

# Operating manual IRB 14000

Trace back information:  
Workspace R15-1 version a12  
Checked in 2015-05-27  
Skribenta version 4.6.058

# **Operating manual**

**IRB 14000**

**RobotWare 6.01**

**Document ID: 3HAC052986-001**

**Revision: -**

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, damages 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.

Additional copies of this manual may be obtained from ABB.

The original language for this publication is English. Any other languages that are supplied have been translated from English.

© Copyright 2015 ABB. All rights reserved.

ABB AB  
Robotics Products  
Se-721 68 Västerås  
Sweden

# Table of contents

Overview of this manual .....	7
Product documentation, IRC5 .....	8
<b>1 Safety</b> .....	<b>11</b>
1.1 About this chapter .....	11
1.2 Applicable safety standards .....	12
1.3 Safety terminology .....	13
1.3.1 Safety signals in the manual .....	13
1.3.2 What is an emergency stop? .....	15
1.3.3 What is a safety stop or protective stop? .....	16
1.3.4 What is Cartesian speed supervision? .....	17
1.4 How to deal with an emergency .....	18
1.4.1 Stop the system .....	18
1.4.2 Recover from emergency stops .....	19
1.4.3 Extinguishing fires .....	20
1.5 Working in a safe manner .....	21
1.5.1 Overview .....	21
1.5.2 For your own safety .....	22
1.5.3 Handling of the FlexPendant .....	23
1.5.4 Safety in manual mode .....	25
1.5.5 Safety in automatic mode .....	26
<b>2 Introduction to the IRB 14000 robot system</b> .....	<b>27</b>
2.1 What is IRB 14000? .....	27
2.2 What is a FlexPendant? .....	28
2.3 What is RobotWare? .....	29
2.4 What is RobotStudio? .....	30
2.5 What is RobotStudio Online? .....	31
<b>3 Using the IRB 14000</b> .....	<b>33</b>
3.1 Axes and coordinate systems .....	33
3.2 Jogging .....	35
3.2.1 What is jogging? .....	35
3.2.2 Motion modes .....	37
3.2.3 Coordinated jogging .....	38
3.3 Operating modes .....	40
3.4 Collision avoidance .....	42
3.5 Programming and testing .....	43
3.6 I/O signals .....	44
3.7 User authorization .....	45
<b>4 Calibration</b> .....	<b>47</b>
4.1 Introduction .....	47
4.2 Calibration scale and correct axis position .....	48
4.3 Updating revolution counters .....	50
4.4 Checking the calibration position .....	55
<b>5 RAPID reference information</b> .....	<b>57</b>
5.1 Instructions .....	57
5.1.1 ContactL - Linear contact movement .....	57
<b>6 System parameters</b> .....	<b>63</b>
6.1 Introduction .....	63
6.2 Topic I/O System .....	64
6.2.1 Collision Avoidance .....	64

**Table of contents**

---

6.3	Topic Motion .....	65
6.3.1	Coll-Pred Safety Distance .....	65
6.3.2	Global Speed Limit .....	66
6.3.3	Arm Check Point Speed Limit .....	67
6.3.4	Arm-Angle Reference Direction .....	68
<b>Index</b> .....		<b>69</b>

---

# Overview of this manual

## About this manual

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

## Usage

This manual should be used during operation.

## 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 - Getting started, IRC5 and RobotStudio</i>	3HAC027097-001
<i>Operating manual - RobotStudio</i>	3HAC032104-001
<i>Operating manual - Trouble shooting 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
<i>Application manual - MultiMove</i>	3HAC050961-001

## Revisions

Revision	Description
-	Released with RobotWare 6.01.

# Product documentation, IRC5

---

## Categories for user documentation from ABB Robotics

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

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

---

## Product manuals

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

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

---

## Technical reference manuals

The technical reference manuals describe reference information for robotics products.

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

*Continues on next page*



---

**Application manuals**

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

An application manual generally contains information about:

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

---

**Operating manuals**

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

The group of manuals includes (among others):

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

**This page is intentionally left blank**

# 1 Safety

## 1.1 About this chapter

---

### Introduction to safety

This chapter describes safety principles and procedures to be used when a robot or robot system is operated.

It does not cover how to design for safety nor how to install safety related equipment. These topics are covered in the Product Manuals supplied with the robot system.

# 1 Safety

## 1.2 Applicable safety standards

## 1.2 Applicable safety standards

### Standards, EN ISO

The robot system is designed in accordance with the requirements of:

Standard	Description
EN ISO 12100	Safety of machinery - Basic concepts, general principles for design
EN ISO 13849-1	Safety of machinery, safety related parts of control systems - Part 1: General principles for design
EN ISO 13850	Safety of machinery - Emergency stop - Principles for design
EN ISO 10218-1	Robots for industrial environments - Safety requirements - Part 1 Robot
EN ISO 9787	Manipulating industrial robots, coordinate systems, and motion nomenclatures
EN ISO 9283	Manipulating industrial robots, performance criteria, and related test methods
EN ISO 14644-1 <sup>i</sup>	Classification of air cleanliness
EN IEC 61000-6-4 (option 129-1)	EMC, Generic emission
EN IEC 61000-6-2	EMC, Generic immunity
EN IEC 60204-1	Safety of machinery - Electrical equipment of machines - Part 1 General requirements
IEC 60529	Degrees of protection provided by enclosures (IP code)

<sup>i</sup> Only robots with protection Clean Room.

### European standards

Standard	Description
EN 614-1	Safety of machinery - Ergonomic design principles - Part 1: Terminology and general principles
EN 574	Safety of machinery - Two-hand control devices - Functional aspects - Principles for design
EN 953	Safety of machinery - General requirements for the design and construction of fixed and movable guards

### Other standards

Standard	Description
ANSI/RIA R15.06	Safety requirements for industrial robots and robot systems
ANSI/UL 1740	Safety standard for robots and robotic equipment
CAN/CSA Z 434-03	Industrial robots and robot Systems - General safety requirements

## 1.3 Safety terminology

### 1.3.1 Safety signals in the manual






#### Introduction to safety signals

This section specifies all dangers that can arise when doing the work described in the user manuals. Each danger consists of:

- A caption specifying the danger level (DANGER, WARNING, or CAUTION) and the type of danger.
- A brief description of what will happen if the operator/service personnel do not eliminate the danger.
- Instruction about how to eliminate danger to simplify doing the work.

#### Danger levels

The table below defines the captions specifying the danger levels used throughout this manual.

Symbol	Designation	Significance
 xx0200000022	DANGER	Warns that an accident <i>will</i> occur if the instructions are not followed, resulting in a serious or fatal injury and/or severe damage to the product. It applies to warnings that apply to danger with, for example, contact with high voltage electrical units, explosion or fire risk, risk of poisonous gases, risk of crushing, impact, fall from height, and so on.
 xx0100000002	WARNING	Warns that an accident <i>may</i> occur if the instructions are not followed that can lead to serious injury, possibly fatal, and/or great damage to the product. It applies to warnings that apply to danger with, for example, contact with high voltage electrical units, explosion or fire risk, risk of poisonous gases, risk of crushing, impact, fall from height, etc.
 xx0200000024	ELECTRICAL SHOCK	Warns for electrical hazards which could result in severe personal injury or death.
 xx0100000003	CAUTION	Warns that an accident may occur if the instructions are not followed that can result in injury and/or damage to the product. It also applies to warnings of risks that include burns, eye injury, skin injury, hearing damage, crushing or slipping, tripping, impact, fall from height, etc. Furthermore, it applies to warnings that include function requirements when fitting and removing equipment where there is a risk of damaging the product or causing a breakdown.
 xx0200000023	ELECTROSTATIC DISCHARGE (ESD)	Warns for electrostatic hazards which could result in severe damage to the product.



*Continues on next page*

# 1 Safety

---

## 1.3.1 Safety signals in the manual

*Continued*

Symbol	Designation	Significance
 xx0100000004	NOTE	Describes important facts and conditions.
 xx01000000098	TIP	Describes where to find additional information or how to do an operation in an easier way.

### 1.3.2 What is an emergency stop?

#### Definition of emergency stop

An emergency stop is a state that takes precedence over all other robot controls, removes drive power from the robot actuators, remains active until it is reset, and can only be reset by manual action.

An emergency stop state means that all power is disconnected from the robot except for the manual brake release circuits. You must perform a recovery procedure, that is, resetting the emergency stop button and pressing the Motors On button on the FlexPendant, to return to normal operation.

The robot system is configured so that the emergency stop results in a category 0 stop, immediately stopping the robot actions by disconnecting power from the motors.

**Note**

The emergency stop function may only be used for the purpose and under the conditions for which it is intended.

**Note**

The emergency stop function is intended for immediately stopping equipment in the event of an emergency.

**Note**

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

For how to perform normal program stops, see section *Stopping programs* in *Operating manual - IRC5 with FlexPendant*.

