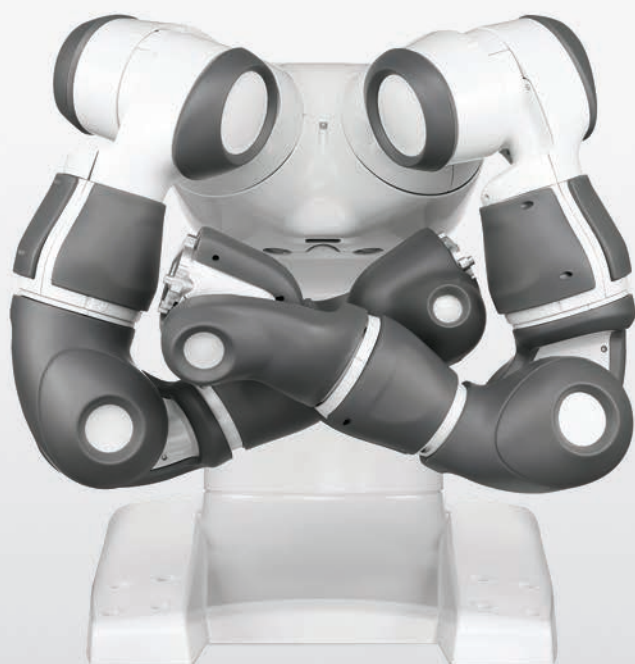


ROBOTICS

Operating manual

IRB 14000



Trace back information:
Workspace 22D version a5
Checked in 2022-12-07
Skribenta version 5.5.019

Operating manual

IRB 14000

RobotWare 6.14.01

Document ID: 3HAC052986-001

Revision: H

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 2015-2022 ABB. All rights reserved.
Specifications subject to change without notice.

Table of contents

Overview of this manual	7
Product documentation	9
1 Introduction to the IRB 14000 robot	11
1.1 What is IRB 14000?	11
1.2 What is a FlexPendant?	12
1.3 RobotStudio	13
1.4 RobotStudio Online	14
2 Using the IRB 14000	17
2.1 Axes and coordinate systems	17
2.2 Jogging	19
2.2.1 What is jogging?	19
2.2.2 Motion modes	21
2.2.3 Coordinated jogging	22
2.3 Lead-through	24
2.4 Operating modes	26
2.5 Collision Avoidance	28
2.6 Collision	31
2.7 Programming and testing	32
2.8 I/O signals	35
2.9 User authorization	36
3 Calibration	37
3.1 Introduction	37
3.2 Calibration scale and correct axis position	38
3.3 Updating revolution counters	40
3.4 Verifying the calibration position	46
4 System parameters	47
4.1 Introduction	47
4.2 Topic I/O System	48
4.2.1 Collision Avoidance	48
4.2.2 Connection Timeout Multiplier	49
4.3 Topic Motion	50
4.3.1 Coll-Pred Safety Distance	50
4.3.2 Global Speed Limit	51
4.3.3 Arm Check Point Speed Limit	52
4.3.4 Arm-Angle Definition	53
4.3.5 Arm-Angle Reference Direction	54
4.3.6 Limit avoidance distance	55
4.3.7 Collision Detection Memory	56
4.3.8 Friction comp. lead through factor	57
Index	59

This page is intentionally left blank

Overview of this manual

About this manual

This manual contains instructions for daily operation of the IRB 14000 robot system.



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.



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

References	Document ID
<i>Product manual - IRB 14000</i>	3HAC052983-001
<i>Operating manual - IRC5 with FlexPendant</i>	3HAC050941-001
<i>Operating manual - Getting started, IRC5 and RobotStudio</i>	3HAC027097-001
<i>Operating manual - RobotStudio</i>	3HAC032104-001
<i>Operating manual - Troubleshooting IRC5</i>	3HAC020738-001
<i>Technical reference manual - System parameters</i>	3HAC050948-001
<i>Technical reference manual - RAPID Overview</i>	3HAC050947-001
<i>Technical reference manual - RAPID Instructions, Functions and Data types</i>	3HAC050917-001
<i>Technical reference manual - RAPID kernel</i>	3HAC050946-001
<i>Application manual - Controller software IRC5</i>	3HAC050798-001

Continues on next page

References	Document ID
Application manual - MultiMove	3HAC050961-001

Revisions

Revision	Description
-	Released with RobotWare 6.01.
A	Released with RobotWare 6.03. <ul style="list-style-type: none"> Minor corrections Value in Coll-Pred Safety Distance on page 50 changed from 0.001 to 0.01. The predefined signal <code>Collision_Avoidance</code> is added. Added information to section Collision Avoidance on page 28. Added section Lead-through on page 24. Applicable ESD-standards added.
B	Released with RobotWare 6.04. <ul style="list-style-type: none"> The RAPID instruction <code>ContactL</code> has been moved from this manual. It is now described in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>. Value in Arm Check Point Speed Limit on page 52 changed from 1.0 to 0.75. Added Programming Move-instructions on page 32. Added the system parameter Limit avoidance distance on page 55. Minor corrections.
C	Released with RobotWare 6.05. <ul style="list-style-type: none"> Added the new section Collision on page 31. Added the new parameter Friction comp. lead through factor on page 57. Updated the parameter Collision Detection Memory on page 56. Updated descriptions of stops.
D	Released with RobotWare 6.06. <ul style="list-style-type: none"> Updated the section Collision Detection Memory on page 56.
E	Released with RobotWare 6.07. <ul style="list-style-type: none"> Updated the system parameter Connection Timeout Multiplier on page 49. Updated the Limitations section of the system parameter Arm-Angle Definition on page 53. Safety section restructured.
F	Released with RobotWare 6.09. <ul style="list-style-type: none"> Updated information about Collision Avoidance on page 28. Value in Coll-Pred Safety Distance on page 50 changed from 0.01 to 0.001.
G	Released with RobotWare 6.10.01. <ul style="list-style-type: none"> Updated the section Lead-through on page 24.
H	Released with RobotWare 6.14.01. <ul style="list-style-type: none"> Added information about a new version of the FlexPendant. Minor corrections.

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.

1 Introduction to the IRB 14000 robot

1.1 What is IRB 14000?

The IRB 14000 robot

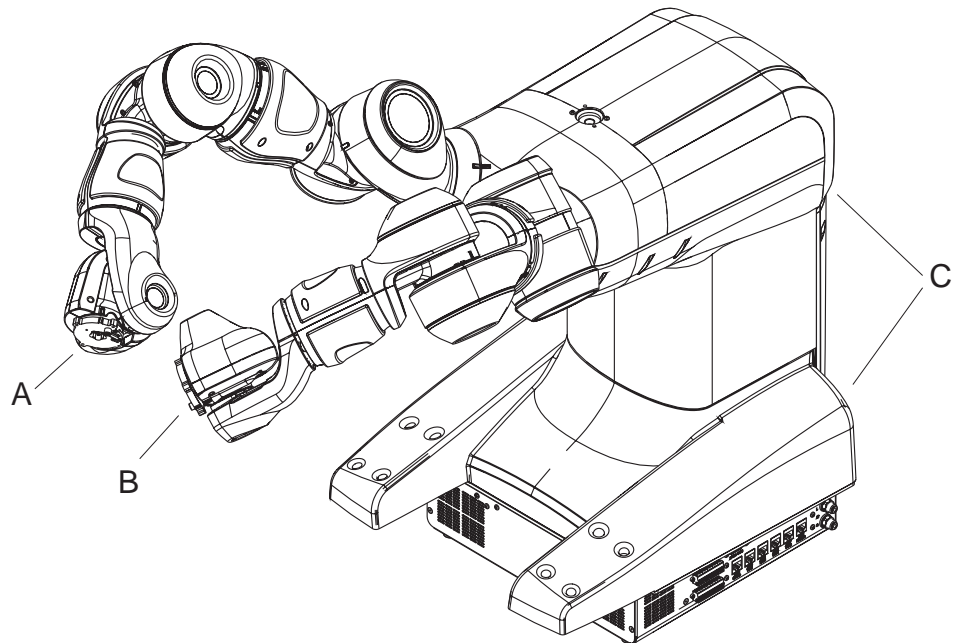
The IRB 14000 robot is a two-armed industrial robot with integrated controller. Each arm has seven axes, which gives an extra degree of freedom compared to traditional 6-axis robots.

The IRB 14000 integrated controller

The IRB 14000 integrated controller is based on the standard IRC5 controller, and contains all functions needed to move and control the robot.

The robot control software, RobotWare, supports every aspect of the robot system, such as motion control, development and execution of application programs, communication etc.

Illustration



xx150000008

A	Right arm
B	Left arm
C	Integrated controller

1 Introduction to the IRB 14000 robot

1.2 What is a FlexPendant?

1.2 What is a 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 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 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

The different versions of the FlexPendant are compatible and can be used on all IRC5 controllers.

For more information on using the FlexPendant, see the section *Navigating and handling FlexPendant* in *Operating manual - IRC5 with FlexPendant*.

1.3 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 Introduction to the IRB 14000 robot

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.


