forward button, 19

H

hard buttons, 19
 hold-to-run, 20, 228
 using, 188, 190
 HotEdit, 34, 249

using, 253
hot plug, 236

I

I/O

about, 37
changing values, 262
menu, 37
most common, 93
safety signals, 266
simulating, 262
viewing groups, 265

I/O, inputs and outputs, 261

incremental movement

definition, 119
quickset, 63
setting size, 119
size settings, 63, 120

insertion point, change, 69

instances

data types, 146

instructions

backward execution, 193
changing motion mode, 142
commenting, 142
copying and pasting, 142
copying arguments, 142
cutting, 142
editing arguments, 140
handling of, 139
running from a specific, 191
undo, redo, 139

international characters, 68

IRC5 controller, 16

overview, 16

IsBrakeCheckActive, 217

J

jogging

about, 99
additional axes, 108
axes in independent mode, 108
coordinated, 109
coordinate systems, 116
non calibrated mechanical units, 108
restrictions, 108
world zones, 108

jogging menu, 38

joystick, 18

locking directions, 117
using, 19

joystick directions

about, 107
illustration, 102
locking, 117

jumper plug, 236

L

left-handed, 89

linear mode

default settings, 101
linear motion mode, 107

LoadIdentify

service routine, 204

loads

identifying, 204

log off, 77

log on, 77

M

main switch

controller, 30

ManLoadIdentify

service routine, 215

manual full speed mode

about, 187
switching to, 247

manual mode

about, 187
limitations, 187
switching to, 246

mechanical unit

quickset, 58
selecting, 58, 110

mechanical units

activate automatically, 44

modifying positions, 249

arrays, 151
data instances, 148
overview, 248

ModPos, 249

Absolute Limit ModPos, 248

modules

creating, 131
deleting, 133
handling of, 131
loading, 132
renaming, 133
saving, 132

most common I/O

configuring, 93

motion mode

Quickset, 59
selecting, 59, 112

motion pointer, MP, 194

Motion Pointer, MP

about, 144

multitasking programs

about, 229
loading, running and stopping, 229
viewing, 230

N

network settings, 91

O

offsets

about, 257
creating, 258
description, 257

operation time counter, 203

operator window, 22, 55

options

Calibration Pendulum, 46
duty time counter, 30
hot plug button, 30, 236
IRC5 Compact, 16
IRC5 Panel Mounted Controller, 16
Levelmeter Calibration, 46
MultiMove, 109
safety chain LEDs, 30
service outlet, 30
service port, 30

P

- path
 - returning to, 240
- path return region, 240
- payloads
 - creating, 179
 - declarations, 180
 - deleting, 184
 - display definitions, 181
 - editing, 181
 - editing declarations, 183
 - identifying, 204
 - selecting, 113
- ports
 - controller, 30
- positions
 - about, 121
 - exact, 121
 - HotEdit, 249
 - modifying, 248–249
 - moving to, 259
 - naming rules, 84
 - offset, 257
 - reading, 121
 - tuning, 34, 248–249, 253
- program a robot, 28
- program data
 - editing, 148
 - menu, 41
- program directory, 128
- program execution start button, 19
- programmable buttons
 - editing, 95
- programmable keys
 - editing, 95
- program pointer, PP, 194
- Program Pointer, PP
 - about, 144
- programs
 - about files, 128
 - creating, 128
 - default file path, 79
 - handling of, 128
 - loading, 129
 - multitasking, 229
 - renaming, 130
 - saving, 129
 - starting, 225
 - step by step execution, 193
 - stopping, 228

Q

- queue backup, 274
- quickset
 - increments, 63
 - mechanical unit, 58
- Quickset
 - coordinate systems, 61
 - motion mode, 59
 - run mode, 64
 - speed mode, 66
 - step mode, 65
 - tasks, 67
 - tools, 60
 - work objects, 60
- quickset menu, 22