#### Emergency stop buttons

In a robot system several emergency stop buttons can be installed and operated in order to achieve an emergency stop. By default there is one emergency stop button available on the FlexPendant. There can also be other types of emergency stops on your robot. Consult your plant or cell documentation to see how your robot system is configured.

### 1.3.3 What is a safety stop or protective stop?

---

#### Definition of safety stops

A safety stop is a state that stops all robot motion and removes power to the robot drive actuators. There is no recovery procedure. You need only to restore motor power to recover from a safety stop. Safety stop is also called protective stop.

The robot system is configured so that the safety stop results in a category 0 stop, immediately stopping the robot actions by disconnecting power from the motors.



#### Note

For IRB 14000, the safety stop is only applicable in the external devices mode, with the safety bridge connector removed.



#### Note

The safety stop function may only be used for the purpose and under the conditions for which it is intended.



#### Note

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

For how to perform normal program stops, see section *Stopping programs* in *Operating manual - IRC5 with FlexPendant*.



### 1.3.4 What is Cartesian speed supervision?

---

#### Definition of Cartesian speed supervision

The Cartesian speed supervision is a safety function that supervises the Cartesian speed of the elbow (arm check point, ACP) and the wrist (wrist center point, WCP). The default speed limit can be modified if needed, based on the risk assessment for the robot installation. If any of the configured speed limits are exceeded, then the robot motion is stopped and a message is displayed to the user.

The Cartesian speed supervision is active in both manual and automatic mode. The setting is defined by system parameters.

# 1 Safety

---

## 1.4.1 Stop the system

## 1.4 How to deal with an emergency

### 1.4.1 Stop the system

---

#### Overview

Press the emergency stop button if you need to stop the robot and its external equipment to protect equipment or personnel.

---

#### The FlexPendant emergency stop button



xx1400001445

A	Emergency stop button
---	-----------------------

---

#### Other emergency stop devices

The plant designer may have placed additional emergency stop devices in convenient places. Refer to your plant or cell documentation to find out where these are placed.

## 1.4.2 Recover from emergency stops

### Overview

Recovering from an emergency stop is a simple but important procedure. This procedure ensures that the manipulator system is not returned to production while maintaining a hazardous condition.

### Reset the latch of emergency stop buttons

All push-button style emergency stop devices have a latching feature that must be released in order to remove the emergency stop condition of the device.

In many cases this is done by twisting the push-button as marked, but there are also devices where you pull the button to release the latch.

### Reset automatic emergency stop devices

All automatic emergency stop devices also have some kind of latching feature that must be released. Please consult your plant or cell documentation to see how your manipulator system is configured.

### Recover from emergency stops

	Action
1	Make sure the hazardous situation that resulted in the emergency stop condition no longer exists.
2	Locate and reset the device or devices that gave the emergency stop condition.
3	Press the Motors On button on the Quickset menu on the FlexPendant to recover from the emergency stop condition.

# 1 Safety

---

## 1.4.3 Extinguishing fires

### 1.4.3 Extinguishing fires

---

#### Precautions

In case of a fire always make sure both you and your coworkers are safe before performing any fire extinguishing activities. In case of injury always make sure these are treated first.

---

#### Select fire extinguisher

Always use carbon dioxide extinguishers when extinguishing fires in electrical equipment such as the robot or the controller. Do not use water or foam.

## 1.5 Working in a safe manner

### 1.5.1 Overview

---

#### About this section

In this section some most basic rules of conduct for you as a robot system user are suggested. However, it is impossible to cover each and every specific situation.

# 1 Safety

---

## 1.5.2 For your own safety

### 1.5.2 For your own safety

---

#### Disconnected jogging device

Always put away a disconnected jogging device safe from any robot cell or controller to avoid that a disconnected unit is used when trying to prevent a hazardous situation.



#### CAUTION

A disconnected jogging device should be stored in such a way that it cannot be mistaken for being connected to the controller.

---

#### Custom jogging device connections

Any means of connecting the jogging device with other than the supplied cable and its standard connector must not render the emergency stop button inoperative. Always test the emergency stop button to make sure it works if a custom connection cable is used.

---

#### Personal protective equipment

Always use suitable personal protective equipment, based on the risk assessment for the robot installation.

### 1.5.3 Handling of the FlexPendant

#### Handling of the FlexPendant

The FlexPendant is a high-quality handheld terminal equipped with highly sensitive state-of-the-art electronics. To avoid malfunctions or damage through improper handling, follow these instructions during operation.

The FlexPendant may only be used for the purposes mentioned in this manual. The FlexPendant was developed, manufactured, tested and documented in accordance with applicable safety standards. If you follow the instructions regarding safety and use as described in this manual, the product will, in the normal case, neither cause personal injury nor damage to machinery and equipment.

#### Handling and cleaning

- Handle with care. Do not drop, throw, or give the FlexPendant strong shock. It can cause breakage or failure.
- If the FlexPendant is subjected to shock, always verify that the safety functions (enabling device and emergency stop) work and are not damaged.
- When not using the device, hang it on the wall bracket provided for storage so it does not accidentally fall.
- Always use and store the FlexPendant in such a way that the cable does not become a tripping hazard.
- Never use sharp objects (such as screwdriver or pen) for operating the touch screen. This could damage the touch screen. Instead use your finger or a stylus (located on the back on FlexPendant with USB port).
- Clean the touch screen regularly. Dust and small particles can clog the touch screen and cause it to malfunction.
- Never clean the FlexPendant with solvents, scouring agent, or scrubbing sponges. Use a soft cloth and a bit of water or mild cleaning agent.  
*See Product manual - IRC5, section Cleaning the FlexPendant.*
- Always close the protective cap on the USB port when no USB device is connected. The port can break or malfunction if exposed to dirt or dust.



#### CAUTION

A disconnected jogging device should be stored in such a way that it cannot be mistaken for being connected to the controller.

#### Cabling and power supply

- Turn off the power supply before opening the cable entrance area of the FlexPendant. Otherwise the components could be destroyed or undefined signals could occur.
- Make sure that nobody trips over the cable to prevent the device from falling to the ground.
- Take care not to squeeze and thus damage the cable with any object.

*Continues on next page*

# 1 Safety

---

## 1.5.3 Handling of the FlexPendant

### Continued

- Do not lay the cable over sharp edges since this can damage the cable sheath.

---

### Enabling device



#### Note

The FlexPendant is always equipped with an enabling device, but for the IRB 14000 system the enabling device is not used. Therefore the enabling device is disabled and inactive when the FlexPendant is connected to an IRB 14000 system, but it is enabled and active when connected to another robot.

The enabling device is a manually operated, constant pressure push-button which, when continuously activated in one position only, allows potentially hazardous functions but does not initiate them. In any other position, hazardous functions are stopped safely.

The enabling device is of a specific type where you must press the push-button only half-way to activate it. In the fully in and fully out positions, operating the robot is impossible.



#### Note

To ensure safe use of the jogging device, the following must be implemented:

- The enabling device must never be rendered inoperational in any way.
- During programming and testing, the enabling device must be released as soon as there is no need for the robot to move.
- Anyone entering the working space of the robot must always bring the jogging device with him/her. This is to prevent anyone else from taking control of the robot without his/her knowledge.

---

### Waste disposal

Observe the national regulations when disposing of electronic components! When replacing components, please dispose of used components properly.



### 1.5.4 Safety in manual mode

---

#### What is the manual mode?

In manual mode the manipulator movement is under manual control.

The manual mode is used when creating and verifying programs, and when commissioning a manipulator system.

#### Active safeguard mechanisms

The safety stop mechanism and the Cartesian speed supervision are always active, both while operating in manual and automatic mode.

#### Operating speed

In manual reduced speed mode the movement is limited to 250 mm/s.

# 1 Safety

---

## 1.5.5 Safety in automatic mode

### 1.5.5 Safety in automatic mode

---

#### What is the automatic mode?

The automatic mode is used for running the robot program in production.

---

#### Active safeguard mechanisms

The safety stop mechanism and the Cartesian speed supervision are always active, both while operating in manual and 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.