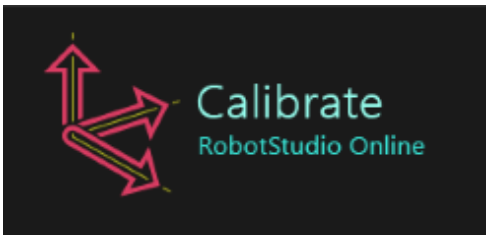
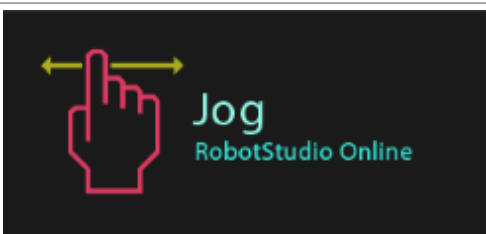
You can run these apps on a tablet that communicates with the robot controller wirelessly. To enable certain functionality, such as entering manual mode and enabling power to the mechanical unit motors, you need a safety device that is connected to the robot using the same plug that alternatively is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

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

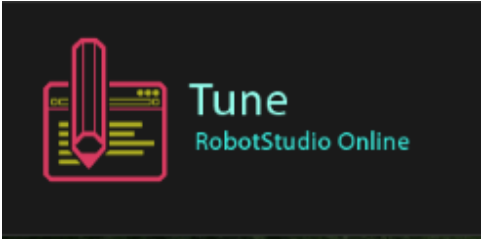
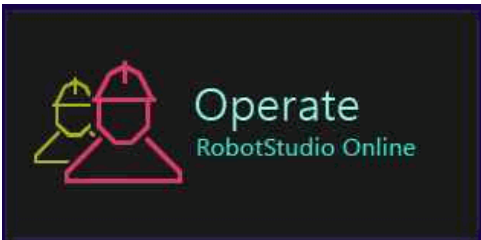
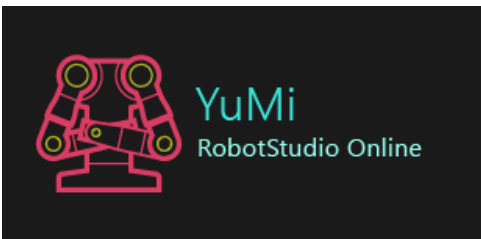


Note

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>

This page is intentionally left blank

2 Using the IRB 14000

2.1 Axes and coordinate systems

What is a coordinate system?

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.

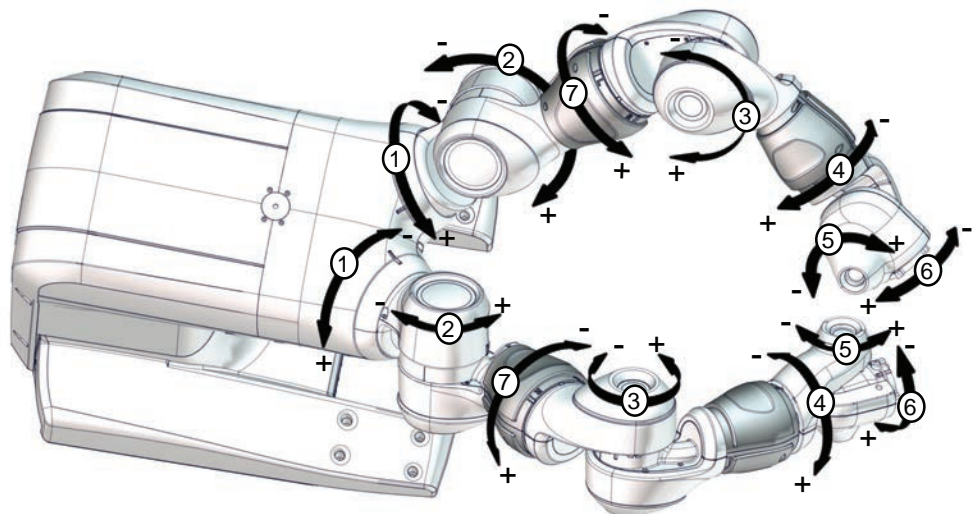
A robot uses several coordinate systems, each suitable for specific types of jogging or programming.

- 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.

For more information on coordinate systems, see the *Jogging* section in *Operating manual - IRC5 with FlexPendant*.

Axes and joystick directions

The axes of the robot can be jogged manually using the joystick. The following illustration shows the location and movement patterns for each axis.



xx150000254

Continues on next page

2 Using the IRB 14000

2.1 Axes and coordinate systems

Continued



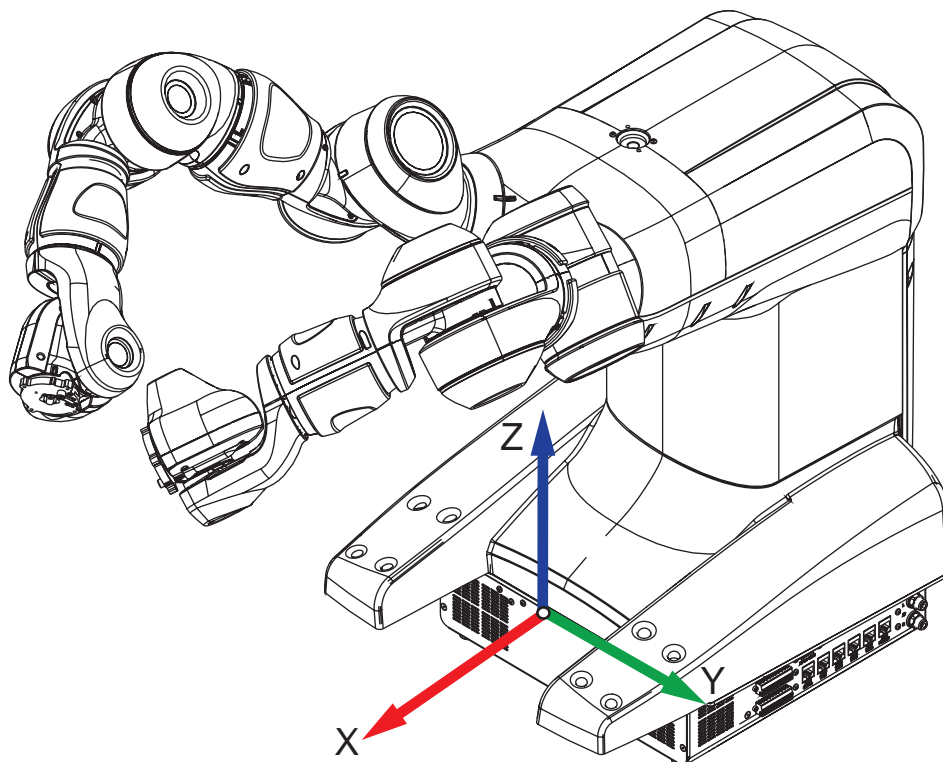
Note

Note that axis 7 is located between axis 2 and axis 3.

The base coordinate system

The base coordinate system has its zero point in the base of the robot.

When you are standing in front of the robot and jog in the base coordinate 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.



xx150000007

2.2 Jogging

2.2.1 What is jogging?

Introduction

To jog is to manually position or move robots or external axes. You can only jog in manual mode, but not during program execution. Jogging is disabled in automatic mode.

The selected motion mode and/or coordinate system determines the way the robot moves. For more information on how to jog robots, see the *Jogging* section in *Operating manual - IRC5 with FlexPendant*.



Note

This manual only describes the settings that are specific for the IRB 14000.

The jogging window

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

Manual
IRB_14000 (SEVST-L-0007293)

Motors On
Stopped (2 of 2) (Speed 100%)

Jogging

Tap a property to change it

Mechanical unit:	ROB_R...
Absolute accuracy:	Off
Motion mode:	Axis 7 - 9...
Coordinate system:	World...
Tool:	tool0...
Work object:	wobj0...
Payload:	load0...
Joystick lock:	None...
Increment:	None...

Position

1:	0.0	°
2:	-130.0	°
3:	30.0	°
4:	0.0	°
5:	40.0	°
6:	0.0	°
7:	-135.0	°

Position Format...

Joystick directions

7

Align... Go To... Activate... Enable Lead-through

Jogging

ROB_R

xx1500000006



Tip

Use the hard buttons on the FlexPendant to toggle between the different motion modes.

Continues on next page

2 Using the IRB 14000

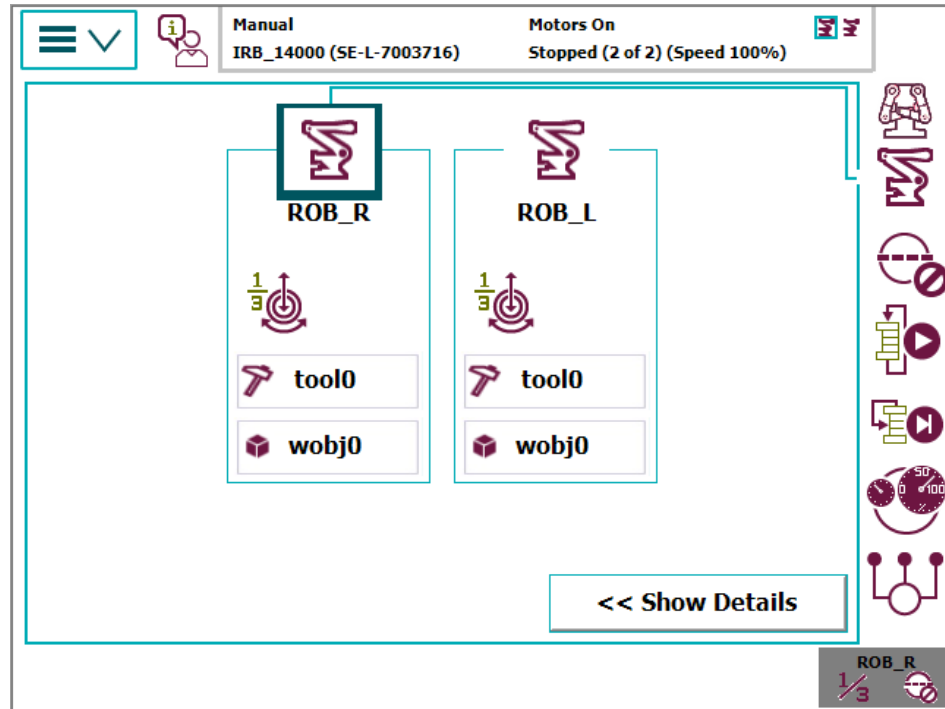
2.2.1 What is jogging?