R

- RAPID, 126
- redo
 - instructions, 139
- reorient mode
 - default settings, 101
- reorient motion mode, 107
- reset button
 - location, 18
 - using, 19
- resolvers
 - about, 121
- restart
 - menu, 53
- restore
 - default file path, 79
 - menu, 45
 - system, 277
- revolution counters
 - about, 121
 - battery shutdown, 200
 - setting, 282
 - updating, 282
- RobotStudio
 - overview, 26
- RobotStudio Online Apps, 24
- RobotStudio Online Apps
 - Calibrate, 24
 - Jog, 24
 - Manage, 24
 - Operate, 25
 - Tune, 25
 - YuMi, 25
- routines
 - changing declarations, 137
 - copying, 137
 - creating, 134
 - defining parameters, 135
 - deleting, 137
 - handling of, 134
 - running a specific, 192
 - running service routines, 196
- run button, 19
- run mode
 - quickset, 64
 - setting, 64

S

- safety I/O signals, 266
- scrolling, 71
- service port, 30
- service routine, 241
- service routines
 - Axis Calibration, 201
 - BatteryShutDown, 200
 - CalPendulum, 202
 - LoadIdentify, 204
 - ManLoadIdentify, 215
 - running, 196
 - ServiceInfo, 203
- signals
 - changing values, 262
 - simulating, 262
 - viewing, 261
- SIS, Service Information System
 - counters, 203
 - service routine, 203

SMB

- battery shutdown, 200
- soft keyboard, 68
- speed mode
 - Quickset, 66
 - setting, 66
- start button, 19
- status bar, 22, 56
- step backward button, 19
- step by step execution, 193
- step forward button, 19
- step mode
 - Quickset, 65
 - setting, 65
- stop button, 19
- stylus pen
 - location, 18
 - using, 19
- system
 - backup, 273
 - restore, 277
- system parameters, 28

T

- targets
 - modifying, 248–249
 - moving to, 259
 - naming rules, 84
 - tuning, 34, 248–249, 253
- task bar, 22
- tasks
 - debugging, 86
 - loading program to, 230
 - normal, static, semistatic, 229
 - setting up, 229
 - starting and stopping, 229
 - tasks panel, 67, 86
- three-position enabling device, 18, 20
 - using, 188
- thumb button
 - using, 20
- tool, overview control tools, 27
- tool center point
 - about, 156
 - calculation result, 162
 - define, 161
 - defining, 162
 - measuring, 164
 - TCP, 156
 - working area variations, 162
- tool coordinates
 - default settings, 101
 - selecting, 116
- tool frame
 - defining, 159
 - methods, 159
 - reorientation test, 162
- tool orientation, 162
 - setting, 114
- tool orientation, definition, 114
- tools
 - aligning, 123

- creating, 156
- deleting, 167
- editing declarations, 166
- editing definitions, 164
- editing tool data, 163
- identifying loads, 204
- make stationary, 168
- Quickset, 60
- selecting, 60, 113
- setting up tool coordinate system, 169
- stationary, 168
- Total Load, 179
- touchscreen, 22
 - background image, 81
 - brightness, 88
 - calibrating, 97
- touch screen
 - rotating, 90
- troubleshooting, 239
- tuning
 - HotEdit, 249
 - positions, 248–249, 253
 - targets, 248–249

U

- UAS
 - configuring views, 82
- uncalibrated mechanical unit, 241
- undo
 - instructions, 139
- USB port
 - FlexPendant, 18

V

- viewing messages in programs, 55
- view settings
 - additional test views, 83
 - configuring, 80

W

- work object coordinates
 - selecting, 116
- work objects
 - creating, 171
 - declarations, 171
 - defining coordinate system, 172
 - deleting, 178
 - displacements, 172
 - editing declarations, 177
 - editing work objects data, 176
 - Quickset, 60
 - selecting, 60, 113
- world coordinates
 - selecting, 116
- wrist optimization, 223
- write access
 - granting, 76
 - message, 70
 - rejecting, 76

Z

- zooming, 71



ABB AB

Robotics & Discrete Automation

S-721 68 VÄSTERÅS, Sweden

Telephone +46 (0) 21 344 400

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