---

#### 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.

## 2 Introduction to the IRB 14000 robot system

### 2.1 What is IRB 14000?

#### The IRB 14000 robot

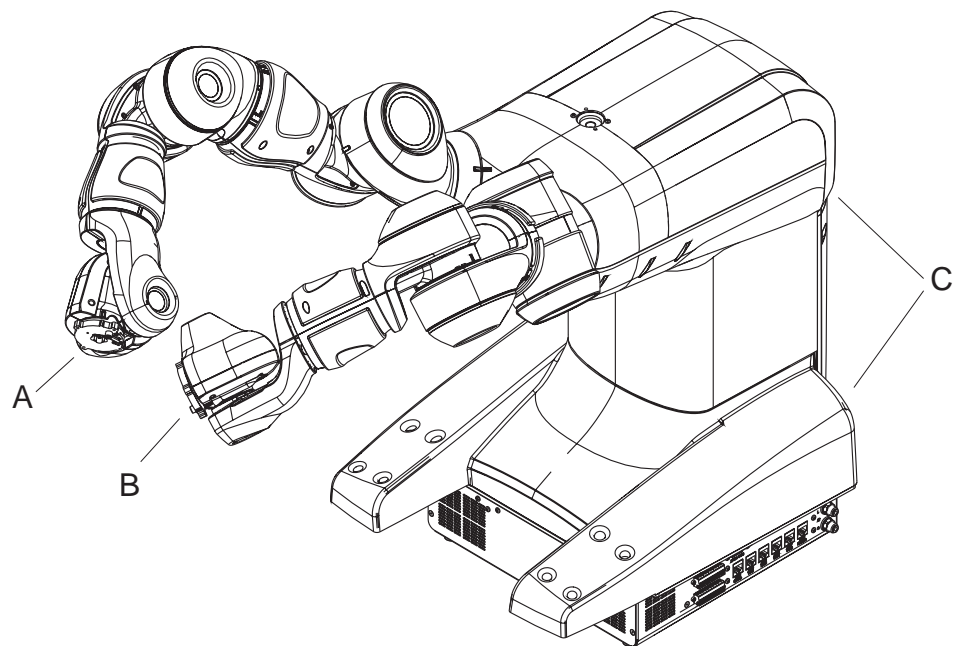
The IRB 14000 robot is a two-armed industrial robot with integrated controller. Each arm has seven axes, which gives an extra degree of freedom compared to traditional 6-axis robots.

#### 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



xx1500000008

A	Right arm
B	Left arm
C	Integrated controller

## 2 Introduction to the IRB 14000 robot system

---

### 2.2 What is a FlexPendant?

### 2.2 What is a FlexPendant?

---

#### Introduction to the FlexPendant

The FlexPendant is a hand held operator unit used to perform many of the tasks involved when operating a robot system, such as running programs, jogging the manipulator, modifying robot programs and so on.

The FlexPendant consists of both hardware and software and is a complete computer in itself.

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



xx1500000702

## 2.3 What is RobotWare?

### Introduction to RobotWare

RobotWare is a family of software products from ABB Robotics. The products are designed to make you more productive and lower your cost of owning and operating a robot. ABB Robotics has invested many years into the development of these products and they represent knowledge and experience based on several thousands of robot installations.

### Product classes

Within the RobotWare family, there are different classes of products:

Product classes	Description
RobotWare-OS	<p>This is the operating system of the robot. RobotWare-OS provides all the necessary features for fundamental robot programming and operation. It is an inherent part of the robot, but can be provided separately for upgrading purposes.</p> <p>For a description of RobotWare-OS, see <i>Product specification - Controller IRC5</i>.</p>
RobotWare options	<p>These products are options that run on top of RobotWare-OS. They are intended for robot users that need additional functionality for motion control, communication, system engineering, or applications.</p> <p>For more information on RobotWare options, see <i>Product specification - Controller software IRC5</i>.</p> <p>For more detailed information on RobotWare options, see <i>Application manual - Controller software IRC5</i>. Note that not all RobotWare options are described in this manual. Some options are more comprehensive and are therefore described in separate manuals.</p>
Process application options	<p>These are extensive packages for specific process application like spot welding, arc welding, and dispensing. They are primarily designed to improve the process result and to simplify installation and programming of the application.</p> <p>The process application options are all described in separate manuals. For more information, see <i>Product specification - Controller software IRC5</i>.</p>
RobotWare Add-ins	<p>A RobotWare Add-in is a self-contained package that extends the functionality of the robot system.</p> <p>Some software products from ABB Robotics are delivered as Add-ins. For example track motion IRBT, positioner IRBP, and stand alone controller.</p> <p>The purpose of RobotWare Add-ins is also that a robot program developer outside of ABB can create options for the ABB robot systems, and sell the options to their customers. For more information on creating RobotWare Add-ins, contact your local ABB Robotics representative at <a href="http://www.abb.com/contacts">www.abb.com/contacts</a>.</p>

### 2.4 What is RobotStudio?

---

#### Overview

RobotStudio is an engineering tool for 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 your 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 the following operations when connected to a real controller:

- 1 Managing RobotWare 6 controllers, using the Installation Manager
- 2 Text-based programming and editing, using the **RAPID Editor**.
- 3 File manager for the controller.
- 4 Administrating the User Authorization System.

### 2.5 What is RobotStudio Online?

#### Introduction to RobotStudio Online

RobotStudio Online is a suite of **Windows Store** applications intended to run on **Windows 8.1** tablets. It provides functionality for shop floor commissioning of robot systems.



#### Note

Some of the functionality requires use of a safety device such as the T10 jogging device or the JSHD4 three position safety device. For more information on T10, see *Operating manual - IRC5 with T10*.



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.

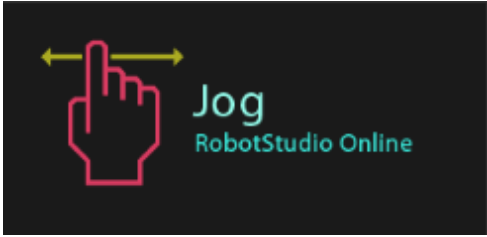
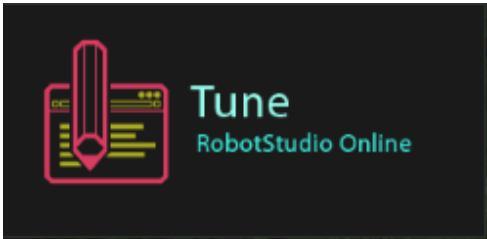

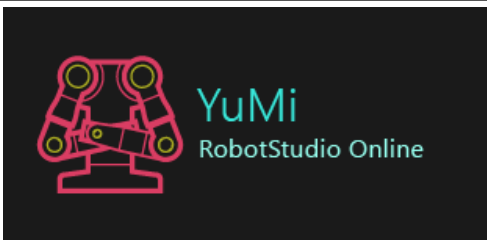
RobotStudio Online Apps	Description
 xx1400002047	<b>Manage</b> is a tool to manage IRC5 controllers on a network.
 xx1400002049	<b>Calibrate</b> is a tool for calibration and definition of frames with IRC5 controllers.

*Continues on next page*

## 2 Introduction to the IRB 14000 robot system

### 2.5 What is RobotStudio Online?

*Continued*

RobotStudio Online Apps	Description
 xx1400002048	<b>Jog</b> is a tool for manual positioning (moving or jogging) with IRC5 controllers.
 xx1400002050	<b>Tune</b> is a tool for shop floor editing of RAPID programs with IRC5 controllers.
 xx1400002511	<b>Operate</b> is a tool used in production to view the program code.
 xx1500000832	<b>YuMi</b> is a tool for programming of the new dual arm and collaborative robot YuMi, IRB 14000, from ABB. It will help the users to get a fast introduction to robot programming using lead-thru and graphical programming.



## 3 Using the IRB 14000

### 3.1 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.

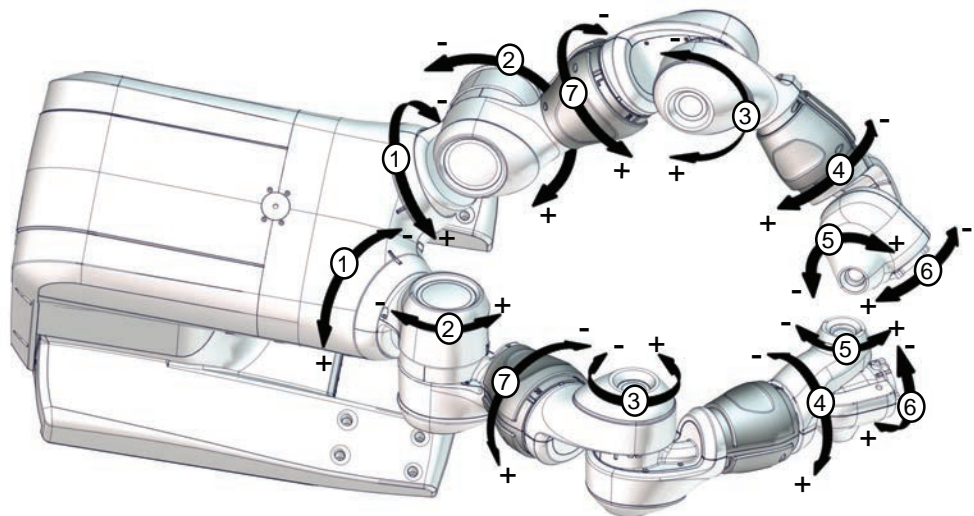
A robot uses several coordinate systems, each suitable for specific types of jogging or programming.