Continued

The Quickset menu

The Quickset menu provides a quicker way to change among other things jog properties rather than using the jogging window.

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



xx150000004



Note

Note that the operator panel is located under the Quickset menu. For more information on how to switch between manual and automatic mode, see [Operating modes on page 26](#).

2.2.2 Motion modes

Standard motion modes

A traditional robot, with four to six axes, has three different motion modes: *axis-by-axis*, *linear*, and *reorientation* mode.

- *Axis-by-axis mode* moves one robot axis at a time. The tool center point, and the orientation of the tool, is not monitored. Axis-by-axis mode is used to manually position the robot before switching over to linear mode.
- In *linear motion mode*, the tool center point moves along a straight line in space, in a “move from point A to point B” fashion. The tool center point is monitored and moves in the direction of the selected coordinate system’s axes. The orientation of the tool is fixed throughout the motion.
- In *reorientation mode*, the tool center point is fixed in space and the orientation of the tool is changed. The tool center point is rotated around the direction of the selected coordinate system’s axes.

The arm mode

The IRB 14000 robot, with seven axes, has one additional motion mode, the *arm mode*. All of the other jogging settings are the same as for other robots.

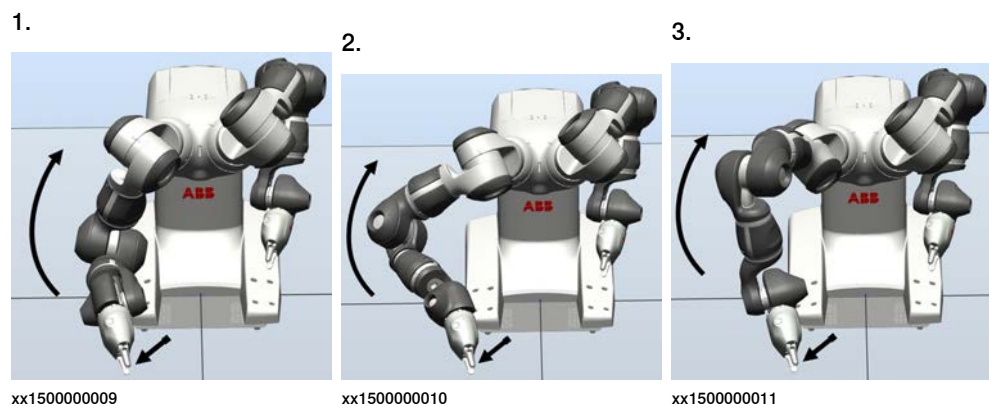
- In *arm mode*, both the tool center point and the orientation of the tool is fixed in space and only the angle of the arm is changed. The tool center point is neither rotated nor moved. See [Jogging in arm mode on page 21](#).

For more information on robot configuration and how the arm angle is calculated, see the data type `confdata` in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Jogging in arm mode

In the pictures below, the robot is jogged in arm mode. Note that both the tool center point and the orientation of the tool are fixed in space and only the angle of the arm is changed.

This is useful when programming to avoid singularity, to find the most natural way the robot can reach a given target, and also how to be able to proceed to the next target.



2 Using the IRB 14000

2.2.3 Coordinated jogging

2.2.3 Coordinated jogging

Introduction

The IRB 14000 robot is pre-installed with the RobotWare option *MultiMove coordinated*, which makes it possible to jog the two arms in coordinated mode.

Coordinated jogging has to be setup by creating a coordinated work object. The work object should be setup for the arm that is holding the work piece. The other arm is holding the tool.

When the arm moving the work object is jogged, the other arm that is currently coordinated with the work object will move so that it maintains its relative position to the work object.

For more information about MultiMove and coordinated jogging, see *Application manual - MultiMove*.



Tip

Use the MultiMove wizard in RobotStudio when setting up and programming MultiMove.

Setup coordinated jogging

Use this procedure to setup coordinated jogging.

	Action	Description
1	Create a work object for the arm that is to be coordinated. That work object will be held and moved by the other arm.	
2	Define the data of the work object. Set <code>hobhold</code> and <code>ufprog</code> to <code>FALSE</code> , set <code>ufmec</code> to the other arm.	In this example the right arm is holding the work object and the left arm is coordinated to it: <pre>PERS wobjdata wobjRight := [FALSE, FALSE, "ROB_R", [[0,0,0],[1,0,0,0]], [[0,0,0],[1,0,0,0]]];</pre>
3	Optional, define x, y, z values for the work object and define a tool for the other arm.	
4	Activate coordinated jogging.	Activate coordinated jogging on page 22

Activate coordinated jogging

Use this procedure to activate coordinated jogging.

	Action	Description
1	Open the Quickset menu and select the arm that is to be coordinated.	The Quickset menu on page 20
2	Activate the previously created work object.	Setup coordinated jogging on page 22
3	Select the work object coordinate system.	
4	Select the other arm.	The coordinated arm is now indicated with a flashing frame.

Continues on next page

	Action	Description
5	Jog the arm, the other will follow.	

Deactivate coordinated jogging

Turn off coordination in one of following ways:

- Click the **Turn coordination off** button on the Quickset menu.
- Deactivate the work object.
- Deactivate the work object coordinate system.

2 Using the IRB 14000

2.3 Lead-through

2.3 Lead-through

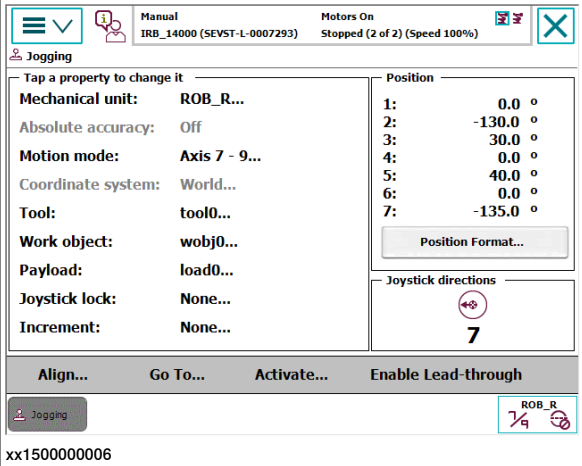
What is lead-through?

The lead-through functionality is available for robots designed for collaborative applications. If lead-through is available, this is shown on the FlexPendant.

Using lead-through, you can grab the robot arm and move it manually to a desired position, as an alternative to jogging.

Using lead-through

Use the following procedure to jog the robot using the lead-through functionality:

	Action	Note/illustration
1	In the jogging window, tap the button Enable Lead-through .	 <p>The screenshot shows the FlexPendant interface in 'Jogging' mode. At the top, it displays 'Manual IRB_14000 (SEVST-L-0007293)' and 'Motors On Stopped (2 of 2) (Speed 100%)'. Below this, there are several settings: Mechanical unit: ROB_R..., Absolute accuracy: Off, Motion mode: Axis 7 - 9..., Coordinate system: World..., Tool: tool0..., Work object: wobj0..., Payload: load0..., Joystick lock: None..., and Increment: None... To the right, a 'Position' table shows values for axes 1 through 7. At the bottom, there are buttons for 'Align...', 'Go To...', 'Activate...', and 'Enable Lead-through', with the latter being highlighted.</p>
2	Gently pull the robot arm to the desired position.	You will feel if an axis reaches its end position. Do not try to force the axis beyond this position.



Note

Lead-through is enabled per robot arm. To enable lead-through for both arms, enable one arm and then select the other arm and enable that too.



Note

If the robot is in motors off mode, it will automatically go to motors on when enabling lead-through.



Note

If lead-through is enabled, it will be temporarily disabled during program execution and jogging. This means that it is possible to combine lead-through, jogging, and testing the RAPID program without having to disable the lead-through.

Continues on next page



Note

When using lead-through, it is important that the load is correctly defined. If the load is heavier than defined, the effect will be the same as if you are pulling the robot arm downwards. If the load is lighter than the defined load, the effect will be the same as if you are pulling the robot arm upwards.

Limitations

When using lead-through, the path planner is not updated until the program is resumed/restarted or jogging is used. For example, this means that World Zones supervision is not accessible when using lead-through.

2 Using the IRB 14000

2.4 Operating modes

2.4 Operating modes

Introduction

The IRB 14000 has two operating modes, *Manual mode* and *Automatic mode*.



Note

Note that the IRB 14000 does not go to motors off when changing the operating mode.



Note

The IRB 14000 goes automatically to motors on when jogging or when tapping the play button or step button on the FlexPendant.

What is the manual mode?

In manual mode the manipulator movement is under manual control. The manual mode is used when programming and for program verification.

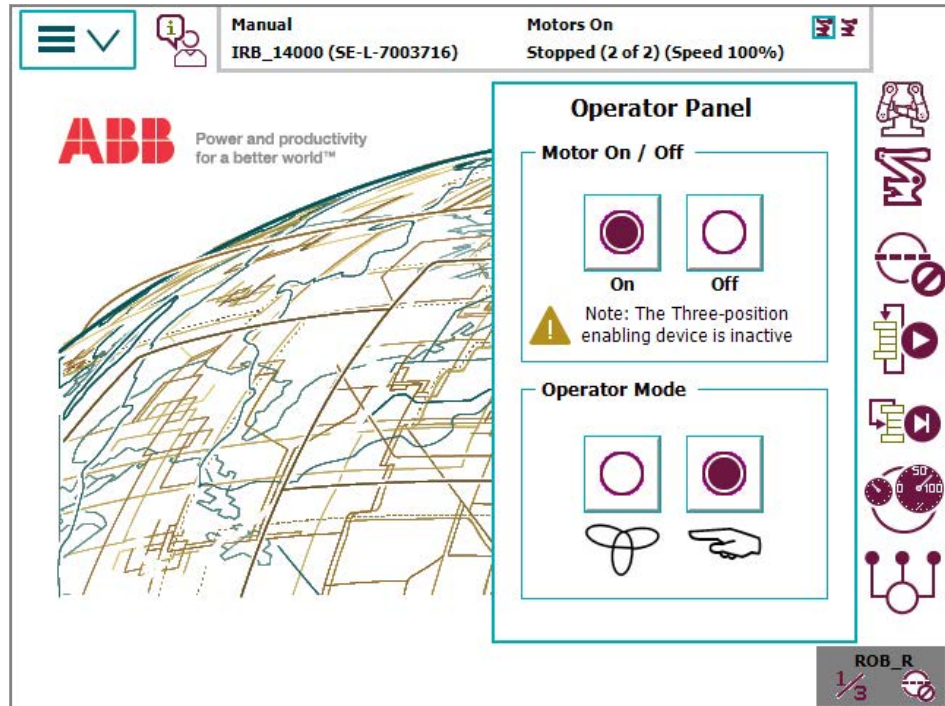
What is the automatic mode?

The automatic mode is the operating mode in which the robot control system operates in accordance with the task program, with functional safeguarding measures. This mode enables controlling the manipulator for example by using the I/O signals on the controller. An input signal may be used to start and stop a RAPID program, another to activate the motors on the manipulator.

Continues on next page

The operator panel

The operator panel is used to switch between *Manual mode* and *Automatic mode*. The operator panel is located under the Quickset menu, see [The Quickset menu on page 20](#).



xx150000005



Note

It is also possible to use the predefined I/O-signals to change and confirm the operating mode, see [I/O signals on page 35](#).

2.5 Collision Avoidance

Introduction

The function *Collision Avoidance* monitors a detailed geometric model of the robot. By defining additional geometrical models of bodies in the robot workarea, the controller will warn about a predicted collision and stops the robot if two bodies come too close to each other. The system parameter *Coll-Pred Safety Distance* determines at what distance the two objects are considered to be in collision.

The function *Collision Avoidance* is useful for example when setting up and testing programs, or for programs where positions are not static but created from sensors, such as cameras (non-deterministic programs). By using trigger-signals (see [Trigger signals on page 29](#)), *Collision Avoidance* can be used for implementing safe workspace sharing between multiple robots.

Besides the robot itself the function will monitor up to 10 objects that is created via the configurator in RobotStudio. Typical objects to be monitored are tool mounted on the robot flange, additional equipment mounted on the robot arm (typically axis 3) or static volume around the robot.

The geometric models are set up in RobotStudio.

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

Collision Avoidance is active both during jogging and when running programs. Also, the RAPID function `IsCollFree` provides a way to check possible collisions before moving to a position.



CAUTION

Always be careful to avoid collisions with external equipment, since a collision could damage the mechanical structure of the arm.

Collision Avoidance is no guarantee for avoiding collisions.



Tip

How to configure *Collision Avoidance* is described in *Operating manual - RobotStudio*.

False collision warning

There are different ways to lower the sensitivity of the function *Collision Avoidance* to avoid false warnings.

- Temporarily disable *Collision Avoidance*, see [Disabling Collision Avoidance on page 30](#).
- For IRB 14000, decrease the safety distance for the arm or geometric model that triggers the false collision warning, see [Decrease sensitivity between links for IRB 14000 on page 30](#).

Continues on next page

- Decrease the general safety distance with the system parameter *Coll-Pred Safety Distance*.

Activation/deactivation of objects

By default, a defined collision object is active all the time. However, it is possible to configure a collision object with an activation signal, which basically connects it to a digital input that determines whether the object is active or not. This is useful, for example, for modelling multiple tools, where only one tool at a time is active. Another use case is modelling of objects that can be present or absent in the robot cell, for example a pallet.

Note that changing the state of an activation signal will immediately change the activation state of the connected collision object, and no synchronization to the robot path planning is made. Activating a collision object while the robot is moving towards the object can thus lead to a collision because the planned path may already have passed by the collision object while it was inactive. If synchronization is important, then activation signals should either be changed in finepoints when the robot is standing still or using *trigg* instructions like *TriggL* or *TriggJ*.

Trigger signals

A non-moving collision object can be configured with a trigger signal. The value of the trigger signal reflects which robots are in contact with the collision object. More specifically, the value of a trigger signal should be interpreted as a bit pattern, where bit *k* is high if robot *k* is in contact with the collision object. For example, if the trigger signal has the value 6, which is 110 in binary, it means that ROB_2 and ROB_3 are in contact with the collision object. Trigger signals can be used to implement safe workspace sharing between multiple robots.

A trigger signal can be configured with two timing behaviors: *immediate* or *on-arrival*. If configured with *immediate* behavior, then the trigger signal is changed as quickly as possible, well before the robot has physically reached the position where it comes into contact with the collision object. If configured with *on-arrival* behavior, then the trigger signal changes state when the robot physically reaches the position where it comes in contact with the zone.

Limitations



CAUTION

Collision Avoidance shall not be used for safety of personnel.

- *Collision Avoidance* is a function included in the option *Collision Detection*.
- Paint robots, IRB 6620LX, and delta robots are not supported.
- *Collision Avoidance* cannot be used in manual mode together with responsive jogging. The system parameter *Jog Mode* must be changed to *Standard*.
- Only stationary/non-moving objects can be configured with a trigger signal. A trigger signal must correspond to a group signal. Furthermore, each collision object must have its own trigger signal.

Continues on next page

2 Using the IRB 14000

2.5 Collision Avoidance

Continued

- There is no support for applications that do corrections to the path, such as conveyor tracking, WeldGuide, Force Control, SoftMove, SoftAct etc.

Disabling *Collision Avoidance*

It is possible to temporarily disable the function *Collision Avoidance* if the robot has already collided or is within the default safety distance, or when the robot arms need to be very close and the risk of collision is acceptable.

Set the system input signal `Collision Avoidance` to 0 to disable *Collision Avoidance*. It is recommended to enable it (set `Collision Avoidance` to 1) as soon as the work is done that required *Collision Avoidance* to be disabled.

Decrease sensitivity between links for IRB 14000

For dual arm robots, the sensitivity can be decreased between individual robot arm links. This is useful if two links come close to each other, but the general safety distance should be maintained.

Open the file `irb_14000_common_config.xml` located in the folder `<SystemName>\PRODUCT\ROBOTWARE_6.XX.XXXX\robots\CA\irb_14000`.

For example, to decrease the safety distance between the left arm's link 3 and the right arm's link 4 to 1 mm, add the following row:

```
<Pair object1="ROB_L_Link3" object2="ROB_R_Link4"
      safetyDistance="0.001" />
```

To decrease the safety distance between the left arm's link 5 and the robot base to 2 mm, add the following row:

```
<Pair object1="ROB_L_Link5" object2="Base" safetyDistance="0.002" />
```

To disable collision avoidance between the left arm's link 2 and the right arm's link 3, add the following row:

```
<Pair object1="ROB_L_Link2" object2="ROB_R_Link3" exclude="true" />
```



Note

The safety distance between two links can be decreased by adding a row to this XML file, but it cannot be increased to a higher value than defined by the system parameter *Coll-Pred Safety Distance*.

2.6 Collision

Overview

This section provides you an overview of the response of IRB 14000 robot system in the case of a collision.

Collision

If a collision is detected on an arm, the collided arm stops and the other arm keeps moving. But for a synchronized movement (*SyncMoveOn*), both arms stop in the case of a collision.

For more information about motion error handling see *Technical reference manual - RAPID kernel*.



Note

In RobotWare versions prior to 6.05 both arms stop in the case of a collision.

2 Using the IRB 14000

2.7 Programming and testing

2.7 Programming and testing

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

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

Programming language

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

Programming Move-instructions

When programming a `jointtarget` for a 7-axis robot, the value of axis 7 (in degrees) is stored as the first external axis value in the `jointtarget` data.

Example:

```
! The value 135 is the angle of axis 7 in degrees:
CONST jointtarget myjointtarget := [[0,-130,30,0,40,0],
  [135,9E+09,9E+09,9E+09,9E+09,9E+09]];

MoveAbsJ myjointtarget \NoEOffs, v1000, fine, tool0;
```

When programming a `robtarget` for a 7-axis robot, the arm-angle is stored as the first external axis value in the `robtarget` data.

Example:

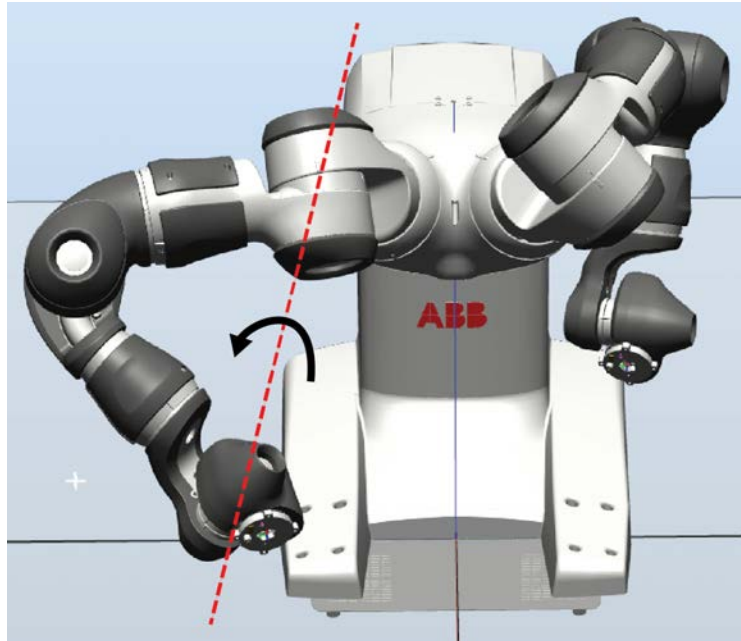
```
! The value 107.816 is the calculated arm-angle:
CONST robtarget myrobtarget :=
  [[-9,182,198],[0.0660087,0.84242,-0.111216,0.52307],[0,0,0,11],
  [107.816,9E+09,9E+09,9E+09,9E+09,9E+09]];

MoveL myrobtarget \NoEOffs, v1000, fine, tool0;
```

The arm-angle can, in a simplified way, be described as the angle of the arm seen from an axis that goes from the centre of axis 2 to the centre of the WCP (wrist)

Continues on next page

center point). This value cannot be measured or calculated by the user since the underlying mathematical calculations are far more complex.



xx1600001072



Note

If necessary the arm-angle will be adjusted automatically during run time in order to avoid singularities and joint limits. The desired distance can be adjusted with the system parameter *Limit avoidance distance*, see [Limit avoidance distance on page 55](#).