- 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.



xx1500000254

*Continues on next page*

## 3 Using the IRB 14000

### 3.1 Axes and coordinate systems

*Continued*



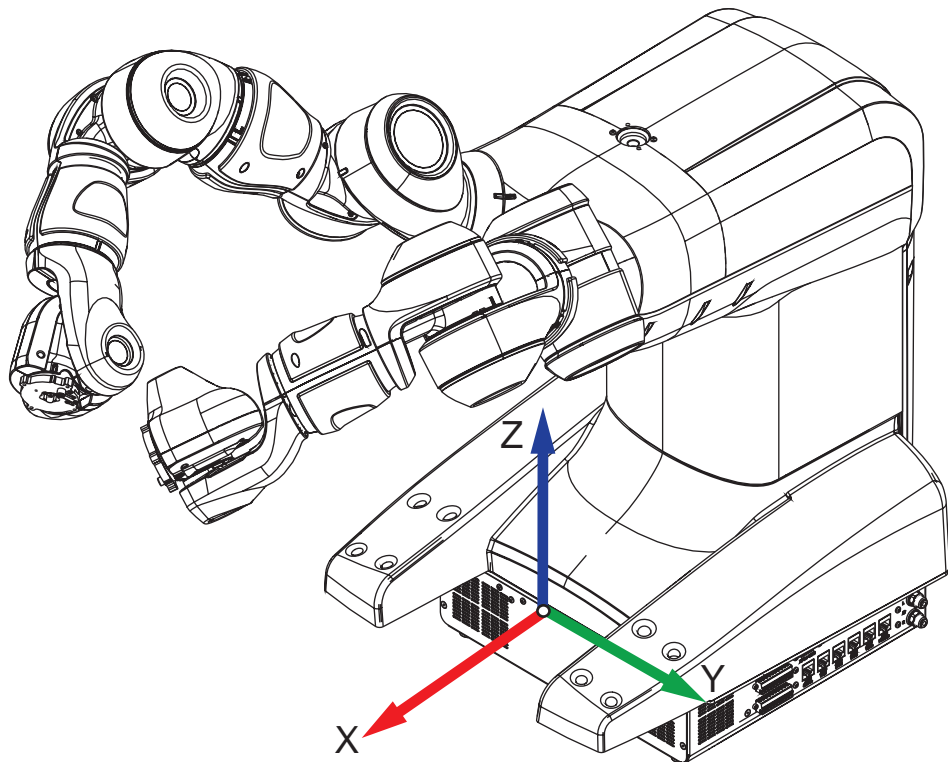
#### Note

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

## 3.2 Jogging

### 3.2.1 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.

xx1500000006



#### Tip

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

*Continues on next page*

### 3 Using the IRB 14000

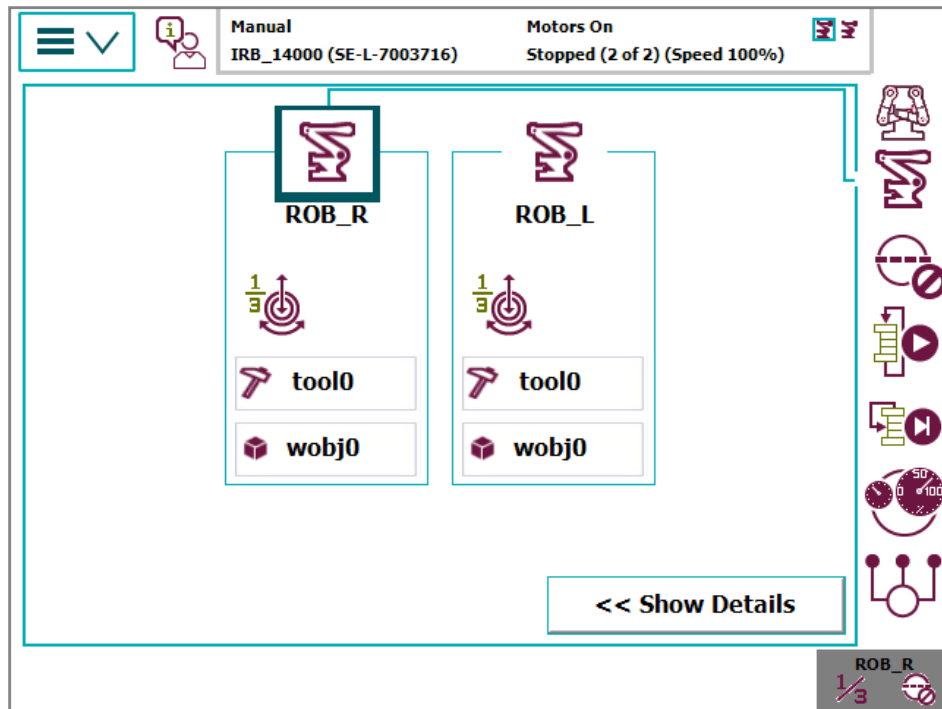
#### 3.2.1 What is jogging?

*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.



xx1500000004



#### 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 40](#).

### 3.2.2 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.

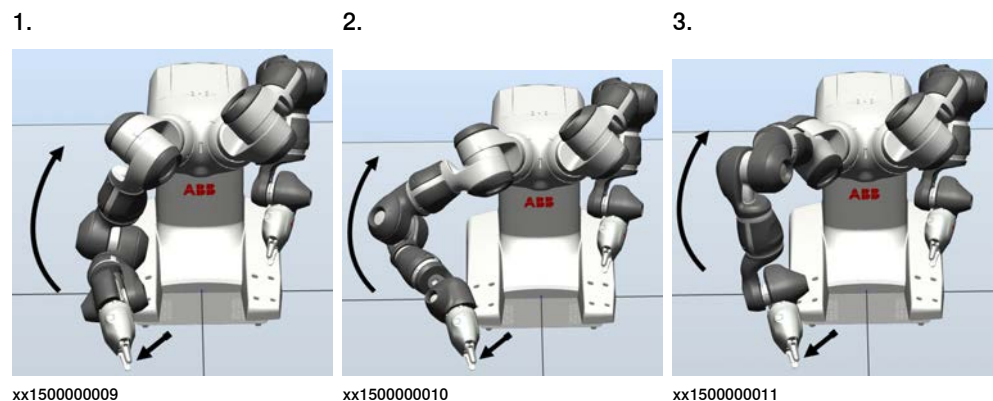
- In *arm mode*, both the tool center point and the orientation of the tool is fixed in space and only the angle of the arm is changed. The tool center point is neither rotated nor moved. See [Jogging in arm mode on page 37](#).

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 the both the tool center point and the orientation of the tool is 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.



## 3 Using the IRB 14000

### 3.2.3 Coordinated jogging

### 3.2.3 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. Typically the arm that is holding the work piece.	
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 left arm is holding the work object and the right arm is moving it: <pre>PERS wobjdata wobjLeft :=     [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.	<a href="#">Activate coordinated jogging on page 38</a>

#### 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.	<a href="#">The Quickset menu on page 36</a>
2	Activate the previously created work object.	<a href="#">Setup coordinated jogging on page 38</a>
3	Select the work object coordinate system.	
4	Select the other arm.	The coordinated arm is now indicated with a flashing frame.
5	Jog the arm, the other will follow.	

*Continues on next page*

---

#### 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.

## 3 Using the IRB 14000

### 3.3 Operating modes

### 3.3 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.

#### What is the manual mode?

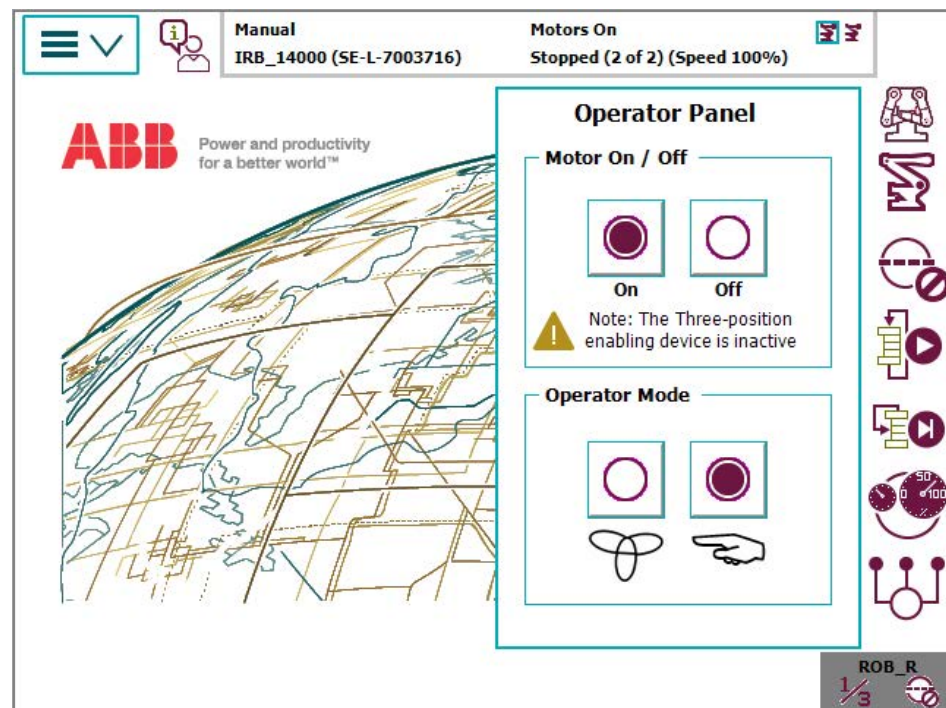
In manual mode the manipulator movement is under manual control. The manual mode is used when programming and for program verification.