Coordinated programming using MultiMove

The IRB 14000 robot is pre-installed with the RobotWare option *MultiMove coordinated*, which makes it possible to program the two arms in coordinated mode. For more information about MultiMove and coordinated jogging, see *Application manual - MultiMove*.



Tip

Use the MultiMove wizard in RobotStudio when setting up and programming MultiMove.

Configuration data

When programming linear movements it is important that the programmed positions have similar configurations, otherwise it will not be possible to move linearly between the positions.

This is important when programming all robots, but especially when programming 7-axis robots, since the *arm mode* adds more complexity.

Continues on next page

2 Using the IRB 14000

2.7 Programming and testing

Continued

The data type `confdata` is used to define the configurations.

For more information about the data type `confdata`, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

Contact applications

The RAPID instruction `ContactL` is designed to be used for contact applications, when the tool held by the robot has to press an object into place.

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

2.8 I/O signals

Introduction

It is possible to connect different types of I/O signals to the IRB 14000, both digital I/O signals and different types of fieldbuses (industrial networks).

For more information about connecting I/O signals, see *Product manual - IRB 14000* and *Circuit diagram - IRB 14000*.

Predefined signals

Operating mode signals

The following output signals are predefined in the system and can be used to change and confirm the operating mode.

Name	Type	Description
VP_ENABLE	output	Enable signal for manual mode.
VP_MODEKEY	output	Operating mode selector.
VP_MOTOPB	output	Motors On push button.

Collision avoidance signal

Name	Type	Description
Collision_Avoidance	output	Default is 1, meaning that collision avoidance is enabled. Setting this signal to 0 will disable collision avoidance. This is usable for tasks where you need the arms to be very close and the risk of collision is acceptable.

2.9 User authorization

Introduction

The data, functionality, and commands on a controller are protected by a User Authorization system (also called UAS). The UAS restricts the parts of the system the user has access to. Different users can have different access grants.

It is recommended to create different user groups for different types of users. For example *Operator*, *Engineer*, and *Service*. Operators should have very limited access to the system.



Note

Only users that are authorized to modify safety functions should have access to the grants *Restore a backup* and *Modify configuration*.

For more information about configuring the User Authorization system and the different controller grants, see *Operating manual - RobotStudio*.

Changing safety related system parameters

When changing the safety related system parameters *Arm Check Point Speed Limit* or *Global Speed Limit* an event message will take focus on the FlexPendant after restart to notify the user of the change.

For more information about the system parameters, see [System parameters on page 47](#).

3 Calibration

3.1 Introduction

General

This chapter includes information about when the robot system must be recalibrated. There are two types of calibration, to update the revolution counters or to do a fine calibration.

When to update the revolution counters

If the revolution counter memory is lost, the counters must be updated. This will occur when:

- The battery is discharged
- A resolver error occurs
- The signal between a resolver and measurement board is interrupted
- A robot axis is moved with the control system disconnected

The revolution counters must also be updated after the robot and controller are connected at the first installation.

To update the revolution counters is a simple procedure which can be performed by the operator, see [Updating revolution counters on page 40](#).

When to do a fine calibration

The system must be fine calibrated when parts affecting the calibration position are replaced on the robot, for example motors or parts of the transmission.

A fine calibration should only be performed by a qualified service engineer. For more information, see *Calibration* in *Product manual - IRB 14000*.

3 Calibration

3.2 Calibration scale and correct axis position

3.2 Calibration scale and correct axis position

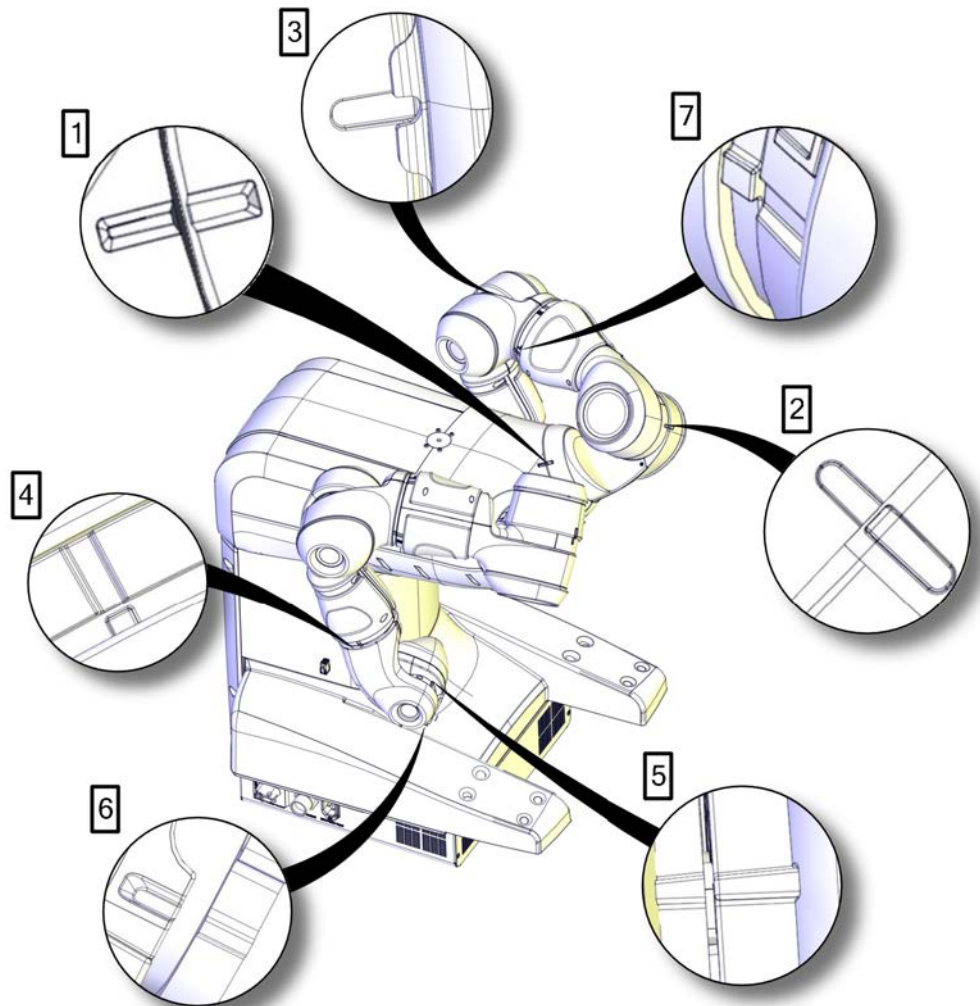
Introduction

This section specifies the calibration scale positions and/or correct axis positions.

Calibration scales/marks

This illustration shows the positions of the calibration scales and marks on the robot.

The number aside of the enlargement corresponds to the axis number.



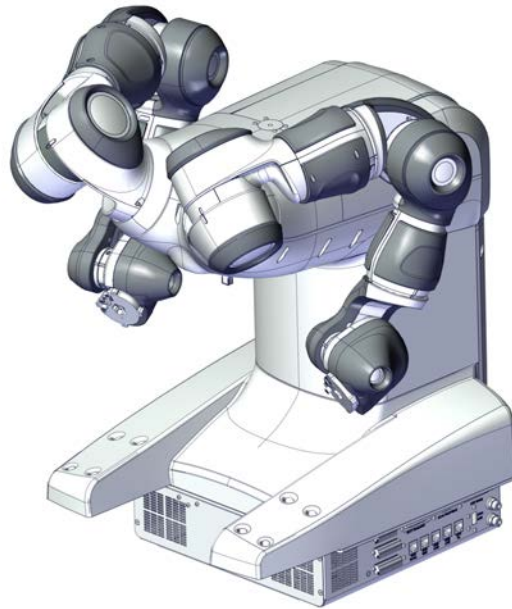
xx150000526

Continues on next page

Calibration position

Illustration of robot in calibration position

The figure shows the robot in its calibration position.



xx150000363

Exact axis positions in degrees

The table below specifies the exact axis positions in degrees.

Axis	IRB 14000 ROB_R	IRB 14000 ROB_L
1	0°	0°
2	-130°	-130°
3	30°	30°
4	0°	0°
5	40°	40°
6	0°	0°
7	-135°	135°

3 Calibration

3.3 Updating revolution counters

3.3 Updating revolution counters

Introduction


This section describes how to do a rough calibration of each robot axis, which updates the revolution counter value for each axis using the FlexPendant.

The procedure can be summarized accordingly:

- 1 Manually move the manipulator to the calibration position.
- 2 Select the Calibration with hall sensors (CalHall) routine.
- 3 Select the function Update of revolution counters.
- 4 Store the revolution counter setting.

Each step is described in detail in following sections.