#### 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.

#### 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 36](#).



xx1500000005

*Continues on next page*





#### 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 44](#).

## 3 Using the IRB 14000

---

### 3.4 Collision avoidance

### 3.4 Collision avoidance

---

#### Introduction

The IRB 14000 has a built in functionality called *Collision Avoidance* that is active both during jogging and when running programs. Collision avoidance monitors the geometrical models of the robot arms and the body, and the robot stops if any of the parts get too close to each other.

#### System parameters

The default safety distance can be set by the system parameter *Coll-Pred Safety Distance*.

For more information, see the system parameter [Coll-Pred Safety Distance on page 65](#).

#### System input

If the robot has already collided or is within the default safety distance, it is possible to temporary disable the collision avoidance functionality by resetting the system input *Collision Avoidance*.

For more information, see the system parameter [Collision Avoidance on page 64](#).

## 3.5 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*.

---

### 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.

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 [ContactL - Linear contact movement on page 57](#)

## 3 Using the IRB 14000

---

### 3.6 I/O signals

### 3.6 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*.

---

#### 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.

## 3.7 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 63](#).

**This page is intentionally left blank**

## 4 Calibration

### 4.1 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 50](#).

---

#### 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 section *Calibration* in *Product manual - IRB 14000*.

---

## 4 Calibration

### 4.2 Calibration scale and correct axis position

### 4.2 Calibration scale and correct axis position

#### Introduction

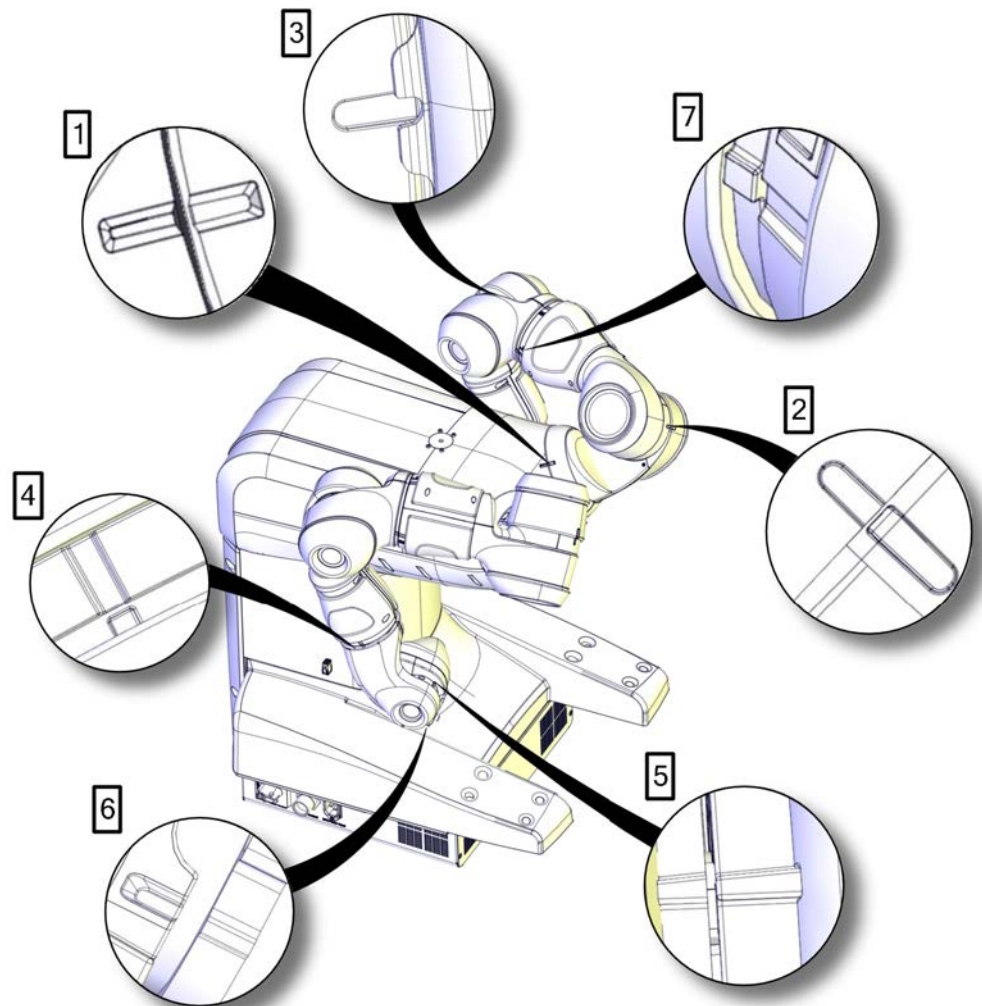
This section specifies the calibration scale positions and/or correct axis positions.

#### 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.

There are two calibration marks on each axis on the robot arm, one valid if the arm is installed as a right arm (R) and one that is valid if the arm is installed as a left arm (L). Use the mark that corresponds to arm installation.



xx1500000526

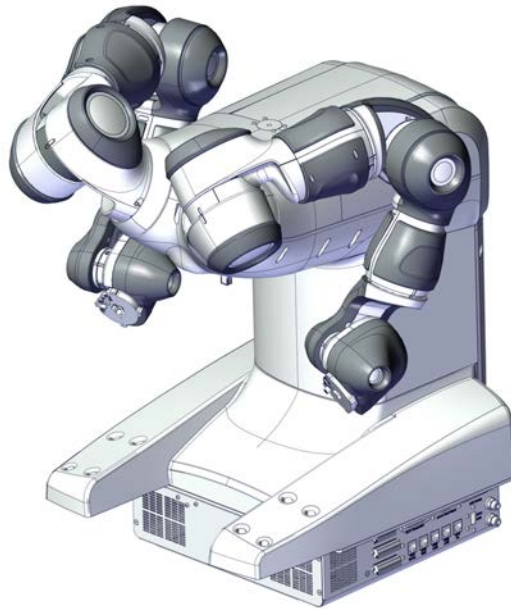
*Continues on next page*



**Calibration position**

Illustration of robot in calibration position

The figure shows the robot in its calibration position.



xx1500000363

**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°

## 4 Calibration

### 4.3 Updating revolution counters

### 4.3 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

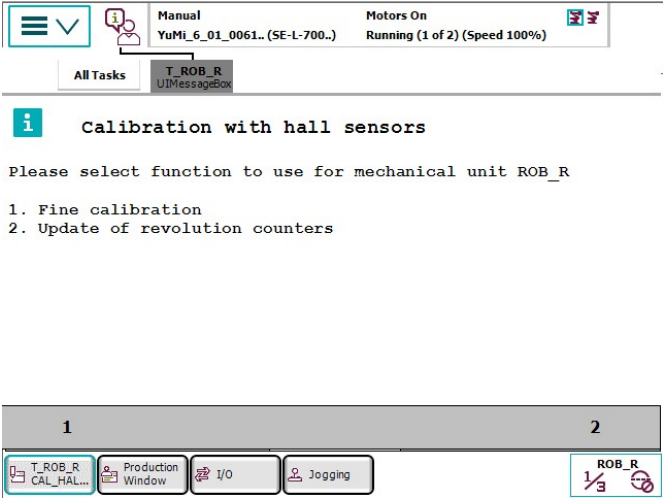
	Action	Note
1	 <b>CAUTION</b> When releasing the holding brakes, the robot axes may move very quickly and sometimes in unexpected ways!	
2	Release the brakes of the robot arm to be calibrated and move the arm manually so that the synchronization mark of each joint is aligned. The robot now stands in its calibration position.	The synchronization marks are shown in <a href="#">Calibration scale and correct axis position on page 48</a> . 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 <b>Program Editor</b> on the FlexPendant.	
2	Select the task that corresponds to the robot arm to be calibrated. Tap <b>Open</b> .	
3	If necessary, create a new program. This needs to be done if no existing program is available.	
4	Select <b>Debug</b> and tap <b>PP to Main</b> .	
5	Select <b>Debug</b> and tap <b>Call Routine...</b>	
6	Select <b>CalHall</b> .	
7	Go to <b>Motor On</b> and press the <b>Start button</b> .	