Step 1 - Manually moving the manipulator to the calibration position

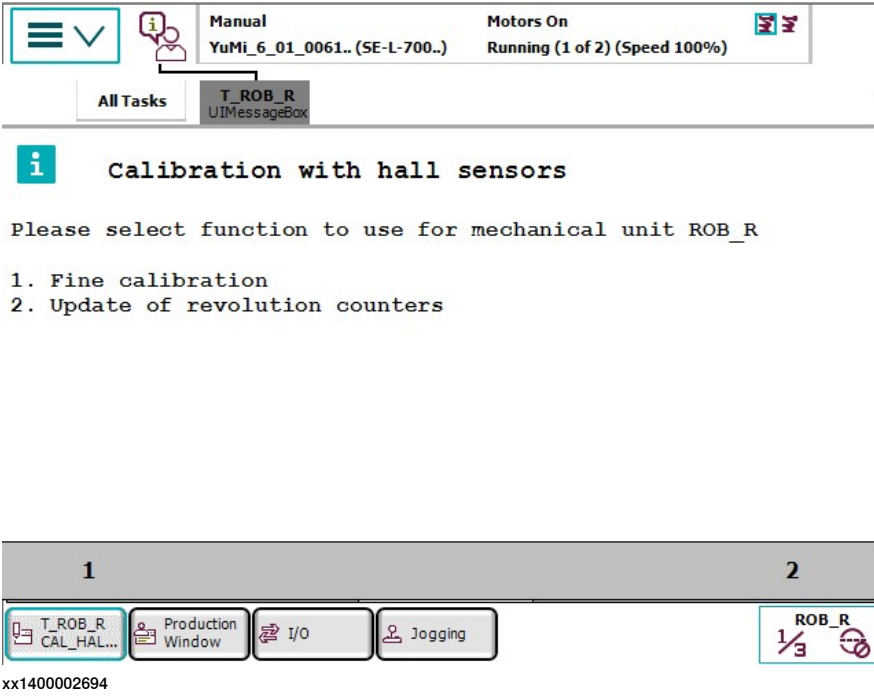
	Action	Note
1	 CAUTION When releasing the holding brakes, the robot axes may move very quickly and sometimes in unexpected ways!	
2	Release the brakes of the robot arm to be calibrated and move the arm manually so that the synchronization mark of each joint is aligned. The robot now stands in its calibration position.	The synchronization marks are shown in Calibration scale and correct axis position on page 38 . There is a tolerance for the joint position. The edge of a mark should be at least within the area of the opposite mark.

Step 2 - Selecting the Calibration with hall sensors (CalHall) routine

	Action	Note
1	Open the Program Editor on the FlexPendant.	
2	Select the task that corresponds to the robot arm to be calibrated. Tap Open .	
3	If necessary, create a new program. This needs to be done if no existing program is available.	
4	Select Debug and tap PP to Main .	
5	Select Debug and tap Call Routine...	
6	Select CalHall .	
7	Go to Motor On and press the Start button .	

Continues on next page

Step 3 - Selecting the function Update of revolution counters

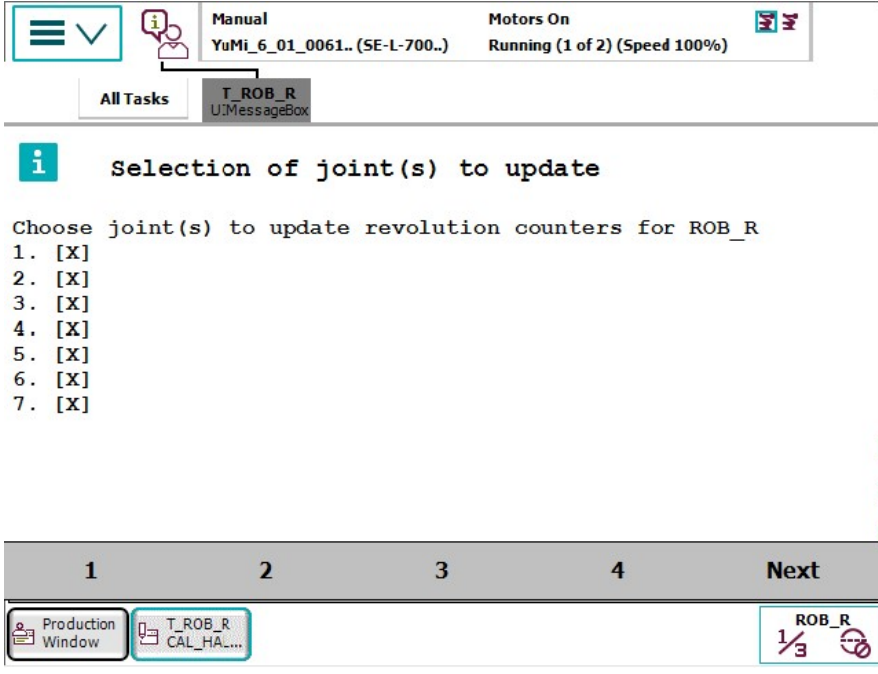
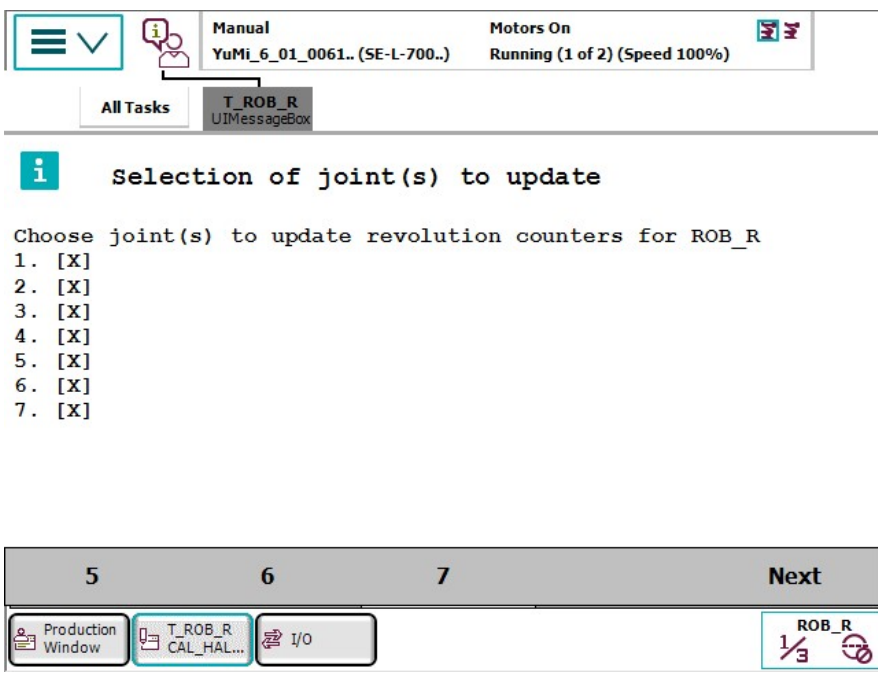
	Action
1	<p>In the calibration with hall sensors routine (CalHall), tap 2 to select the function of updating the revolution counters.</p>  <p>i Calibration with hall sensors</p> <p>Please select function to use for mechanical unit ROB_R</p> <ol style="list-style-type: none"> 1. Fine calibration 2. Update of revolution counters <p>1 2</p> <p>T_ROB_R CAL_HAL... Production Window I/O Jogging ROB_R 1/3</p> <p>xx1400002694</p>

Continues on next page

3 Calibration


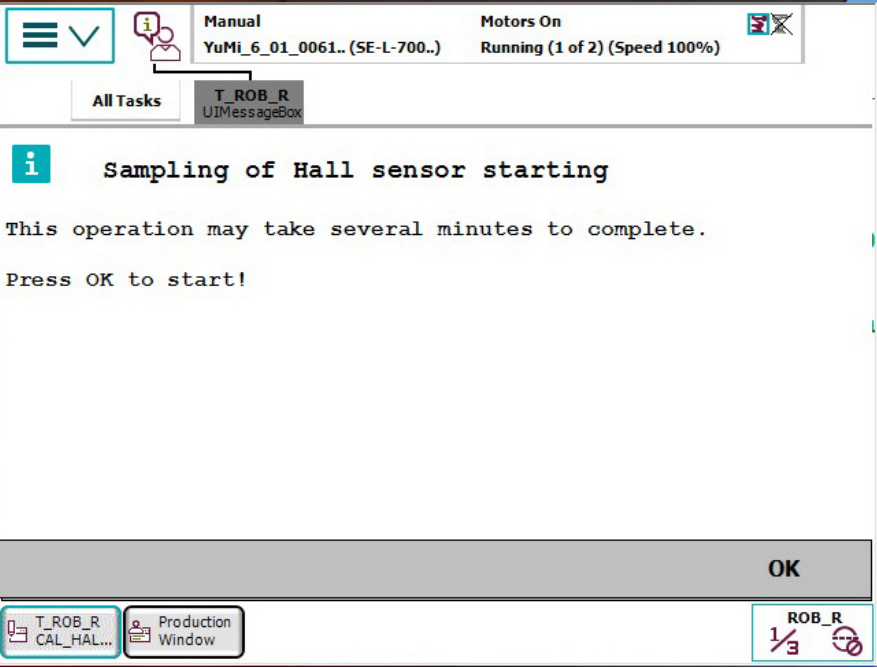
3.3 Updating revolution counters

Continued

	Action
2	<p>Tap to select which joint(s) to update revolution counters for. Joints 1, 2, 3 and 4 are selectable in the first window. Tap Next to open the second window where joints 5, 6 and 7 are selectable.</p>  <p>The screenshot shows a control panel at the top with a menu icon, a status bar indicating 'Manual YuMi_6_01_0061.. (SE-L-700..)' and 'Motors On Running (1 of 2) (Speed 100%)', and task buttons for 'All Tasks' and 'T_ROB_R'. Below this is an information icon and the title 'Selection of joint(s) to update'. The instruction reads: 'Choose joint(s) to update revolution counters for ROB_R'. A list of joints 1 through 7 is shown, each with a checked checkbox [X]. At the bottom, a navigation bar contains buttons for '1', '2', '3', '4', and 'Next'. Below the navigation bar, a 'Production Window' is visible with a 'T_ROB_R CAL_HAL...' task selected. A 'ROB_R' status indicator shows '1/3' and a stop icon. The ID 'xx1400002695' is displayed at the bottom of the window.</p>  <p>The second screenshot is identical to the first one, showing the same control panel and information section. However, the navigation bar at the bottom now contains buttons for '5', '6', '7', and 'Next'. The 'Production Window' below shows three tasks selected: 'T_ROB_R CAL_HAL...', 'I/O', and 'ROB_R'. The 'ROB_R' status indicator still shows '1/3'. The ID 'xx1400002696' is displayed at the bottom of the window.</p>