*Continues on next page*

Step 3 - Selecting the function Update of revolution counters

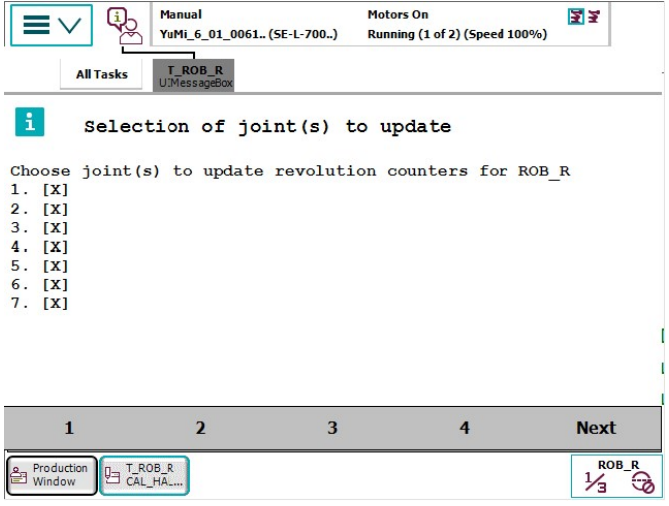
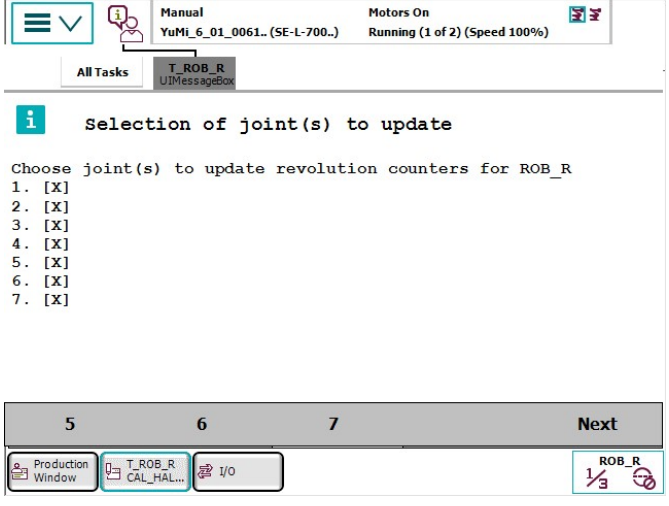
Action
<div>1</div> <div>In the calibration with hall sensors routine (CalHall), tap 2. to select the function of updating the revolution counters</div> <div></div> <div>xx1400002694</div>

Continues on next page

## 4 Calibration


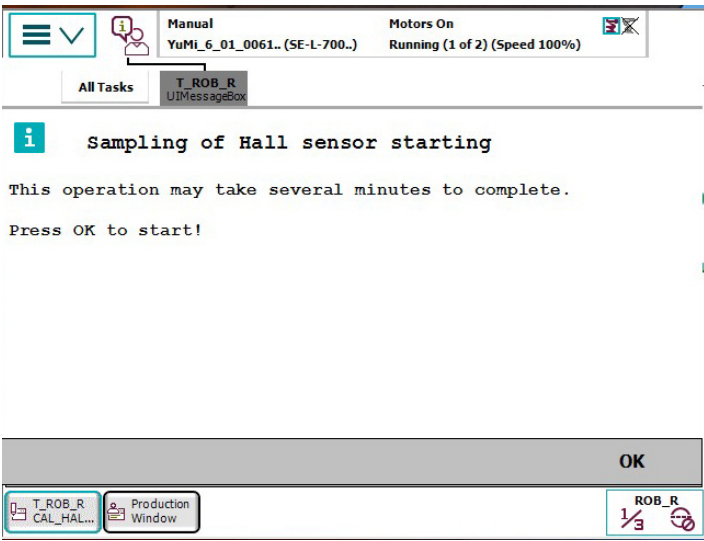
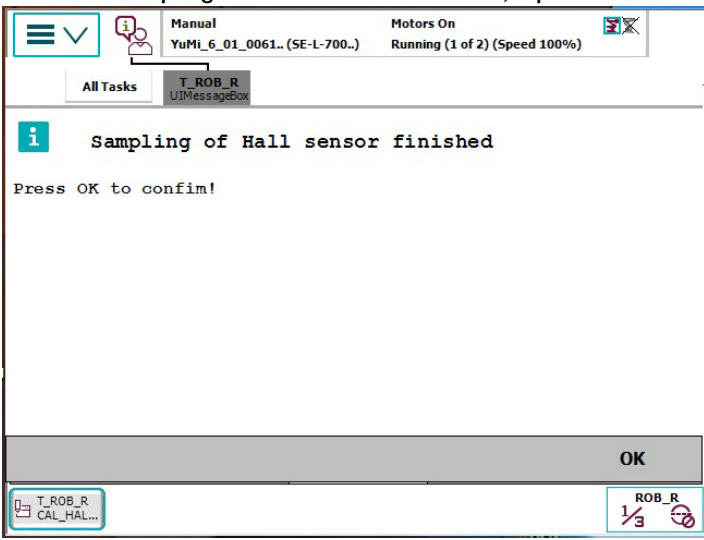
### 4.3 Updating revolution counters

Continued

Action	
2	<p>Tap to select which joint(s) to update revolution counters for.</p> <p>Joints 1, 2, 3 and 4 are selectable in the first window. Tap <b>Next</b> to open the second window where joints 5, 6 and 7 are selectable.</p>  <p>xx1400002695</p>  <p>xx1400002696</p>

Continues on next page

## Step 4 - Storing the revolution counter setting

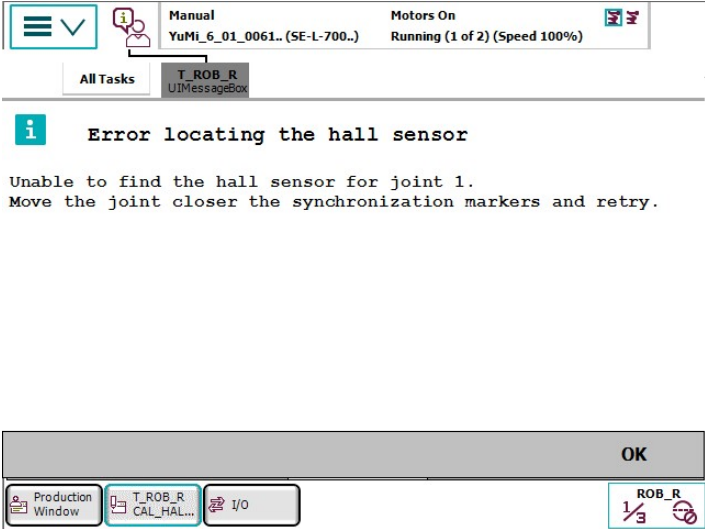
	Action
1	<p>Tap OK to start the sampling procedure of the hall sensor.</p> <p>One at a time, each selected joint now rotates back and forth several times to calculate and find the exact position where the hall sensor is aligned with the magnet.</p> <p>Calibration order: axis 1-2-3-4-5-6-7.</p> <p> <b>Note</b></p> <p>This procedure may take several minutes.</p>  <p>xx1400002697</p>
2	<p>When the sampling of hall sensors is finished, tap OK to confirm.</p>  <p>xx1400002898</p>

Continues on next page

## 4 Calibration

### 4.3 Updating revolution counters

Continued

	Action
3	<p data-bbox="520 313 1326 342">If the hall sensor can not be found, following error message is displayed.</p> <div data-bbox="520 342 1228 869"><p>The screenshot shows the T_ROB_R calibration interface. At the top, there's a status bar with 'Manual', 'YuMi_6_01_0061.. (SE-L-700..)', and 'Motors On Running (1 of 2) (Speed 100%)'. Below this is a tab bar with 'All Tasks' and 'T_ROB_R'. The main area displays an information icon and the title 'Error locating the hall sensor'. The message reads: 'Unable to find the hall sensor for joint 1. Move the joint closer the synchronization markers and retry.' At the bottom right of the message area is an 'OK' button. Below the message area is a row of icons: 'Production Window', 'T_ROB_R CAL_HAL...', 'I/O', and 'ROB_R'.</p></div> <p data-bbox="520 880 627 898">xx1400002698</p> <p data-bbox="520 916 991 943">There can be several causes for this error.</p> <ul data-bbox="555 943 1410 1160" style="list-style-type: none"><li>• The joints are not sufficiently aligned according to the synchronization marks. Move the joints so that the synchronization marks are aligned. Tap <b>OK</b> 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.)</li><li>• The hall sensor is defect. Perform troubleshooting of I/O signals. Possible replacement of hall sensor may be required.</li></ul>

## 4.4 Checking the calibration position

### Introduction

Check 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 <b>Program editor</b> .	
2	Create a new program.	
3	Use <b>MoveAbsJ</b> in the <b>Motion&amp;Proc</b> menu.	
4	Create the following program for the right arm: <pre>MoveAbsJ [[0,-130,30,0,40,0], [- 135,9E9,9E9,9E9,9E9,9E9]] \NoEOffs, v1000, fine, tool0;</pre> Create the following program for the left arm: <pre>MoveAbsJ [[0,-130,30,0,40,0], [135,9E9,9E9,9E9,9E9,9E9]] \NoEOffs, v1000, fine, tool0;</pre>	
5	Run the program in manual mode.	
6	Check that the calibration marks for the axes align correctly. If they do not, update the revolution counters.	See <a href="#">Calibration scale and correct axis position on page 48</a> , and <a href="#">Updating revolution counters on page 50</a> .