Continues on next page

Step 4 - Storing the revolution counter setting

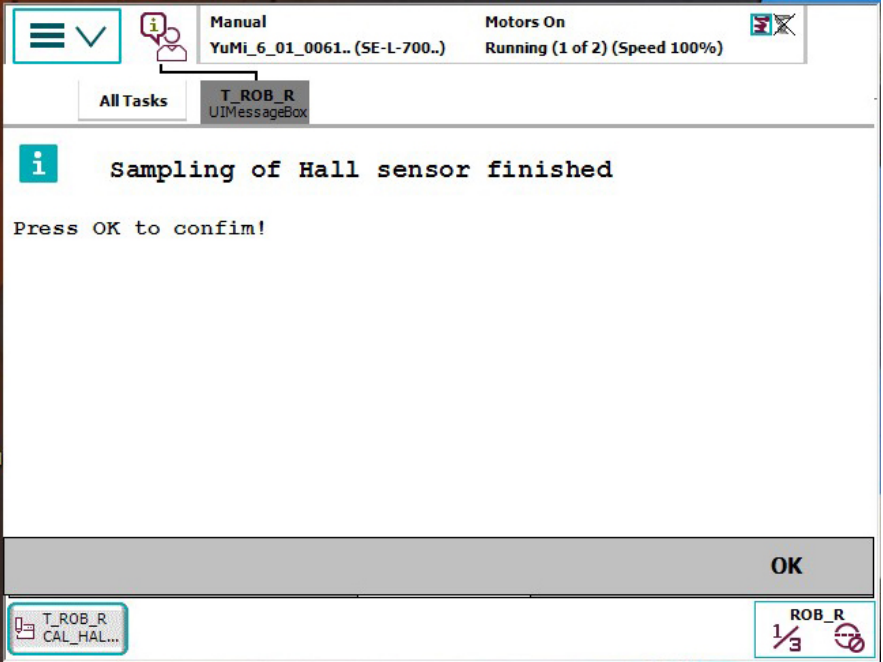
	Action
1	<p>Tap OK to start the sampling procedure of the hall sensor. One at a time, each selected joint now rotates back and forth several times to calculate and find the exact position where the hall sensor is aligned with the magnet. Calibration order: axis 1-2-3-4-5-6-7.</p> <p> Note This procedure may take several minutes.</p>  <p>xx1400002697</p>

Continues on next page

3 Calibration

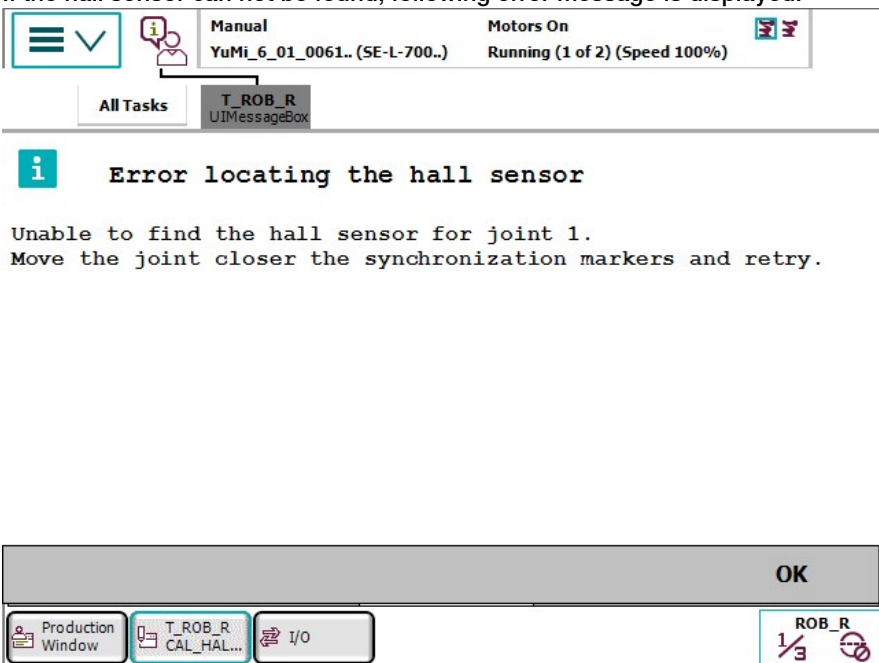
3.3 Updating revolution counters

Continued

Action	
2	<p>When the sampling of hall sensors is finished, tap OK to confirm.</p>  <p>The screenshot displays a mobile application interface. At the top, there is a status bar with a menu icon, a user profile icon, and text: "Manual YuMi_6_01_0061.. (SE-L-700..)" and "Motors On Running (1 of 2) (Speed 100%)". Below the status bar, there are two buttons: "All Tasks" and "T_ROB_R UIMessageBox". The main content area features a blue information icon followed by the text "Sampling of Hall sensor finished" and "Press OK to confirm!". At the bottom right of the main area is an "OK" button. The bottom of the screen shows a task list with "T_ROB_R CAL_HAL..." and "ROB_R 1/3" with a refresh icon.</p>

xx1400002898

Continues on next page

	Action
3	<p>If the hall sensor can not be found, following error message is displayed.</p>  <p>Unable to find the hall sensor for joint 1. Move the joint closer the synchronization markers and retry.</p> <p>OK</p> <p>Production Window T_ROB_R CAL_HAL... I/O ROB_R 1/3</p> <p>xx140002698</p> <p>There can be several causes for this error.</p> <ul style="list-style-type: none"> The joints are not sufficiently aligned according to the synchronization marks. Move the joints so that the synchronization marks are aligned. Tap OK and restart the fine calibration procedure. The joints prior to the joint that stopped the calibration procedure are calibrated and do not need to be calibrated again. Joints coming after the joint that stopped the calibration procedure has not been calibrated. (Calibration order: axis 1-2-3-4-5-6-7.) The hall sensor is defect. Perform troubleshooting of I/O signals. Possible replacement of hall sensor may be required.

3 Calibration

3.4 Verifying the calibration position

3.4 Verifying the calibration position

Introduction

Verify the calibration position of the robot before beginning any programming of the robot system. This may be done:

- Using a `MoveAbsJ` instruction with argument according to calibration position degrees on all axes.
- Using the **Jogging** window on the FlexPendant.

Using a `MoveAbsJ` instruction

Use this procedure to create a program that runs all the robot axes to their calibration position.

	Action	Note
1	On ABB menu tap Program editor .	
2	Create a new program.	
3	Use MoveAbsJ in the Motion&Proc menu.	
4	Create the following program for the right arm: <code>MoveAbsJ [[0,-130,30,0,40,0], [-135,9E9,9E9,9E9,9E9,9E9]]</code> <code>\NoEOffs, v1000, fine, tool0;</code> Create the following program for the left arm: <code>MoveAbsJ [[0,-130,30,0,40,0], [135,9E9,9E9,9E9,9E9,9E9]]</code> <code>\NoEOffs, v1000, fine, tool0;</code>	
5	Run the program in manual mode.	
6	Verify that the calibration marks for the axes align correctly. If they do not, then update the revolution counters.	See Calibration scale and correct axis position on page 38 and Updating revolution counters on page 40 .

Using the jogging window

Use this procedure to jog the robot to the calibration position for all axes.

	Action	Note
1	On the ABB menu, tap Jogging .	
2	Tap Motion mode to select group of axes to jog.	
3	Tap to select the axis to jog, axis 1, 2, or 3.	
4	Manually run the robots axes to a position where the axis position value read on the FlexPendant, is equal to the calibration position degrees.	Degrees are specified in Exact axis positions in degrees on page 39 .
5	Verify that the calibration marks for the axes align correctly. If they do not, then update the revolution counters!	See Calibration scale and correct axis position on page 38 and Updating revolution counters on page 40 .

4 System parameters

4.1 Introduction

About the system parameters

This section describes the IRB 14000 specific system parameters. The parameters are divided into the topic and type they belong to.

For information about other system parameters, see *Technical reference manual - System parameters*.

Topic I/O System

Parameter	For more information, see
Collision Avoidance	Collision Avoidance on page 48
Connection Timeout Multiplier	Connection Timeout Multiplier on page 49

Topic Motion

Parameter	For more information, see
Coll-Pred Safety Distance	Coll-Pred Safety Distance on page 50
Global Speed Limit	Global Speed Limit on page 51
Arm Check Point Speed Limit	Arm Check Point Speed Limit on page 52
Arm-Angle Definition	Arm-Angle Definition on page 53
Arm-Angle Reference Direction	Arm-Angle Reference Direction on page 54
Limit avoidance distance	Limit avoidance distance on page 55
Collision Detection Memory	Collision Detection Memory on page 56
Friction compensation lead through factor	Friction comp. lead through factor on page 57

4 System parameters

4.2.1 Collision Avoidance

Collision Detection

4.2 Topic I/O System

4.2.1 Collision Avoidance

Parent

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

Description

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

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

The function *Collision Avoidance* monitors a detailed geometric model of the robot. If two bodies of the model come too close to each other, the controller warns about a predicted collision and stops the robot. The system parameter *Coll-Pred Safety Distance* determines at what distance the two objects are considered to be in collision, see [Coll-Pred Safety Distance on page 50](#).