### Using the jogging window

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

	Action	Note
1	On the <b>ABB</b> menu, tap <b>Jogging</b> .	
2	Tap <b>Motion mode</b> 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 <a href="#">Exact axis positions in degrees on page 49</a> .
5	Check that the calibration marks for the axes align correctly. If they do not, update the revolution counters!	See <a href="#">Calibration scale and correct axis position on page 48</a> , and <a href="#">Updating revolution counters on page 50</a> .

**This page is intentionally left blank**



## 5 RAPID reference information

### 5.1 Instructions

#### 5.1.1 ContactL - Linear contact movement

##### Usage

**ContactL** (*Contact Linear*) is used to obtain contact with an object at a desired position while moving the tool center point (TCP) linearly.

The collision detection level is raised to its maximum value, and during the movement the robot supervises the internal torque and compares it to a torque level given by the user. When the requested user torque level is reached, the robot performs a stiff stop and continues with the rest of the program.

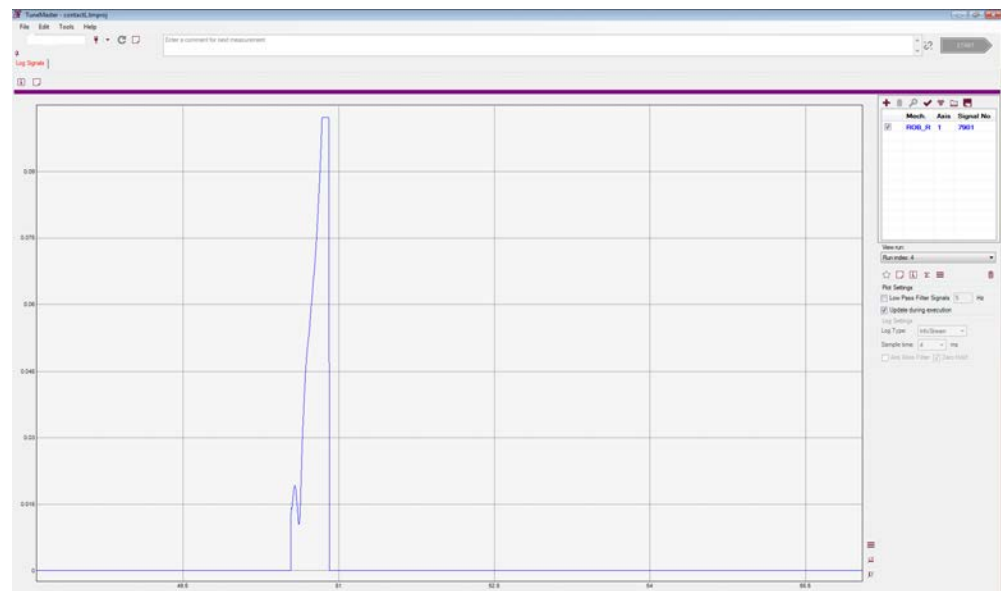
This instruction can typically be used when the tool held by the robot has to press an object into place.

This instruction can only be used in the main task `T_ROB1`, or in Motion tasks when in a *MultiMove* system.

The maximum speed for a **ContactL** instruction is 100 mm/s.

##### Description

To find out the value for the torque level `desiredTorque` it is necessary to test the application and to view an internal test signal, signal 7901, using TuneMaster.



xx1500000649

For more information about TuneMaster, see the help section included in the application.

##### Basic examples

The following examples illustrate the instruction **ContactL**:

*Continues on next page*

## 5 RAPID reference information

---

### 5.1.1 ContactL - Linear contact movement

*Continued*

See also [More examples on page 60](#).

#### Example 1

```
desiredTorque := 0.1;
ContactL \DesiredTorque:=desiredTorque, p10, v100, tool1;
```

The TCP of tool1 is moved linearly towards the position p10 at a speed of v100. When the value of the internal torque level exceeds the desiredTorque level specified by the user, the robot will perform a stiff stop and then the program will continue from the position where the robot is stopped.

The argument DesiredTorque is optional. When DesiredTorque is omitted, the ContactL instruction will only raise the collision detection level to its maximum value, i.e. giving the opportunity to keep pressure on an object while moving the TCP.

If the desiredTorque is not reached when the robot reaches the desired position, there will be an execution error and the system stops with an event log. Therefore it is recommended to implement an error handler for such cases, see [Error handling on page 60](#).

#### Example 2

```
ContactL RelTool (CRobT(),5,5,0), v100, \Zone:=z10, tool1;
```

The robot is moved to a position that is 5 mm from its current position in the x direction and 5 mm from its current position in the y direction of the tool. If the Zone argument is omitted, the ContactL instruction will use a fine-point as default.

In the example the argument DesiredTorque is omitted. The instruction will only raise the collision detection level to its maximum value and the ContactL instruction will function similar to a MoveL instruction.

#### Example 3

```
desiredTorque := 0.9;
ContactL \DesiredTorque:=desiredTorque, p10, v100, tool1;
ContactL RelTool (CRobT(),5,5,0), v100, \Zone:=z10, tool1;
ContactL RelTool (CRobT(),5,5,-10), v100, \Zone:=z10, tool1;
MoveL ...
```

It is important to remember to use the ContactL instruction while in contact, but also when leaving contact. A normal move instruction will most probably trigger the motion supervision.

---

#### Arguments

```
ContactL [\DesiredTorque] ToPoint [\ID] Speed [\Zone] Tool [\WObj]
```

[ \DesiredTorque ]

**Data type:** num

User defined desired torque level.

ContactL will always use a fine-point as zone data for the destination if DesiredTorque is defined. When DesiredTorque is omitted the ContactL instruction will only raise the collision detection level and not supervise the internal torque level.

*Continues on next page*

ToPoint

**Data type:** `robtarget`

The destination point of the robot and external axes. It is defined as a named position or stored directly in the instruction (marked with an \* in the instruction).

[ \ID ]

*Synchronization id*

**Data type:** `identno`

The argument [ \ID ] is mandatory in the MultiMove systems, if the movement is synchronized or coordinated synchronized. This argument is not allowed in any other case. The specified id number must be the same in all the cooperating program tasks. By using the id number the movements are not mixed up at the runtime.

Speed

**Data type:** `speeddata`

The speed data that applies to movements. Speed data defines the velocity for the tool center point, the tool reorientation, and external axes.

[ \Zone ]

**Data type:** `zonedata`

Zone data for the movement. Zone data describes the size of the generated corner path and is only used when `DesiredTorque` is omitted.

If the [ \Zone ] argument is omitted the `ContactL` instruction will use a fine-point as default.

Tool

**Data type:** `tooldata`

The tool in use when the robot moves. The tool center point is the point that is moved to the specified destination position.

[ \WObj ]

*Work Object*

**Data type:** `wobjdata`

The work object (coordinate system) to which the robot position in the instruction is related.

This argument can be omitted and if so then the position is related to the world coordinate system. If, on the other hand, a stationary TCP or coordinated external axes are used then this argument must be specified for a linear movement relative to the work object to be performed.

---

#### Program execution

See the instruction `MoveL` for information about linear movement.

The robot movement stops when the internal torque level has exceeded the user defined torque level, assuming that the argument `DesiredTorque` is defined.

Otherwise the robot movement always continues to the programmed destination point.

*Continues on next page*

## 5 RAPID reference information

---

### 5.1.1 ContactL - Linear contact movement

*Continued*

If the argument `DesiredTorque` is omitted the collision detection level is raised to its maximum value and no supervision of the internal torque level is performed, i.e. giving the opportunity to keep pressure on an object while moving the TCP.

---

#### Error handling

An error is reported during a `ContactL` when:

- `ContactL` reaches the point specified in the argument `ToPoint` without reaching the `DesiredTorque` specified by the user. This generates the error `ERR_CONTACTL`.

Errors can be handled in different ways depending on the selected running mode:

- **Continuous forward/Instruction forward:**  
No position is returned and the movement always continues to the programmed destination point. The system variable `ERRNO` is set to `ERR_CONTACTL` and the error can be handled in the error handler of the routine.
- **Instruction backward:**  
During backward execution the instruction carries out the movement without any torque supervision.

#### Example

```
VAR num desiredTorque;
...
desiredTorque := 0.1;
MoveL p10, v100, fine, tool1;
ContactL \DesiredTorque:=desiredTorque, p20, v100, tool1;
...
ERROR
  IF ERRNO=ERR_CONTACTL THEN
    StorePath;
    MoveL p10, v100, fine, tool1;
    RestoPath;
    ClearPath;
    StartMove;
    RETRY;
  ELSE
    Stop;
  ENDIF
ENDPROC
```

The robot moves from position `p10` to `p20`. If the robot reaches `p20` without reaching the `DesiredTorque` specified by the user, then the robot moves back to `p10` and tries once more.

---

#### More examples

More examples of the instruction `ContactL` are illustrated below.

##### Example 1

```
ContactL p10, v100, \Zone:=z10, tool1;
```

*Continues on next page*

The TCP of `tool1` is moved linearly towards the position `p10` at a speed of `v100` and a zone size of 10 mm.

Since the argument `DesiredTorque` is omitted, the `ContactL` instruction will only raise the collision detection level to its maximum value and not supervise the internal torque level.

#### Syntax

```
ContactL
[ '\ ' DesiredTorque ' ,' ]
[ ToPoint ' := ' ] < expression (IN) of robtarget >
[ '\ ' ID ' := ' < expression (IN) of identno > ] ' ,'
[ Speed ' := ' ] < expression (IN) of speeddata >
[ '\ ' Zone ' := ' < expression (IN) of zonedata > ] ' ,'
[ Tool ' := ' ] < persistent (PERS) of tooldata >
[ '\ ' WObj ' := ' < persistent (PERS) of wobjdata > ] ' ;'
```

#### Related information

For information about	See
Writes to a corrections entry	<code>CorrWrite</code> - Writes to a correction generator, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Moves the robot linearly	<code>MoveL</code> - Moves the robot linearly, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Definition of load	<code>loaddata</code> - Load data, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Definition of velocity	<code>speeddata</code> - Speed data, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Definition of tools	<code>tooldata</code> - Tool data, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Definition of work objects	<code>wobjdata</code> - Work object data, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Defining the payload for a robot	<code>GripLoad</code> - Defines the payload for a robot, in <i>Technical reference manual - RAPID Instructions, Functions and Data types</i>
Using error handlers	<i>Technical reference manual - RAPID overview</i>
Linear movement	
Motion in general	
<code>LoadIdentify</code> , load identification service routine	<i>Operating manual - IRC5 with FlexPendant</i>

*Continues on next page*

## 5 RAPID reference information

---

### 5.1.1 ContactL - Linear contact movement

*Continued*

For information about	See
System input signal <i>SimMode</i> for running the robot in simulated mode without payload. (Topic I/O, Type System Input, Action values, <i>SimMode</i> )	<i>Technical reference manual - System parameters</i>
System parameter <i>ModalPayloadMode</i> for activating and deactivating payload. (Topic Controller, Type System Misc, Action values, <i>ModalPayloadMode</i> )	

## 6 System parameters

### 6.1 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	<a href="#">Collision Avoidance on page 64</a>

#### Topic Motion

Parameter	For more information, see
Coll-Pred Safety Distance	<a href="#">Coll-Pred Safety Distance on page 65</a>
Global Speed Limit	<a href="#">Global Speed Limit on page 66</a>
Arm Check Point Speed Limit	<a href="#">Arm Check Point Speed Limit on page 67</a>
Arm-Angle Reference Direction	<a href="#">Arm-Angle Reference Direction on page 68</a>

## 6 System parameters

---

### 6.2.1 Collision Avoidance

## 6.2 Topic I/O System

### 6.2.1 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*.

---

#### Cfg name

CollAvoidance

---

#### Description

The action value *Collision Avoidance* shall be set when the functionality for Collision Avoidance is activated. If no value is defined, then the functionality is not activated. *Collision Avoidance* monitors a detailed geometric model of the robot. If two bodies of the model come too close to each other, the controller warns about a predicted collision and stops the robot. The system parameter *Coll-Pred Safety Distance* (*coll\_pred\_default\_safety\_distance*) determines at what distance the two objects are considered to be in collision. The default value for this parameter is 0.01 meter, but it can be set to any value between 0.001 and 1 meters.

---

#### Prerequisites

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

---

#### Limitations

This parameter is currently applicable only to IRB 14000 (YuMi robot).



## 6.3 Topic Motion

### 6.3.1 Coll-Pred Safety Distance

---

**Parent**

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

---

**Cfg name**

coll\_pred\_default\_safety\_distance

---

**Description**

The parameter *Coll-Pred Safety Distance* determines at what distance two geometric objects (for example robot-links) are considered to be in collision.

---

**Allowed values**

A value between 0.001 to 1 meter.  
Default value is 0.001 meter.

---

**Related information**

[Collision Avoidance on page 64](#)

## 6 System parameters

### 6.3.2 Global Speed Limit

### 6.3.2 Global Speed Limit

---

#### Parent

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

---

#### Cfg name

Global\_max\_speed\_limit\_custom

---

#### Description

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



#### Note

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



#### Note

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

---

#### Limitations

*Global Speed Limit* is only used for the following robots:

- IRB 14000

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

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

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000	1.5 m/s

---

#### Allowed values

A number between 0.1 and 20.

The default value is 20.

### 6.3.3 Arm Check Point Speed Limit

#### Parent

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

#### Cfg name

Global\_max\_speed\_limit\_acp\_custom

#### Description

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



#### Note

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



#### Note

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

#### Limitations

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

- IRB 14000

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

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

The maximum value for the robot types are:

Robot type	Maximum value
IRB 14000	1.0 m/s

#### Allowed values

A number between 0.1 and 20.

The default value is 20.

## 6 System parameters

### 6.3.4 Arm-Angle Reference Direction

### 6.3.4 Arm-Angle Reference Direction

---

#### Parent

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

---

#### Cfg name

arm\_angle\_ref\_dir

---

#### Description

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

---

#### Usage

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

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

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



#### Note

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

---

#### Allowed values

*Arm-Angle Reference Direction* can have the following values:

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

The default value is 0.

---

#### Related information

*Operating manual - IRB 14000*

# Index

## A

arm mode, 37  
axes, 33

## B

base coordinate system, 34

## C

calibration  
  when to do a fine calibration, 47  
  when to update the revolution counters, 47  
calibration position  
  jogging to, 55  
Cartesian speed supervision, 17  
cleaning  
  FlexPendant, 23  
ContactL, 57  
coordinate systems, 33

## D

danger levels, 13

## E

emergency stop  
  buttons, 15  
  definition, 15  
emergency stop button  
  FlexPendant, 18  
emergency stops  
  recovering, 19  
enabling device, 24

## F

fieldbus, 44  
FlexPendant, 28  
  cleaning, 23  
  emergency stop button, 18  
  jogging to calibration position, 55  
  MoveAbsJ instruction, 55

## I

I/O signals, 44  
industrial networks, 44

## J

jogging window, 35  
joystick directions, 33

## M

MoveAbsJ instruction, 55

## O

operator panel, 40

## P

product classes, 29  
protection standards, 12  
protective equipment, 22  
protective stop, 16  
protective wear, 22

## Q

Quickset menu, 36

## R

RobotStudio  
  overview, 30  
RobotStudio Oline Apps, 31  
  Calibrate, 31  
  Jog, 32  
  Manage, 31  
  Operate, 32  
  Tune, 32  
  YuMi, 32  
RobotWare, 29

## S

safety  
  emergency stop, 15  
  signals, 13  
  signals in manual, 13  
  symbols, 13  
safety signals  
  in manual, 13  
safety standards, 12  
safety stop, 16  
signals  
  safety, 13  
standards  
  ANSI, 12  
  CAN, 12  
  EN, 12  
  EN IEC, 12  
  EN ISO, 12  
  safety, 12  
states  
  emergency stop, 15  
symbols  
  safety, 13

## U

UAS, user authorization system, 45





# Contact us

## **ABB AB**

**Discrete Automation and Motion  
Robotics**

S-721 68 VÄSTERÅS, Sweden

Telephone +46 (0) 21 344 400

## **ABB AS, Robotics**

**Discrete Automation and Motion**

Nordlysvegen 7, N-4340 BRYNE, Norway

Box 265, N-4349 BRYNE, Norway

Telephone: +47 51489000

## **ABB Engineering (Shanghai) Ltd.**

5 Lane 369, ChuangYe Road

KangQiao Town, PuDong District

SHANGHAI 201319, China

Telephone: +86 21 6105 6666

[www.abb.com/robotics](http://www.abb.com/robotics)