Note

For RobotWare versions before RobotWare 6.08, this parameter is only applicable to IRB 14000.

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

Prerequisites

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

4.2.2 Connection Timeout Multiplier

Parent

Connection Timeout Multiplier belongs to the type *Device*, in the topic *I/O System*.

Description

Connection Timeout Multiplier specifies the multiplier applied to the expected packet rate value to derive the value for the Inactivity/Watchdog Timer.

Usage

The *Connection Timeout Multiplier* is a number among 4, 8, 16, 32, 64, 128, 256. It is used together with RPI to calculate the timeout on connections. *RPI* multiplied by *Connection Timeout Multiplier* gives the maximum time before dropping the connection.



Note

For the IRB 14000 and IRB 14050 robots this parameter may have to be tuned depending on your network setup.

Prerequisites

The option *EtherNet/IP Scanner/Adapter* must be installed.

Allowed values

Allowed values are 4, 8, 16, 32, 64, 128, 256, 512.
Default value is 4.

4 System parameters

4.3.1 Coll-Pred Safety Distance

4.3 Topic Motion

4.3.1 Coll-Pred Safety Distance

Parent

Coll-Pred Safety Distance belongs to the type *Motion System*, in the topic *Motion*.

Description

The function *Collision Avoidance* monitors a detailed geometric model of the robot. If two bodies of the model come too close to each other, the controller warns about a predicted collision and stops the robot. The system parameter *Coll-Pred Safety Distance* determines at what distance the two objects are considered to be in collision.

The geometric model for the robot is integrated in RobotWare. The geometric models for external or surrounding equipment are set up in RobotStudio.

The functionality is activated by a system input, see [Collision Avoidance on page 48](#).

Allowed values

A value between 0.001 and 1 meters.

The default value is 0.001.

Limitation

Collision Avoidance is only activated together with the option *Collision Detection*. *Collision Avoidance* can only be used by six and seven axis serial link robots.

4.3.2 Global Speed Limit

Parent

Global Speed Limit belongs to the type *Robot*, in the topic *Motion*.

Description

Global Speed Limit sets the speed limit in meters per second for the tool center point (TCP), the arm check point (ACP), and the wrist center point (WCP).



Note

This parameter is used to configure the safety function Cartesian speed supervision.



Note

When changing this safety related system parameter, an event message will take focus on the FlexPendant after restart to notify the user of the change. The user then has to verify that the intended setting was made.

Limitations

Global Speed Limit is only used for the following robots:

- IRB 14000
- IRB 14050

Setting this parameter for any other robot will not have any effect.

Global Speed Limit can only be used to lower the speed limit from maximum speed limit for each robot type. If a higher value is set, the maximum value for the robot type is used.

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000 and IRB 14050	1.5 m/s

Allowed values

A number between 0.1 and 20.

The default value is 20.

4 System parameters

4.3.3 Arm Check Point Speed Limit

4.3.3 Arm Check Point Speed Limit

Parent

Arm Check Point Speed Limit belongs to the type *Robot*, in the topic *Motion*.

Description

Arm Check Point Speed Limit sets the speed limit in meter per second for the arm check point (ACP).



Note

This parameter is used to configure the safety function Cartesian speed supervision.



Note

When changing this safety related system parameter, an event message will take focus on the FlexPendant after restart to notify the user of the change. The user then has to verify that the intended setting was made.

Limitations

Arm Check Point Speed Limit is only used for the following robots:

- IRB 14000
- IRB 14050

Setting this parameter for any other robot will not have any effect.

Arm Check Point Speed Limit can only be used to lower the speed limit from a maximum speed limit for each robot type. If a higher value is set, the maximum value for the robot type is used.

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000	0.75 m/s
IRB 14050	0.75 m/s

Allowed values

A number between 0.1 and 20.

The default value is 0.75.

4.3.4 Arm-Angle Definition

Parent

Arm-Angle Definition belongs to the type *Robot*, in the topic *Motion*.

Description

To completely specify the pose for a robot with 7 axes, an additional parameter called arm-angle is needed.

The parameter *Arm-Angle Definition* controls how the arm-angle is defined.

Users are advised to always use the new arm-angle definition. The old definition is kept only for backwards compatibility and can in some cases lead to non-optimal movements of the robot.

Limitations

Arm-Angle Definition is only applicable for 7-axis robots.

**Note**

Arm-Angle Definition parameter is not supported in RW 6.07 or later versions.

Allowed values

New or Old.

The default value is New.

Related information

Product manual for the robot.

4 System parameters

4.3.5 Arm-Angle Reference Direction

4.3.5 Arm-Angle Reference Direction

Parent

Arm-Angle Reference Direction belongs to the type *Robot*, in the topic *Motion*.

Description

Arm-Angle Reference Direction controls how the arm-angle property is calculated and affects the location of certain singularities for seven-axis robots.

Usage

In addition to position and orientation, seven-axis robots also depend on the arm-angle concept to fully specify a `robtarget`.

The calculation of the arm-angle depends on a chosen reference direction, and by default this reference direction is chosen as the line passing through axis 2 origin of the robot and being parallel with the Y-axis of the world frame. When the WCP is on the axis chosen as the reference direction, the arm-angle becomes undefined. Hence, the inverse kinematics is singular for all positions with the WCP on the line, and linear movement on and across this line will not work.

If linear movement in this area of the workspace is important for your application, then you can configure the robot to use another reference direction. The choices available are: the world Y-axis, the world Z-axis, and the line passing through axis 1 of the robot.



Note

A RAPID program created with one value for this parameter will behave differently or maybe not work at all if the parameter value is changed.

Allowed values

Arm-Angle Reference Direction can have the following values:

Value:	Name:	Description:
0	World Y	Reference direction parallel with the Y-axis of the world frame.
1	World Z	Reference direction parallel with the Z-axis of the world frame.
2	Axis 1	Reference direction parallel with a line passing through axis 1 of the robot.

The default value is 0.

Related information

Product manual for the robot.

4.3.6 Limit avoidance distance

Parent

Limit avoidance distance belongs to the type *Robot*, in the topic *Motion*.

Description

Limit avoidance distance controls the distance to the nearest singularity or joint limit when automatically adjusting the arm-angle.

Usage

The singularities that can be handled are where axis 2 or axis 5 is equal to zero.

Allowed values

A value between -1 to 100 radians.

The default value is 0.017453 radians.

Setting a negative value will disable the functionality.

Related information

Product manual for the robot.

4 System parameters

4.3.7 Collision Detection Memory

4.3.7 Collision Detection Memory

Parent

Collision Detection Memory belongs to the type *Motion Supervision*, in the topic *Motion*.

Description

Collision Detection Memory defines how much the robot moves back on the path after a collision.

The parameter requires a restart of the controller when modified.

Usage

The movement of robot back on the path after a collision is specified in seconds. If the robot was moving quickly before the collision, it will move further back than if the speed was lower. For detailed information, see the *Application manual - Controller software IRC5*.

Allowed values

A value in the interval 0 to 0.5, specifying the movement in seconds.

For the IRB 14000 robots the default value is 0 s and hence the robot does not back off.

Setting the value to 0 s (disabling backing after collision) may leave the robot in a state with residual forces remaining after a collision. This could trigger new collisions when trying to move away from that position. To move away robustly after a collision, the following are some of the recommended solutions:

- Enable lead-through for a short period of time to release the tension.
 - Set the value of `MotionSup\` to `Off` before executing the `move` instructions.
 - Use `ContactL` instead of `MoveL`.
-

Related information

See *How to tune the motion supervision* in *Technical reference manual - System parameters*.

Application manual - Controller software IRC5

4.3.8 Friction comp. lead through factor

Parent

Friction comp. lead through factor belongs to the type *Robot*, in the topic *Motion*.

Description

Friction comp. lead through factor determines how soft a robot should be in lead through mode.

Usage

A higher value makes the robot softer in lead through mode and a lower value makes the robot less soft.

Setting a high value can make the robot sensitive to errors such as wrong payload in the tool definition. The robot can then start to drift by itself.

Setting the value to 0 removes all friction compensation in lead through mode.



Note

This parameter does not need a reboot to apply the changes. Hence the tests of different levels can be done directly after changing the parameter value.

Limitations

Friction comp. lead through factor is only used for the following robots:

- IRB 14000
- IRB 14050

Configuring this parameter in any other robot will not have any effect.

Allowed values

A value between 0.0 and 1.0.

Default value is 0.6.

This page is intentionally left blank

Index

A

arm mode, 21
axes, 17

B

base coordinate system, 18

C

calibration
 when to do a fine calibration, 37
 when to update the revolution counters, 37
calibration position
 jogging to, 46
Cartesian speed supervision, 51–52
collision avoidance, 35
Collision Avoidance, 28
coordinate systems, 17

F

fieldbus, 35
FlexPendant, 12
 jogging to calibration position, 46
 MoveAbsJ instruction, 46
 overview, 12

I

I/O signals, 35
industrial networks, 35

J

jogging, 19
jogging window, 19
joystick directions, 17

L

lead-through, 24
load, 25

M

MoveAbsJ instruction, 46

O

operator panel, 27

P

payload, 25

Q

Quickset menu, 20

R

RobotStudio
 overview, 13
RobotStudio Oline Apps, 14
RobotStudio Online Apps
 Calibrate, 14
 Jog, 14
 Manage, 14
 Operate, 15
 Tune, 15
 YuMi, 15

S

system parameters
 Connection Timeout Multiplier, 49

U

UAS, user authorization system, 36



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